HAL open science

Pratique de l' Analyse Discriminante Linéaire

Ricco Rakotomalala

To cite this version:

Ricco Rakotomalala. Pratique de l' Analyse Discriminante Linéaire. 2020. ⟨hal-04868585⟩

HAL Id: hal-04868585

https://hal.science/hal-04868585v1

Submitted on 6 Jan 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons CC BY-NC-ND 4.0 - Attribution - Non-commercial use - No Derivative Works - International License

Ricco Rakotomalala

Pratique de l' Analyse Discriminante Linéaire

Version 1.0

Université Lumière Lyon 2

Avant-propos

L'analyse discriminante me paraissait très mystérieuse durant mes premières années d'études. Il faut dire qu'elle n'était pas enseignée dans les formations en économétrie que j'avais suivies. Il fallait que je me fasse une idée à partir de la documentation dont j'avais l'accès dans les bibliothèques (on n'avait pas internet à l'époque). Et, dans mes lectures, on la présentait avant tout comme une méthode descriptive… que l'on pouvait utiliser à des fins de classement. Le mode de fonctionnement de la fonction lda() du package MASS de R est d'ailleurs très révélateur à cet égard. Les ouvrages ne mettaient pas en avant, comme on pouvait le voir en régression linéaire multiple qui était ma principale référence en matière de prédiction, un modèle explicite, lisible, avec des indications sur la pertinence globale et le degré d'influence des variables. Forcément, l'étudiant que j'étais était dérouté. J'en étais arrivé à l'idée que seule la régression logistique était réellement intéressante lorsque nous sommes dans un schéma prédictif où l'on souhaite prédire et expliquer les valeurs prises par une variable cible qualitative à partir d'un ensemble de variables explicatives, quantitatives ou qualitatives.

La situation a changé au mitan des années 90. Parce que j'ai eu accès à des ouvrages en langue anglaise qui présentaient les choses différemment, à plus de documentation grâce aux débuts d'internet. Parce que si je connaissais assez bien la statistique classique et l'économétrie, j'avais envie d'investir la statistique exploratoire, que l'on appelait aussi « analyse de données », domaine dans lequel j'avais conscience d'avoir des lacunes. Et surtout parce que j'ai pu manipuler des outils – STATISTICA notamment qui avait fait forte impression sur moi à l'époque, je m'en suis inspiré pour désigner SIPINA V3 d'ailleurs – qui présentaient de manière compréhensible pour moi les résultats. On apprend mieux en pratiquant, je l'ai toujours pensé. En scrutant ses sorties, j'ai retrouvé ce que j'attendais : les coefficients estimés d'une équation linéaire qui permet d'affecter simplement les classes aux individus supplémentaires, une évaluation globale du modèle accompagnée d'un test de significativité, une évaluation de la pertinence individuelle des variables, des mécanismes de sélection de variables que je reconnaissais. Aujourd'hui, je me

rends compte que ces éléments étaient en réalité présents dans certaines références que j'avais pourtant épluchées. Dans Diday et al. (1982) par exemple, un ouvrage de très grande qualité que je lis et relis avec beaucoup d'intérêt toujours, je vois maintenant les équations des fonctions de classement explicites telles que je les attendais (équation 49, page 328). Mais faute d'une écriture plus accessible peut être, d'exemples didactiques qui permettent d'appréhender simplement les notions essentielles, faute de background et de recul suffisant de ma part aussi, je le reconnais, je n'étais pas en état de les identifier.

J'enseigne l'analyse discriminante linéaire (ADL) depuis plus de 20 ans maintenant. Je la privilégie dans mes cours introductifs à l'analyse prédictive (à la data science, au data mining, au machine learning, etc.) parce que je peux présenter la méthode dans un temps raisonnable, simplement, schématiquement, à différents types de publics, y compris à des non-statisticiens parfois/souvent réfractaires aux mathématiques. Mon principal enjeu est de mettre le focus sur la finalité, l'estimation des paramètres (coefficients) d'un modèle explicite qui s'exprime sous la forme d'une combinaison linéaire des variables explicatives. Après, selon les formations dans lesquelles j'interviens, je fais des digressions sur les différents aspects de la méthode : probabilistes, statistiques, géométriques, le calcul matriciel, etc. Et ça marche plutôt bien parce que les notions fondamentales de la méthode, les barycentres conditionnels, les formes des nuages de points exprimées par les matrices de variance covariance, peuvent être présentés à l'aide de graphiques simples au tableau dont je fais beaucoup usage spécialement pour cette méthode. Par la suite, dans les configurations où cela est possible, les étudiants sont tout à fait capables de mettre en œuvre la technique sur un jeu de données et d'apprécier la teneur des résultats immédiatement après mon exposé. C'est un signe qui ne trompe pas.

Dans cet ouvrage, « **Pratique de l'Analyse Discriminante Linéaire** », contrairement à la présentation usuelle que l'on en fait, je mettrai l'accent avant tout sur les **aspects prédictifs** avec pour finalité la production d'un modèle explicite. La trame adoptée sera assez proche du support que j'avais consacré à la régression logistique (« Pratique de la Régression Logistique », 2011). Je m'appuierai sur le cadre probabiliste pour mettre en avant la construction et l'estimation du modèle. Nous passerons en revue par la suite les aspects pratiques liés au processus de modélisation : l'évaluation et l'interprétation des résultats, les aménagements à mettre en place

pour prendre en compte les cas particuliers comme les descripteurs qualitatifs, l'appréhension des problèmes liés à la colinéarité et/ou à la dimensionnalité, les relations avec les autres algorithmes d'apprentissage automatique (machine learning), la jonction avec le prisme descriptif. Fidèle à mes habitudes, le texte et les formules seront constamment accompagnés d'exemples illustratifs. Mais plutôt que d'utiliser le tableur Excel, je vais m'appuyer sur les commandes et sorties du logiciel R pour détailler les calculs, en m'en tenant exclusivement aux fonctions des librairies historiques qui font référence (base, stats). Parce qu'aujourd'hui, en 2020, le niveau en informatique des étudiants a considérablement évolué. Ils n'ont aucune peine à suivre une présentation étayée par du code informatique, pourvu que l'on prenne la peine d'expliciter les étapes, et que l'on évite d'utiliser des packages qui masquent la subtilité des opérations. On va essayer d'éviter cela. Quand cela est nécessaire, je mettrai en référence les sorties d'autres outils, principalement SAS avec les procédures DISCRIM, STEPDISC et CANDISC, TANAGRA avec les outils « Linear Discriminant Analysis » et « STEPDISC ».

Enfin, je le dis souvent, un document ne sort jamais du néant. Je me suis beaucoup servi de nombreuses références citées en bibliographie. Sans chauvinisme excessif, j'ai décidé de faire la part belle aux ouvrages en langue française, avec trois exceptions quand-même : « Applied Manova and Discriminant Analysis », de Huberty et Olejnik (2006), parce qu'il s'agit – à ma connaissance – d'un des très rares ouvrages exclusivement consacrés à l'analyse discriminante, avec à peu près les mêmes contours que j'essaie moi-même de dessiner dans ce document ; « The Elements of Statistical Learning » de Hastie et al. (12th printing, 2017), parce qu'il est incontournable dans tout ce qui a trait à l'apprentissage statistique ; la documentation de SAS PROC DISCRIM enfin, elle nous permettra de retracer les formules utilisées dans les logiciels.

Voilà. Plus de 20 ans d'enseignement de cette technique disais-je. Il est temps de partager cette expérience avec le plus grand nombre. J'espère que ce support, débuté dans des circonstances assez particulières, permettra aux fans de data science, aux étudiants en quête de connaissances en particulier, de progresser dans l'apprentissage des méthodes de l'exploration statistique des données.

Lyon, le 16 avril 2020.

Notations et données

L'objectif est de prédire les valeurs prises par une variable cible (expliquée, dépendante) qualitative nominale Y à **K** modalités $\{y_1, y_2, \ldots, y_k, \ldots, y_K\}$.

Dans un échantillon Ω comportant n observations, la valeur prise par l'individu ω sera notée $Y(\omega)$.

Pour la modélisation, nous disposons de p variables explicatives (indépendantes, descripteurs, prédicteurs), notées X_j (j = 1, …, p). Pour des raisons de concision, nous regroupons ces variables sous la lettre X, où $X = (X_1, X_2, \ldots, X_j, \ldots, X_p)$. Un individu ω est décrit par le vecteur $X(\omega)$.

Pour alléger l'écriture, nous omettons le symbole ω lorsque cela est possible.

Dans la population, la probabilité a priori d'une classe y_k est notée $P(Y = y_k) = \pi_k$

Dans notre échantillon de taille n, nous notons n_k le nombre d'individus appartenant à la classe y_k. L'estimation de la prévalence de la classe est la fréquence relative observée : $\hat{\pi}_k = \frac{n_k}{n}$

La probabilité d'appartenance à une classe sachant la description, la probabilité a posteriori, est la quantité que l'on cherche à modéliser en apprentissage supervisé : $P(Y=y_k / X)$

Autant que possible, nous utiliserons les jeux de données suivants tout au long de cet ouvrage pour montrer la progression des analyses.

Données Artificielles (DATA 1)

Le premier jeu de données est une variante des données artificielles extraites de Tomassone et al. (1988 ; Tableau 1, page 29). Je l'ai un peu modifié pour que les barycentres conditionnels ne coïncident pas avec les observations, et que les classes ne soient pas exactement équilibrées afin d'élargir la portée des résultats. Il est situé dans la feuille « DATA_1 » du fichier « **Data_Illustration_Livre_ADL.xlsx** » qui accompagne ce document.

Nous chargeons les données et nous préparons les structures pour les calculs à venir.

```
#** importation des données **

#lecture
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_1")

#inspection
str(D)

## 'data.frame':    11 obs. of  3 variables:
##  $ Groupe: Factor w/ 2 levels "g1","g2": 1 1 1 1 1 2 2 2 2 2 ...
##  $ X1    : num  0 2 4 6 8 5 7 9 11 13 ...
##  $ X2    : num  3 1 6 9 7 2 0 4 8 6 ...

#affichage
print(D)

##    Groupe X1 X2
## 1      g1  0  3
## 2      g1  2  1
## 3      g1  4  6
## 4      g1  6  9
## 5      g1  8  7
## 6      g2  5  2
## 7      g2  7  0
## 8      g2  9  4
## 9      g2 11  8
## 10     g2 13  6
## 11     g2  4  0

#** préparation des structures **

#matrice des descripteurs X
X <- D[c("X1","X2")]

#vecteur cible y
y <- D$Groupe

#nombre d'observations
n <- nrow(D)
print(n)

## [1] 11

#nombre de classes
K <- nlevels(y)
print(K)

## [1] 2

#nombre d'explicatives
p <- ncol(X)
print(p)

## [1] 2
```

La variable cible « Groupe » possède deux modalités {g1, g2}.

Puisque nous avons deux explicatives, nous pouvons représenter les données dans le plan.

```
#position relatives des classes dans le plan
plot(X$X1,X$X2,col=c("blue","green")[y], pch = 19, main="Données DATA 1")
```

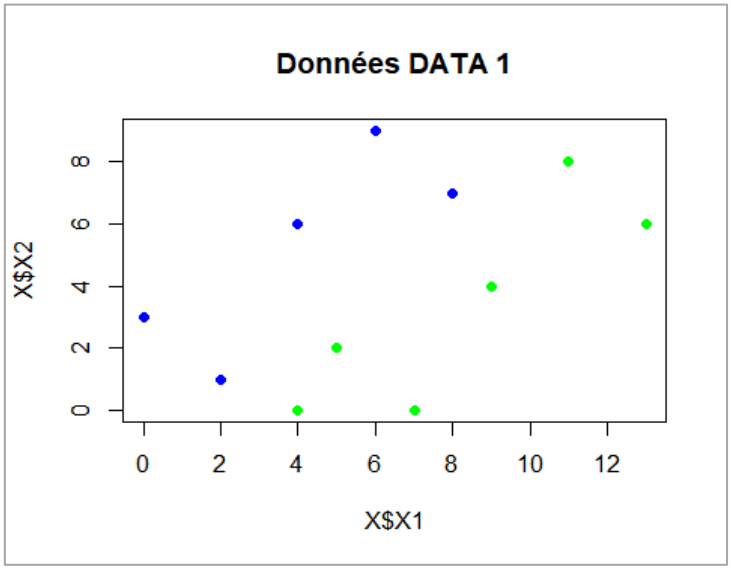


Figure 1 – Position relatives des classes dans le plan – Données "DATA 1"

Le problème à résoudre semble très simple. Pas besoin d'être grand clerc pour deviner que les classes seront linéairement séparables. Nous verrons si l'analyse discriminante linéaire (ADL) est capable d'identifier une bonne solution. Autre élément qui semble évident également, c'est la combinaison des variables (X1, X2) qui permet de discriminer les classes. Chacune prise individuellement n'en est pas capable. Nous verrons également si l'ADL saura restituer cette information.

Données Eaux de Vie (DATA 2)

Le second jeu de données « Eaux de vie » a été utilisé dans deux tutoriels « Analyse discriminante sous Python » (avril 2020) et « Analyse discriminante sous R » (avril 2020) où nous explorions la pratique de l'ADL sous ces deux outils phares de la data science.

Nous cherchons à identifier le TYPE d'alcools, K = 3 classes {KIRSCH, MIRAB, POIRE}, à partir de leur composition (méthanol, acétone, butanol, etc. ; p = 8 variables). Les données ont été scindées en échantillons d'apprentissage (DATA_2_TRAIN, n = 52 observations) et de test (DATA_2_TEST, 50 observations).

Echantillon d'apprentissage

Nous chargeons les données et nous montrons leurs caractéristiques.

```
#** importation des données **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")

#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#affichage des premières lignes
print(head(DTrain))

##     TYPE  MEOH  ACET BU1  BU2 ISOP MEPR   PRO1 ACAL
## 1 KIRSCH 336.0 225.0 1.0  1.0   92 37.0  177.0  0.0
## 2 KIRSCH 442.0 338.0 1.9 10.0   91 30.0  552.0 31.0
## 3 KIRSCH 373.0 356.0 0.0 29.0   83 27.0  814.0 11.0
## 4 KIRSCH 418.0  62.0 0.8  0.0   89 24.0  342.0  7.0
## 5 KIRSCH  84.0  65.0 2.0  2.0    2  0.0  288.0  6.0
## 6 KIRSCH 632.5 342.3 2.2 52.1  123 38.2 2070.1 13.3

#** préparation des structures **

#matrice des descripteurs X
XTrain <- DTrain[-1]

#vecteur cible y
yTrain <- DTrain$TYPE

#nombre d'observations
n <- nrow(DTrain)
print(n)

## [1] 52

#nombre de classes
K <- nlevels(yTrain)
print(K)

## [1] 3

#nombre d'explicatives
p <- ncol(XTrain)
print(p)
```

```
## [1] 8

#position relatives des classes dans les repères
#définis par les variables prises deux à deux
pairs(XTrain,col=c("blue","green","red")[yTrain], main="Données DATA 2 - TRAIN", cex = 0.6)
```

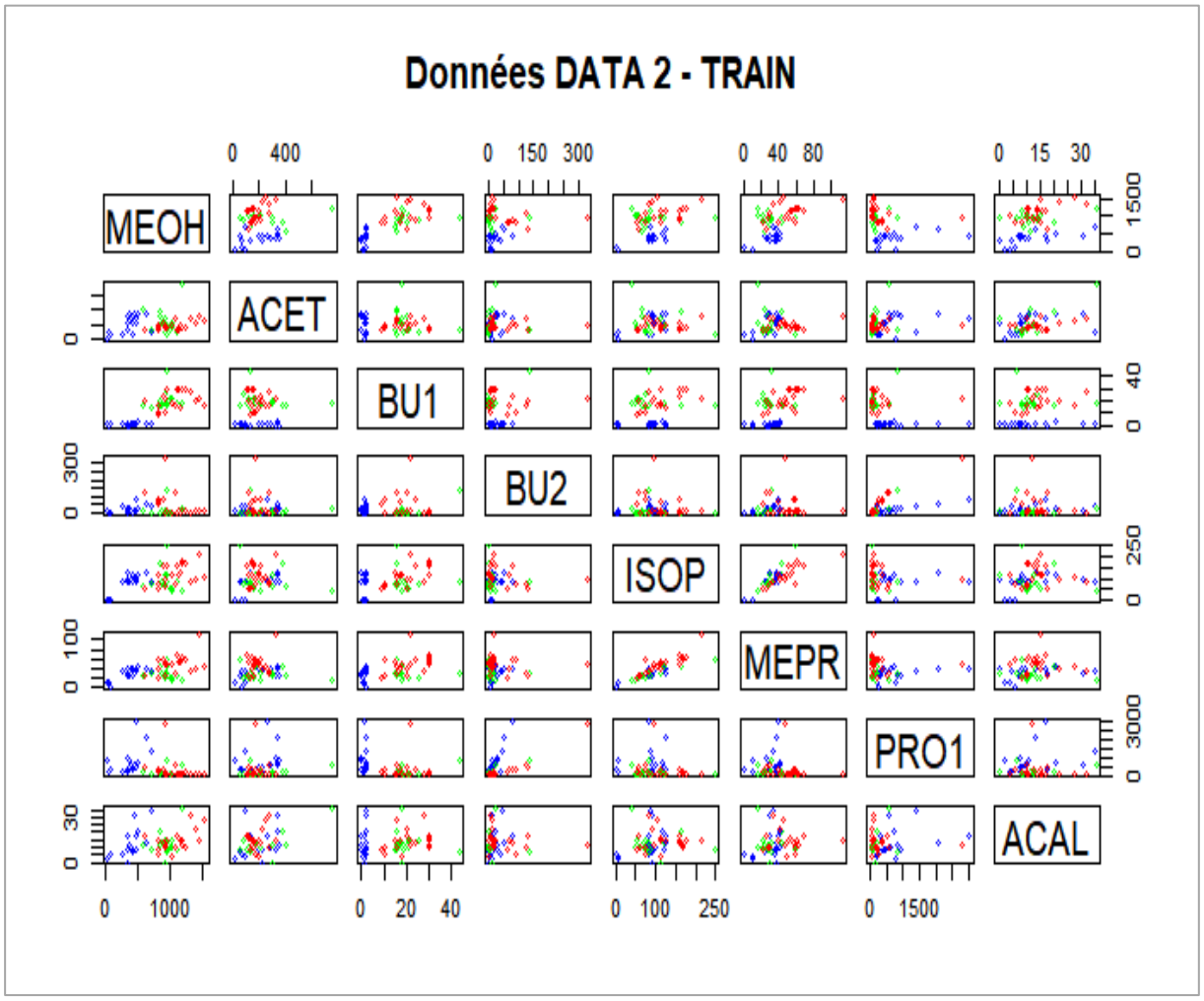


Figure 2 – Inspection visuelle rapide des données – DATA 2 / TRAIN

Ce genre de graphique est toujours instructif quand le nombre de variables n'est pas trop élevé. Nous notons qu'il y a des valeurs atypiques (BU2) ; des variables très corrélées (ISOP, MEPR) ; que l'on distingue plutôt bien les classes dans certains repères, (MEOH, BU1) ou (MEOH, MEPR) par exemple.

Voyons plus en détail pour (MEOH, MEPR).

```
plot(XTrain$MEOH,XTrain$MEPR,col=c("blue","green","red")[yTrain],main="(MEOH, MEPR)")
legend(0,100,c('KIRSCH','MIRAB','POIRE'),text.col=c("blue","green","red"))
```

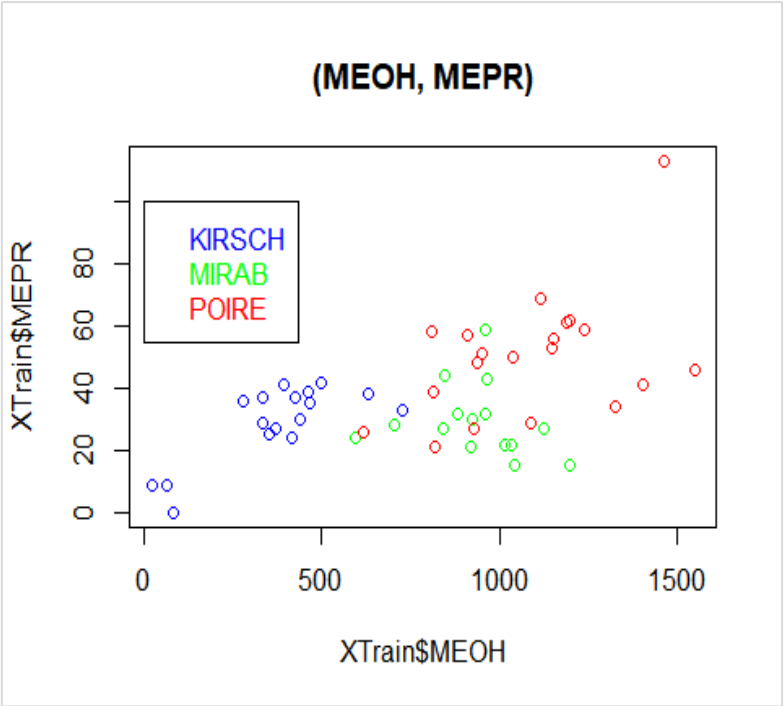


Figure 3 – Positions des classes dans le repère (MEOH, MEPR) – DATA 2 / TRAIN

La modalité KIRSCH est à part, elle pourra être reconnue sans difficultés. Il sera plus difficile de discriminer MIRAB et POIRE, en tous les cas avec uniquement ces deux variables.

Echantillon test

Nous chargeons l'échantillon test qui présente la même structure, mais avec des observations différentes (cf. les 6 premières lignes en apprentissage et en test).

```
DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")

#inspection
str(DTest)

## 'data.frame':    50 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  3 475 186 371 583 0 421 557 167 523 ...
##  $ ACET: num  15 172 101 414 226 25 142 447 86 367 ...
##  $ BU1 : num  0.2 1.9 0 1.2 2.3 0.1 1.6 0 0 2.6 ...
##  $ BU2 : num  30 7 1.6 0 19 8 8 34 0 30 ...
##  $ ISOP: num  9 113 36 97 120 0 75 107 32 116 ...
##  $ MEPR: num  9 33 11 39 46 6 24 39 10 45 ...
##  $ PRO1: num  350 546 128 502 656 253 128 162 114 787 ...
##  $ ACAL: num  9 14 8 9 11 7 31 94 8 25 ...

#affichage des premières lignes
print(head(DTest))

##     TYPE MEOH ACET BU1  BU2 ISOP MEPR PRO1 ACAL
## 1 KIRSCH    3   15 0.2 30.0    9    9  350    9
```

```
## 2 KIRSCH  475  172 1.9  7.0  113   33  546   14
## 3 KIRSCH  186  101 0.0  1.6   36   11  128    8
## 4 KIRSCH  371  414 1.2  0.0   97   39  502    9
## 5 KIRSCH  583  226 2.3 19.0  120   46  656   11
## 6 KIRSCH    0   25 0.1  8.0    0    6  253    7

#matrice des descripteurs X
XTest <- DTest[-1]

#vecteur cible y
yTest <- DTest$TYPE
```

Table des matières

1 Analyse discriminante linéaire – Principe 17
1.1 Apprentissage supervisé 17
1.2 Un cadre probabiliste pour le classement 18
1.3 Fonction de classement linéaire 26
1.4 Prédiction avec les fonctions de classement 33
1.5 Redressement pour les données non représentatives 35
1.6 Calcul des probabilités d'affectation 37
1.7 Interprétation – Lecture des coefficients 39
1.8 Interprétation – Distance aux centres de classes 40
1.9 Interprétation – Frontière(s) de décision 43
1.10 Distance à la frontière séparatrice 48
1.11 Avantages et inconvénients de l'analyse discriminante linéaire 50
2 Evaluation statistique 55
2.1 Distance entre centres de classes 56
2.2 Evaluation globale – Test MANOVA 58
2.3 Pertinence d'une variable 64
2.4 Pertinence d'un groupe de variables 68
3 Traitement des descripteurs qualitatifs 73
3.1 Données « Votes au congrès » 73
3.2 Codage 0/1 des descripteurs 76
3.3 La méthode DISQUAL 80
3.4 Cas du mix de descripteurs quantitatifs et qualitatifs 85
4 Sélection de variables 87
4.1 Approche ascendante forward 89
4.2 Approche descendante backward 95
4.3 Sélection pour les descripteurs qualitatifs 102
5 Particularité du problème à (K = 2) classes 109

5.1 Distance entre les centres de classes 109
5.2 Fonction score 110
5.3 Equivalence avec la régression linéaire 115
5.4 Combinaison de classifieurs binaires pour (K > 2) 128
6 Naïve Bayes – Modèle d'Indépendance conditionnelle 135
6.1 Modèle bayésien naïf 136
6.2 Lien avec l'analyse discriminante 145
6.3 Cas des prédicteurs qualitatifs 147
6.4 Analyse discriminante barycentrique 151
7 Régularisation 153
7.1 Shrinkage 153
7.2 Analyse discriminante sur facteurs de l'ACP 154
7.3 Analyse discriminante PLS 164
8 Analyse factorielle discriminante 177
8.1 Principe de l'analyse factorielle discriminante 178
8.2 Affectation des classes dans le repère factoriel 212
8.3 Déduction des fonctions de classement explicites 217
9 Analyse des correspondances discriminante 223
9.1 Principe de la méthode 224
9.2 Réaliser une ACD via une AFC 231
9.3 Pratique de l'analyse des correspondances discriminante 233
9.4 Affectation des classes 245
10 Outils logiciels 251
10.1 Proc DISCRIM et STEPDISC de SAS 251
10.2 ADL sous R – Le package « discriminR » 256
10.3 TANAGRA (LDA et STEPDISC) 271
10.4 ADL sous Python – Le package « scikit-learn » 273
11 Références 277
11.1 Références en français 277
11.2 Références en anglais 278
11.3 Tutoriels 278
12 Annexes 281
12.1 Gestion des versions 281
12.2 Fichier de données 281

1 Analyse discriminante linéaire – Principe

Dans ce chapitre, nous adoptons sciemment une démarche purement probabiliste pour présenter l'analyse discriminante, avec le cadre bayésien de l'apprentissage supervisé et l'utilisation des hypothèses de distributions paramétriques. D'autres approches étaient possibles, géométriques notamment avec l'analyse discriminante de Fisher (Saporta, 2006 ; section 18.2). Il fallait choisir un prisme et s'en tenir à une certaine cohérence assumée, notamment par rapport à mes autres supports, comme celui consacré à la régression logistique (Rakotomalala, 2011). Mais il est important pour le lecteur de savoir qu'il est possible d'appréhender la méthode autrement.

1.1 Apprentissage supervisé

L'objectif de l'apprentissage supervisé est de prédire ou expliquer les valeurs prises par une variable cible catégorielle Y à partir d'un ensemble de p descripteurs, quantitatifs ou qualitatifs. On cherche à établir l'existence d'une liaison fonctionnelle sous-jacente, on parle de « modèle », dont on estimera les paramètres à partir des données.

Les classifieurs linéaires font partie des techniques les plus populaires. La liaison s'exprime sous la forme d'une ou plusieurs combinaisons linéaires des prédicteurs

$$a_0 + a_1 x_1 + \cdots + a_p x_p$$

a = (a_0, a_1, …, a_p) est le vecteur de paramètres, coefficients, dont on doit estimer les valeurs à partir des données durant un processus dit d'apprentissage. Il est supervisé parce qu'il est guidé par Y dont on souhaite approximer au mieux les valeurs.

Les domaines d'applications sont nombreux. Il sont résumés dans mon précédent ouvrage consacré à la régression logistique (Rakotomalala, 2011 ; section 1.1.1).

1.2 Un cadre probabiliste pour le classement

L'approche probabiliste de l'apprentissage supervisé repose sur l'estimation de la probabilité conditionnelle, dite probabilité a posteriori, de y_k sachant la valeur prise par X :

$$P(Y = y_k/X)$$

Pour un individu ω à classer, on minimise l'erreur d'affectation en utilisant la règle suivante :

$$y_{k*} = \arg\max_k P[Y(\omega) = y_k/X(\omega)]$$

C.-à-d. on affecte l'individu à la classe qui maximise sa probabilité d'appartenance.

On parle de règle d'affectation bayésienne ou de critère de décision de Bayes (Bardos, 2001 ; section IV ; Tenenhaus, 2007, section 4). Dans notre cas, seule l'erreur de classement est prise en compte, nous passons outre les éventuelles matrices de coûts de mauvais classements non unitaires.

Avec le théorème de Bayes, nous écrivons facilement (Lebart et al., 2000 ; section 3.3.5.a) :

$$P(Y = y_k \,/\, X) = \frac{P(Y = y_k) \times P(X \,/Y = y_k)}{P(X)}$$
$$= \frac{P(Y = y_k) \times P(X \,/Y = \; y_k)}{\sum_{c=1}^{K} P(Y = y_c) \times P(X \,/Y = \; y_c)}$$

- Comme il s'agit de maximiser cette quantité par rapport à y_k, nous pouvons évacuer le dénominateur puisque qu'il s'agit d'un terme de normalisation qui sera toujours le même quel que soit la modalité (y_k) considérée.
- $P(Y = y_k)$ est soit fournie, si nous avons l'information sur les prévalences des classes dans la population (disposition prévue par certains logiciels), soit estimée simplement à partir des données avec la proportion $\frac{n_k}{n}$

L'enjeu réside donc dans l'estimation de la probabilité conditionnelle P(X / Y = y_k) à partir des données. Pour rendre ce calcul possible, nous allons émettre des hypothèses statistiques simplificatrices.

1.2.1 Hypothèse 1 : Distribution multinormale

1.2.1.1 Distribution conditionnelle de X sachant Y

La première hypothèse porte sur la distribution de P(X/Y = y_k). Nous supposons que X suit une loi normale multidimensionnelle (loi multinormale) conditionnellement aux valeurs de Y. Elle est paramétrée par l'espérance μ_k (paramètre de position) et la matrice de variance covariance Σ_k (paramètre de dispersion)

Sa fonction de densité s'écrit :

$$f_k(X) = \frac{1}{(2\pi)^{\frac{p}{2}}[\det(\Sigma_k)]^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_k)\Sigma_k{}^{-1}(x-\mu_k)^T}$$

En passant au logarithme et el supprimant tout ce qui ne dépend pas de y_k, nous obtenons la fonction d'affectation suivante :

$$Q(y_k, X) = \ln \hat{\pi}_k - \frac{1}{2}\ln\det(V_k) - \frac{1}{2}(x-\bar{x}_k)\mathrm{V}_k{}^{-1}(x-\bar{x}_k)^T$$

Où

- $\hat{\pi}_k$ est la proportion observée de la classe y_k ;
- $\bar{x}_k$ est le vecteur moyenne pour les individus correspondant à la classe y_k ;
- V_k est la matrice de variance covariance estimée pour y_k
- det(V_k) est le déterminant de la matrice V_k

Ces paramètres sont très faciles à calculer en pratique.

La règle d'affectation aux classes devient alors :

$$y_{k*} = \arg\max_k Q(y_k, X(\omega))$$

Pour l'individu ω, on lui affecte la classe qui maximise le « score » $Q(y_k, X(\omega))$

Exemple DATA 1

Calcul des paramètres. Essayons de reproduire les calculs pour DATA 1 (Figure 1). Nous calculons tout d'abord les fréquences et moyennes conditionnelles. Nous plaçons ces dernières (▲) dans notre espace de représentation.

```
#calcul des fréquences relatives des classes
pi_k <- table(y)/n
print(pi_k)

## y
##        g1        g2
## 0.4545455 0.5454545

#moyennes conditionnelles selon le groupe d'appartenance
xb_k <- aggregate(X,list(y),FUN=mean)
print(xb_k)

##   Group.1       X1       X2
## 1      g1 4.000000 5.200000
## 2      g2 8.166667 3.333333

#positionner les moyennes conditionnelles dans le plan
plot(X$X1,X$X2,col=c("skyblue3","green")[y], pch = 19, main="Données DATA 1")
points(xb_k$X1,xb_k$X2,col=c("blue","green4"),pch=17,cex=1.5)
```

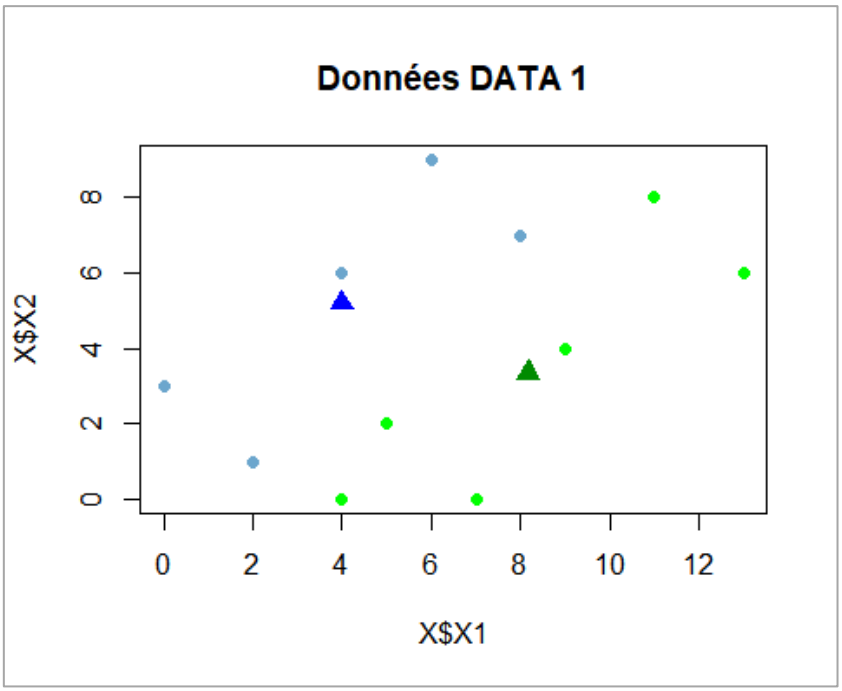


Figure 4 – Données DATA 1 avec les moyennes conditionnelles (▲)

Les barycentres conditionnels sont bien situés « au milieu » des observations correspondantes. Cela est vrai parce que nos nuages de points conditionnels sont unimodaux, dispersés autour de leurs moyennes respectives justement. L'hypothèse de distribution conditionnelle multinormale paraît viable.

Nous passons au calcul des matrices de covariances conditionnelles.

```
#matrice de variance covariance par groupe
V_k <- by(as.matrix(X),list(y),cov)
print(V_k)

## : g1
##    X1   X2
## X1 10  8.0
## X2  8 10.2
## ------------------------------------------------------------
## : g2
##           X1        X2
## X1 12.166667  9.733333
## X2  9.733333 10.666667
```

Pour rendre la chose plus concrète, nous matérialisons ces matrices – qui définissent la dispersion – sous la forme d'ellipses centrées autour des moyennes conditionnelles. Nous utilisons la fameuse librairie « ggplot » pour aller à l'essentiel. En cherchant un peu sur le web, nous trouvons facilement des tutoriels qui expliquent la séquence de commandes ci-dessous.

```
#libraire ggplot
library(ggplot2)

#graphique avec ellispes
gel <- ggplot(D,aes(X1,X2,color=y)) + geom_point() + scale_color_manual(values=c("skyblue3","green"))
gel + stat_ellipse(type="norm",level=0.7)
```

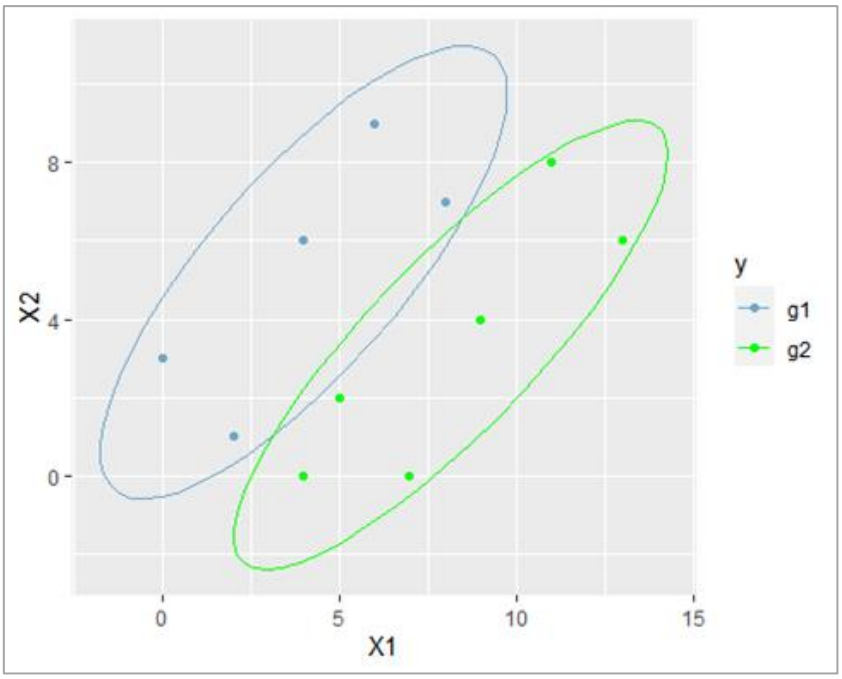


Figure 5 – Données DATA 1 – Ellipses de dispersion

Plus encore que précédemment, l'hypothèse de multinormalité est crédible.

Individu supplémentaire à classer. Nous disposons de tous les éléments pour calculer le score d'appartenance d'un individu à classe (y_k) avec $Q(y_k,X)$. Mettons que nous souhaitons positionner l'individu ω de coordonnées ($X_1 = 6$, $X_2 = 7$). Plaçons-le déjà dans le plan pour voir comment il se situe par rapport aux autres observations.

```
#individu supplémentaire à classer
omega <- c(6,7)
names(omega) <- c("X1","X2")
print(omega)

## X1 X2
##  6  7

#position dans le plan
#positionner les moyennes dans le plan
plot(X$X1,X$X2,col=c("skyblue3","green")[y], pch = 19, main="Données DATA 1")
points(xb_k$X1,xb_k$X2,col=c("blue","green4"),pch=17,cex=1.5)
points(omega['X1'],omega['X2'],col="red",pch=15,cex=1.5)
```

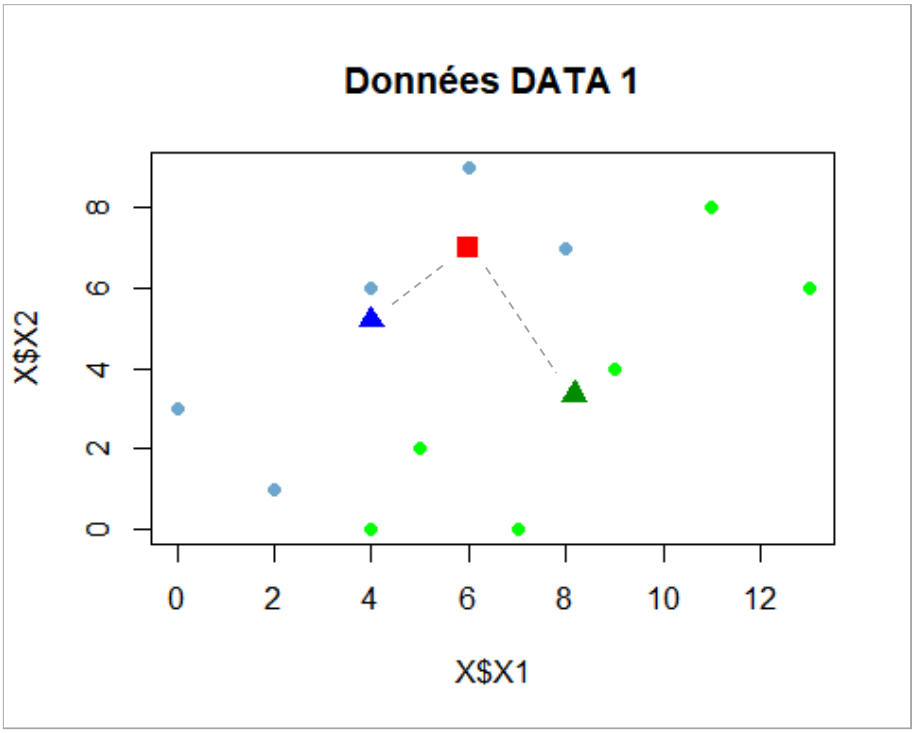


Figure 6 – Données DATA 1 – Position de l'individu supplémentaire à classer

A priori, l'individu supplémentaire (■) appartient plutôt au groupe « g1 », d'autant plus qu'il semble plus proche du barycentre conditionnel correspondant (▲) (Figure 6). Ce n'est pas qu'une vue de l'esprit. En effet, si nous considérons la dernière partie de la fonction $Q(y_k,X)$,

$$(x-\bar{x}_k)V_k{}^{-1}(x-\bar{x}_k)^T$$

Désigne une distance aux moyennes conditionnelles pondérée par l'inverse de la matrice de variance covariance c.-à-d. (le carré d') une distance de Mahalanobis. Elle est positive, mais précédée par le facteur $-\frac{1}{2}$ dans $Q(y_k,X)$. De fait, l'individu a tendance à être associé à la classe dont le barycentre est le plus proche. Ce qui paraît assez intuitif finalement. Vérifions tout cela par le calcul maintenant. Calculons le score de l'individu (■) en appliquant la fonction $Q(y_k,X)$.

```
#transformation en matrice
omega <- matrix(omega,nrow=1,ncol=2)

#matrice des moyennes
mb_k <- as.matrix(xb_k[c('X1','X2')])

#calcul du score pour le groupe 1
score_g1 <- log(pi_k[1])-0.5*log(det(V_k$g1))-0.5*(omega-mb_k[1,])%*%solve(V_k$g1)%*%t(omega-mb_k[1,])
print(score_g1)

##           [,1]
## [1,] -2.812514
```

```r
#calcul du score pour le groupe 2
score_g2 <- log(pi_k[2])-0.5*log(det(V_k$g2))-0.5*(omega-mb_k[2,])%*%solve(V_k$g2)%*%t(omega-mb_k[2,])
print(score_g2)

##           [,1]
## [1,] -7.639803
```

La configuration qui maximise le score pour l'individu ω est Q(g1,X(ω)) = -2.81. Le calcul rejoint l'intuition, le modèle l'affecte au groupe « g1 ».

1.2.1.2 Analyse discriminante quadratique

On parle d'analyse discriminante quadratique avec cette première hypothèse c.-à-d. nous avons des termes au carré des variables dans les fonctions de classement. Vérifions cela sur notre exemple à (p = 2) dimensions.

Les deux premiers termes de Q() sont des scalaires qui ne dépendent pas de X. Tout se joue dans la formule de distance. Sans limiter la portée de la démarche, nous retirons l'indice « k » pour simplifier les écritures, et nous notons v_{ij}^{-1} les termes de l'inverse de la matrice de covariance (V^{-1}). Notons z_j la variable centrée ($z_j = x_j - \bar{x}_j$), la distance de Mahalanobis s'écrit :

$$
\begin{aligned}
(z_1, z_2)\begin{pmatrix} v_{11}^{-1} & v_{12}^{-1} \\ v_{21}^{-1} & v_{22}^{-1} \end{pmatrix}\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} &= (z_1 v_{11}^{-1} + z_2 v_{21}^{-1}, z_1 v_{12}^{-1} + z_2 v_{22}^{-1})\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \\
&= [z_1(z_1 v_{11}^{-1} + z_2 v_{21}^{-1}) + z_2(z_1 v_{12}^{-1} + z_2 v_{22}^{-1})] \\
&= v_{11}^{-1} z_1^2 + (v_{12}^{-1} + v_{21}^{-1}) z_1 z_2 + v_{22}^{-1} z_2^2
\end{aligned}
$$

Ainsi, notre fonction d'affectation quadratique Q(y,X) s'écrit, toujours en omettant l'indice « k » pour simplifier les écritures :

$$
Q(y, X) = \ln \hat{\pi} - \frac{1}{2}\ln \det(V) - \frac{1}{2}[v_{11}^{-1} z_1^2 + (v_{12}^{-1} + v_{21}^{-1}) z_1 z_2 + v_{22}^{-1} z_2^2]
$$

Avec (z_1^2, $z_1 z_2$, z_2^2), le classifieur dessine une frontière quadratique dans l'espace de représentation pour discriminer les classes. L'appellation se justifie. On se rend compte que le nombre de paramètres (de coefficients de l'équation) à calculer devient vite important à mesure que le nombre de descripteurs augmente.

1.2.2 Hypothèse 2 : Homoscédasticité – Analyse discriminante linéaire

Une seconde hypothèse est introduite pour simplifier la fonction de classement de manière à ce qu'elle ne comporte que des termes linéaires.

Dilemme biais-variance : Les étudiants s'interrogent souvent sur l'intérêt de s'appuyer sur un système de représentation moins puissant. On peut s'attendre à ce que le classifieur soit moins performant. C'est vrai et ce n'est pas vrai. En simplifiant, nous risquons d'augmenter la composante de l'erreur due au biais puisque le concept (la forme de la fonction liant la cible aux descripteurs) qu'il peut représenter est moins riche. Mais a contrario, puisque l'on a moins de paramètres à estimer, le processus d'apprentissage lisse mieux les informations portées par les données. La propension au surapprentissage est moindre. On réduit la variance de l'erreur. Et il n'est pas sûr que l'arbitrage soit au désavantage du système le plus simple. Tout dépend du problème à traiter et de la configuration des données (Wikipédia, « Dilemme biais-variance »). Dans le même temps, un classifieur linéaire est plus lisible, plus facile à interpréter, plus aisé à déployer dans les systèmes d'informations. C'est pour ces différentes raisons que les modèles linéaires sont en bonne place dans les projets de data science, et pour longtemps encore.

La seconde hypothèse de l'analyse discriminante considère que les matrices de covariances conditionnelles sont identiques (Bardos, 2001 ; Section V.B) :

$$\Sigma_1 = \cdots = \Sigma_K = \Sigma$$

Concrètement, cela veut dire que les nuages de points conditionnels ont la même forme. Sur nos données « DATA 1 » (Figure 5), la conjecture semble tout à fait crédible.

En pratique, la matrice de covariance commune est estimée à l'aide de la matrice de variance covariance intra-classe (pooled covariance matrix en anglais) :

$$W = \frac{1}{n-K}\sum_{k=1}^{K}(n_k - 1) \times V_k$$

La fonction de classement à maximiser est simplifiée. Nous retirons encore une fois les termes qui ne dépendent pas de « k ».

$$d(y_k, X) = \ln \hat{\pi}_k - \frac{1}{2}(x - \bar{x}_k)W^{-1}(x - \bar{x}_k)^{\mathrm{T}}$$
$$= \ln \hat{\pi}_k - \frac{1}{2}\,\bar{x}_k W^{-1}\bar{x}_k^T + \bar{x}_k W^{-1} x^T$$

Ainsi, nous avons des fonctions de classement linéaires de la forme :

$$d(y_k, X) = a_{k0} + a_{k1}x_1 + \cdots + a_{kp}x_p$$

La constante (a_{k0}) fait référence aux deux premiers termes de la notation matricielle ci-dessus, le dernier terme permet d'obtenir les coefficients (a_{kj}).

La règle d'affectation pour un individu ω à classer est inchangée :

$$y_{k*} = \arg\max_k d(y_k, X(\omega))$$

1.3 Fonction de classement linéaire

La fonction $d(y_k,X)$ est nommée différemment selon les références : fonctions discriminantes linéaires (Tenenhaus, 2007 ; page 368) ; fonction score discriminant (Lebart et al., 2000 ; page 265) ; fonction score (Bardos, 2001 ; page 38). Pour ma part, je préfère utiliser « fonction de classement » en référence à l'anglais « classification function » qui ne souffre pas d'ambiguïtés (Huberty et Olejnik, 2006 ; section 13.4). Ce choix me permet de faire le lien avec la francisation « classifieur » pour « classifier » en anglais lorsque je désigne un modèle prédictif.

La valeur retournée par la fonction de classement lorsqu'elle est appliquée sur un individu ω sera nommée « score » c.-à-d. $d(y_k,X(\omega))$ est le score de l'individu pour la classe « y_k ». Il est proportionnel à la probabilité d'appartenance à la classe $P(Y=y_k \,/\, X(\omega))$, mais il n'en a pas les propriétés : $d(y_k,X(\omega))$ peut être négatif ou supérieur à 1 ; et $\sum_{k=1}^{K} d(y_k, X(\omega)) \neq 1$

Dans cette section, nous détaillons le calcul des coefficients des fonctions de classements pour les exemples DATA 1 (très simple avec K = 2, p = 2) et DATA 2 (plus complexe avec K = 3, p = 8). Nous verrons que les étapes sont de même nature et, c'est un des avantages de l'analyse discriminante prédictive, la technique s'applique naturellement au cas multiclasses ($K \geq 2$) sans aménagements contraignants.

1.3.1 Exemple – Cas à (K = 2) classes

Nous poursuivons le traitement de l'exemple DATA 1 (section 1.2.1). Nous affichons tout d'abord les résultats intermédiaires produits précédemment, puis nous enchaînons avec le calcul de la matrice de covariance intra-classe qui joue un rôle fondamental ici. Les coefficients et les constantes des fonctions de classement viennent naturellement.

```
#effectifs par classe
n_k <- table(y)
print(n_k)

## y
## g1 g2
##  5  6

#pour rappel - les matrices de covariance conditionnelles
print(V_k)

## : g1
##    X1   X2
## X1 10  8.0
## X2  8 10.2
## ------------------------------------------------------------
## : g2
##           X1        X2
## X1 12.166667  9.733333
## X2  9.733333 10.666667

#matrice de covariance intra-classe W
#calculée à partir des matrices conditionnelles V_k
W <- 1/(n-K) * Reduce("+",lapply(levels(y),function(k){(n_k[k]-1)*V_k[[k]]}))
print(W)

##           X1        X2
## X1 11.203704  8.962963
## X2  8.962963 10.459259

#inverse de la matrice W
invW <- solve(W)
print(invW)

##            X1         X2
## X1  0.2838508 -0.2432432
## X2 -0.2432432  0.3040541

#pour rappel les moyennes conditionnelles – en ligne les classes {g1, g2}
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333

#calcul matriciel pour les coefficients associés aux variables
#de la fonction de classement
coef_ <- t(mb_k %*% invW)
```

```r
#étiquetage de l'entête de colonne
colnames(coef_) <- levels(y)

#affichage des coefficients calculés
print(round(coef_,6))

##           g1        g2
## X1 -0.129462  1.507304
## X2  0.608108 -0.972973

#constantes des fonctions de classement
intercept_ <- log(pi_k)-0.5*diag(mb_k %*% invW %*% t(mb_k))
print(round(intercept_,6))

## y
##        g1        g2
## -2.110615 -5.139339
```

A titre de comparaison, voici les valeurs fournies par le logiciel TANAGRA sur les mêmes données (Figure 7). Nos calculs sous R (coefficients des variables, constantes) concordent en tous points. Notons au passage que les opérations sont très simples en définitive. C'est ce qui fait en grande partie le succès de cette approche en pratique. Nous y reviendrons dans la section 1.11.

Classification functions		
Attribute	**g1**	**g2**
X1	-0.129462	1.507304
X2	0.608108	-0.972973
constant	-2.110615	-5.139339

Figure 7 – Coefficients des fonctions de classement – DATA 1 – TANAGRA

1.3.2 Exemple – Cas à (K = 3) classes

Nous traitons les données « DATA 2 » maintenant, dont les caractéristiques sont décrites en page 9 et suivantes. Pour éviter la confusion avec notre premier jeu de données de référence (DATA 1), nous réitérons le chargement des données d'apprentissage et la préparation des structures. Comme nous allons à l'essentiel dans cette section, nous nous rendons bien compte combien les calculs de l'analyse discriminante linéaire sont simples.

Seules les données d'apprentissage nous intéressent pour l'instant, avec n = 52 observations.

```r
#** importation des données d'apprentissage **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")
```

```r
#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#affichage des premières lignes
print(head(DTrain))

##     TYPE  MEOH  ACET BU1  BU2 ISOP MEPR   PRO1 ACAL
## 1 KIRSCH 336.0 225.0 1.0  1.0   92 37.0  177.0  0.0
## 2 KIRSCH 442.0 338.0 1.9 10.0   91 30.0  552.0 31.0
## 3 KIRSCH 373.0 356.0 0.0 29.0   83 27.0  814.0 11.0
## 4 KIRSCH 418.0  62.0 0.8  0.0   89 24.0  342.0  7.0
## 5 KIRSCH  84.0  65.0 2.0  2.0    2  0.0  288.0  6.0
## 6 KIRSCH 632.5 342.3 2.2 52.1  123 38.2 2070.1 13.3

#** préparation des structures **

#matrice des descripteurs X
XTrain <- DTrain[-1]

#vecteur cible y
yTrain <- DTrain$TYPE

#nombre d'observations
n <- nrow(DTrain)
print(n)

## [1] 52

#nombre de classes
K <- nlevels(yTrain)
print(K)

## [1] 3

#nombre d'explicatives
p <- ncol(XTrain)
print(p)

## [1] 8
```

Nous enchaînons ensuite les différents calculs nécessaires à l'estimation des coefficients des fonctions de classement. Si l'on met de côté les affichages, 8 instructions suffisent pour obtenir le résultat attendu sous R.

```r
#effectifs par classe
n_k <- table(yTrain)
print(n_k)
```

```
## yTrain
## KIRSCH  MIRAB  POIRE
##     17     15     20

#proportions par classe
pi_k <- n_k / n
print(pi_k)

## yTrain
##    KIRSCH     MIRAB     POIRE
## 0.3269231 0.2884615 0.3846154

#calcul des moyennes conditionnelles - lignes = classes
mb_k <- as.matrix(aggregate(XTrain,list(yTrain),mean)[,2:(p+1)])
print(mb_k)

##           MEOH     ACET   BU1      BU2      ISOP     MEPR     PRO1     ACAL
## [1,]  371.6765 203.0176  1.20 21.01765  81.58824 28.89412 790.7706 12.01176
## [2,]  934.2000 235.0667 20.20 13.56667  90.93333 29.40000 195.2667 12.35333
## [3,] 1084.3500 185.2500 21.33 49.38000 118.05000 50.00000 317.4000 14.49500

#calcul des matrices de covariance conditionnelles
V_k <- by(as.matrix(XTrain),list(yTrain),cov)
print(V_k)

## : KIRSCH
##             MEOH       ACET        BU1          BU2       ISOP       MEPR
## MEOH 34259.65441 11295.7342  63.556250  2044.777941 5676.95221 1755.31360
## ACET 11295.73419 14266.4678  17.725000   784.245294 2728.96397  854.78699
## BU1     63.55625    17.7250   0.616250     3.604375    4.08125    2.91875
## BU2   2044.77794   784.2453   3.604375   461.584044  191.37647  109.92199
## ISOP  5676.95221  2728.9640   4.081250   191.376471 1543.63235  424.02868
## MEPR  1755.31360   854.7870   2.918750   109.921985  424.02868  150.84059
## PRO1 67355.33051 31507.7937 176.537500 14519.423676 5333.71838 3019.76919
## ACAL  1198.61342   347.5929   3.123750    77.290404  142.38640   40.16132
##             PRO1       ACAL
## MEOH  67355.3305 1198.61342
## ACET  31507.7937  347.59290
## BU1     176.5375    3.12375
## BU2   14519.4237   77.29040
## ISOP   5333.7184  142.38640
## MEPR   3019.7692   40.16132
## PRO1 563401.9322 2659.62349
## ACAL   2659.6235   94.12610
## ------------------------------------------------------------
## : MIRAB
##            MEOH        ACET         BU1        BU2        ISOP        MEPR
## MEOH 22953.1714  2391.20000  188.171429  448.92857  -825.70000 -340.228571
## ACET  2391.2000 31890.78095 -321.800000  -45.63333 -2517.78095 -672.457143
## BU1    188.1714  -321.80000   50.885714  214.19286   -71.98571   -9.085714
## BU2    448.9286   -45.63333  214.192857 1157.14810  -190.33095   -1.000000
## ISOP  -825.7000 -2517.78095  -71.985714 -190.33095  2886.49524  595.242857
## MEPR  -340.2286  -672.45714   -9.085714   -1.00000   595.24286  137.542857
## PRO1  7059.8000 14724.69524 1050.442857 6526.73810 -2045.12381 -328.257143
## ACAL   404.1029  1031.91048  -10.604286  -29.24310   -91.21762  -22.951429
##            PRO1       ACAL
## MEOH  7059.8000  404.10286
## ACET 14724.6952 1031.91048
## BU1   1050.4429  -10.60429
## BU2   6526.7381  -29.24310
## ISOP -2045.1238  -91.21762
## MEPR  -328.2571  -22.95143
## PRO1 46172.7810  481.79190
## ACAL   481.7919   62.55552
```

```
## ------------------------------------------------------------
## : POIRE
##             MEOH        ACET        BU1        BU2        ISOP        MEPR
## MEOH  57971.7132  8464.59211  580.28895 -8558.5558  4840.40263  2229.47368
## ACET   8464.5921  3412.93421  -61.66579  -243.2263   313.82895   351.47368
## BU1     580.2889   -61.66579   55.39905  -102.9952   230.74053    60.82105
## BU2   -8558.5558  -243.22632 -102.99516  6046.5796 -1132.56211  -327.62105
## ISOP   4840.4026   313.82895  230.74053 -1132.5621  2134.47105   792.63158
## MEPR   2229.4737   351.47368   60.82105  -327.6211   792.63158   405.26316
## PRO1 -41487.4105 -1165.42105 -219.82316 45159.2084 -5651.28421 -1513.94737
## ACAL    842.9018   140.16447   13.83121  -102.7454   -13.83658   -10.70526
##             PRO1       ACAL
## MEOH -41487.4105  842.90184
## ACET  -1165.4211  140.16447
## BU1    -219.8232   13.83121
## BU2   45159.2084 -102.74537
## ISOP  -5651.2842  -13.83658
## MEPR  -1513.9474  -10.70526
## PRO1 370844.2526 -534.60316
## ACAL   -534.6032   45.22892
```

```
#matrice de covariance intra-classes W
W <- 1/(n-K) * Reduce("+",lapply(levels(yTrain),function(k){(n_k[k]-1)*V_k[[k]]}))
print(W)
```

```
##            MEOH       ACET         BU1         BU2        ISOP       MEPR
## MEOH 40223.7025  7653.7918  299.526327 -2522.67577  3494.67521 1340.44526
## ACET  7653.7918 15093.4728 -110.066327   148.72976   293.41307  223.26922
## BU1    299.5263  -110.0663   36.221265    22.43800    70.23612   21.94082
## BU2  -2522.6758   148.7298   22.438000  2825.92714  -431.04673  -91.42956
## ISOP  3494.6752   293.4131   70.236122  -431.04673  2156.40818  615.87467
## MEPR  1340.4453   223.2692   21.940816   -91.42956   615.87467  245.69489
## PRO1  7923.7079 14043.4170  272.533878 24116.53208 -1034.01305  305.21851
## ACAL   833.6814   462.6808    3.353327   -22.95753    15.06614    2.40533
##             PRO1       ACAL
## MEOH   7923.7079 833.681423
## ACET  14043.4170 462.680778
## BU1     272.5339   3.353327
## BU2   24116.5321 -22.957528
## ISOP  -1034.0130  15.066136
## MEPR    305.2185   2.405330
## PRO1 340956.9520 798.808215
## ACAL    798.8082  66.145806
```

```
#et son inverse W-1
invW <- solve(W)
print(invW)
```

```
##               MEOH          ACET           BU1           BU2          ISOP
## MEOH  5.151673e-05 -5.665060e-06 -2.889789e-04  8.289551e-05  1.286527e-05
## ACET -5.665060e-06  9.451608e-05  4.412034e-04  7.641926e-06  3.334750e-05
## BU1  -2.889789e-04  4.412034e-04  3.350219e-02 -9.473232e-04 -7.762110e-04
## BU2   8.289551e-05  7.641926e-06 -9.473232e-04  1.181606e-03  1.420886e-04
## ISOP  1.286527e-05  3.334750e-05 -7.762110e-04  1.420886e-04  1.715551e-03
## MEPR -2.402043e-04 -1.657079e-04 -2.446560e-04 -1.886501e-04 -4.272374e-03
## PRO1 -5.175614e-06 -3.060095e-06  3.051027e-05 -8.531588e-05 -8.282882e-07
## ACAL -4.979462e-04 -5.740558e-04 -1.653935e-03  3.646985e-04 -5.321351e-04
##               MEPR          PRO1          ACAL
## MEOH -2.402043e-04 -5.175614e-06 -4.979462e-04
## ACET -1.657079e-04 -3.060095e-06 -5.740558e-04
## BU1  -2.446560e-04  3.051027e-05 -1.653935e-03
## BU2  -1.886501e-04 -8.531588e-05  3.646985e-04
## ISOP -4.272374e-03 -8.282882e-07 -5.321351e-04
## MEPR  1.616193e-02 -1.241638e-05  4.668851e-03
```

```
## PRO1 -1.241638e-05  9.330523e-06 -5.656059e-05
## ACAL  4.668851e-03 -5.656059e-05  2.625442e-02

#calcul des coefficients des variables akj
#pour la fonction de classement
coef_ <- t(mb_k %*% invW)

#en-tête de colonnes
colnames(coef_) <- levels(yTrain)

#affichage
print(round(coef_,6))

##         KIRSCH     MIRAB     POIRE
## MEOH  0.003428  0.029028  0.033390
## ACET  0.006390  0.016413  0.007513
## BU1  -0.063681  0.405390  0.318047
## BU2  -0.000883  0.071352  0.114993
## ISOP  0.023082  0.029763 -0.008486
## MEPR  0.037494 -0.128942  0.061780
## PRO1  0.001971 -0.005413 -0.008318
## ACAL  0.066184 -0.226424 -0.130332

#les constantes ak0
intercept_ <- log(pi_k)-0.5*diag(mb_k %*% invW %*% t(mb_k))
print(round(intercept_,6))

## yTrain
##     KIRSCH      MIRAB      POIRE
##  -5.016453 -18.840685 -24.764879
```

A titre de comparaison, voici les résultats de TANAGRA sur les mêmes données (Figure 8).

Classification functions			
Attribute	**KIRSCH**	**MIRAB**	**POIRE**
MEOH	0.003428	0.029028	0.033390
ACET	0.006390	0.016413	0.007513
BU1	-0.063681	0.405390	0.318047
BU2	-0.000883	0.071352	0.114993
ISOP	0.023082	0.029763	-0.008486
MEPR	0.037494	-0.128942	0.061780
PRO1	0.001971	-0.005413	-0.008318
ACAL	0.066184	-0.226424	-0.130332
constant	-5.016453	-18.840686	-24.764879

Figure 8 – Coefficients des fonctions de classement sous TANAGRA – DATA 2 / TRAIN

Nos calculs sous R sont tout à fait conformes.

1.4 Prédiction avec les fonctions de classement

Pour un individu supplémentaire à classer, la prédiction repose sur deux étapes. Nous calculons dans un premier temps le score de chaque classe à l'aide des K fonctions de classement.

$$d(y_k, X(\omega)) = a_{k1}x_1(\omega) + a_{k2}x_2(\omega) + \cdots + a_{kp}x_p(\omega) + a_{k0}$$
$$= \left(x_1(\omega), \cdots, x_p(\omega)\right)\begin{pmatrix} a_{k1} \\ \cdots \\ a_{kp} \end{pmatrix} + a_{k0}$$

Avec les capacités de R en matière de calcul matriciel, les opérations tiennent en 2 lignes de commandes.

Dans un second temps, nous identifions le score le plus élevé pour chaque individu, et nous associons la classe correspondante. Une ligne de commande sous R suffit amplement.

Remarque : En cas d'ex-aequo sur le maximum, ce qui arrive vraiment très rarement, une règle possible consiste à attribuer la classe avec la prévalence π_k maximale. Nous allons au plus simple, nous n'en tenons pas compte dans le code ci-dessous.

Exemple DATA 2

Chargement de l'échantillon test. Illustrons le processus avec DATA 2. Nous chargeons l'échantillon test avec 50 observations.

```
DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")

#inspection
str(DTest)

## 'data.frame':    50 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  3 475 186 371 583 0 421 557 167 523 ...
##  $ ACET: num  15 172 101 414 226 25 142 447 86 367 ...
##  $ BU1 : num  0.2 1.9 0 1.2 2.3 0.1 1.6 0 0 2.6 ...
##  $ BU2 : num  30 7 1.6 0 19 8 8 34 0 30 ...
##  $ ISOP: num  9 113 36 97 120 0 75 107 32 116 ...
##  $ MEPR: num  9 33 11 39 46 6 24 39 10 45 ...
##  $ PRO1: num  350 546 128 502 656 253 128 162 114 787 ...
##  $ ACAL: num  9 14 8 9 11 7 31 94 8 25 ...

#affichage des premières lignes
print(head(DTest))
```

```
##     TYPE MEOH ACET BU1  BU2 ISOP MEPR PRO1 ACAL
## 1 KIRSCH    3   15 0.2 30.0    9    9  350    9
## 2 KIRSCH  475  172 1.9  7.0  113   33  546   14
## 3 KIRSCH  186  101 0.0  1.6   36   11  128    8
## 4 KIRSCH  371  414 1.2  0.0   97   39  502    9
## 5 KIRSCH  583  226 2.3 19.0  120   46  656   11
## 6 KIRSCH    0   25 0.1  8.0    0    6  253    7

#matrice des descripteurs X
XTest <- DTest[-1]

#vecteur cible y
yTest <- DTest$TYPE
```

Calcul des scores des individus. Nous calculons les scores par classe de ces individus à partir des paramètres des fonctions de classement (coefficients des variables + constantes).

```
#calcul des "scores" en test
#application des coefficients a_kj
scores_ <- as.matrix(XTest) %*% coef_

#rajouter la constante a_k0
scores_ <- t(apply(scores_,1,function(ligne){ligne + intercept_}))

#affichage des premières lignes
print(head(scores_))

##         KIRSCH      MIRAB      POIRE
## [1,] -3.118809 -21.110618 -24.643294
## [2,]  3.432298  -7.976511 -11.489500
## [3,] -1.709626 -14.520620 -19.344743
## [4,]  4.111012  -7.686396 -12.647287
## [5,]  4.778819  -4.320783  -5.750754
## [6,] -3.683179 -21.547029 -26.271420
```

Prédiction des classes à partir des scores. Il reste à identifier le maximum dans chaque ligne pour produire la prédiction.

```
#déduction de la prédiction à partir des scores
pred_ <- apply(scores_,1, function(ligne){levels(yTrain)[which.max(ligne)]})
print(pred_)

##  [1] "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH"
##  [9] "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "POIRE"  "MIRAB"
## [17] "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"
## [25] "MIRAB"  "MIRAB"  "MIRAB"  "POIRE"  "MIRAB"  "MIRAB"  "POIRE"  "MIRAB"
## [33] "MIRAB"  "MIRAB"  "POIRE"  "KIRSCH" "POIRE"  "POIRE"  "MIRAB"  "POIRE"
## [41] "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "MIRAB"
## [49] "POIRE"  "POIRE"
```

Evaluation des performances prédictives. Ce n'est pas le propos de cette section. Mais, puisque nous en sommes à ce stade, disons un mot sur l'évaluation des performances à partir des prédictions en test.

Dans un schéma « holdout » où nous construisons le modèle sur un échantillon d'apprentissage, puis nous effectuons une prédiction sur un échantillon test, puisque le second n'a pas participé à l'estimation des coefficients des fonctions de classement, la confrontation des prédictions avec les classes observées, via un tableau croisé nommé « matrice de confusion », permet d'obtenir une évaluation non biaisée des performances du modèle en déploiement. De cette matrice sont déduits des indicateurs de performances que l'on peut interpréter. Nous nous contentons du taux d'erreur ici. D'autres mesures sont disponibles. Elles sont détaillées dans un de mes tutoriels consacrés à la pratique de l'analyse discriminante sous R (TUTO 1, section 4.5).

```
#confrontation avec les classes observées en test
#matrice de confusion
mc <- table(yTest,pred_)
print(mc)

##         pred_
## yTest    KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       0    14     3
##   POIRE       1     5    13

#calcul du taux d'erreur
err <- 1 - sum(diag(mc)) / sum(mc)
print(err)

## [1] 0.18
```

Appliqué à un individu quelconque de la population, le modèle a 18% de chances (si on peut dire) d'effectuer une prédiction erronée.

1.5 Redressement pour les données non représentatives

L'idéal est de travailler sur un échantillon représentatif de la population. Ça n'est pas toujours possible malheureusement. Lorsque qu'une des classes est très rare, celle que l'on cherche à identifier parfois (ex. les personnes victimes d'une certaine maladie, les fraudeurs dans des transactions bancaires, etc.), un tirage aléatoire simple ne permet pas d'obtenir suffisamment d'observations. On procède alors à un tirage rétrospectif (Celeux et Nakache, 1994 ; section

1.3.2) : on effectue un tirage aléatoire au sein des individus associés à la classe d'intérêt, on fait de même dans les autres (dans l'autre si nous sommes dans un cadre binaire), on s'arrange pour que les effectifs soient équilibrés parce que cela améliore la puissance et la robustesse de certains tests statistiques. Dans ce cas, les proportions estimées des classes $\hat{\pi}_k = \frac{n_k}{n}$ ne reflète plus la vraie probabilité a priori (π_k), les calculs sont faussés.

Si l'on a connaissance des vraies valeurs des prévalences des classes (π_k), grâce à des études antérieures par exemple, j'avais détaillé la solution analytique pour la régression logistique (Rakotomalala, 2011 ; section 10.1). Elle était relativement simple. Elle l'est encore plus pour l'analyse discriminante linéaire. Rappelons que la fonction de classement s'écrit comme suit :

$$d(y_k, X) = \ln \hat{\pi}_k - \frac{1}{2}\, \bar{x}_k W^{-1} \bar{x}_k^T + \bar{x}_k W^{-1} x^T$$

Si l'on connaît la vraie valeur (π_k), il suffit de la substituer dans l'équation pour obtenir :

$$d(y_k, X) = \ln \pi_k - \frac{1}{2}\, \bar{x}_k W^{-1} \bar{x}_k^T + \bar{x}_k W^{-1} x^T$$

Seule la constante a_{k0} est impactée par la correction. Des logiciels, et non des moindres, nous donnent cette possibilité avec l'option PRIORS : SAS par exemple (SAS, voir « PRIORS Statement »), le logiciel R avec la procédure lda() du package « MASS », ou encore Python avec la classe LinearDiscriminantAnalysis du package « scikit-learn » dont le mode opératoire a été largement étudié dans un tutoriel (TUTO 2).

Dans le cas où l'outil utilisé ne sait pas en tenir compte, il nous est facile de corriger manuellement la constante (a_{k0}) fournie par le logiciel en retranchant le logarithme des proportions apparentes (obtenues dans l'échantillon d'apprentissage) et en additionnant celui de la probabilité fournie, soit :

$$a_{k0}^* = a_{k0} - \ln \hat{\pi}_k + \ln \pi_k$$

Le redressement du modèle avec des informations sur la « vraie » prévalence des classes (π_k) en améliore considérablement les performances prédictives. Je l'ai montré pour le cas particulier de la régression logistique sur des données simulées (Rakotomalala 2011 ; sections 10.1.2 et

10.1.3). La démarche est généralisable à tout type d'algorithme. Dans le cadre binaire tout du moins, quand une solution analytique n'est pas possible, l'idée est de jouer sur le seuil d'affectation aux classes (Rakotomalala R., « Redressement – Affectation optimale dans le cadre du tirage rétrospectif »).

Exemple DATA 1

Nous reprenons l'exemple sur le cas binaire (DATA 1) que nous avions traité précédemment (section 1.3.1). La dernière instruction permettait d'obtenir la constante (intercept_) du modèle. Mettons que la « vraie » distribution des classes est ($\pi_k = \frac{1}{K} = \frac{1}{2}$). La première solution consiste à la substituer dans le calcul de la constante.

```
#constante corrigée
#calcul direct à partir des prévalences connues (πk = 1/K = 1/2)
intercept_corrige_1 <- log(1/K)-0.5*diag(mb_k %*% invW %*% t(mb_k))
print(round(intercept_corrige_1,6))

## [1] -2.015305 -5.226350
```

La seconde consiste à corriger la constante obtenue à partir des prévalences apparentes ($\hat{\pi}_k$). Cette piste est à privilégier si le logiciel utilisé ne sait pas prendre en compte une option PRIORS.

```
#constante corrigée
#correctif de la constante obtenue à partir des prévalences apparentes
#estimées sur les données (π̂k = pi_k dans le programme)
intercept_corrige_2 <-  intercept_ - log(pi_k) + log(1/K)
print(round(intercept_corrige_2,6))

## y
##        g1        g2
## -2.015305 -5.226350
```

Bien évidemment, les deux stratégies fournissent exactement le même résultat.

1.6 Calcul des probabilités d'affectation

Dans certaines circonstances, il est plus intéressant de disposer d'une estimation de la probabilité $P(Y = y_k / X)$ plutôt qu'un score lorsque l'on applique le modèle sur des individus supplémentaires : pour l'interprétation des résultats (une probabilité varie entre 0 et 1, on évalue mieux le degré d'appartenance aux classes) ; pour le scoring (s'il faut fixer un seuil pour le ciblage, c'est plus facile avec une probabilité dont on comprend la teneur) ; pour des éventuels

post-traitements (la correction des prédictions avec des coûts de mauvaise affectation n'a de sens que si l'on dispose de probabilités) ; etc.

La fonction softmax est tout à fait appropriée dans notre contexte. Elle transforme un vecteur de K réels de signes quelconques en un vecteur de valeurs comprises entre 0 et 1, dont la somme fait 1. Quelque chose qui ressemble fortement à une probabilité. Elle s'applique naturellement aux problèmes multiclasses (K ≥ 2). Enfin, cette transformation ne dénature en rien la prédiction. La classe associée à la valeur maximum sera toujours la même.

Dans notre cas, la probabilité a posteriori estimée s'écrit :

$$\hat{P}(Y = y_k/X) = \frac{e^{d(y_k,X)}}{\sum_{c=1}^{K} e^{d(y_c,X)}}$$

Exemple DATA 2

Sous R, l'opération est très simple. Nous reprenons les scores_ estimés sur l'échantillon test DATA 2 (section 1.4) utilisés pour la prédiction. La transformation softmax pour l'ensemble des individus tient en une seule ligne de commande.

```
#transformation softmax à partir des scores_
probas_ <- t(apply(scores_,1,function(ligne){exp(ligne)/sum(exp(ligne))}))
print(head(probas_))

##         KIRSCH        MIRAB        POIRE
## [1,] 1.0000000 1.535523e-08 4.487814e-10
## [2,] 0.9999886 1.109717e-05 3.307808e-07
## [3,] 0.9999972 2.730580e-06 2.193638e-08
## [4,] 0.9999924 7.523981e-06 5.271811e-08
## [5,] 0.9998616 1.116948e-04 2.673034e-05
## [6,] 1.0000000 1.745132e-08 1.549001e-10
```

Le 1er individu appartient presque à coup sûr à la classe KIRSCH, etc. Le second également, etc. Les valeurs sont très tranchées pour les 6 premiers individus. Si l'on considère l'individu n°15 en revanche, les valeurs sont plus nuancées, et c'est la classe POIRE qui l'emporte.

```
#individu n°15
print(round(probas_[15,],6))

##   KIRSCH    MIRAB    POIRE
## 0.000001 0.308998 0.691001
```

Car, comme annoncé plus haut, nous pouvons baser les prédictions sur ces estimations des probabilités a posteriori. Elles sont exactement identiques à celles fondées sur les scores.

```
#prédiction à partir des probas - recherche de classe max. par ligne
pred_from_probas_ <- apply(probas_,1, function(ligne){levels(yTrain)[which.max(ligne)]})

#confrontation avec les prédictions basées sur les scores
print(table(pred_from_probas_, pred_))

##                  pred_
## pred_from_probas_ KIRSCH MIRAB POIRE
##            KIRSCH     15     0     0
##            MIRAB       0    19     0
##            POIRE       0     0    16
```

1.7 Interprétation – Lecture des coefficients

Les coefficients des fonctions de classement, par le niveau des valeurs et leurs signes, donnent des indications sur le sens de la relation entre les prédicteurs et les classes.

(K = 2). Pour le cas à (K = 2) classes, nous avons deux coefficients à opposer par variable (Figure 7). Leur lecture est très simple (plus simple encore, nous le verrons plus loin, section 5.2). Pour DATA 1 :

- X1, nous verrons qu'elle joue un rôle pertinent dans le modèle (section 2.3), discrimine « g1 » et « 2 » parce que ($a_{g1,X1}$ = −0.129 vs. $a_{g2,X1}$ = 1.507). Il y a un gap entre les coefficients qui sont de surcroît de signes opposés. Ainsi, la classe « g1 » (resp. « g2 ») est caractérisée par les valeurs faibles (resp. valeurs élevées) de « X1 ».
- X2 pèse également, mais dans l'autre sens, « g1 » (« g2 ») est caractérisée par les valeurs élevées (faibles) de X2.
- Dans ce cas particulier où les variables X1 et X2 sont exprimées sur la même échelle, nous pouvons même conclure au regard de l'intensité de l'écart entre les coefficients que X1 a un impact légèrement plus élevé dans le modèle.

Ces déductions sont confirmées par le graphique montrant les nuages de points conditionnels dans le plan (X1, x2) (Figure 1).

(K > 2). Lorsque (K > 2), la lecture est un peu plus complexe mais reste tout à fait réalisable. Pour DATA 2, à la lecture du tableau des coefficients (Figure 8), nous déduisons concernant KIRSCH qu'elle est en opposition avec le tandem (MIRAB, POIRE) pour MEOH avec les coefficients (0.003428, 0.029028, 0.033390). Elle (la classe KIRSCH) est caractérisée par des valeurs faibles de MEOH (en moyenne), c'est l'inverse pour (MIRAB, POIRE). Nous pouvons le confirmer dans un boxplot conditionnel (qui analyse la variable individuellement, indépendamment des autres certes).

```
#caractérisation de TYPE par MEOH
boxplot(MEOH ~ TYPE, data = DTrain)
```

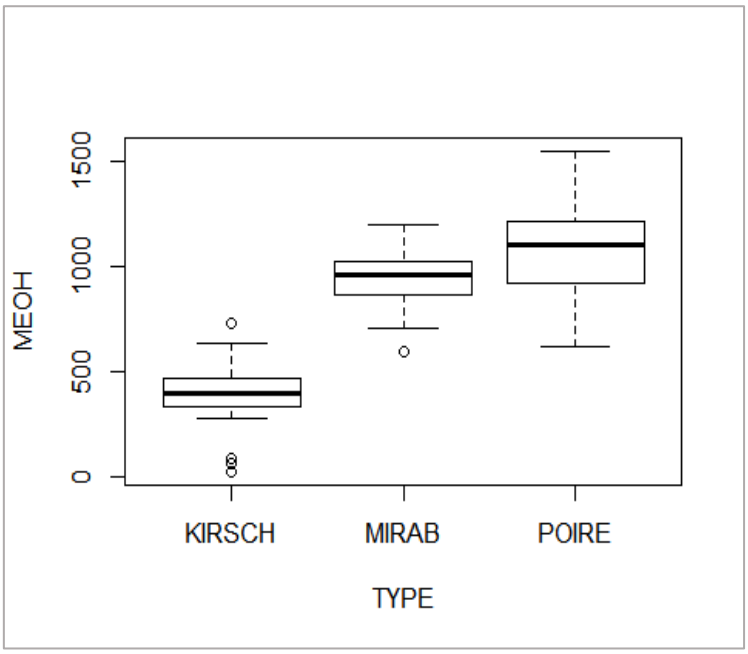


Figure 9 – Caractérisation de TYPE selon la variable MEOH – DATA 2 / TRAIN

Il faut néanmoins être prudent dans les interprétations. Comme en régression, les colinéarités entre les variables peuvent fausser les lectures. Il faut aussi s'assurer de l'absence de différences d'échelles avant de se lancer dans des comparaisons inter-variables.

1.8 Interprétation – Distance aux centres de classes

A ce stade, lorsque je présente ce cours à mes étudiants, arrive un intermède que j'apprécie tout particulièrement. Je leur rappelle que je me suis basé sur une démarche exclusivement probabiliste jusqu'à présent, avec des hypothèses qui portent sur les distributions statistiques.

Et je leur dis qu'en réalité il y a d'autres prismes de lecture possibles de l'analyse discriminante. J'enchaîne alors avec deux thèmes importants : les proximités entre les individus dans l'espace de représentation, en particulier la distance aux barycentres conditionnels pour ce qui est de l'analyse discriminante ; les frontières de séparation (droite ou hyperplan séparateur dans le cas linéaire) pour discriminer les classes en machine learning. A cette occasion, j'essaie de faire le lien avec d'autres méthodes prédictives.

La lecture de la fonction de classement en termes de distances aux barycentres conditionnels qui a été mise en avant pour l'analyse discriminante quadratique (Figure 6) reste valable pour l'analyse discriminante linéaire. En multipliant $d(y_k,X)$ par (-2), nous obtenons le carré de la distance généralisée aux groupes (TUTO 3, section 3.4.5 ; SAS, page 2175), basée toujours sur la distance de Mahalanobis aux barycentres dans le repère originel, mais avec une matrice de pondération commune aux différentes classes y_k :

$$D(y_k,X) = (x-\bar{x}_k)W^{-1}(x-\bar{x}_k)^{\mathrm{T}} - 2 \times \ln \hat{\pi}_k$$

Comme nous avons multiplié par (-2), la règle d'affectation correspond à une minimisation pour un individu ω à classer :

$$y_{k*} = \arg\min_k D(y_k, X(\omega))$$

Nous affectons l'individu à la classe le plus proche, mais en tenant compte de la prévalence des classes. Si elles sont équilibrées ($\pi_k = \frac{1}{K}$), seule la proximité avec les barycentres compte.

Exemple DATA 1

Calculons la distance du point supplémentaire (■) aux barycentres conditionnels (▲) dans notre repère (X1, X2) de DATA 1. Nous rappelons les principaux repères pour les calculs, puis nous enchaînons avec $D(y_k,X)$.

```
#rappel barycentres
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333
```

```r
#rappel coordonnées du point à classer
print(omega)

##      [,1] [,2]
## [1,]    6    7

#distribution apparente des classes
print(pi_k)

## y
##        g1        g2
## 0.4545455 0.5454545

#matrice W-1
print(invW)

##            X1         X2
## X1  0.2838508 -0.2432432
## X2 -0.2432432  0.3040541

#distance carrée généralisée à g1
D1 <- (omega-mb_k[1,]) %*% invW %*% t(omega-mb_k[1,]) - 2 * log(pi_k[1])
print(round(D1,2))

##      [,1]
## [1,] 1.95

#distance carrée généralisée à g2
D2 <- (omega-mb_k[2,]) %*% invW %*% t(omega-mb_k[2,]) - 2 *log(pi_k[2])
print(round(D2,2))

##      [,1]
## [1,] 10.5
```

Le calcul confirme l'impression visuelle. L'individu est manifestement rattaché à la première classe « g1 ». Son barycentre (▲) est nettement plus proche (Figure 10).

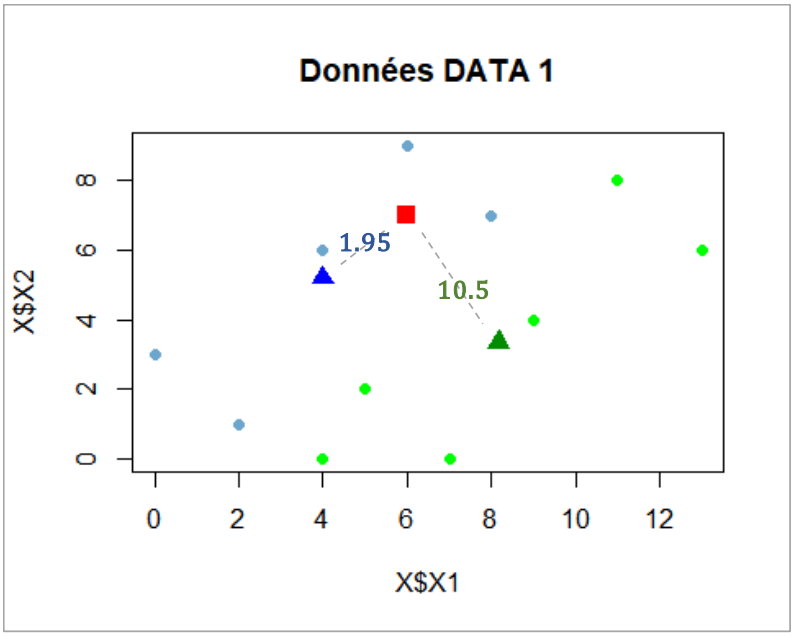


Figure 10 – Distance généralisées aux centres de classes du point supplémentaire – DATA 1

1.9 Interprétation – Frontière(s) de décision

La frontière de séparation des classes est un autre thème important de l'analyse prédictive. Elle s'applique au cadre multiclasses (K ≥ 2), mais elle est surtout simple à appréhender pour le cas binaire (K = 2). En déploiement, on affecte à la classe qui maximise la fonction de classement. Nous sommes dans une zone d'indécision qui définit une frontière de séparation lorsqu'il y a égalité. Ainsi, selon que l'en est au-delà ou en-deçà de cette frontière, l'une ou l'autre classe est attribuée à l'individu. Lorsque (p = 2), nous avons une droite, sinon pour (p > 2) nous avons un hyperplan séparateur. Dans la littérature, cette notion est souvent mise en avant lorsqu'on parle de certaines méthodes comme les SVM (support vector machine). Moins quand il s'agit d'analyse discriminante linéaire. Pourtant les principes sont les mêmes, seul le mode d'obtention des coefficients de l'équation associée est différent (TUTO 2, section 4.2 ; TUTO 4, section 4).

1.9.1 Cas à (K = 2) classes

Voyons cela concrètement pour DATA 1. Rappelons que les fonctions de classement s'écrivent (Figure 7) :

$$d(g_1, X) = -0.129\, x_1 + 0.608\, x_2 - 2.111$$
$$d(g_2, X) = 1.507\, x_1 - 0.973\, x_2 - 5.139$$

La frontière est définie par l'égalité entre ces deux équations :

$$-0.129\, x_1 + 0.608\, x_2 - 2.111 = 1.507\, x_1 - 0.973\, x_2 - 5.139$$

Son équation explicite de la forme ($x_2 = c_1\, x_1 + c_0$) s'écrit :

$$x_2 = 1.035\, x_1 - 1.916$$

Exemple DATA 1

Pour DATA 1, nous disposons de (p = 2) descripteurs (X1, X2) avec (K = 2) classes. Nous matérialisons la droite de séparation dans l'espace de représentation (Figure 11).

```
#rappel coefs fonctions de classement
print(coef_)

##            g1        g2
## X1 -0.1294617  1.507304
## X2  0.6081081 -0.972973
```

```r
#et les constantes
print(intercept_)

## y
##        g1        g2
## -2.110615 -5.139339

#pente de la droite de séparation
c1 <- (coef_[1,1] - coef_[1,2])/(coef_[2,2] - coef_[2,1])
print(c1)

## [1] 1.035219

#constante de la droite
c0 <- (intercept_[1] - intercept_[2])/(coef_[2,2] - coef_[2,1])
print(c0)

##        g1
## -1.915603

#tracé de la droite dans l'espace de représentation
plot(X$X1,X$X2,col=c("skyblue3","green")[y], pch = 19, main="Frontière entre les classes")
abline(a=c0,b=c1,lwd=2)
```

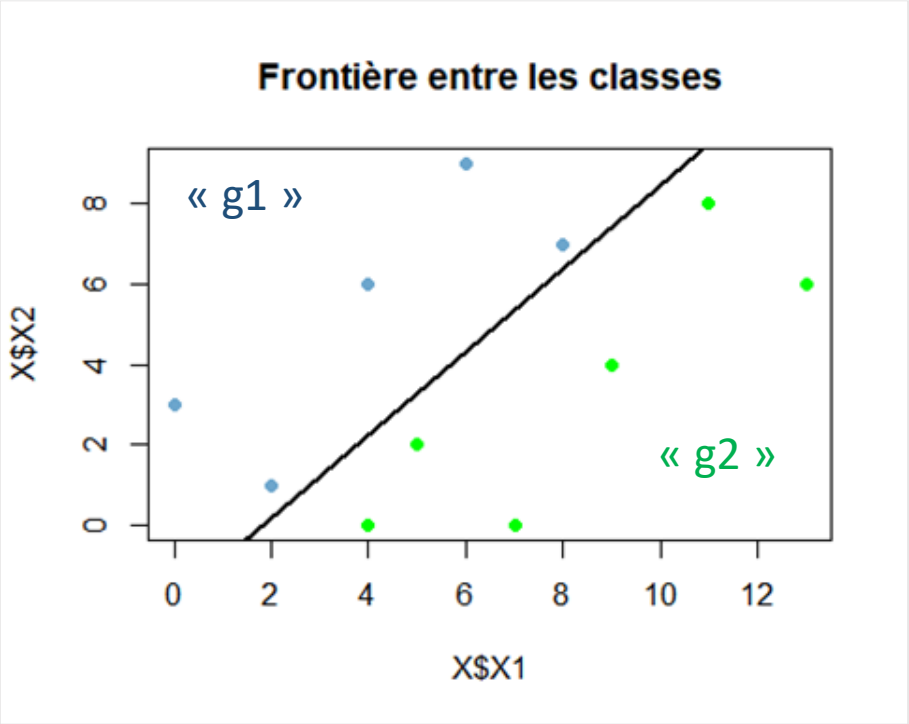


Figure 11 – Frontière entre les classes – DATA 1 (K = 2)

Les classes sont parfaitement discriminées. Nous observons 2 zones distinctes d'appartenance aux classes. Le taux d'erreur en resubstitution, calculé sur les données ayant servi à la construction du modèle, est nul.

En déploiement, il s'agit alors de positionner les individus supplémentaires par rapport à la frontière. Pour l'exemple du point (■), de coordonnées (x1 = 6, x2 = 7) (section 1.7), il est situé « au-dessus » de la frontière. La modalité « g1 » lui est attribuée sans ambiguïtés (Figure 12).

```
#position des observations et droite de séparation
plot(X$X1,X$X2,col=c("skyblue3","green")[y], pch = 19, main="Individu supplémentaire")
abline(a=c0,b=c1,lwd=2)

#position du point supplémentaire (X1 = 6, x2 = 7)
points(6,7,col="red",pch=15,cex=1.5)
```

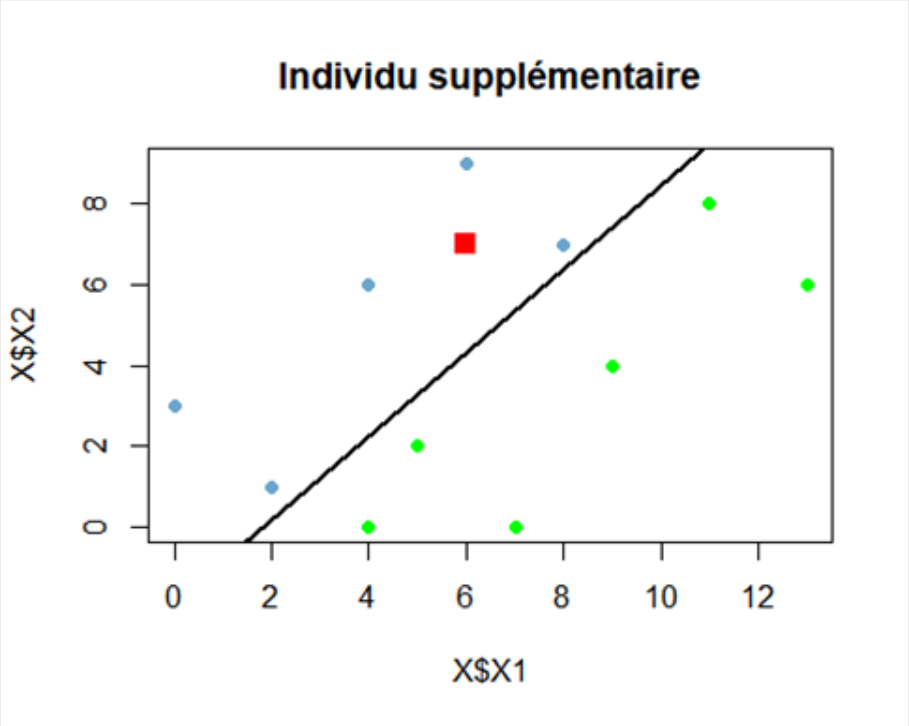


Figure 12 – Position du point supplémentaire (■) par rapport à la frontière de séparation des classes – DATA 1

Les décisions basées sur la distance aux classes et sur la position par rapport à la droite séparatrice sont totalement cohérentes. Heureusement ! Les prismes sont certes différents, mais les deux approches reposent sur des formules identiques en définitive.

1.9.2 Cas à (K = 3) classes

L'appréhension des frontières est moins évidente lorsque le nombre de classes est supérieur à deux (K > 2). Elles peuvent être définies par l'opposition des fonctions de classement prises deux à deux. Nous travaillons sur les données d'apprentissage (52 observations) de DATA 2, mais

en restreignant la prédiction de TYPE avec les variables MEOH (X1) et MEPR (X2). Les calculs sous TANAGRA fournissent les 3 fonctions de classement suivantes (Figure 13).

Classification functions			
Attribute	**KIRSCH**	**MIRAB**	**POIRE**
MEOH	0.006504	0.023512	0.024660
MEPR	0.082120	-0.008616	0.068968
constant	-3.513038	-12.099105	-16.049554

Figure 13 – TYPE = f(MEOH, MEPR), DATA 2 / TRAIN

A partir de ces fonctions, nous établissons 3 équations formées par les oppositions des classes (KIRSCH vs. MIRAB ; KIRSCH vs. POIRE ; MIRAB vs. POIRE) (Figure 14).

Attribute	KIRSCH/MIRAB	KIRSCH/POIRE	MIRAB/POIRE
MEOH	-0.017008	-0.018156	-0.001148
MEPR	0.090736	0.013152	-0.077584
constant	8.586067	12.536516	3.950449

Figure 14 – Opposition des classes prises par paires – DATA 2 / TRAIN

Pour qu'il n'y ait pas de doutes sur ce que nous manipulons, prenons l'exemple de la première équation.

$$-\begin{cases} d(KIRSCH, X) = 0.006504 \times MEOH + 0.082120 \times MEPR - 3.513038 \\ d(MIRAB, X) = 0.023512 \times MEOH - 0.008616 \times MEPR - 12.099105 \end{cases}$$

$$\mathrm{d(KIRSCH/MIRAB)} = -0.017008 \times MEOH + 0.090736 \times MEPR + 8.586067$$

Dans l'espace de représentation défini par (MEOH, MEPR), nous affichons les points selon leur groupe d'appartenance et les 3 frontières définies ci-dessus (Figure 15).

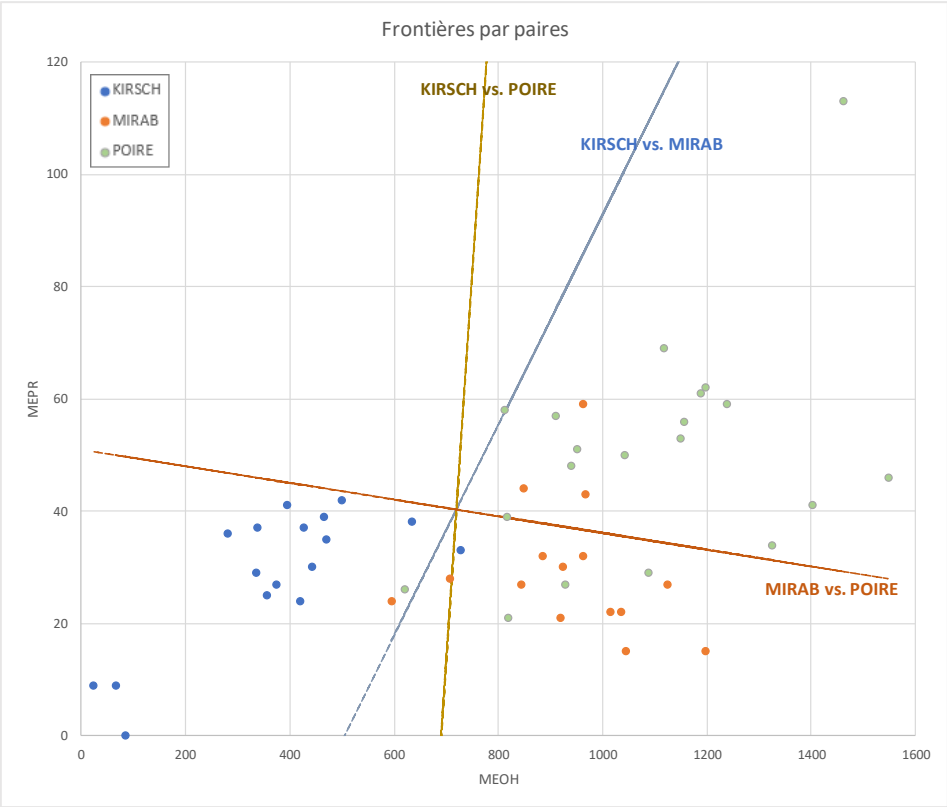


Figure 15 – Frontières définies par les oppositions de classes – DATA 2 / TRAIN

Nous pouvons ainsi définir les zones d'influence des classes (Figure 16).

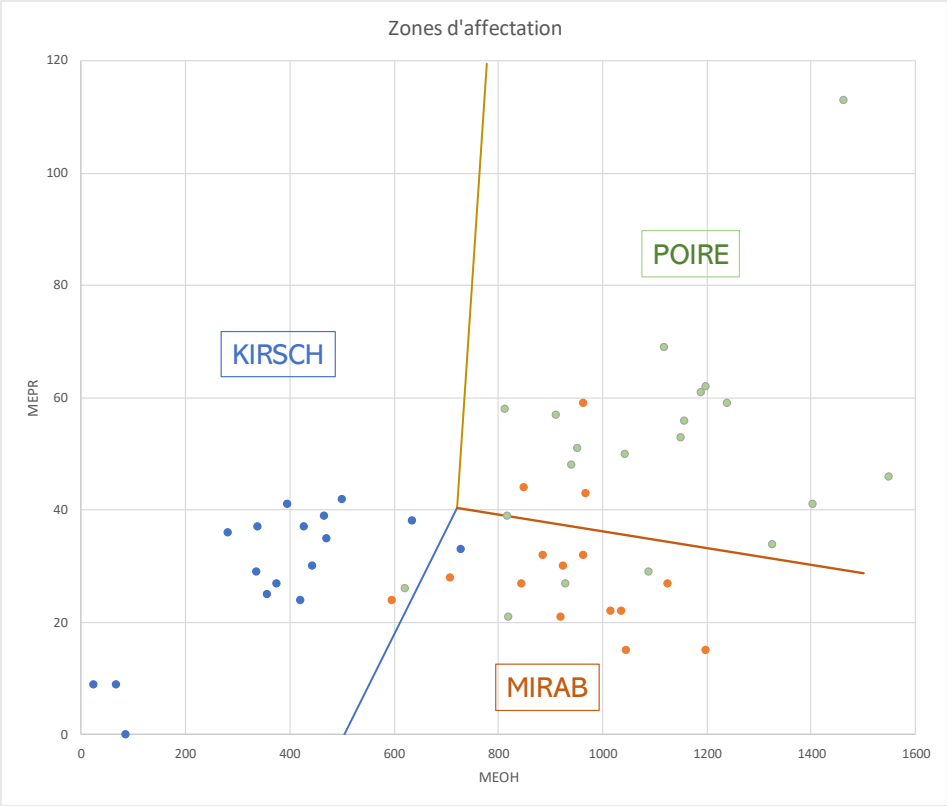


Figure 16 – Zones d'appartenance aux classes – DATA 2 / TRAIN

Les observations placées dans les « mauvaises zones » correspondent aux erreurs constatées dans la matrice de confusion en resubstitution (Figure 17). Il y a 1 MIRAB (un orange) et 1 POIRE (un vert) étiquetés KIRSCH (parmi les bleus) ; 1 KIRSCH et 3 POIRES étiquetés MIRAB ; 3 MIRAB étiquetés POIRE.

Matrice de confusion					
		Classes prédites			
		KIRSCH	MIRAB	POIRE	Sum
Classes observées	KIRSCH	16	1	0	17
	MIRAB	1	11	3	15
	POIRE	1	3	16	20
	Sum	18	15	19	52

Figure 17 – Matrice de confusion – DATA 2 / TRAIN – TYPE = f(MEOH, MEPR)

1.10 Distance à la frontière séparatrice

Une autre notion importante du classement est la distance à la frontière. Elle n'a vraiment de sens que pour le cadre binaire (K = 2). L'idée est de calculer la distance à l'hyperplan séparateur d'un point. Plus il en est éloigné, tout en étant dans la bonne zone, plus son appartenance à la classe est crédible. Cet indicateur – mis en avant par certains outils tels que « scikit-learn » pour Python – est en liaison directe avec la probabilité d'appartenance aux classes [TUTO 2, section 4.3, decision_function()].

Exemple DATA 1

Pour DATA 1, les observations sont décrites dans le plan (p = 2). La distance à la frontière correspond à la distance d'un point à une droite. La formule est très simple.

Rappelons que l'équation implicite de la frontière pour DATA 1 est obtenue par la différence entre les fonctions de classement $d(g_1, X)$ et $d(g_2,X)$:

$$a_1\, x_1 + a_2\, x_2 + a_0 = 0$$

Soit :

$$-\begin{cases} d(g_1,X) = -0.129\, x_1 + 0.608\, x_2 - 2.111 \\ d(g_2,X) = 1.507\, x_1 - 0.973\, x_2 - 5.139 \end{cases}$$

$$d(X) = -1.637\ x_1 + 1.581\ x_2 + 3.029$$

La distance à cette droite d'un individu ω de coordonnées ($x_1(\omega)$, $x_2(\omega)$) est égal à :

$$\frac{a_1\ x_1(\omega)\ + a_2\ x_2(\omega) + a_0}{\sqrt{a^2 + b^2}}$$

Par rapport à la formule originelle, la distance est signée pour que l'on sache de quel côté de la frontière est positionné l'individu. La règle d'affectation ici est (section 5.2) :

Si d(X) ≥ 0 Alors « g_1 » sinon « g_2 »

```
#droite séparatrice - fonction implicite
a1 <- coef_[1,1] - coef_[1,2]
a2 <- coef_[2,1] - coef_[2,2]
a0 <- intercept_[1] - intercept_[2]

#affichage des coefficients
print(round(c(a1,a2,a0),3))

##                    g1
## -1.637  1.581  3.029

#distance à la frontière de chaque point
#la valeur est signée pour que l'on sache de quel côté de la frontière
#est située l'observation
dist_f <- (as.matrix(X) %*% matrix(c(a1,a2),nrow=2,ncol=1) + a0) / sqrt(sum(c(a1,a2)^2))
print(dist_f)

##             [,1]
##  [1,]  3.4151955
##  [2,]  0.5871918
##  [3,]  2.6225524
##  [4,]  3.2683803
##  [5,]  0.4403766
##  [6,] -0.8757485
##  [7,] -3.7037522
##  [8,] -2.3631579
##  [9,] -1.0225637
## [10,] -3.8505674
## [11,] -1.5460456

#représentation des points
#position des observations et droite de séparation
plot(X$X1,X$X2,type="n", main="Distance à la frontière")
text(X$X1,X$X2,label=1:length(y),col=c("skyblue3","green")[y])
abline(a=c0,b=c1,lwd=2)
```

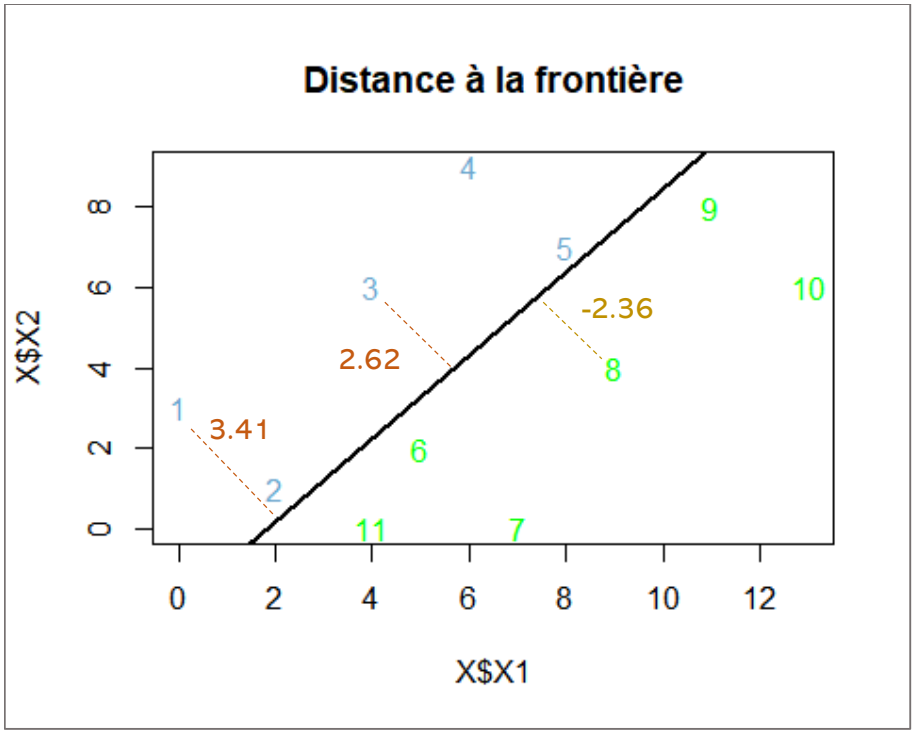


Figure 18 – Distance de quelques points à la frontière – DATA 1

1.11 Avantages et inconvénients de l'analyse discriminante linéaire

Il est temps de faire un premier point sur les qualités et défauts de cette méthode. Tordons déjà le coup à une fausse idée. L'analyse discriminante linéaire est robuste par rapport à ses hypothèses fondatrices. Par conséquent, non, il n'est pas nécessaire de vérifier la multinormalité des distributions conditionnelles et l'homoscédasticité avant de pouvoir l'appliquer. Elle l'est même suffisamment pour être efficace dans des conditions qui semblent pourtant totalement inappropriées, comme l'utilisation des descripteurs qualitatifs (section 3.2) (Hastie et al., 2017 ; section 4.4.5). En réalité, elle se positionne comme une alternative possible parmi les classifieurs linéaires (TUTO 4). C'est dans cette optique qu'il faut la juger (Tufféry, 2012 ; sections 12.8 et 12.9 ; Huberty et Olejnik, 2006 ; Partie V).

1.11.1 Par rapport à la régression logistique

On associe souvent l'analyse discriminante linéaire à la régression logistique dans la littérature (Bardos, 2001, chapitres 2 et 3 ; Saporta, 2006, chapitre 18 ; Tenenhaus, 2007, chapitres 10 et 11 ; Tufféry, 2012, chapitres 12 et 14, etc.). L'analyse discriminante linéaire est dite paramétrique parce que nous émettons des hypothèses sur la distribution conditionnelle $P(X / Y = y_k)$.

L'objectif de l'algorithme d'apprentissage est d'estimer au mieux les paramètres (μ_k, Σ_k) de la loi associée. La régression logistique est dite semi-paramétrique parce que l'hypothèse porte sur le ratio $\frac{P(X\ /\ Y=+)}{P(X\ /\ Y=-)}$ dans le cadre binaire Y $\in$ {+, -}. Cette dernière est moins restrictive, avec un champ d'application théorique plus large, elle peut prendre en compte nativement les descripteurs binaires par exemple. En pratique, ces méthodes présentent des performances similaires parce qu'elles ont un biais de représentation identique : ce sont des classifieurs linéaires, c.-à-d. ils induisent un hyperplan séparateur pour discerner les classes dans l'espace de représentation. Elles diffèrent par le mode d'estimation des coefficients des fonctions de classement simplement.

1.11.2 Avantages de l'analyse discriminante linéaire

L'analyse discriminante linéaire possède plusieurs atouts qui en font une des méthodes favorites pour le scoring (Bardos, 2001 ; page 29).

- Elle repose sur une solution analytique exacte. Il n'y a pas d'option « random_state » à manipuler, les férus de Python / Scikit-Learn me comprendront. Nous sommes assurés d'obtenir la même solution à chaque lancement de l'algorithme sur les mêmes données.
- Elle est très rapide même sur de très grandes bases, pourvu que la dimensionnalité (p) ne soit pas trop importante. Sur les bases usuelles où le ratio (n/p) est favorable, l'ADL fait partie des méthodes de référence.
- Elle est relativement stable (faible variance) sur les petites bases.
- Nous disposons d'un modèle explicite avec les fonctions de classement. Elles sont lisibles, faciles à interpréter et à déployer.
- Elle dispose d'une procédure statistique permettant d'évaluer la pertinence d'un, de plusieurs, ou de la totalité des prédicteurs (chapitre 2).
- Ce qui lui permet de mettre en œuvre des stratégies performantes de sélection de variables (chapitre 3).
- Comme la régression logistique, elle sait produire une estimation viable de la probabilité d'appartenance aux classes, cruciale pour certains post-traitements (intégration des coûts de mauvaise affectation, ...).

- Appliquée sur des bases non-représentatives c.-à-d. où la proportion des classes ne reflète pas leur prévalence réelle, les coefficients associés aux variables sont toujours valables. Ce n'est pas le cas des constantes.
- Mais si nous disposons de la bonne information sur les probabilités a priori des classes, le mécanisme de correction est facile à mettre en œuvre.

Enfin, ce n'est pas un critère opérationnel mais il a son importance pour moi, l'analyse discriminante linéaire est très intéressante pédagogiquement. La mécanique est simple, elle se prête à une multiplicité de points de vue qui enrichissent considérablement notre compréhension des méthodes prédictives. Je prends beaucoup de plaisir à l'enseigner.

1.11.3 Inconvénients de l'analyse discriminante linéaire

Mais toute médaille a son revers.

- C'est un classifieur linéaire, il faut le garder à l'esprit. Une variante non-linéaire est possible avec l'astuce du noyau (Kernel Discriminant Analysis), mais nous perdons la lisibilité des modèles dans ce cas.
- Elle est robuste par rapport à la normalité mais il n'en reste pas moins que les distributions conditionnelles doivent être au moins convexes et bien entendu monomodales.
- Les barycentres conditionnels jouent un rôle primordial, l'analyse discriminante linéaire est de fait très sensible aux points atypiques qui pourraient en fausser l'estimation.
- Il y a des problèmes de calculs lorsque les descripteurs sont fortement colinéaires. La sélection de variables peut être une solution pour y remédier.
- Son vrai problème à mon sens est la dimensionnalité (p élevé et/ou se rapproche dangereusement de n), tant en termes d'efficacité des calculs, qu'en termes de stabilité et performances des modèles. Des procédés de régularisation existent (chapitre 7), mais il y a du boulot par rapport à la régression logistique dont les implémentations ont connu de nombreuses améliorations ces dernières années, d'une part avec les algorithmes basés sur la descente de gradient et ses perfectionnements, et d'autre part avec les techniques de régularisation efficaces telles que Ridge et Lasso.

Mais ne désespérons pas. Dans les versions récentes des librairies, « scikit-learn » version 0.22.2 par exemple, je vois que des options et des modes de calcul pour les traitements des fortes dimensionnalités sont avancés. Comme quoi il est toujours possible de progresser, même pour une méthode qui est presque centenaire aujourd'hui.

2 Evaluation statistique

Les techniques présentées dans ce chapitre ne sont pas des tests de significativité à proprement parler où l'on cherche à établir ou réfuter la nullité des coefficients des fonctions de classement, comme on le ferait en régression logistique (Rakotomalala, 2011 ; chapitre 3) ou en régression linéaire multiple (Rakotomalala R., « Econométrie – La régression linéaire simple et multiple », version 1.1., 2015 ; chapitre 10). Ils ne seraient pas adaptés d'ailleurs puisque la pertinence d'une variable peut s'évaluer à la nullité des coefficients associés certes, mais aussi à leur égalité d'une fonction de classement à une autre.

L'évaluation statistique ici repose sur les mécanismes de la MANOVA (analyse de variance multivariée ; Rakotomalala R., 2010, Partie 3) qui vise à caractériser l'écartement entre les barycentres conditionnels qui tiennent une place prépondérante dans l'analyse discriminante. Curieusement, elle est peu décrite dans la littérature, en tous les cas sous l'angle que nous avons choisi. J'ai trouvé dans quelques références seulement la notion de contributions au modèle des variables prises individuellement ou en groupe (Tomassone et al., 1988, section 3.2 ; Diday et al., 1982, section 4.6). Pourtant leur rôle est reconnu dans la sélection de variables (chapitre 3), où elles servent de critères pour l'adjonction ou le retrait durant les processus pas-à-pas (stepwise) (Tenehaus, 2007, section 10.5 ; Nakache et Confais, 2003, section 1.5.2).

Il en est de même pour les logiciels. A ma connaissance, STATISTICA, du temps où j'y avais accès, est un des rares à présenter une statistique de test de pertinence de variables, sous l'appellation « F d'exclusion », nous comprendrons pourquoi dans ce chapitre. J'avais repris la même idée dans l'implémentation de TANAGRA (TUTO 3, section 3.2.4). Les autres outils n'en

font pas mention, nativement tout du moins. Elle apparaît explicitement en revanche dès lors que l'on met en œuvre un processus de sélection de variables.

2.1 Distance entre centres de classes

Les barycentres conditionnels tiennent un rôle central (c'est le cas de le dire) dans la loi multinormale. Les distributions censées être convexes, plus ils sont écartés, meilleure sera la qualité de l'analyse discriminante qui pourra ainsi tracer des frontières de séparation nettes entre les classes. La distance entre les centres de classes, toujours selon la métrique de Mahalanobis, donne une idée assez bonne, d'une part de la pertinence du modèle, d'autre part, des classes que l'on saura mieux discerner que d'autres lorsque (K > 2).

Le carré de la distance de Mahalanobis entre deux classes y_1 et y_2 s'écrit :

$$D_M^2(y_1, y_2) = (\bar{x}_1 - \bar{x}_2)W^{-1}(\bar{x}_1 - \bar{x}_2)^T$$

Exemple DATA 1

Prenons l'exemple DATA 1 pour calculer la distance entre les 2 moyennes conditionnelles, que nous pouvons matérialiser dans le plan (Figure 19).

```
#moyennes conditionnelles
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333

#ecart entre les centres de classes
ecart <- matrix(mb_k[1,] - mb_k[2,],nrow=1,ncol=2,byrow=TRUE)
print(ecart)

##           [,1]     [,2]
## [1,] -4.166667 1.866667

#carré de la distance de Mahalanobis
DM <- ecart %*% invW %*% t(ecart)
print(DM)

##          [,1]
## [1,] 9.771208

#graphique des points avec les centres de classes
plot(X$X1,X$X2,col=c("skyblue3","green")[y], pch = 19, main="Données DATA 1")
lines(xb_k$X1,xb_k$X2,lty=2,col="gray")
```

```
points(xb_k$X1,xb_k$X2,col=c("blue","green4"),pch=17,cex=1.5)
text(6,5,round(DM,2),col="red")
```

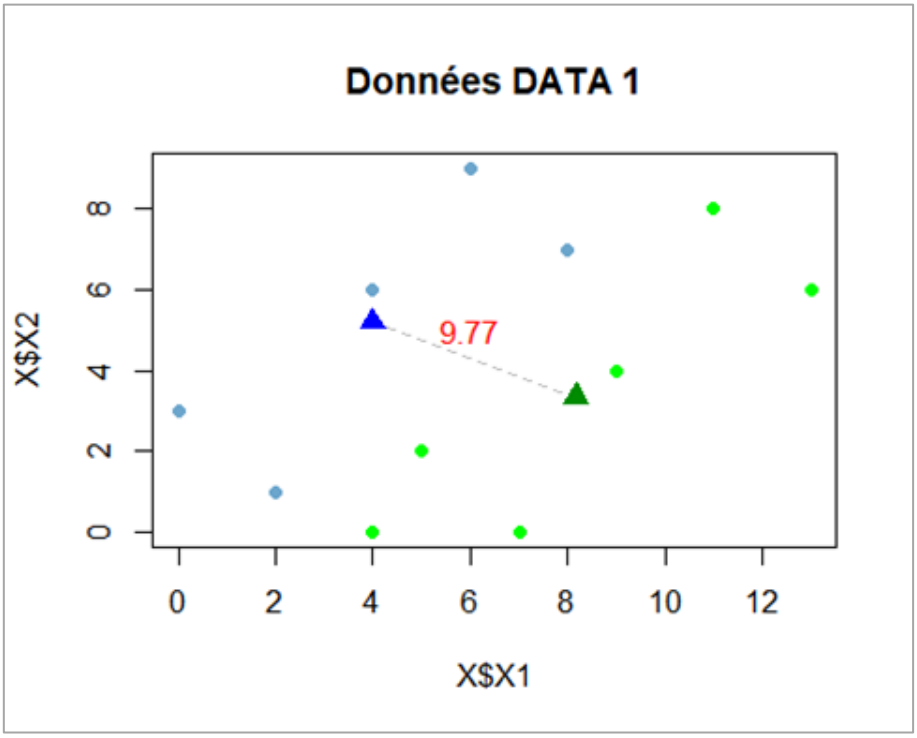


Figure 19 – Carré de la distance de Mahalanobis entre les centres de classes – DATA 1

D_M^2 = 9.77. Apprécier l'intensité de l'écartement avec une valeur brute comme cela n'est pas vraiment évident. D'où l'intérêt de passer par un test statistique que nous verrons dans la section suivante (section 2.2).

Distance au carré généralisé – PROC DISCRIM / SAS. Avant cela, penchons-nous un instant sur une des sorties de SAS / PROC DISCRIM qui m'a beaucoup intrigué. Le résultat ci-dessus équivaut à celui que l'on obtient dans le tableau « Distance au carré à Groupe ». Mais il y en a un second intitulé « Distance au carré généralisée », qui n'est pas symétrique et dont la diagonale n'est pas nulle (Figure 20).

Distance au carré généralisée à Groupe		
De Groupe	g1	g2
g1	1.57691	10.98348
g2	11.34812	1.21227

Figure 20 – SAS / PROC DISCRIM – Distance au carré généralisé – DATA 1

Après investigations, il apparaît que l'outil utilise la formule présentée plus haut pour le calcul de la distance aux centres de classes ($D(y_k,X)$ – section 1.7), soit :

$$D_G^2(y_1/\ y_2) = (\bar{x}_1 - \bar{x}_2)W^{-1}(\bar{x}_1 - \bar{x}_2)^T - 2 \times \ln \hat{\pi}_1$$

Il est dès lors facile de reproduire cette matrice sous R.

```
#prévalence estimée des classes
print(pi_k)

## y
##        g1        g2
## 0.4545455 0.5454545

#préparation de la structure
MDG <- matrix(0,nrow=2,ncol=2)

#diagonale matrice de la distance au carré généralisée
diag(MDG) <- -2.0 * log(pi_k)

#g1 (en colonne) / g2 (en ligne)
MDG[2,1] <- DM - 2.0 * log(pi_k[1])

#g2 (en colonne) / g1 (en ligne)
MDG[1,2] <- DM - 2.0 * log(pi_k[2])

#affichage
print(MDG)

##           [,1]      [,2]
## [1,]  1.576915 10.983480
## [2,] 11.348123  1.212272
```

2.2 Evaluation globale – Test MANOVA

La MANOVA (multivariate analysis of variance) est la généralisation multidimensionnelle ($p \geq 2$) de l'analyse de variance (ANOVA). Nous n'avons qu'un seul facteur ici, la variable cible, dont les modalités définissent les groupes. Lorsqu'ils sont au nombre de ($K = 2$), nous pouvons faire le parallèle avec le test de Student de comparaison de moyennes. La configuration est on ne peut plus favorable puisque, selon les hypothèses fondatrices de l'analyse discriminante, les échantillons sont indépendants, compatibles avec la loi normale et homoscédastiques.

2.2.1 T2 de Hotelling pour ($K = 2$)

Lorsque ($K = 2$), nous testons formellement :

$$\begin{cases} H_0 : \mu_1 = \mu_2 \\ H_1 : \mu_1 \neq \mu_2 \end{cases}$$

μ_1 et μ_2 représentent les barycentres conditionnels.

La statistique de test, dite T^2 de Hotelling (Rakotomalala, 2010 ; section 6.1), repose sur la distance carrée de Mahalanobis entre les moyennes conditionnelles. Le lien avec l'analyse discriminante est manifeste. Elle s'écrit :

$$T^2 = \frac{n_1 n_2}{n} D_M^2$$

Sous l'hypothèse nulle, la statistique de test :

$$F = \frac{n-p-1}{p(n-2)} T^2$$

Suit une loi de Fisher à (p, n-p-1) degrés de liberté.

Les moyennes conditionnelles ne sont pas statistiquement discernables si les données ne conduisent pas au rejet de l'hypothèse nulle. Il est impossible de conduire une analyse discriminante performante. A contrario, plus elles sont écartées, avec toujours l'hypothèse d'une distribution convexe autour des moyennes, plus efficace sera le modèle prédictif.

Exemple DATA 1

Voyons si le test statistique confirme l'impression visuelle ci-dessus qui laisse à penser que les centres de classes sont manifestement distincts (Figure 19).

```
#effectifs par groupe
print(n_k)

## y
## g1 g2
##  5  6

#carré de la distance de Mahalanobis entre les classes
print(DM)

##          [,1]
## [1,] 9.771208

#calcul du T2 de Hotelling
T2 <- (n_k[1]*n_k[2])/(n) * DM
print(T2)

##          [,1]
## [1,] 26.64875
```

```r
#stat. de test
FT2 <- (n - p - 1) / (p * (n-2)) * T2
print(FT2)

##          [,1]
## [1,] 11.84389

#p-value -- loi de Fisher
pvalue <- pf(FT2,df1 = p, df2 = n - p -1, lower.tail= FALSE)
print(pvalue)

##             [,1]
## [1,] 0.004062495
```

Malgré le faible effectif, la différence est significative à 5%. Nous le savions déjà mais c'est mieux quand les procédures statistiques le confirment.

2.2.2 Lambda de Wilks pour (K ≥ 2)

Pour la généralisation à (K ≥ 2), le test d'hypothèses s'écrit :

$$\begin{cases} H_0 : \mu_1 = \mu_2 = \cdots = \mu_K \\ H_1 : \textit{deux au moins des vecteurs moyennes sont différents} \end{cases}$$

La statistique privilégiée est le Λ de Wilks (Rakotomalala, 2010 ; section 7.1). Elle est constituée par le rapport entre les déterminants des estimateurs biaisés (divisés par n et non pas par les degrés de liberté, voir le détail dans le code R ci-dessous) des matrices de variance covariance intra-classes et totales.

$$\Lambda = \frac{\det(W_b)}{\det(V_b)}$$

Où :

$$W_b = \frac{n-K}{n} W$$

$$V_b = \frac{n-1}{n} COV(X) = \frac{n-1}{n} V$$

Concrètement, il s'agit d'opposer le volume (surface si p = 2) moyen des sous-nuages avec le volume global de l'ensemble des points. Plus faible est le ratio, plus écartés sont les sous-nuages, induisant une dispersion totale importante.

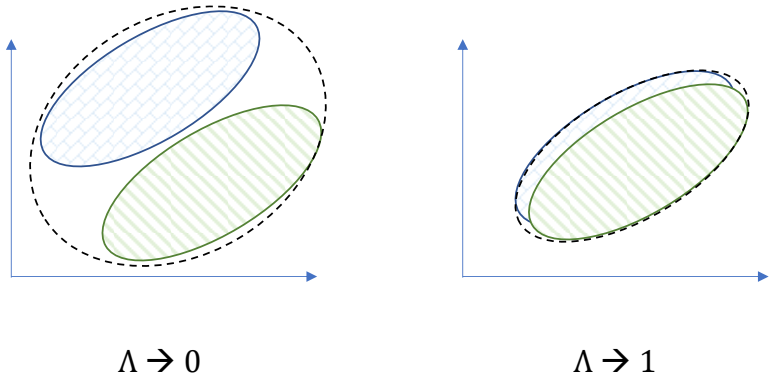


Figure 21 – Configurations opposées pour ($\Lambda \rightarrow 0$) et ($\Lambda \rightarrow 1$)

Nous essayons de résumer ces idées à l'aide deux graphiques (K = 2, p = 2) (Figure 21) :

- Dans celui de gauche, le volume moyen des sous-nuages bleus et verts est nettement plus faible par rapport au volume global du nuage (noir pointillés) : $\Lambda \rightarrow 0$. Les groupes sont bien discernables.
- Dans celui de droite, les sous-nuages sont quasiment confondus. Le volume moyen des sous-nuages est quasiment identique au volume global : $\Lambda \rightarrow 1$.
- Nous comprenons par ailleurs que $(0 \leq \Lambda \leq 1)$.

Deux transformations permettent de définir les régions critiques avec des lois connues.

Transformation de Bartlett. Elle est moins précise que la suivante sur les petits effectifs. Lorsque n est grand en revanche, elle suffit amplement. La statistique de test s'écrit :

$$\chi^2 = -\left(n - 1 - \frac{p + K}{2}\right) \times \ln \Lambda$$

Sous Ho, elle suit une loi du χ^2 à $[p \times (K - 1)]$ degrés de liberté.

Réalisons les calculs pour DATA 1.

```
#covariance totale
V <- cov(X)
print(V)

##           X1        X2
## X1 14.818182  5.945455
## X2  5.945455 10.363636
```

```r
#matrice intra-classe
print(W)

##           X1        X2
## X1 11.203704  8.962963
## X2  8.962963 10.459259

#version biaisées des matrices
Vb <- (n-1)/n * V
Wb <- (n-K)/n * W


#lambda de Wilks
lambda = det(Wb) / det(Vb)
print(lambda)

## [1] 0.2524633

#transformation de Bartlett
LB <- -(n - 1 - (p + K)/2) * log(lambda)
print(LB)

## [1] 11.01192

#probabilité critique
print(pchisq(LB,df = p * (K - 1), lower.tail = FALSE))

## [1] 0.004062495
```

Nous obtenons ($\Lambda = 0.2524633$). La décision du test est cohérente avec celle basée sur le T2 de Hotelling, spécifique au cas (K = 2) classes (section 2.2.1).

Transformation de Rao. Elle est à privilégier sur les petits effectifs. Mais la formule de la statistique de test est pour le moins tarabiscotée (TUTO 2, section 2.4.3, pour l'implémentation sous Python) :

$$F = \left(\frac{1 - \Lambda^{\frac{1}{B}}}{\Lambda^{\frac{1}{B}}}\right) \times \left(\frac{A \times B - C}{p \times (K-1)}\right)$$

Où

$$A = n - K - \frac{p - K + 2}{2}$$

$$B = \begin{cases} \sqrt{\dfrac{p^2(K-1)^2 - 4}{p^2 + (K-1)^2 - 5}} & \text{si } (p^2 + (K-1)^2 - 5 > 0) \\ 1 & \text{sinon} \end{cases}$$

$$C = \frac{p(K-1) - 2}{2}$$

Sous Ho, elle suit une loi de Fisher à $[p(K-1), AB-C]$ degrés de liberté.

Réalisons de nouveau les calculs pour DATA 1.

```
#** transformation de RAO **

#A
A <- n - K - (p - K + 2)/2

#B
B <- p^2 + (K - 1)^2 - 5
B <- ifelse(B > 0, sqrt((p^2 * (K - 1)^2 - 4)/(B)), 1)

#C
C <- (p * (K - 1) - 2)/2

#F de RAO
FRAO <- ((1 - lambda^(1/B))/(lambda^(1/B))) * ((A * B - C)/(p * (K - 1)))
print(FRAO)

## [1] 11.84389

# ddl1
print(p * (K - 1))

## [1] 2

#ddl2
print(A * B - C)

## [1] 8

#p-value
print(pf(FRAO, df1 = p * (K - 1), df2 = A * B - C, lower.tail = FALSE))

## [1] 0.004062495
```

Nous sommes exactement sur les mêmes valeurs que pour le T^2 de Hotelling : même statistique de test, mêmes degrés de liberté, et par conséquent même probabilité critique.

2.2.3 Autres tests

D'autres tests sont présents dans les logiciels (ex. PROC DISCRIM DE SAS) : Trace de Pillai, Trace de Hotelling-Lawley, la plus grande valeur propre de Roy. Ils procèdent de la même logique que le lambda de Wilks et aboutissent très souvent à des conclusions similaires (Rakotomalala, 2010, section 7.1.4). Dans notre cas particulier de DATA 1, nous avons carrément des valeurs F identiques (Figure 22). Ce type de résultat reste néanmoins exceptionnel.

Statistiques multivariées et statistique F exacte					
S=1 M=0 N=3					
Statistique	Valeur	Valeur F	DDL num.	DDL den.	Pr > F
Wilks' Lambda	0.25246327	11.84	2	8	0.0041
Pillai's Trace	0.74753673	11.84	2	8	0.0041
Hotelling-Lawley Trace	2.96097224	11.84	2	8	0.0041
Roy's Greatest Root	2.96097224	11.84	2	8	0.0041

Figure 22 – Sortie MANOVA de PROC DISCRIM – DATA 1

2.3 Pertinence d'une variable

La pertinence d'une variable dans le modèle se conçoit à l'aune de sa contribution dans la distance entre les centres de classes. Dans la Figure 23, nous étudions deux cas qui permettent de comprendre le mécanisme. Pour (A), c'est la synergie des deux variables X1 et X2 qui détermine l'écartement. Pour (B), seule X2 joue un rôle. Il faut que le test statistique puisse rendre compte de ces différentes situations.

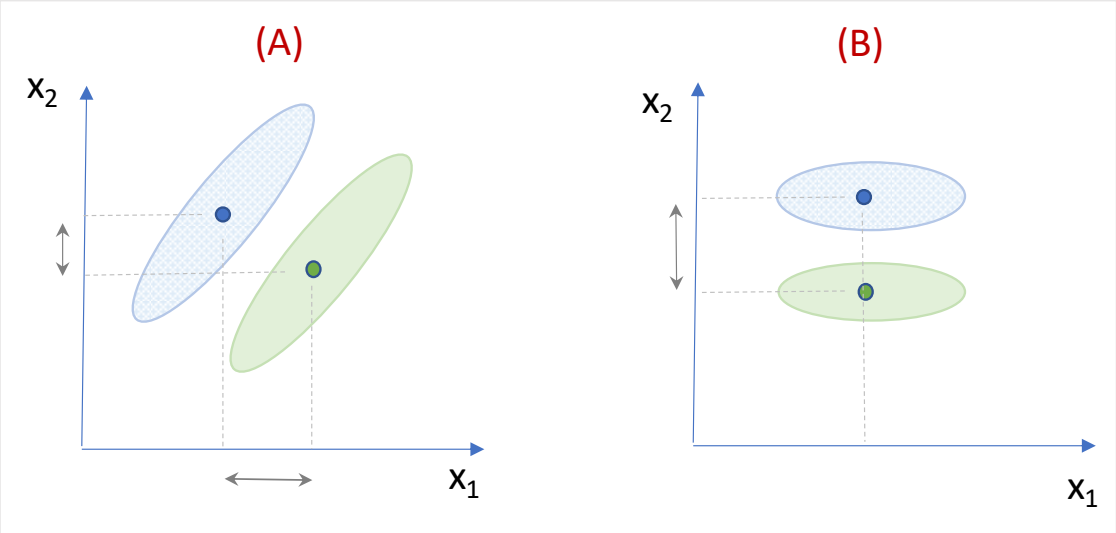


Figure 23 – Contribution des variables dans l'écartement des centres de classes

En pratique, le test repose sur la confrontation des lambda de Wilks avec et sans la variable dont on cherche à établir la pertinence. Les variables ne sont pas appréciées de manière indépendante. La procédure tient compte du rôle des autres descripteurs et de la forme des nuages de points par l'entremise des matrices de variances W (intra-classe) et V (totale).

Le lambda de Wilks se dégrade toujours lorsque nous retirons une variable (c.-à-d. sa valeur augmente, $\Lambda_{p-1} \geq \Lambda_p$), on souhaite savoir si cette évolution est suffisamment forte pour être significative. On comprend mieux la notion de « F d'exclusion » évoquée en introduction de ce chapitre. La statistique de test est définie comme suit :

$$F = \frac{n-K-p+1}{K-1}\left(\frac{\Lambda_{p-1}}{\Lambda_p} - 1\right)$$

Une forte évolution est symptomatique du rôle important de la variable. La région critique correspond aux valeurs élevées de F qui suit une loi de Fisher à (K-1, n-K-p+1) degrés de liberté.

Enfin, cet indicateur permet d'établir une hiérarchie des descripteurs selon leur importance. Il sera mis à contribution dans les algorithmes de sélection de variables (chapitre 3).

Exemple DATA 1

Nous n'avons que (p = 2) variables pour DATA 1. Lorsque l'on retire un des descripteurs, le calcul du lambda de Wilks se résume à un ratio entre les variances intra-classes et totales. Les manipulations sont grandement simplifiées. Nous montrons les calculs pour l'évaluation de X1. Réitérer les calculs pour X2 ne pose aucune difficulté.

```
#lambda global avec 2 variables
print(lambda)

## [1] 0.2524633

#matrice W utilisée pour la lambada
print(Wb)

##          X1       X2
## X1 9.166667 7.333333
## X2 7.333333 8.557576

#matrice V utilisée pour la soca
print(Vb)

##           X1       X2
## X1 13.471074 5.404959
## X2  5.404959 9.421488

#lambda si on retirer la variable x1
lambda_sans_x1 <- Wb[2,2] / Vb[2,2]
print(lambda_sans_x1)

## [1] 0.9083041
```

```r
#stat. de test
f1 <- (n-K-p+1)/(K-1)*(lambda_sans_x1/lambda - 1)
print(f1)

## [1] 20.78214

#p-value
print(pf(f1, df1 = K-1, df2 = n-K-p+1, lower.tail = FALSE))

## [1] 0.001852965
```

Le retrait de la variable X1 entraîne une très forte dégradation du Λ dans l'espace de représentation incluant les autres variables restantes. L'écart entre les deux lambda { $\Lambda_{\{x1,\ x2\}}$ = 0.2524633 et $\Lambda_{\{X2\}}$ = 0.9083041]), sous forme de ratio, est significatif. Nous concluons que X1 joue un rôle décisif dans le modèle.

Voici les résultats de TANAGRA sur DATA 1 à titre de comparaison (Figure 24).

LDA Summary

Classification functions			Statistical Evaluation			
Attribute	g1	g2	Wilks L.	Partial L.	F(1,8)	p-value
X1	-0.129	1.507	0.908304	0.277950	20.782140	0.001853
X2	0.608	-0.973	0.680470	0.371013	13.562590	0.006195
constant	-2.111	-5.139	-			

Figure 24 – Contribution des variables – DATA 1 – TANAGRA

Dans la partie « Statistical Evaluation » de TANAGRA :

- « Wilks L. » correspond à la valeur du lambda de Wilks du modèle (Λ_{p-1}) lorsque l'on retire la variable incriminée.
- « Partial .L » est le ratio (Λ_p / Λ_{p-1}).
- « F » est la statistique F telle que définie ci-dessus, avec les degrés de liberté : (K – 1 = 2 – 1 = 1) et (n – K – p + 1 = 11 – 2 – 2 + 1 = 8).
- « p-value » est la probabilité critique. Si elle est inférieure au risque α que l'on s'est choisi, on peut conclure à la pertinence de la variable dans le modèle.

Dans notre exemple DATA 1 (Figure 19), X1 et X2 contribuent significativement au risque 5%, X1 un peu plus que X2 puisqu'elle entraîne une dégradation plus marquée du Λ.

Exemple DATA 2

Pour (p > 2), les opérations semblent complexes puisqu'il y a p ratios de déterminants à comptabiliser. En réalité les matrices W et V sont calculées une fois pour toutes sur l'ensemble des variables. Il nous reste par la suite à extraire au fur et à mesure les (p) sous-matrices de dimension (p-1, p-1) pour calculer les déterminants. Avec une implémentation efficace de ce dernier, comme la fonction det() de R par exemple, l'affaire est vite bouclée.

Nous adaptons ici – sous forme de fonction - le code rédigé pour le tutoriel consacré à l'analyse discriminante sous R (TUTO 2, sections 3.2 et 5.2). Le calcul des matrices de variance covariance intra-classe et totale constitue le point de départ, puis vient le lambda de Wilks. La fonction test.retrait() permet de mesurer la contribution de chaque variable indicée par son numéro j via le différentiel de lambda. Il reste alors à l'appliquer (sapply) sur chaque variable et présenter les résultats de manière compréhensible.

```
#mat. de covar. totale
Vb <- (n-1)/n * cov(XTrain)

#mat. de covar. intra-classe
Wb <- (n-K)/n * W

#lambda avec la totalité des variables
lambda <- det(Wb)/det(Vb)
print(lambda)

## [1] 0.06671324

#fonction pour le test du retrait d'une variable numéro j
#SW : matrice de covariance intra
#ST : matrice de covariance totale
#N : nombre d'observations
#K : nombre de classes
#prevLW : valeur du lambda avant le retrait
test.retrait <- function(j,SW,ST,N,K,prevLW){
  #nombre de variables à cette étape
  nbVar <- ncol(SW)
  #lamdba
  lambada <- det(SW[-j,-j])/det(ST[-j,-j])
  #ddl
  ddlNum <- K - 1
  ddlDenom <- N - K - nbVar + 1
  #f
  f <- ddlDenom / ddlNum * (lambada / prevLW - 1)
  #p-value
  pvalue <- pf(f,ddlNum,ddlDenom,lower.tail=FALSE)
  #renvoyer sous forme de vecteur
  return(c(lambada,f,pvalue))
}
```

```r
#tester chaque variable allant de n° 1 à n°p
eval_var <- t(sapply(1:p,test.retrait,SW=Wb,ST=Vb,N=n,K=K,prevLW=lambda))

#noms des variables
rownames(eval_var) <- colnames(XTrain)

#noms des classes
colnames(eval_var) <- c("Wilks L.","F(2,42)","p-value")

#affichage
print(round(eval_var,5))

##      Wilks L.  F(2,42) p-value
## MEOH  0.11797 16.13607 0.00001
## ACET  0.07415  2.34196 0.10857
## BU1   0.08418  5.49926 0.00756
## BU2   0.09570  9.12300 0.00051
## ISOP  0.07231  1.76176 0.18420
## MEPR  0.08780  6.63695 0.00313
## PRO1  0.09240  8.08434 0.00107
## ACAL  0.07588  2.88670 0.06688
```

A titre de comparaison, voici les résultats obtenus sous TANAGRA (Figure 25).

Classification functions				Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,42)	p-value
MEOH	0.003428	0.029028	0.033390	0.11798	0.56549	16.13607	0.00001
ACET	0.006390	0.016413	0.007513	0.07415	0.89967	2.34196	0.10857
BU1	-0.063681	0.405390	0.318047	0.08418	0.79248	5.49926	0.00756
BU2	-0.000883	0.071352	0.114993	0.09570	0.69714	9.12300	0.00051
ISOP	0.023082	0.029763	-0.008486	0.07231	0.92260	1.76176	0.18420
MEPR	0.037494	-0.128942	0.061780	0.08780	0.75985	6.63695	0.00313
PRO1	0.001971	-0.005413	-0.008318	0.09240	0.72204	8.08434	0.00107
ACAL	0.066184	-0.226424	-0.130332	0.07588	0.87915	2.88670	0.06689
constant	-5.016453	-18.840686	-24.764879	-			

Figure 25 – Contribution des variables – DATA 2 / TRAIN – TANAGRA

Les descripteurs qui contribuent significativement à 5% sont : MEOH, BU1, BU2, MEPR, PRO1. Mais nous ne pouvons pas pour autant retirer d'emblée les autres variables. Il faut soit effectuer un retrait pas à pas (section 4.2), soit effectuer un test qui permet de prendre en compte les interactions entre les variables puisqu'elles ne sont pas deux à deux indépendantes (section 2.4).

2.4 Pertinence d'un groupe de variables

Le test de la pertinence de q ($q \leq p$) variables pris en bloc résulte également du ratio entre les lambda de Wilks avec p et (p-q) variables. Il doit correspondre à une « vraie » généralisation c.-à-d. la statistique de test doit être équivalente à celle du test global lorsque (q = p), et à celle

du test individuel lorsque (q = 1). Pour mesurer la contribution de « q » variables, elle s'écrit [Timm, N.H., « Applied Multivariate Analysis », Springer, 2002 ; équations 3.5.3 (page 102) et 7.4.7 (page 437)] :

$$F = \left(\frac{1 - \Lambda_{partial}^{\frac{1}{B}}}{\Lambda_{partial}^{\frac{1}{B}}} \right) \times \left(\frac{A \times B - C}{q \times (K-1)} \right)$$

Où

$$\Lambda_{partial} = \frac{\Lambda_p}{\Lambda_{p-q}}$$

On sait que ($\Lambda_{\text{partial}} \leq 1$)

Et

$$A = n - K - (p - q) - \frac{q - K + 2}{2}$$

$$B = \begin{cases} \sqrt{\dfrac{q^2(K-1)^2 - 4}{q^2 + (K-1)^2 - 5}} & \text{si } (q^2 + (K-1)^2 - 5 > 0) \\ 1 & \text{sinon} \end{cases}$$

$$C = \frac{q(K-1) - 2}{2}$$

Sous Ho, F suit une loi de Fisher à [$q\ (K-1)\ , AB - C$] degrés de liberté.

<u>Remarque :</u> Nous disposons à ce stade d'une « vraie » généralisation de la transformation de Rao à partir du lambda partiel. Pour (q = p), on retrouve aisément le test de significativité globale ci-dessus (section 2.2.2) puisque ($\Lambda_{\emptyset} = 1$) c.-à-d. il n'est pas possible de discerner les barycentres dans un espace sans variables. Et lorsque (q = 1), nous retrouvons la formule du test individuel (section 2.3) (Timm, 2002 ; page 438).

<u>Exemple DATA 2 (Tests globaux et individuels)</u>

Ecrivons une fonction R qui correspond à la généralisation ci-dessus. Puis vérifions qu'elle permet d'englober les test globaux et individuels étudiés dans les deux sections précédentes

(sections 2.2.2 et 2.3). La fonction prend en entrée le lambda partiel c.-à-d. le ratio du lambda à p variables et lambda à (p-q) variables (LPartial) ; le nombre initial de variables (p) ; le nombre de variables à évaluer (q) ; le nombre de classes (K) ; et la taille de l'échantillon de travail (n). Nous la lançons ensuite pour le test global et pour éprouver la contribution de la variable MEOH.

```
#F de Rao - généralisation
FRAO <- function(LPartial, p, q, K, n){
  #A
  A <- n-K-(p-q)-(q-K+2)/2
  #B
  B <- q^2+(K-1)^2-5
  B <- ifelse(B > 0, sqrt((q^2*(K-1)^2-4)/B), 1)
  #C
  C <- (q*(K-1)-2)/2
  #
  FTest <- (1-LPartial^(1/B))/(LPartial^(1/B))*((A*B-C)/(q*(K-1)))
  #ddl1
  ddl1 <- q*(K-1)
  #ddl2
  ddl2 <- A*B-C
  #pvalue
  pval <- pf(FTest,ddl1,ddl2,lower.tail = FALSE)
  #résultats
  return(list(FStat=FTest,df1=ddl1,df2=ddl2,pvalue=pval))
}

#rappel de la valeur du lambda global
print(lambda)

## [1] 0.06671324

#** vérification pour le test global **
#lambda = 1 lorsque espace à zéro variables
print(FRAO(lambda/1,p=p,q=p,K=K,n=n))

## $FStat
## [1] 15.07606
##
## $df1
## [1] 16
##
## $df2
## [1] 84
##
## $pvalue
## [1] 2.286587e-18

#** vérification pour retrait de MEOH, première variable **

#valeur du lambda lorsque la variable MEOH est exclue
lambda_sans_MEOH <- det(Wb[-1,-1])/det(Vb[-1,-1])
print(lambda_sans_MEOH)

## [1] 0.1179746
```

```r
#lambda partiel
print(lambda/lambda_sans_MEOH)

## [1] 0.565488

#utilisation de FRAO
print(FRAO(lambda/lambda_sans_MEOH,p=p,q=1,K=K,n=n))

## $FStat
## [1] 16.13607
##
## $df1
## [1] 2
##
## $df2
## [1] 42
##
## $pvalue
## [1] 6.322533e-06
```

Les résultats sont tout à fait corrects. A titre de comparaison, voici les résultats fournis par TANAGRA sur les mêmes données (Figure 26).

MANOVA

Stat	Value	p-value
Wilks' Lambda	**0.0667**	-
Bartlett -- C(16)	123.1845	0
Rao -- F(16, 84)	15.0761	0

LDA Summary

Classification functions				Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,42)	p-value
MEOH	0.003428	0.029028	0.033390	0.11798	0.56549	16.13607	0.000006
ACET	0.006390	0.016413	0.007513	0.07415	0.89967	2.34196	0.10857
BU1	-0.063681	0.405390	0.318047	0.08418	0.79248	5.49926	0.00756
BU2	-0.000883	0.071352	0.114993	0.09570	0.69714	9.12300	0.00051
ISOP	0.023082	0.029763	-0.008486	0.07231	0.92260	1.76176	0.18420
MEPR	0.037494	-0.128942	0.061780	0.08780	0.75985	6.63695	0.00313
PRO1	0.001971	-0.005413	-0.008318	0.09240	0.72204	8.08434	0.00107
ACAL	0.066184	-0.226424	-0.130332	0.07588	0.87915	2.88670	0.06689
constant	-5.016453	-18.840686	-24.764879	-			

Figure 26 – Test global et pertinence de MEOH – DATA 2 / TRAIN – TANAGRA

Exemple DATA 2 (Test d'un bloc de variables)

Pour DATA 2, testons maintenant l'exclusion simultanée des variables {ACET, ISOP, ACAL} c.-à-.d les variables n°2, 5 et 8.

```r
#vérification des n° de variables
str(XTrain)
```

```
## 'data.frame':    52 obs. of  8 variables:
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...
```

```r
#lambda sans ACET, ISOP et ACAL
lamdba_sans_les_3 <- det(Wb[-c(2,5,8),-c(2,5,8)])/det(Vb[-c(2,5,8),-c(2,5,8)])
print(lamdba_sans_les_3)
```

```
## [1] 0.08648809
```

```r
#lambda partiel
print(lambda/lamdba_sans_les_3)
```

```
## [1] 0.7713576
```

```r
#tester la réalité de leur contribution
print(FRAO(LPartial=lambda/lamdba_sans_les_3,p=p,q=3,K=K,n=n))
```

```
## $FStat
## [1] 1.940435
##
## $df1
## [1] 6
##
## $df2
## [1] 84
##
## $pvalue
## [1] 0.08361076
```

Nous obtenons F = 1.94 avec une p-value de 0.0803. Au risque 5%, nous pouvons conclure que ces 3 variables prises en bloc ne contribuent pas significativement à l'écartement des centres de classes conditionnels.

Pour conclure ce chapitre, nous dirons qu'au-delà de leur utilité intrinsèque, ces outils joueront un rôle central dans la sélection de variables (chapitre 4).

3 Traitement des descripteurs qualitatifs

L'utilisation des descripteurs qualitatifs en analyse discriminante prédictive paraît être une hérésie tant leurs propriétés ne cadrent absolument pas avec l'hypothèse fondatrice : la multinormalité des distributions conditionnelles. Et pourtant ça marche ! (Et pourtant elle tourne…). Parce que la robustesse est une notion clé en statistique. Une méthode s'applique même à la marge de ses présupposés de départ. Le tout est de délimiter jusqu'où on peut aller.

Nous étudierons deux approches dans ce chapitre. La première, très simple, consiste à coder en indicatrices 0/1 (dummy variable) les modalités des variables explicatives catégorielles (section 3.2). La seconde, plus sophistiquée, consiste à se projeter dans un espace continu avant de réaliser une analyse discriminante sur les variables latentes créées. La technique est ainsi placée dans une configuration qui lui sied mieux en théorie (section 3.3). Mais, auparavant, introduisons un nouveau jeu de données que nous utiliserons tout au long de ce chapitre.

3.1 Données « Votes au congrès »

Les données « Votes au congrès » recense les votes de (n = 435) parlementaires américains identifiés selon leur appartenance politique (Y = « group » avec K = 2 valeurs possibles {'republican', 'democrat'}) sur différents thèmes en 1984. Deux descripteurs ont été retenus dans ce chapitre : ADOP_BUDGET (adoption du budget) et EDU_SPENDING (engager des dépenses pour l'éducation). Les valeurs possibles sont « y » (yea), « n » (nay) et « u » (unknow). Voici le descriptif du sens réel de ces différentes modalités : « … nine different types of votes : voted for, paired for, and announced for (these three simplified to yea) ; voted

against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition). »

Voici les 10 premières lignes du jeu de données.

group	adop_budget	edu_spending
republican	n	y
republican	n	y
democrat	y	n
democrat	y	n
democrat	y	u
democrat	y	n
democrat	n	n
republican	n	n
republican	n	y
democrat	y	n

Figure 27 – Premières lignes des données "VOTE"

Pour bien comprendre la teneur des résultats qui seront présentés dans les deux prochaines sections, nous explorons succinctement les variables qui composent la base.

3.1.1 Statistiques univariées

Quelques statistiques descriptives permettent d'en apprécier la teneur.

- La distribution de fréquence de la cible « group ». Les démocrates sont majoritaires dans l'assemblée.

Étiquettes de lignes	Nombre de group	Nombre de group
democrat	267	61.38%
republican	168	38.62%
Total général	**435**	**100.00%**

Figure 28 – Distribution de « group »

- Celle de « ADOP_BUDGET ». La modalité « u » est relativement rare.

Étiquettes de lignes	Nombre de adop_	Nombre de adop_
n	171	39.31%
u	11	2.53%
y	253	58.16%
Total général	**435**	**100.00%**

Figure 29 – Distribution de « adop_budget »

- Et de « EDU_SPENDING ». Idem, « u » est peu fréquente.

Étiquettes de lignes	Nombre de edu_s	Nombre de edu_s
n	233	53.56%
u	31	7.13%
y	171	39.31%
Total général	**435**	**100.00%**

Figure 30 – Distribution de « edu_spending »

3.1.2 Relations entre les variables

Passons aux statistiques bivariées.

- Relation entre « adop_budget » (en ligne) et « edu_spending » (en colonne). Nous constatons que les personnes qui ont voté « n » sur le premier thème a plutôt voté « y » au second, et inversement.

Nombre de adop_budget	Étiquettes de			
Étiquettes de lignes	n	u	y	Total général
n	28	10	133	171
u	4	4	3	11
y	201	17	35	253
Total général	**233**	**31**	**171**	**435**

Figure 31 – Tableau croisé « adop_budget » vs. « edu_spending »

- Relation entre « group » et « adop_budget » sous la forme P(adop_budget / group). Les démocrates ont plutôt voté « y ».

Nombre de group	Étiquettes de			
Étiquettes de lignes	n	u	y	Total général
democrat	10.86%	2.62%	86.52%	100.00%
republican	84.52%	2.38%	13.10%	100.00%
Total général	**39.31%**	**2.53%**	**58.16%**	**100.00%**

Figure 32 – P(adop_budget / group)

- Relation entre « group » et « edu_spending ». Les démocrates ont plutôt voté « n ».

Nombre de group	Étiquettes de			
Étiquettes de lignes	n	u	y	Total général
democrat	79.78%	6.74%	13.48%	100.00%
republican	11.90%	7.74%	80.36%	100.00%
Total général	**53.56%**	**7.13%**	**39.31%**	**100.00%**

Figure 33 – P(edu_spending / group)

Au regard de ces premiers éléments, on aurait tendance à dire que voter « y » à « adop_budget » et « n » à « edu_spending » désignerait plutôt les démocrates. Ce serait l'inverse pour les républicains. Voyons ce que va nous annoncer l'analyse discriminante.

3.2 Codage 0/1 des descripteurs

3.2.1 Principe des dummy variables

Comme en régression (Rakotomalala R., « Pratique de la régression linéaire multiple – Diagnostic et sélection de variables », version 2.1, mai 2015 ; chapitre 4), la solution la plus simple pour l'appréhension des descripteurs catégoriels consiste à utiliser un codage disjonctif. Aux variables sont substituées des indicatrices (dummy variable en anglais ; « variables muettes » en français, mais le terme n'est pas très heureux je trouve) associées aux catégories. Concrètement, si X possède M modalités, nous créons (M-1) variables 0/1 précisant la présence ou l'absence des catégories.

(M-1) et non pas M indicatrices pour deux raisons principalement :

- M indicatrices 0/1 à partir d'une variable X engendre de la colinéarité c.-à-d. la somme des colonnes correspondantes est systématiquement égale à 1. Ce qui posera problème dans les calculs matriciels inhérents à la méthode.
- Il n'y a pas de perte d'informations puisqu'il est possible de reconstituer la variable initiale avec les indicatrices. Les étudiants comprennent très bien cet aspect lorsque la variable est binaire (ex. sexe avec deux modalités {homme, femme} est codée en sexe_homme avec la valeur 1/0, lorsque sex_homme = 0, on en déduit que l'individu est une femme). Il me faut un peu plus expliquer lorsque (M > 2). Mais le principe est bien le même, puisque la somme des M indicatrices vaut toujours 1, la $M^{ème}$ indicatrice peut être déduite des (M – 1) autres.

Le choix de l'indicatrice exclue, qui correspond à ce qu'on appelle la « modalité de référence », n'impacte pas la qualité prédictive du modèle. Il pèse en revanche sur la lecture des résultats produits par l'analyse. Les coefficients des indicatrices s'interprètent en écarts par rapport à la modalité exclue. La compréhension n'est déjà pas évidente dans la régression linéaire ou logistique où nous avons une seule équation à manipuler. Elle devient inextricable en analyse discriminante où nous devons de plus comparer les coefficients d'une fonction de classement à

l'autre. Seul le cas de la prédiction binaire (K = 2), qui permet de revenir à une fonction score unique (section 5.2), est relativement facile à appréhender, à l'instar de la régression.

Pour les données « VOTE », X1 (adop_budget) comprend M_1 = 3 modalités ('y',' n', 'u'), nous créons (M_1 – 1 = 2) indicatrices : « adop_n » (si adop_budget = 'n' alors adop_n = 1 sinon adop_n = 0) et « adop_u » (si adop_budget = 'u' alors adop_u = 1 sinon adop_u = 0). Nous adoptons le même principe pour X2 (edu_spending) qui possède également M2 = 3 modalités.

Voici les 10 premières lignes du nouveau tableau de données (Figure 34).

group	adop_budget	edu_spending	adop_n	adop_u	edu_n	edu_u
republican	n	y	1	0	0	0
republican	n	y	1	0	0	0
democrat	y	n	0	0	1	0
democrat	y	n	0	0	1	0
democrat	y	u	0	0	0	1
democrat	y	n	0	0	1	0
democrat	n	n	1	0	1	0
republican	n	n	1	0	1	0
republican	n	y	1	0	0	0
democrat	y	n	0	0	1	0

Figure 34 – Données « VOTE » avec les 4 indicatrices (adop_n, adop_u, edu_n, edu_u)

Notons que (adop_n, adop_u) permettent de reconstituer parfaitement (adop_budget). La valeur nulle simultanée (adop_n = 0, adop_u = 0) désigne sans ambiguïtés la modalité qui n'a pas été prise en compte dans le codage (adop_budget = 'y'). Les indicatrices permettent de reconstituer les variables initiales.

3.2.2 Inspection des quelques concepts statistiques de l'ADL

L'affaire ne semble pas évidente de prime abord. S'imaginer des nuages de points dans un espace de représentation défini par des variables catégorielles n'est pas intuitif. Les notions de moyennes et variance-covariance marginales et conditionnelles jouent un rôle central dans l'analyse discriminante linéaire. Voyons ce qu'elles évoquent lorsqu'elles sont calculées sur les indicatrices.

Les moyennes sont représentées dans le tableau ci-dessous (Figure 35) :

Étiquettes de lignes	Moyenne de ado	Moyenne de adop	Moyenne de ed	Moyenne de edu
democrat	0.1086	0.0262	0.7978	0.0674
republican	0.8452	0.0238	0.1190	0.0774
Total général	**0.3931**	**0.0253**	**0.5356**	**0.0713**

Figure 35 – Moyennes marginales et conditionnelles des indicatrices – VOTE

Nous retrouvons les probabilités marginales (Figure 29) et conditionnelles (Figure 32) des variables « adop_budget ». Il en est de même pour « edu_spending » (Figure 30 et Figure 33). Matérialiser les écarts entre les vecteurs des moyennes conditionnelles s'apparente à ce qui se fait en régression logistique, sauf qu'on y calcule plutôt des ratios (odds-ratio) et que les probabilités sont calculée dans l'autre sens P(Y / X). Mais, nous retenons surtout que les moyennes sur indicatrices revêtent une signification statistique, nous ne manipulons pas des concepts totalement absurdes.

Voici la matrice variance-covariance totale (Figure 36).

Covariance totale (1/n)	adop_n	adop_u	edu_n	edu_u
adop_n	0.2386	-0.0099	-0.1462	-0.0050
adop_u	-0.0099	0.0246	-0.0043	0.0074
edu_n	-0.1462	-0.0043	0.2487	-0.0382
edu_u	-0.0050	0.0074	-0.0382	0.0662

Figure 36 – Matrice V pour VOTE

Nous observons des informations reconnaissables :

- Sur la diagonale nous avons les variances. Si f_m est la fréquence de la modalité « m », elle correspond à l'expression variance = f_m x (1 – f_m)
- (adop_budegt = n) est plutôt lié négativement avec (edu_spendig = n). Nous l'avions constaté dans le tableau croisant les deux explicatives (Figure 31).

La covariance de 2 indicatrices issues de la même variable est un concept mystérieux en revanche. Elles sont forcément liées négativement puisqu'elles ne peuvent pas prendre simultanément la valeur 1. Mais l'exploitation qui est faite de cette covariance dans l'analyse discriminante reste une question ouverte.

3.2.3 Résultats de l'analyse discriminante

Lancer l'analyse discriminante (group ~ adop_n + adop_u + edu_n + edu_u) sous TANAGRA produit les sorties suivantes (Figure 37).

MANOVA

Stat	Value	p-value
Wilks' Lambda	0.3764	-
Bartlett -- C(4)	421.0751	0
Rao -- F(4, 430)	178.0632	0

LDA Summary

Classification functions			Statistical Evaluation			
Attribute	republican	democrat	Wilks L.	Partial L.	F(1,430)	p-value
adop_n	9.194079	3.507947	0.525985	0.715704	170.80741	0.0000
adop_u	4.119102	2.285905	0.379091	0.993031	3.01772	0.0831
edu_n	3.792938	7.901733	0.451937	0.832967	86.22676	0.0000
edu_u	3.740122	5.484684	0.382077	0.985271	6.42802	0.0116
constant	-5.256489	-4.045261	-			

Figure 37 – Analyse discriminante linéaire – VOTE

Le système a parfaitement fonctionné avec des résultats viables avec un taux d'erreur de 12.41% (Figure 38), en resubstitution certes, mais avec une configuration où nous disposons de 100 fois plus d'observations que de descripteurs avec un classifieur linéaire, le risque de surapprentissage reste contenu.

Confusion matrix - Taux d'erreur : 12.41%			
	Classes prédites		
Classes observées	republican	democrat	Sum
republican	144	24	168
democrat	30	237	267
Sum	174	261	435

Figure 38 – Matrice de confusion en resubstitution – VOTE

La lecture des coefficients n'est pas facile (Figure 37). Il faut tenir compte des modalités de référence pour chaque variable (qui n'apparaissent pas dans les fonctions de classement), tout en opposant les classes à identifier. Il reste quand-même qu'une lecture rapide montre bien que les républicains ont plus tendance à voter « n » à « adop_budget » par rapport aux démocrates. C'est l'inverse pour le thème « edu_spending ».

Je conclurais cette section en disant que l'analyse discriminante sur indicatrices fonctionne très bien en pratique. Y compris lorsque nous avons un mélange de descripteurs quantitatifs et qualitatifs. Malgré que les conditions théoriques ne soient manifestement pas réunies, les nombreuses expérimentations que j'ai pu mener montrent que les performances prédictives sont tout à fait comparables à celles des autres approches destinées à l'appréhension des prédictives catégorielles.

3.3 La méthode DISQUAL

Due à Saporta (1975), DISQUAL (discrimination sur variables qualitatives) est une solution bien établie pour l'introduction des explicatives qualitatives dans l'analyse discriminante (Saporta 2006, section 18.4 ; Nakache et Confais, 2003 ; chapitre 2). La méthode repose sur l'enchaînement de plusieurs étapes :

1. Une analyse des correspondances multiples (ACM) est lancée sur les variables explicatives catégorielles. Les individus sont projetés dans un espace factoriel c.-à-d. ils sont décrits par des variables latentes, quantitatives. On ne tient pas compte de l'appartenance aux classes à ce stade.
2. Une analyse discriminante est réalisée où l'on prédit la variable cible à l'aide des variables synthétiques élaborées à l'étape précédente.
3. Nous exprimons les fonctions de classement définitives à partir des indicatrices des variables initiales.

La solution est élégante et cumule les avantages : nous plaçons l'analyse discriminante linéaire dans une configuration qui cadre mieux avec ses présupposés statistiques ; nous pouvons en moduler les propriétés de régularisation, et donc la robustesse au surapprentissage, en jouant sur le nombre de facteurs de l'ACM à retenir pour l'analyse discriminante ; toutes les modalités sont présentes dans la fonction de classement, nous n'avons plus à réaliser la gymnastique périlleuse qui consiste à interpréter les coefficients en tenant compte des modalités de référence qui n'apparaissent pas dans les fonctions de classement.

Nous nous basons sur les résultats de TANAGRA sur les données « VOTE » dans cette section. Le détail des manipulations est retracé dans un tutoriel, dans lequel nous détaillons également les manipulations sous Python (TUTO 5).

3.3.1 Analyse des correspondances multiples

L'analyse des correspondances multiples (ACM) est une technique factorielle adaptée au traitement des variables catégorielles. La description de la méthode dépasse le cadre de cet ouvrage. Le lecteur trouvera facilement des supports dédiés sur le web, j'y ai moi-même contribué (Rakotomalala R., « Analyse des correspondances multiples – Diapos », août 2013).

L'idée est de travailler à partir des indicatrices des variables, en incluant toutes les modalités cette fois-ci. On parle de codage disjonctif complet. Voici les 10 premières lignes pour les données VOTE (Figure 39).

group	adop_y	adop_n	adop_u	edu_y	edu_n	edu_u
republican	0	1	0	1	0	0
republican	0	1	0	1	0	0
democrat	1	0	0	0	1	0
democrat	1	0	0	0	1	0
democrat	1	0	0	0	0	1
democrat	1	0	0	0	1	0
democrat	0	1	0	0	1	0
republican	0	1	0	0	1	0
republican	0	1	0	1	0	0
democrat	1	0	0	0	1	0

Figure 39 – Codage disjonctif complet – Données VOTE

Coordonnées des modalités. Deux tableaux de résultats sont importants en ACM. Le premier correspond aux coordonnées des indicatrices sur les axes factoriels. Il permet de comprendre la nature des facteurs extraits par la méthode (Figure 40).

Values	Overall			Coordinate			
Attribute = Value	Mass	Sq.Dist	Inertia	coord_1	coord_2	coord_3	coord_4
edu_spending = y	0.1966	1.5439	0.3034	-1.10552	-0.18779	-0.15606	-0.51192
edu_spending = n	0.2678	0.867	0.2322	0.79957	-0.23144	-0.19233	0.37024
edu_spending = u	0.0356	13.0323	0.4644	0.08853	2.77544	2.30644	0.04099
adop_budget = n	0.1966	1.5439	0.3034	-1.11883	-0.1183	0.09831	0.51808
adop_budget = y	0.2908	0.7194	0.2092	0.75477	-0.12764	0.10607	-0.3495
adop_budget = u	0.0126	38.5455	0.4874	0.03314	4.77485	-3.96799	-0.01535

Figure 40 – Coordonnées factorielles des modalités – ACM – Données VOTE

Le premier axe est défini par la conjonction des votes (edu_spending = y) et (adop_budget = n). Et par opposition des votes (edu_spending = n et adop_budget = y).

Le second axe met en avant les modalités (edu_spending = u) et (adop_budget = u). Mais nous remarquons par ailleurs qu'elles sont très rares (colonne « Mass »). Elles peuvent peser indument sur les résultats, il faut être très prudent quant à leur interprétation.

Coefficients de projection. Le second tableau (Figure 41) fournit les coefficients de projection qui, lorsqu'ils sont appliqués sur les indicatrices, permet d'obtenir les coordonnées factorielles des individus. Ils définissent les variables latentes.

Attribute = Value	Axis_1	Axis_2	Axis_3	Axis_4
adop_budget = y	0.416	-0.083	0.083	-0.416
adop_budget = n	-0.616	-0.077	0.077	0.616
adop_budget = u	0.018	3.104	-3.104	-0.018
edu_spending = y	-0.609	-0.122	-0.122	-0.609
edu_spending = n	0.441	-0.150	-0.150	0.441
edu_spending = u	0.049	1.804	1.804	0.049

Figure 41 – Coefficients de projection – ACM – Données VOTE

« Axis_1 » (Z1) est une variable intermédiaire, une variable synthétique, qui est exprimée par la combinaison linéaire des indicatrices initiales. Comme nous manipulons exclusivement des indicatrices, les valeurs et signes des coefficients sont comparables.

$$Z_1 = 0.416\, adop_y - 0.616\, adop_n + 0.018\, adop_u - 0.609\, edu_y + 0.441\, edu_n + 0.049\, edu_u$$

A partir de ces coefficients et de leurs coordonnées, nous pouvons obtenir les positions relatives des individus dans le nouvel espace de représentation. Pour le premier individu par exemple, décrit par le vecteur (0, 1, 0, 1, 0, 0), sa cordonnée sur le premier facteur s'écrit :

$$Z_1(1) = 0.416 \times 0 - 0.616 \times 1 + 0.018 \times 0 - 0.609 \times 1 + 0.441 \times 0 + 0.049 \times 0$$
$$= -1.226$$

Si l'on s'en tient aux deux premiers facteurs, voici la représentation des individus dans le plan, étiquetés selon leur classe d'appartenance (Note : il n'y a que 6 valeurs possibles, beaucoup de points sont superposés, j'ai rajouté du bruit [jittering] pour que l'on distingue mieux les blocs d'observations) (Figure 42).

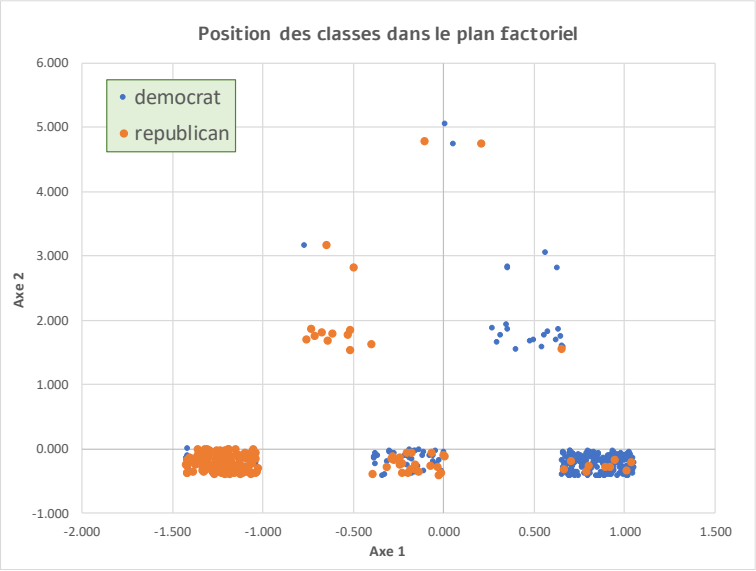


Figure 42 – Position des observations dans le premier plan factoriel de l'ACM – VOTE

C'est dans ce repère que l'analyse discriminante linéaire va construire la frontière de séparation entre les classes. Notons au passage que les « republicans » sont plutôt situés sur les valeurs négatives du premier facteur, ce qui correspond aux caractéristiques (edu_spendig = y) et (ado_budget = n). Ce résultat est cohérent avec celui mis en avant par l'analyse discriminante directement appliquée sur les indicatrices (section 3.2).

3.3.2 Analyse discriminante sur facteurs

Avec l'analyse discriminante linéaire, nous cherchons à prédire le « group » d'appartenance politique à partir des NB_FACT premiers facteurs (Zj) de l'ACM. NB_FACT est un

hyperparamètre de l'algorithme DISQUAL. Le réduire améliore les propriétés de régularisation, mais nous prenons le risque de ne pas capter suffisamment les informations véhiculées par les données. L'augmenter nous fait prendre le risque du surapprentissage. En tous les cas, il faut bien prendre les NB_FACT premiers facteurs (TUTO 5, section 3.2). Il ne s'agit de laisser des « trous » parmi les facteurs à utiliser dans l'analyse discriminante. N'oublions pas qu'un axe est le fruit du calcul sur la partie résiduelle des précédents.

Voici les résultats de l'ADL sur les 2 premiers facteurs (Z1, Z2) (Figure 43).

Classification functions			Statistical Evaluation			
Attribute	republican	democrat	Wilks L.	Partial L.	F(1,432)	p-value
Z1	-2.851	1.794	1.000	0.382	700.356	0.000
Z2	0.096	-0.060	0.382	0.999	0.564	0.453
constant	-2.235	-0.996	-			

Figure 43 – ADL sur facteurs – VOTE

Nous aurions pu nous contenter du premier facteur (Z1) en réalité. Mais nous passons outre pour pouvoir effectuer les représentations graphiques. Puisque nous sommes dans le plan (Z1, Z2), nous matérialisons la droite de séparation (Figure 44).

$$Z_2 = 29.86\, Z_1 + 7.96$$

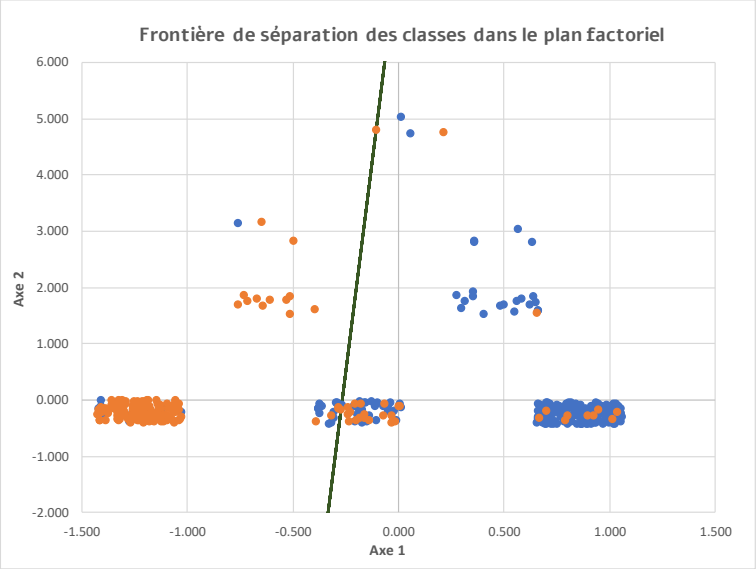


Figure 44 – Frontière de séparation des classes dans le plan factoriel – VOTE

3.3.3 Expression du modèle à partir des indicatrices

Dernière étape du processus, nous exprimons les fonctions de classement dans l'espace originel des indicatrices. Pour cela, nous effectuons le produit matriciel entre les coefficients de projection de l'ACM (Figure 41) et ceux de l'ADL (Figure 43), sans oublier la constante.

Concrètement pour « republican » par exemple :

$$
\begin{aligned}
& d(republican, X) \\
& = -2.851\, Z_1 + 0.096\, Z2 - 2.235 \\
& = -2.851\left(0.416\, adop_y - 0.0616\, adop_n + \cdots\right) + 0.096\left(-0.083\, adop_y - 0.077\, adop_n + \cdots\right) - 2.235 \\
& = -1.194\, adop_y + 1.750\, adop_n + 0.244\, adop_u + 1.725\, edu_y + \cdots - 2.235
\end{aligned}
$$

Voici le tableau avec l'ensemble des coefficients (Figure 45).

Attribute = Value	republican	democrat
adop_budget = y	-1.194	0.751
adop_budget = n	1.750	-1.101
adop_budget = u	0.244	-0.154
edu_spending = y	1.725	-1.086
edu_spending = n	-1.271	0.799
edu_spending = u	0.033	-0.021
constant	-2.235	-0.996

Figure 45 – Fonctions de classement sur les indicatrices – DISQUAL – VOTE

Comme toutes les indicatrices sont représentées, la lecture est facilitée. Nous retrouvons des résultats bien connus maintenant : les républicains votent volontiers (adop_budget = n et edu_spending = y), les démocrates (adop_budget = y et edu_spending = n).

3.4 Cas du mix de descripteurs quantitatifs et qualitatifs

Plusieurs approches sont envisageables lorsque nous avons un mix de variables prédictives quantitatifs et qualitatifs :

- Laisser telles quelles les quantitatives et coder en 0/1 les qualitatives en veillant à exclure les modalités de référence.

- Discrétiser les quantitatives et passer par DISQUAL. Le choix de l'algorithme de discrétisation est crucial dans ce cas.
- Utiliser une technique factorielle qui sait appréhender les bases composées de variables de différents types comme l'analyse factorielle des données mixtes (AFDM ; Rakotomalala R., « Analyse factorielle des données mixtes – Diapos », août 2013). Appliquer l'ADL sur les axes factoriels. Revenir ensuite dans l'espace initial en exploitant les coefficients des fonctions de classement de l'ADL et ceux des fonctions de projection de l'AFDM. Cette stratégie constitue une véritable généralisation car l'AFDM correspond à une ACP normée (analyse en composantes principales) lorsque les variables sont toutes quantitatives, à une ACM (analyse des correspondances multiples) lorsqu'elles sont toutes qualitatives. Tout comme avec DISQUAL, le nombre de facteurs à retenir est un des enjeux du système.

Très honnêtement, je n'ai jamais constaté de réel différentiel de performances entre ces différentes approches. La première solution est la plus simple à mettre en œuvre rapidement, au moins dans une phase de première appréhension des données. La troisième est certainement la plus intéressante intellectuellement. Il s'agit d'une généralisation naturelle de la méthode DISQUAL aux prédicteurs composés d'un mix de variables quantitatives et qualitatives. Nous pourrions l'appeler DISMIX de ce point de vue. Lorsque toutes les variables sont qualitatives, nous retrouvons à l'identique la méthode DISQUAL.

4 Sélection de variables

La sélection de variables est un dispositif essentiel de l'analyse prédictive (Rakotomalala, 2011, section 7.1). Un modèle est destiné à être interprété et déployé. Il ne doit pas être pollué par des descripteurs non pertinents et/ou redondants.

Lorsque leur nombre est contenu et que l'on dispose de connaissances du domaine exogènes aux données, la détermination experte du sous-ensemble de variables pertinentes est la meilleure solution. Mais en pratique, on compte plutôt sur les données pour extraire de la connaissance, et nous disposons d'un large ensemble de variables candidates dont on ne maîtrise pas toujours la teneur. La recherche automatique est dès lors incontournable.

Trois familles de techniques sont mises en avant dans la littérature.

L'approche « filtre » agit en amont, avant la phase de modélisation. Son principal avantage est la rapidité. En revanche, rien ne dit que des variables sélectionnées sur la base de critères ad hoc d'association et de redondance soient les meilleures quelles que soient les algorithmes d'apprentissage mis en œuvre par la suite.

L'approche « wrapper » optimise un critère de performance (souvent le taux d'erreur, ou son complément à 1, le taux de succès) via des techniques de rééchantillonnage. Les modèles sont utilisés comme des boîtes noires, on ne s'intéressent absolument pas à leur caractéristiques intrinsèques, le procédé leur présente des scénarios de solutions à évaluer, ils renvoient le niveau de performances. Elle a pour avantage l'efficacité prédictive. On remarque d'ailleurs qu'un système analogue de grille de recherche est utilisé pour définir les valeurs adéquates des hyper

paramètres des algorithmes. En revanche, lorsque nous traitons une base comportant un très grand nombre de descripteurs générées automatiquement (ex. text mining), le temps de traitement peut être prohibitif.

L'approche « intégrée » que nous étudions dans ce chapitre s'appuie sur les mécanismes et critères internes de la méthode d'apprentissage pour déterminer le « meilleur » sous-ensemble de variables. Pour l'analyse discriminante, on le devine aisément à la lecture du chapitre 2, tout tournera autour de la séparabilité des classes représentées par leurs centres respectifs. Le critère incontournable à cet égard est le lambda de Wilks.

Les algorithmes d'exploration associés reposent essentiellement sur la recherche pas-à-pas. Ils sont regroupés dans la PROC STEPDISC (stepwise discriminant analysis) dans le logiciel SAS. Les variables sont ajoutées ou retirées graduellement. A chaque étape, un test de significativité permet de valider l'opération. Dans les deux sections qui suivent, nous nous intéressons aux procédures qui génèrent une succession de solutions emboîtées, les sélections ascendantes (forward) et descendantes (backward). Une troisième piste, dite directionnelle (method = STEPWISE dans STEPDISC), est souvent citée dans les références. Elle mixe les approches ascendantes et descendantes. A chaque ajout de variables, le système vérifie s'il est possible d'en retirer. Les sous-ensembles successifs ne sont plus emboîtés dans ce cas.

Rappelons que ces techniques numériques - paramétrées de surcroît, ce qui pèse sur la teneur des résultats – nous présentent des scénarios de sous-ensemble de variables. Dans une expérimentation menée sous TANAGRA sur la base de données SONAR, les algorithmes forward et backward ont tous deux proposés la sélection de 7 variables, mais avec seulement 2 en commun (TUTO 6). Quelle solution choisir ? Personne ne peut y répondre d'emblée. Il nous appartient de les inspecter et de les valider, en fonction des explorations que nous menons par ailleurs sur les données (les statistiques descriptives et la data visualisation font énormément pour la data science, il ne faudrait jamais l'oublier), en fonction également des connaissances du domaine qui sont indispensables pour guider ce que nous faisons.

Par rapport à backward, l'approche forward est à privilégier sur les grandes bases de données (p élevé). En effet, dans la mesure où le nombre de variables définitivement sélectionnées (q*) est

souvent faible par rapport à (p), la quantité d'opérations à réaliser – et par conséquent le temps de calcul – sera moindre.

A contrario, la stratégie backward paraît plus intéressante lorsque (p) est modéré parce que nous démarrons avec la totalité de l'information disponible. Et nous avons la garantie que toutes les variables sont statistiquement pertinentes à l'issue des traitements.

Dans ce que suit, « q » représente le nombre de variables sélectionnées à l'étape courante. Le nombre final de variables sélectionnées (q*) varie entre $0 \leq q^* \leq p$.

4.1 Approche ascendante forward

La sélection ascendante part du modèle nul, sans variables, et les ajoute graduellement. L'algorithme identifie la meilleure candidate à chaque étape [pour passer de (q) à (q+1)] à l'aide d'un critère numérique, il l'ajoute à la sélection si la règle d'arrêt n'est pas déclenchée, le processus est stoppé sinon.

La variable sélectionnée à l'étape (q) est définitivement entérinée, elle ne peut plus être remise en cause, elle est donc retirée du pool des variables candidates.

En pratique, bien que nous soyons sur une stratégie forward, il peut être avantageux de calculer les matrices de variance covariance intra-classes (Wb) et totales (Vb) sur la totalité des (p) variables, tant que (p) reste raisonnable. En effet, elles peuvent être calculées en une seule passe sur les données si on s'y prend bien. Puis de piocher dans ces matrices pour calculer les différentes valeurs de Λ.

4.1.1 Point de départ

Lorsqu'aucune variable n'est sélectionnée (q = 0), les barycentres conditionnels ne sont pas discernables, forcément ($\Lambda_0 = 1$). Toutes les variables sont initialement candidates.

4.1.2 Identification de la variable à introduire

Pour identifier la plus contributive pour passer de (q) à (q+1) variables sélectionnées, nous comparons les lambda de Wilks avec la statistique (Tufféry, 2012 ; page 414) :

$$F = \frac{n - K - q}{K - 1} \times \left(\frac{\Lambda_q}{\Lambda_{q+1}} - 1 \right)$$

La variable qui maximise F (F-to-Enter dans les logiciels) est celle qui nous intéresse.

Dans certains outils (SAS - STEPDISC), l'indicateur $R^2_{partiel}$ est mis en avant :

$$R^2_{partiel} = 1 - \frac{\Lambda_{q+1}}{\Lambda_q}$$

Plus élevé est le $R^2_{partiel}$, plus intéressante est la variable.

4.1.3 Règle d'arrêt

La première règle d'arrêt naturelle est qu'il n'y a plus de variables candidates (p – q = 0).

Sinon, nous nous basons sur un mécanisme de test de significativité. F suit une loi de Fisher à (K – 1, n – K – q) degrés de liberté sous l'hypothèse nulle (la variable ne joue aucun rôle dans la discrimination des classes). Si la p-value de la meilleure variable est inférieure au risque (α) (paramètre de l'algorithme) que l'on s'est fixé, elle est acceptée dans la sélection.

Il faut être très prudent quant à la signification statistique du paramètre (α) dans ce processus. En effet, en effectuant des comparaisons réitérées à chaque étape, le véritable risque de première espèce est faussé. Il est certainement plus élevé. A force de chercher nous augmentons la probabilité d'inclure à tort une variable non pertinente dans le modèle. Mais ce n'est pas vraiment gênant dans une démarche exploratoire. Il faut plutôt voir (α) comme un outil de pilotage qui permet d'orienter le processus de recherche. A nous de le moduler en fonction des caractéristiques des solutions que nous souhaitons mettre en exergue.

Néanmoins, pour dépasser cet inconvénient, il est possible dans certains logiciels de fixer directement le seuil sur F (la variable est acceptée si F-to-enter > seuil entrée). Mais déterminer

la bonne valeur de coupure n'est pas aisée. On retombe sur les mêmes problématiques qu'avec la probabilité critique.

Enfin, il possible avec certains outils de fixer arbitrairement le nombre (q*) de variables à sélectionner, en fonction de considérations métiers.

4.1.4 Un exemple détaillé – DATA 2

Dans cette section nous effectuons une sélection forward sur les données DATA 2 / TRAIN. Rappelons que n = 52, K = 3, p = 8. Pour règle d'arrêt, nous choisissons le seuil (α = 0.01). Nous utilisons l'outil STEPDISC du logiciel TANAGRA (TUTO 6).

(**q = 0, $\Lambda_0 = 1$**) Au départ, puisqu'aucune variable n'est sélectionnée, nous avons : $\Lambda_0 = 1$

L'influence des 8 variables ont été mesurées lorsqu'elles sont introduites dans le modèle. Nous obtenons le tableau suivant.

d.f	Best	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 49)	**MEOH**	**MEOH**	**BU1**	**MEPR**	**PRO1**	**ISOP**	**BU2**	**ACET**	**ACAL**
	L : 0.2826	L : 0.2826	L : 0.2862	L : 0.6919	L : 0.8355	L : 0.8877	L : 0.9146	L : 0.9719	L : 0.9796
	F : 62.19	F : 62.19	F : 61.11	F : 10.91	F : 4.82	F : 3.10	F : 2.29	F : 0.71	F : 0.51
	p : 0.0000	p : 0.0000	p : 0.0000	p : 0.0001	p : 0.0122	p : 0.0541	p : 0.1122	p : 0.4969	p : 0.6042

La variable la plus intéressante est MEOH, parce qu'elle induit la plus forte réduction du lambda (Λ_1 = 0.2826). Ou, de manière complètement équivalente, elle présente le F le plus élevé. Nous avons :

$$F = \frac{n-K-q}{K-1} \times \left(\frac{\Lambda_q}{\Lambda_{q+1}} - 1\right) = \frac{52-3-0}{3-1} \times \left(\frac{1}{0.2826} - 1\right) = 62.19$$

La p-value pour une loi de Fisher à (K-1 = 2, n-K-q = 49) degrés de liberté est < 0.00001. Puisqu'elle est inférieure à (α = 0.01), la variable MEOH est acceptée.

(**q = 1, $\Lambda_{(MEOH)}$ = 0.2826**) Nous cherchons la seconde variable qui pourrait épauler MEOH dans le modèle. Nous testons l'impact d'une variable additionnelle. La meilleure serait BU1 avec un

lambda global (incluant les deux variables MEOH et BU1) = Λ_2 = 0.1925. Nous constatons que MEOH ne fait plus partie des variables candidates à cette étape.

d.f	Best	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 48)	**BU1** L : 0.1925 F : 11.23 p : 0.0001	**BU1** L : 0.1925 F : 11.23 p : 0.0001	**MEPR** L : 0.2212 F : 6.66 p : 0.0028	**ACAL** L : 0.2357 F : 4.78 p : 0.0128	**BU2** L : 0.2441 F : 3.79 p : 0.0297	**ACET** L : 0.2536 F : 2.75 p : 0.0743	**PRO1** L : 0.2557 F : 2.53 p : 0.0902	**ISOP** L : 0.2641 F : 1.69 p : 0.1958	-

$$F = \frac{n-K-q}{K-1} \times \left(\frac{\Lambda_q}{\Lambda_{q+1}} - 1\right) = \frac{52-3-1}{3-1} \times \left(\frac{0.2826}{0.1925} - 1\right) = 11.23$$

La p-value est 0.0001 < 0.01. La variable BU1 est acceptée.

(**q = 2, $\Lambda_{\{MEOH,\ BU1\}}$ = 0.2826**) La meilleure pour passer à (q+1 = 3) serait MEPR, où le $\Lambda_{\{MEOH,\ BU1,\ MEPR\}}$ = 01478. Est-ce que le gain est significatif ?

d.f	Best	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 47)	**MEPR** L : 0.1478 F : 7.12 p : 0.0020	**MEPR** L : 0.1478 F : 7.12 p : 0.0020	**BU2** L : 0.1703 F : 3.07 p : 0.0558	**ACAL** L : 0.1735 F : 2.58 p : 0.0864	**ISOP** L : 0.1744 F : 2.45 p : 0.0970	**PRO1** L : 0.1761 F : 2.19 p : 0.1227	**ACET** L : 0.1787 F : 1.82 p : 0.1737	-	-

$$F = \frac{n-K-q}{K-1} \times \left(\frac{\Lambda_q}{\Lambda_{q+1}} - 1\right) = \frac{52-3-2}{3-1} \times \left(\frac{0.1925}{0.1478} - 1\right) = 7.12$$

Avec une p-value = 0.0020 < 0.01. MEPR est validée. 3 variables sont sélectionnées à ce stade.

(**q = 3, $\Lambda_{\{MEOH,\ BU1,MPER\}}$ = 0.1478**) Pour passer à (q + 1 = 4) variables, ACAL serait la meilleure.

d.f	Best	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 46)	-	**ACAL** L : 0.1274 F : 3.69 p : 0.0327	**ISOP** L : 0.1298 F : 3.18 p : 0.0507	**BU2** L : 0.1315 F : 2.85 p : 0.0679	**PRO1** L : 0.1366 F : 1.89 p : 0.1628	**ACET** L : 0.1380 F : 1.63 p : 0.2076	-	-	-

$$F = \frac{n-K-q}{K-1} \times \left(\frac{\Lambda_q}{\Lambda_{q+1}} - 1\right) = \frac{52-3-3}{3-1} \times \left(\frac{0.1478}{0.1274} - 1\right) = 3.69$$

Avec une p-value = 0.0327 > 0.01. La variable n'est pas validée. Ce qui signe l'arrêt de l'algorithme.

Conclusion : Les variables sélectionnées en définitive sont {MEOH, BU1, MEPR}.

Voici les caractéristiques du modèle (TYPE ~ MEOH + BU1 + MEPR).

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,47)	p-value
MEOH	0.006922	0.021321	0.022619	0.220542	0.670102	11.56927	0.000082
BU1	-0.076617	0.401044	0.373522	0.221217	0.668058	11.67659	0.000076
MEPR	0.086678	-0.032474	0.046747	0.192547	0.767532	7.1176	0.001994
constant	-3.610717	-14.775375	-18.371099		-		

Figure 46 – Modèle pour DATA 2 / TRAIN après sélection forward (α = 0.01)

4.1.5 Sélection forward sous R

Exceptionnellement, nous utilisons un package sous R, « klaR » qui implémente la sélection forward pour l'analyse discriminante avec la fonction greedy.wilks() (TUTO 3, section 5.5).

Nous réimportons les données et nous lançons directement la procédure, avec (niveau = α = 0.01).

```
#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")

#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#chargement de la librairie
library(klaR)

## Loading required package: MASS

#pour affichage plus agréable
library(knitr)

#appel de la fonction
selfwd <- greedy.wilks(TYPE ~ ., data = DTrain, niveau = 0.01)
kable(selfwd$results,digits=4,caption="Résultats de la sélection forward")
```

Résultats de la sélection forward

vars	Wilks.lambda	F.statistics.overall	p.value.overall	F.statistics.diff	p.value.diff
MEOH	0.2826	62.1861	0	62.1861	0e+00
BU1	0.1925	30.6944	0	11.2283	1e-04
MEPR	0.1478	25.0864	0	7.1176	2e-03

Nous disposons de la liste des variables tour à tour sélectionnées, avec :

- « Wilks.Lambda », le lambda de Wilks global lorsque la variable est sélectionnée.
- « F.statistics.overall », le F de Rao calculé sur le lambda global. C'est l'information additionnelle ici par rapport à la présentation étape par étape dans la section précédente.
- « p.value.overall », la p-value correspondant à la statistique « F.statistics.overall ».
- « F.statistics.diff » est le F différentiel, lorsqu'on tente de rajouter une variable, tel qu'il est présenté ci-dessus.
- « p.value.diff » est la p-value associée.

4.1.6 Décroissance du lambda de Wilks

Lorsque nous travaillons sur des grandes bases (n très grand), toutes les variables ont tendance à être statistiquement pertinentes. Fixer la valeur d'arrêt (α) devient très compliqué.

Une piste possible consiste à étudier la courbe de décroissance du (Λ) en fonction du nombre de variables sélectionnées (q). On peut penser qu'il est temps de s'arrêter lorsque l'adjonction d'une variable supplémentaire n'induit plus une réduction sensible du (Λ). Cela équivaut à chercher la solution (q*) correspondant au « coude » de la courbe. Il reste alors à reparamétrer l'algorithme en spécifiant ce nombre identifié empiriquement.

Dans l'exemple ci-dessous, nous traitons la base WAVEFORM de n= 5000 observations avec 21 (originelles) + 40 (bruit ajouté) variables, avec (α = 0.1). La courbe est édifiante. A partir de 14 variables, les ajouts engendrent une réduction négligeable du Λ (Figure 47).

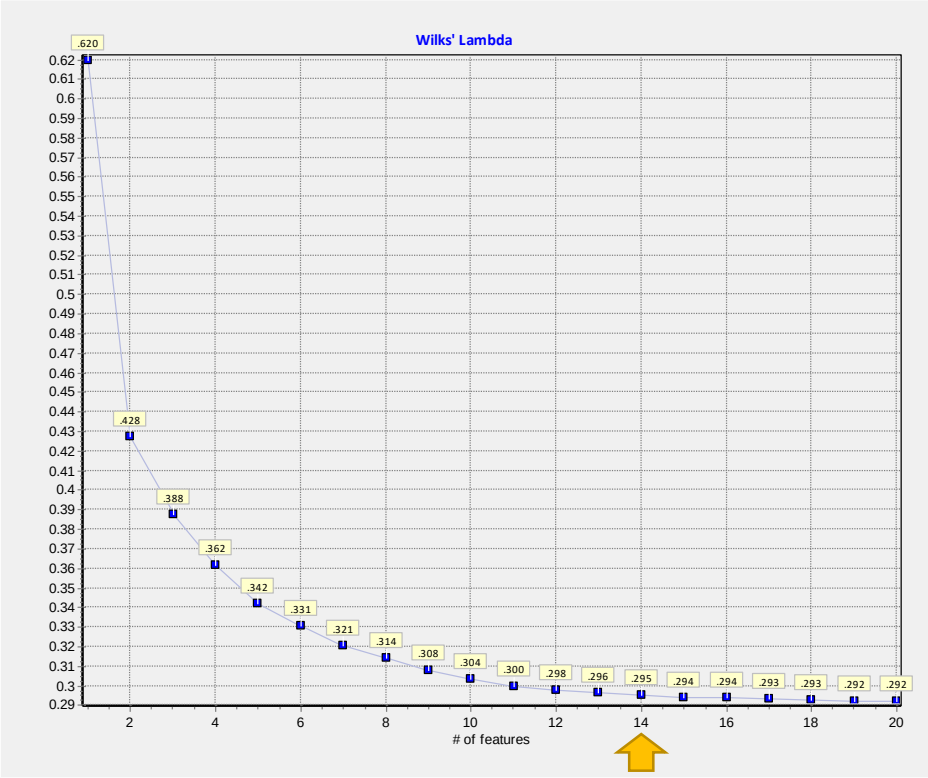


Figure 47 – Sélection forward – Décroissance du lambda vs. le nombre de variables sélectionnées – WAVEFORM

4.2 Approche descendante backward

La sélection descendante part du modèle intégrant la totalité des variables et les retire graduellement. L'algorithme identifie à chaque étape la variable la moins pertinente [pour passer de (q) à (q-1) descripteurs] à l'aide d'un critère numérique. Il la retire si la règle d'arrêt n'est pas déclenchée, le processus est stoppé sinon.

Tout retrait est définitif.

4.2.1 Point de départ

La première valeur du lambda est celle du modèle incluant la totalité des (p) variables : Λ_p

4.2.2 Identification de la variable à retirer

Pour identifier la variable la moins contributive, nous utilisons le test de pertinence des variables dans le modèle (section 2.3) :

$$F = \frac{n-K-q+1}{K-1}\left(\frac{\Lambda_{q-1}}{\Lambda_q} - 1\right)$$

La variable qui minimise F (F-to-Remove) est celle qui est menacée de retrait.

Le $R^2_{partiel}$ est calculé comme suit :

$$R^2_{partiel} = 1 - \frac{\Lambda_q}{\Lambda_{q-1}}$$

Est susceptible de retrait la variable qui présente de $R^2_{partiel}$ le plus faible.

4.2.3 Règle d'arrêt

La première règle d'arrêt naturelle est que toutes les variables ont été retirées.

Sinon, comme avec la sélection ascendante, nous nous appuyons sur un test de significativité, F suit une loi de Fisher à (K-1, n-K-q+1) degrés de liberté. Nous comparons la p-value avec un seuil α qui fait office de paramètre de l'algorithme. L'élimination des variables se poursuit tant que (p-value > α).

Ici aussi, nous pouvons directement fixer le seuil sur F pour dépasser les problèmes liés aux comparaisons multiples. Les variables sont retirées tant que (F-to-remove < seuil sortie).

Nous pouvons enfin spécifier arbitrairement le nombre de variables à retirer (ou à conserver).

4.2.4 Un exemple détaillé – DATA 2

Nous opérons une sélection backward sur DATA 2 / TRAIN avec (α = 0.01). Nous utilisons l'outil STEPDISC de TANAGRA (TUTO 6).

($\mathbf{q = p = 8,\ \Lambda_8 = 0.0667}$) Toutes les variables sont intégrées dans le modèle au départ.

La pertinence des variables a été mesurée à l'aide de la statistique F. Celle qui pèse le moins c.-à-d. qui induit la plus faible dégradation (augmentation) du lambda lorsque nous la retirons, est ISOP (Λ = 0.0723).

d.f	Worst	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 42)	ISOP	ISOP	ACET	ACAL	BU1	MEPR	PRO1	BU2	MEOH
	L : 0.0723	L : 0.0723	L : 0.0742	L : 0.0759	L : 0.0842	L : 0.0878	L : 0.0924	L : 0.0957	L : 0.1180
	F : 1.76	F : 1.76	F : 2.34	F : 2.89	F : 5.50	F : 6.64	F : 8.08	F : 9.12	F : 16.14
	p : 0.1842	p : 0.1842	p : 0.1086	p : 0.0669	p : 0.0076	p : 0.0031	p : 0.0011	p : 0.0005	p : 0.0000

$$F = \frac{n-K-q+1}{K-1}\left(\frac{\Lambda_{q-1}}{\Lambda_q}-1\right) = \frac{52-3-8+1}{3-1}\left(\frac{0.0723}{0.0667}-1\right) = 1.76$$

La p-value correspondante est 0.1842 > 0.01. La variable est définitivement éliminée.

($\mathbf{q = 7, \Lambda_7 = 0.0723}$) Nous recherchons la seconde variable à éliminer.

d.f	Worst	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 43)	ACET	ACET	ACAL	BU1	MEPR	PRO1	BU2	MEOH	-
	L : 0.0798	L : 0.0798	L : 0.0818	L : 0.0929	L : 0.0982	L : 0.1012	L : 0.1098	L : 0.1297	
	F : 2.23	F : 2.23	F : 2.83	F : 6.14	F : 7.69	F : 8.59	F : 11.15	F : 17.06	
	p : 0.1193	p : 0.1193	p : 0.0703	p : 0.0045	p : 0.0014	p : 0.0007	p : 0.0001	p : 0.0000	

ACET est sur la sellette avec :

$$F = \frac{n-K-q+1}{K-1}\left(\frac{\Lambda_{q-1}}{\Lambda_q}-1\right) = \frac{52-3-7+1}{3-1}\left(\frac{0.0798}{0.0723}-1\right) = 2.23$$

Avec p-value = 0.1193 > 0.01, ACET est retirée.

($\mathbf{q = 6, \Lambda_6 = 0.0798}$) Est-ce qu'il y a une troisième variable à retirer ?

d.f	Worst	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 44)	ACAL	ACAL	BU1	MEPR	PRO1	BU2	MEOH	-	-
	L : 0.0865	L : 0.0865	L : 0.0986	L : 0.1072	L : 0.1128	L : 0.1221	L : 0.1471		
	F : 1.84	F : 1.84	F : 5.17	F : 7.55	F : 9.08	F : 11.65	F : 18.54		
	p : 0.1714	p : 0.1714	p : 0.0096	p : 0.0015	p : 0.0005	p : 0.0001	p : 0.0000		

ACAL peut-être, avec :

$$F = \frac{n-K-q+1}{K-1}\left(\frac{\Lambda_{q-1}}{\Lambda_q}-1\right) = \frac{52-3-6+1}{3-1}\left(\frac{0.0865}{0.0798}-1\right) = 1.84$$

Avec p-value = 0.1193 > 0.01, exit ACAL.

($q = 5$, $\Lambda_5 = 0.0865$) Une quatrième ?

d.f	Worst	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5	Sol.6	Sol.7	Sol.8
(2, 45)	-	BU1 L : 0.1102 F : 6.16 p : 0.0043	MEPR L : 0.1118 F : 6.57 p : 0.0031	PRO1 L : 0.1315 F : 11.70 p : 0.0001	BU2 L : 0.1366 F : 13.03 p : 0.0000	MEOH L : 0.1551 F : 17.85 p : 0.0000	-	-	-

La pire est BU1 :

$$F = \frac{n-K-q+1}{K-1}\left(\frac{\Lambda_{q-1}}{\Lambda_q} - 1\right) = \frac{52-3-5+1}{3-1}\left(\frac{0.1102}{0.0865} - 1\right) = 6.16$$

Avec une p-value = 0.0043 < 0.01 c.-à-d. le retrait de ACAL entraînerait une augmentation significative du Λ (la discrimination des classes est significativement moins bonne), elle ne peut pas être retirée. Ce qui signe la fin du processus.

Conclusion : Les variables sélectionnées sont donc celles qui n'ont pas été exclues, soit : {BU1, MEPR, PRO1, BU2, MEOH}. Voici les caractéristiques du modèle correspondant (Figure 48) :

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,45)	p-value
MEOH	0.006172	0.026956	0.031912	0.155113	0.557582	17.85282	0.000002
BU1	-0.086181	0.346433	0.280117	0.110174	0.785016	6.16184	0.004313
BU2	-0.005208	0.07042	0.116972	0.136572	0.633277	13.02945	0.000034
MEPR	0.086537	-0.025564	0.058299	0.111753	0.773921	6.57274	0.003131
PRO1	0.002536	-0.005289	-0.00836	0.131479	0.657808	11.70452	0.000081
constant	-4.411341	-16.918659	-24.263717		-		

Figure 48 – Modèle DATA 2 / TRAIN après sélection backward (α = 0.01)

Remarque : Malgré que nous ayons spécifié une valeur identique du paramètre (α = 0.01) pour les sélections forward et backward, nous n'obtenons pas les mêmes ensembles de variables à la sortie. La stratégie de recherche pèse sur les résultats. Son choix n'est pas anodin. Cet exemple le montre bien.

4.2.5 Sélection backward sous R

J'avais développé une petite fonction de mon cru pour la sélection de variables backward sous R (TUTO 1, section 5.2). Elle travaille à partir des matrices de variances covariances intra-classes

et totales incluant tous les descripteurs. Ce sont les seules informations réellement utiles pour réaliser le processus. Nous le reprenons ici puis nous l'appliquons aux données DATA 2 / TRAIN. Par rapport au tutoriel, le détail des calculs à chaque tentative d'élimination est affiché.

Le dispositif est composé de deux fonctions : test.retrait() est chargée d'évaluer l'ensemble des variables restantes pour une étape donnée ; sel.backward() englobe la précédente et met en place la procédure de sélection, paramétrée par α. Un mode « bavard » détaille les résultats à chaque stade.

```
#fonction pour le test du retrait d'une variable numéro j
#SW : matrice de covariance intra
#ST : matrice de covariance totale
#N : nombre d'observations
#K : nombre de classes
#precLW : valeur du lambda avant le retrait
test.retrait <- function(j,SW,ST,N,K,precLW){
  #nombre de variables à cette étape
  nbVar <- ncol(SW)
  #lamdba
  lambada <- det(SW[-j,-j])/det(ST[-j,-j])
  #ddl
  ddlNum <- K - 1
  ddlDenom <- N - K - nbVar + 1
  #f
  f <- ddlDenom / ddlNum * (lambada / precLW - 1)
  #p-value
  pvalue <- pf(f,ddlNum,ddlDenom,lower.tail=FALSE)
  #renvoyer sous forme de vecteur
  return(c(lambada,f,pvalue))
}

#fonction de sélection backward
#DW : matrice initiale de covariance intra
#DT : matrice initiale de covariance totale
#lstInit : liste initiale des variables
#lambdaInit : valeur initiale du lambda de wilks
#alpha : risque pour piloter la sélection
sel.backward <- function(DW,DT,lstInit,lambdaInit,alpha=0.05){
  #matrice pour récupérer les résultats à chaque étape
  mResult <- matrix(0,nrow=0,ncol=3)
  colnames(mResult) <- c('Lambda','F-Test','p-value')
  #liste des variables retirées
  lstvarDel <- c()
  #******************
  #boucle de recherche
  repeat{
    #resultats d'une étape
    res <- t(sapply(1:ncol(DW),test.retrait,SW=DW,ST=DT,N=n,K=K,precLW=lambdaInit))
    #affichage pour un mode "verbose"
    aff <- res
    colnames(aff) <- c("Lambda","F","p-value")
    rownames(aff) <- lstInit
    print(aff)
    #trouver la ligne qui maximise la p-value
    id <- which.max(res[,3])
    #vérifier si retrait (comparer p-value à alpha)
    if (res[id,3] > alpha){
```

```
      #nom de la variable à retirer
      lstvarDel <- c(lstvarDel,lstInit[id])
      #rajouter la ligne de résultats (lambda,F-Test,p-value)
      mResult <- rbind(mResult,res[id,])
      #retirer
      lstInit <- lstInit[-id]
      #si plus de variables
      if (length(lstInit) == 0){
        #arrêt puisque plus de variables à retirer
        break
      } else {
        #retirer les colonnes des matrices
        DW <- DW[-id,-id]
        DT <- DT[-id,-id]
        #màj du lambda
        lambdaInit <- res[id,1]
      }
    } else {
      #plus de retrait parce que condition alpha
      break
    }
  }
  #resultats sous la forme d'un data frame
  resultats <- data.frame(var=lstvarDel,mResult)
  return(resultats)
}
```

Pour lancer la procédure, nous chargeons les données et calculons les matrices de variance covariance nécessaires au calcul de la première valeur du lambda de Wilks incluant toutes les variables.

```
#** importation des données **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")

#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#quelques structures intermédiaires
XTrain <- DTrain[-1] # matrice des X
yTrain <- DTrain$TYPE # vecteur y
n <- nrow(DTrain) # nombre d'observations
K <- nlevels(yTrain) # nombre de classes
p <- ncol(XTrain) # nombre de variables
n_k <- table(yTrain) #effectifs des classes
```

```r
#** calculs des matrices de var.covariance pour wilks **

#totale
Vb <- (n-1)/n * cov(XTrain)

#intra-classe
Wb <- (1.0/n)*Reduce("+",lapply(levels(yTrain),function(niveau){(n_k[niveau]-1)*co
v(XTrain[yTrain==niveau,])}))

#** lambda de Wilks initial (incluant les p variables) **
lambda_depart <- det(Wb)/det(Vb)
print(lambda_depart)

## [1] 0.06671324
```

Nous pouvons lancer la sélection avec (alpha = 0.01). La variable éliminée à chaque stade est soulignée en vert dans les tableaux intermédiaires.

```r
#lancement de la sélection
selBwd <- sel.backward(DW=Wb,DT=Vb,lstInit=colnames(XTrain),lambdaInit=lambda_depart,alpha = 0.01)

##          Lambda         F      p-value
## MEOH 0.11797463 16.136067 6.322533e-06
## ACET 0.07415324  2.341965 1.085724e-01
## BU1  0.08418341  5.499262 7.563028e-03
## BU2  0.09569536  9.122996 5.125589e-04
## ISOP 0.07231005  1.761764 1.841962e-01
## MEPR 0.08779763  6.636945 3.128343e-03
## PRO1 0.09239573  8.084336 1.070935e-03
## ACAL 0.07588377  2.886700 6.688456e-02


##          Lambda         F      p-value
## MEOH 0.12969222 17.061483 3.508502e-06
## ACET 0.07982573  2.234643 1.193160e-01
## BU1  0.09294479  6.135344 4.528269e-03
## BU2  0.10980892 11.149568 1.256209e-04
## MEPR 0.09817091  7.689228 1.396821e-03
## PRO1 0.10121569  8.594537 7.243394e-04
## ACAL 0.08181253  2.825379 7.033071e-02


##          Lambda         F      p-value
## MEOH 0.14709728 18.540063 1.444913e-06
## BU1  0.09860081  5.174419 9.588992e-03
## BU2  0.12208684 11.647178 8.718238e-05
## MEPR 0.10721973  7.549796 1.517400e-03
## PRO1 0.11277663  9.081281 4.992557e-04
## ACAL 0.08648809  1.836148 1.714373e-01


##         Lambda         F      p-value
## MEOH 0.1551128 17.852822 1.958591e-06
## BU1  0.1101737  6.161841 4.312853e-03
## BU2  0.1365722 13.029454 3.434509e-05
## MEPR 0.1117531  6.572738 3.130928e-03
## PRO1 0.1314793 11.704517 8.076606e-05
```

Dans le dernier tableau, BU1 est celle qui a été la moins convaincante. Mais elle n'a pas été évincée parce que sa p-value est < 0.01.

La liste des variables retirées et leurs caractéristiques sont regroupées dans un tableau.

```
#affichage de la liste des variables retirées
print(selBwd)

##    var     Lambda   F.Test   p.value
## 1 ISOP 0.07231005 1.761764 0.1841962
## 2 ACET 0.07982573 2.234643 0.1193160
## 3 ACAL 0.08648809 1.836148 0.1714373
```

Nous retrouvons à l'identique les résultats de TANAGRA (section 4.2.4).

4.3 Sélection pour les descripteurs qualitatifs

4.3.1 Sélection en présence de variables prédictives qualitatives

La sélection des variables prédictives qualitatives posent des problèmes supplémentaires.

Lorsque nous utilisons des systèmes à deux étages de type DISQUAL (section 3.3), discerner la pertinence des prédicteurs est très difficile. Il faut se tourner vers les solutions wrapper ou, si le temps de calcul est une contrainte forte, au moins hiérarchiser les variables selon leur influence mesurée à l'aide de méthodes agnostiques c.-à-d. génériques, qui ne dépendent pas de la méthode de machine learning sous-jacente (Ex. « Importance des variables dans les modèles », février 2019).

L'affaire semble plus simple lorsque nous substituons des indicatrices aux variables qualitatives originelles (section 3.2). Mais cette approche pose une autre question : faut-il traiter individuellement ou en bloc les indicatrices issues d'une variable ?

Si nous les traitons individuellement, nous retrouvons les situations décrites précédemment (sections 4.1 et 4.2). Le processus n'est en rien modifié. Nous pouvons utiliser un programme standard de sélection pour variable comme la procédure développée pour R en section 4.2.5. Mais il faut être prudent quant à la notion de modalité de référence. Lorsque certaines indicatrices

d'une variable sont éliminées, les coefficients des indicatrices restantes se lisent en référence aux modalités qui n'apparaissent pas dans le modèle, qui forment alors un bloc.

Prenons un exemple pour clarifier cela. Mettons que la variable « taille » possède 3 modalités {petit, moyen, grand} (elles sont ordinales ici, mais cela n'altère en rien la généralité du propos). Nous créons deux indicatrices « taille_moyen » et « taille_grand », « petit » est dite modalité de référence et les coefficients a_{taille_moyen} et a_{taille_grand} se lisent en écart par rapport à {petit} (section 3.2.1). Si « taille_moyen » est éliminée et que seule subsiste « taille_grand » dans le modèle, le coefficient a_{taille_grand} se lit maintenant en écart par rapport au bloc de modalités constituées par les références {petit, moyen}. Sa valeur et sa lecture sont modifiées.

Si nous traitons en bloc les indicatrices d'une variable lors de l'ajout ou de la suppression dans la sélection forward ou backward, les calculs usuels ne sont plus valables, les statistiques F doivent être calculées différemment lorsque plusieurs indicatrices sont en jeu (section 2.4), et elles ne sont plus directement comparables parce qu'elles sont susceptibles de présenter des degrés de liberté différentes. Il faut alors considérer les p-values pour déterminer les variables à ajouter ou à retirer à chaque étape.

Quelle est la meilleure stratégie ? Difficile de trancher.

Le traitement individuel dispose de plus de libertés dans la recherche des solutions. Il aboutit de ce fait souvent à des modèles plus performants en prédiction. Mais il implique une gymnastique supplémentaire dans l'interprétation des résultats qui n'est pas toujours bien comprise, ni adaptée par rapport aux données que l'on traite. Et le modèle final dépend du choix initial des modalités de références pour chaque variable.

Le traitement en bloc a au moins le mérite de la cohérence, puisque nous parlons bien de sélection de « variables », mais il impose des calculs plus complexes, et il est moins performant.

4.3.2 Un exemple de traitement sous R

La base « HEART_STATLOG » décrit les caractéristiques de patients pour lesquels nous cherchons à prédire la présence ou absence d'une maladie (« disease »). Les variables prédictives

sont pêle-mêle quantitatives ou qualitatives. Une colonne « status » permettant de distinguer les individus en apprentissage (train) et en test (test) a été ajoutée pour que l'expérimentation puisse être menée à l'identique sur d'autres outils.

L'idée est de créer une base intermédiaire où les explicatives quantitatives sont laissées telles quelles, les qualitatives sont codée en indicatrices 0/1 (la première modalité dans l'ordre alphabétique servant de référence). Dans un premier temps, nous lançons l'analyse discriminante sur la totalité des descripteurs disponibles et nous mesurons les performances en test. Dans un deuxième temps, sur les données d'apprentissage, nous effectuons une sélection forward à 5% avec la fonction greedy.wilks() du package « klaR » (section 4.1.5). Elle ne sait pas que certains descripteurs sont en réalité des indicatrices issues d'une seule variable. Elle les traitent de manière individuelle. Nous réitérons l'apprentissage-test sur les variables sélectionnées et nous mesurons le taux d'erreur. Le modèle sera-t-il meilleur ou moins bon que le précédent incluant tous les descripteurs ? A voir.

Importation des données. Nous importons tout d'abord les données.

```
#** importation des donnes **

#lecture
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "HEART_STATLOG")

#inspection
str(D)

## 'data.frame':    270 obs. of  15 variables:
##  $ status        : Factor w/ 2 levels "test","train": 2 2 2 2 2 2 2 2 2 2 ...
##  $ disease       : Factor w/ 2 levels "absence","presence": 2 1 1 2 2 1 2 ...
##  $ age           : num  70 64 65 60 63 59 61 57 46 53 ...
##  $ sex           : Factor w/ 2 levels "female","male": 2 2 2 2 1 2 2 1 2 2 ...
##  $ chestpain     : Factor w/ 4 levels "atypicalAngina",..: 2 2 2 2 2 2 2 ...
##  $ restbpress    : num  130 128 120 140 150 135 134 128 140 140 ...
##  $ cholesteral   : num  322 263 177 293 407 234 234 303 311 203 ...
##  $ sugar         : Factor w/ 2 levels "high","low": 2 2 2 2 2 2 2 2 2 1 ...
##  $ electro       : Factor w/ 3 levels "normal","sttAbnormality",..: 3 1 1 3...
##  $ maxHeartRate  : num  109 105 140 170 154 161 145 159 120 155 ...
##  $ ExerciseAngina: Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 2 2 ...
##  $ oldpeak       : num  2.4 0.2 0.4 1.2 4 ...
##  $ slope         : Factor w/ 3 levels "downsloping",..: 2 2 3 2 2 2 2 3 2 1 ...
##  $ vesselsColored: num  3 1 0 2 3 0 2 1 2 0 ...
##  $ thal          : Factor w/ 3 levels "fixedEffect",..: 2 3 3 3 3 3 2 2 3 3 ...
```

« status » nous permettra de scinder en apprentissage (150) – test (120) notre jeu de données.

« disease » est la variable cible à K = 2 modalités {absence, presence}.

Pour mieux comprendre les résultats qui vont suivre, nous affichons les caractéristiques complètes de l'ensemble des variables.

```
#liste des caractéristiques
print(summary(D))

##    status        disease         age            sex                chestpain
##  test :120   absence :150   Min.   :29.00   female: 87   asymptomatic  :129
##  train:150   presence:120   1st Qu.:48.00   male  :183   atypicalAngina: 42
##                             Median :55.00                nonAnginal    : 79
##                             Mean   :54.43                typicalAngina : 20
##                             3rd Qu.:61.00
##                             Max.   :77.00

##    restbpress     cholesteral     sugar                    electro
##  Min.   : 94.0   Min.   :126.0   high: 40   normal            :131
##  1st Qu.:120.0   1st Qu.:213.0   low :230   sttAbnormality    :  2
##  Median :130.0   Median :245.0              ventricHypertrophy:137
##  Mean   :131.3   Mean   :249.7
##  3rd Qu.:140.0   3rd Qu.:280.0
##  Max.   :200.0   Max.   :564.0

##   maxHeartRate   ExerciseAngina    oldpeak             slope
##  Min.   : 71.0   no :181        Min.   :0.00   downsloping: 18
##  1st Qu.:133.0   yes: 89        1st Qu.:0.00   flat       :122
##  Median :153.5                  Median :0.80   upsloping  :130
##  Mean   :149.7                  Mean   :1.05
##  3rd Qu.:166.0                  3rd Qu.:1.60
##  Max.   :202.0                  Max.   :6.20

##  vesselsColored                 thal
##  Min.   :0.0000   fixedEffect     : 14
##  1st Qu.:0.0000   normal          :152
##  Median :0.0000   reversableEffect:104
##  Mean   :0.6704
##  3rd Qu.:1.0000
##  Max.   :3.0000
```

Construction des indicatrices. Nous écrivons une fonction pour générer les indicatrices 0/1 à partir des variables de type « factor » de R. Elle est appliquée à notre data frame : seuls les types « factor » sont concernés, « status » et « disease » ne sont pas inclus dans ce recodage.

```
#fonction pour créer des dummies
#à partir d'une variable de type factor
set_dummies <- function(x){
  #nombre de modalités
  L <- nlevels(x)
  #liste des modalités
  modalites <- levels(x)
  #création des variables 0/1 - la première modalité sert de référence
  M <- sapply(2:L,function(j){return(as.double(x==modalites[j]))})
  #nommer les variables avec les modalités
```

```r
  colnames(M) <- modalites[2:L]
  #renvoyer sous la forme de data frame
  return(as.data.frame(M))
}

#traitement des variables - seuls les factor sont concernés
#status et disease ne sont pas inclus dans les traitements
DD <- data.frame(lapply(D[-c(1,2)],function(x){if(is.factor(x)){set_dummies(x)}else{x}}))
str(DD)

## 'data.frame':    270 obs. of  18 variables:
##  $ age                       : num  70 64 65 60 63 59 61 57 46 53 ...
##  $ male                      : num  1 1 1 1 0 1 1 0 1 1 ...
##  $ chestpain.atypicalAngina  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ chestpain.nonAnginal      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ chestpain.typicalAngina   : num  0 0 0 0 0 0 1 0 0 0 ...
##  $ restbpress                : num  130 128 120 140 150 135 134 128 140 140 ...
##  $ cholesteral               : num  322 263 177 293 407 234 234 303 311 203 ...
##  $ low                       : num  1 1 1 1 1 1 1 1 1 0 ...
##  $ electro.sttAbnormality    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ electro.ventricHypertrophy: num  1 0 0 1 1 0 0 1 0 1 ...
##  $ maxHeartRate              : num  109 105 140 170 154 161 145 159 120 155 ...
##  $ yes                       : num  0 1 0 0 0 0 0 0 1 1 ...
##  $ oldpeak                   : num  2.4 0.2 0.4 1.2 4 ...
##  $ slope.flat                : num  1 1 0 1 1 1 1 0 1 0 ...
##  $ slope.upsloping           : num  0 0 1 0 0 0 0 1 0 0 ...
##  $ vesselsColored            : num  3 1 0 2 3 0 2 1 2 0 ...
##  $ thal.normal               : num  1 0 0 0 0 0 1 1 0 0 ...
##  $ thal.reversableEffect     : num  0 1 1 1 1 1 0 0 1 1 ...
```

Rappelons que la première modalité dans l'ordre alphabétique sert de référence. Lorsque la variable est binaire, R n'indique que la dénomination de la modalité codée (ex. male, ...) ; lorsqu'elle présente plus de 2 modalités, il lui associe le nom de la variable (ex. slope.flat, ...). Nous pourrions améliorer cela. Mais notre sujet est la sélection de variables dans cette section. Nous passons outre.

Nous associons la variable cible au data frame, que nous scindons ensuite en apprentissage et test à l'aide du champ « status ».

```r
#ajouter la cible au data.frame
DD['disease'] <- D$disease

#partition app-test
DTrain <- DD[D$status=="train",]
print(nrow(DTrain))

## [1] 150

DTest <- DD[D$status=="test",]
print(nrow(DTest))

## [1] 120
```

Apprentissage-test sur la totalité des descripteurs. Nous pouvons lancer la modélisation sur l'ensemble des descripteurs. Nous utilisons la fonction lda() du package « MASS » qui fait référence sous R, malgré que sa présentation des résultats soit tout à fait inhabituelle par rapport à la pratique de l'analyse discriminante prédictive. Elle ne fournit pas de fonction de classement explicite notamment (TUTO 1).

```
#** apprentissage test sur la totalité des variables

#package MASS
library(MASS)
adl_All <- lda(disease ~ ., data = DTrain)

#taux d'erreur en test
mean(DTest$disease != predict(adl_All,DTest)$class)

## [1] 0.1666667
```

En prédiction sur l'échantillon test, le taux d'erreur est de 16.67%. Voyons s'il est possible d'améliorer cela via la sélection de variables.

Sélection de descripteurs. La fonction greedy.wilks() réalise une sélection forward. Nous fixons (α = 0.05). Elle est menée sur les données d'apprentissage (DTrain). Cette précision est importante, l'échantillon test ne doit en rien participer à l'élaboration du modèle.

```
#** sélection de variables forward incluant les indicatrices **

#librairie KlaR
library(klaR)

#pour affichage
library(knitr)

#liste des variables sélectionnées - forward à 5%
fwd_sel <- greedy.wilks(disease ~ ., data = DTrain, niveau = 0.05)
print(kable(fwd_sel$results,digits=4))

##
##
## vars                      Wilks.lambda   F.statistics.overall   p.value.overall   F.statistics.diff   p.value.diff
## ------------------------  -------------  ---------------------  ----------------  ------------------  -------------
## thal.normal                     0.7877                39.8961                 0             39.8961         0.0000
## vesselsColored                  0.6612                37.6565                 0             28.1092         0.0000
## yes                             0.5681                36.9932                 0             23.9226         0.0000
## maxHeartRate                    0.5246                32.8526                 0             12.0393         0.0007
## oldpeak                         0.5003                28.7681                 0              6.9962         0.0091
## chestpain.typicalAngina         0.4637                27.5652                 0             11.2809         0.0010
## male                            0.4482                24.9707                 0              4.8969         0.0285
```

Trois des variables qualitatives sont intégrées dans le modèle via certaines de leurs indicatrices : « thal », « ExerciseAngina », « chestpain » et « sex ». Les modalités de références sont composites ou unitaires. Pour « thal », il s'agit de {fixedEffect, reversableEffect} ; pour

« ExerciseAngina », {no} ; pour « chestpain », {asymptomatic, atypicalAngina, nonAnginal} ; pour « sex », {female}.

Réitérons le schéma apprentissage-test sur les variables sélectionnées.

```
#** apprentissage test sur variables sélectionnées

#variables utilisées pour l'apprentissage
print(fwd_sel$formula)

## disease ~ thal.normal + vesselsColored + yes + maxHeartRate +
##     oldpeak + chestpain.typicalAngina + male
## <environment: 0x000000001a52e0a8>

#lancer l'apprentissage sur ces variables
adl_fwd <- lda(fwd_sel$formula, data = DTrain)

#taux d'erreur en test
mean(DTest$disease != predict(adl_fwd,DTest)$class)

## [1] 0.1666667
```

La sélection a été bénéfique. Nous avons préservé le taux d'erreur (16.67%) avec un modèle simplifié. C'est ce que l'on constate le plus souvent sur les bases usuelles où le ratio p nombre d'observations sur nombre de variables est favorable.

5 Particularité du problème à (K = 2) classes

L'analyse discriminante linéaire présente des propriétés intéressantes dans le cadre binaire. Y prend (K = 2) modalités que nous noterons $\{y_1, y_2\}$. Le problème n'est pas symétrique souvent. L'une (y_1) d'entre elles revêt une importance particulière, on parle de modalité cible ou modalité positive : les personnes présentant une maladie que l'on cherche à caractériser, les clients que l'on cherche à solliciter, les fraudeurs que l'on cherche à identifier, etc.

Plusieurs simplifications de l'ADL sont possibles dans ce cas, facilitant sa manipulation et son interprétation. Des connexions seront faites avec d'autres méthodes statistiques bien connues, notamment la régression linéaire multiple qui permet de retrouver à l'identique ses résultats moyennant quelques menues corrections. Ainsi, nous pouvons utiliser un programme de régression, très répandu dans les outils y compris ceux qui n'ont pas nativement une vocation statistique (ex. DROITEREG du tableur EXCEL), pour réaliser une analyse discriminante.

Cette relation permet d'ouvrir une porte vers le traitement multiclasse (K > 2) par le truchement de combinaisons de classifieurs binaires obtenus par la régression linéaire. Il s'agit plutôt d'un exercice de style pour l'analyse discriminante puisqu'elle s'applique nativement pour ($K \geq 2$). Mais les idées sont toujours intéressantes à explorer.

5.1 Distance entre les centres de classes

Comme nous n'avons que 2 classes, il y a une relation directe entre la distance de Mahalanobis des barycentres conditionnels et le Λ de Wilks. Elle s'écrit :

$$D_M^2 = \frac{1-\Lambda}{\Lambda} \times \frac{n\,(n-2)}{n_1 \times n_2}$$

Exemple DATA 1

Nous reprenons les principaux résultats de DATA 1, notamment le lambda (section 2.2.2) et la distance (section 2.1). Nous les confrontons.

```
#valeur du lambda
print(lambda)

## [1] 0.2524633

#effectif
print(n)

## [1] 11

#effectifs par classe
print(n_k)

## y
## g1 g2
##  5  6

#application de la formule
resultat <- ((1-lambda)/lambda) * ((n*(n-2))/(n_k[1]*n_k[2]))
print(resultat)

##       g1
## 9.771208

#calcul direct (cf. section 2.1)
print(DM)

##          [,1]
## [1,] 9.771208
```

La correspondance est exacte.

5.2 Fonction score

Le modèle prédictif s'exprime à l'aide de 2 fonctions de classement lorsque Y ϵ {y1, y2} :

$$d(y_1, X) = a_{10} + a_{11}\,x_1 + \cdots + a_{1p}\,x_p$$

$$d(y_2, X) = a_{20} + a_{21}\,x_1 + \cdots + a_{2p}\,x_p$$

En réalisant une différence termes à termes des coefficients, nous pouvons produire une fonction unique simplifiée, que j'appelle « fonction score » dans mes enseignements.

$$d(X) = c_0 + c_1\, x_1 + \cdots +\ c_p\, x_p$$

Appliqué à un individu ω, d(X(ω)) correspond au « score » de l'individu c.-à-d. le score d'appartenance à la modalité cible (y_1).

Exemple DATA 1

Pour DATA 1, nous affichons les coefficients des fonctions de classement (section 1.3.1), puis celles de la fonction score.

```
#coefficients des fonctions de classement
print(coef_)

##            g1        g2
## X1 -0.1294617  1.507304
## X2  0.6081081 -0.972973

#intercept
print(intercept_)

## y
##        g1        g2
## -2.110615 -5.139339

#coefficients de la fonction score
s_coef_ <- coef_[,1] - coef_[,2]
print(round(s_coef_,3))

##     X1     X2
## -1.637  1.581

#et intercept
s_intercept_ <- intercept_[1] - intercept_[2]
print(round(s_intercept_,3))

##    g1
## 3.029
```

Elle s'écrit donc :

$$d(X) = 3.029 - 1.637\, x_1 + 1.581\, x_2$$

Cette fonction score présente plusieurs intérêts.

5.2.1 Règle d'affectation et interprétation des coefficients

La fonction d(X) désigne la classe « y1 » par rapport à « y2 », la règle d'affectation devient :

$$\text{Si } d(X) > 0 \text{ Alors } Y = y_1 \text{ Sinon } Y = y_2$$

Une fonction unique est toujours plus facile à déployer, elle est aussi plus lisible. Le signe donne le sens de la liaison entre la cible et les explicatives. Pour l'exemple DATA 1, nous observons dans l'équation ci-dessus que les valeurs élevées de X1 correspond plutôt à la classe « g2 » (le signe de c_1 est négatif), que les valeurs élevées de X2, la classe « g1 » (c_2 est positif). Les positions relatives des observations dans le plan confirment ce constat (Figure 1). Lorsque (p > 2), un graphique devient compliqué, pouvoir tirer des interprétations à partir de l'expression du modèle devient primordial.

Les valeurs des c_j donne des indications sur l'influence relative des variables si ces dernières sont exprimées sur les mêmes échelles, ou sont centrées et réduites (surtout réduites). Et de toute manière, il est toujours possible de rendre les coefficients comparables en les multipliant par les écarts-type des variables comme en régression logistique (Rakotomalala, 2011, section 5.3.2).

En effet, nous avons la relation :

$$c_j^{std} = c_j \times \sigma_j$$

Où c_j^{std} est le coefficient de la fonction score dans le cas où l'analyse discriminante linéaire était calculée sur des données centrées et réduites ; σ_j est l'écart-type de la variable X_j.

5.2.2 Frontière de séparation des classes

La fonction score d(X) fournit directement la forme implicite de l'équation de la frontière de séparation des classes $[d(X) = 0]$.

Pour DATA 1, elle s'écrirait :

$$3.029 - 1.637\, x_1 + 1.581\, x_2 = 0$$

Passer à la forme explicite ne pose difficulté et nous retrouvons bien le tracé de la droite séparatrice dans l'espace de représentation (Figure 11).

5.2.3 Probabilité d'affectation

Le calcul des probabilités d'affectation P(Y = y1 / X) à partir du score est également simplifié puisque nous pouvons passer par la transformation logistique (Rakotomalala, 2011, section 1.3) :

$$P(Y = y_1 \,/\, X) = \frac{e^{d(X)}}{1 + e^{d(X)}}$$

Qui est un cas particulier de la fonction softmax. En effet, si nous posons $d = d(X)$, $d_1 = d(y_1,X)$ et $d_2 = d(y_2,X)$, nous avons :

$$\begin{aligned}\frac{e^{d}}{1+e^{d}} &= \frac{e^{d_1-d_2}}{1+e^{d_1-d_2}} \\ &= \frac{\frac{e^{d_1}}{e^{d_2}}}{1+\frac{e^{d_1}}{e^{d_2}}} \\ &= \frac{e^{d_1}}{e^{d_2}+e^{d_1}}\end{aligned}$$

Et comme nous n'avons que deux classes, forcément la probabilité d'appartenir à (y2) est le complémentaire à 1 :

$$P(Y = y_2/X) = 1 - P(Y = y_1 \,/\, X)$$

La règle d'affectation aux classes basées sur les probabilités devient :

Si $P(Y = y_1 \,/\, X) > 0.5$ Alors $Y = y_1$ Sinon $Y = y_2$

Exemple DATA 1

Nous affichons tout d'abord les données, puis nous appliquons les coefficients de la fonction score obtenus dans la section précédente. Nous effectuons la transformation logistique pour afficher les probabilités d'appartenance à la modalité cible « g1 ». Nous les faisons apparaître dans l'espace de représentation.

```
#coordonnées des individus
print(X)
```

```
##    X1 X2
## 1   0  3
## 2   2  1
## 3   4  6
## 4   6  9
## 5   8  7
## 6   5  2
## 7   7  0
## 8   9  4
## 9  11  8
## 10 13  6
## 11  4  0

#score
scores <-  s_intercept_ + s_coef_[1] * X$X1 + s_coef_[2] * X$X2
print(scores)

##  [1]  7.771967  1.336273  5.968148  7.437859  1.002166 -1.992943 -8.428636
##  [8] -5.377843 -2.327050 -8.762744 -3.518339

#probas d'être positif (g1)
probas <- exp(scores)/(1+exp(scores))
print(probas)

##  [1] 0.9995787939 0.7918764494 0.9974475541 0.9994118025 0.7314841921
##  [6] 0.1199459070 0.0002184716 0.0045965454 0.0889073182 0.0001564303
## [11] 0.0287949113

#affichage dans le graphique
plot(X$X1,X$X2,type="n", main="Probas d'appartenance à 'g1'")
abline(a=c0,b=c1,lwd=1)
text(X$X1,X$X2,col=c("deepskyblue3","seagreen3")[y],labels=round(probas,3),cex=0.8)
```

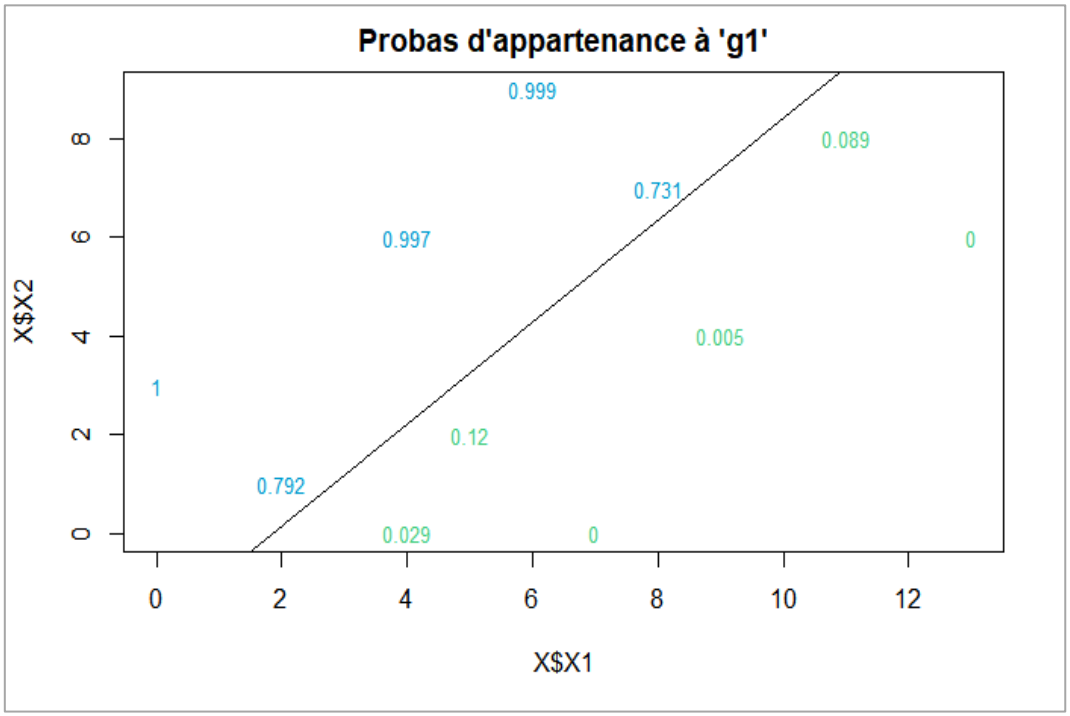


Figure 49 – Probabilités d'appartenance aux classes – DATA 1

Avec la fonction de transfert logistique, nous comprenons aisément pourquoi les probabilités d'appartenance sont en relation directe avec la distance à la frontière (Figure 18) : d(X) correspond à son numérateur.

5.3 Equivalence avec la régression linéaire

Bien que s'inscrivant toutes deux dans le cadre de l'analyse prédictive, l'analyse discriminante linéaire et la régression linéaire multiple (Rakotomalala, R., « Econométrie – Régression linéaire simple et multiple », version 1.1, juin 2015) répondent à des problématiques différentes. La première cherche à prédire une variable cible qualitative nominale à partir d'un ensemble de variables prédictives quantitatives (ou qualitatives recodées en indicatrices numériques). Pour la seconde, la variable cible est quantitative. La finalité, les calculs sous-jacents et le mécanisme inférentiel ne sont pas les mêmes.

Pourtant, de nombreux auteurs indiquent qu'il y a des connexions fortes entre ces deux approches. Mieux même, dans le cas particulier d'une variable cible binaire (K = 2), il est possible de reproduire à l'identique les sorties de l'analyse discriminante à partir des résultats de la régression (Huberty et Olejnik, 2006, pages 353 – 355 ; Nakache et Confais, 2003, pages 14 – 16 ; Saporta, 2006, pages 451 – 452 ; Tomassone et al., 1988, pages 36 – 38). Malheureusement, si les références consultées montrent effectivement les relations entre les expressions matricielles, certaines explicitant les formules de transition, aucune ne détaille les calculs sur un exemple numérique, rendant la démonstration par trop abstraite. Le lecteur a du mal à percevoir la portée réelle de cette équivalence. A force de chercher sur le web (en français et en anglais), j'ai fini par trouver un exemple traité qui met en évidence la relation. Les coefficients des fonctions linéaires issues des deux approches sont proportionnels, hélas sans que l'auteur n'indique l'expression mathématique du rapport entre les coefficients (Desbois, 2003 ; page 31). J'ai fini par la reconstituer.

Le bénéfice de passer par une régression pour réaliser une analyse discriminante binaire est double. (1) La régression requiert moins de calculs, elle est plus rapide et occupe moins d'espace mémoire. Elle est de ce fait plus efficace pour le traitement des très grandes bases de données

si elle est bien implémentée. (2) Cette équivalence nous permet de profiter des mécanismes performants de sélection de variables propres à la régression linéaire (Rakotomalala R., « Pratique de la régression linéaire multiple – Diagnostic et sélection de variables », version 2.1, mai 2015, chapitre 3 ; voir aussi Darlington R. B., « The SWEEP Algorithm for Multiple Regression »).

Dans ce qui suit, nous reprenons une étude approfondie que j'avais décrite dans un tutoriel (TUTO 7), en l'inscrivant dans le cadre de cet ouvrage, et en travaillant sur notre base de référence DATA 1. Nous constaterons qu'avec la régression nous pourrons reproduire à l'identique non seulement les coefficients de la fonction score, mais également tous les éléments d'évaluation de pertinence globale et par variable de l'analyse discriminante linéaire.

5.3.1 Premier codage (trivial) de la variable cible

Il faut transformer Y, catégorielle, en variable numérique Z pour pouvoir utiliser la régression linéaire. Z prendrait alors 2 valeurs possibles, z_1 lorsque $(Y = y_1)$, z_2 sinon (quand $Y = y_2$).

Un codage évident de prime abord est :

$$Z = \begin{cases} z_1 = 1 \\ z_2 = 0 \end{cases}$$

L'équation de régression s'écrit :

$$Z = r'(X) = b_0 + b_1 x_1 + b_2\, x_2 + \cdots + b_p\, x_p$$

L'analogie avec la fonction score est manifeste, reste à savoir quelle relation peut-il y avoir entre les coefficients c_j et b_j, et par là entre d(X) et r'(X).

La règle d'affectation naturelle est :

Si $r'(X) > \bar{z}$ Alors Y = « g1 » sinon Y = « g2 »

Où $\bar{z}$ est la moyenne des valeurs de Z c.-à-d.

$$\bar{z} = \frac{n_1\, z_1 + n_2\, z_2}{n_1 + n_2}$$

$$\bar{z} = \frac{n_1\, z_1 + n_2\, z_2}{n}$$

Remarque 1 : Puisque $(z_1 = 1, z_2 = 0)$ ici, le seuil $\bar{z} = \frac{n_1}{n}$

Remarque 2 : Utiliser le seuil 0.5 correspond donc à un cas particulier où $(n_1 = n_2)$.

Remarque 3 : On note facilement qu'il est possible de réécrire la fonction d'affectation,

$$r(X) = (b_0 - \bar{z}) + b_1 x_1 + b_2\, x_2 + \cdots + b_p\, x_p$$

Et la règle d'affectation :

Si $r(X) > 0$ Alors Y = « g1 » sinon Y = « g2 »

Le modèle opératoire est analogue à celui de d(X).

Exemple DATA 1

Dans ce qui suit, nous appliquons le codage de Z ci-dessus, et nous lançons la régression :

```
#vecteur Z_1 (1ère version)
Z_1 <- ifelse(y=="g1",1,0)

#moyenne de Z_1
zbar <- mean(Z_1)
print(zbar)

## [1] 0.4545455

#régression linéaire
reg_1 <- lm(Z_1 ~ X1 + X2,data=X)
print(summary(reg_1))

##
## Call:
## lm(formula = Z_1 ~ X1 + X2, data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3500 -0.1652 -0.0715  0.1551  0.4209
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.73418    0.18349   4.001  0.00394 **
## X1          -0.12522    0.02747  -4.559  0.00185 **
```

```
## X2           0.12096    0.03284   3.683  0.00619 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2934 on 8 degrees of freedom
## Multiple R-squared:  0.7475, Adjusted R-squared:  0.6844
## F-statistic: 11.84 on 2 and 8 DF,  p-value: 0.004062
```

L'équation de régression s'écrit :

$$r'(X) = 0.73418 - 0.12522\, x_1 + 0.12096\, x_2$$

Nous devons la comparer à $(\bar{z} = 0.4545)$ lorsque nous classons un individu. Nous réalisons l'opération en resubstitution c.-à-d. nous appliquons le modèle sur les données qui ont servi à sa construction.

```
#prédiction numérique avec la régression
numreg_1 <- predict(reg_1,X)
print(numreg_1)

##           1           2           3           4           5           6
##  1.09705946  0.60470302  0.95906001  1.07149888  0.57914244  0.35000461
##           7           8           9          10          11
## -0.14235182  0.09104611  0.32444403 -0.16791240  0.23330565

#prédiction par application de la règle d'affectation
preg_1 <- ifelse(numreg_1 > zbar,"g1","g2")
print(preg_1)

##    1    2    3    4    5    6    7    8    9   10   11
## "g1" "g1" "g1" "g1" "g1" "g2" "g2" "g2" "g2" "g2" "g2"
```

Les classes d'appartenance ont été parfaitement reconstituées sur cet exemple.

Pour construire r(X), nous remplaçons la constante par

$$b_0 - \bar{z} = 0.73417 - 0.4545 = 0.2796$$

5.3.2 Relation entre Lambda de Wilks et le R^2 de la régression

Ouvrons une petite parenthèse à ce stade parce que nous en aurons besoin pour la correction des coefficients de la régression (section 5.3.4).

Le R^2 de la régression exprime le rapport entre la variance expliquée par le modèle et la variance totale de la variable cible. Le Λ de Wilks de la MANOVA est le rapport entre les variances

généralisées intra-classes (résiduelle, non expliquée par l'appartenance aux groupes) et totales.

Il vient la relation suivante :

$$\Lambda = 1 - R^2$$

Le R^2 fourni par la régression permet de retrouver le lambda de Wilks.

Exemple DATA 1

```
#rappel lambda de Wilks (calculé par ailleurs)
print(lambda)

## [1] 0.2524633

#R2 de la régression
print(summary(reg_1)$r.squared)

## [1] 0.7475367

#déduction du lambda à partir du R2
print(1 - summary(reg_1)$r.squared)

## [1] 0.2524633
```

5.3.3 Second codage de la variable cible

Il est possible d'obtenir directement les coefficients de r(X), y compris la constante idoine, en codant judicieusement Z de manière à ce que ($\bar{z} = 0$).

Par exemple avec (Tomassone et al., 1988, page 1988) :

$$Z = \begin{cases} z_1 = \dfrac{n_2}{n} \\ z_2 = -\dfrac{n_1}{n} \end{cases}$$

Remarque : D'autres codages sont avancés dans la littérature : ($z_1 = \frac{n}{n_1}$; $z_2 = -\frac{n}{n_2}$) (Saporta, 2006, page 451), ou encore ($z_1 = \sqrt{\frac{n_2}{n_1}}$; $z_2 = -\sqrt{\frac{n_1}{n_2}}$) (Nakache et Confais, 2003, page 14). En réalité, tout codage peut convenir pourvu que ($\bar{z} = 0$).

Exemple DATA 1

Voyons ce que donne la régression (**reg_2**) sur DATA 1 avec ce nouveau codage de Z.

```r
#effectif total
print(n)

## [1] 11

#effectifs par classe
print(n_k)

## y
## g1 g2
##  5  6

#vecteur Z_2 (2ème version)
Z_2 <- ifelse(y=="g1",n_k[2]/n,-n_k[1]/n)

#moyenne de Z - nulle cette fois-ci
zbar <- mean(Z_2)
print(zbar)

## [1] -1.012251e-17

#régression linéaire
reg_2 <- lm(Z_2 ~ X1 + X2,data=X)
print(summary(reg_2))

##
## Call:
## lm(formula = Z_2 ~ X1 + X2, data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3500 -0.1652 -0.0715  0.1551  0.4209
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.27964    0.18349   1.524  0.16602
## X1          -0.12522    0.02747  -4.559  0.00185 **
## X2           0.12096    0.03284   3.683  0.00619 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2934 on 8 degrees of freedom
## Multiple R-squared:  0.7475, Adjusted R-squared:  0.6844
## F-statistic: 11.84 on 2 and 8 DF,  p-value: 0.004062
```

Les coefficients des variables n'ont pas été modifiés, et nous avons bien la constante corrigée maintenant :

$$r(X) = 0.27964 - 0.12522\, x_1 + 0.12096\, x_2$$

Rappelons que les coefficients de la fonction score de l'ADL sur DATA 1 sont :

$$d(X) = 3.029 - 1.637\, x_1 + 1.581\, x_2$$

Nous avons donc, a priori, deux frontières différentes dans l'espace de représentation. Traçons-les dans notre espace de représentation (X1, X2) pour les situer (Figure 50).

```
#** coefficients pour frontière régression **

#pente
d1 <- -reg_2$coefficients[2]/reg_2$coefficients[3]

#origine
d0 <- -reg_2$coefficients[1]/reg_2$coefficients[3]

#tracé des frontiers pour l'ADL et la Régression
plot(X$X1,X$X2,col=c("deepskyblue3","seagreen3")[y],main="Frontières ADL et Régression",pch=19)
abline(a=c0,b=c1,lwd=1,lty=2,col="brown") #d(X)
abline(a=d0,b=d1,lwd=1,lty=4,col="blueviolet") #r(X)
```

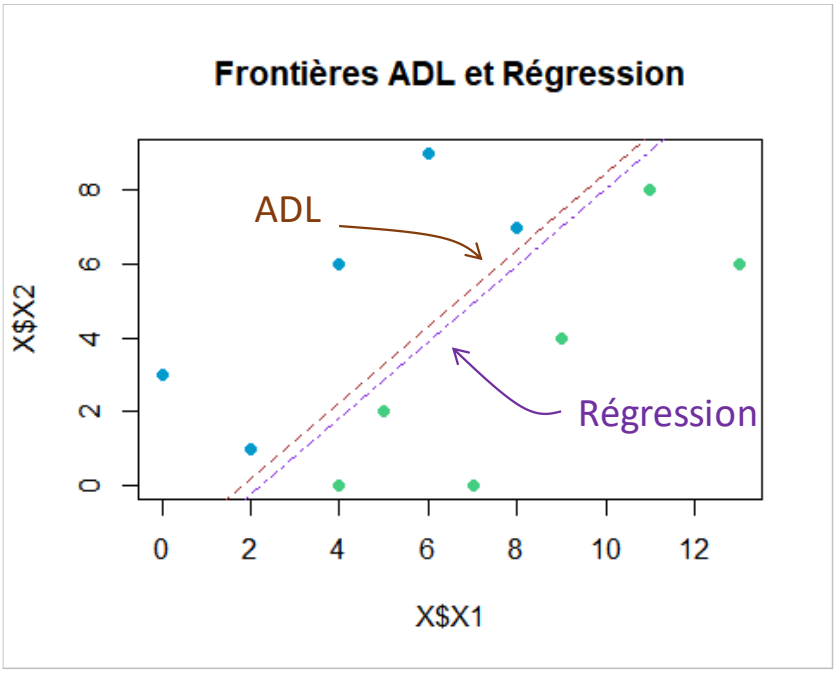


Figure 50 – Frontières de l'ADL vs. Régression – DATA 1

Nous avons deux frontières parallèles. Nous en tirons plusieurs informations très instructives bien connues dans la littérature (Hastie et al., 2017, page 110) :

- Même si sa finalité initiale est autre, réaliser une prédiction/explication pour une variable quantitative, la régression peut être utilisée pour produire un classifieur linéaire.
- Nativement, l'ADL et la régression linéaire sur variable cible recodée ne fournissent pas les mêmes classifieurs pour les problèmes binaires. Les frontières sont certes parallèles mais décalées. Appliquées sur le même échantillon test, les deux approches sont

susceptibles de produire des taux d'erreur différents. Il n'y a équivalence totale que lorsque ($n_1 = n_2$).

- Les coefficients des variables sont cependant proportionnels. Un ratio les relie. C'est ainsi que les droites de séparation présentent la même pente dans notre espace à deux dimensions.
- Il y a un décalage supplémentaire sur la constante quand ($n_1 \neq n_2$).

La vraie question est de savoir s'il est possible d'obtenir facilement ce ratio et ce décalage permettant de retrouver à l'identique les résultats de l'analyse discriminante à partir de la régression. La réponse est oui. Voyons cela dans les deux sous-sections suivantes.

5.3.4 Equivalence avec les coefficients de l'ADL

Le ratio entre les coefficients des variables de la régression (b_1, b_2, ..., b_p) et de la fonction score (c_1, c_2, ..., c_p) est défini par (TUTO 7, page 14) :

$$\rho = \frac{h_1 + h_2\, D_M^2}{h_2(z_1 - z_2)}$$

Où

$$h_1 = n_1 + n_2 - 2$$
$$h_2 = \frac{n_1 \times n_2}{n_1 + n_2}$$

- z_1 et z_2 sont les codes utilisés pour la constitution de la variable Z
- D_M^2 est la distance de Mahalanobis entre les centres de classes. Rappelons qu'il peut être déduit du Λ de Wilks lorsque (K = 2) (section 5.1), et que ce dernier peut être déterminé à partir du R^2 de la régression (section 5.3.2).

Ainsi, ρ est obtenu via uniquement les résultats de la régression. Il n'est pas nécessaire de réaliser des calculs supplémentaires qui pourraient s'apparenter à ceux de l'analyse discriminante, en particulier la formation de la matrice de variance covariance W, gourmande en espace mémoire et en temps CPU.

La relation entre les coefficients de l'ADL et la régression s'écrit alors :

$$c_j = \rho \times b_j \; ; \; j = 1, \dots, p$$

Exemple DATA 1

```
#** calcul et application du ratio reliant les coefs. de l'ADL et Régression **

#rappel distance de Mahlanobis entre classes
print(DM)

##          [,1]
## [1,] 9.771208

#codage z1
z1 <- n_k[2]/n

#codage z2
z2 <- -n_k[1]/n

#h1
h1 <- n - 2

#h2
h2 <- (n_k[1]*n_k[2])/n

#rho
rho <- (h1 + h2 * DM)/(h2*(z1-z2))
print(rho)

##          [,1]
## [1,] 13.07121

#appliquer rho aux coefficients de la régression
print(reg_2$coefficients[2:3] * as.vector(rho))

##        X1        X2
## -1.636766  1.581081

#pour rappel les coefficients de la fonction score
print(s_coef_)

##        X1        X2
## -1.636766  1.581081
```

5.3.5 Correction de la constante pour les classes non-équilibrées

Le ratio s'applique également à la constante. Mais il ne suffit pas. Une correction additionnelle est nécessaire.

$$c_0 = \rho \times b_0 + \delta$$

Et δ prend en compte, entres autres, la fréquence observée des classes (n_1, n_2) :

$$\delta = \ln\frac{n_1}{n_2} - \frac{1}{2}\sum_{j=1}^{p} c_j \times \left[\left(\bar{x}_{1j} + \bar{x}_{2j}\right) - 2 \times \bar{x}_j\right]$$

Où $(\bar{x}_j)$ est la moyenne marginale pour X_j et $(\bar{x}_{kj})$ la moyenne conditionnellement à la classe (y_k).

Remarque : On vérifie facilement que $(\delta = 0)$ lorsque $(n_1 = n_2)$ (TUTO 7, page 15).

Exemple DATA 1

Appliquons la double correction à la constante de la régression.

```
#moyenne globale pour chaque variable
xbar <- colMeans(X)
print(xbar)

##       X1       X2
## 6.272727 4.181818

#moyennes conditionnelles
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333

#effectifs conditionnels
print(n_k)

## y
## g1 g2
##  5  6

#coefficients de la fonctions score
print(s_coef_)

##        X1        X2
## -1.636766  1.581081

#constante de la régression
print(reg_2$coefficients[1])

## (Intercept)
##   0.2796368

#1ère correction -- ratio
temp <- reg_2$coefficients[1] * rho
print(temp)

##          [,1]
## [1,] 3.655191
```

```r
#calcul de delta
#possible avec un calcul matriciel mais pas très lisible dans ce cas
delta <- log(n_k[1]/n_k[2]) - 0.5 * sum(sapply(1:p,function(j){s_coef_[j]*(sum(mb_k[,j])-2*xbar[j])}))
print(delta)

##         g1
## -0.6264674

#correction finale - ajout de delta au coefficient corrigé
temp <- temp + delta
print(temp)

##          [,1]
## [1,] 3.028724

#pour rappel - la constante de la fonction score d(X)
print(s_intercept_)

##       g1
## 3.028724

CQFD…
```

Conclusion : Ainsi, dans le cas de la prédiction binaire (K = 2), nous pouvons retrouver intégralement les coefficients de l'analyse discriminante linéaire à partir de la régression linéaire multiple : en codant judicieusement la variable cible Y, en disposant des effectifs globaux et conditionnelles, des moyennes marginales et conditionnelles.

5.3.6 Inférence statistique – Tests de pertinence

L'équivalence ne s'arrête pas aux coefficients. Nous verrons dans cette section qu'il est également possible de reproduire les tests de significativité globale et individuelle des variables avec un effort calculatoire négligeable. Pour bien comprendre ce que nous manipulons, nous affichons ci-dessous les résultats détaillés de l'analyse discriminante sur DATA 1 (Figure 51).

MANOVA

Stat	Value	p-value
Wilks' Lambda	0.2525	-
Bartlett --	11.0119	0.0041
Rao -- F(2, 8)	11.8439	0.0041

LDA Summary

Classification functions			Statistical Evaluation			
Attribute	g1	g2	Wilks L.	Partial L.	F(1,8)	p-value
X1	-0.1295	1.5073	0.908304	0.277950	20.782140	0.001853
X2	0.6081	-0.9730	0.680470	0.371013	13.562590	0.006195
constant	-2.1106	-5.1393	-			

Figure 51 – Analyse discriminante – Résultats détaillés – DATA 1

Nous avons mis en surbrillance les éléments de tests de significativité qu'il faut retrouver.

Puis nous faisons de même (summary) pour la régression sur DATA 1, nous mettons en évidence les résultats que utiliserons par la suite.

```
#** évaluation globale **

#liste des propriétés de l'objet summary de la régression
sreg <- summary(reg_2)
print(attributes(sreg))

## $names
##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"
##
## $class
## [1] "summary.lm"

#affichage des résultats
print(sreg)

##
## Call:
## lm(formula = Z_2 ~ X1 + X2, data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3500 -0.1652 -0.0715  0.1551  0.4209
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.27964    0.18349   1.524  0.16602
## X1          -0.12522    0.02747  -4.559  0.00185 **
## X2           0.12096    0.03284   3.683  0.00619 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2934 on 8 degrees of freedom
## Multiple R-squared:  0.7475, Adjusted R-squared:  0.6844
## F-statistic: 11.84 on 2 and 8 DF,  p-value: 0.004062
```

Λ de Wilks. Nous pouvons le retrouver à partir du R2 de la régression (section 5.3.2) :

$$\Lambda = 1 - R^2 = 1 - 0.7475 = 0.2525$$

F de Rao. Avec (K = 2), la formule initiale (section 2.2.2) est grandement simplifiée et correspond tout simplement au F de significativité globale de la régression (Ho : tous les coefficients hors constante sont nuls) :

$$\mathrm{F} = 11{,}84, \text{ à } (\mathrm{p} = 2\,,\ \mathrm{n} - \mathrm{p} - 1 = 11 - 2 - 1 = 8) \text{ degrés de liberté}$$

Test de pertinence des variables. Dans la régression, le test de pertinence d'une variable revient à tester la significativité des coefficients (pour X_j - $H0 : b_j = 0$). La statistique « t value » suit une loi de Student à (n– p – 1) degrés de liberté. Or une loi de Student au carré correspond à une loi de Fisher (1, n –p – 1). Quand nous passons « t value » au carré, nous retrouvons la statistique F de pertinence (section 2.3). Pour la variable X1 par exemple,

$$F_1 = \left(t_{\hat{a}_j}\right)^2 = (-4.559)^2 = 20.782$$

Nous reproduisons ces calculs sous R :

```
#R2
print(sreg$r.squared)

## [1] 0.7475367

#en déduire le lambda de Wilks
print(1-sreg$r.squared)

## [1] 0.2524633

#équivalent de la stat. F de Rao
print(sreg$fstatistic)

##    value    numdf    dendf
## 11.84389  2.00000  8.00000

#p-value du F global
print(pf(sreg$fstatistic[1],sreg$fstatistic[1],sreg$fstatistic[2],lower.tail=FALSE))

##      value
## 0.08041748

#tableau des coefficients
tabCoef <- sreg$coefficients

#remplacer la colonne de t par f en passant au carré
tabCoef[,3] <- tabCoef[,3]^2

 #modifier les en-têtes de colonnes
colnames(tabCoef)[3] <- "F-Value"

#affichage
print(tabCoef)

##               Estimate Std. Error   F-Value    Pr(>|t|)
## (Intercept)  0.2796368 0.18349010  2.322541 0.166016519
## X1          -0.1252192 0.02746791 20.782138 0.001852965
## X2           0.1209591 0.03284483 13.562593 0.006194621
```

Définitivement, nous pouvons résoudre dans sa globalité un problème d'analyse discriminante linéaire avec un programme de régression linéaire multiple lorsque (K = 2).

5.4 Combinaison de classifieurs binaires pour (K > 2)

On peut produire une prédiction multiclasses (K > 2) à partir de la combinaison de classifieurs binaires. Cet artifice est nécessaire lorsque la méthode sous-jacente est nativement définie pour les cas où (K = 2) (ex. SVM – Support Vector Machine). La question ne semble pas se poser pour l'analyse discriminante puisqu'elle sait directement traiter les différentes configurations (K ≥ 2). Mais au regard de la section précédente, la possibilité de résoudre une analyse discriminante binaire à l'aide d'une régression linéaire multiple ouvre de nouvelles perspectives.

Deux stratégies sont souvent mises en avant pour combiner des classifieurs binaires.

L'approche « one-vs-one » consiste à opposer par paires les classes c.-à-d. (y_1 vs. y_2), (y_1 vs. y_3), …, (y_1 vs. y_K) ; (y_2 vs. y_3), (y_2 vs. y_4), …, (y_2 vs. y_K) ; …. ; …, (y_{K-1} vs. y_K). En déploiement, on attribue à l'individu supplémentaire la classe qui a remporté le plus de victoires. Il y a $\frac{K\,(K-1)}{2}$ régressions linéaires complètes à réaliser explicitement avec cette stratégie, sur autant de versions des données puisqu'il faut à chaque fois restreindre les observations aux paires de classes concernées. Il n'y a aucun avantage pour nous à l'adopter.

L'approche « one vs. rest » est plus intéressante dans notre contexte. En effet, nous construisons K classifieurs en opposant chaque classe aux autres : (y_1 vs. autres), (y_2 vs. autres), …, (y_K vs. autres). En déploiement, on attribue la classe qui obtient le score le plus élevé. Particularité très importante : nous travaillons toujours sur les mêmes observations, avec la même matrice X, seule le codage Z de la variable cible est modifiée. Or les coefficients de la régression sont obtenus de la manière suivante :

$$\hat{b} = (X^T X)^{-1} X^T Z$$

$(X^T X)^{-1} X^T$ est la matrice inverse généralisée de Moore-Penrose que l'on peut obtenir avec la fonction ginv() du package MASS pour R.

Toute la partie de la formule qui repose sur X peut être calculée une seule fois. Les (K) jeux de coefficients des classifieurs binaires sont obtenus en modifiant (K) fois le vecteur Z, puis en

effectuant un produit matriciel. La masse des opérations est considérablement réduite. Dans un contexte de traitement des données massives, cette stratégie peut se révéler très efficace.

Remarque 1 : Une colonne de valeurs 1 a été ajoutée à la colonne des X pour prendre en compte la constante dans le vecteur des coefficients $\hat{b}$.

Remarque 2 : En pratique, la régression est rarement programmée aussi naïvement. On passe souvent par une décomposition QR de la matrice X. Mais qui ne concerne que la matrice X encore une fois. L'opération peut être menée en amont.

Remarque 3 : Il ne faut surtout pas utiliser un simple codage 0/1 de Z (section 5.3.1). En effet, les valeurs seuils seraient différents d'un modèle à l'autre, les scores bruts ne sont pas comparables en classement. Un codage prenant en compte les effectifs des classes est plus adapté (section 5.3.3).

Remarque 4 : La stratégie « one vs. rest » est une approche viable pour traiter les problèmes multiclasses. Mais elle n'est pas sans défauts. Elle crée notamment un déséquilibre artificiel des classes pour la construction des modèles binaires. Il faut en être conscient.

Exemple DATA 2

Dans ce qui suit, nous menons une étude complète sur DATA 2 : chargement des données d'apprentissage, construction K des modèles prédictifs « one vs. rest » via la régression en appliquant très scolairement les idées ci-dessus, chargement de l'échantillon test, prédiction et évaluation des performances. Nous comparerons la matrice de confusion avec celle obtenue avec la fonction lda() dédiée à l'analyse discriminante sous R (TUTO 1).

Création des modèles prédictifs.

```
#** importation et inspection des données **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")

#inspection
str(DTrain)
```

```
## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...
```

```r
#** préparation des structures intermédiaires de calcul **

#n
n <- nrow(DTrain)
print(n)
```

```
## [1] 52
```

```r
#y
yTrain <- DTrain$TYPE

#nombre de modalités
K <- nlevels(yTrain)
print(K)
```

```
## [1] 3
```

```r
#liste des modalités
modalites <- levels(yTrain)
print(modalites)
```

```
## [1] "KIRSCH" "MIRAB"  "POIRE"
```

```r
#efectifs par classe
n_k <- table(yTrain)
print(n_k)
```

```
## yTrain
## KIRSCH  MIRAB  POIRE
##     17     15     20
```

```r
#X
XTrain <- DTrain[2:ncol(DTrain)]

#nombre de variables
p <- ncol(XTrain)
print(p)
```

```
## [1] 8
```

```r
#** construction de X et inverse généralisée **

#ajouter la colonne de 1 en première position
X <- cbind(rep(1,n),as.matrix(XTrain))

#calcul du pseudo-inverse
library(MASS)
psdX <- MASS::ginv(X)
print(dim(psdX))
```

```
## [1]  9 52
```

Ce calcul sur les X n'est réalisé qu'une seule fois. L'inverse généralisée sera alors utilisée autant de fois qu'il y a de classes dans la boucle for() qui suit.

```
## ** calcul des coefficients de régression -- 3 modèles **
## KIRSCH vs. autres, MIRAB vs. autres, POIRE vs. autres

#matrice des coefficients - 1 colonne par modèle
modeles <- matrix(0,nrow=p+1,ncol=K)
colnames(modeles) <- modalites

#construction des modèles - calcul des coefficients
for (k in modalites){
  #variable Z - seul modifié durant le processus
  Z <- ifelse(yTrain==k,(n-n_k[k])/n,-n_k[k]/n)
  #coefficients
  coef_ <- psdX %*% matrix(Z,nrow=n,ncol=1)
  #ajouter dans la matrice des coefficients
  modeles[,k] <- coef_
}

#affichage de tous les coefs
#3 modèles
#constante en première ligne
rownames(modeles) <- c('intercept',colnames(XTrain))
print(modeles)

##                  KIRSCH         MIRAB         POIRE
## intercept  5.949755e-01 -5.157471e-02 -0.5434007496
## MEOH      -7.877131e-04  2.789856e-04  0.0005087275
## ACET      -2.626890e-04  1.032444e-03 -0.0007697545
## BU1       -1.371259e-02  1.967032e-02 -0.0059577311
## BU2       -2.357645e-03 -1.940035e-03  0.0042976793
## ISOP      -3.588981e-05  3.501124e-03 -0.0034652343
## MEPR       4.177336e-03 -2.088228e-02  0.0167049396
## PRO1       2.342945e-04  6.292173e-05 -0.0002972162
## ACAL       8.374691e-03 -1.589186e-02  0.0075171724
```

Le nœud de l'affaire est la boucle for(). Seul le vecteur Z est modifié pour chaque classe (y_k). Nous réalisons un produit matriciel pour obtenir les coefficients des K modèles de régression.

Remarque : L'organisation des calculs dans une boucle permet de bien comprendre l'idée de la répétition des régressions. En réalité, nous pouvons regrouper les variables cibles recodées dans une matrice Z de dimension (n, K), et calculer les coefficients des modèles en une seule opération matricielle. La solution n'en est que plus rapide, au prix d'une augmentation de l'occupation mémoire certes. Voici le code et les résultats associés. Bien sûr, ils sont complètement cohérents avec la solution basée sur la succession de régressions.

```r
## ** calcul des coefficients de régression -- 3 modèles **
## Autre solution directement matricielle

#préparer Z sous une forme matricielle
Z <- sapply(modalites,function(k){ifelse(yTrain==k,(n-n_k[k])/n,-n_k[k]/n)})

#coefficients des modèles
modeles <- psdX %*% Z
rownames(modeles) <- c('intercept',colnames(XTrain))
print(modeles)

##                  KIRSCH         MIRAB         POIRE
## intercept  5.949755e-01 -5.157471e-02 -0.5434007496
## MEOH      -7.877131e-04  2.789856e-04  0.0005087275
## ACET      -2.626890e-04  1.032444e-03 -0.0007697545
## BU1       -1.371259e-02  1.967032e-02 -0.0059577311
## BU2       -2.357645e-03 -1.940035e-03  0.0042976793
## ISOP      -3.588981e-05  3.501124e-03 -0.0034652343
## MEPR       4.177336e-03 -2.088228e-02  0.0167049396
## PRO1       2.342945e-04  6.292173e-05 -0.0002972162
## ACAL       8.374691e-03 -1.589186e-02  0.0075171724
```

Prédiction et évaluation sur l'échantillon test.

```r
#** chargement et préparation de l'échantillon test **

#chargement de l'échantillon test
DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")

#XTest
XTest <- cbind(rep(1,nrow(DTest)),as.matrix(DTest[2:ncol(DTest)]))
print(dim(XTest))

## [1] 50  9

#** calcul des scores par classe et prédiction **

#prédiction - calcul des scores
scores <- XTest %*% modeles

#affichage des 6 premières lignes
print(head(scores))

##         KIRSCH       MIRAB      POIRE
## [1,] 0.7098484 -0.36695261 -0.3428958
## [2,] 0.5120388 -0.19930180 -0.3127370
## [3,] 0.5598029 -0.12125614 -0.4385467
## [4,] 0.5299484 -0.08327483 -0.4466735
## [5,] 0.4337062 -0.32119643 -0.1125098
## [6,] 0.7111392 -0.25993437 -0.4512048

#prediction à partir des scores
pred <- modalites[max.col(scores)]
print(pred)

##  [1] "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH"
##  [9] "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "KIRSCH" "POIRE"  "MIRAB"
## [17] "KIRSCH" "MIRAB"  "MIRAB"  "MIRAB"  "MIRAB"  "POIRE"  "MIRAB"  "MIRAB"
## [25] "MIRAB"  "MIRAB"  "MIRAB"  "POIRE"  "MIRAB"  "MIRAB"  "POIRE"  "KIRSCH"
```

```
## [33] "MIRAB"  "MIRAB"  "POIRE"  "KIRSCH" "POIRE"  "POIRE"  "MIRAB"  "POIRE"
## [41] "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "POIRE"  "MIRAB"
## [49] "POIRE"  "POIRE"

#matrice de confusion
mc <- table(DTest$TYPE,pred)
print(mc)

##         pred
##          KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       1    12     4
##   POIRE       2     4    13



#** apprentissage-test avec lda() du package MASS **

#prédiction de la fonction lda de MASS
predLda <- predict(lda(TYPE ~ ., DTrain),DTest)$class

#nouvelle matrice de confusion
mcLda <- table(DTest$TYPE,predLda)
print(mcLda)

##         predLda
##          KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       0    14     3
##   POIRE       1     5    13
```

Les résultats diffèrent de ceux de l'analyse discriminante linéaire puisque ce sont là deux manières dissemblables d'appréhender le classement multiclasses. Cette expérimentation n'a certainement pas valeur de preuve (1 seul essai sur un échantillon de taille réduite), nous nous bornons à constater que les performances ne sont pas foncièrement divergentes sur nos données (2 individus d'écarts pour 50 observations en test).

6 Naïve Bayes – Modèle d'Indépendance conditionnelle

Le bayésien naïf (naive bayes classifier en anglais ; « modèle d'indépendance conditionnelle » en français, Celeux et Nakache, 1994, section 2.3) est une méthode d'apprentissage supervisé qui repose sur une hypothèse simplificatrice forte : les descripteurs (X_j) sont deux à deux indépendants conditionnellement aux valeurs de la variable à prédire (Y). Pourtant, malgré cela, il se révèle robuste et efficace. Ses performances sont comparables aux autres techniques d'apprentissage. Diverses raisons sont avancées dans la littérature. La principale est qu'il s'agit en réalité d'un classifieur linéaire, au même titre que la régression logistique ou l'analyse discriminante prédictive qui nous préoccupe dans cet ouvrage. Ses capacités prédictives se situent – forcément serais-je tenté de dire – dans les mêmes eaux.

Etrangement, si le bayésien naïf est très utilisé dans le monde de la recherche, notamment dans le traitement des problèmes à forte dimensionnalité comme en text mining, il est en revanche très peu répandu dans le monde de l'entreprise. Dans le premier cas, les chercheurs constatent surtout qu'il est aisé de le programmer et de le mettre en œuvre, ses paramètres sont faciles à estimer, son apprentissage est très rapide y compris sur de très grandes bases en nombre d'observations et en nombre de variables, ses performances en classement sont raisonnablement bonnes. Dans le second cas, le néophyte ne disposant pas de modèle explicite lisible, facile à interpréter et à déployer, sans indications sur la pertinence des variables prédictives, ne comprend pas l'intérêt d'une telle technique.

Et pourtant tous ces mécanismes sont présents dans le bayésien naïf. Il sait produire des fonctions de classement qui s'expriment sous la forme de combinaisons linéaires des variables

prédictives. Par rapport aux autres méthodes du même ordre, il se distingue simplement par le mode d'estimation des coefficients. Et nous disposons bien d'un cadre statistique pour mesurer l'influence des prédicteurs. Malheureusement, alors que la méthode est implémentée dans de nombreux logiciels, ces aspects en sont absents. TANAGRA est le seul à ma connaissance à les proposer.

Dans ce chapitre, nous présentons la méthode prioritairement pour les descripteurs quantitatifs. Nous étudions comment en dériver une fonction de classement linéaire (TUTO 8). Nous montrons ensuite que le modèle bayésien naïf est en réalité un cas particulier de l'analyse discriminante linéaire où nous aurions procédé à une régularisation extrême. Cette relation est trop peu mise en avant dans la littérature, et pourtant elle nous donne un éclairage très intéressant sur la bonne tenue de la méthode en prédiction.

6.1 Modèle bayésien naïf

Nous nous situons toujours dans le cadre probabiliste de l'apprentissage supervisé, nous cherchons à maximiser le numérateur de la probabilité d'appartenance aux classes.

$$y_{k*} = \arg\max_k P(Y = y_k) \times P(X \,/\, Y = y_k)$$

$P(Y = y_k)$, si elle n'est pas fournie, est toujours estimée à l'aide de la proportion observées des classes dans l'échantillon de travail.

L'enjeu de la modélisation repose essentiellement sur $P(X \,/\, Y = y_k)$. Des hypothèses sont introduites pour rendre possible l'estimation de cette probabilité conditionnelle.

6.1.1 Hypothèse 0 – Indépendance conditionnelle

L'hypothèse fondatrice du bayésien naïf est l'indépendance conditionnelle des descripteurs (X_j) c.-à-d. à classe fixée, on considère que les variables prédictives sont deux à deux indépendantes. $P(X \,/\, Y = y_k)$ s'exprime alors sous la forme d'un produit de probabilités :

$$P(X/Y = y_k) = \prod_{j=1}^{p} P(X_j \,/\, Y = y_k)$$

Lorsque (X_j) est une variable quantitative, $P(X_j / Y = y_k)$ correspond à la fonction de densité $f_k(X_j)$ de sa loi de distribution.

6.1.2 Hypothèse 1 – Distribution normale conditionnelle – Modèle quadratique

Plusieurs hypothèses peuvent être émise quant aux distributions conditionnelles. La plus commune est la loi normale dont la fonction de densité s'écrit :

$$f_k(X_j) = \frac{1}{\sigma_{kj}\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x_j-\mu_{kj}}{\sigma_{kj}}\right)^2}$$

Où est μ_{kj} la moyenne de la variable X_j pour les individus de la classe (y_k), σ_{kj} l'écart-type. Ils sont estimés respectivement par $\bar{x}_{kj}$ et s_{kj} :

$$\bar{x}_{kj} = \frac{1}{n_k} \sum_{\omega:Y(\omega)=y_k} x_j(\omega)$$

$$s_{kj} = \frac{1}{n_k - 1} \sum_{\omega:Y(\omega)=y_k} \left(x_j(\omega) - \bar{x}_{kj}\right)^2$$

L'hypothèse gaussienne peut paraître contraignante. Son domaine d'application est plus large qu'il n'y paraît. Elle est viable dès lors que les distributions conditionnelles sont plus ou moins symétriques et unimodales (TUTO 8, pages 3 et 4).

En passant aux logarithmes, nous obtenons une première variante de l'expression à maximiser :

$$\ln \hat{\pi}_k + \sum_{j=1}^{p} -\frac{1}{2}\ln 2\pi - \ln s_{kj} - \frac{1}{2}\left(\frac{x_j - \bar{x}_{kj}}{s_{kj}}\right)^2$$

Comme nous maximisons par rapport (y_k), tout ce qui n'en dépend pas peut être éliminé. Nous avons ainsi une fonction de classement que l'on peut simplifier comme suit :

$$G(y_k, X) = \ln \hat{\pi}_k + \sum_j -\frac{1}{2\, s_{kj}^2} x_j^2 + \frac{\bar{x}_{kj}}{s_{kj}^2} x_j - \left(\frac{\bar{x}_{kj}^2}{2\, s_{kj}^2} + \ln s_{kj}\right)$$

Nous avons une équation quadratique mais, contrairement à l'analyse discriminante quadratique (section 1.2.1.2), sans les termes d'interactions entre les (X_j) conformément à l'hypothèse d'indépendance conditionnelle des descripteurs. Déjà à ce stade, nous pouvons produire une forme explicite du modèle qui fait intervenir les variables directement ou montés au carré.

Exemple DATA 1

Nous utilisons TANAGRA pour calculer les coefficients du modèle quadratique (Figure 52).

Classification functions		
Descriptors	**g1**	**g2**
Intercept	-5.2264	-6.3007
(X1)^2	-0.0500	-0.0411
X1	0.4000	0.6712
(X2)^2	-0.0490	-0.0469
X2	0.5098	0.3125

Figure 52 – Naive Bayes – Modèle quadratique – DATA 1

Un tel modèle est directement utilisable en déploiement. Son interprétation est un peu plus complexe en revanche, chaque variable intervenant deux fois sous des formes différentes.

6.1.3 Hypothèse 2 – Homoscédasticité – Modèle linéaire

A l'instar de l'analyse discriminante, nous introduisions une autre hypothèse pour linéariser la fonction de classement : l'homoscédasticité, les variances conditionnelles sont identiques d'une classe à l'autre

$$\sigma_{kj}^2 = \sigma_j^2 \ \ , \forall k$$

Nous utilisons la variance intra-classes pour estimer cette variance commune :

$$s_j^2 = \frac{1}{n-K} \sum_{k=1}^{K} (n_k - 1)\, s_{kj}^2$$

En introduisant cette information dans $G(y_k, X)$, nous obtenons :

$$\ln \hat{\pi}_k + \sum_j -\frac{1}{2\,s_j^2} x_j^2 + \frac{\bar{x}_{kj}}{s_j^2} x_j - \left(\frac{\bar{x}_{kj}^2}{2\,s_j^2} + \ln s_j\right)$$

De nouveau, nous élimions tout de ce qui ne dépend pas de (y_k), nous avons une fonction de classement linéaire g(y_k,X) :

$$g(y_k\,, X) = \ln \hat{\pi}_k + \sum_j \frac{\bar{x}_{kj}}{s_j^2} x_j - \frac{\bar{x}_{kj}^2}{2\,s_j^2}$$

$$g(y_k, X) = \ln \hat{\pi}_k \; - \frac{1}{2} \sum_j \frac{\bar{x}_{kj}^2}{s_j^2} + \sum_j \frac{\bar{x}_{kj}}{s_j^2} x_j$$

6.1.4 Fonctions de classements et fonctions score

La fonction de classement du bayésien naïf est une combinaison linéaire des prédicteurs :

$$g(y_k\,, X) = a_{k0} + a_{k1}\, x_1 + \cdots + a_{kp}\, x_p$$

Ses paramètres sont très faciles à estimer :

$$a_{k0} = \; \ln \hat{\pi}_k \; - \frac{1}{2} \sum_j \frac{\bar{x}_{kj}^2}{s_j^2}$$

$$a_{kj} = \frac{\bar{x}_{kj}}{s_j^2}$$

La méthode est générique, elle s'applique aux cas binaires et multiclasses (K ≥ 2). La règle d'affectation usuelle reste de mise :

$$y_{k*} = \arg\max_k g(y_k, X)$$

(K = 2). Dans le cas binaire (Y ϵ {y_1, y_2}), nous pouvons dériver une fonction score en effectuant une différence termes à termes des coefficients des fonctions de classement (section 5.2) :

$$c_j = a_{1j} - a_{2j}$$

Exemple DATA 1

Nous estimons les paramètres des fonctions de classement pour DATA 1 à l'aide du bayésien naïf. Nous en déduisons la fonction score et nous faisons apparaître la frontière de séparation des classes dans l'espace de représentation. Nous verrons comme elle se situe par rapport à celle de l'analyse discriminante.

```
#effectif total
print(n)

## [1] 11

#nombre de classes
print(K)

## [1] 2

#nombre de variables
print(p)

## [1] 2

#effectifs par classe
print(n_k)

## y
## g1 g2
##  5  6

#proportions par classe
print(pi_k)

## y
##        g1        g2
## 0.4545455 0.5454545

#moyennes des variables par classe
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333

#variance des variables par classe
var_k <- as.matrix(aggregate(X,list(y),var)[c("X1","X2")])

#pooled variance
s2 <- apply(var_k,2,function(x){sum((n_k-1)*x)/(n-K)})
print(s2)

##       X1       X2
## 11.20370 10.45926

#coefficients des fonctions de classement Naive Bayes
coef_nb_ <- apply(mb_k,1,function(x){x/s2})
print(round(coef_nb_,6))

##        [,1]     [,2]
## X1 0.357025 0.728926
## X2 0.497167 0.318697
```

```r
#intercept
intercept_nb_ <- log(pi_k) - 0.5 * colSums(apply(mb_k,1,function(x){x^2/s2}))
print(round(intercept_nb_,6))

## y
##        g1        g2
## -2.795142 -4.113744
```

Voici les fonctions de classement issues du bayésien naïf sur DATA 1 :

$$g(g_1, X) = -2.795142 + 0.357025\, x_1 + 0.497167\, x_2$$
$$g(g_2, X) = -4.113744 + 0.728926\, x_1 + 0.318697\, x_2$$

A titre de comparaison, voici les sorties de TANAGRA sur les mêmes données (Figure 53) :

Classification functions		
Descriptors	**g1**	**g2**
Intercept	-2.795142	-4.113744
X1	0.357025	0.728926
X2	0.497167	0.318697

Figure 53 – Naive Bayes Linéaire sur DATA 1 – TANAGRA

Nous formons les coefficients de la fonction score...

```r
#fonction score - coefficients
snb_coef_ <- coef_nb_[,1] - coef_nb_[,2]
print(round(snb_coef_,6))

##        X1        X2
## -0.371901  0.178470

#intercept
snb_intercept_ <- intercept_nb_[1] - intercept_nb_[2]
print(round(snb_intercept_,6))

##       g1
## 1.318602
```

..., elle s'écrit :

$$g(X) = 1.138602 - 0.371901\, x_1 + 0.178470\, x_2$$

Nous pouvons tracer la frontière dans l'espace de représentation.

```r
#équation explicite pour la forntière
e1 <- -snb_coef_[1]/snb_coef_[2] #pente
print(e1)

##       X1
## 2.083825
```

```
e0 <- -snb_intercept_/snb_coef_[2] #constante
print(e0)

##        g1
## -7.388358

#représentation des points et des frontières
plot(X$X1,X$X2,col=c("deepskyblue3","seagreen3")[y],main="Frontières ADL et Naive Bayes",pch=19)
abline(a=c0,b=c1,lwd=1,lty=2,col="brown") #d(X)
abline(a=e0,b=e1,lwd=1,lty=4,col="darkcyan") #g(X)
```

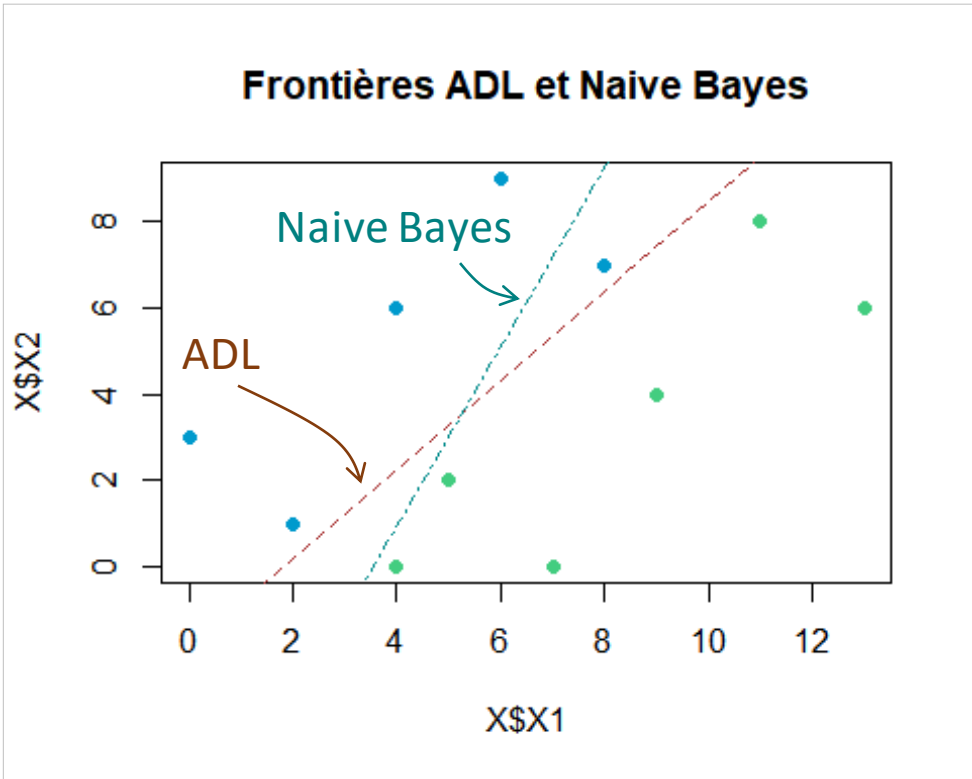


Figure 54 – Frontières de l'ADL et du Bayésien Naïf – DATA 1

Les frontières diffèrent sensiblement. L'allongement selon une pente commune est caractéristique d'une covariance intra-classe forte des variables X1 et X2 sur les données DATA 1. Or, le bayésien naïf ne prend pas en compte la direction des nuages de points. Il est faussé et classe même mal 1 des individus de l'échantillon d'apprentissage. L'analyse discriminante, elle, exploite cette information dans la matrice W, d'où son bon comportement sur ce jeu de données.

6.1.5 Pertinence des prédicteurs

Une variable (X_j) contribue à la discrimination si ses coefficients a_{kj} sont significativement différents d'une classe à l'autre. On peut mettre en place un test d'hypothèses avec :

$$H_0 : a_{1j} = \cdots = a_{kj} = \cdots = a_{Kj}$$

$$H_1 : \text{un coefficient au moins diffère des autres}$$

Or, rappelons que le coefficient est estimé par :

$$a_{kj} = \frac{\overline{x}_{kj}}{s_j^2}$$

s_j^2 est commun à l'ensemble des classes. L'évaluation de la pertinence se résume à une comparaison de moyennes conditionnelles. Formellement, le test devient :

$$H_0 : \mu_{1j} = \cdots = \mu_{kj} = \cdots = \mu_{Kj}$$

$$H_1 : \text{une des moyennes au moins diffère des autres}$$

Une comparaison de K moyennes avec l'hypothèse d'homoscédasticité est un schéma typique d'analyse de variance (ANOVA) à 1 facteur (Rakotomalala, 2010, section 1.3).

La statistique de test exprime le rapport entre la variabilité expliquée et résiduelle, corrigées par les degrés de liberté. Sous Ho, ...

$$F_j = \frac{\dfrac{\sum_{k=1}^{K} n_k \left(\bar{x}_{kj} - \bar{x}_j\right)^2}{K-1}}{\dfrac{\sum_{k=1}^{K} \sum_{\omega : Y(\omega) = y_k} \left(x_j(\omega) - \bar{x}_{kj}\right)^2}{n-K}}$$

$$F_j = \frac{\dfrac{\sum_{k=1}^{K} n_k \left(\bar{x}_{kj} - \bar{x}_j\right)^2}{K-1}}{s_j^2}$$

... suit une loi de Fisher à (K – 1, n – K) degrés de liberté. La région critique est constituée par les valeurs élevées de F.

Exemple DATA 1

Etudions l'impact des variables dans le modèle bayésien naïf pour DATA 1.

```
#effectif
print(n)

## [1] 11
```

```r
#nombre de variables
print(p)

## [1] 2

#nombre de classes
print(K)

## [1] 2

#effectifs par classe
print(n_k)

## y
## g1 g2
##  5  6

#moyenne globale
print(xbar)

##       X1       X2
## 6.272727 4.181818

#moyennes conditionnelles
print(mb_k)

##            X1       X2
## [1,] 4.000000 5.200000
## [2,] 8.166667 3.333333

#pooled variance par variable
print(s2)

##       X1       X2
## 11.20370 10.45926

#numérateur de la stat. de test
FNumNb <- sapply(1:p,function(j){sum(n_k*(mb_k[,j]-xbar[j])^2)})/(K-1)
print(FNumNb)

## [1] 47.34848  9.50303

#stat de test
FStat <- FNumNb / s2
print(round(FStat,6))

##       X1       X2
## 4.226146 0.908576

#p-value
print(round(pf(FStat,K-1,n-K,lower.tail=FALSE),6))

##       X1       X2
## 0.069959 0.365390
```

Ils concordent avec les résultats de TANAGRA encore une fois (Figure 55).

Classification functions			Tests	
Descriptors	g1	g2	F(1,9)	p-value
Intercept	-2.795142	-4.113744	-	-
X1	0.357025	0.728926	4.226146	0.069959
X2	0.497167	0.318697	0.908576	0.365390

Figure 55 – Test de pertinence des variables – Naive Bayes – DATA 1

Au risque 10%, seule la variable X1 ait un réel impact dans la fonction de classement du bayésien naïf. Ce résultat est logique si l'on considère la pente de la frontière de séparation des classes (Figure 54). Elle est quasiment à la verticale.

6.2 Lien avec l'analyse discriminante

Quel rapport y a-t-il entre l'analyse discriminante linéaire et le bayésien naïf ? Comparons les deux fonctions de classement avec respectivement $d(y_k,X)$ et $g(y_k,X)$:

$$d(y_k\,,X) = \ln \hat{\pi}_k - \frac{1}{2}\,\bar{x}_k W^{-1}\bar{x}_k^T + \bar{x}_k W^{-1} x^T$$

$$g(y_k\,,X) = \ln \hat{\pi}_k \; - \frac{1}{2}\sum_j \frac{\bar{x}_{kj}^2}{s_j^2} + \sum_j \frac{\bar{x}_{kj}}{s_j^2}\,x_j$$

La relation saute aux yeux : le bayésien naïf pour variables quantitatives, avec l'hypothèse gaussienne et d'homoscédasticité, est un cas particulier de l'analyse discriminante linéaire où la matrice W est réduite à une matrice diagonale composée des variances intra-classes des variables. La notion d'indépendance conditionnelle prend toute sa signification. La covariance intra-classe des couples de variables, qui est une somme pondérée des covariances pour chaque classe (normalisé par les degrés de liberté), est nulle par hypothèse. D'une certaine manière, on peut le voir comme une forme de régularisation extrême de l'analyse discriminante (Hastie et al., 2017, sections 4.3.1 et [surtout] 18.2).

Exemple DATA 1

Nous reprenons les calculs des fonctions de classement de l'analyse discriminante décrite en section 1.3.1, mais en mettant à zéro les éléments non diagonaux de la matrice de variance covariance intra-classe W. Nous devrions retrouver les coefficients du modèle bayésien naïf.

```r
#matrice W initiale
print(W)

##           X1        X2
## X1 11.203704  8.962963
## X2  8.962963 10.459259

#diagonale de W
print(diag(W))

##       X1       X2
## 11.20370 10.45926

#variances intra-classes par calcul direct (cf. ci-dessus - section 6.1.4)
#on a bien les mêmes valeurs
print(s2)
##       X1       X2
## 11.20370 10.45926

#matrice W avec hypothèse Naive Bayes
WNb <- matrix(0,p,p)
diag(WNb) <- diag(W)
print(WNb)

##         [,1]     [,2]
## [1,] 11.2037  0.00000
## [2,]  0.0000 10.45926

#inversion de la matrice
#l'inverse d'un matrice diagonale
#est une matrice diagonale avec les valeurs inversées
invWNb <- solve(WNb)
print(invWNb)

##           [,1]       [,2]
## [1,] 0.0892562 0.00000000
## [2,] 0.0000000 0.09560907

#** coefficients de l'ADL avec cette hypothèse naive bayes **

#calul matriciel pour les coefficients
#de la fonction de classement
coef_adl_nb_ <- t(mb_k %*% invWNb)

#etiquetage de l'en-tête de colonne
colnames(coef_adl_nb_) <- levels(y)

#affichage des coefficients calculés
print(round(coef_adl_nb_,6))

##            g1       g2
## [1,] 0.357025 0.728926
## [2,] 0.497167 0.318697

#calcul de la constante
intercept_adl_nb_ <- log(pi_k)-0.5*diag(mb_k %*% invWNb %*% t(mb_k))
print(round(intercept_adl_nb_,6))

## y
##        g1        g2
## -2.795142 -4.113744
```

Ce sont bien là les coefficients (celles des variables et les constantes) des fonctions de classement du bayésien naïf que nous avions obtenu par calcul direct (Figure 53).

6.3 Cas des prédicteurs qualitatifs

6.3.1 Principe

Lorsque les descripteurs sont catégoriels, il n'est plus question d'estimer les probabilités conditionnelles $P(X_j / Y = y_k)$ par des fonctions de densité. On passe par les fréquences conditionnelles (TUTO 9, section 2.1). Pour une variable X_j, comportant M_j modalités, la probabilité que (X_j = m) sachant (Y = y_k) est estimé par :

$$P(X = m / Y = y_k) = \frac{n_{km}}{n_k}$$

Où n_{km} est le nombre d'observations présentant les caractéristiques (X_j = m, Y = y_k).

Parfois, surtout lorsque nous travaillons sur des petits effectifs, un paramètre de lissage (β) est utilisé, on parle alors d'estimateur Laplacien des probabilités (Laplace Smoothing). Il a au moins le mérite d'éviter les estimations de fréquences nulles lorsqu'aucun individu ne correspond à la combinaison (X_j = m, Y = y_k) :

$$P(X_j = m / Y = y_k) = \frac{n_{km} + \beta}{n_k + \beta \times M_j}$$

Le score d'un individu est obtenu en additionnant les logarithmes des fréquences conditionnelles, sans oublier la probabilité a priori :

$$g(y_k, X) = \ln \hat{\pi}_k + \sum_{j=1}^{p} \ln P(X_j / Y = y_k)$$

Exemple VOTE

Nous préparons les tableaux de calculs pour les données « VOTE » (section 3.1) où, rappelons-le, l'objectif est de prédire le groupe d'appartenance politique à partir des votes effectués sur

deux thématiques « adoption du budget », « dépenses d'éducation ». Puis nous classerons manuellement un individu pour comprendre le mécanisme d'affectation.

```
#** importation des données **

#lecture
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "VOTE")

#inspection des données
str(D)

## 'data.frame':    435 obs. of  7 variables:
##  $ group       : Factor w/ 2 levels "democrat","republican": 2 2 1 1 1 1 1 2 2
1 ...
##  $ adop_budget : Factor w/ 3 levels "n","u","y": 1 1 3 3 3 3 1 1 1 3 ...
##  $ edu_spending: Factor w/ 3 levels "n","u","y": 3 3 1 1 2 1 1 1 3 1 ...
##  $ adop_n      : num  1 1 0 0 0 0 1 1 1 0 ...
##  $ adop_u      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ edu_n       : num  0 0 1 1 0 1 1 1 0 1 ...
##  $ edu_u       : num  0 0 0 0 1 0 0 0 0 0 ...

#distribution a priori des classes
#passage au log
p_group <- log(prop.table(table(D$group)))
print(round(p_group,2))

##
##   democrat republican
##      -0.49      -0.95

#ln[P(adop_budget/group)]
p_adop <- log(prop.table(table(D$group,D$adop_budget),margin=1))
print(round(p_adop,2))

##
##                  n     u     y
##   democrat   -2.22 -3.64 -0.14
##   republican -0.17 -3.74 -2.03

#ln[P(edu_spending/group)]
p_edu <- log(prop.table(table(D$group,D$edu_spending),margin=1))
print(round(p_edu,2))

##
##                  n     u     y
##   democrat   -0.23 -2.70 -2.00
##   republican -2.13 -2.56 -0.22
```

Pour déterminer la classe d'affectation d'un individu ayant voté (adop_budget = 'n', edu_spending = 'y'), nous effectuons le double calcul :

$$g(democrat, X) = \ln P(democrat) + \ln P(adop = n/democrat) + \ln P(edu = y/democrat)$$

$$g(republican, X) = \ln P(republican) + \ln P(adop = n/republican) + \ln P(edu = y/republican)$$

```r
#scores pour individu (adop=n, edu=y)

#democrat
s_democrat <- p_group['democrat']+p_adop['democrat','n']+p_edu['democrat','y']
print(s_democrat)

## democrat
## -4.71178

#republican
s_republican <- p_group['republican']+p_adop['republican','n']+p_edu['republican','y']
print(s_republican)

## republican
##  -1.338208

#proba. democrat
print(exp(s_democrat)/(exp(s_democrat)+exp(s_republican)))

##  democrat
## 0.0331317

#proba. republican
print(exp(s_republican)/(exp(s_democrat)+exp(s_republican)))

## republican
##  0.9668683
```

Nous obtenons les scores, puis les probabilités d'affectation via la transformation softmax. Il y a de très fortes chances que l'individu soit un 'republican'.

Nous le constatons, la méthode est très facile à appréhender, à programmer, à déployer, elle peut traiter activement de très grandes volumétries. Et son efficacité prédictive est dans la moyenne des classifieurs linéaires. Son succès en recherche repose sur ces qualités.

Les présentations dévolues au bayésien naïf dans le cadre des prédictives qualitatives s'arrêtent à ce stade généralement (Celeux et Nakache, 1994, section 2.3, Hastie et al., 2012, section 6.6.3). Or, le plus intéressant commence ici. Il est également possible de produire une fonction de classement explicite dans le contexte des prédicteurs catégoriels.

6.3.2 Passage aux indicatrices et fonction de classement explicite

Passer par les indicatrices permet d'écrire le modèle prédictif sous la forme de fonctions de classement lisibles (TUTO 9, section 2.3). Pour notre exemple « VOTE », « adop_budget » et « edu_spending » sont transformés en (adop_n, adop_u, adop_y) et (edu_n, edu_u, edu_y), nous avons :

$$g(democrat, X) = -0.49 - 2.22\ adop_n - 3.64\ adop_u - 0.14\ adop_y - 0.23\ edu_n - 2.70\ edu_u - 2.00\ edu_y$$

$$g(republican, X) = -0.95 - 0.17\ adop_n - 3.74\ adop_u - 2.03\ adop_y - 2.13\ edu_n - 2.56\ edu_u - 0.22\ edu_y$$

Les avantages sont multiples :

- Le sens et l'intensité des relations entre les descripteurs et la variable cible sont plus facilement interprétables. D'autant plus qu'ayant affaire à exclusivement des indicatrices, les coefficients sont directement comparables.
- Cette lisibilité est encore meilleure lorsque nous pouvons dériver une fonction score dans le cas à (K = 2) classes (TUTO 9, section 2.5). Si, pour simplifier, « democrat » constitue la modalité cible, nous observons que les modalités (adop_budget = 'u' et edu_spending = 'u') ont très peu d'impact dans la désignation du groupe d'appartenance ; on est plus démocrate lorsqu'on a voté (adop_budget = 'y' et edu_spending = 'n'), républicain pour (adop_budget = 'n' et edu_spending = 'y'). Ces conclusions rejoignent ceux de l'analyse discriminante sur indicatrices (Figure 37) ou sur facteurs d'analyse des correspondances multiples (DISQUAL) (Figure 45).

$$g(X) = 0.46 - 2.05\ adop_n + 0.10\ adop_u + 1.89\ adop_y + 1.90\ edu_n - 0.14\ edu_u - 1.79\ edu_y$$

- Les coefficients des indicatrices se lisent comme des logarithmes de risques relatifs (Rakotomalala R., « Dépendance des variables qualitatives », version 2.1, avril 2020, section 5.3). Pour (adop = y) par exemple, en nous référant aux valeurs de la (Figure 32), nous avons :

$$\ln \frac{P(adop = y/group\ = democrat)}{P(adop = y/group\ = republican)}$$
$$= \ln 0.8652 - \ln 0.1310 = 1.89$$

  La probabilité de voter (adop = y) est ($e^{1.89} = 6.6$) fois plus élevée chez un démocrate que chez un républicain.

- L'expérimentation montre qu'utiliser des indicatrices permet de passer par les calculs matriciels lors du déploiement, sous R tout du moins. Les temps de calculs en sont

spectaculairement (mais vraiment !) améliorés pour le traitement et le déploiement sur des bases volumineuses (« Implémentation du Naive Bayes sous R », décembre 2019).

Comme pour les prédicteurs numériques, un mécanisme d'évaluation de la pertinence des variables peut être mis en place pour le classifieur bayésien naïf. Il repose toujours sur le contraste entre les coefficients des variables dans les fonctions de classement (Rakotomalala R., « Le classifieur bayésien naïf (Modèle d'indépendance conditionnelle) »).

6.4 Analyse discriminante barycentrique

L'analyse discriminante barycentrique (Nakache et Confais, 2003, chapitre 3) est une variante, additive et inversée, du modèle bayésien naïf pour les descripteurs catégoriels.

Elle prend pour point de départ le tableau de contingence croisant les modalités de la cible avec ceux de l'ensemble des variables explicatives. Pour les données « VOTE », voici la matrice :

	adop_budget			edu_spending		
	adop_n	adop_u	adop_y	edu_n	edu_u	edu_y
democrat	29	7	231	213	18	36
republican	142	4	22	20	13	135

Figure 56 – Tableau de contingence pour l'analyse discriminante barycentrique – VOTE

Dans un problème à deux classes, mettons que la première (y_1) est la modalité cible (« democrat » pour notre exemple VOTE). La représentation barycentrique des variables correspond au vecteur des probabilités conditionnelles $P(Y = y_1 \,/\, X_j = m)$ où « m » représente une des modalités de la variable Xj. Le tableau de travail devient :

	adop_budget			edu_spending		
	adop_n	adop_u	adop_y	edu_n	edu_u	edu_y
P(dem / modalite)	0.17	0.64	0.91	0.91	0.58	0.21

Figure 57 – Représentation barycentrique des modalités-variables – VOTE

Pour obtenir la coordonnée de l'individu ω, nous effectuons une moyenne non-pondérée de ces probabilités (Celeux et Nakache, 1994, section 5.3.3, équation 5.3) :

$$\frac{1}{p}\sum_{j=1}^{p} P(Y = y_1 \,/\, X_j(\omega))$$

Où $Xj(\omega)$ désigne la modalité de la variable X_j prise par l'individu ω.

L'analogie avec le bayésien naïf saute aux yeux, sauf que nous utilisons directement P(Y/X) et non pas P(X/Y), et que nous effectuons une somme directe et non pas une somme des logarithmes. Il n'en reste pas moins que, pour ce qui est des données VOTE, nous percevons d'ores et déjà l'influence relative des modalités-variables dans la désignation de la classe « democrat » à la lecture des valeurs du tableau de la Figure 57. A bien y regarder, en les divisant par (p), nous disposons d'une fonction d'affectation basée sur les indicatrices :

$$h(X) = 0.08\ adop_n + 0.32\ adop_u + 0.46\ adop_y + 0.46\ edu_n + 0.29\ edu_u + 0.11\ edu_y$$

Reste la question du seuil d'affectation. Il n'existe pas de formule analytique pour le spécifier. La stratégie conseillée consiste à le déterminer empiriquement de manière à minimiser le taux d'erreur sur l'échantillon d'apprentissage (Celeux et Nakache, 1994, page 129).

Lorsque (K > 2), le tableau de contingence de départ reste le même. Une analyse factorielle des correspondances (Rakotomalala R., « Analyse factorielle des correspondances – Diapos », juillet 2013) lui est appliquée (Nakache et Confais, 2003, section 3.2). Nous disposons alors d'un repère factoriel où peuvent être représentés les points « classes » et « modalités-variables ». Pour prédire sur un individu supplémentaire, nous calculons ses coordonnées factorielles et nous lui attribuons la classe dont le représentant est le plus proche. Nous sommes dans une démarche intrinsèquement descriptive. Nous nous y attarderons plus longuement dans le chapitre dédié (chapitre 9).

7 Régularisation

La fragilité de l'analyse discriminante linéaire réside dans l'estimation de la matrice de variance covariance W. Lorsqu'elle est quasi-singulière, ce qui arrive lorsque les variables sont fortement corrélées, ou lorsque la dimensionnalité est élevée et que (p), nombre de variables, se rapproche dangereusement de (n), nombre d'observations, les résultats sont fortement erratiques, sur-dépendantes de l'échantillon d'apprentissage. Les capacités prédictives en généralisation sont altérées.

Dans ce chapitre, nous passerons rapidement sur une première solution directe qui consiste à manipuler la matrice W pour stabiliser les calculs. Puis nous nous attarderons sur les approches basées sur la combinaison de l'ADL avec des méthodes factorielles. L'idée est de produire un espace de représentation intermédiaire décrit par des variables latentes, moins soumis à la variabilité, et d'élaborer le modèle prédictif sur ces facteurs. L'enjeu par la suite est de pouvoir exprimer les fonctions de classement à partir des variables originelles.

7.1 Shrinkage

Les solutions « mécaniques » manipulent la matrice de W de manière à atténuer le rôle des valeurs hors diagonale principale. On parle de « shrinkage » c.-à-d. les valeurs hors diagonale sont rétrécies, ramenées vers zéro, avec plus ou moins d'intensité, de manière à se rapprocher de la matrice diagonale composée exclusivement des variances intra-classes des variables (Hastie et al., 2017, section 4.3.1 ; voir aussi la documentation de « scikit-learn », section 1.2.4 « Shrinkage »). C'est en ce sens que la méthode bayésienne naïve est une forme de

régularisation extrême de l'analyse discriminante (Hastie et al., 2017, section 18.2). La version modifiée (shrinkée) de la matrice W s'écrit :

$$W_\gamma = \gamma\, W + (1 - \gamma)\, s^2\, I$$

Où γ est le paramètre de régularisation ($\gamma = 1$, ADL « normale » ; $\gamma = 0$, bayésien naïf) ; s^2 est le vecteur des variances intra-classes ; I est la matrice identité.

La valeur du paramètre γ joue un rôle essentiel dans ce procédé. Il existe des propositions d'expressions analytiques (Ledoit O., Wolf M., 2004), mais la formule n'est pas des plus accessibles. Sinon, elle peut être optimisée par validation croisée via les outils de « grid-search » que l'on retrouve dans certaines librairies de machine learning (ex. « scikit-learn »).

7.2 Analyse discriminante sur facteurs de l'ACP

7.2.1 Principe de l'analyse régularisée par axes principaux

L'analyse régularisée par axes principaux (Lebart et al., 2000, section 3.3.6.b) consiste à combiner l'analyse en composantes principales (ACP) avec l'analyse discriminante linéaire. Finalement, elle fonctionne sur le même schéma que DISQUAL (section 3.3), sauf que les variables originelles sont déjà quantitatives et que la méthode factorielle vise avant tout à lisser les données afin de ne conserver que l'information utile pour le classement (« Analyse discriminante sur axes principaux », janvier 2004). Le paramétrage repose sur le bon nombre NB_FACT des premiers facteurs de l'ACP à présenter à l'ADL. Ici également, une optimisation par validation croisée peut très bien faire l'affaire.

L'analyse en composantes principales est une technique de réduction de dimensionnalité et de data visualisation très populaire. Il n'est pas question de développer la méthode ici, de très nombreux supports de cours s'en chargent très bien, j'y ai moi-même contribué (Rakotomalala R., « Analyse en composantes principales – Diapos », juillet 2013). Dans la section suivante, nous retraçons chaque étape du processus « ACP + ADL » en traitant les données DATA 2 à l'aide du logiciel TANAGRA. Nous comparerons alors les coefficients des fonctions de classement issus de cette approche avec ceux d'une analyse discriminante directe.

7.2.2 Un exemple de traitements – DATA 2

Pour rappel, les données DATA 2 / TRAIN sont composés de (n = 52) observations, (K = 3) classes (TYPE ∈ {KIRSCH, MIRAB, POIRE}), et (p = 8) variables prédictives.

7.2.2.1 Analyse en composantes principale normée

L'ACP ne tient pas compte des classes d'appartenance, nous la lançons sur la matrice des descripteurs incluant l'ensemble des observations de l'échantillon d'apprentissage.

Le tableau des valeurs propres semble indiquer 4 facteurs pertinents pour la représentation des données, si l'on s'en tient au critère de Kaiser Guttman (valeur propre > 1).

Axis	Eigen value	Difference	Proportion (%)	Cumulative (%)
1	2.7988	1.0799	34.98%	34.98%
2	1.7188	0.3154	21.49%	56.47%
3	1.4034	0.3590	17.54%	74.01%
4	1.0444	0.5076	13.06%	87.07%
5	0.5368	0.3344	6.71%	93.78%
6	0.2024	0.0173	2.53%	96.31%
7	0.1851	0.0748	2.31%	98.62%
8	0.1103	-	1.38%	100.00%
Tot.	8	-	-	-

Figure 58 – Tableau des valeurs propres – ACP normée – DATA 2 / TRAIN

Si on utilise le critère de Karlis-Saporta-Spinaki, qui présente l'avantage de tenir compte de la conformation des données (n et p), la valeur seuil serait (Saporta, 2006, section 7.3.2.2) :

$$1 + 2\sqrt{\frac{p-1}{n-1}} = 1.74096$$

Seul le premier facteur serait adéquat, le second échoue d'un souffle.

Le décrochement dans l'éboulis des valeurs propres semble plutôt suggérer 2 composantes (Figure 59). Ce résultat nous arrange au moins pour les représentations graphiques. Mais il est évident que ce choix des facteurs à utiliser pour l'analyse discriminante subséquente est crucial pour les performances à venir du système.

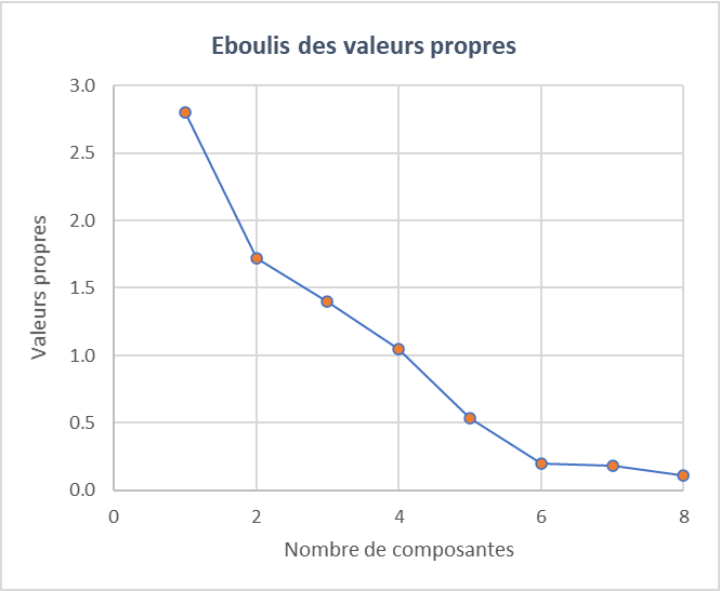


Figure 59 – Eboulis des valeurs propres – ACP – DATA 2 / TRAIN

Le cercle des corrélations nous indique que le premier facteur est plutôt déterminé par les variables (MEOH, BU1, ISOP, MEPR), le second par (BU2, PRO1) (Figure 60).

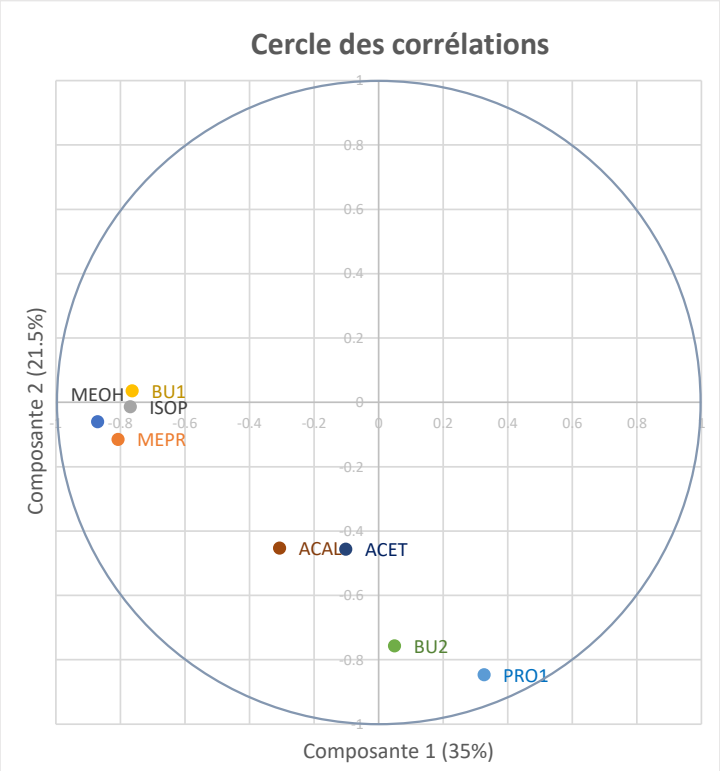


Figure 60 – Cercle des corrélations – DATA 2 / TRAIN

Le nuage des points étiquetés est très important pour comprendre le rôle des axes dans la discrimination (Figure 61). Des indicateurs numériques peuvent le caractériser. Nous les étudierons en détail dans l'analyse discriminante descriptive (chapitre 8). Dans notre exemple,

c'est le premier axe qui semble primer pour distinguer les classes : KIRSCH se démarque, il y a un recouvrement en revanche entre MIRAB et POIRE.

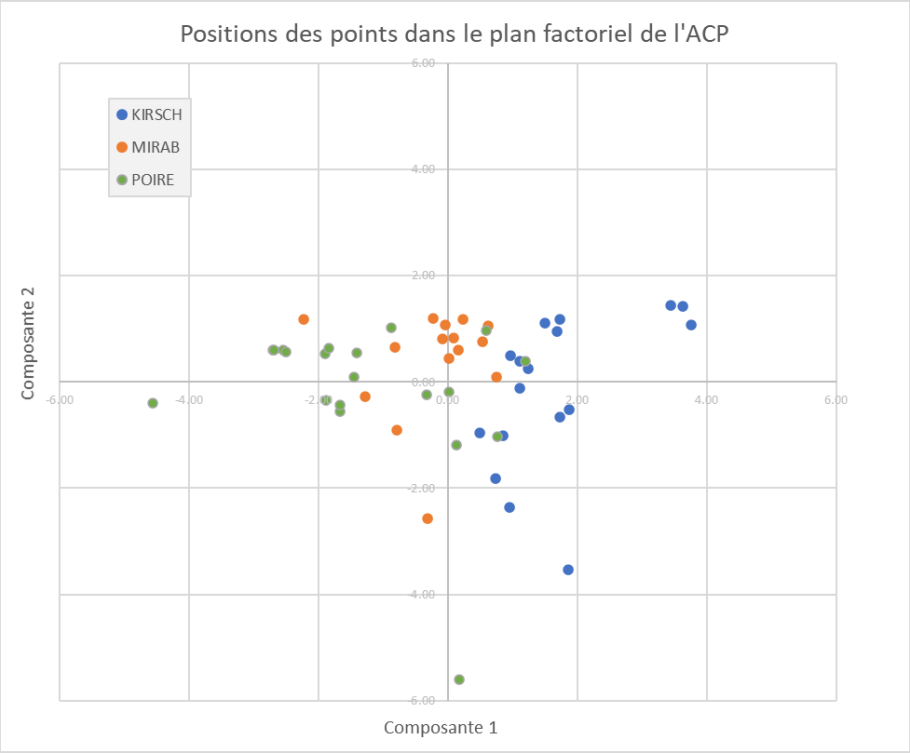


Figure 61 – Observations étiquetées dans le premier plan factoriel de l'ACP – DATA 2 / TRAIN

Le tableau « Factor Score Coefficients » joue un rôle fondamental pour la suite. Il fournit les coefficients de la fonction permettant de projeter les individus dans le repère factoriel. Il est défini sur les variables centrées et réduites, les paramètres de centrage (moyenne) et de réduction (écart-type) l'accompagne (Figure 62).

Attribute	Mean	Std-dev	Axis_1	Axis_2
MEOH	808.0	366.2	-0.52	-0.05
ACET	205.4	121.0	-0.06	-0.35
BU1	14.4	10.9	-0.46	0.03
BU2	29.8	54.0	0.03	-0.58
ISOP	98.3	47.8	-0.46	-0.01
MEPR	37.2	18.3	-0.48	-0.09
PRO1	436.9	620.1	0.20	-0.65
ACAL	13.1	8.0	-0.18	-0.34

Figure 62 – Factor score coefficients – ACP – DATA 2 / TRAIN

Pour la première composante :

$$Z_1 = -0.52 \times \left(\frac{MEOH - 808.0}{366.2}\right) + \cdots - 0.18 \times \left(\frac{ACAL - 13.1}{8.0}\right)$$

Laisser les coefficients standardisés (qui s'appliquent sur les variables centrées et réduites) permet de comparer l'influence des descripteurs. Ils sont en relation directe avec les corrélations (Figure 60), (MEOH, BU1, ISOP et MEPR) sont ceux qui pèsent le plus sur le premier facteur.

Pour faciliter les opérations subséquentes, nous exprimons ces composantes avec des coefficients non-standardisés :

$$Z_1 = \frac{-0.52}{366.2}\,MEOH + \cdots - \frac{0.18}{8.0}\,ACAL + \left(\frac{-0.52 \times\ -808.0}{366.2} + \cdots + \frac{-0.18 \times\ -13.1}{8.0}\right)$$
$$Z_1 =\ -0.0014\,MEOH + \cdots - 0.0230\,ACAL + 3.9272$$

De même pour le second facteur :

$$Z_2 = -0.001\,MEOH + \cdots - 0.0432\,ACAL + 2.1923$$

7.2.2.2 Analyse discriminante sur facteurs

Nous travaillons sur un cas d'école. La Figure 61 nous renseigne très précisément sur le pouvoir discriminatoire des facteurs. En pratique, nous n'avons pas des résultats aussi évidents. Nous devons fonder le choix du nombre de facteurs sur exclusivement les informations non-supervisées de l'ACP. Mettons que nous souhaitons utiliser (NB_FACT = 2) composantes dans l'analyse discriminante. Voici les sorties de TANAGRA (Figure 63).

MANOVA

Stat	Value	p-value
Wilks' Lambda	0.4279	-
Bartlett -- C(4)	41.172	0.0000
Rao -- F(4, 96)	12.6901	0.0000

LDA Summary

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,48)	p-value
PCA_1_Axis_1	1.2748	-0.1782	-0.9499	0.9608	0.4454	29.8899	0.0000
PCA_1_Axis_2	-0.1129	0.2359	-0.0810	0.4455	0.9604	0.9909	0.3787
constant	-2.1999	-1.3116	-1.5608		-		

Figure 63 – ADL sur facteurs de l'ACP – DATA 2 / TRAIN

La discrimination est globalement significative à 5% (F de Rao et p-value associée). Confirmant l'intuition visuelle ci-dessus (Figure 61), nous aurions pu nous contenter du premier facteur qui paraît seul pertinent pour discriminer les groupes.

Nous disposons de 3 fonctions de classement :

$$d(KIRSCH, Z) = 1.2748\, Z_1 - 0.1129\, Z_2 - 2.1999$$
$$d(MIRAB, Z) = -0.1782\, Z_1 + 0.2359\, Z_2 - 1.3116$$
$$d(POIRE, Z) = -0.9499\, Z_1 - 0.0810\, Z_2 - 1.5608$$

7.2.2.3 Retour sur les variables originelles

Pour exprimer les fonctions de classement à partir des variables originelles, il faut remplacer les Z par leur formule dans la projection de l'ACP. Pour la classe KIRSCH par exemple :

$$d(KIRSCH, X) = 1.2748 \times (-0.0014\, MEOH + \cdots - 0.0230\, ACAL + 3.9272)$$
$$-0.1129 \times (-0.001\, MEOH + \cdots - 0.0432\, ACAL + 2.1923)$$
$$-2.1999$$
$$d(KIRSCH, X) = -0.0018\, MEOH + \cdots - 0.0245\, ACAL + 2.5591$$

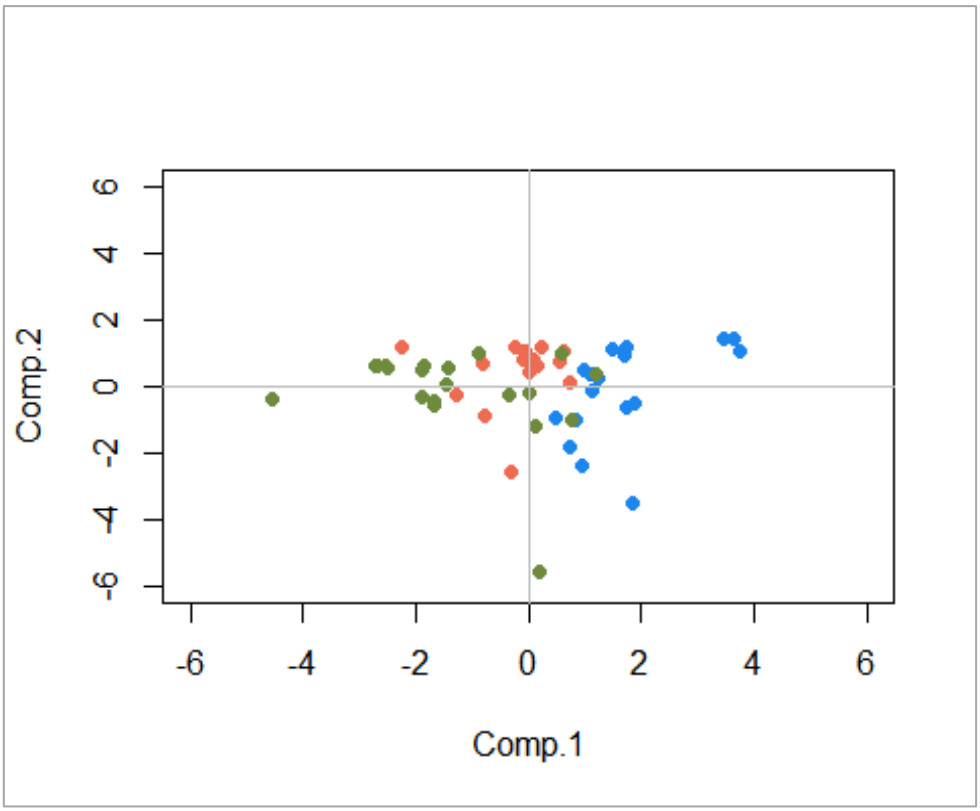
Nous pouvons faire de même pour les classes MIRAB et POIRE.

7.2.2.4 Performances prédictives

Dans cette section, dans un premier temps, nous reproduisons les étapes ci-dessus sous R pour construire le modèle prédictif sur DATA 2 / TRAIN, avec (NB_FACT = 2). Nous utilisons les fonctions basiques princomp() de « stats » et lda() de « MASS » pour éviter d'avoir à installer des packages additionnels. Leurs sorties ne sont pas très folichonnes, nous ne nous y attarderons pas. Dans un deuxième temps, nous appliquons le dispositif sur l'échantillon test DATA 2 / TEST. Deux étapes sont nécessaires : projection des individus supplémentaires dans l'espace factoriel, prédiction à partir de cette nouvelle description.

```
#** importation des données d'apprentissage **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")
```

```r
#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#matrice des descripteurs
XTrain <- DTrain[-1]

#ACP normée
acp <- princomp(XTrain,cor=TRUE,scores=TRUE)
print(acp)

## Call:
## princomp(x = XTrain, cor = TRUE, scores = TRUE)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 1.6729563 1.3110472 1.1846631 1.0219752 0.7326694 0.4498444 0.4302229 0.3320487
##
##  8  variables and  52 observations.

#affichage des valeurs propres - voir (Figure 58)
print(acp$sdev^2)

##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 2.7987827 1.7188448 1.4034266 1.0444334 0.5368045 0.2023600 0.1850918 0.1102563

#affichage des coordonnées factorielles
#des premières observations
print(head(round(acp$scores,3)))

##      Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## [1,] -1.490 -1.115 -0.497 -1.264  0.982 -0.270  0.058 -0.099
## [2,] -0.848  1.014  2.139 -0.945 -1.040 -0.412 -0.649 -0.164
## [3,] -1.726  0.657  0.658 -1.064  0.742 -0.029 -0.146  0.061
## [4,] -1.726 -1.172 -0.635 -0.598 -0.575  0.023  0.069  0.332
## [5,] -3.626 -1.419 -0.141  0.654 -0.379  0.004 -0.178 -0.231
## [6,] -0.947  2.361 -0.100 -1.488  0.309  1.047  0.457  0.216

#position relatives des classes - voir (Figure 61)
plot(-acp$scores[,1],-acp$scores[,2],col=c("dodgerblue2","coral2","darkolivegreen4
")[DTrain$TYPE],pch=19,xlab="Comp.1",ylab="Comp.2",xlim=c(-6,6),ylim=c(-6,6))
abline(h=0,v=0,col="gray")
```




```
#package MASS
library(MASS)

#nombre de composantes à prendre en compte : 2 facteurs
NB_FACT <- 2

#ADL sur les deux premiers facteurs
#les sorties de lda() ne sont pas usuelles en analyse discriminante prédictive
#nous nous contentons des moyennes conditionnelles des classes sur les facteurs
adl <- lda(x=acp$scores[,1:NB_FACT,drop=FALSE],grouping=DTrain$TYPE)
print(adl$means)
##            Comp.1     Comp.2
## KIRSCH -1.6834256  0.1572247
## MIRAB   0.2283473 -0.4077085
## POIRE   1.2596513  0.1721404

#** chargement de l'échantillon test **
DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")

#inspection
str(DTest)

## 'data.frame':    50 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  3 475 186 371 583 0 421 557 167 523 ...
##  $ ACET: num  15 172 101 414 226 25 142 447 86 367 ...
##  $ BU1 : num  0.2 1.9 0 1.2 2.3 0.1 1.6 0 0 2.6 ...
##  $ BU2 : num  30 7 1.6 0 19 8 8 34 0 30 ...
##  $ ISOP: num  9 113 36 97 120 0 75 107 32 116 ...
##  $ MEPR: num  9 33 11 39 46 6 24 39 10 45 ...
##  $ PRO1: num  350 546 128 502 656 253 128 162 114 787 ...
##  $ ACAL: num  9 14 8 9 11 7 31 94 8 25 ...
```

```r
#matrice des descripteurs X
XTest <- DTest[-1]

#** projection des individus supplémentaires avec l'ACP **
#==> nous obtenons leurs coordonnées factorielles
coord_test <- predict(acp,XTest)
print(head(round(coord_test,3)))
##      Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## [1,] -3.502 -1.030 -0.570  0.448 -0.915 -0.454 -0.297 -0.397
## [2,] -0.982 -0.213  0.055 -1.087 -0.568  0.005 -0.153  0.218
## [3,] -2.833 -1.322  0.001  0.014 -0.371 -0.295 -0.203 -0.038
## [4,] -1.131  0.159  0.908 -1.633  1.240 -0.135 -0.143 -0.277
## [5,] -0.485  0.131 -0.191 -1.402  0.016 -0.135  0.188  0.036
## [6,] -3.675 -1.440 -0.379  0.476 -0.708 -0.290 -0.174 -0.448

#prediction de l'ADL à partir des coordonnées factorielles
pred_adl <- predict(adl,coord_test[,1:NB_FACT,drop=FALSE])
print(pred_adl$class)

##  [1] KIRSCH KIRSCH KIRSCH KIRSCH MIRAB  KIRSCH KIRSCH POIRE  KIRSCH MIRAB
## [11] KIRSCH KIRSCH KIRSCH KIRSCH MIRAB  MIRAB  MIRAB  MIRAB  POIRE  POIRE
## [21] MIRAB  MIRAB  MIRAB  POIRE  MIRAB  MIRAB  KIRSCH KIRSCH MIRAB  POIRE
## [31] POIRE  KIRSCH POIRE  POIRE  MIRAB  POIRE  POIRE  POIRE  MIRAB  MIRAB
## [41] POIRE  POIRE  POIRE  POIRE  POIRE  POIRE  POIRE  KIRSCH POIRE  POIRE
## Levels: KIRSCH MIRAB POIRE

#matrice de confusion
mc <- table(DTest$TYPE,pred_adl$class)
print(mc)

##
##          KIRSCH MIRAB POIRE
##   KIRSCH     11     2     1
##   MIRAB       2    10     5
##   POIRE       2     3    14

#taux d'erreur
print(1-sum(diag(mc))/sum(mc))

## [1] 0.3
```

Le taux d'erreur en test est de 30%. Il était de 18% pour l'analyse discriminante sur variables originelles (section 1.4). La déception est grande. Cet exemple illustre bien un des écueils de l'algorithme : le choix judicieux du nombre de facteurs à présenter à l'ADL.

Manifestement, NB_FACT = 2 était insuffisant sur notre base VOTE. L'inefficacité de la composante n°2 dans l'analyse discriminante nous a induit en erreur, nous détournant des suivantes. Or, si l'on relance les calculs sur l'ensemble des facteurs susceptibles d'être générés par l'ACP (8 facteurs), on se rend compte que des axes postérieurs au second sont significatifs (Figure 64). Il fallait le deviner ! Et, à l'instar de DISQUAL, il n'est pas possible d'appliquer une stratégie standard de sélection de variables (chapitre 4) sur les facteurs. N'oublions pas qu'une

composante est calculée sur les résidus des précédentes. Nous devons toujours prendre en bloc les « NB_FACT premières » composantes.

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,42)	p-value
PCA_1_Axis_1	4.8666	-1.4512	-3.0482	0.3677	0.1815	94.7307	0.0000
PCA_1_Axis_2	-0.5342	0.5291	0.0573	0.0728	0.9165	1.9142	0.1601
PCA_1_Axis_3	0.4368	0.3650	-0.6451	0.0761	0.8772	2.9395	0.0639
PCA_1_Axis_4	-5.9392	2.4950	3.1771	0.2315	0.2882	51.8736	0.0000
PCA_1_Axis_5	-2.7479	1.7728	1.0061	0.0909	0.7335	7.6285	0.0015
PCA_1_Axis_6	2.5866	1.2250	-3.1173	0.0929	0.7180	8.2483	0.0010
PCA_1_Axis_7	-1.5060	-0.4556	1.6218	0.0726	0.9194	1.8404	0.1713
PCA_1_Axis_8	-3.2763	2.2919	1.0659	0.0746	0.8943	2.4810	0.0958
constant	-8.1469	-2.8863	-3.7619		-		

Figure 64 – ADL sur l'ensemble des facteurs de l'ACP – Données DATA 2 / TRAIN

7.2.3 Avantages et inconvénients

La combinaison (ACP + ADL) est une stratégie de régularisation destinée à lutter contre les problèmes de dimensionnalité. A force d'expérimentations, j'ai constaté qu'elle était rarement décisive sur les bases usuelles où le ratio nombre d'observations (n) sur nombre de variables est favorable (p). Elle a une meilleure tenue en revanche dès lors que le ratio (n/p) se dégrade. Heureusement, c'est un peu ce qu'on en attend. Elle s'applique même dans un contexte de très forte dimensionnalité en utilisant des algorithmes de type NIPALS pour produire les NB_FACT nécessaires à la partie supervisée (« Construction de variables avec NIPALS », avril 2005).

La détermination du nombre de facteurs reste un problème ouvert.

Mais le principal péril est la nature non-supervisée de l'ACP. Elle ne tient pas compte des classes pour construire les variables synthétiques présentées à l'analyse discriminante. Et ce ne sont pas forcément les composantes à plus forte dispersion qui ont le meilleur pouvoir discriminant. Dans l'illustration ci-dessous (Figure 65), l'ACP ne « voit » que le nuage global (pointillés noir). La dispersion des points font que le premier facteur capte une très grande partie de l'information. Le second serait rejeté à tort parce qu'associé à une très faible valeur propre. Pourtant, alors que les classes (bleu vs. vert) sont parfaitement séparables dans l'espace

originel, elles sont complètement confondues lorsque les observations sont projetées sur uniquement le premier axe factoriel de l'ACP, rendant inopérante l'analyse discriminante.

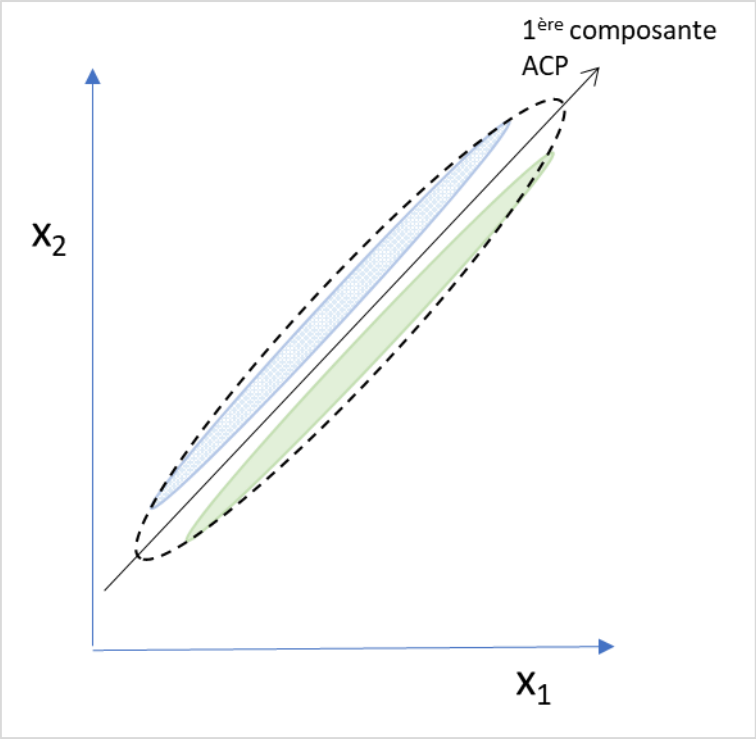


Figure 65 – Nature non-supervisée de l'ACP

7.3 Analyse discriminante PLS

7.3.1 Régression PLS

La régression PLS (partial least squares regression) a pour objectif d'expliquer un ensemble de (K) variables cibles Z (dont les valeurs sont retranscrites dans une matrice dans ce contexte) à partir d'un ensemble de (p) variables explicatives X.

Dans le cas où l'on a qu'une seule variable cible (Z est un vecteur), régression PLS1, elle se positionne comme une alternative à la régression linéaire multiple avec des propriétés de régularisation avérées qui la rende apte à traiter des problèmes à très forte dimensionnalité (Tufféry, 2012, section 13.10).

Lorsque le nombre de variables cibles est supérieur à 1, régression PLS2, les descripteurs sont résumés en une série de facteurs t_h (variables latentes, scores X) deux à deux orthogonaux.

Mais, à la différence de l'analyse en composantes principales, c'est la vertu qui nous intéresse dans cette section, l'approche est supervisée, ces facteurs sont construits de manière à expliquer au mieux Z, laquelle est résumée également en une série de composantes u_h (scores Z), pour lesquelles la contrainte d'orthogonalité ne s'applique pas. Durant le processus de modélisation, la régression PLS va construire une série de composantes (u_h, t_h) de manière à ce que leur covariance soit maximale. Le nombre de facteurs ne peut pas excéder le nombre de variables explicatives (p).

Comme toujours pour les méthodes factorielles, la détermination du nombre adéquat h^* de composantes pour restituer l'information contenue dans les données est un enjeu de la méthode (« Régression PLS – Sélection du nombre d'axes », septembre 2006). D'autant plus que nous sommes dans un schéma supervisé. Il faut voir sous 2 angles la qualité de la représentation pour évaluer les h^* premiers facteurs : la proportion de variance restituée pour les descripteurs X (R_X^2 ; si $h^* = p$, elle est forcément égale à 1) ; la proportion de variance expliquée des Z, qui traduit le pouvoir explicatif du modèle (R_Z^2 ; même si $h^* = p$, elle n'est pas forcément égale à 1). En pratique, si h^* est trop faible, nous sommes en sous-apprentissage, nous n'exploitons pas suffisamment les données ; s'il est trop élevé, nous sommes en situation de sur-apprentissage, le modèle colle trop aux données et ingère des informations qui ne sont pas reproductibles dans la population.

La mise en œuvre de la méthode dans divers logiciels (TANAGRA, SIMCA-P, R, etc.) et la lecture des résultats sont décrites en détail dans un tutoriel (TUTO 11).

7.3.2 Principe de l'analyse discriminante PLS

L'analyse discriminante PLS est une appellation générique pour désigner l'utilisation de la régression PLS pour le classement (TUTO 10). La première étape consiste à coder la variable cible catégorielle Y en une série de K indicatrices 1/0 (ou un autre codage, voir section 5.3), puis de présenter les données, la nouvelle matrice Z des indicatrices et les X, à l'algorithme PLS. Dans la seconde étape, nous exploitons les résultats de la PLS pour effectuer un classement. Dans cette section, nous nous intéressons à une approche qui consiste à lancer une analyse

discriminante linéaire pour prédire Y à partir des composantes (t_h) de la PLS, on parle de « PLS-LDA » (analyse discriminante linéaire sur les composantes de la régression PLS).

Nous sommes bien dans un schéma à deux étages, analogue à l'analyse discriminante sur facteurs principaux (section 7.2). Sauf que les composantes ont été élaborées dans une démarche supervisée. Elles sont censées être plus efficaces pour le classement. Et c'est le cas. Pour l'avoir énormément pratiqué, j'ai constaté que PLS-LDA avait une excellente tenue dans les espaces de représentation composés d'un grand nombre de descripteurs bruités et/ou redondants. Elle tient la dragée haute à des techniques dont les propriétés de régularisation sont connues et reconnues (voir par exemple le comparatif, « Analyse Discriminante PLS – Etude comparative », mai 2008).

7.3.3 Un exemple de traitements – DATA 2

Nous travaillons de nouveau sur DATA 2 / TRAIN en suivant la même trame que pour la combinaison ACP + ADL (section 7.2.2). Pour réaliser la régression PLS dans TANAGRA, nous combinons deux composants PLSR et PLS FACTORIAL. Le processus complet est lui-même encapsulé dans PLS-LDA, mais les résultats intermédiaires ne sont pas accessibles dans ce cas.

7.3.3.1 Régression PLS

La variable TYPE a été transformée en une matrice Z composée de 3 indicatrices {TYPE.MEOH, TYPE.MIRAB, TYPE.POIRE}. Le premier résultat important de la régression PLS est la qualité de la restitution des facteurs (TUTO 11, section 4.4.1).

-	Input variables (X)		Target Variables (Z)	
Latent Factor	Current X (%)	Cumulative X (%)	Current Z (%)	Cumulative Z (%)
1	33.596	33.60	37.300	37.30
2	16.202	49.80	14.733	52.03
3	16.043	65.84	4.330	56.36
4	9.687	75.53	6.142	62.51
5	13.969	89.50	2.127	64.63
6	6.714	96.21	0.972	65.61
7	1.436	97.65	0.724	66.33
8	2.353	100.00	0.004	66.33

Figure 66 – Qualité de restitution – Régression PLS – DATA 2 / TRAIN

Lorsque l'on sélection tous les (p = 8) facteurs, nous épuisons la totalité de l'information portée par les descripteurs X. En revanche, la qualité globale (R_Z^2) de restitution de l'ensemble des variables cibles est de 66.33% c.-à-d. nous expliquons près de 2/3 de leur variabilité. Nous ne le montrons pas ici, mais il est possible d'obtenir la qualité d'explication pour chaque variable composant Z. Nous verrons cela lors du traitement sous R (section 7.2.2.4).

Un graphique montrant la progression du pouvoir explicatif de la régression suggère qu'après la 4ème composante, le gain en variance expliquée est faible (Figure 67). Cela nous donne une première idée sur le nombre de facteurs à retenir pour la suite.

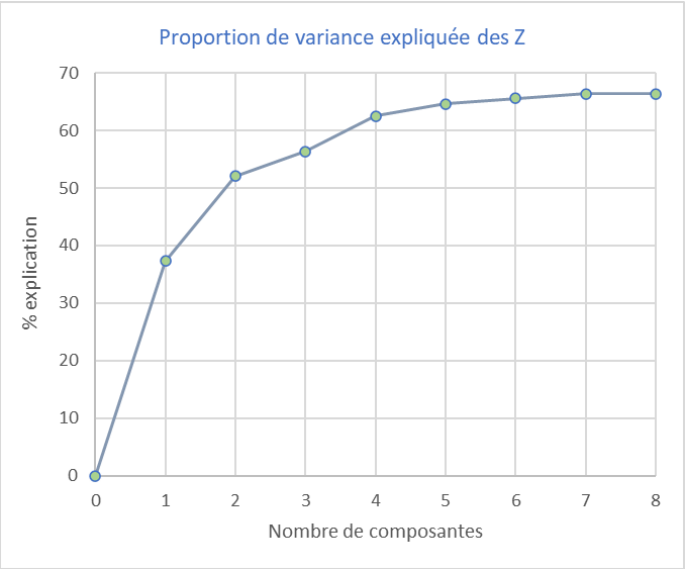


Figure 67 – Proportion de variance expliquée sur les cibles en fonction du nombre de composantes

Ce sont les scores (t_h) qui seront présentés à l'analyse discriminante par la suite. Les coefficients de projection permettent de les obtenir à partir des variables originelles. Elles sont définies sur les variables centrées et réduites (Figure 68).

Attribute	Avg	Std-dev	Axis_1	Axis_2	Axis_3	Axis_4	Axis_5	Axis_6	Axis_7	Axis_8
MEOH	808.0481	369.7820	-0.6214	-0.2253	-0.2342	0.0560	-0.2519	-0.1340	-0.4796	-0.5797
ACET	205.4288	122.1538	0.0268	-0.2260	0.3485	-0.5259	-0.3289	-0.4526	0.0427	0.1135
BU1	14.4231	11.0276	-0.6035	-0.4251	-0.1179	-0.4537	0.5018	0.3639	0.4610	0.2139
BU2	29.7769	54.4855	-0.1387	0.2477	-0.7671	0.1225	-0.4249	-0.3210	-0.2482	0.4288
ISOP	98.3077	48.3101	-0.2219	0.3662	0.4867	-0.4144	0.7314	0.3318	-0.4584	0.3099
MEPR	37.1577	18.4716	-0.3219	0.6201	0.0023	0.5127	-0.5445	-0.5264	0.5331	-0.1504
PRO1	436.9250	626.1791	0.2626	0.3634	-0.0050	-0.5111	0.3494	0.4569	0.1510	-0.5021
ACAL	13.0654	8.0543	-0.0887	0.1721	0.1661	0.2904	-0.5717	0.5927	0.0638	0.2300

Figure 68 – Coefficients de projection standardisées (Coordonnées t_h) – DATA 2 / TRAIN

Pour le premier facteur, nous avons :

$$t_1 = -0.6214 \times \left(\frac{MEOH - 808.0481}{369.7820}\right) + \cdots - 0.0887 \times \left(\frac{ACAL - 13.0654}{8.0543}\right)$$

Avec les scores (t_h), nous pouvons représenter les observations dans les différents plans factoriels en les étiquetant selon leur classe d'appartenance pour nous faire une idée de leur pouvoir discriminant. Nous choisissons les deux premiers (Figure 69).

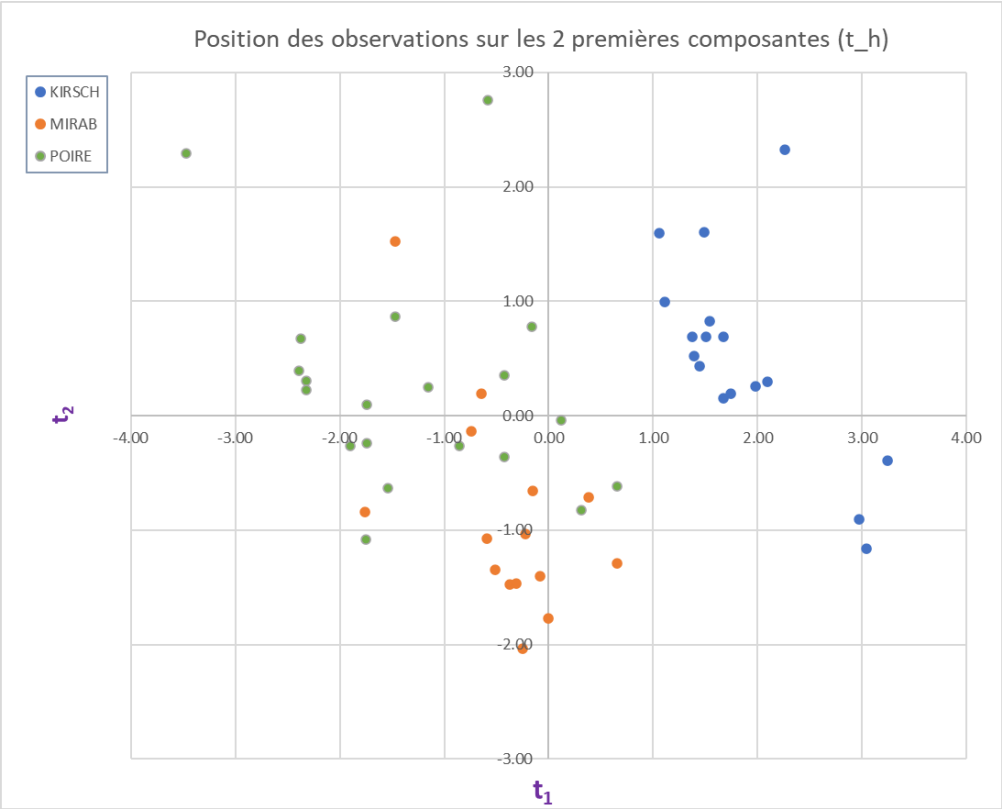


Figure 69 – Positions des observations dans le 1er plan factoriel de la régression PLS – DATA 2 / TRAIN

Le résultat est sans surprises : KIRSCH est bien discriminé, il y a un recouvrement entre MIRAB et POIRE. Le 1er facteur semble déterminant. Pour le 2nd, difficile de trancher visuellement, l'analyse discriminante qui va suivre nous donnera les bonnes indications.

7.3.3.2 Analyse discriminante sur facteurs

Echaudé par l'expérience de l'ACP où des composantes postérieures à celles qui n'étaient pas pertinentes se sont révélés finalement importants (Figure 64), nous décidons de présenter à l'analyse discriminante linéaire tous les facteurs $(t_h\,;\, h = 1, \ldots, 8)$ de la régression PLS. Nous comptons sur les tests de pertinence pour obtenir une indication sur le nombre (h^*) à retenir.

Plusieurs commentaires importants viennent à la lecture des sorties de l'ADL (Figure 70) :

- Puisque nous utilisons tous les facteurs, le Λ de Wilks ($\Lambda = 0.0667$) est identique à celui de l'analyse initiée sur les variables originelles.
- Les facteurs sont d'importance décroissante dans l'analyse discriminante. Les résultats ne sont pas toujours aussi ostensibles. Les données DATA 2 sont faciles à traiter. Mais la pratique montre que les fortes incohérences dans l'ordonnancement des facteurs surviennent rarement. N'oublions pas que, par construction, ils ont pouvoir explicatif décroissant sur la matrice des indicatrices de la variable cible Y (Figure 66).
- Seuls les 4 premiers sont pertinents à 5% pour DATA 2, rejoignant en cela la suggestion du graphique montrant l'évolution de la proportion de variance expliquée (Figure 67).

MANOVA

Stat	Value	p-value
Wilks' Lambda	0.0667	-
Bartlett -- C(16)	123.1845	0.0000
Rao -- F(16, 84)	15.0761	0.0000

LDA Summary

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,42)	p-value
plsr_t_1_1	6.032	-1.900	-3.702	0.471	0.142	127.189	0.0000
plsr_t_2_1	3.082	-2.142	-1.014	0.137	0.488	22.015	0.0000
plsr_t_3_1	2.738	-0.614	-1.867	0.105	0.635	12.081	0.0001
plsr_t_4_1	0.270	-1.326	0.765	0.082	0.818	4.678	0.0147
plsr_t_5_1	0.562	0.308	-0.709	0.073	0.914	1.982	0.1504
plsr_t_6_1	1.433	-0.166	-1.094	0.073	0.914	1.972	0.1519
plsr_t_7_1	3.152	-1.656	-1.438	0.072	0.924	1.732	0.1894
plsr_t_8_1	0.135	-0.005	-0.110	0.067	1.000	0.007	0.9929
constant	-8.147	-2.886	-3.762	-			

Figure 70 – Analyse discriminante sur la totalité des composantes de la régression PLS – DATA 2 / TRAIN

Forts de ces constations, nous relançons l'analyse discriminante en choisissant les ($h^* = 4$) premières composantes (t_h). Voici les fonctions de classement (Figure 71).

Classification functions				Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,46)	p-value
plsr_t_1_1	5.158	-1.722	-3.093	0.53	0.16	118.53	0.0000
plsr_t_2_1	2.741	-1.980	-0.845	0.17	0.51	22.04	0.0000
plsr_t_3_1	2.319	-0.548	-1.560	0.13	0.67	11.12	0.0001
plsr_t_4_1	0.343	-1.243	0.641	0.10	0.83	4.68	0.0141
constant	-7.029	-2.686	-3.178		-		

Figure 71 – Analyse discriminante sur les 4 premières composantes de la régression PLS – DATA 2 / TRAIN

Pour KIRSCH par exemple :

$$d(KIRSCH, t) = 5.158\, t_1 + 2.741\, t_2 + 2.319\, t_3 + 0.343\, t_4 - 7.029$$

7.3.3.3 Retour sur les variables originelles

Pour exprimer les fonctions de classement à partir des variables originelles, il suffit de substituer aux (t_h) leur expression sous forme de fonction de projection (Figure 68). Voici le modèle prédictif définitif (Figure 72).

Classification functions

Number of PLS axis used = 4

Attribute	KIRSCH	MIRAB	POIRE
MEOH	-0.012	0.004	0.007
ACET	0.001	0.007	-0.006
BU1	-0.427	0.228	0.192
BU2	-0.033	0.000	0.027
ISOP	0.018	-0.002	-0.013
MEPR	0.012	-0.071	0.043
PRO1	0.003	-0.001	-0.002
ACAL	0.062	-0.079	0.007
constant	4.865	-6.630	-10.329

Figure 72 – Fonctions de classement – PLS-LDA – DATA 2 / TRAIN

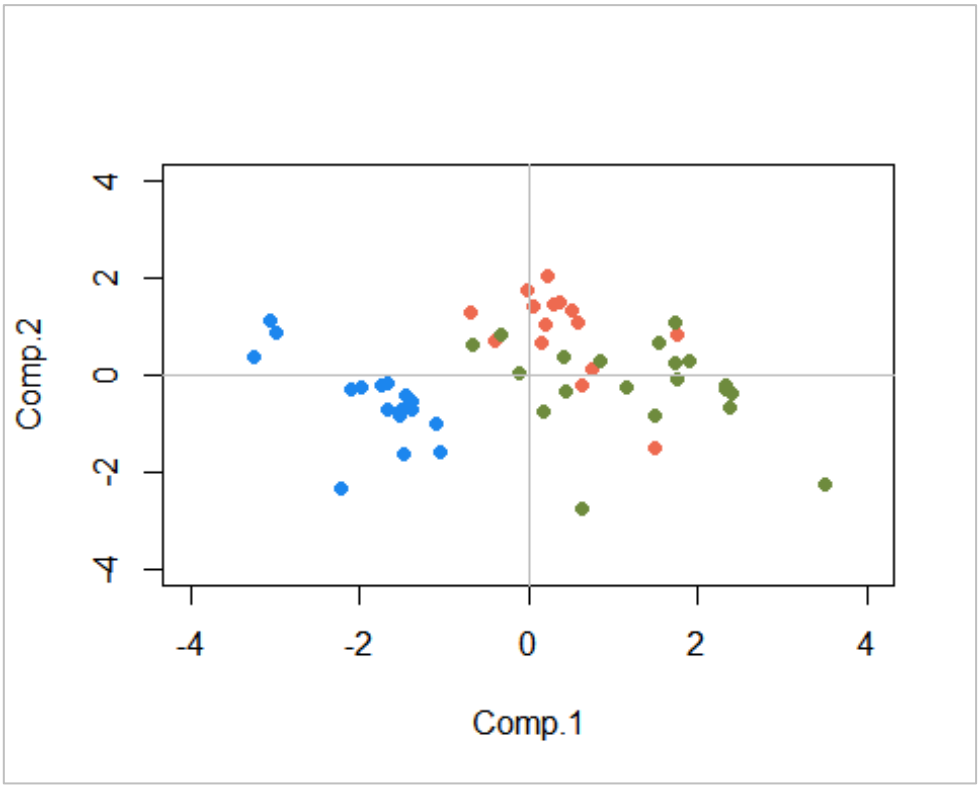
Malheureusement, mis à part l'inspection empirique des disparités entre les coefficients, nous ne disposons pas d'outils pour mesurer l'impact des descripteurs dans la prédiction.

7.3.3.4 Performances prédictives

Nous passons sous R pour étudier les performances en classement de la combinaison PLS-LDA. Nous utilisons le package « pls » dont nous avions étudié le mode de fonctionnement dans un tutoriel (TUTO 11, sections 5.3 et 7). La séquence des opérations est exactement la même que pour l'analyse discriminante sur facteurs principaux de l'ACP (section 7.2.2.4).

```r
#** importation des données d'apprentissage **

#lecture
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header=TRUE, sheetName = "DATA_2_TRAIN")

#inspection
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...

#matrice des descripteurs X
XTrain <- as.matrix(DTrain[-1])

#transformation de TYPE en indicatrices

#fonction pour créer des dummies
#à partir d'une variable de type factor
#codage disjonctif complet - sans modalité de référence
set_dummies_plus <- function(x){
  #nombre de modalités
  L <- nlevels(x)
  #liste des modalités
  modalites <- levels(x)
  #création des variables 0/1
  M <- sapply(1:L,function(j){return(as.double(x==modalites[j]))})
  #nommer les variables avec les modalités
  colnames(M) <- modalites[1:L]
  #renvoyer sous la forme de data frame
  return(as.data.frame(M))
}

#recodage de Y en Z – 3 indicatrices pour les modalités de TYPE
Z <- as.matrix(set_dummies_plus(DTrain$TYPE))
print(head(Z))

##      KIRSCH MIRAB POIRE
## [1,]      1     0     0
## [2,]      1     0     0
## [3,]      1     0     0
## [4,]      1     0     0
## [5,]      1     0     0
## [6,]      1     0     0

#package 'pls'
library(pls)

##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings

#lancer la régression PLS
#4 facteurs demandés, variables X centrées et réduites, méthode NIPALS
#/!\ la matrice Z des indicatrices est laissée telle quelle
reg <- mvr(Z ~ XTrain, ncomp = 4, center = TRUE, scale = TRUE, method="oscorespls")
summary(reg)

#nous disposons des R2expliqués par variable cible
#l'indicatrice de KIRSCH est la mieux restituée (85.36%)

## Data:    X dimension: 52 8
##  Y dimension: 52 3
## Fit method: oscorespls
## Number of components considered: 4
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps
## X        33.642    49.79    65.80    75.37
## KIRSCH   67.480    79.77    85.01    85.36
## MIRAB     2.512    32.04    32.35    43.91
## POIRE    41.532    44.36    51.80    58.56

#premières lignes de la matrice t_h
print(head(reg$scores[,]))

##      Comp 1     Comp 2     Comp 3     Comp 4
## 1 -1.674095 -0.1659095  0.5658803 -0.1322879
## 2 -1.387389 -0.5333241  1.3036679 -0.2774277
## 3 -1.985014 -0.2716605  0.6744243  0.6393304
## 4 -1.744692 -0.2056038  0.1688830 -0.6169540
## 5 -3.061850  1.1413034 -0.5489803 -0.5883183
## 6 -1.468358 -1.6135694  0.5789782  1.5628396

#position relatives des classes dans le plan (t1, t2)
plot(reg$scores[,1],reg$scores[,2],col=c("dodgerblue2","coral2","darkolivegreen4")
[DTrain$TYPE],pch=19,xlab="Comp.1",ylab="Comp.2",xlim=c(-4,4),ylim=c(-4,4))
abline(h=0,v=0,col="gray")
```




```
#package MASS
library(MASS)

#nombre de composantes à prendre en compte pour l'ADL
NB_FACT <- 4

#ADL sur les NB_FACT premiers facteurs
adl <- lda(x=reg$scores[,1:NB_FACT,drop=FALSE],grouping=DTrain$TYPE)
print(adl$means)

#moyenne des composantes conditionnellement aux classes

##             Comp 1      Comp 2       Comp 3       Comp 4
## KIRSCH -1.8557932 -0.5306500  0.33473616 -0.05750409
## MIRAB   0.3919383  0.9003765  0.08933446  0.36466294
## POIRE   1.2834705 -0.2242299 -0.35152658 -0.22461873

DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")

#inspection
str(DTest)

## 'data.frame':    50 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  3 475 186 371 583 0 421 557 167 523 ...
##  $ ACET: num  15 172 101 414 226 25 142 447 86 367 ...
##  $ BU1 : num  0.2 1.9 0 1.2 2.3 0.1 1.6 0 0 2.6 ...
##  $ BU2 : num  30 7 1.6 0 19 8 8 34 0 30 ...
##  $ ISOP: num  9 113 36 97 120 0 75 107 32 116 ...
##  $ MEPR: num  9 33 11 39 46 6 24 39 10 45 ...
##  $ PRO1: num  350 546 128 502 656 253 128 162 114 787 ...
##  $ ACAL: num  9 14 8 9 11 7 31 94 8 25 ...
```

```r
#matrice des descripteurs X (échantillon test)
XTest <- as.matrix(DTest[-1])

#projection des individus supplémentaires (échantillon test)
#c.-à-d. calcul de leurs coordonnées factorielles (t1,t2,t3,t4)
coord_test <- predict(reg,newdata=XTest,type="scores")
print(head(round(coord_test,3)))

##      Comp 1 Comp 2 Comp 3 Comp 4
## [1,] -3.005  0.351 -0.890 -1.175
## [2,] -1.330 -0.707  0.726 -0.262
## [3,] -2.565  0.622 -0.101 -0.756
## [4,] -1.629 -0.271  1.337  0.600
## [5,] -0.922 -1.066  0.651 -0.168
## [6,] -3.156  0.730 -0.680 -1.087

#prediction à partir des coordonnées factorielles
pred_adl <- predict(adl,coord_test[,1:NB_FACT,drop=FALSE])
print(pred_adl$class)

##  [1] KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH KIRSCH
## [11] KIRSCH KIRSCH KIRSCH KIRSCH POIRE  MIRAB  MIRAB  MIRAB  MIRAB  MIRAB
## [21] MIRAB  MIRAB  MIRAB  MIRAB  POIRE  MIRAB  MIRAB  POIRE  MIRAB  MIRAB
## [31] POIRE  MIRAB  MIRAB  MIRAB  POIRE  KIRSCH POIRE  POIRE  MIRAB  POIRE
## [41] POIRE  POIRE  POIRE  POIRE  POIRE  POIRE  POIRE  MIRAB  POIRE  POIRE
## Levels: KIRSCH MIRAB POIRE

#matrice de confusion
mc <- table(DTest$TYPE,pred_adl$class)
print(mc)

##
##          KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       0    13     4
##   POIRE       1     5    13

#taux d'erreur
print(1-sum(diag(mc))/sum(mc))

## [1] 0.2
```

Le taux d'erreur est du même ordre que celui obtenu pour l'analyse discriminante sur les variables initiales (1 individu supplémentaire mal classé sur 50). C'est un exemple didactique. Les caractéristiques de notre fichier ne permet pas de mettre en exergue les qualités de régularisation de PLS-LDA.

7.3.4 Avantages et inconvénients

En termes d'efficacité, PLS-LDA se compare très bien aux autres méthodes linéaires capables de produire un modèle explicite sous forme de fonctions de classement, y compris face aux techniques réputées très performantes dans un contexte de sur-dimensionnalité comme les

SVM linéaires (support vector machine) (« Analyse Discriminante PLS – Etude comparative », mai 2008). Le nombre (h^*) de facteurs à utiliser permet de moduler la sensibilité aux données d'apprentissage et combattre les problèmes de surapprentissage. En pratique, comme nous avons pu le constater dans l'expérimentation ci-dessus, il est relativement facile d'en déterminer la bonne valeur.

Le principal bémol vient de son incapacité à expertiser le rôle des variables. La combinaison empêche de situer clairement l'influence des prédicteurs dans la discrimination. Et il n'est pas possible de bénéficier d'un mécanisme intégré de sélection de variables.

8 Analyse factorielle discriminante

Initialement, j'avais pour idée de ne pas parler du tout de l'analyse factorielle discriminante (AFD) pour ne pas polluer le discours de la prédiction sous la forme de fonctions de classement explicites, évaluable globalement, et pour lesquelles nous pouvons mesurer les contributions des variables, à l'instar de ce que nous avons en régression logistique. Au fil de la rédaction de cet ouvrage, je me suis rendu que les aspects descriptifs et prédictifs étaient intimement liés dans l'analyse discriminante, et qu'il était difficile de les dissocier. Rarement j'ai aligné autant de représentations graphiques dans la description d'une méthode d'apprentissage automatique, pour situer la configuration des points dans un espace de description, leurs barycentres conditionnels, les distances, les dispersions, la séparabilité des classes, etc. D'ailleurs, dans certaines références, on conseille de passer par une représentation factorielle pour réaliser l'affectation aux groupes des individus supplémentaires (Hastie et al., 2017, section 4.3.2 « Computations for LDA »). L'étude du code source des fonctions liées à lda() du package « MASS » de R est édifiante. Elles passent par une décomposition en valeurs singulières et s'appuient fondamentalement sur les résultats de l'analyse discriminante descriptive pour effectuer les prédictions (sections 8.2.1 et 8.2.2).

Dans ce qui suit sont présentés les principes, la pratique et l'interprétation de l'analyse discriminante descriptive. Nous reprenons la trame du support de cours accessible en ligne (Rakotomalala R., « Analyse factorielle discriminante – Diaporama », janvier 2011), en retraçant la démarche sur le même exemple des « Vins de Bordeaux » (Tenenhaus, 2007, page 353). La particularité de ce chapitre sera la mise en évidence des liens qu'elle peut avoir avec l'analyse

prédictive, et la possibilité, rarement mise en avant, de déduire des fonctions de classement explicites à partir des résultats de l'AFD (TUTO 1, section 2.6, TUTO 3, section 3.4.7), équivalentes – c.-à-d. prédisant de manière identique – à celles de l'analyse discriminante linéaire. La dernière section sera consacrée par l'analyse des correspondances discriminante qui est une extension de la méthode aux descripteurs qualitatifs.

8.1 Principe de l'analyse factorielle discriminante

8.1.1 Données « Vins de Bordeaux »

Nous utilisons un nouveau jeu de données pour accompagner la présentation. La « Qualité des vins de Bordeaux » (WINE) nous vient de l'ouvrage de Tenenhaus (2007, Tableau 10.1, page 353). Elle décrit les caractéristiques des vins sur les millésimes allant de 1924 à 1957 ($n = 34$). Chaque observation correspond à une année, les ($p = 4$) descripteurs sont des variables météorologiques : la température (somme des températures moyennes journalières en °C), le soleil (durée d'insolation en heures), la chaleur (le nombre de jours de grande chaleur), la pluie (hauteur des pluies en mm). Les classes sont définies par la qualité des vins {Bon, Moyen, Médiocre}, soit ($K = 3$) modalités. Il n'est pas tenu compte du caractère ordinal de la variable cible dans notre analyse.

Nous chargeons les données et nous préparons les structures sous R.

```
#** importation des données **

#lecture
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "WINE")

#inspection
str(D)

## 'data.frame':    34 obs. of  5 variables:
##  $ Temperature: num  3064 3000 3155 3085 3245 ...
##  $ Soleil     : num  1201 1053 1133 970 1258 ...
##  $ Chaleur    : num  10 11 19 4 36 35 13 12 14 29 ...
##  $ Pluie      : num  361 338 393 467 294 225 417 488 677 427 ...
##  $ Qualite    : Factor w/ 3 levels "bad","good","medium": 3 1 3 1 2 2 1 1 1 3 .
..

#affichage des premières lignes
print(head(D))
```

```
##   Temperature Soleil Chaleur Pluie Qualite
## 1        3064   1201      10   361  medium
## 2        3000   1053      11   338     bad
## 3        3155   1133      19   393  medium
## 4        3085    970       4   467     bad
## 5        3245   1258      36   294    good
## 6        3267   1386      35   225    good
```

```
#** préparation des structures **

#matrice des descripteurs X
X <- D[1:4]

#vecteur cible y
y <- D$Qualite

#nombre d'observations
n <- nrow(D)
print(n)

## [1] 34

#nombre de classes
K <- nlevels(y)
print(K)

## [1] 3

#nombre d'explicatives
p <- ncol(X)
print(p)

## [1] 4

#effectifs par classe
n_k <- table(y)
print(n_k)

## y
##    bad   good medium
##     12     11     11

#calcul des fréquences relatives des classes
pi_k <- table(y)/n
print(pi_k)

## y
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294

#moyennes conditionnelles selon le groupe d'appartenance
xb_k <- aggregate(X,list(y),FUN=mean)
print(xb_k)

##   Group.1 Temperature   Soleil  Chaleur    Pluie
## 1     bad    3037.333 1126.417 12.08333 430.3333
## 2    good    3306.364 1363.636 28.54545 305.0000
## 3  medium    3140.909 1262.909 16.45455 339.6364
```

8.1.2 Objectif de l'analyse factorielle discriminante

L'AFD est une méthode descriptive. Elle vise à produire un système de représentation de dimension réduite qui permet de discerner les classes lorsqu'on y projette les individus. Il s'agit d'une analyse factorielle. On peut la voir comme une variante de l'analyse en composantes principales où les centres de classes sont les individus, pondérés par leurs effectifs, et avec une métrique particulière (Saporta, 2006, section 18.1.2.2 ; Lebart et al., 2000, section 3.3.4.c). Les variables latentes (ou variables discriminantes) sont exprimées par des combinaisons linéaires des variables originelles. Elles sont deux à deux orthogonales. Elles cherchent à assurer un écartement maximal entre les centres de classes. In fine, l'objectif est de mettre en évidence les caractéristiques qui permettent de distinguer au mieux les groupes.

8.1.2.1 Rapport de corrélation

Pour matérialiser le degré de discrimination des classes sur un facteur, nous utilisons le rapport de corrélation. Il est basé sur la dispersion des moyennes conditionnelles.

Le point de départ est la décomposition de la variance bien connue en ANOVA (analyse de variance à un facteur, Rakotomalala, 2010, section 1.3) :

$$SCT = SCE + SCR$$

$$\sum_{\omega}(z(\omega) - \bar{z})^2 = \sum_{k=1}^{K} n_k(\bar{z}_k - \bar{z})^2 + \sum_{k=1}^{K} \sum_{\omega:Y(\omega)=y_k} (z(\omega) - \bar{z}_k)^2$$

Où $z(\omega)$ est la coordonnée de l'individu ω sur le facteur ; $\bar{z}$ est la moyenne du facteur ; $\bar{z}_k$ est la moyenne de z conditionnellement à la classe (y_k). Attention, cette formule est générique, $\bar{z}$ sera nulle par construction dans l'AFD, quel que soit le facteur étudié.

SCT correspond à la variabilité totale (somme des carrés totaux), SCE est la variabilité expliquée par l'appartenance aux classes, SCR est la variabilité résiduelle, non-expliquée.

Le carré du rapport de corrélation est défini par le ratio entre SCE et SCT :

$$\eta^2_{z,y} = \frac{SCE}{SCT}$$

$0 \leq \eta^2 \leq 1$, le rapport de corrélation est un indicateur de séparabilité des groupes :

- η^2 = 0, la discrimination est impossible, les moyennes conditionnelles sont confondues, SCE = 0.
- η^2 = 1, la discrimination est parfaite, les points associés aux groupes sont agglutinés autour de leur moyenne respectives, SCR = 0 ou, c'est équivalent, SCE = SCT.

Prenons un exemple simple dans le plan. Nous réalisons une AFD sur les données WINE avec seulement les variables (Qualité ~ Température + Soleil). Dans le plan, le premier facteur (z_1) sera une droite où les centres de classes projetés seront le plus écartés possible (Figure 73).

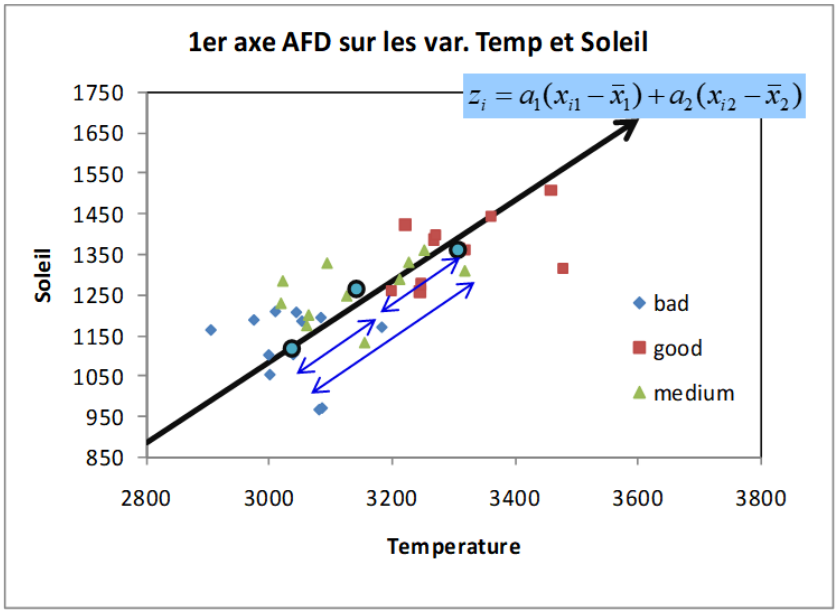


Figure 73 – 1er facteur de l'AFD sur (Qualité ~ Température + Soleil)

8.1.2.2 Processus AFD

L'objectif de l'analyse factorielle discriminante est de produire un nouvel espace de représentation. Il est formé par une succession de variables latentes (z_h) – où $z_h = z_h(x) = a_{h0} + a_{h1}x_1 + \cdots + a_{hp}x_p$ sont exprimés en fonction des (X_1, X_2, ..., X_p) – qui visent à maximiser tour à tour le rapport de corrélation c.-à-d. qui assurent l'écartement maximal entre les centres de classes. Les fonctions $z_h(x)$ sont appelées « fonctions discriminantes canoniques » :

- Le nombre maximal d'axes factoriels est égal à $H = \min(K-1, p)$. Si (p = 1), le facteur est la variable elle-même.
- Les facteurs z_h sont deux à deux orthogonaux. En effet, un axe factoriel maximise l'écart entre les barycentres conditionnels en contrôlant l'effet des facteurs précédents c.-à-d. il essaie d'expliquer les écarts résiduels qui n'ont pas été pris en comptes par les axes précédents.
- Le pouvoir discriminatoire de chaque facteur est exprimé par le (carré du) rapport de corrélation.

Reprenons l'exemple (Qualité ~ Température + Soleil), il est possible de produire $H = \min(3\text{-}1,2) = 2$ variables latentes sous forme de droites, les voici représentées dans le plan (Figure 74).

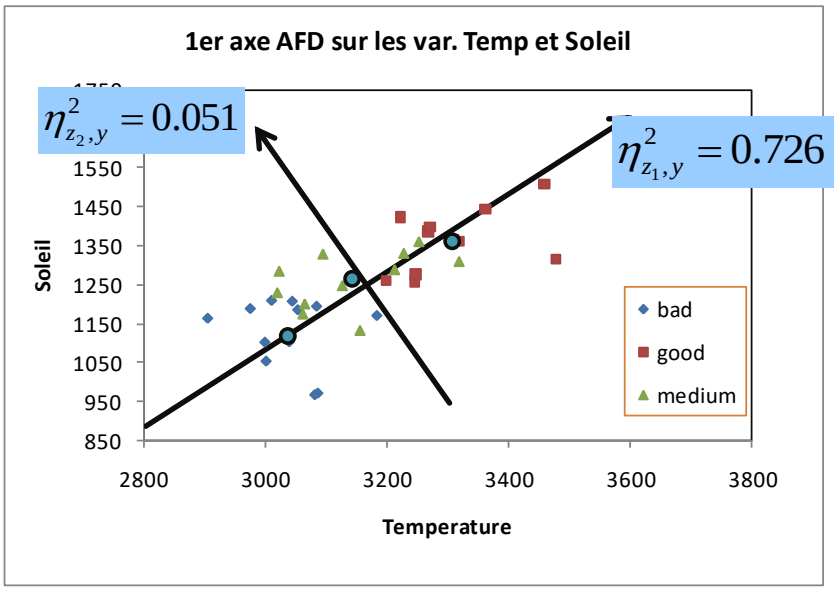


Figure 74 – Les 2 facteurs (z_1, z_2) de l'AFD sur (Qualité ~ Température + Soleil)

8.1.2.3 Recherche de la représentation optimale – Solution (1)

Reprenons quelques définitions importantes :

- V_b est la matrice de variance covariance globale. L'indice « b » signifie qu'elle est pondérée par (1/n) et non pas par les degrés de liberté [1/(n-1)].
- W_b est la matrice de variance de variance covariance intra-classes.
- B_b est la matrice de variance covariance inter-classes et, en vertu du théorème d'Huyghens, $(B_b = V_b - W_b)$.

Soit $a = (a_1, a_2, \ldots, a_p)$ le vecteur (colonne) de coefficients permettant de définir le premier facteur c.-à-d. $z = a_1\,(x_1 - \bar{x}_1) + \cdots + a_p\,(x_p - \bar{x}_p)$ (Remarque : la constante n'est pas nécessaire puisque les variables sont centrées).

Alors les différentes sommes des carrés peuvent s'écrire comme suit :

- $SCT = n \times (a^T\, V_b\, a)$
- $SCR = n \times\ (a^T\, W_b\, a)$
- $SCE = n \times\ (a^T\, B_b\, a)$

L'analyse factorielle discriminante cherche à estimer le vecteur de coefficients « a » qui définit une variable latente permettant de maximiser le rapport de corrélation. En d'autres termes,

$$\max_a \eta^2_{z,y} \Leftrightarrow\ \max_a \frac{a^T B_b\, a}{a^T\, V_b\, a}$$

Cette optimisation est équivalente à

$$\begin{cases} \max\limits_a a^T\, B_b\, a \\ sc.\ a^T\, V_b\, a = 1 \end{cases}$$

On souhaite que le vecteur « a » soit normé.

La solution passe par la formation du lagrangien...

$$L(a) = a^T\, B_b\, a - \lambda\,(a^T\, V_b\, a - 1)$$

... dont la dérivée par rapport à « a » doit être nulle :

$$\frac{\partial L(a)}{\partial a} = 0\ \Rightarrow B_b a = \lambda\, V_b a$$

$$\Rightarrow V_b^{-1} B_b\, a = \lambda\, a$$

« λ » est la première valeur propre de la matrice $(V_b^{-1} B_b)$, « a » est la valeur propre associée.

Par extension, les coefficients des fonctions discriminantes canoniques de l'analyse factorielle discriminante sont définis par les valeurs et vecteurs propres de $(V_b^{-1}B_b)$. Au maximum, nous avons H = min(K-1, p) valeurs propres non-nulles, et autant d'axes factoriels.

- $(\lambda_h = \eta_h^2)$: la valeur propre correspond au carré du rapport de corrélation associé à l'axe, avec toujours $(0 \leq \lambda \leq 1)$.
- $(\sqrt{\lambda_h} = \eta_h)$ est appelé « corrélation canonique ».

Les (λ_h) permettent de caractériser les facteurs, mais ils ne s'additionnent pas. Elle ne permet pas de décomposer la variance expliquée en contributions des facteurs.

Exemple « WINE »

Réalisons les calculs sous R, nous reprenons la suite de la section 8.1.1. Nous calculons tour à tour les matrices W_b, V_b, et B_b (par différence). Nous appliquons ensuite la diagonalisation.

```
#matrice de variance covariance par groupe
V_k <- by(as.matrix(X),list(y),cov)
print(V_k)

## : bad
##             Temperature      Soleil   Chaleur       Pluie
## Temperature   4807.8788 -1003.60606 233.51515   227.06061
## Soleil       -1003.6061  7813.35606 157.41667   -59.78788
## Chaleur        233.5152   157.41667  39.71970   -25.57576
## Pluie          227.0606   -59.78788 -25.57576 10992.60606
## ------------------------------------------------------------
## : good
##             Temperature     Soleil    Chaleur   Pluie
## Temperature   8474.4545  3390.8455  585.28182 -2709.1
## Soleil        3390.8455  6449.0545  163.11818 -1336.1
## Chaleur        585.2818   163.1182   77.47273  -306.1
## Pluie        -2709.1000 -1336.1000 -306.10000  2734.6
## ------------------------------------------------------------
## : medium
##             Temperature    Soleil   Chaleur     Pluie
## Temperature  10009.0909 3541.5909 587.94545 3793.0636
## Soleil        3541.5909 5175.4909 188.24545  912.0636
## Chaleur        587.9455  188.2455  45.27273  226.2818
## Pluie         3793.0636  912.0636 226.28182 3023.4545

#matrice de covariance intra-classe
#utilise les matrices conditionnelles V_k
Wb <- 1/(n) * Reduce("+",lapply(levels(y),function(k){(n_k[k]-1)*V_k[[k]]}))
print(Wb)

##             Temperature    Soleil   Chaleur      Pluie
## Temperature   6991.8271 1714.2558 420.61586  392.27362
```

```
## Soleil        1714.2558 5946.8344 154.27117 -144.05971
## Chaleur        420.6159  154.2712  48.95209  -31.75045
## Pluie          392.2736 -144.0597 -31.75045 5249.97683

#covariance totale
Vb <- (n-1)/n * cov(X)
print(Vb)

##             Temperature     Soleil    Chaleur      Pluie
## Temperature   19346.751 12360.3028 1187.42042 -5130.4481
## Soleil        12360.303 15561.8071  795.79239 -5317.7604
## Chaleur        1187.420   795.7924   97.38062  -356.4516
## Pluie         -5130.448 -5317.7604 -356.45156  8108.5407

#covariance inter-classe, obtenue par défférence
Bb <- Vb - Wb
print(Bb)

##             Temperature     Soleil    Chaleur      Pluie
## Temperature  12354.9238 10646.0470  766.80455 -5522.7217
## Soleil       10646.0470  9614.9726  641.52122 -5173.7007
## Chaleur        766.8046   641.5212   48.42853  -324.7011
## Pluie        -5522.7217 -5173.7007 -324.70111  2858.5638

#formation de la matrice à diagonaliser (solution 1)
M1 <- solve(Vb) %*% (Bb)
print(M1)

##             Temperature     Soleil      Chaleur      Pluie
## Temperature   0.4160061  0.3718448  0.025236713 -0.1985302
## Soleil        0.2960122  0.3032117  0.016275535 -0.1775628
## Chaleur      -0.5200100 -1.3951496  0.008965648  1.1224839
## Pluie        -0.2466117 -0.2652594 -0.013008537  0.1598178

#diagonalisation
sol1 <- eigen(M1,symmetric=FALSE)
print(sol1)

## eigen() decomposition
## $values
## [1] 7.662929e-01 1.217084e-01 3.440930e-16 1.761447e-16
##
## $vectors
##            [,1]          [,2]        [,3]       [,4]
## [1,] -0.2878446 -0.0003616251  0.09215347  0.0306755
## [2,] -0.2276222 -0.0416687914 -0.02531075  0.1294118
## [3,]  0.9091115  0.9979643480 -0.99523824 -0.9741055
## [4,]  0.1971038  0.0482777512  0.01918192  0.1828391
```

On observe que la matrice ($M_1 = V_b^{-1}B_b$) à diagonaliser n'est pas symétrique.

Nous avons H = 2 valeurs propres non-nulles puisque H = min(K-1, p) = min(2, 4) = 2. Nous obtenons ($\lambda_1 = 0.766293$, $\lambda_2 = 0.121708$). Les vecteurs propres correspondent aux deux premières colonnes du champ ($vectors) de la solution.

8.1.2.4 Recherche de la représentation optimale – Solution (2)

Le problème d'analyse factorielle discriminante peut être formulé différemment. On cherche maximiser le rapport :

$$\max_{a} \frac{a^T B_b\, a}{a^T\, W_b\, a}$$

En passant par une optimisation sous contrainte, et en calculant le lagrangien, nous en venons à résoudre :

$$W_b^{-1} B_b\, a = \rho\, a$$

La solution revient à diagonaliser $(W_b^{-1} B_b)$. Les vecteurs propres sont identiques à précédemment (Solution 1), les valeurs propres sont liées par la relation :

$$\rho_h = \frac{\lambda_h}{1 - \lambda_h}$$

Ou, inversement,

$$\lambda_h = \frac{\rho_h}{1 + \rho_h}$$

Quel intérêt ? Les valeurs propres (ρ_h) s'additionnent d'un facteur à l'autre. Nous pouvons exprimer le pouvoir explicatif de chaque facteur sous forme de proportion.

Exemple « WINE »

Les calculs sous R sont toujours aussi simples.

```
#formation de la matrice à diagonaliser (solution 2)
M2 <- solve(Wb) %*% (Bb)
print(M2)

##              Temperature    Soleil     Chaleur      Pluie
## Temperature     1.780936  1.590820  0.10808539 -0.8489187
## Soleil          1.370806  1.268425  0.08128061 -0.6947282
## Chaleur        -4.720914 -5.275679 -0.24041111  3.2452899
## Pluie          -1.175957 -1.101436 -0.06914774  0.6084845
```

```r
#diagonalisation
sol2 <- eigen(M2,symmetric=FALSE)
print(sol2)

## eigen() decomposition
## $values
## [1]  3.278860e+00  1.385740e-01 -6.804973e-16  3.491695e-16
##
## $vectors
##            [,1]          [,2]        [,3]       [,4]
## [1,] -0.2878446  0.0003616251  0.09096624 -0.2179566
## [2,] -0.2276222  0.0416687914 -0.02221086  0.6289767
## [3,]  0.9091115 -0.9979643480 -0.99535231 -0.3002264
## [4,]  0.1971038 -0.0482777512  0.02248572  0.6831891
```

Les vecteurs propres sont identiques à précédemment (Solution 1 – section 8.1.2.3). Les valeurs sont ($\rho_1 = 3.2789$, $\rho_2 = 0.1386$). On peut aisément vérifier la relation entre (ρ_h) et (λ_h).

8.1.2.5 Recherche de la représentation optimale – Solution (3)

Les deux solutions ci-dessus sont décrites dans la très majorité des ouvrages en français. Mais elles ne correspondent pas aux résultats retournés par les logiciels (SAS Proc Candisc, SPSS, TANAGRA, lda() de MASS sous R). A titre de référence, voici les « coefficients canoniques bruts », correspondants aux coefficients de la fonction discriminante canonique, affichés par PROC CANDISC sous SAS (Figure 75).

Coefficients canoniques bruts			
Variable	Libellé	Can1	Can2
Temperature	Temperature	0.0085660455	-.0000462506
Soleil	Soleil	0.0067738690	-.0053292933
Chaleur	Chaleur	-.0270544919	0.1276361644
Pluie	Pluie	-.0058656650	0.0061745562

Figure 75 – Coefficients canoniques bruts – Proc Candisc SAS – DATA "WINE"

Comment obtenir ces coefficients ?

Différentes pistes sont possibles. L'une d'entre elles consiste à retravailler la matrice à diagonaliser (Lebart et al., 2000, section 3.3.2.c). En effet, les matrices ($V_b^{-1}B_b$) et ($W_b^{-1}B_b$) ne sont pas symétriques, et de dimensions (p x p), ce qui peut poser des difficultés aux librairies de calcul lorsque l'on traite des données comportant un grand nombre de descripteurs. Et, quoiqu'il en soit, une grande partie des calculs n'est pas nécessaire car le nombre de valeurs

propres non-nulles est souvent réduit : H = min (K – 1, p), (K) est très inférieur à (p) dans la très grande majorité des situations.

La matrice des covariances inter-classes peut être exprimée par le produit d'une matrice C par sa transposée ($V_b = CC^T$), où C est de dimension (p x K) de termes :

$$c_{jk} = \sqrt{\frac{n_k}{n}}\left(\bar{x}_{kj} - \bar{x}_j\right)$$

La résolution de ...

$$B_b a = \lambda\, V_b a$$

... est équivalente à la résolution de

$$CC^T V_b^{-1} Cb = \lambda Cb$$

c.-à-d. à la diagonalisation de la matrice ($C^T V_b^{-1} C$) de taille (K x K), nettement plus réduite. Elle est symétrique de surcroît. Les valeurs propres sont identiques à celles de la formulation initiale (λ_h). Et nous pourrons utiliser les vecteurs propres (b_h) pour produire les coefficients de la fonction discriminante canonique tels qu'ils sont affichés dans les logiciels.

Exemple « WINE »

De nouveau sous R, nous formons la matrice C, nous vérifions sa correspondance avec B_b, puis nous effectuons la diagonalisation.

```
#** passage par une matrice (K, K) (solution 3) **

#passer xb_k en matrice (moyennes conditionnelles)
mb_k <- as.matrix(xb_k[,2:(p+1)])
print(mb_k)

##      Temperature   Soleil  Chaleur    Pluie
## [1,]    3037.333 1126.417 12.08333 430.3333
## [2,]    3306.364 1363.636 28.54545 305.0000
## [3,]    3140.909 1262.909 16.45455 339.6364

#proportion des classes
print(pi_k)

## y
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294
```

```r
#créer la matrice C
C <- sapply(1:K,function(k){sqrt(pi_k[k])*(mb_k[k,]-xbar)})
print(C)

##                   [,1]       [,2]       [,3]
## Temperature -71.616789  84.455628  -9.654331
## Soleil      -71.829380  66.158328   8.865012
## Chaleur      -4.004273   5.529797  -1.347470
## Pluie        41.522128 -31.534745 -11.833704

#afficher Bb
print(Bb)

##             Temperature     Soleil    Chaleur      Pluie
## Temperature  12354.9238 10646.0470  766.80455 -5522.7217
## Soleil       10646.0470  9614.9726  641.52122 -5173.7007
## Chaleur        766.8046   641.5212   48.42853  -324.7011
## Pluie        -5522.7217 -5173.7007 -324.70111  2858.5638

#vérifier équivalence avec C x C^T /!\
print(C %*% t(C))

##             Temperature     Soleil    Chaleur      Pluie
## Temperature  12354.9238 10646.0470  766.80455 -5522.7217
## Soleil       10646.0470  9614.9726  641.52122 -5173.7007
## Chaleur        766.8046   641.5212   48.42853  -324.7011
## Pluie        -5522.7217 -5173.7007 -324.70111  2858.5638

#matrice à diagonaliser
M3 <- t(C) %*% solve(Vb) %*% C
print(M3)

##             [,1]        [,2]        [,3]
## [1,]  0.40774702 -0.36258652 -0.06329135
## [2,] -0.36258652  0.39642896 -0.01771969
## [3,] -0.06329135 -0.01771969  0.08382534

#diagonalisation
sol3 <- eigen(M3,symmetric=TRUE)
print(sol3)

## eigen() decomposition
## $values
## [1] 7.662929e-01 1.217084e-01 7.404215e-18
##
## $vectors
##             [,1]       [,2]       [,3]
## [1,]  0.71442131 -0.3696769 -0.5940885
## [2,] -0.69805851 -0.4349539 -0.5687965
## [3,] -0.04813021  0.8210689 -0.5687965
```

Nous remarquons que :

- (V_b) correspond bien à (CC^T).
- La matrice (M3) à diagonaliser est symétrique, de dimension (K x K).
- (H = 2) valeurs propres (λ_h) sont non-nulles.

- La matrice des vecteurs propres (b_h) est organisée en colonnes. Elle est naturellement de dimension (K x K), mais seules les (H = 2) premières (colonnes) vont servir par la suite.

8.1.2.6 Calcul des coefficients des fonctions discriminantes canoniques

Pour obtenir le vecteur (a_h) des coefficients canoniques pour l'axe (h), nous appliquons une première transformation au vecteur propre (b_h) obtenu à l'étape précédente (section 8.1.2.5). Elle est liée à l'utilisation de la matrice C :

$$\beta_h = (V_b^{-1} C)\, b_h$$

Puis nous appliquons une correction qui tient compte des effectifs, du nombre de classes, et des valeurs propres (λ_h).

$$a_h = \beta_h \sqrt{\frac{n - K}{n \times \lambda_h \times (1 - \lambda_h)}}$$

Ces coefficients s'appliquent sur les variables centrées. Pour que l'on ait des fonctions pouvant directement travailler à partir des variables brutes, nous devons calculer les constantes (les « intercept »).

$$a_{h0} = -\sum_{j=1}^{p} a_{hj} \bar{x}_j$$

Les fonctions discriminantes canoniques nous permettront de calculer les coordonnées factorielles des individus.

Exemple « WINE »

Nous repartons du résultat précédent sous R. Nous déduisons les coefficients des variables à partir des vecteurs et valeurs propres. Nous comparons nos résultats avec ceux de la fonction lda() du package « MASS ».

```
#moyenne des variables
print(xbar)
```

```
## Temperature      Soleil     Chaleur       Pluie
##  3157.88235  1247.32353    18.82353   360.44118

#nombre maximal de facteurs
H <- min(K-1,p)

#coef1 (beta dans le texte)
coef1 <- (solve(Vb) %*% C) %*% sol3$vectors[,1:H]
print(coef1)

##                     [,1]          [,2]
## Temperature -0.003796403  1.583637e-05
## Soleil      -0.003002125  1.824769e-03
## Chaleur      0.011990336 -4.370307e-02
## Pluie        0.002599616 -2.114190e-03

#coefficients des fonctions discriminantes canoniques
coef_ <- sapply(1:H,function(k){coef1[,k]*sqrt((n-K)/(n*sol3$values[k]*(1-sol3$values[k]))))})
print(coef_)

##                     [,1]          [,2]
## Temperature -0.008566046  4.625059e-05
## Soleil      -0.006773869  5.329293e-03
## Chaleur      0.027054492 -1.276362e-01
## Pluie        0.005865665 -6.174556e-03

#à titre de comparaison les résultats de lda() de MASS
library(MASS)
add <- lda(X,y)
print(add)

## Call:
## lda(X, y)
##
## Prior probabilities of groups:
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294
##
## Group means:
##        Temperature   Soleil  Chaleur    Pluie
## bad       3037.333 1126.417 12.08333 430.3333
## good      3306.364 1363.636 28.54545 305.0000
## medium    3140.909 1262.909 16.45455 339.6364
##
## Coefficients of linear discriminants:
##                      LD1           LD2
## Temperature -0.008566046 -4.625059e-05
## Soleil      -0.006773869 -5.329293e-03
## Chaleur      0.027054492  1.276362e-01
## Pluie        0.005865665  6.174556e-03
##
## Proportion of trace:
##    LD1    LD2
## 0.9595 0.0405
```

Les coefficients concordent parfaitement avec ceux de lda() de R / MASS qui fait référence auprès des statisticiens (Venables W.N., Ripley B.D., « Modern Applied Statistics with S », 4th Edition, Springer, 2002, section 12.1).

Reste à estimer les « intercept ».

```
#calcul de l'intercept
intercept_ <- -t(coef_) %*% matrix(xbar,nrow=p,ncol=1)
print(intercept_)

##           [,1]
## [1,] 32.876282
## [2,] -2.165279
```

En posant (X1 : Temperature, X2 : Soleil, X3 : Chaleur et X4 : Pluie), les (H = 2) fonctions discriminantes canoniques sont :

$$z_1 = 32.88 - 0.0857\, x_1 - 0.00677\, x_2 + 0.02705\, x_3 + 0.00587\, x_4$$
$$z_2 = -2.17 + 0.0005\, x_1 + 0.00533\, x_2 - 0.12764\, x_3 - 0.00617\, x_4$$

A titre de comparaison, voici les valeurs fournies par TANAGRA (composant « Canonical Discriminant Analysis ») (Figure 76).

Coefficients	Unstandardized	
Attribute	Root n°1	Root n°2
Temperature	-0.0085660	0.0000463
Soleil	-0.0067739	0.0053293
Chaleur	0.0270545	-0.1276362
Pluie	0.0058657	-0.0061746
constant	32.87628192	-2.16527944

Figure 76 – Canonical Discriminant Function de TANAGRA – Données WINE

Nous sommes parés pour calculer les coordonnées factorielles des individus et des centres de classes. Nous pourrons ainsi passer aux représentations graphiques qui font le charme de l'analyse factorielle discriminante.

Remarque : Notons que ces fonctions discriminantes canoniques sont utilisés pour calculer les coordonnées factorielles des individus supplémentaires (TUTO 3, section 3.4.2) et participent au processus d'affectation des classes (sections 8.2.1 et 8.2.2).

8.1.3 Représentations factorielles

Représenter graphiquement les individus en les étiquetant selon leur classe d'appartenance donne une bonne indication de la qualité de la représentation. Elle sera confirmée, ou pas, par

les positions relatives des centres de classes, qui sont primordiales puisque l'analyse factorielle discriminante a essayé explicitement de les séparer au maximum dans le repère factoriel.

8.1.3.1 Calcul des coordonnées des individus

Il suffit d'appliquer les coefficients des fonctions discriminantes sur la description des individus pour obtenir leurs coordonnées factorielles. Les valeurs sont conformes à celles de lda() de MASS. Une représentation graphique rend la chose plus concrète.

```
#calculer les coordonnées factorielles des individus
coord <- as.matrix(X) %*% coef_
coord <- t(apply(coord,1,function(v){v+intercept_}))
print(head(coord))

##            [,1]        [,2]
## [1,]  0.8825518  0.87153717
## [2,]  2.3254555  0.09422036
## [3,]  0.9948565 -0.83295725
## [4,]  2.7268621 -0.24724429
## [5,] -0.7435957 -1.72116676
## [6,] -2.2308894 -0.48431916

#à titre de comparaison celles de lda() de MASS - tout va bien
print(head(predict(add)$x))

##             LD1         LD2
## [1,]  0.8825518 -0.87153717
## [2,]  2.3254555 -0.09422036
## [3,]  0.9948565  0.83295725
## [4,]  2.7268621  0.24724429
## [5,] -0.7435957  1.72116676
## [6,] -2.2308894  0.48431916

#graphique
plot(coord[,1],coord[,2],col=c("cornflowerblue","darkseagreen","coral")[y],main="AFD - Données WINE",xlab=
"Z1",ylab="Z2",pch=19,ylim=c(-4,4))
abline(v=0,h=0,col="snow3")
legend(-4,4,legend=c('bad','good','medium'),col=c("cornflowerblue","darkseagreen","coral"),cex=0.7,pch=19)
```

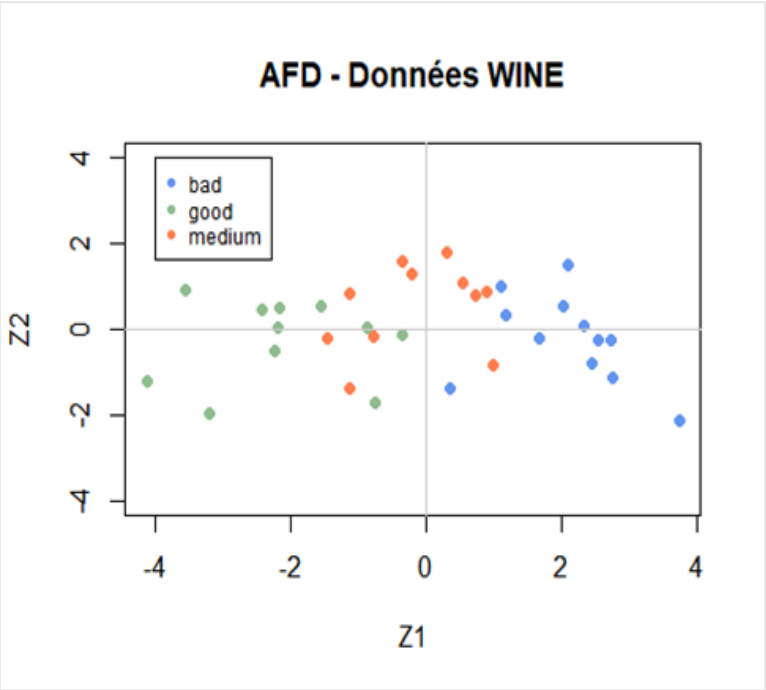


Figure 77 – Représentation des individus dans le plan factoriel – Données WINE

Les classes semblent assez bien discriminés sur le premier facteur. Le second ne paraît pas très décisif en revanche.

8.1.3.2 Représentation graphique avec les centres de classes

L'introduction des barycentres permet de mieux situer la qualité relative des facteurs dans la discrimination des classes. Ils sont calculés en appliquant les coefficients des fonctions discriminantes sur les moyennes conditionnelles dans l'espace originel.

```
#** calcul des centres de classes dans le repère factoriel **

#rappel des moyennes conditionnelles
print(mb_k)

##      Temperature   Soleil  Chaleur    Pluie
## [1,]    3037.333 1126.417 12.08333 430.3333
## [2,]    3306.364 1363.636 28.54545 305.0000
## [3,]    3140.909 1262.909 16.45455 339.6364

#application des coefficients des fonctions de classement
zb_k <- mb_k %*% coef_

#rajouter les constantes
zb_k <- t(apply(zb_k,1,function(v){v+intercept_}))
colnames(zb_k) <- c("zbar_1","zbar_2")
rownames(zb_k) <- levels(y)
print(zb_k)
```

```
##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#rajouter dans le graphique
plot(coord[,1],coord[,2],col=c("cornflowerblue","darkseagreen","coral")[y],main="AFD - Données W
INE",xlab="Z1",ylab="Z2",pch=19,ylim=c(-4,4))
abline(v=0,h=0,col="snow3")
legend(-4,4,legend=c('bad','good','medium'),col=c("cornflowerblue","darkseagreen","coral"),cex=0.7,pch=19)
points(zb_k[,1],zb_k[,2],col=c('midnightblue','green4','red'),pch=17,cex=1.5)
```

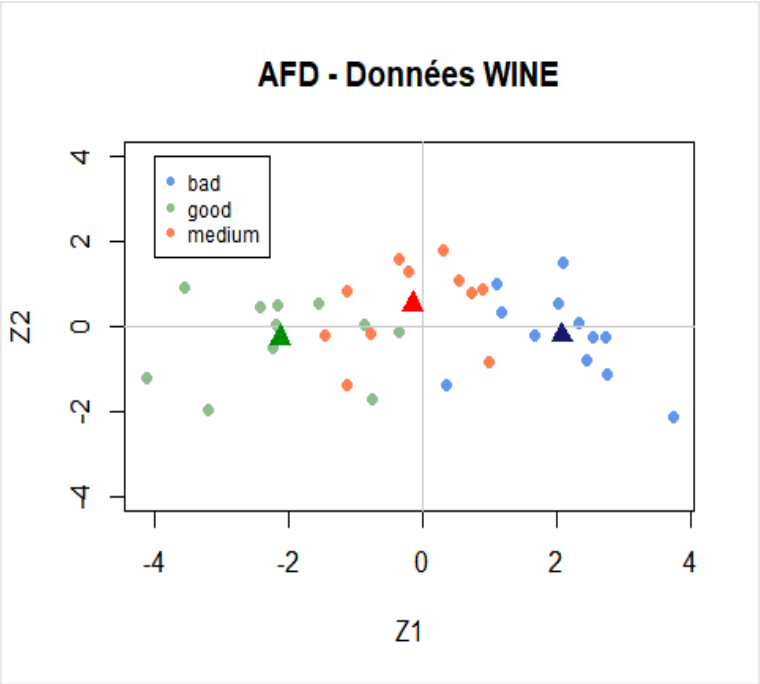


Figure 78 – Représentation des centres de classes dans le repère factoriel – Données WINE

Plus perceptible encore ici, la discrimination se joue essentiellement sur le premier facteur (Z1), avec une évolution décroissante de la qualité du vin de gauche à droite. Le second facteur (Z2) permet de distinguer un peu les vins « medium » des deux autres, mais avec une amplitude moindre. Dans la section suivante, nous verrons si son rôle est statistiquement négligeable.

8.1.4 Evaluation statistique des facteurs

Plusieurs marqueurs permettent de jauger la qualité de la représentation. Ils sont basés sur l'écartement entre les centres de classes, dans le repère pris globalement, où relativement à un ou plusieurs facteurs. L'objectif est d'apprécier leurs contributions dans la séparation des classes.

8.1.4.1 Distance entre centres de classes

Les distances entre les centres de classes sont un premier outil important. Dans le repère factoriel, qui est orthogonal, elles sont comptabilisées à l'aide d'une simple distance euclidienne. Elles doivent être équivalentes à la distance de Mahalanobis dans l'espace initial puisque l'objectif de l'analyse factorielle discriminante était prioritairement de les disposer de manière à restituer l'information disponible.

$$D_E^2(y_1, y_2) = \sum_{h=1}^{H} (\bar{z}_{1h} - \bar{z}_{2h})^2$$

$$D_E^2(y_1, y_2) = (\bar{z}_1 - \bar{z}_2)(\bar{z}_1 - \bar{z}_2)^T$$

Où $(\bar{z}_{kh})$ est la moyenne de la modalité (y_k) sur le facteur (h), $(\bar{z}_k)$ est le vecteur moyenne de la modalité (y_k).

Exemple « WINE »

Nous calculons et représentons les écarts – au sens de la distance euclidienne – entre centres de classes dans l'espace factoriel pour les données « WINE ».

```
#centres de classes dans le repère factoriel
print(zb_k)

##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#distance euclidienne entre les centres de classes
#pris deux à deux

#préparation de la structure
D2E <- matrix(0,nrow=K,ncol=K)

#boucle de calcul - la matrice est forcément symétrique
for (k1 in 1:(K-1)){
  for (k2 in (k1+1):K){
    D2E[k1,k2] <- sum((zb_k[k1,]-zb_k[k2,])^2)
    D2E[k2,k1] <- D2E[k1,k2]
  }
}

#affichage des valeurs
print(D2E)
```

```
##            [,1]      [,2]     [,3]
## [1,]  0.000000 17.652729 5.492267
## [2,] 17.652729  0.000000 4.519311
## [3,]  5.492267  4.519311 0.000000
```

```
#représentées dans le repère factoriel
plot(coord[,1],coord[,2],type="n",main="Distances entre barycentres",xlab="Z1",ylab="Z2",ylim=c(-4,4),xlim=c(-4,4))
abline(v=0,h=0,col="gray95")
legend(-4,4,legend=c('bad','good','medium'),col=c("cornflowerblue","darkseagreen","coral"),cex=0.7,pch=19)
points(zb_k[,1],zb_k[,2],col=c('midnightblue','green4','red'),pch=17,cex=1.5)
lines(zb_k[c(1,2),1],zb_k[c(1,2),2],lty=2,col="gray47")
text(0,-0.7,round(D2E[1,2],2),cex=0.85)
lines(zb_k[c(1,3),1],zb_k[c(1,3),2],lty=2,col="gray47")
text(1,0.5,round(D2E[1,3],2),cex=0.85)
lines(zb_k[c(2,3),1],zb_k[c(2,3),2],lty=2,col="gray47")
text(-1,0.5,round(D2E[2,3],2),cex=0.85)
```

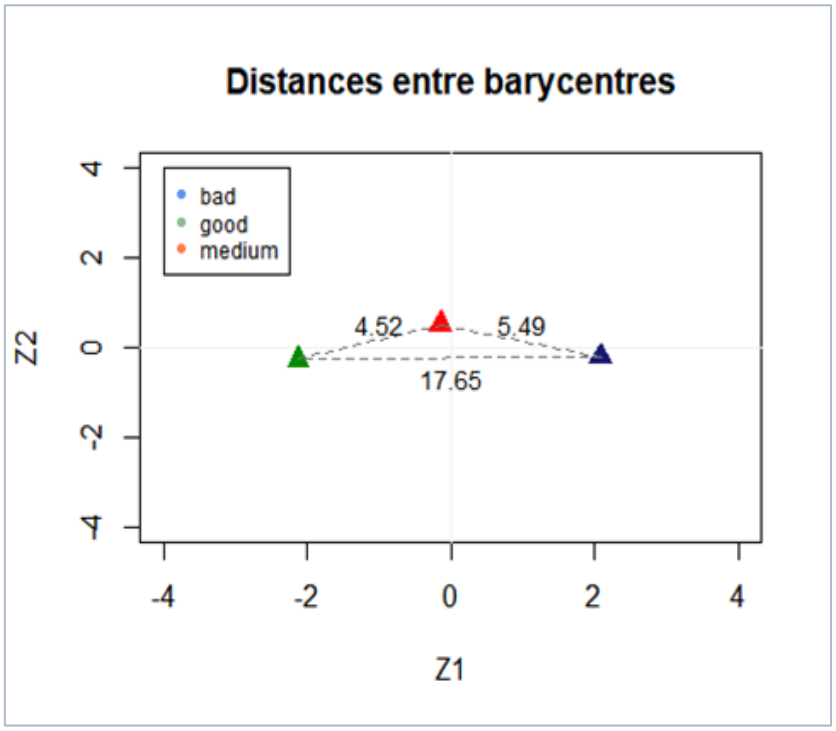


Figure 79 – Distances au carré entre centres de classes dans le repère factoriel – WINE

On devrait obtenir les mêmes valeurs si l'on calculait les distances de Mahalanobis dans le repère originel. Vérifions cela.

```
#matrice de variance covariance intra-classes - utilisé pour Mahalanobis
W <- n/(n-K)*Wb
print(W)
```

```
##             Temperature    Soleil   Chaleur      Pluie
## Temperature   7668.4555 1880.1515 461.32063  430.23558
## Soleil        1880.1515 6522.3346 169.20064 -158.00098
## Chaleur        461.3206  169.2006  53.68939  -34.82307
## Pluie          430.2356 -158.0010 -34.82307 5758.03910
```

```r
#inversion
invW <- solve(W)
print(invW)

##               Temperature        Soleil       Chaleur         Pluie
## Temperature  2.792825e-04 -2.016726e-05 -0.0023592977 -3.568953e-05
## Soleil      -2.016726e-05  1.684375e-04 -0.0003549582  3.982125e-06
## Chaleur     -2.359298e-03 -3.549582e-04  0.0402823491  4.101614e-04
## Pluie       -3.568953e-05  3.982125e-06  0.0004101614  1.789267e-04

#moyennes conditionnelles
print(mb_k)

##      Temperature   Soleil  Chaleur    Pluie
## [1,]    3037.333 1126.417 12.08333 430.3333
## [2,]    3306.364 1363.636 28.54545 305.0000
## [3,]    3140.909 1262.909 16.45455 339.6364

#matrice des distances
D2M <- matrix(0,nrow=K,ncol=K)

#boucle de calcul
for (k1 in 1:(K-1)){
  for (k2 in (k1+1):K){
    #écart entre les 2 vecteurs moyennes
    ecart <- mb_k[k1,,drop=FALSE]-mb_k[k2,,drop=FALSE]
    #distance de Mahalanobis
    D2M[k1,k2] <- ecart %*% invW %*% t(ecart)
    D2M[k2,k1] <- D2M[k1,k2]
  }
}

#affichage
print(D2M)

##           [,1]      [,2]     [,3]
## [1,]  0.000000 17.652729 5.492267
## [2,] 17.652729  0.000000 4.519311
## [3,]  5.492267  4.519311 0.000000
```

Alléluia ! C'est pour ce genre de choses que j'adore les statistiques !

8.1.4.2 Pouvoir discriminant des facteurs

Le pouvoir discriminant des facteurs est traduit par les valeurs propres qui leurs sont associées. Parce la méthode essaie de la maximiser explicitement lors de la recherche des solutions, le premier critère important est (λ_h) qui est égal au carré du rapport de corrélation $(\eta_h^2 = \eta_{z_h,y}^2)$. On le trouve sous le nom de « corrélation canonique au carré » dans les logiciels, ou « corrélation canonique » avec $(\sqrt{\lambda_h})$. L'ennui est que (λ_h) ne s'additionne pas d'un facteur à l'autre, nous ne pouvons pas décomposer la part d'explication portée par un facteur.

De fait, un autre critère (ρ_h) l'accompagne dans les tableaux de résultats sous l'appellation « valeurs propres de $(W_b^{-1}B_b)$ ». Rappelons que ($\rho_h = \frac{\lambda_h}{1-\lambda_h}$), mais ($\rho_h$) présente l'incommensurable avantage de pouvoir s'additionner. Nous pouvons ainsi calculer la contribution d'un facteur dans la discrimination des classes sous la forme de « proportion » d'explication :

$$\frac{\rho_h}{\sum_{h=1}^{H} \rho_h}$$

Ce dispositif peut être étendu à la caractérisation du rôle d'un ensemble de facteurs.

Exemple « WINE »

Nous revenons sur les différentes diagonalisations ci-dessus (sections 8.1.2.3et 8.1.2.4), nous affichons les valeurs propres. Nous faisons le parallèle avec les sorties de procédure lda() de MASS (TUTO 1), dont je conseille vraiment la lecture du code source[1].

```
#** reprise des solutions précédentes **

#nombre de facteurs
print(H)

## [1] 2

#solution 1 - pour les lambda_h, les valeurs non-nulles
#carré des corrélations canoniques
print(sol1$values[1:H])

## [1] 0.7662929 0.1217084

#solution 1' - les corrélations canoniques
print(sqrt(sol1$values[1:H]))

## [1] 0.8753816 0.3488673

#solution 2 - pour les rho_h, les valeurs non-nulles
print(sol2$values[1:H])

## [1] 3.278860 0.138574

#en proportion d'explication
print(round(sol2$values[1:H]/sum(sol2$values[1:H]),4))

## [1] 0.9595 0.0405
```

[1] Plutôt que de passer par une diagonalisation, elle utilise une décomposition en valeurs singulières. La préparation des données à présenter à la fonction svd() est très instructive sur la méthode programmée, l'analyse factorielle discriminante, très instructive également sur les bonnes pratiques de programmation sous R, en termes de manipulation des structures matricielles en particulier.

```r
#à comparer avec les "proportion of trace" de lda() / MASS
print(MASS::lda(X,y))

## Call:
## lda(X, y)
##
## Prior probabilities of groups:
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294
##
## Group means:
##        Temperature   Soleil  Chaleur    Pluie
## bad       3037.333 1126.417 12.08333 430.3333
## good      3306.364 1363.636 28.54545 305.0000
## medium    3140.909 1262.909 16.45455 339.6364
##
## Coefficients of linear discriminants:
##                      LD1           LD2
## Temperature -0.008566046 -4.625059e-05
## Soleil      -0.006773869 -5.329293e-03
## Chaleur      0.027054492  1.276362e-01
## Pluie        0.005865665  6.174556e-03
##
## Proportion of trace:
##    LD1    LD2
## 0.9595 0.0405
```

L'impression visuelle ci-dessous est confirmée (Figure 78), le second facteur joue très peu (4.05%) dans la discrimination des classes.

Voici les tableau « Roots and Wilks' Lambda » produit par TANAGRA sur les mêmes données (Figure 80). Nous retrouvons successivement : « eigenvalue » (ρ_h), « proportion » qui est sous une forme cumulée ici, « canonical R » ($\sqrt{\lambda_h}$). Les autres indicateurs seront explicités dans les sections qui viennent.

Roots and Wilks' Lambda

Root	Eigenvalue	Proportion	Canonical R	Wilks Lambda	CHI-2	d.f.	p-value
1	3.27886	0.95945	0.875382	0.205263	46.7122	8	0.000000
2	0.13857	1.00000	0.348867	0.878292	3.8284	3	0.280599

Test of H0: The canonical correlation in the current row and all that follow are zero (Bartlett's chi-square approximation)

Figure 80 – Tableau des valeurs propres – Données "WINE" – TANAGRA

8.1.4.3 Lambda de Wilks – Test global

Le lambda de Wilks reste l'indicateur clé pour caractériser la séparabilité générale des classes. Il peut être déterminé plus simplement dans l'espace factoriel :

$$\Lambda = \prod_{h=1}^{H}(1-\eta_h^2)$$

Exemple « WINE »

Nous calculons le lambda de Wilks de 2 manières pour les données WINE. La première est basée sur le ratio des dispersions intra-classes et totales (section 2.2.2), la seconde, dans le contexte de l'analyse factorielle discriminante, sur la formule exposée dans cette section. Les résultats sont bien évidemment cohérents puisque c'est le même concept qui est mesuré : l'éloignement entre les centres de classes.

```
#lambda de Wilks - Formule 1 (contexte ADL)
LW1 <- det(Wb)/det(Vb)
print(LW1)

## [1] 0.205263

#nombre total de facteurs
print(H)

## [1] 2

#récupération des carrés de rapports de corrélation
RC2 <- sol1$values[1:H]
print(RC2)

## [1] 0.7662929 0.1217084

#lambda de Wilks - Formule 2 (contexte AFD)
LW2 <- prod((1-RC2))
print(LW2)

## [1] 0.205263
```

Test de significativité globale. Le test MANOVA (analyse de variance multivariée), encadré toujours par ses présupposés de distributions gaussiennes et homoscédastiques, s'enrichit dans ce nouveau contexte. L'hypothèse nulle dans le repère originel retranscrivait l'empiètement entre les barycentres conditionnels :

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_K$$

Elle devient relative aux rapports de corrélation associés aux facteurs ici. La question de référence est « les rapports de corrélation sont-ils tous nuls ? » :

$$H_0 : \eta_1^2 = \eta_2^2 = \cdots = \eta_H^2 = 0$$

Les deux transformations, Bartlett (loi du χ^2) et Rao (loi de Fisher), avec les mêmes degrés de liberté, restent de mise (section 2.2.2).

Appliquons les formules à nos données « WINE ».

```
#** Test de Bartlett **

#transformation
LB <- -(n - 1 - (p + K)/2) * log(LW2)
print(LB)

## [1] 46.71217

#degré de liberté
print(p * (K - 1))

## [1] 8

#probabilité critique
print(pchisq(LB,df = p * (K - 1), lower.tail = FALSE))

## [1] 1.739815e-07


#** Test de RAO **

#A
A <- n - K - (p - K + 2)/2

#B
B <- p^2 + (K - 1)^2 - 5
B <- ifelse(B > 0, sqrt((p^2 * (K - 1)^2 - 4)/(B)), 1)

#C
C <- (p * (K - 1) - 2)/2

#F de RAO
FRAO <- ((1 - LW2^(1/B))/(LW2^(1/B))) * ((A * B - C)/(p * (K - 1)))
print(FRAO)

## [1] 8.450507

# ddl1
print(p * (K - 1))

## [1] 8

#ddl2
print(A * B - C)

## [1] 56

#p-value
print(pf(FRAO, df1 = p * (K - 1), df2 = A * B - C, lower.tail = FALSE))
```

```
## [1] 1.890358e-07
```

Les valeurs des statistiques de test et les degrés de liberté recoupent celles du composant MANOVA de TANAGRA (Figure 81). C'est toujours rassurant.

Descriptive stat. (Mean)					Tests results		
Group	medium	bad	good	ALL	Stat	Value	p-value
Group Size	11	12	11	34	Wilks' Lambda	0.2053	-
Temperature	3140.91	3037.33	3306.36	3157.88	Bartlett -- C(8)	46.7122	0
Soleil	1262.91	1126.42	1363.64	1247.32	Rao -- F(8, 56)	8.4505	0
Chaleur	16.45	12.08	28.55	18.82			
Pluie	339.64	430.33	305.00	360.44			

Figure 81 – Test MANOVA – Données "WINE" – TANAGRA

8.1.4.4 Tester un ensemble de facteurs

« Combien de facteurs faut-il retenir ? » est une question récurrente de l'analyse factorielle. Pour l'analyse discriminante, elle est traduite par « est-ce que nous pouvons éliminer (négliger) les q derniers facteurs ? ». Le cadre est différent de la sélection de variables. Les composantes viennent en bloc. L'explication fournie par l'une d'entre elles dépend du pouvoir discriminant des précédentes. Il n'est pas question de tester individuellement un facteur, au risque de l'éliminer, tout en gardant les suivants. Ce qui paraît très difficile de toute manière puisqu'elles sont d'importances décroissantes.

Formellement, la question de référence est « les rapports de corrélation des 'q' derniers facteurs sont-ils tous nuls ? » :

$$H_0 : \eta^2_{H-q+1} = \eta^2_{H-q+2} = \cdots = \eta^2_H$$

La statistique de test est :

$$\Lambda_q = \prod_{h=H-q+1}^{H} (1 - \eta_h^2)$$

Remarque : Tester les (q = H) facteurs revient au test global de la section précédente.

Les transformations de Bartlett et de Rao doivent être adaptées pour prendre en considération le nombre de composantes « q » à éprouver (Tenenhaus, 2007, section 3.3.2).

Transformation de Bartlett. Elle est toujours aussi simple :

$$\chi^2 = -\left(n - 1 - \frac{p+K}{2}\right)\ln \Lambda_q$$

Elle suit une loi du χ^2 à $[q\,(p - K + q + 1)]$ degrés de liberté.

Transformation de Rao. Elle est un peu moins simple, mais elle est réputée plus précise sur les petits effectifs :

$$F = \left(\frac{1 - \Lambda_q^{1/B}}{\Lambda_q^{1/B}}\right) \times \left[\frac{A \times B - C}{q \times (p - K + q + 1)}\right]$$

Où

$$A = n - 1 - \frac{p+K}{2}$$

$$B = \begin{cases} \sqrt{\dfrac{q^2(p-K+q+1)^2 - 4}{(p-K+q+1)^2 + q^2 - 5}}\,, & si\ (p-K+q+1)^2 + q^2 - 5 > 0 \\ 1\,, & sinon \end{cases}$$

$$C = \frac{q(p-K+q+1) - 2}{2}$$

Elle suit une loi de Fisher à $[\ q\,(p-K+q+1)\,,\ A \times B - C\ ]$ degrés de liberté.

En pratique, j'ai rarement constaté de grandes divergences entre ces deux approximations.

Exemple WINE

Nous testons la suppression du ($q = 1$) dernier facteur qui semble peu pertinent sur le graphique des barycentres conditionnels (Figure 78). Nous verrons si les résultats statistiques rejoignent l'intuition visuelle.

Pour la transformation de Bartlett.

```
#** test du dernier facteur **
q <- 1
```

```r
#stat. de test
LQ <- 1 - sol1$values[2]
print(LQ)

## [1] 0.8782916

#transformation de BARTLETT
BQ <- -(n - 1 - (p + K) / 2) * log(LQ)
print(BQ)

## [1] 3.82841

#degrés de liberté
print(q * (p - K + q + 1))

## [1] 3

#p-value
print(pchisq(BQ,q*(p-K+q+1),lower.tail=FALSE))

## [1] 0.2805988
```

La statistique de Bartlett, les degrés de liberté et la p-value sont raccords avec la dernière ligne du tableau des valeurs propres de TANAGRA qui retranscrit le même test (Figure 80). Supprimer le deuxième et dernier facteur ne dégraderait pas significativement la qualité de la représentation.

Passons à la transformation de RAO.

```r
#** transformation de RAO **

#A
A <- n - 1 - (p + K)/2

#B
B <- (p - K + q + 1)^2 + q^2 - 5
B <- ifelse(B > 0, sqrt((q^2 * (p - K + q + 1)^2 - 4)/(B)), 1)

#C
C <- (q * (p - K + q + 1) - 2)/2

#F de RAO
BRAO <- ((1 - LQ^(1/B))/(LQ^(1/B))) * ((A * B - C)/(q * (p - K + q + q)))
print(BRAO)

## [1] 1.339549

# ddl1
print(q * (p - K + q + 1))

## [1] 3

#ddl2
print(A * B - C)

## [1] 29
```

```r
#p-value
print(pf(BRAO, df1 = q * (p - K + q + 1), df2 = A * B - C, lower.tail = FALSE))

## [1] 0.280785
```

Nous retrouvons la dernière ligne de la sortie de la procédure CANDISC de SAS (Figure 82). « Rapport de vraisemblance » est l'indicateur (Λ_q).

Test de H0 : les corrélations canoniques de la ligne en cours et toutes celles qui suivent sont égales à zéro				
Rapport de vraisemblance	Valeur de F approchée	DDL num.	DDL den.	Pr > F
0.20526297	8.45	8	56	<.0001
0.87829160	1.34	3	29	0.2808

Figure 82 – Test de significativité des "q" derniers facteurs – Proc Candisc – SAS – Données WINE

8.1.5 Interprétation des facteurs

Il reste une dernière étape importante dans la partie descriptive de l'analyse factorielle discriminante : comprendre la nature des facteurs. Elle est nécessaire pour mieux saisir les caractéristiques qui différencient les classes.

L'interprétation repose sur différents types de corrélations : totales, intraclasses, interclasses (Tenenhaus, 2007, section 10.2.2). Voici le tableau « Factor Structure Matrix » (Structure Canonique) de TANAGRA sur les données « WINE » (Figure 83).

Factor Structure Matrix - Correlations

Root	Root n1		
Descriptors	Total	Within	Between
Temperature	-0.9006	-0.7242	-0.9865
Soleil	-0.8967	-0.7013	-0.9987
Chaleur	-0.7705	-0.5254	-0.9565
Pluie	0.6628	0.3982	0.9772

Figure 83 – Structure canonique – Données "WINE"

Dans les sous-sections qui suivent, nous détaillons leur mode de calcul et la lecture que l'on peut en faire pour comprendre l'influence des variables dans la définition des facteurs. Nous ferons un focus sur la relation entre le premier facteur (Z1) et la variable « température » c.-à-d. les valeurs de la première ligne du tableau (Figure 83).

8.1.5.1 Structure canonique totale

La structure canonique totale mesure la corrélation brute entre le facteur et la variable étudiée, sans tenir compte de la structuration en classes. Elle caractérise une relation entre les deux entités avec un concept que nous connaissons bien (Rakotomalala R., « Analyse de corrélation », version 1.1, mars 2015, chapitre 2). Elle a le mérite de la simplicité.

Nous calculons la corrélation entre (Z1) et (Température), puis nous représentons le nuage de points associé dans le plan.

```
#rappel des coordonnées factorielles
print(head(coord))

##            [,1]        [,2]
## [1,]  0.8825518  0.87153717
## [2,]  2.3254555  0.09422036
## [3,]  0.9948565 -0.83295725
## [4,]  2.7268621 -0.24724429
## [5,] -0.7435957 -1.72116676
## [6,] -2.2308894 -0.48431916

#récupération du premier facteur
z1 <- coord[,1]

#corrélation avec température
print(round(cor(X$Temperature,z1),4))

## [1] -0.9006

#graphique
plot(X$Temperature,z1,main="Température vs. z1")
```

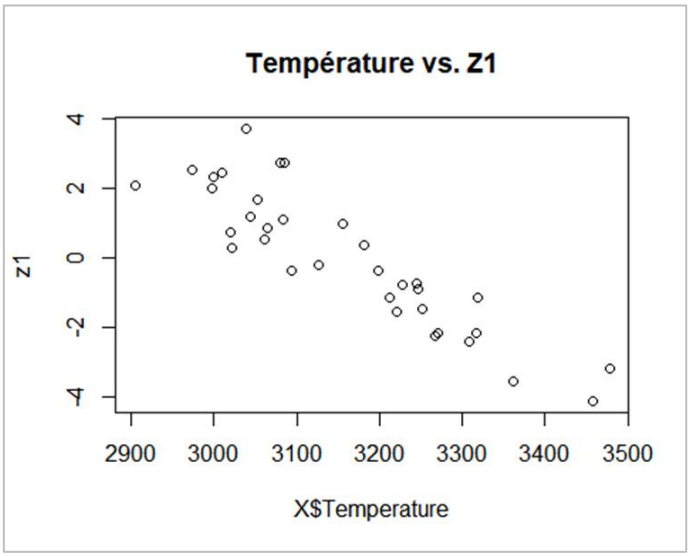


Figure 84 – Corrélation totale – Température vs. Z1 – Données "WINE"

Nous avons calculé la corrélation avec la fonction cor() usuelle de R. Nous retrouvons la valeur de la « corrélation totale » négative de -0.9006 entre le premier facteur et température : Z1 est une fonction décroissante de la température. Sachant par ailleurs que le facteur Z1 lui-même permet de discriminer les classes selon une qualité décroissante (Figure 78), on en déduit que les hautes températures sont plutôt bénéfiques aux vins de Bordeaux.

8.1.5.2 Structure canonique intra-classe

La structure canonique intra-classe calcule la corrélation avec les facteurs après centrage des valeurs par les moyennes des classes d'appartenance. Elle permet de caractériser le lien après avoir annihilé l'organisation en groupes des observations. On peut aussi la voir comme une corrélation partielle c.-à-d. une corrélation entre le facteur et la variable en contrôlant l'effet de l'appartenance aux classes ; ou encore comme un indicateur synthétique des corrélations internes aux classes.

Sous R, il faut centrer la variable et le facteur selon les moyennes des classes d'appartenance, puis calculer la corrélation.

```
#vecteur indiquant l'appartenance aux classes
print(y)

##  [1] medium bad    medium bad    good   good   bad    bad    bad    medium
## [11] good   bad    bad    good   medium medium medium bad    medium good
## [21] medium good   medium good   medium good   medium bad    good   good
## [31] bad    good   bad    bad
## Levels: bad good medium

#moyennes conditionnelles
print(mb_k)

##      Temperature   Soleil  Chaleur    Pluie
## [1,]    3037.333 1126.417 12.08333 430.3333
## [2,]    3306.364 1363.636 28.54545 305.0000
## [3,]    3140.909 1262.909 16.45455 339.6364

#nommer les lignes
rownames(mb_k) <- levels(y)
print(mb_k)

##        Temperature   Soleil  Chaleur    Pluie
## bad       3037.333 1126.417 12.08333 430.3333
## good      3306.364 1363.636 28.54545 305.0000
## medium    3140.909 1262.909 16.45455 339.6364
```

```r
#vecteur des valeurs moyennes indéxées par les classes
moyTemp_k <- mb_k[y,'Temperature']
print(moyTemp_k)

##   medium      bad   medium      bad     good     good      bad      bad
## 3140.909 3037.333 3140.909 3037.333 3306.364 3306.364 3037.333 3037.333
##      bad   medium     good      bad      bad     good   medium   medium
## 3037.333 3140.909 3306.364 3037.333 3037.333 3306.364 3140.909 3140.909
##   medium      bad   medium     good   medium     good   medium     good
## 3140.909 3037.333 3140.909 3306.364 3140.909 3306.364 3140.909 3306.364
##   medium     good   medium      bad     good     good      bad     good
## 3140.909 3306.364 3140.909 3037.333 3306.364 3306.364 3037.333 3306.364
##      bad      bad
## 3037.333 3037.333

#valeurs centrées sur les moyennes selon les classes
cTemp <- X$Temperature - moyTemp_k
print(cTemp)

##       medium          bad       medium          bad         good         good
##  -76.9090909  -37.3333333   14.0909091   47.6666667  -61.3636364  -39.3636364
##          bad          bad          bad       medium         good          bad
##   42.6666667  -63.3333333    0.6666667  177.0909091   10.6363636  144.6666667
##          bad         good       medium       medium       medium          bad
##  -39.3333333  -85.3636364 -121.9090909 -118.9090909  -46.9090909  -28.3333333
##       medium         good       medium         good       medium         good
##   86.0909091    1.6363636   71.0909091   54.6363636  -79.9090909  171.6363636
##       medium         good       medium          bad         good         good
##  -14.9090909  151.6363636  111.0909091   14.6666667  -36.3636364 -108.3636364
##          bad         good          bad          bad
## -133.3333333  -59.3636364   45.6666667    5.6666667

#moyennes conditionnelles sur les facteurs
print(zb_k)

##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#valeurs centrées sur les moyennes des classes
cz1 <- z1 - zb_k[y,1]
print(cz1)

##      medium         bad      medium         bad        good        good
##  1.02885833  0.24620848  1.14116299  0.64761508  1.37836721 -0.10892639
##         bad         bad         bad      medium        good         bad
##  0.66774796  0.45458372  1.65162921 -0.98409971 -0.05276857 -1.72258996
##         bad        good      medium      medium      medium         bad
## -0.05816568  0.56985120  0.87576794  0.45236419 -0.19716840  0.37523628
##      medium        good      medium        good      medium        good
## -0.63953040 -0.28791339 -0.99170967 -1.41332537  0.69821954 -1.06015098
##      medium        good      medium         bad        good        good
## -0.06337443 -1.99721160 -1.32049037 -0.40309216 -0.04516537  1.76951916
##         bad        good         bad         bad
##  0.02300398  1.24772412 -0.98482615 -0.89735075

#corrélation intra-classe
print(round(cor(cTemp,cz1),4))

## [1] -0.7242
```

C'est bien la valeur en première ligne et deuxième colonne de notre tableau des structures canoniques (Figure 83). Le graphique montre que les nuages conditionnels ont été en réalité mélangées avant de calculer la corrélation (Figure 85).

```
#graphique
plot(cTemp,cz1,main="Température vs. Z1 (intra-classe)",col=c("cornflowerblue","darkseagreen","coral")[y],pch=19)
```

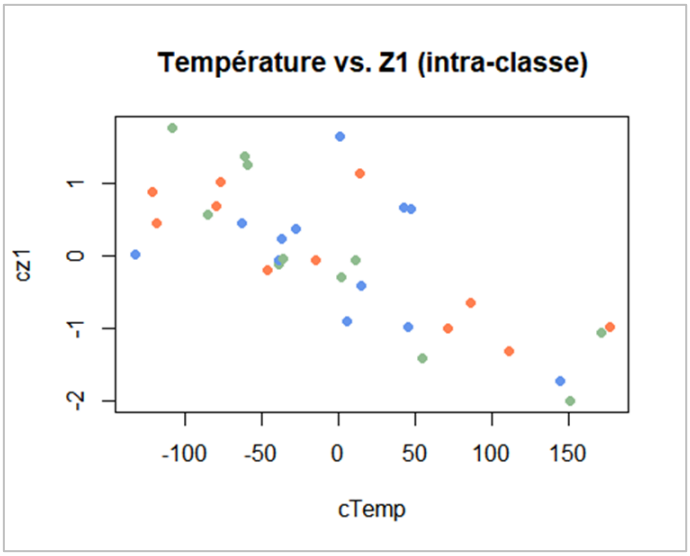


Figure 85 – Corrélation intra-classe – Température vs. Z1 – Données "WINE"

On comprend la rhétorique mais, très honnêtement, la lecture directe de cet indicateur n'est pas des plus intéressantes. Elle le devient si l'on observe un gap fort ou même une incohérence avec la corrélation totale. Là, oui, il faut s'interroger. Ce type de phénomène est souvent annonciateur de configurations particulières dans la conformation des classes.

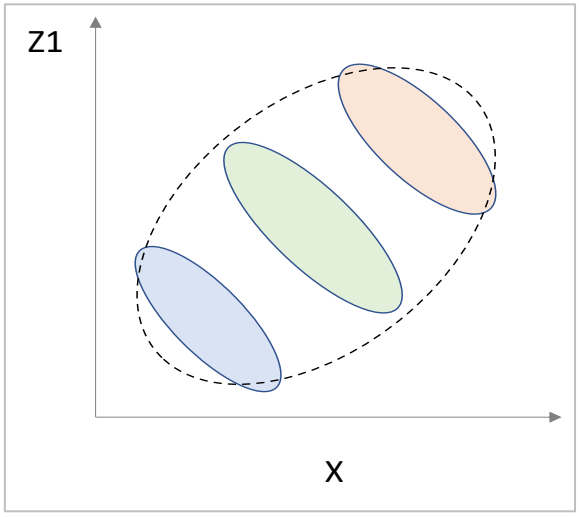


Figure 86 – Incohérence entre corrélation totale et corrélation intra-classe

Dans l'exemple ci-dessus (Figure 86), la corrélation totale entre la variable X et le facteur Z1 est positive (nuage de points global en pointillés), mais la corrélation intra-classe est négative (la corrélation est négative dans chaque sous-nuage, l'indicateur synthétique est négatif).

8.1.5.3 Structure canonique interclasse

La corrélation interclasse est calculée sur les centres de classes pondérés par leurs effectifs. Elle permet d'exacerber l'organisation en classes des données. Elle est la plus intéressant en théorie puisqu'elle permet d'identifier le rôle de la variable dans la discrimination des classes sur le facteur étudié. Mais elle est calculée sur K points, très peu, et lorsque que (K = 2), elle prend mécaniquement la valeur de -1 ou +1, très difficilement interprétable.

Sous R, nous formons une structure temporaire pour récupérer les moyennes conditionnelles de Z1 et « Température », puis nous lançons le calcul de la corrélation en passant en paramètres de poids les effectifs des classes.

```
#les effectifs par classe
print(n_k)

## y
##    bad   good medium
##     12     11     11

#moyennes par classe des variables
print(mb_k)

##        Temperature   Soleil  Chaleur    Pluie
## bad       3037.333 1126.417 12.08333 430.3333
## good      3306.364 1363.636 28.54545 305.0000
## medium    3140.909 1262.909 16.45455 339.6364

#moyennes par classe des facteurs
print(zb_k)

##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#former une structure ad-hoc
temporaire <- data.frame(v=mb_k[,'Temperature'],z=zb_k[,1])
print(temporaire)

##               v          z
## bad    3037.333  2.0792470
## good   3306.364 -2.1219630
## medium 3140.909 -0.1463065
```

```
#calculer la corrélation - cov.wt() du package 'stats'
bcor <- result <- cov.wt(x=temporaire,wt=as.numeric(n_k),cor=TRUE)
print(round(bcor$cor,4))

##         v       z
## v  1.0000 -0.9865
## z -0.9865  1.0000
```

La corrélation interclasse entre le facteur Z1 et « Température » est **-0.9865** (1ère ligne – 3ème colonne dans Figure 83). L'alignement des points dans le graphique est manifeste (Figure 87), « Température » joue un rôle important dans la capacité de Z1 à discriminer les classes.

```
#graphique
plot(temporaire,main="Température vs. Z1 (interclasse)",col=c("cornflowerblue","darkseagreen","coral"),pch=18,cex=1.75)
```

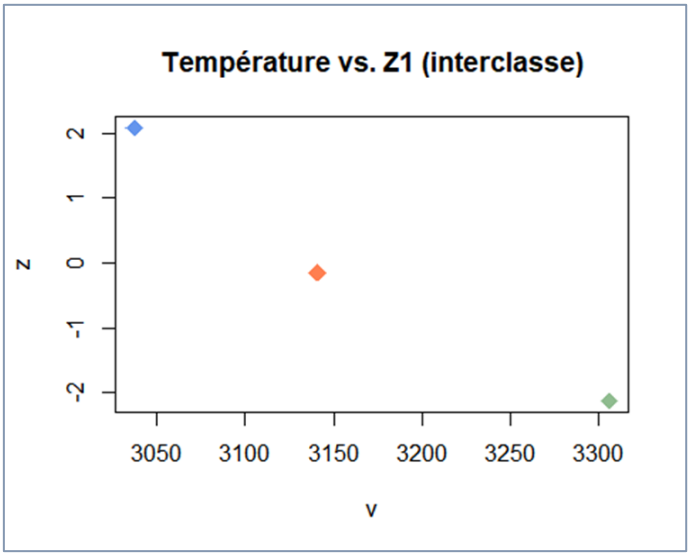


Figure 87 – Corrélation interclasse – Température vs. Z1 – Données "WINE"

8.2 Affectation des classes dans le repère factoriel

Même si ce n'est pas sa vocation première, l'analyse factorielle discriminante peut être utilisée pour classer des individus supplémentaires avec un mécanisme très simple d'affectation au centre de classe le plus proche. La procédure predict.lda() de MASS sous R procède ainsi par exemple.

Ce dispositif permet d'introduire éventuellement une stratégie de régularisation en ne sélectionnant que les facteurs réputés discriminants pour le classement. Lorsque (K = 2), nous n'avons que (H = 1) facteur, ce n'est pas trop possible. Mais lorsque K est élevé, elle peut être

décisive (Hastier et al., 2017, section 4.3.3 « Reduced-Rank Discriminant Analysis », en particulier la figure 4.10). Pour déterminer le nombre de facteurs à retenir, nous pouvons exploiter les tests statistiques (section 8.1.4.4) ou, plus prosaïquement, utiliser des grilles de recherche avec une estimation des performances en validation croisée.

8.2.1 Affectation basée sur la distance aux classes

Nous souhaitons classer un nouveau millésime avec les caractéristiques (Température : 3000, Soleil : 1100, Chaleur : 20, Pluie : 300) (Tenenhaus, 2007, page 365). Deux étapes sont nécessaires pour le classer : (1) calculer ses coordonnées dans le repère factoriel ; (2) identifier le centre de classe qui lui est le plus proche.

8.2.1.1 Calcul des coordonnées factorielles

Nous utilisons les fonctions discriminantes linéaires pour obtenir les coordonnées d'un individu supplémentaire. Pour rappel, elles étaient pour les données « WINE » (section 8.1.2.6) :

$$z_1 = 32.88 - 0.0857\, x_1 - 0.00677\, x_2 + 0.02705\, x_3 + 0.00587\, x_4$$
$$z_2 = -2.17 + 0.0005\, x_1 + 0.00533\, x_2 - 0.12764\, x_3 - 0.00617\, x_4$$

Sous R :

```
#coefficients de projection
print(coef_)

##                     [,1]          [,2]
## Temperature -0.008566046  4.625059e-05
## Soleil      -0.006773869  5.329293e-03
## Chaleur      0.027054492 -1.276362e-01
## Pluie        0.005865665 -6.174556e-03

#intercept
print(intercept_)

##           [,1]
## [1,] 32.876282
## [2,] -2.165279

#individu supplémentaire
indSup <- matrix(c(3000,1100,20,300),nrow=1,ncol=p,byrow = TRUE)
colnames(indSup) <- colnames(X)
print(indSup)

##      Temperature Soleil Chaleur Pluie
## [1,]        3000   1100      20   300
```

```
#coordonnées factorielles
ZSup <- indSup %*% coef_ + t(intercept_)
print(round(ZSup,4))

##        [,1]    [,2]
## [1,] 2.0277 -0.5694
```

L'individu supplémentaire a pour coordonnées (2.0277, -0.5694). Placé parmi les points de la base d'apprentissage (Figure 88)...

```
#placement dans le plan factoriel
plot(coord[,1],coord[,2],col=c("cornflowerblue","darkseagreen","coral")[y],main="A
FD - Individu supplémentaire",xlab="Z1",ylab="Z2",pch=19,ylim=c(-4,4))
abline(v=0,h=0,col="snow3")
legend(-4,4,legend=c('bad','good','medium'),col=c("cornflowerblue","darkseagreen",
"coral"),cex=0.75,pch=19)
points(zb_k[,1],zb_k[,2],col=c('blue','green4','red'),pch=17,cex=1.25)
points(ZSup,pch=19,cex=1.25,col="violet")
```

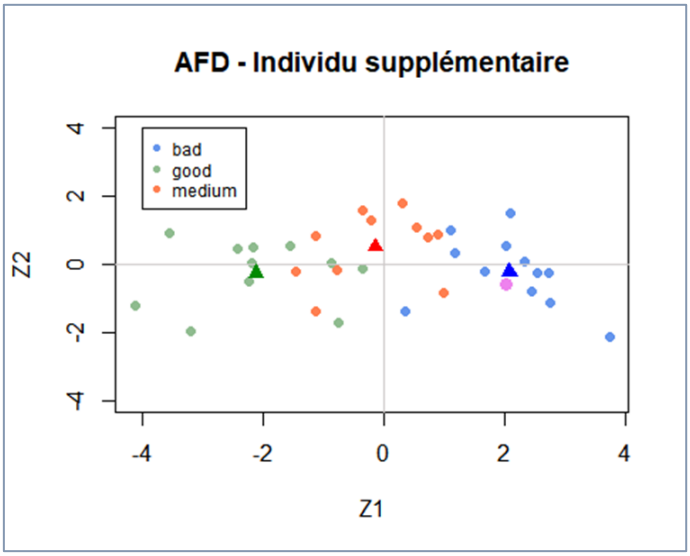


Figure 88 – Position de l'individu supplémentaire (violet) dans le repère factoriel – Données "WINE"

... le situe très proche du barycentre de la classe « bad ».

8.2.1.2 Distances euclidiennes aux centres de classes

Une perception visuelle est sympathique, un indicateur statistique est toujours plus rassurant.

Calculons la distance aux centres de classes de cette nouvelle observation ω.

$$D_E^2(y_k, \omega) = \sum_{h=1}^{H} (z_h(\omega) - \bar{z}_{kh})^2$$

Où $z_h(\omega)$ est la coordonnée de l'individu ω pour le facteur z_h ; $\bar{z}_{kh}$ est la moyenne de la modalité (y_k) sur le facteur (h).

```
#barycentres des classes
print(zb_k)

##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#calculer la distance avec les barycentres
DESup <- apply(zb_k,1,function(v){sum((v-ZSup)^2)})
print(DESup)

##        bad       good     medium
##  0.1239104 17.3080827  5.8980158
```

$D_E^2(bad, X(\omega)) = 0.1239104$ correspond à la distance minimale, et de loin. Visiblement, le nouveau millésime est un mauvais vin.

8.2.1.3 Equivalence avec la distance de Mahalanobis

Nous avions vu précédemment (section 8.1.4.1) que les écarts entre les centres de classes, mesurés avec la distance de Mahalanobis dans l'espace initial, avec la distance euclidienne dans l'espace factoriel, étaient identiques. Parce que l'analyse factorielle discriminante vise justement à les respecter. Est-ce que le même principe est applicable au calcul des distances aux barycentres conditionnels des individus supplémentaires ?

Vérifions-le sur nos données « WINE ».

```
#inverse de la matrice W
print(invW)

##               Temperature        Soleil       Chaleur         Pluie
## Temperature  2.792825e-04 -2.016726e-05 -0.0023592977 -3.568953e-05
## Soleil      -2.016726e-05  1.684375e-04 -0.0003549582  3.982125e-06
## Chaleur     -2.359298e-03 -3.549582e-04  0.0402823491  4.101614e-04
## Pluie       -3.568953e-05  3.982125e-06  0.0004101614  1.789267e-04

#moyennes conditionnelles
print(mb_k)

##        Temperature   Soleil  Chaleur    Pluie
## bad       3037.333 1126.417 12.08333 430.3333
## good      3306.364 1363.636 28.54545 305.0000
## medium    3140.909 1262.909 16.45455 339.6364
```

```r
#écarts de l'individu avec les moyennes conditionnelles
ecartSup <- t(apply(mb_k,1,function(v){v-indSup}))
print(ecartSup)

##             [,1]      [,2]      [,3]      [,4]
## bad     37.33333  26.41667 -7.916667 130.33333
## good   306.36364 263.63636  8.545455   5.00000
## medium 140.90909 162.90909 -3.545455  39.63636

#distance de Mahalanobis... pas les mêmes valeurs
DMSup <- diag(ecartSup %*% invW %*% t(ecartSup))
print(DMSup)

##       bad      good    medium
##  6.407816 23.591989 12.181922
```

$D_M^2(y_k, X(\omega))$ prend des valeurs différentes. Mais, et c'est ici que l'affaire devient très intéressante, les différences avec les distances euclidiennes sont les mêmes quelles que soient les classes.

```r
#différence entre D^2_E et D^2_M : pas les mêmes valeurs, mais même comportement
print(DMSup-DESup)

##      bad     good   medium
## 6.283906 6.283906 6.283906
```

Le gap de valeurs ne dépend donc pas de la classe d'appartenance. Cela signifie que le comportement en classement est strictement identique, avec les mêmes probabilités d'affectation.

8.2.2 Distance généralisée

Il manque la prise en compte de la prévalence empirique des classes pour que l'équivalence entre l'analyse factorielle discriminante et l'analyse discriminante linéaire tel que nous l'avions développé dans cet ouvrage soit totale. La distance généralisée permet d'y remédier (TUTO 3, section 3.4.5) :

$$D_G^2(y_k, \omega) = D_E^2(y_k, \omega) - 2 \times \ln \hat{\pi}_k$$

Sachant qu'il faut minimiser cette distance pour affecter un individu à une classe, nous utilisons une variante de la transformation softmax pour déduire les probabilités d'affectation :

$$P(Y = y_k \,/\, X(\omega)) \;=\; \frac{e^{-0.5 \times D_G^2(y_k\,,\omega)}}{\sum_{c=1}^{K} e^{-0.5 \times D_G^2(y_c\,,\omega)}}$$

Appliquons ces formules à notre millésime supplémentaire.

```
#proportion des classes
print(pi_k)

## y
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294

#distances de l'individu supplémentaire
print(DESup)

##        bad       good     medium
##  0.1239104 17.3080827  5.8980158

#distance généralisée
DGSup <- DESup - 2 * log(pi_k)
print(DGSup)

## y
##       bad      good    medium
##  2.206818 19.565013  8.154946

#transformation
ProbaSup <- exp(-0.5 * DGSup)/sum(exp(-0.5 * DGSup))
print(ProbaSup)

## y
##          bad         good       medium
## 0.9512346363 0.0001618093 0.0486035544
```

Le millésime a 95.12% de chances d'être une piquette. Argh ! Il n'y a plus d'espoir.

8.3 Déduction des fonctions de classement explicites

L'affectation aux centres de classes fonctionne bien en pratique, mais le mécanisme de décision n'est pas très lisible, et le déploiement dans les systèmes d'information n'est pas toujours facile à implémenter : il faudrait dans un premier temps calculer les coordonnées factorielles des individus à l'aide des fonctions discriminantes canoniques, puis calculer les distances avec des vecteurs représentant les centres de classes, qu'il faut stocker par ailleurs. Est-ce qu'il n'est pas possible de combiner ces deux actions pour extraire des fonctions de classement similaires à ceux que l'on obtient avec l'analyse discriminante linéaire ?

La réponse est oui. En développant la distance généralisée, en la multipliant par -0.5 pour basculer dans une démarche de maximisation, en supprimant tous les éléments qui ne dépendent

pas des classes, on peut expliciter des fonctions de décision linéaires dont le comportement est identique aux classifieurs de l'ADL (TUTO 1, section 2.6 ; TUTO 3, sections 3.4.7 et 3.4.8).

Voyons le détail du passage :

$$-\frac{1}{2} \times D_G^2(y_k, \omega) = -\frac{1}{2}\sum_{h=1}^{H}(z_h(\omega) - \bar{z}_{kh})^2 + \ln \hat{\pi}_k$$

$$= -\frac{1}{2}\sum_{h} z_h(\omega)^2 - \frac{1}{2}\sum_{h} \bar{z}_{kh}^2 + \sum_{h} z_h(\omega) \times \bar{z}_{kh} + \ln \hat{\pi}_k$$

Plusieurs commentaires à ce stade :

- Puisque nous devons maximiser l'expression selon (y_k), tout ce qui ne dépend pas de (k) peut être retiré.
- La fonction discriminante canonique est une fonction linéaire des variables (X_j)

$$z_h(x) = a_{h0} + a_{h1}\, x_1 + \cdots + a_{hp}\, x_p$$

- Tant que cela ne prête pas à confusion, nous retirons le symbole (ω) pour alléger les écritures.

La fonction de décision issu de l'analyse factorielle discriminante s'écrit donc :

$$S(y_k, \omega) = \ln \hat{\pi}_k + \sum_{h} a_{h0}\, \bar{z}_{kh} - \frac{1}{2}\sum_{h} \bar{z}_{kh}^2 + \sum_{j=1}^{p}\left(\sum_{h} a_{hj}\, \bar{z}_{kh}\right) x_j$$

$$S(y_k, \omega) = t_{k0} + t_{k1}\, x_1 + \cdots + t_{kp}\, x_p$$

Nous disposons d'une fonction linéaire des variables originales, applicable directement via une règle d'affectation que nous reconnaissons parfaitement :

$$y^* = \arg\max_{k}\ S(Y_k, \omega)$$

Remarque : Nous pourrions finasser en remarquant que les moyennes conditionnelles sur les facteurs ($\bar{z}_{kh}$) peuvent être obtenues par projection des moyennes conditionnelles des variables originelles dans l'espace factoriel c.-à-d.

$$\bar{z}_{kh} = a_{h0} + a_{h1}\,\bar{x}_{k1} + \cdots + a_{hp}\,\bar{x}_{kp}$$

Où $(\bar{x}_{k1}, \ldots,\ \bar{x}_{kp})$ est le vecteur moyenne des variables pour les individus de la classe (y_k). Mais introduire cette information complexifierait inutilement la formule. Nous disposons de tous les éléments à ce stade de l'analyse factorielle discriminante pour former les coefficients de ces fonctions de décision $S(y_k, \omega)$.

Exemple « WINE »

Les résultats de l'AFD sur WINE (coefficients des variables des fonctions discriminantes canoniques, section 8.1.2.6 ; moyennes conditionnelles sur les facteurs, section 8.1.3.2) nous permettent de démarrer. Nous commençons par le calcul des coefficients associés aux variables $(t_{k1}, \ldots, t_{kp})$. L'enjeu est de retracer la formule sous forme de code R.

```
#** calculs des coefficients - hors constante - des fonctions de décision

#nombre de facteurs
print(H)

## [1] 2

#coefficients des fonctions discriminantes canoniques
print(coef_)

##                     [,1]          [,2]
## Temperature -0.008566046  4.625059e-05
## Soleil      -0.006773869  5.329293e-03
## Chaleur      0.027054492 -1.276362e-01
## Pluie        0.005865665 -6.174556e-03

#moyennes conditionnelles sur les facteurs
print(zb_k)

##            zbar_1     zbar_2
## bad     2.0792470 -0.2211839
## good   -2.1219630 -0.2718120
## medium -0.1463065  0.5131035

#coefficients de la fonction de décision (t_kj, k = 1,…, K ; j = 1,…, p)
S_coef_ <- sapply(levels(y),
              #pour chaque modalité de la cible
              function(k){
                rowSums(sapply(1:H,
                  #pour chaque facteur
                  function(h){coef_[,h] * zb_k[k,h]}))
                  })

#affichage – la présentation est transposée (j : ligne, k : colonne)
print(S_coef_)
```

```
##                      bad         good       medium
## Temperature -0.01782115  0.01816426  0.001277000
## Soleil      -0.01526330  0.01292533  0.003725540
## Chaleur      0.08448403 -0.02271559 -0.069448814
## Pluie        0.01356188 -0.01076841 -0.004026372
```

Pour les constantes des fonctions de décision, nous avons besoin en plus des constantes des fonctions discriminantes canoniques, et des probabilités a priori estimées des classes.

```
#proportion des classes
print(pi_k)

## y
##       bad      good    medium
## 0.3529412 0.3235294 0.3235294

#constante des fonctions discriminantes canoniques
print(intercept_)

##           [,1]
## [1,] 32.876282
## [2,] -2.165279

#intercept de la fonction de décision
S_intercept_ <- sapply(levels(y),
                 #pour chaque modalité
                 function(k){
                  log(pi_k[k])+sum(mapply(prod,intercept_[,1],zb_k[k,]))-0.5*sum(zb_k[k,]^2)}
                  )

#affichage
print(S_intercept_)

##       bad.bad     good.good medium.medium
##     65.609287    -72.590473     -7.191833
```

Sachant que (X1 : Température, X2 : Soleil, X3 : Chaleur, X4 : Pluie), les fonctions de décisions issues de l'analyse factorielle discriminante pour les données « WINE » sont :

$$S(bad, X) = 65.6093 - 0.0178\, x_1 - 0.0153\, x_2 + 0.0845\, x_3 + 0.0136\, x_4$$

$$S(good, X) = -72.5905 + 0.0182\, x_1 + 0.0129\, x_2 - 0.0227\, x_3 - 0.0108\, x_4$$

$$S(medium, X) = -7.1918 + 0.0013\, x_1 + 0.0037\, x_2 - 0.0694\, x_3 - 0.0040\, x_4$$

Comme nous l'avions fait remarquer à plusieurs reprises, ces coefficients sont différents de ceux de l'analyse discriminante linéaire. A titre de comparaison, nous affichons ci-dessous les fonctions de classement calculés avec l'analyse discriminante linéaire prédictive (Figure 89).

Classification functions			
Attribute	medium	bad	good
Temperature	0.8008	0.7817	0.8177
Soleil	0.1449	0.1259	0.1541
Chaleur	-7.0565	-6.9026	-7.0098
Pluie	-0.0395	-0.0220	-0.0463
constant	-1285.4501	-1212.6490	-1350.8487

Figure 89 – Fonctions de classement de l'ADL – Données "WINE"

Mais les écarts entre les coefficients des variables sont les mêmes d'une classe à l'autre (TUTO 1, section 3.4.7). Ainsi, en déploiement, les probabilités d'affectation, et par conséquent les prédictions, sont exactement les mêmes. Les deux systèmes de classement sont interchangeables (TUTO 1, section 4.4).

Vérifions cela sur les données « WINE ». Nous calculons les probabilités d'affectation des individus avec les coefficients des fonctions de décision déduites de l'analyse factorielle discriminante, et nous les comparons avec celles de la fonction lda() de MASS

```
#** calcul des probabilités d'appartenance des individus **
#avec les fonctions de décision explicites déduites de l'AFD

#appliquer les coefficients pour une première version des scores
scores <- as.matrix(X) %*% S_coef_

#rajouter la constante
scores <- t(apply(scores,1,function(v){v+S_intercept_}))
print(head(scores))

##             bad      good    medium
## [1,] -1.5852759 -5.526406 -0.952740
## [2,]  1.5868073 -8.376910 -1.562690
## [3,] -0.9747601 -5.301410 -1.843753
## [4,]  2.4969573 -9.135866 -1.796625
## [5,] -4.3929741 -1.371053 -2.045149
## [6,] -7.7589956  1.448739 -1.192918

#transformation softmax pour les probas
probas <- t(apply(scores,1,function(v){exp(v)/sum(exp(v))}))
print(head(probas))

##               bad         good     medium
## [1,] 3.446132e-01 6.694541e-03 0.64869229
## [2,] 9.588456e-01 4.513996e-05 0.04110923
## [3,] 6.980386e-01 9.222298e-03 0.29273906
## [4,] 9.865194e-01 8.750537e-06 0.01347182
## [5,] 3.125637e-02 6.417150e-01 0.32702867
## [6,] 9.358431e-05 9.334075e-01 0.06649889
```

```r
#probas fournies par lda() de MASS
library(MASS)
probas_MASS <- predict(lda(X,y),X)$posterior
print(head(probas_MASS))
```

```
##               bad         good     medium
## [1,] 3.446132e-01 6.694541e-03 0.64869229
## [2,] 9.588456e-01 4.513996e-05 0.04110923
## [3,] 6.980386e-01 9.222298e-03 0.29273906
## [4,] 9.865194e-01 8.750537e-06 0.01347182
## [5,] 3.125637e-02 6.417150e-01 0.32702867
## [6,] 9.358431e-05 9.334075e-01 0.06649889
```

Les valeurs sont parfaitement concordantes.

9 Analyse des correspondances discriminante

L'analyse des correspondances discriminante (ACD) est le pendant de l'analyse factorielle discriminante pour les descripteurs catégoriels. Etonnamment, on la rencontre peu dans la littérature. En regardant de plus près, on la reconnaît sous les traits de l'analyse discriminante barycentrique (Nackache et Confais, 2003, chapitre 3 ; Celeux et Nakache, 1994, section 5.3.3 ; Lebart et al., 2000, section 3.3.7.c) que nous avons succinctement exposé plus haut (section 6.4). Lorsque (K > 2), l'approche passe par un tableau de contingence particulier soumis à une analyse factorielle des correspondances (AFC). Mais honnêtement, mis à part des alignements de formules matricielles, je n'ai pas trouvé une présentation avec un exemple didactique suffisamment étayé pour que l'on puisse comprendre et reproduire les étapes qui la composent.

Je l'ai reconnue sous un nom différent dans une autre référence (Abdi H., « Discriminant Correspondance Analysis », in N.J. Salkind (Ed.), Encyclopedia of Measurements and Statistics, Sage, pages 270 à 275, 2007) qui a attiré mon attention (ça tient à peu de choses…). Je me suis dit qu'il y avait finalement matière à monter un support de cours intéressant sur cette thématique (Rakotomalala R., « Analyse des correspondances discriminante – Diapos », juillet 2013). J'en reprends les grandes lignes avec pour trame : le mode de construction des facteurs permettant de distinguer au mieux les groupes ; leur interprétation à partir des descripteurs ; leur expression sous la forme d'une fonction discriminante canonique explicite qui permet de projeter les individus supplémentaires dans l'espace factoriel ; et enfin, l'affectation des classes basée sur la distance aux barycentres conditionnels.

Concernant les logiciels, TANAGRA est un des rares à proposer nativement l'analyse des correspondances discriminante. Mais bon, avec un peu de programmation on y arrive très dans d'autres outils, comme R par exemple dans ce chapitre. Le lecteur pourra faire le parallèle entre les résultats intermédiaires que nous décrivons et les sorties de TANAGRA sur les mêmes données (TUTO 12).

9.1 Principe de la méthode

Une solution « évidente » pourrait être le remplacement des descripteurs qualitatifs par leurs indicatrices, à l'instar de ce que nous pouvons faire dans une analyse discriminante linéaire (section 3.2), puis utiliser le programme usuel d'analyse factorielle discriminante. Mais si elle est viable dans une démarche prédictive où prime avant tout la performance en classement, elle est plus discutable dans une démarche descriptive où l'interprétation tient une place centrale. Comme lire la corrélation entre deux indicatrices issues de la même variable ? A l'évidence, il faut développer un paradigme spécifique pour appréhender le cas des descripteurs catégoriels en analyse discriminante descriptive.

9.1.1 Données « DIVAY »

Nous inaugurons un nouveau jeu de données dans ce chapitre. La base « DIVAY » décrit les caractéristiques de vins provenant de (K = 3) régions {Loire, Rhône, Beaujolais} (sacré René…).

Region	Woody	Fruity	Sweet	Alcohol	Hedonic
Loire	A	C	B	A	A
Loire	B	C	C	B	C
Loire	A	B	B	A	B
Loire	A	C	C	B	D
Rhone	A	B	A	C	C
Rhone	B	A	A	C	B
Rhone	C	B	B	B	A
Rhone	B	C	C	C	D
Beaujolais	C	A	C	A	A
Beaujolais	B	A	C	A	B
Beaujolais	C	B	B	B	D
Beaujolais	C	A	A	A	C

Figure 90 – Données "DIVAY" (Abdi, 2007)

Nous disposons de (n = 12) observations et (p = 5) descripteurs, tous qualitatifs. L'objectif est de produire un système de description composé de facteurs (z_h) qui permet de distinguer au mieux les régions, et d'expliquer la nature de ces facteurs à partir des variables explicatives. Comme en analyse factorielle discriminante, l'idée est d'aboutir à une représentation graphique intuitive où les positions relatives des centres de classe jouent un rôle primordial (Figure 91).

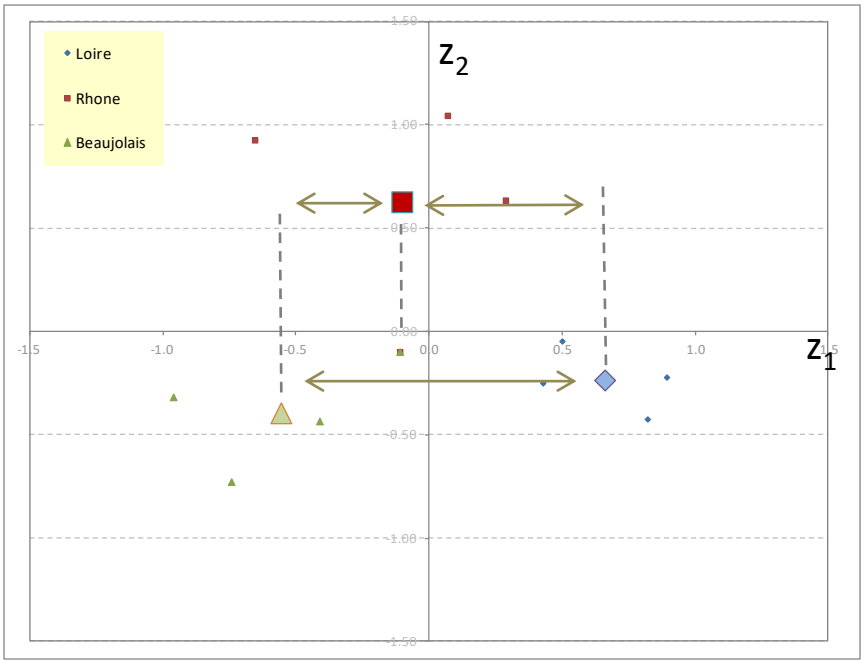


Figure 91 – Position des régions dans le repère factoriel – Données "DIVAY"

9.1.2 Tableaux, profils, distances, inertie

Tableau des indicatrices. Une première transformation consiste à substituer aux données initiales (n, p) une matrice des indicatrices (n, L) où L est le nombre total de modalités observées sur l'ensemble des variables (Figure 92). Nous disposons d'un nuage de (n) points en dimension (L).

Region	Woody_A	Woody_B	Woody_C	Fruity_A	Fruity_B	Fruity_C	Sweet_A	Sweet_B	Sweet_C	Alcohol_A	Alcohol_B	Alcohol_C	Hedonic_A	Hedonic_B	Hedonic_C	Hedonic_D
Loire	1	0	0	0	0	1	0	1	0	1	0	0	1	0	0	0
Loire	0	1	0	0	0	1	0	0	1	0	1	0	0	0	1	0
Loire	1	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0
Loire	1	0	0	0	0	1	0	0	1	0	1	0	0	0	0	1
Rhone	1	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0
Rhone	0	1	0	1	0	0	1	0	0	0	0	1	0	1	0	0
Rhone	0	0	1	0	1	0	0	1	0	0	1	0	1	0	0	0
Rhone	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	1
Beaujolais	0	0	1	1	0	0	0	0	1	1	0	0	1	0	0	0
Beaujolais	0	1	0	1	0	0	0	0	1	1	0	0	0	1	0	0
Beaujolais	0	0	1	0	1	0	0	1	0	0	1	0	0	0	0	1
Beaujolais	0	0	1	1	0	0	1	0	0	1	0	0	0	0	1	0

Figure 92 – Matrice des indicatrices – Données "DIVAY"

Dans notre exemple, « Woody » possède (3 modalités), « Fruity » (3), « Sweet » (3), « Alcohol » (3) et Hedonic (4). Nous avons :

$$L = (3 + 3 + 3 + 3 + 4) = 16$$

Quand il n'y a pas d'ambiguïtés, chaque indicatrice sera notée (x_l) dans ce chapitre $(l = 1, \dots, 16)$.

Tableau de contingence. De cette représentation, nous pouvons déduire un tableau de contingence particulier formé par le croisement de chaque indicatrice avec la variable cible Y (région) (Figure 93).

Region	Woody_A	Woody_B	Woody_C	Fruity_A	Fruity_B	Fruity_C	Sweet_A	Sweet_B	Sweet_C	Alcohol_A	Alcohol_B	Alcohol_C	Hedonic_A	Hedonic_B	Hedonic_C	Hedonic_D	**Total**
Loire	3	1	0	0	1	3	0	2	2	2	2	0	1	1	1	1	20
Rhone	1	2	1	1	2	1	2	1	1	0	1	3	1	1	1	1	20
Beaujolais	0	1	3	3	1	0	1	1	2	3	1	0	1	1	1	1	20
Total	4	4	4	4	4	4	3	4	5	5	4	3	3	3	3	3	60

Figure 93 – Tableau de contingence M pour l'analyse des correspondances discriminante

Nous noterons M cette matrice, de termes (m_{kl}) où :

- La marge totale $m = n \times p = 12 \times 5 = 60$
- La marge ligne $m_k = \sum_l m_{kl} = n_k \times p$ représente la distribution observée des classes multipliée par le nombre de variables initiales.
- La marge colonne $m_l = \sum_k m_{kl}$ représente la fréquence absolue de l'indicatrice (x_l).

Profils. Ce tableau M est surtout important parce qu'il va nous permettre de définir différents profils qui vont jouer le rôle de barycentres :

- Le profil global est un vecteur G composé par $(\frac{m_l}{m})$
- Les profils conditionnels G_k sont composés par $(\frac{m_{kl}}{m_k})$

Region	Woody_A	Woody_B	Woody_C	Fruity_A	Fruity_B	Fruity_C	Sweet_A	Sweet_B	Sweet_C	Alcohol_A	Alcohol_B	Alcohol_C	Hedonic_A	Hedonic_B	Hedonic_C	Hedonic_D	Total
Loire	0.150	0.050	0.000	0.000	0.050	0.150	0.000	0.100	0.100	0.100	0.100	0.000	0.050	0.050	0.050	0.050	1
Rhone	0.050	0.100	0.050	0.050	0.100	0.050	0.100	0.050	0.050	0.000	0.050	0.150	0.050	0.050	0.050	0.050	1
Beaujolais	0.000	0.050	0.150	0.150	0.050	0.000	0.050	0.050	0.100	0.150	0.050	0.000	0.050	0.050	0.050	0.050	1
Total	0.067	0.067	0.067	0.067	0.067	0.067	0.050	0.067	0.083	0.083	0.067	0.050	0.050	0.050	0.050	0.050	1

Figure 94 – Tableau des profils – Données "DIVAY"

Distance entre profils. L'écart entre les profils est quantifié par la distance du χ^2 c.-à-d. pondérée par l'inverse de la fréquence marginale des modalités. Elle a pour particularité, à écarts identiques, d'exacerber ceux observés sur les modalités peu fréquentes.

$$d^2(k_1, k_2) = \sum_l \frac{1}{\frac{m_l}{m}} \left(\frac{m_{k_1 l}}{m_{k_1}} - \frac{m_{k_2 l}}{m_{k_2}} \right)^2$$

Pour la distance du groupe « Loire » au barycentre global, nous avons :

$$d^2(Loire, G) = \frac{1}{0.067}(0.150 - 0.067)^2 + \frac{1}{0.067}(0.050 - 0.067)^2 + \cdots + \frac{1}{0.050}(0.050 - 0.050)^2 = 0.490$$

Pour la distance entre les groupes « Loire » et « Rhône » :

$$d^2(Loire, Rhone) = \frac{1}{0.067}(0.150 - 0.050)^2 + \frac{1}{0.067}(0.050 - 0.100)^2 + \cdots + \frac{1}{0.050}(0.050 - 0.050)^2 = 1.325$$

Nous pouvons ainsi construire la matrice des distances par paires de classes.

d^2	Beaujolais	Loire	Rhône
Beaujolais	0	1.505	1.250
Loire	1.505	0	1.325
Rhône	1.250	1.325	0

L'objectif de l'analyse des correspondances discriminante est de construire un espace de représentation de dimension réduite qui permet de respecter au mieux les distances entre les centres de groupes.

Inertie. L'inertie est la généralisation de la variance. Elle quantifie la dispersion autour du barycentre globale. Elle traduit surtout la quantité d'information portée par les données.

$$\phi^2 = \sum_{k=1}^{K} \frac{n_k}{n} \, d^2(G_k, G)$$

Une autre manière de voir l'analyse des correspondances discriminante est de construire une succession de facteurs orthogonaux qui vont décomposer cette inertie totale.

Exemple « DIVAY »

Essayons de calculer sous R ces différents éléments pour la base « DIVAY ». Nous commençons par charger les données et préparer les structures.

```
#lecture des données d'apprentissage
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DIVAY")

#affichage des caractéristiques des variables
print(summary(D))

##         Region  Woody Fruity Sweet Alcohol Hedonic
##  Beaujolais:4   A:4   A:4    A:3   A:5     A:3
##  Loire     :4   B:4   B:4    B:4   B:4     B:3
##  Rhone     :4   C:4   C:4    C:5   C:3     C:3
##                                             D:3

#nombre de variables epxlicatives
p <- ncol(D)-1

#nombre de classes
K <- nlevels(D$Region)

#nombre d'observations
n <- nrow(D)

#effectifs par classe
n_k <- table(D$Region)
print(n_k)

##
## Beaujolais      Loire      Rhone
##          4          4          4
```

Nous créons le tableau des indicatrices de la Figure 92.

```
#fonction pour créer des dummies
#à partir d'une variable de type factor
set_dummies <- function(x){
  #nombre de modalités
  L <- nlevels(x)
  #liste des modalités
  modalites <- levels(x)
  #création des variables 0/1 - la première modalité sert de référence
  M <- sapply(1:L,function(j){return(as.double(x==modalites[j]))})
  #nommer les variables avec les modalités
  colnames(M) <- modalites[1:L]
  #renvoyer sous la forme de data frame
  return(as.data.frame(M))
}

#construction des indicatrices
IND <- data.frame(lapply(D[2:ncol(D)],set_dummies))
print(IND)

##    Woody.A Woody.B Woody.C Fruity.A Fruity.B Fruity.C Sweet.A Sweet.B Sweet.C
## 1        1       0       0        0        0        1       0       1       0
## 2        0       1       0        0        0        1       0       0       1
## 3        1       0       0        0        1        0       0       1       0
## 4        1       0       0        0        0        1       0       0       1
## 5        1       0       0        0        1        0       1       0       0
## 6        0       1       0        1        0        0       1       0       0
## 7        0       0       1        0        1        0       0       1       0
## 8        0       1       0        0        0        1       0       0       1
## 9        0       0       1        1        0        0       0       0       1
## 10       0       1       0        1        0        0       0       0       1
## 11       0       0       1        0        1        0       0       1       0
## 12       0       0       1        1        0        0       1       0       0
##    Alcohol.A Alcohol.B Alcohol.C Hedonic.A Hedonic.B Hedonic.C Hedonic.D
## 1          1         0         0         1         0         0         0
## 2          0         1         0         0         0         1         0
## 3          1         0         0         0         1         0         0
## 4          0         1         0         0         0         0         1
## 5          0         0         1         0         0         1         0
## 6          0         0         1         0         1         0         0
## 7          0         1         0         1         0         0         0
## 8          0         0         1         0         0         0         1
## 9          1         0         0         1         0         0         0
## 10         1         0         0         0         1         0         0
## 11         0         1         0         0         0         0         1
## 12         1         0         0         0         0         1         0
```

Puis la matrice M de la Figure 93.

```
#formation de la matrice M
M <- as.matrix(aggregate(IND,by=list(D$Region),sum)[2:(ncol(IND)+1)])
rownames(M) <- levels(D$Region)
print(M)

##            Woody.A Woody.B Woody.C Fruity.A Fruity.B Fruity.C Sweet.A Sweet.B
## Beaujolais       0       1       3        3        1        0       1       1
## Loire            3       1       0        0        1        3       0       2
```

```
## Rhone            1       2       1        1        2        1       2       1
##            Sweet.C Alcohol.A Alcohol.B Alcohol.C Hedonic.A Hedonic.B Hedonic.C
## Beaujolais       2         3         1         0         1         1         1
## Loire            2         2         2         0         1         1         1
## Rhone            1         0         1         3         1         1         1
##            Hedonic.D
## Beaujolais         1
## Loire              1
## Rhone              1
```

Nous en déduisons les profils conditionnels et globaux (Figure 94).

```
#matrice des profils
profils <- t(apply(M,1,function(v){v/sum(v)}))
print(profils)

##            Woody.A Woody.B Woody.C Fruity.A Fruity.B Fruity.C Sweet.A Sweet.B
## Beaujolais    0.00    0.05    0.15     0.15     0.05     0.00    0.05    0.05
## Loire         0.15    0.05    0.00     0.00     0.05     0.15    0.00    0.10
## Rhone         0.05    0.10    0.05     0.05     0.10     0.05    0.10    0.05
##            Sweet.C Alcohol.A Alcohol.B Alcohol.C Hedonic.A Hedonic.B Hedonic.C
## Beaujolais    0.10      0.15      0.05      0.00      0.05      0.05      0.05
## Loire         0.10      0.10      0.10      0.00      0.05      0.05      0.05
## Rhone         0.05      0.00      0.05      0.15      0.05      0.05      0.05
##            Hedonic.D
## Beaujolais      0.05
## Loire           0.05
## Rhone           0.05

#profil marginal
G <- colSums(M)/(n*p)
print(G)

##    Woody.A    Woody.B    Woody.C   Fruity.A   Fruity.B   Fruity.C    Sweet.A
## 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667 0.06666667 0.05000000
##    Sweet.B    Sweet.C  Alcohol.A  Alcohol.B  Alcohol.C  Hedonic.A  Hedonic.B
## 0.06666667 0.08333333 0.08333333 0.06666667 0.05000000 0.05000000 0.05000000
##  Hedonic.C  Hedonic.D
## 0.05000000 0.05000000
```

Avec une fonction ad hoc, nous calculons les exemples de distances ci-dessus (Loire vs. Global)

et (Loire vs. Rhône). Nous produisons alors la matrice des distances entre classes.

```
#fonction pour distance entre profils
DKHI2 <- function(p1,p2,ponderation){
  d <- sum((1/ponderation)*(p1-p2)^2)
  return(d)
}

#ex. distance au profil global de Loire
print(DKHI2(profils['Loire',],G,G))

## [1] 0.49

#ex. distance entre Loire et Rhone
print(DKHI2(profils['Loire',],profils['Rhone',],G))

## [1] 1.325
```

```r
#** distance par paires des classes
DG <- matrix(0,K,K)
rownames(DG) <- levels(D$Region)
colnames(DG) <- levels(D$Region)

#double boucle -- allons au plus simple, beaucoup de calculs superflus
for (k1 in levels(D$Region)){
  for (k2 in levels(D$Region)){
    DG[k1,k2] <- DKHI2(profils[k1,],profils[k2,],G)
  }
}

#affichage - matrice de distance entre centres de classes DG
print(DG)

##            Beaujolais Loire Rhone
## Beaujolais      0.000 1.505 1.250
## Loire           1.505 0.000 1.325
## Rhone           1.250 1.325 0.000
```

Reste enfin à calculer l'inertie totale qu'il faudra décomposer avec l'ACD.

```r
#inertie totale - ϕ²
IT <- sum(sapply(1:K,function(k){n_k[k]/n * DKHI2(profils[k,],G,G)}))
print(IT)

## [1] 0.4533333
```

Gardons bien en tête ces deux derniers résultats. Nous y reviendrons tout au long de la présentation de l'analyse des correspondances discriminante.

9.2 Réaliser une ACD via une AFC

9.2.1 Rapport de corrélation

L'idée du rapport de corrélation de l'analyse factorielle discriminante reste valable dans ce nouveau cadre. Pour un facteur (z), qui est une variable latente définie sur les indicatrices, l'objectif de la méthode est de maximiser $\eta^2_{z,y} = \frac{SCE}{SCT}$ (section 8.1.2.1), qui traduit l'intensité de l'écart entre les centres de classes. Si la discrimination n'est pas parfaite sur le premier facteur c.-à-d. la technique ne restitue pas parfaitement ces écarts, la partie résiduelle, non expliquée, sera traitée sur le 2nd, etc.

9.2.2 Formulation de l'ACD en termes d'AFC

Pour répondre à ce cahier des charges, l'analyse des correspondances discriminante se tourne vers l'analyse factorielle des correspondances (AFC) du tableau M (Figure 93). Rappelons que l'AFC s'applique sur tout tableau croisé – tableau de contingence le plus souvent – pourvu que les marges, et donc les profils lignes et colonnes, aient un sens (Rakotomalala R., « Analyse Factorielle des Correspondances – Diapos », juillet 2013). Elle cherche un système de représentation qui maximise l'étalement des points lignes et colonnes. C'est exactement ce que nous cherchons à faire pour notre tableau M où les lignes correspondent aux barycentres des classes. Les coefficients de la variable latente (z_h) est définie par la maximisation de la dispersion inter-classe :

$$\lambda_h = \sum_{k=1}^{K} \frac{n_k}{n} \times \bar{z}_{kh}^2$$

Deux commentaires :

- Par définition, la moyenne pondérée des points modalités est nulle ($\bar{z}_h = 0$), d'où l'expression simplifiée ci-dessus.
- L'analyse est symétrique avec l'AFC, elle effectue simultanément la même optimisation pour les points colonnes (les indicatrices), ce qui introduit une contrainte supplémentaire dans les calculs.
- Il y a une relation entre le carré du rapport de corrélation (η^2) et (λ) pour un facteur (h) :

$$\eta_h^2 = \frac{\lambda_h}{\frac{1}{n} SCT_h} \text{ , où } SCT_h = \sum_{\omega} z_h(\omega)^2$$

Nous disposons de suffisamment d'éléments pour caractériser la démarche ACD.

9.2.3 Caractérisation de l'ACD

L'analyse des correspondances discriminante vise à :

- Trouver les facteurs (z_h), des variables latentes qui sont des combinaisons linéaires des indicatrices, qui maximisent les écarts entre les centres de classes représentés par les lignes du tableau M.
- Le nombre maximal de facteurs est égal à H = min (K -1, L -1). Souvent (K -1) puisque le nombre de classes est plus faible que le nombre d'indicatrices dans la majorité des bases traitées.
- Les facteurs (z_h) sont deux à deux orthogonaux.
- Le facteur n°h explique l'écartement entre les centres de classes qui n'ont pas été pris en compte dans les (h-1) précédents facteurs.
- Les calculs basés sur l'AFC assure la décroissance de la variance expliquée (λ_h) sur les axes, pas forcément celle du rapport de corrélation. Ce dernier est post-calculé après l'analyse pour être affiché dans les logiciels.
- Le pouvoir de représentation d'un facteur est traduit par la part d'inertie reproduite.

Lorsque nous considérons l'ensemble des H facteurs, la représentation doit permettre de retrouver, avec la distance euclidienne, les écarts entre centres de classes et l'inertie totale comptabilisées dans l'espace initial avec la distance du χ^2.

9.3 Pratique de l'analyse des correspondances discriminante

9.3.1 Résoudre un problème d'AFC

9.3.1.1 Résidus standardisés

Il y a plusieurs manières de résoudre un problème d'analyse factorielle des correspondances. La plus simple – et la plus révélatrice de la nature duale de l'analyse – est de passer par une décomposition en valeurs singulières de la matrice des résidus standardisés, fruit de l'écart entre le tableau observé et le tableau sous l'hypothèse d'indépendance (Rakotomalala R., « Etude des dépendances – Variables qualitatives », version 2.1, avril 2020 ; section 2.3.2).

Le tableau sous indépendance E est obtenu par le produit de ses marges :

$$e_{kl} = \frac{m_k \times m_l}{m}$$

Le tableau des résidus standardisés S est défini par :

$$s_{kl} = \frac{m_{kl} - e_{kl}}{\sqrt{e_{kl}}}$$

La matrice R qui sera présentée à la décomposition en valeurs singulière sera alors :

$$R = \frac{1}{\sqrt{m}} S$$

Exemple « DIVAY »

Les calculs sont très simples à mener sous R. Voici successivement les matrices E, S et R :

```
#effectif total dans M
m <- sum(M)

#marges lignes et colonnes de M
sLig <- rowSums(M)
sCol <- colSums(M)

#tableau théorique (sous indépendance) : E
mL <- matrix(sLig,nrow=length(sLig),ncol=1)
mC <- matrix(sCol,nrow=1,ncol=length(sCol))
E <- (mL %*% mC)/m
print(E)

##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6] [,7]     [,8]
## [1,] 1.333333 1.333333 1.333333 1.333333 1.333333 1.333333    1 1.333333
## [2,] 1.333333 1.333333 1.333333 1.333333 1.333333 1.333333    1 1.333333
## [3,] 1.333333 1.333333 1.333333 1.333333 1.333333 1.333333    1 1.333333
##          [,9]    [,10]    [,11] [,12] [,13] [,14] [,15] [,16]
## [1,] 1.666667 1.666667 1.333333     1     1     1     1     1
## [2,] 1.666667 1.666667 1.333333     1     1     1     1     1
## [3,] 1.666667 1.666667 1.333333     1     1     1     1     1

#résidu standardisé : S
S <- (M-E)/sqrt(E)
print(S)

##               Woody.A    Woody.B    Woody.C   Fruity.A   Fruity.B   Fruity.C
## Beaujolais -1.1547005 -0.2886751  1.4433757  1.4433757 -0.2886751 -1.1547005
## Loire       1.4433757 -0.2886751 -1.1547005 -1.1547005 -0.2886751  1.4433757
## Rhone      -0.2886751  0.5773503 -0.2886751 -0.2886751  0.5773503 -0.2886751
##            Sweet.A    Sweet.B    Sweet.C  Alcohol.A  Alcohol.B Alcohol.C
## Beaujolais       0 -0.2886751  0.2581989  1.0327956 -0.2886751        -1
## Loire           -1  0.5773503  0.2581989  0.2581989  0.5773503        -1
## Rhone            1 -0.2886751 -0.5163978 -1.2909944 -0.2886751         2
##            Hedonic.A Hedonic.B Hedonic.C Hedonic.D
## Beaujolais         0         0         0         0
```

```
## Loire              0         0         0         0
## Rhone              0         0         0         0

#tableau de travail pour la décomposition : R
R <- 1/sqrt(m) * S
print(R)

##               Woody.A    Woody.B    Woody.C   Fruity.A   Fruity.B   Fruity.C
## Beaujolais -0.1490712 -0.0372678  0.1863390  0.1863390 -0.0372678 -0.1490712
## Loire       0.1863390 -0.0372678 -0.1490712 -0.1490712 -0.0372678  0.1863390
## Rhone      -0.0372678  0.0745356 -0.0372678 -0.0372678  0.0745356 -0.0372678
##               Sweet.A    Sweet.B     Sweet.C   Alcohol.A  Alcohol.B  Alcohol.C
## Beaujolais  0.0000000 -0.0372678  0.03333333  0.13333333 -0.0372678 -0.1290994
## Loire      -0.1290994  0.0745356  0.03333333  0.03333333  0.0745356 -0.1290994
## Rhone       0.1290994 -0.0372678 -0.06666667 -0.16666667 -0.0372678  0.2581989
##            Hedonic.A Hedonic.B Hedonic.C Hedonic.D
## Beaujolais         0         0         0         0
## Loire              0         0         0         0
## Rhone              0         0         0         0
```

9.3.1.2 Décomposition en valeurs singulières

La matrice R est décomposée en :

$$R = U\,\Delta\,\mathrm{V}^{\mathrm{T}}$$

Où :

- $\Delta\ (\delta_h)$ est une matrice diagonale composée des valeurs singulières, qui sont en relation directe avec les valeurs propres, les inerties, portées par les axes factoriels :

$$\lambda_h = \delta_h^2$$

  Le nombre de valeurs singulières non-nulles est égal à $\mathrm{H} = \min(\mathrm{K} - 1, \mathrm{L} - 1)$

- U est une matrice des vecteurs singuliers à gauche, orthogonale, elle fournira les coordonnées des points modalités-lignes, les classes, dans le repère factoriel.
- V est une matrice des vecteurs singuliers à droite, orthogonale également, elle fournira les coordonnées des points modalités-colonnes, les modalités des descripteurs.

Exemple « DIVAY »

La fonction svd() de R convient parfaitement pour ce que nous voulons faire.

```r
#décomposition en valeurs singulières
sol <- svd(R)
print(sol)
```

```
## $d
## [1] 5.018844e-01 4.488267e-01 7.580364e-17
##
## $u
##            [,1]       [,2]      [,3]
## [1,] -0.6406472 -0.5061994 0.5773503
## [2,]  0.7587051 -0.3017171 0.5773503
## [3,] -0.1180579  0.8079165 0.5773503
##
## $v
##              [,1]        [,2]         [,3]
##  [1,]  0.48074453 -0.02422131 -0.542257890
##  [2,] -0.02629944  0.20125318 -0.530291962
##  [3,] -0.45444509 -0.17703187  0.067433106
##  [4,] -0.45444509 -0.17703187  0.067433106
##  [5,] -0.02629944  0.20125318  0.197705736
##  [6,]  0.48074453 -0.02422131  0.362447019
##  [7,] -0.22552930  0.31917233 -0.160585530
##  [8,]  0.16901466 -0.07515816  0.231078213
##  [9,]  0.02352294 -0.18000632 -0.051762096
## [10,] -0.08060250 -0.47279545 -0.331437402
## [11,]  0.16901466 -0.07515816  0.231078213
## [12,] -0.09110394  0.69716147  0.009214631
## [13,]  0.00000000  0.00000000  0.000000000
## [14,]  0.00000000  0.00000000  0.000000000
## [15,]  0.00000000  0.00000000  0.000000000
## [16,]  0.00000000  0.00000000  0.000000000
```

Nous notons d'emblée que seuls H = min(3 −1, 16 − 1) = 2 valeurs singulières sont non-nulles. L'enjeu pour nous est de pouvoir exploiter à bon escient ces différentes informations ($d, $u, $v) pour la suite.

9.3.2 Qualité de représentation sur les facteurs

Les valeurs singulières (δ_1 = 0.5019, δ_2 = 0.4488) passées au carré (λ_1 = 0.2519, λ_2 = 0.2014) représentent l'information portée par le facteur. Nous remarquons que la somme des λ_h sur l'ensemble des facteurs disponibles ($\lambda_1 + \lambda_2 = 0.2519 + 0.2014 = 0.4533 = \phi^2$) correspond à l'inertie totale. Toute l'information disponible a été restituée.

Les (λ_h) apparaissent sous l'appellation « valeur propre » (eigen value) dans les logiciels. Elles peuvent être exprimées en pourcentages ou en pourcentages cumulés. Nous pourrions en dériver les rapports de corrélation sur les facteurs (canonical correlation) mais il faudrait pour cela disposer des SCT_h. Ce n'est pas le cas à ce stade.

Exemple « DIVAY »

Nous mettons le focus sur le champ ($d).

```
#nombre de facteurs
H <- 2

#inertie portée par les facteurs
lambda <- sol$d[1:H]^2
print(lambda)

## [1] 0.2518879 0.2014454

#inertie totale - ϕ²
print(sum(lambda))

## [1] 0.4533333

#en proportion
print(lambda/sum(lambda))

## [1] 0.5556351 0.4443649

#en proportion cumulée
print(cumsum(lambda)/sum(lambda))

## [1] 0.5556351 1.0000000
```

9.3.3 Coordonnées des classes

9.3.3.1 Calcul et représentation graphique

Les coordonnées des points-classes s'obtiennent avec la matrice U :

$$F_{kh} = \frac{u_{kh} \times \delta_h}{\sqrt{\frac{m_k}{m}}} = \frac{u_{kh} \times \delta_h}{\sqrt{\frac{n_k}{n}}}$$

Puisque ($m = n \times p$, et $m_k = n_k \times p$).

Exemple « DIVAY »

Nous exploitons le champ ($u) et, puisque nous sommes dans le plan, nous pouvons passer à la représentation graphique. Nous en profiterons pour procéder à quelques vérifications.

```
#coordonnées des points-classes
zc <- sapply(1:H,function(h){sol$u[,h]*sol$d[h]/sqrt(n_k/n)})
print(zc)
```

```
##                   [,1]       [,2]
## Beaujolais -0.5569077 -0.3935147
## Loire       0.6595342 -0.2345520
## Rhone      -0.1026265  0.6280667

#représentation graphique
plot(zc[,1],zc[,2],xlim=c(-1,1),ylim=c(-1.5,1.5),type="n")
text(zc[,1],zc[,2],labels=levels(D$Region),col=c("red","green","blue"))
abline(h=0,v=0,col="gray")
```

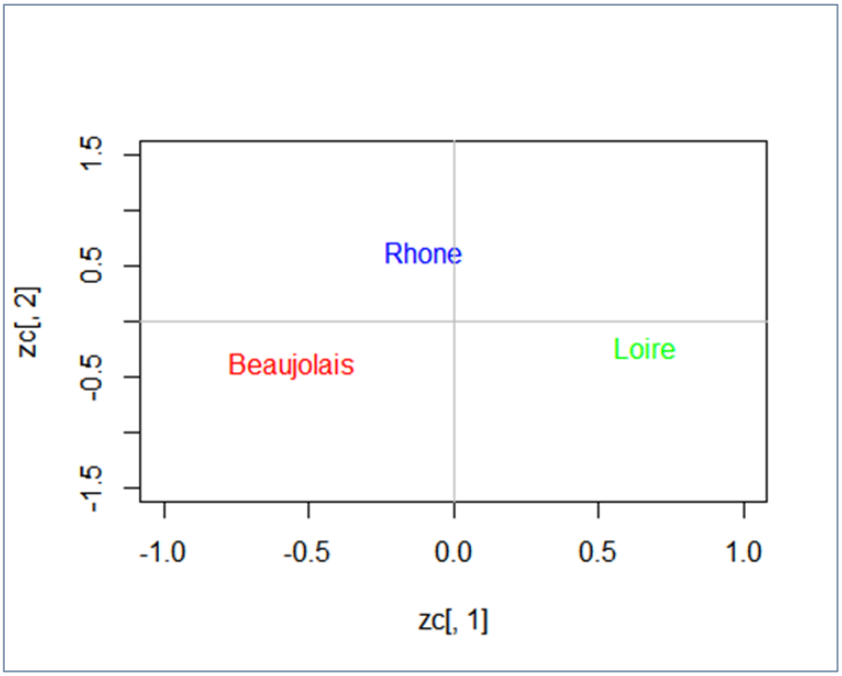


Figure 95 – ACD – Position des classes dans le plan factoriel – Données "DIVAY"

Visiblement, « Loire » et « Beaujolais » s'opposent sur le premier facteur. « Rhône » se démarque des deux autres sur le second. Pourquoi ? Nous le saurons en analysant les coordonnées des points-modalités dans la section suivante.

Revenons sur quelques résultats fondamentaux de l'analyse des correspondances discriminante. La distance euclidienne entre les classes dans le repère factoriel devrait correspondre à la distance du χ^2 dans le repère initial disions-nous plus haut (section 9.2.3). Vérifions cela.

```
#matrice des distances euclidiennes
#dans le repère factoriel
DE <- matrix(0,K,K)
rownames(DE) <- levels(D$Region)
colnames(DE) <- levels(D$Region)

#double boucle -- allons au plus simple, beaucoup de calculs superflus
for (k1 in 1:K){
  for (k2 in 1:K){
    DE[k1,k2] <- sum((zc[k1,]-zc[k2,])^2)
  }
```

```
}

#affichage - matrice des distances euclidiennes entre les classes
print(DE)

##            Beaujolais Loire Rhone
## Beaujolais      0.000 1.505 1.250
## Loire           1.505 0.000 1.325
## Rhone           1.250 1.325 0.000
```

Oui, on ne nous a pas raconté des salades, DE correspond exactement à la matrice DG (page 231) calculée via la distance du χ^2.

Autre assertion à vérifier. La dispersion des classes sur un facteur correspondrait à son inertie (λ_h) (section 9.2.2). Nous exploitons les coordonnées « zc » ci-dessus.

```
#inertie sur les facteurs
apply(zc,2,function(v){sum(n_k/n*v^2)})

## [1] 0.2518879 0.2014454
```

Ce sont bien les valeurs de (λ_1, λ_2). Tout va pour le mieux.

9.3.3.2 Qualité de la représentation des classes

Pour matérialiser la qualité de représentation d'une classe, nous utilisons habituellement le COS2 qui représente l'angle entre le point et l'axe. Son intérêt est que l'indicateur s'additionne d'un facteur à l'autre. Il s'obtient par le carré des coordonnées normalisées.

Exemple « DIVAY »

Il suffit de passer les coordonnées au carré et de diviser par la somme en ligne.

```
#qualité de représentation des classes - COS2
print(zc^2/rowSums(zc^2))

##                  [,1]      [,2]
## Beaujolais 0.66698104 0.3330190
## Loire      0.88772520 0.1122748
## Rhone      0.02600542 0.9739946
```

Le graphique (Figure 95) ne laissait aucun doute, mais c'est toujours mieux quand les chiffres confirment : les informations portées par « Loire » et « Beaujolais » sont bien captées par le premier facteur, « Rhône » est mieux situé sur le second. Et la somme en ligne dans le tableau des COS2 fait bien 100%.

9.3.4 Coordonnées des indicatrices

9.3.4.1 Calcul et représentation graphique

Les coordonnées des points modalités sur le facteur (h) s'obtient avec :

$$O_{lh} = \frac{v_{lh} \times \delta_h}{\sqrt{\frac{m_l}{m}}}$$

Exemple « DIVAY »

Nous exploitons le champ ($v) de la décomposition en valeurs singulières.

```
#somme en colonne de M
m_l <- colSums(M)
print(m_l)

##   Woody.A   Woody.B   Woody.C  Fruity.A  Fruity.B  Fruity.C   Sweet.A   Sweet.B
##         4         4         4         4         4         4         3         4
##   Sweet.C Alcohol.A Alcohol.B Alcohol.C Hedonic.A Hedonic.B Hedonic.C Hedonic.D
##         5         5         4         3         3         3         3         3

#somme totale de M
print(m)

## [1] 60

#coordonnées des points modalités
zl <- sapply(1:H,function(h){sol$v[,h]*sol$d[h]/sqrt(m_l/m)})
print(zl)

##                  [,1]        [,2]
## Woody.A    0.93446630 -0.04210386
## Woody.B   -0.05112059  0.34983808
## Woody.C   -0.88334571 -0.30773422
## Fruity.A  -0.88334571 -0.30773422
## Fruity.B  -0.05112059  0.34983808
## Fruity.C   0.93446630 -0.04210386
## Sweet.A   -0.50619940  0.64064720
## Sweet.B    0.32852896 -0.13064732
## Sweet.C    0.04089647 -0.27987047
## Alcohol.A -0.14013376 -0.73509355
## Alcohol.B  0.32852896 -0.13064732
## Alcohol.C -0.20448234  1.39935234
## Hedonic.A  0.00000000  0.00000000
## Hedonic.B  0.00000000  0.00000000
## Hedonic.C  0.00000000  0.00000000
## Hedonic.D  0.00000000  0.00000000
```

```r
#représentation graphique
plot(zl[,1],zl[,2],xlim=c(-1,1),ylim=c(-1.5,1.5),type="n")
abline(h=0,v=0,col="gray")
set.seed(1)
text(zl[,1],jitter(zl[,2],factor=60),labels=colnames(M),cex=0.7)
```

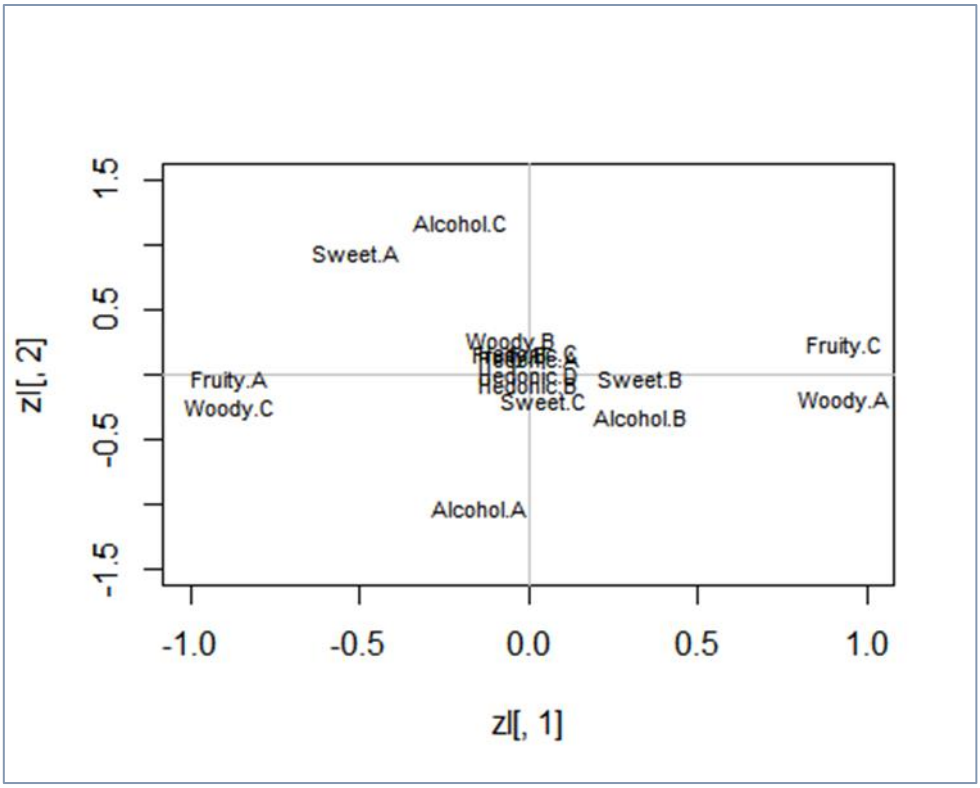


Figure 96 – ACD – Coordonnées factorielles des points modallités – Données "DIVAY"

J'ai rajouté du bruit (jitter) pour amoindrir le phénomène de superpositions entre les points. Un premier coup d'œil semble montrer un rôle fort de (Fruity.A, Woody.C) vs. (Fruity.C, Woody.A) pour le premier facteur ; (Alcohol.A vs. Alcohol.C) pour le second. L'étude des contributions dans la section suivante nous confirmera cela.

De nouveau, vérifions que les dispersions des modalités correspondent aux inerties portées par les facteurs. Nous retrouvons les valeurs de (λ_1, λ_2). L'analyse de données est une science exacte.

```r
#dispersion sur les facteurs
print(colSums(apply(zl,2,function(v){v^2*m_l/m})))

## [1] 0.2518879 0.2014454
```

9.3.4.2 Contribution aux facteurs

L'AFC traite de manière symétrique les lignes et les colonnes du tableau de contingence. Mais l'analyse des correspondances discriminante est asymétrique par nature, nous souhaitons expliquer les différences entre les classes en utilisant les (modalités des) variables explicatives. De fait, nous nous intéressons plutôt aux contributions (CTR) pour ce qui est des points modalités. Elles correspondent aux coordonnées passées au carré pondérées par les effectifs et normalisées.

Exemple « DIVAY »

Le calcul tient en une ligne de commande sous R.

```
#contributions des points modalités
sapply(1:H,function(h){(zl[,h]^2*m_l/m)/lambda[h]})

##                   [,1]         [,2]
## Woody.A   0.2311153044 0.0005866718
## Woody.B   0.0006916607 0.0405028431
## Woody.C   0.2065203392 0.0313402838
## Fruity.A  0.2065203392 0.0313402838
## Fruity.B  0.0006916607 0.0405028431
## Fruity.C  0.2311153044 0.0005866718
## Sweet.A   0.0508634634 0.1018709757
## Sweet.B   0.0285659545 0.0056487496
## Sweet.C   0.0005533285 0.0324022745
## Alcohol.A 0.0064967626 0.2235355361
## Alcohol.B 0.0285659545 0.0056487496
## Alcohol.C 0.0082999278 0.4860341172
## Hedonic.A 0.0000000000 0.0000000000
## Hedonic.B 0.0000000000 0.0000000000
## Hedonic.C 0.0000000000 0.0000000000
## Hedonic.D 0.0000000000 0.0000000000
```

L'indicateur numérique (CTR) ne fait que confirmer le graphique qui était édifiant de toute manière.

9.3.5 Représentation simultanée

Il y a une relation, dite quasi-barycentrique, entre les coordonnées des points-classes et points-modalités :

$$F_{kh} = \frac{1}{\sqrt{\lambda_h}} \sum_{l=1}^{L} \frac{m_{kl}}{m_k} O_{lh}$$

Et

$$O_{lh} = \frac{1}{\sqrt{\lambda_h}} \sum_{k=1}^{K} \frac{m_{kl}}{m_l} F_{kh}$$

C'est grâce à ces relations que nous pouvons projeter dans le même graphique les points-classes et les points modalités.

Exemple « DIVAY »

Vérifions tout d'abord cette fameuse relation entre les coordonnées des points-classes et des points modalités. Calculons les points-classes (section 9.3.3.1) à partir des points-modalités (section 9.3.4.1). Nous avons la correspondance.

```
#somme en ligne de M
m_k <- rowSums(M)
print(m_k)

## Beaujolais      Loire      Rhone
##         20         20         20

#somme en colonne de M
print(m_l)

##   Woody.A   Woody.B   Woody.C  Fruity.A  Fruity.B  Fruity.C   Sweet.A   Sweet.B
##         4         4         4         4         4         4         3         4
##   Sweet.C Alcohol.A Alcohol.B Alcohol.C Hedonic.A Hedonic.B Hedonic.C Hedonic.D
##         5         5         4         3         3         3         3         3

#coordonnées des points-classes à partir des points modalités
tempF <- t(sapply(1:K,function(k){1/sqrt(lambda)*(t(zl) %*% as.matrix(M[k,]/m_k[k]))}))
rownames(tempF) <- levels(D$Region)
print(tempF)

##                  [,1]       [,2]
## Beaujolais -0.5569077 -0.3935147
## Loire       0.6595342 -0.2345520
## Rhone      -0.1026265  0.6280667
```

Faisons l'inverse, calculons les coordonnées des points-modalités (section 9.3.4.1) à partir des points-classes (section 9.3.3.1). Ça marche également.

```
#nombre de points modalités
L <- ncol(M)

#coordonnées des points-modalités à partir des points-classes
tempO <- t(sapply(1:L,function(l){1/sqrt(lambda)*(t(zc) %*% as.matrix(M[,l]/m_l[l]))}))
rownames(tempO) <- colnames(M)
print(tempO)
```

```
##                     [,1]          [,2]
## Woody.A    9.344663e-01 -4.210386e-02
## Woody.B   -5.112059e-02  3.498381e-01
## Woody.C   -8.833457e-01 -3.077342e-01
## Fruity.A  -8.833457e-01 -3.077342e-01
## Fruity.B  -5.112059e-02  3.498381e-01
## Fruity.C   9.344663e-01 -4.210386e-02
## Sweet.A   -5.061994e-01  6.406472e-01
## Sweet.B    3.285290e-01 -1.306473e-01
## Sweet.C    4.089647e-02 -2.798705e-01
## Alcohol.A -1.401338e-01 -7.350936e-01
## Alcohol.B  3.285290e-01 -1.306473e-01
## Alcohol.C -2.044823e-01  1.399352e+00
## Hedonic.A -1.382568e-16 -2.473612e-16
## Hedonic.B -1.382568e-16 -2.473612e-16
## Hedonic.C -1.382568e-16 -2.473612e-16
## Hedonic.D -1.382568e-16 -2.473612e-16
```

Reste à réaliser la projection.

```r
#représentation simultanée
plot(zl[,1],zl[,2],xlim=c(-1,1),ylim=c(-1.5,1.5),type="n")
abline(h=0,v=0,col="gray")
text(zc[,1],zc[,2],labels=levels(D$Region),col=c("red","green","blue"))
set.seed(1)
text(zl[,1],jitter(zl[,2],factor=60),labels=colnames(M),cex=0.7)
```

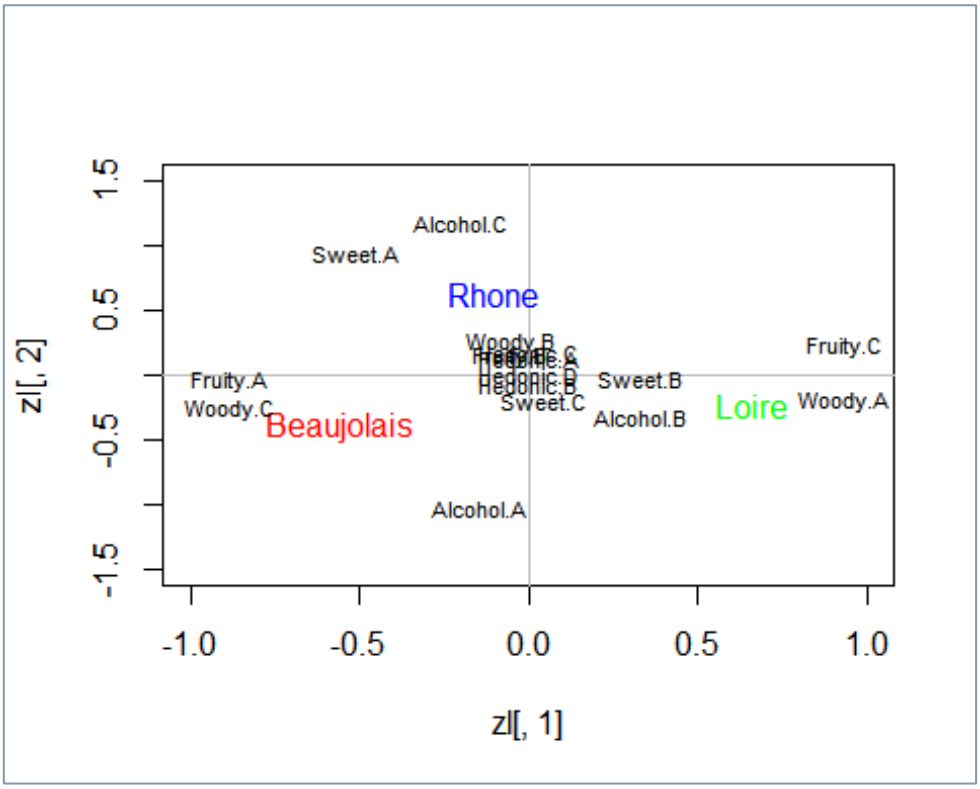


Figure 97 – ACD – Représentation simultanée des points-classes et points-modalités

Plus que les proximités qui peuvent être trompeuses, ce sont surtout les positions relatives qui importent dans ce genre de graphique. Il n'en reste pas moins que les associations semblent

évidentes. Elles le sont d'autant plus que nous les décelons après coup en inspectant attentivement le tableau des profils (Figure 94).

9.4 Affectation des classes

Le travail sur le tableau de contingence M ne doit pas nous induire en erreur. L'objectif final est d'une part de projeter les observations de notre échantillon de travail (n = 12) dans le repère factoriel et, d'autre part, de disposer d'un outil qui nous permettra d'y placer un individu supplémentaire pour pouvoir identifier sa classe d'appartenance.

9.4.1 Fonction discriminante canonique et coordonnées des individus

Les fonctions discriminantes canoniques doivent être définies sur les indicatrices des modalités. L'idée est d'exploiter la relation quasi-barycentrique pour en préciser les coefficients.

$$a_{lh} = \frac{O_{lh}}{p \times \sqrt{\lambda_h}}$$

Le coefficient $(\frac{1}{p})$ sert à transformer le vecteur des indicatrices en un vecteur de profil conforme à l'organisation du tableau de contingence M (Figure 94). Ainsi, la coordonnée d'un individu, décrit par ses indicatrices, sur le facteur (h) peut être obtenu à l'aide de la fonction suivante :

$$z_h = a_{1h}\, x_1 + a_{2h}\, x_2 + \cdots + a_{Lh} x_L$$

Exemple « DIVAY »

Nous calculons les coefficients des fonctions canoniques, que nous appliquons aux individus décrits par leurs indicatrices (Figure 92), nous les projetons enfin dans le plan factoriel, en faisant ressortir leur classe d'appartenance.

```
#nombre de variables
print(p)

## [1] 5

#les lambda_h
print(lambda)

## [1] 0.2518879 0.2014454
```

```r
#coefficients des fonctions discriminantes canoniques (a_lh)
coef_ <- t(apply(zl,1,function(v){v/(p*sqrt(lambda))}))
print(coef_)
```

```
##                  [,1]        [,2]
## Woody.A    0.37238311 -0.01876175
## Woody.B   -0.02037146  0.15589004
## Woody.C   -0.35201165 -0.13712830
## Fruity.A  -0.35201165 -0.13712830
## Fruity.B  -0.02037146  0.15589004
## Fruity.C   0.37238311 -0.01876175
## Sweet.A   -0.20171953  0.28547641
## Sweet.B    0.13091819 -0.05821726
## Sweet.C    0.01629717 -0.12471204
## Alcohol.A -0.05584305 -0.32756230
## Alcohol.B  0.13091819 -0.05821726
## Alcohol.C -0.08148584  0.62356018
## Hedonic.A  0.00000000  0.00000000
## Hedonic.B  0.00000000  0.00000000
## Hedonic.C  0.00000000  0.00000000
## Hedonic.D  0.00000000  0.00000000
```

```r
#coordonnées des individus à partir du tableau des indicatrices
coord <- as.matrix(IND) %*% coef_
print(coord)
```

```
##              [,1]        [,2]
##  [1,]  0.81984137 -0.42330305
##  [2,]  0.49922701 -0.04580100
##  [3,]  0.42708680 -0.24865126
##  [4,]  0.89198158 -0.22045279
##  [5,]  0.06880628  1.04616488
##  [6,] -0.65558849  0.92779833
##  [7,] -0.11054673 -0.09767278
##  [8,]  0.28682298  0.63597644
##  [9,] -0.74356919 -0.72653093
## [10,] -0.41192899 -0.43351259
## [11,] -0.11054673 -0.09767278
## [12,] -0.96158589 -0.31634248
```

```r
#affichage avec les barycentres de classes
plot(coord[,1],coord[,2],pch=19,col=c("darkorange","darkseagreen","deepskyblue")[D$Region])
abline(h=0,v=0,col="gray")
points(zc[,1],zc[,2],col=c("red","green","blue"),pch=18,cex=2)
text(zc[,1],zc[,2]+0.15,col=c("red","green","blue"),labels=levels(D$Region))
```

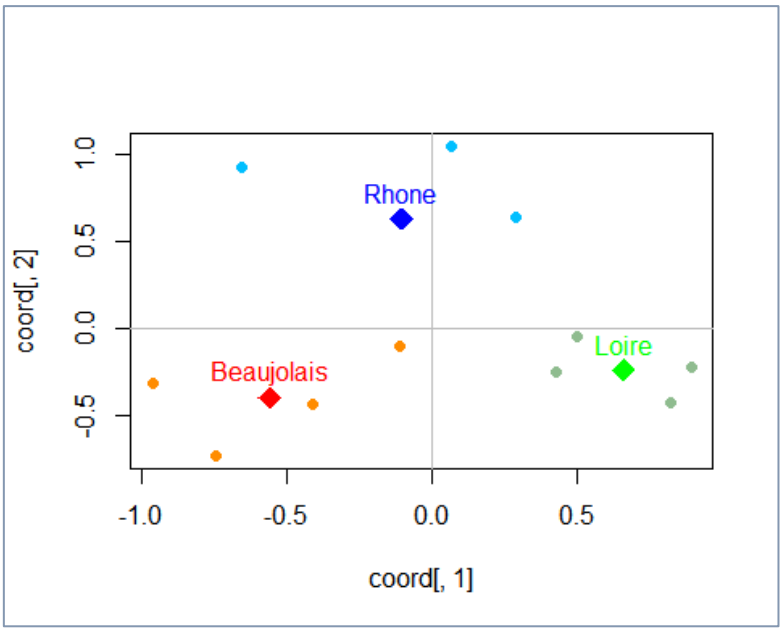


Figure 98 – ACD – Position des individus dans le plan factoriel – Données "DIVAY"

Il y a bien (n = 12) observations, les n°7 et n°11 sont superposés.

Les points-classes sont situés « au milieu » des individus qui leur sont associés. Nous contrôlons très facilement que les moyennes conditionnelles calculées sur les coordonnées des individus sont en adéquation.

```
#vérification des moyennes conditionnelles
aggregate(coord,by=list(D$Region),mean)

##      Group.1         V1         V2
## 1 Beaujolais -0.5569077 -0.3935147
## 2      Loire  0.6595342 -0.2345520
## 3      Rhone -0.1026265  0.6280667
```

9.4.2 Rapport de corrélation

Puisque nous disposons des coordonnées des individus sur les axes factoriels, nous pouvons facilement calculer les dispersions totales

$$SCT_h = \sum_{\omega} [z_h(\omega)]^2$$

Et en dériver les carrés des rapports de corrélation

$$\eta_h^2 = \frac{\lambda_h}{\frac{1}{n} SCT_h} = \frac{n \times \lambda_h}{SCT_h} = \frac{SCE_h}{SCT_h}$$

Exemple « DIVAY »

```
#nombre d'observations
print(n)

## [1] 12

#lambda
print(lambda)

## [1] 0.2518879 0.2014454

#SCT par facteur
SCT <- colSums(coord^2)
print(SCT)

## [1] 4.087869 3.486380

#carré des rapports de corrélation
print(n * lambda/SCT)

## [1] 0.7394207 0.6933683
```

Les deux facteurs présentent des qualités de discrimination (à peu près) identiques.

9.4.3 Traitement d'un individu supplémentaire

Les fonctions discriminantes canoniques nous permettent de positionner les individus supplémentaires dans le repère factoriel. A l'instar de l'analyse factorielle discriminante, nous pouvons mettre en place un système d'attribution de classes basé sur la proximité avec les barycentres conditionnels (section 8.2.1). Bien sûr, passer à la distance généralisée est possible si l'on souhaite prendre en compte la prévalence des classes.

Mettons que nous souhaitons identifier la région d'origine d'un vin présentant les caractéristiques suivantes :

Woody	Fruity	Sweet	Alcohol	Hedonic
A	C	B	B	A

En passant aux indicatrices, son vecteur de description est :

$$i = (1, 0, 0, 0, 0, 1, 0, 1, 0, 0,1,0, 1,0,0,0)$$

Nous utilisons les coefficients des fonctions discriminantes factorielles pour obtenir ses coordonnées.

```
#description de l'individu supplémentaire
i <- matrix(c(1,0,0,0,0,1,0,1,0,0,1,0,1,0,0,0),nrow=1,ncol=L)
print(i)

##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
## [1,]    1    0    0    0    0    1    0    1    0     0     1     0     1     0
##      [,15] [,16]
## [1,]     0     0

#coordonnées : produit matriciel avec les coefficients des fonctions discriminantes
coord_i <- i %*% coef_
print(coord_i)

##          [,1]      [,2]
## [1,] 1.006603 -0.153958

#placement dans le repère factoriel
plot(coord[,1],coord[,2],type="n",xlim=c(-1.5,1.5))
abline(h=0,v=0,col="gray")
points(zc[,1],zc[,2],col=c("red","green","blue"),pch=18,cex=2)
text(zc[,1],zc[,2]-0.15,col=c("red","green","blue"),labels=levels(D$Region))
points(coord_i[1],coord_i[2],pch=19, cex=2, col="pink")
```

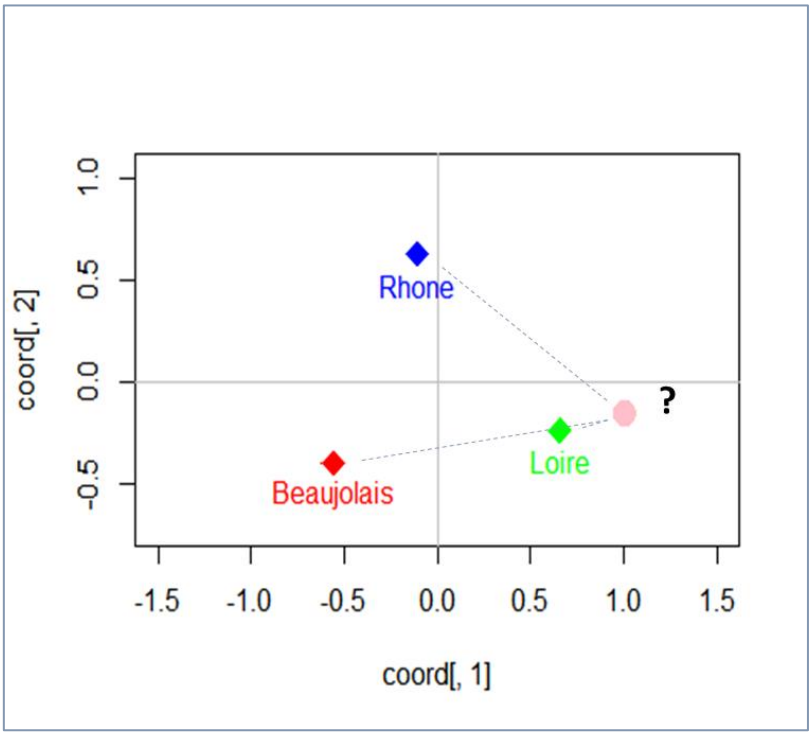


Figure 99 – ACD – Proximité avec les centres de classes de l'individu supplémentaire – Données "DIVAY"

La proximité avec « LOIRE » semble évidente. N'oublions pas que seules sont comptabilisées les distances euclidiennes aux centres de classes. Voyons ce qu'il en est.

```r
#coordonnées des centres de classe
print(zc)

##                  [,1]       [,2]
## Beaujolais -0.5569077 -0.3935147
## Loire       0.6595342 -0.2345520
## Rhone      -0.1026265  0.6280667

#coordonnée de l'individu à classer
print(coord_i)

##          [,1]      [,2]
## [1,] 1.006603 -0.153958

#distances euclidiennes aux centres de classes
apply(zc,1,function(v){sum((v-coord_i)^2)})

## Beaujolais      Loire      Rhone
##  2.5019519  0.1269519  1.8419519
```

Il est levé si doute il y avait : il s'agit très vraisemblablement d'un vin de Loire.

10 Outils logiciels

Les méthodes de machine learning n'ont de sens que si elles sont disponibles dans les outils logiciels. Elles ne servent absolument à rien si personne ne peut en faire usage. La programmation des algorithmes de statistique et de machine learning ont été pendant longtemps l'apanage des éditeurs commerciaux. La situation a changé au début des années 2000 avec l'arrivée des logiciels libres, dont TANAGRA d'ailleurs. Elle a complètement basculé dans les années 2010 avec R et Python, qui offrent des fonctionnalités de plus en plus étendues, qui sont entrés dans le monde de l'entreprise et apparaissent de ce fait dans les offres d'emploi qui touchent à nos domaines (data science, machine learning, statistique exploratoire, analyse de données, ...). Cette lame de fond me paraît irréversible. Ils sont aujourd'hui incontournables dans les formations en data science.

Nous passons en revue 4 outils dans ce chapitre. Nous prendrons pour repère les résultats des procédures DISCRIM et STEPDISC de SAS.

10.1 Proc DISCRIM et STEPDISC de SAS

Les procédures DISCRIM et STEPDISC du logiciel SAS font référence en matière d'analyse discriminante prédictive. Pour l'analyse factorielle discriminante, descriptive, il faut plutôt voir du côté de CANDISC. Nous étudions le comportement de DISCRIM et STEPDISC dans cette section (cf. aussi TUTO 3, section 4). Nous en décrivons les sorties sur les données « DATA 2 / TRAIN » en utilisant les options basiques. Rappelons que l'objectif est de prédire le TYPE de liqueur {KIRSCH, MIRAB, POIRE} en fonction de ses composants (MEOH, BU1, ..., ACAL).

10.1.1 Analyse discriminante linéaire

Voici la commande pour initier une analyse discriminante linéaire :

```
PROC DISCRIM DATA = EAU_DE_VIE;
        CLASS TYPE;
        VAR MEOH ACET BU1 BU2 ISOP MEPR PRO1 ACAL;
        PRIORS proportional;
RUN;
```

Où : CLASS désigne la variable cible ; VAR les prédicteurs ; PRIORS la distribution a priori des classes, elle est estimée à partir des fréquences observées avec l'option « proportional ». Deux options par défaut, non visibles dans la commande ci-dessus, sont importantes : (METHOD = NORMAL) pour une hypothèse gaussienne des distributions conditionnelles ; (POOL = YES) pour l'homoscédasticité.

Informations générales. SAS fournit une indication globale sur l'analyse menée.

La procédure DISCRIM

Taille d'échantillon totale	52	Total DDL	51
Variables	8	DDL dans les classes	49
Classes	3	DDL entre les classes	2

Nombre d'observations lues	52
Nombre d'observations utilisées	52

Information au niveau classe

TYPE	Nom de la variable	Fréquence	Poids	Proportion	Probabilité a priori
KIRSCH	KIRSCH	17	17.0000	0.326923	0.326923
MIRAB	MIRAB	15	15.0000	0.288462	0.288462
POIRE	POIRE	20	20.0000	0.384615	0.384615

Informations sur la matrice de covariance combinée

Rang de la matrice de covariance	Log. naturel du déterminant de la matrice de covariance
8	58.32674

Figure 100 – Informations générales – Proc DISCRIM – DATA 2 / TRAIN

Le rang de la matrice de la variance covariance intra-classe (W) donne une idée de la colinéarité entre les variables. Si (rang < p), soit nous avons des colonnes composées de valeurs constantes,

soit des variables sont redondantes (ex. âge et année de naissance). Le logarithme népérien du déterminant de W procède de la même idée.

Distance généralisée entre centres de classes. SAS affiche ensuite les distances au carré généralisées entre les centres de classes (section 2.1). L'information n'est pas anodine. Nous constatons notamment que « POIRE » et « MIRAB », qui sont plus proches, seront certainement plus difficiles à discriminer. Il s'agit de distance « généralisée », la matrice n'est donc pas symétrique, et la diagonale n'est pas nulle.

Distance au carré généralisée à TYPE			
De TYPE	KIRSCH	MIRAB	POIRE
KIRSCH	2.23606	29.85787	37.95913
MIRAB	29.60754	2.48639	7.21611
POIRE	38.28417	7.79147	1.91102

Figure 101 – Distance entre centres de classes – DATA 2 / TRAIN – Proc DISCRIM

Fonctions de classement. Viennent ensuite les fameuses fonctions de classement qui permettent d'affecter un score d'appartenance aux individus.

Fonction discriminante linéaire pour TYPE				
Variable	Libellé	KIRSCH	MIRAB	POIRE
Constante		-5.01645	-18.84069	-24.76488
MEOH	MEOH	0.00343	0.02903	0.03339
ACET	ACET	0.00639	0.01641	0.00751
BU1	BU1	-0.06368	0.40539	0.31805
BU2	BU2	-0.0008832	0.07135	0.11499
ISOP	ISOP	0.02308	0.02976	-0.00849
MEPR	MEPR	0.03749	-0.12894	0.06178
PRO1	PRO1	0.00197	-0.00541	-0.00832
ACAL	ACAL	0.06618	-0.22642	-0.13033

Figure 102 – Fonctions de classement – Proc DISCRIM – DATA 2 / TRAIN

Matrice de confusion en resubstitution. La matrice de confusion en resubstitution, calculée sur les données d'apprentissage, vient ponctuer les sorties. Les classes observées sont en ligne, les prédictions sont en colonne. Nous avons à la fois le décompte et les valeurs en pourcentage ligne (rappel).

La procédure DISCRIM
Synthèse de classification pour données de calibration : WORK.EAU_DE_VIE
Synthèse de resubstitution utilisant Fonction discriminante linéaire

Nombre d'observations et pourcentage classifiés dans TYPE				
De TYPE	KIRSCH	MIRAB	POIRE	Total
KIRSCH	17 100.00	0 0.00	0 0.00	17 100.00
MIRAB	0 0.00	14 93.33	1 6.67	15 100.00
POIRE	0 0.00	2 10.00	18 90.00	20 100.00
Total	17 32.69	16 30.77	19 36.54	52 100.00
A priori	0.32692	0.28846	0.38462	

Estimations du compte des erreurs pour TYPE				
	KIRSCH	MIRAB	POIRE	Total
Taux	0.0000	0.0667	0.1000	0.0577
A priori	0.3269	0.2885	0.3846	

Figure 103 – Matrice de confusion et décompte des erreurs – Proc DISCRIM – DATA 2 / TRAIN

D'autres sorties sont disponibles. La liste est longue (SAS, Table 35.1).

10.1.2 Sélection de variables

La Proc STEPDISC est dédiée à la sélection de variables (chapitre 4), avec les 3 stratégies : forward, backward et bidirectionnelle (stepwise). Nous avons soumis la commande suivante pour une sélection ascendante contrôlée par un risque de (SLENTRY = 1%) (section 4.1).

```
PROC STEPDISC DATA = EAU_DE_VIE METHOD = forward SLENTRY= 0.01;
        class TYPE;
        VAR MEOH ACET BU1 BU2 ISOP MEPR PRO1 ACAL;
RUN;
```

SAS détaille chaque étape.

Au premier passage, la variable MEOH est sélectionnée. $R^2 = 1 - \Lambda_1$, lorsque la variable est introduite. Nous pouvons aussi la voir sous l'angle d'un R² partiel avec ($R^2_{partiel} = 1 - \frac{\Lambda_1}{\Lambda_0}$), où ($\Lambda_0 = 1$), les classes ne sont pas discernables lorsqu'aucune variable n'est sélectionnée.

La procédure STEPDISC
Sélection ascendante : Etape 1

Statistiques pour l'entrée, DDL = 2, 49					
Variable	Libellé	R-carré	Valeur F	Pr > F	Tolérance
MEOH	MEOH	0.7174	62.19	<.0001	1.0000
ACET	ACET	0.0281	0.71	0.4969	1.0000
BU1	BU1	0.7138	61.11	<.0001	1.0000
BU2	BU2	0.0854	2.29	0.1122	1.0000
ISOP	ISOP	0.1123	3.10	0.0541	1.0000
MEPR	MEPR	0.3081	10.91	0.0001	1.0000
PRO1	PRO1	0.1645	4.82	0.0122	1.0000
ACAL	ACAL	0.0204	0.51	0.6042	1.0000

La variable MEOH sera saisie.

Figure 104 – Proc STEPDISC – Etape 1 – DATA 2 / TRAIN

Après l'introduction de la variable MEOH, une évaluation globale de la partition est réalisée (section 2.2.2).

Variable(s) ayant été saisie(s)
MEOH

Statistiques multivariées					
Statistique	Valeur	Valeur F	DDL num.	DDL den.	Pr > F
Lambda de Wilks	0.282629	62.19	2	49	<.0001
Trace de Pillai	0.717371	62.19	2	49	<.0001
Corrélation canonique moyenne au carré	0.358686				

Figure 105 – Evaluation globale après étape 1 – Proc STEPDISC – DATA 2 / TRAIN

Le processus continue ainsi jusqu'à ce qu'il ne soit plus possible d'introduire une nouvelle variable à la 4ème étape c.-à-d. dont la p-value du test F soit inférieur au seuil SLENTRY.

La procédure STEPDISC
Sélection ascendante : Etape 4

Statistiques pour l'entrée, DDL = 2, 46					
Variable	Libellé	R carré partiel	Valeur F	Pr > F	Tolérance
ACET	ACET	0.0661	1.63	0.2076	0.2987
BU2	BU2	0.1103	2.85	0.0679	0.3256
ISOP	ISOP	0.1216	3.18	0.0507	0.2511
PRO1	PRO1	0.0759	1.89	0.1628	0.3300
ACAL	ACAL	0.1382	3.69	0.0327	0.2672

Aucune variable ne peut être saisie.

Figure 106 – Proc STEPDISC – Arrêt du processus – DATA 2 / TRAIN

Un tableau récapitulatif vient enfin résumer les étapes de la sélection.

La procédure STEPDISC

Synthèse de la sélection ascendante										
Etape	Nombre dans	Saisi	Libellé	R carré partiel	Valeur F	Pr > F	Lambda de Wilk	Pr < Lambda	Corrélation canonique moyenne au carré	Pr > ASCC
1	1	MEOH	MEOH	0.7174	62.19	<.0001	0.28262884	<.0001	0.35868558	<.0001
2	2	BU1	BU1	0.3187	11.23	<.0001	0.19254694	<.0001	0.41786644	<.0001
3	3	MEPR	MEPR	0.2325	7.12	0.0020	0.14778601	<.0001	0.51868897	<.0001

Figure 107 – Proc STEPDISC – Résumé des étapes

STEPDISC possède aussi de nombreuses options et sorties.

10.2 ADL sous R – Le package « discriminR »

Sous R, la procédure lda() de la librairie MASS, même si elle est la plus connue, n'est pas des plus pratiques pour l'analyse discriminante prédictive, notamment parce qu'elle n'inclut pas une fonction de classement explicite dans ses sorties, et qu'elle ne propose pas de mécanisme de sélection de variables. Je décris longuement ses fonctionnalités dans un tutoriel (TUTO 1).

Le package « discriminR » me paraît plus approprié dans le contexte de cet ouvrage. Il s'inscrit dans l'esprit dans lequel j'enseigne l'analyse discriminante linéaire. Il a été conçu par 3 étudiants du Master SISE (promotion 2017-2018) : Tom Alran, Benoît Courbon et Samuel Rasser-Chinta. Ils l'ont développé dans le cadre de mon cours de « Programmation R » où, pour valider la matière, je demande aux étudiants d'élaborer un package R en mettant le focus sur une méthode de machine learning. J'y vois un triple avantage : ils étudient en détail une technique statistique ; ils mettent en pratique les notions de programmation avancée sous R ; ils apprennent à élaborer un package R respectant les normes et comprenant un fichier d'aide en anglais.

Pour l'analyse discriminante linéaire, ne pas retrouver sous R l'équivalent de ce que propose SAS (section 10.1) me chagrinait depuis un moment déjà. J'ai donc déboulé un beau jour en salle de cours avec un cahier des charges simple (argh…, les étudiants le savent par expérience, plus la description d'un projet à faire est simple, plus il sera compliqué à réaliser) : « Développez-moi une librairie pour R qui permet de reproduire les principaux résultats des procédures DISCRIM

et STEPDISC de SAS. Si vous proposez une solution pour la prise en compte des descripteurs discrets, c'est bien ». Ils ont généralement un mois pour mener à bien ce type d'entreprise. Sachant que dans le même temps, ils suivent les cours et doivent aussi travailler sur d'autres réalisations pour les autres matières.

Ah, nos amis n'ont pas fait semblant, c'est le moins qu'on puisse dire. Nous le verrons dans la suite de cette section, l'outil est pleinement opérationnel y compris dans la mise en œuvre de la sélection de variables lorsque les descripteurs sont composés d'un mix de prédicteurs quantitatifs et qualitatifs. Voilà un projet étudiant qui aura une vraie pérennité. Merci à Tom, Benoît et Samuel d'avoir bien voulu partager le fruit de leur travail avec le plus grand nombre.

10.2.1 Installation du package

Le package « **discriminR_0.1.0.tar.gz** » est autonome. Il ne nécessite pas de dépendances toujours compliquées à gérer lorsqu'on procède à une installation locale. Sous RSTUDIO, il faut aller dans le menu TOOLS / INSTALL PACKAGES... Dans la boîte de dialogue qui apparaît, nous sélectionnons l'item INSTALL FROM « Package Archive File (.zip ; .tar.gz) » puis nous désignons le fichier « discriminR_0.1.0.tar.gz » dans « Package Archive ».

Dans notre code, nous chargeons la librairie et nous en affichons les informations.

```
#importation de la librairie discriminR
library(discriminR)

#affichage des informations sur le package
library(help = discriminR)
```

Une fenêtre apparaît avec la fiche signalétique du package.

```
                    Information sur le package 'discriminR'

Description :

Package:            discriminR
Title:              Linear Discriminant Analysis
Version:            0.1.0
Authors@R:          c(person("Tom", "Alran", role = "aut", email =
                    "tom.alran@univ-lyon2.fr"), person("Benoit", "Courbon", role = "aut",
                    email = "benoit.courbon@univ-lyon2.fr"), person("Samuel", "Rasser", role
                    = c("aut","cre"), email = "samuel.rasser-chinta@univ-lyon2.fr") )
Description:        Functions and datasets to perform linear discriminant analysis (PROC
                    DISCRIM) and feature selection (PROC STEPDISC).
Depends:            R (>= 3.3.1)
License:            GPL-2
```

```
Encoding:           UTF-8
LazyData:           true
RoxygenNote:        7.1.0
NeedsCompilation:   no
Packaged:           2020-05-05 08:59:45 UTC; Zatovo
Author:             Tom Alran [aut], Benoit Courbon [aut], Samuel Rasser [aut, cre]
Maintainer:         Samuel Rasser <samuel.rasser-chinta@univ-lyon2.fr>
Built:              R 3.6.3; ; 2020-05-05 09:00:18 UTC; windows

Index :

HRdata                  HRanalytics
eval_preds              Evaluation of the predictions' quality
lda                     Linear Discriminant Analysis.
predict.LDA             Classes prediction
print                   Printing Linear Discriminant Analysis object
stepdisc                Feature selection.
```

Nous pouvons lancer les traitements dorénavant. Nous reprenons les données « DATA 2 » afin de pouvoir caler les résultats avec ceux de SAS (section 10.1).

10.2.2 Modélisation sur données d'apprentissage

Nous chargeons les données d'apprentissage...

```r
#lecture des données d'apprentissage
library(xlsx)
DTrain <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TRAIN")

#vérification du data.frame
str(DTrain)

## 'data.frame':    52 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  336 442 373 418 84 ...
##  $ ACET: num  225 338 356 62 65 ...
##  $ BU1 : num  1 1.9 0 0.8 2 2.2 1.9 0.4 0.9 0.8 ...
##  $ BU2 : num  1 10 29 0 2 52.1 46 3 36 12 ...
##  $ ISOP: num  92 91 83 89 2 123 85 6 84 7 ...
##  $ MEPR: num  37 30 27 24 0 38.2 33 9 36 9 ...
##  $ PRO1: num  177 552 814 342 288 ...
##  $ ACAL: num  0 31 11 7 6 13.3 35 4 4.8 2 ...
```

... et nous créons le modèle prédictif avec la fonction lda() de « discriminR ». Nous lui passons : le data frame à traiter (DTrain), le nom de la variable cible « TYPE », et nous réalisons un apprentissage sans sélection de variables pour l'instant (stepdisc = FALSE).

```r
#modélisation sur les données d'apprentissage
adlAll <- lda(DTrain,"TYPE",stepdisc = FALSE)
print(adlAll,digits=5)

##                    Count
## Total Sample Size     52
## Variables              8
## Classes                3
```

```
##
## Class Level Information :
##        frequency proportion
## KIRSCH        17  0.3269231
## MIRAB         15  0.2884615
## POIRE         20  0.3846154
##
##
## Classification functions coefficients:
##           KIRSCH     MIRAB     POIRE
## _CONST_ -5.01645 -18.84069 -24.76488
## MEOH     0.00343   0.02903   0.03339
## ACET     0.00639   0.01641   0.00751
## BU1     -0.06368   0.40539   0.31805
## BU2     -0.00088   0.07135   0.11499
## ISOP     0.02308   0.02976  -0.00849
## MEPR     0.03749  -0.12894   0.06178
## PRO1     0.00197  -0.00541  -0.00832
## ACAL     0.06618  -0.22642  -0.13033
```

L'affichage standard fournit les informations générales sur la base de données, les fréquences absolues et relatives des classes, et enfin les fameuses fonctions de classement, strictement identiques à ceux de Proc DISCRIM (Figure 102). La constante (intercept) est en première ligne, puis viennent les coefficients des variables. L'option « digits = 5 » de print() permet de moduler la précision des affichages.

Nous avons accès à des résultats détaillés en précisant « detailed = TRUE » lors de l'appel de print(). Les informations sur la matrice de variance covariance intra-classes (pooled) sont ajoutées dans les sorties.

```r
#affichage avec informations supplémentaires
print(adlAll,detailed=TRUE)
```

```
##                   Count
## Total Sample Size    52
## Variables             8
## Classes               3
##
## Class Level Information :
##        frequency proportion
## KIRSCH        17  0.3269231
## MIRAB         15  0.2884615
## POIRE         20  0.3846154
##
##                                                          Value
## Covariance Matrix Rank                                 8.00000
## Natural Log of the Determinant of the Covariance Matrix 58.32674
##
##
## Pooled Covariance Matrix:
##            MEOH       ACET         BU1         BU2        ISOP       MEPR
```

```
## MEOH 40223.7025  7653.7918  299.526327 -2522.67577  3494.67521 1340.44526
## ACET  7653.7918 15093.4728 -110.066327   148.72976   293.41307  223.26922
## BU1    299.5263  -110.0663   36.221265    22.43800    70.23612   21.94082
## BU2  -2522.6758   148.7298   22.438000  2825.92714  -431.04673  -91.42956
## ISOP  3494.6752   293.4131   70.236122  -431.04673  2156.40818  615.87467
## MEPR  1340.4453   223.2692   21.940816   -91.42956   615.87467  245.69489
## PRO1  7923.7079 14043.4170  272.533878 24116.53208 -1034.01305  305.21851
## ACAL   833.6814   462.6808    3.353327   -22.95753    15.06614    2.40533
##             PRO1       ACAL
## MEOH   7923.7079 833.681423
## ACET  14043.4170 462.680778
## BU1     272.5339   3.353327
## BU2   24116.5321 -22.957528
## ISOP  -1034.0130  15.066136
## MEPR    305.2185   2.405330
## PRO1 340956.9520 798.808215
## ACAL    798.8082  66.145806
##
##
## Classification functions coefficients:
##                KIRSCH         MIRAB         POIRE
## _CONST_ -5.0164526431 -18.840685373 -24.764879274
## MEOH     0.0034281727   0.029028464   0.033390239
## ACET     0.0063904459   0.016412816   0.007513488
## BU1     -0.0636813168   0.405389956   0.318047131
## BU2     -0.0008831867   0.071352049   0.114992814
## ISOP     0.0230821919   0.029763415  -0.008486278
## MEPR     0.0374935009  -0.128941667   0.061779984
## PRO1     0.0019711377  -0.005412661  -0.008318109
## ACAL     0.0661839107  -0.226423790  -0.130331850
```

10.2.3 Prédiction sur données test

Pour mesurer les performances prédictives, nous chargeons l'échantillon test...

```r
#lecture test
DTest <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "DATA_2_TEST")
str(DTest)
```

```
## 'data.frame':    50 obs. of  9 variables:
##  $ TYPE: Factor w/ 3 levels "KIRSCH","MIRAB",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ MEOH: num  3 475 186 371 583 0 421 557 167 523 ...
##  $ ACET: num  15 172 101 414 226 25 142 447 86 367 ...
##  $ BU1 : num  0.2 1.9 0 1.2 2.3 0.1 1.6 0 0 2.6 ...
##  $ BU2 : num  30 7 1.6 0 19 8 8 34 0 30 ...
##  $ ISOP: num  9 113 36 97 120 0 75 107 32 116 ...
##  $ MEPR: num  9 33 11 39 46 6 24 39 10 45 ...
##  $ PRO1: num  350 546 128 502 656 253 128 162 114 787 ...
##  $ ACAL: num  9 14 8 9 11 7 31 94 8 25 ...
```

... et nous effectuons la prédiction en faisant appel à predict(). Nous lui passons le modèle prédictif (model) et le data frame des descripteurs (test_X). Remarque : nous devons veiller à passer uniquement les descripteurs en paramètre « test_X » de la fonction de prédiction.

```r
#prediction en test
#attention, les colonnes de test_X doivent être limitées aux variables prédictives
predAll <- predict(model=adlAll,test_X=DTest[-1])
print(table(predAll))

## predAll
## KIRSCH  MIRAB  POIRE
##     15     19     16
```

Sur les 50 observations qui composent l'échantillon test, 15 ont été désigné KIRSCH, 19 MIRAB et 16 POIRE. Est-ce à juste titre ? Nous le saurons en confrontant les prédictions avec les classes observées. La fonction eval_preds() va plus loin en produisant les indicateurs de performances.

```r
#procédure matrice de confusion et indicateurs - ne renvoie pas de valeurs
#classe observée vs. classe prédite
eval_preds(DTest$TYPE,predAll)

##
## Confusion matrix :
##
##         predicted
## observed KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       0    14     3
##   POIRE       1     5    13
##
## Error = 0.18
##
## Recall =
##    KIRSCH     MIRAB     POIRE
## 1.0000000 0.8235294 0.6842105
##
## Precision =
##    KIRSCH     MIRAB     POIRE
## 0.9333333 0.7368421 0.8125000
```

Le taux d'erreur en test est de 18%, conforme à ce que nous avions obtenu en construisant pas à pas le classifieur sur les mêmes données (section 1.4).

10.2.4 Sélection de variables

10.2.4.1 Déroulement de la sélection

Nous passons à la sélection de variables. La librairie « discriminR », à l'instar de la Proc STEPDISC de SAS, propose les 3 stratégies : forward, backward, bidirectionnelle (stepwise). Nous souhaitons procéder à une sélection (stepdisc = TRUE) ascendante (method = « F ») avec pour seuil d'entrée (pval_cut = 0.01) ; (verbose = TRUE) fait afficher les tableaux intermédiaires à chaque étape.

```r
#sélection forward à 1% - détail des résultats
adlFwd <- lda(DTrain,"TYPE",stepdisc=TRUE,method="F",verbose=TRUE,pval_cut=0.01)

##
##  ************ Step  1  , forward ************
##
##      partial R2    F-value      p-value
## MEOH 0.71737116 62.1861291 3.586020e-14
## ACET 0.02814533  0.7095305 4.968583e-01
## BU1  0.71382712 61.1125850 4.862777e-14
## BU2  0.08541186  2.2880141 1.122087e-01
## ISOP 0.11226921  3.0984570 5.406192e-02
## MEPR 0.30814621 10.9121065 1.203236e-04
## PRO1 0.16453476  4.8249781 1.222491e-02
## ACAL 0.02035764  0.5091269 6.041644e-01
##
##  Variable  MEOH  will be entered
##
##  Variable(s) that have been entered :
## [1] "MEOH"
##
```

Ce premier tableau n'est pas sans rappeler celui de SAS (Figure 104). La meilleure variable est MEOH avec F = 62.19. Elle est acceptée parce que sa p-value est inférieure à 0.01.

Le processus se poursuit avec l'identification de la seconde variable.

```
##  ************ Step  2  , forward ************
##
##      partial R2   F-value      p-value
## ACET 0.10265978  2.745708 7.429700e-02
## BU1  0.31872862 11.228252 9.992715e-05
## BU2  0.13632009  3.788072 2.968026e-02
## ISOP 0.06569721  1.687604 1.957508e-01
## MEPR 0.21728681  6.662572 2.795527e-03
## PRO1 0.09536500  2.530037 9.023214e-02
## ACAL 0.16605431  4.778852 1.280284e-02
##
##  Variable  BU1  will be entered
##
##  Variable(s) that have been entered :
## [1] "MEOH" "BU1"
```

... qui s'avère être BU1. Et ainsi de suite...

```
##
##  ************ Step  3  , forward ************
##
##      partial R2  F-value     p-value
## ACET 0.07178629 1.817445 0.173671095
## BU2  0.11558499 3.071236 0.055772294
## ISOP 0.09450247 2.452583 0.097017683
## MEPR 0.23246765 7.117602 0.001994216
## PRO1 0.08541879 2.194821 0.122665772
## ACAL 0.09894342 2.580493 0.086431774
```

```
##
##  Variable  MEPR  will be entered
##
##  Variable(s) that have been entered :
## [1] "MEOH" "BU1"  "MEPR"
##
##  ************ Step  4  , forward ************
##
##      partial R2  F-value    p-value
## ACET 0.06606904 1.627088 0.20760596
## BU2  0.11034031 2.852582 0.06794383
## ISOP 0.12157018 3.183082 0.05072973
## PRO1 0.07587867 1.888507 0.16284217
## ACAL 0.13818038 3.687719 0.03270218
##
##
##  No more variables can be entered
```

Il n'est plus possible de sélectionner une variable à l'étape 4 (Figure 106).

Un tableau récapitulatif vient résumer la procédure de sélection (Figure 107 ; voir aussi section 4.1.5 avec le package « klaR »).

```
##
##      Step Entered Removed Partial R2          F-value
## [1,] "1"  "MEOH"  ""      "0.717371161327467" "62.1861291122058"
## [2,] "2"  "BU1"   ""      "0.318728619231166" "11.2282521730405"
## [3,] "3"  "MEPR"  ""      "0.23246764713538"  "7.11760186693394"
##      P-value                Wilk's lambda
## [1,] "3.58602036953926e-14" "0.282628838672533"
## [2,] "9.99271464310336e-05" "0.192546939167529"
## [3,] "0.00199421617469675"  "0.147786005256134"
##
##  3 features selected with forward approach :
##
## [1] "MEOH" "BU1"  "MEPR"
## attr(,"class")
## [1] "Selected Features"
```

10.2.4.2 L'objet « sélection » produit par lda()

La fonction de sélection fait bien plus que l'affichage étape par étape du processus (qui n'apparaît pas d'ailleurs si nous mettons verbose = FALSE). Nous disposons d'une part de la liste des variables sélectionnées....

```
#liste des variables sélectionnées
print(adlFwd$features)

## [1] "MEOH" "BU1"  "MEPR"
## attr(,"class")
## [1] "Selected Features"
```

... et du modèle élaboré sur ces variables.

```
#affichage des caractéristiques du modèle
print(adlFwd,digits=6)
```

```
##                    Count
## Total Sample Size    52
## Variables             3
## Classes               3
##
## Class Level Information :
##        frequency proportion
## KIRSCH        17  0.3269231
## MIRAB         15  0.2884615
## POIRE         20  0.3846154
##
##
## Classification functions coefficients:
##            KIRSCH      MIRAB      POIRE
## _CONST_ -3.610717 -14.775375 -18.371099
## MEOH     0.006922   0.021321   0.022619
## BU1     -0.076617   0.401044   0.373522
## MEPR     0.086678  -0.032474   0.046747
```

10.2.4.3 Prédiction et évaluation sur l'échantillon test

Il n'est donc pas nécessaire de réaliser un ré-apprentissage sur les variables sélectionnées. Nous pouvons utiliser directement l'objet pour la prédiction en test en lui passant le data frame avec l'ensemble des descripteurs. La fonction predict() se charge elle-même d'identifier ceux réellement utilisés. Ce mécanisme est très pratique et nous évite de triturer nous-même le data frame avec les risques de mauvaises manipulations que cela comporte.

```
#prédiction
#comme il y a sélection, l'outil s'occupe seul d'identifier les variables
#mais on n'indique pas la cible toujours
predFwd <- predict(adlFwd,DTest[-1])
```

```
##
## This model has been obtained with feature selection.
##  Same variables will be used to predict.
```

```
#évaluation
eval_preds(DTest$TYPE,predFwd)
```

```
##
## Confusion matrix :
##
##         predicted
## observed KIRSCH MIRAB POIRE
##   KIRSCH     14     0     0
##   MIRAB       0    12     5
##   POIRE       2     8     9
##
```

```
## Error = 0.3
##
## Recall =
##    KIRSCH     MIRAB     POIRE
## 1.0000000 0.7058824 0.4736842
##
## Precision =
##    KIRSCH     MIRAB     POIRE
## 0.8750000 0.6000000 0.6428571
```

Le taux d'erreur en test de 30%, moins bon que le modèle initial (18%). Même sur un échantillon réduit (50 observations), un tel écart reste élevé. Nous avons manifestement introduit trop peu de variables dans le modèle, il faudrait modifier (augmenter) le paramètre (pval_cut) et relancer la procédure.

10.2.5 Mix de descripteurs quantitatifs et qualitatifs

Par rapport au tandem « lda de MASS + KlaR » (section 4.1.5), le package « discriminR » s'avère d'un maniement bien plus pratique tout en produisant des sorties conformes à ceux de SAS. C'était le cœur du cahier des charges. L'outil devient réellement décisif en introduisant la gestion des variables prédictives qualitatives via un codage automatique 0/1 (section 3.2), que l'on peut intégrer dans une procédure de sélection de variables. Il nous facilite la vie en nous épargnant la gestion explicite des indicatrices additionnelles dans les data frame. Ces manipulations intermédiaires sont fastidieuses et souvent sources d'erreurs.

Nous reproduisons dans cette section l'expérimentation menée sur les données « HEART_STATLOG » (section 4.3.2), mais en optant pour une sélection backward à 1%. Pour rappel, la base décrit les caractéristiques de patients pour lesquels nous cherchons à prédire la présence ou absence d'une maladie (« disease »). Les variables prédictives sont pêle-mêle quantitatives et qualitatives. Une colonne « status » permet de distinguer les individus en apprentissage (train) et en test (test).

10.2.5.1 Chargement et préparation des données

Nous chargeons les données. Une petite inspection montre que les variables sont de différents types. Nous observons la liste des modalités pour les catégorielles.

```r
#lecture des données
library(xlsx)
D <- read.xlsx("Data_Illustration_Livre_ADL.xlsx", header = TRUE, sheetName = "HEART_STATLOG")

#affichage des caractéristiques des variables
print(summary(D))

##    status        disease         age            sex                chestpain
##  test :120   absence :150   Min.   :29.00   female: 87   asymptomatic  :129
##  train:150   presence:120   1st Qu.:48.00   male  :183   atypicalAngina: 42
##                             Median :55.00                nonAnginal    : 79
##                             Mean   :54.43                typicalAngina : 20
##                             3rd Qu.:61.00
##                             Max.   :77.00
##    restbpress     cholesteral     sugar                   electro
##  Min.   : 94.0   Min.   :126.0   high: 40   normal            :131
##  1st Qu.:120.0   1st Qu.:213.0   low :230   sttAbnormality    :  2
##  Median :130.0   Median :245.0              ventricHypertrophy:137
##  Mean   :131.3   Mean   :249.7
##  3rd Qu.:140.0   3rd Qu.:280.0
##  Max.   :200.0   Max.   :564.0
##   maxHeartRate   ExerciseAngina    oldpeak             slope
##  Min.   : 71.0   no :181        Min.   :0.00   downsloping: 18
##  1st Qu.:133.0   yes: 89        1st Qu.:0.00   flat       :122
##  Median :153.5                  Median :0.80   upsloping  :130
##  Mean   :149.7                  Mean   :1.05
##  3rd Qu.:166.0                  3rd Qu.:1.60
##  Max.   :202.0                  Max.   :6.20
##  vesselsColored                 thal
##  Min.   :0.0000   fixedEffect     : 14
##  1st Qu.:0.0000   normal          :152
##  Median :0.0000   reversableEffect:104
##  Mean   :0.6704
##  3rd Qu.:1.0000
##  Max.   :3.0000
```

« disease » est la variable cible, il ne sera pas nécessaire de la recoder. En revanche, il faudra traiter « sex » (2 modalités), « chestpain » (4), « sugar » (2), « electro » (3), « ExerciseAngina » (2), « slope » (3), « thal » (3).

Nous subdivisons la base en échantillons d'apprentissage et de test à l'aide de la colonne « status », qui est exclue pour la suite. Nous vérifions que les distributions des classes sont similaires dans les deux ensembles de données.

```r
#échantillon d'apprentissage
DTrain <- D[D$status=="train",2:ncol(D)]
print(prop.table(table(DTrain$disease)))

##
##   absence  presence
## 0.5466667 0.4533333
```

```r
#échantillon test
DTest <- D[D$status=="test",2:ncol(D)]
print(prop.table(table(DTest$disease)))

##
##   absence  presence
## 0.5666667 0.4333333
```

10.2.5.2 Apprentissage-test sans sélection de variables

Nous modélisons en utilisant l'ensemble des descripteurs dans un premier temps.

```r
#apprentissage sur TRAIN
adlAll <- lda(DTrain,"disease",stepdisc=FALSE)

##
## Categorical features have been encoded into binary variables.

print(adlAll)

##                   Count
## Total Sample Size   150
## Variables            18
## Classes               2
##
## Class Level Information :
##          frequency proportion
## absence         82  0.5466667
## presence        68  0.4533333
##
##
## Classification functions coefficients:
##                                  absence      presence
## _CONST_                    -124.35463814 -127.38630158
## age                           1.18362440    1.19136522
## sexmale                      14.26590352   15.91231810
## chestpainatypicalAngina       0.56676653   -1.88393458
## chestpainnonAnginal           3.48719945    1.66522654
## chestpaintypicalAngina       -2.80809449   -7.42225728
## restbpress                    0.34602885    0.36902694
## cholesteral                   0.04071031    0.03732874
## sugarlow                     10.40571066   11.71455646
## electrosttAbnormality       -15.81183260  -12.82133574
## electroventricHypertrophy    -1.85526002   -1.24105261
## maxHeartRate                  0.48558165    0.45793094
## ExerciseAnginayes             4.91703079    5.96121377
## oldpeak                       3.06519933    3.77512564
## slopeflat                    18.64287061   19.82149125
## slopeupsloping               14.74668502   15.25260096
## vesselsColored               -2.50872142   -1.03082538
## thalnormal                   21.15251326   20.12111204
## thalreversableEffect         14.85398895   16.16356606
```

Les fonctions de classement comportent 18 « variables », composées :

- Des variables quantitatives introduites telles quelles.

- Des indicatrices issues de qualitatives (section 3.2.1), avec comme modalité de référence la première par ordre alphabétique.

Nous passons les variables originelles (sauf la classe à prédire bien sûr) en prédiction, charge à la fonction predict() de procéder en interne à la création des indicatrices adéquates pour les qualitatives. Le taux d'erreur en test de 16.67 % (section 4.3.2).

```
#évaluation en test
#la colonne "disease" n'est pas prise en compte dans predict()
eval_preds(DTest$disease,predict(adlAll,DTest[-1]))

##
## Categorical features have been encoded into binary variables.
##
## Confusion matrix :
##
##           predicted
## observed   absence presence
##   absence       59        9
##   presence      11       41
##
## Error = 0.166666666666667
##
## Recall =
##   absence  presence
## 0.8676471 0.7884615
##
## Precision =
##   absence  presence
## 0.8428571 0.8200000
```

10.2.5.3 Apprentissage-test avec sélection de variables

Nous réitérons la démarche mais avec une sélection backward à 1%. L'outil traite les indicatrices indépendamment les unes des autres, les modalités de références peuvent être composites pour certaines variables à l'issue du traitement. Seules les premières et dernières étapes sont reprises dans les sorties ci-dessous.

```
#sélection backward à 1%
adlBwd <- lda(DTrain,"disease",stepdisc=TRUE,method="B",pval_cut=0.01,verbose=TRUE)

##
## Categorical features have been encoded into binary variables.
##
##  ************ Step  18  , backward ************
##
##                              partial R2    F-value      p-value
## age                        0.0002866056  0.0375561 8.466374e-01
## sexmale                    0.0390442788  5.3226183 2.261913e-02
```

```
## chestpainatypicalAngina   0.0481966419  6.6334711 1.111753e-02
## chestpainnonAnginal       0.0422200814  5.7746363 1.766036e-02
## chestpaintypicalAngina    0.1031577020 15.0680437 1.636021e-04
## restbpress                0.0120291286  1.5950023 2.088562e-01
## cholesteral               0.0024551245  0.3224129 5.711331e-01
## sugarlow                  0.0187843396  2.5078569 1.156911e-01
## electrosttAbnormality     0.0049105421  0.6464555 4.228394e-01
## electroventricHypertrophy 0.0077627506  1.0248762 3.132313e-01
## maxHeartRate              0.0243690395  3.2720817 7.276147e-02
## ExerciseAnginayes         0.0152825773  2.0330885 1.562862e-01
## oldpeak                   0.0339506092  4.6038327 3.374467e-02
## slopeflat                 0.0081968731  1.0826648 3.000188e-01
## slopeupsloping            0.0011587337  0.1519702 6.972927e-01
## vesselsColored            0.1185150681 17.6128637 4.960842e-05
## thalnormal                0.0045251940  0.5954951 4.416918e-01
## thalreversableEffect      0.0075095661  0.9911966 3.212855e-01
##
##  Variable  age  will be removed
##
```

La variable « âge » est la première retirée. Le processus continue jusqu'à l'étape ci-dessous où il n'est plus possible de retirer une variable.

```
##
##  ************ Step  7  , backward ************
##
##                          partial R2   F-value      p-value
## sexmale                  0.05147844  7.706666 6.243847e-03
## chestpainatypicalAngina 0.07193508 11.006537 1.153251e-03
## chestpainnonAnginal     0.09941603 15.675469 1.184453e-04
## chestpaintypicalAngina  0.13636216 22.420771 5.246363e-06
## oldpeak                 0.12021182 19.402486 2.070543e-05
## vesselsColored          0.13639381 22.426798 5.232171e-06
## thalreversableEffect    0.06676170 10.158350 1.766931e-03
##
##  No variable can be removed
##
##
##  No further steps are possible.
##
##       Step Entered Removed                     Partial R2
##  [1,] "18" ""      "age"                       "0.000286605584117153"
##  [2,] "17" ""      "slopeupsloping"            "0.00113486662163507"
##  [3,] "16" ""      "cholesteral"               "0.00223601887799008"
##  [4,] "15" ""      "electrosttAbnormality"     "0.00472281032518476"
##  [5,] "14" ""      "thalnormal"                "0.00490040416610563"
##  [6,] "13" ""      "electroventricHypertrophy" "0.00489774253087127"
##  [7,] "12" ""      "ExerciseAnginayes"         "0.0164708832619989"
##  [8,] "11" ""      "sugarlow"                  "0.0189260526274469"
##  [9,] "10" ""      "restbpress"                "0.0170804401774022"
## [10,] "9"  ""      "slopeflat"                 "0.0200342951531068"
## [11,] "8"  ""      "maxHeartRate"              "0.0433073816955"


##       F-value              P-value              Wilk's lambda
##  [1,] "0.0375560953059741" "0.846637399679428"  "0.390536689220165"
##  [2,] "0.14997259294572"   "0.699185235651142"  "0.390980399825641"
##  [3,] "0.298056971788304"  "0.586017670563407"  "0.391856598577525"
##  [4,] "0.635859627991193"  "0.426624805769733"  "0.393716044778998"
##  [5,] "0.664812411937397"  "0.416302285749319"  "0.395654913766761"
```

```
##  [6,] "0.66937139293864"   "0.414702016869297"  "0.397602267301696"
##  [7,] "2.29430015694692"   "0.132153485439282"  "0.404260799741642"
##  [8,] "2.66217981792547"   "0.10503980549686"   "0.412059458743458"
##  [9,] "2.41543792768495"   "0.122418302570125"  "0.419219919499647"
## [10,] "2.86214232555534"   "0.0929113063861602" "0.427790398609046"
## [11,] "6.38276150796217"   "0.0126279722829885" "0.447155533997114"
##
##  7 features selected with backward approach :
##
## [1] "sexmale"                 "chestpainatypicalAngina"
## [3] "chestpainnonAnginal"     "chestpaintypicalAngina"
## [5] "oldpeak"                 "vesselsColored"
## [7] "thalreversableEffect"
## attr(,"class")
## [1] "Selected Features"
```

Il reste 7 « variables » dans le modèle prédictif. La modalité de référence a été modifiée pour « thal » où elle est composée de {fixedEffect, normal}. Voici les fonctions de classement.

```
#coefficients du modèle
print(adlBwd,6)
```

```
##                   Count
## Total Sample Size   150
## Variables             7
## Classes               2
##
## Class Level Information :
##          frequency proportion
## absence         82  0.5466667
## presence        68  0.4533333
##
##
## Classification functions coefficients:
##                           absence  presence
## _CONST_                 -3.500184 -6.264811
## sexmale                  3.007833  4.621107
## chestpainatypicalAngina  4.655284  1.935244
## chestpainnonAnginal      4.645275  2.158346
## chestpaintypicalAngina   2.688064 -2.079371
## oldpeak                  0.786850  1.811195
## vesselsColored           0.713752  2.029111
## thalreversableEffect     0.608489  2.443855
```

En test, nous passons tous les descripteurs, l'outil se charge encore une fois de ne mettre à contribution que ce qui est nécessaire.

```
#évaluation en test du modèle simplifié
eval_preds(DTest$disease,predict(adlBwd,DTest[-1]))
```

```
##
## Categorical features have been encoded into binary variables.
##
## This model has been obtained with feature selection.
##  Same variables will be used to predict.
## Confusion matrix :
```

```
##
##            predicted
## observed    absence presence
##   absence        59        9
##   presence       11       41
##
## Error = 0.166666666666667
##
## Recall =
##   absence  presence
## 0.8676471 0.7884615
##
## Precision =
##   absence  presence
## 0.8428571 0.8200000
```

Le taux d'erreur en test est maintenu à 16.67 %, mais avec plus de deux fois moins de « descripteurs ». C'est tout l'intérêt de la sélection de variables. C'est tout l'intérêt aussi du package « discriminR » qui nous permet de mener des études complexes tout en nous épargnant une gestion des données délicate.

10.3 TANAGRA (LDA et STEPDISC)

J'ai beaucoup travaillé sur l'analyse discriminante linéaire dans TANAGRA, avec les composants « Linear Discriminant Analysis » (onglet Spv Learning) pour la modélisation et « StepDisc » (onglet Feature Selection) pour la sélection de variables.

MANOVA

Stat	Value	p-value
Wilks' Lambda	0.0667	-
Bartlett -- C(16)	123.1845	0.0000
Rao -- F(16, 84)	15.0761	0.0000

LDA Summary

	Classification functions			Statistical Evaluation			
Attribute	KIRSCH	MIRAB	POIRE	Wilks L.	Partial L.	F(2,42)	p-value
MEOH	0.003428	0.029028	0.033390	0.117975	0.565488	16.13607	0.000006
ACET	0.006390	0.016413	0.007513	0.074153	0.899667	2.34196	0.108572
BU1	-0.063681	0.405390	0.318047	0.084183	0.792475	5.49926	0.007563
BU2	-0.000883	0.071352	0.114993	0.095695	0.697142	9.12300	0.000513
ISOP	0.023082	0.029763	-0.008486	0.072310	0.922600	1.76176	0.184196
MEPR	0.037494	-0.128942	0.061780	0.087798	0.759852	6.63695	0.003128
PRO1	0.001971	-0.005413	-0.008318	0.092396	0.722038	8.08434	0.001071
ACAL	0.066184	-0.226424	-0.130332	0.075884	0.879150	2.88670	0.066885
constant	-5.016453	-18.840686	-24.764879	-			

Figure 108– Output de TANAGRA sur DATA 2 / TRAIN

Leur mode opératoire et les sorties sont longuement décrits dans un tutoriel (TUTO 3, section 3.1 à 3.3). Pour « Linear Discriminant Analysis », l'évaluation globale et les coefficients de classement sur les données « DATA 2 / TRAIN » sont regroupés dans la Figure 108. TANAGRA propose une évaluation de la pertinence des descripteurs sur la base du F d'exclusion explicitée en section 2.3. « StepDisc » fournit la liste des variables sélectionnées et les résultats détaillés par étape (Figure 109).

Selected attributes' subset

N°	Selected atts
1	MEOH
2	BU1
3	MEPR
4	ACAL

Detailed results

N°	d.f	Best	Sol.1	Sol.2	Sol.3	Sol.4	Sol.5
1	(2, 49)	MEOH L : 0.2826 F : 62.19 p : 0.0000	MEOH L : 0.2826 F : 62.19 p : 0.0000	BU1 L : 0.2862 F : 61.11 p : 0.0000	MEPR L : 0.6919 F : 10.91 p : 0.0001	PRO1 L : 0.8355 F : 4.82 p : 0.0122	ISOP L : 0.8877 F : 3.10 p : 0.0541
2	(2, 48)	BU1 L : 0.1925 F : 11.23 p : 0.0001	BU1 L : 0.1925 F : 11.23 p : 0.0001	MEPR L : 0.2212 F : 6.66 p : 0.0028	ACAL L : 0.2357 F : 4.78 p : 0.0128	BU2 L : 0.2441 F : 3.79 p : 0.0297	ACET L : 0.2536 F : 2.75 p : 0.0743
3	(2, 47)	MEPR L : 0.1478 F : 7.12 p : 0.0020	MEPR L : 0.1478 F : 7.12 p : 0.0020	BU2 L : 0.1703 F : 3.07 p : 0.0558	ACAL L : 0.1735 F : 2.58 p : 0.0864	ISOP L : 0.1744 F : 2.45 p : 0.0970	PRO1 L : 0.1761 F : 2.19 p : 0.1227
4	(2, 46)	ACAL L : 0.1274 F : 3.69 p : 0.0327	ACAL L : 0.1274 F : 3.69 p : 0.0327	ISOP L : 0.1298 F : 3.18 p : 0.0507	BU2 L : 0.1315 F : 2.85 p : 0.0679	PRO1 L : 0.1366 F : 1.89 p : 0.1628	ACET L : 0.1380 F : 1.63 p : 0.2076
5	(2, 45)	-	ISOP L : 0.1120 F : 3.09 p : 0.0553	BU2 L : 0.1128 F : 2.91 p : 0.0648	ACET L : 0.1159 F : 2.23 p : 0.1195	PRO1 L : 0.1221 F : 0.97 p : 0.3859	-

Figure 109 – Output de StepDisc de TANAGRA – DATA 2 / TRAIN

Par rapport aux autres outils de ce chapitre, TANAGRA a pour lui sa facilité d'utilisation et sa capacité d'interfaçage avec Excel ou LibreOffice/OpenOffice. Je l'utilise systématiquement dans mes cours d'initiation pour m'assurer que les étudiants comprennent bien la notion de fonction de classement, et qu'ils sont capables d'exploiter les coefficients pour reproduire le mécanisme de prédiction et d'évaluation sur tableur (calcul des scores, affectation au maximum, construction de la matrice de confusion, déduction des indicateurs de performances).

10.4 ADL sous Python – Le package « scikit-learn »

Le mode de fonctionnement et les sorties de la classe « LinearDiscriminantAnalysis » de la librairie « scikit-learn » sous Python ont été longuement décrits dans un tutoriel (TUTO 2). J'y montre comment dériver ses résultats – qui sont relativement restreints – pour retrouver à l'identique ceux de SAS. L'outil mixe l'analyse prédictive et descriptive. Il ne propose pas nativement de procédure de sélection de variables mais, avec un peu d'huile de coude, nous pouvons facilement bricoler une procédure ad hoc à partir d'une fonction d'évaluation des contributions des variables (TUTO 2, section 2.5).

Pour des données « DATA 2 », nous en reprenons les grandes lignes en nous limitant à la modélisation sur l'échantillon d'apprentissage, la prédiction et l'évaluation sur l'échantillon test.

10.4.1 Modélisation sur les données d'apprentissage

Nous commençons par le chargement et l'inspection des données d'apprentissage.

```
#importation de l'échantillon train
import pandas
DTrain = pandas.read_excel("Data_Illustration_Livre_ADL.xlsx",sheet_name="DATA_2_TRAIN")
print(DTrain.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52 entries, 0 to 51
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   TYPE    52 non-null     object
 1   MEOH    52 non-null     float64
 2   ACET    52 non-null     float64
 3   BU1     52 non-null     float64
 4   BU2     52 non-null     float64
 5   ISOP    52 non-null     int64
 6   MEPR    52 non-null     float64
 7   PRO1    52 non-null     float64
 8   ACAL    52 non-null     float64
dtypes: float64(7), int64(1), object(1)
memory usage: 3.8+ KB
None
```

Nous préparons des structures spécifiques pour manipuler la matrice des descripteurs (XTrain) et le vecteur cible (yTrain).

```
#préparation des structures
yTrain = DTrain.iloc[:,0]
XTrain = DTrain.iloc[:,1:]
```

Puis nous importons la classe dédiée à l'analyse discriminante et nous lançons la modélisation.

```
#importation de la classe de calcul
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

#instanciation
lda = LinearDiscriminantAnalysis(solver="eigen")

#apprentissage
lda.fit(XTrain,yTrain)

#affichage brut des coefficients
print(lda.coef_)
```

```
[[ 0.00363806  0.0067817  -0.06758017 -0.00093726  0.02449539  0.03978902
   0.00209182  0.07023599]
 [ 0.03080572  0.01741768  0.43020975  0.07572054  0.03158566 -0.13683605
  -0.00574405 -0.24028647]
 [ 0.03543454  0.0079735   0.3375194   0.12203319 -0.00900585  0.06556243
  -0.00882738 -0.13831135]]
```

L'affichage des coefficients est un peu brut de décoffrage. Une petite réorganisation cosmétique s'impose.

```
#structure temporaire pour affichage des coefficients
tmp = pandas.DataFrame(lda.coef_.transpose(),columns=lda.classes_,index=XTrain.columns)
print(tmp)
```

```
        KIRSCH     MIRAB     POIRE
MEOH  0.003638  0.030806  0.035435
ACET  0.006782  0.017418  0.007973
BU1  -0.067580  0.430210  0.337519
BU2  -0.000937  0.075721  0.122033
ISOP  0.024495  0.031586 -0.009006
MEPR  0.039789 -0.136836  0.065562
PRO1  0.002092 -0.005744 -0.008827
ACAL  0.070236 -0.240286 -0.138311
```

Sans oublier la constante bien sûr.

```
#et les constantes
print(lda.intercept_)
```

```
[ -5.25513156 -19.91808283 -26.22259567]
```

Les coefficients sont identiques à ceux de SAS à un coefficient ($\frac{n-K}{n}$) près, qui ne dépend pas de la classe d'appartenance donc. Les comportements en classement des modèles sont strictement identiques.

10.4.2 Prédiction et évaluation en test

Nous chargeons et inspectons l'échantillon test.

```
#chargement échantillon test
DTest = pandas.read_excel("Data_Illustration_Livre_ADL.xlsx",sheet_name="DATA_2_TE
ST")
print(DTest.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   TYPE    50 non-null     object
 1   MEOH    50 non-null     int64
 2   ACET    50 non-null     int64
 3   BU1     50 non-null     float64
 4   BU2     50 non-null     float64
 5   ISOP    50 non-null     int64
 6   MEPR    50 non-null     int64
 7   PRO1    50 non-null     int64
 8   ACAL    50 non-null     float64
dtypes: float64(3), int64(5), object(1)
memory usage: 3.6+ KB
None
```

Nous préparons la matrice des descripteurs (XTest) et nous effectuons la prédiction en faisant appel à la méthode... predict(). Nous vérifions les effectifs des classes prédites.

```
#récupérer la matrice des X
XTest = DTest.iloc[:,1:]

#prediction simple
pred = lda.predict(XTest)

#distribution des classes prédites
import numpy
print(numpy.unique(pred,return_counts=True))

(array(['KIRSCH', 'MIRAB', 'POIRE'], dtype='<U6'), array([15, 19, 16], dtype=int64))
```

Comme avec « discriminR » (section 10.2.3), 15 objets ont été classés KIRSCH, 19 MIRAB et 16 POIRE. Via le module « metrics » de la librairie « scikit-learn » toujours, on ne s'étonnera pas d'obtenir la même matrice de confusion...

```
#matrice de confusion
from sklearn import metrics
print(metrics.confusion_matrix(DTest.TYPE,pred))

[[14  0  0]
 [ 0 14  3]
 [ 1  5 13]]
```

... et les mêmes valeurs des indicateurs de performances.

```
#rapport sur les performances prédictives
print(metrics.classification_report(DTest.TYPE,pred))

              precision    recall  f1-score   support

      KIRSCH       0.93      1.00      0.97        14
       MIRAB       0.74      0.82      0.78        17
       POIRE       0.81      0.68      0.74        19

    accuracy                           0.82        50
   macro avg       0.83      0.84      0.83        50
weighted avg       0.82      0.82      0.82        50
```

11 Références

11.1 Références en français

- Bardos M., « Analyse Discriminante – Application au risque et scoring financier », Dunod, 2001 ; Chapitre 2 « Analyse discriminante de Fisher ».
- Celeux G., Nakache P.J., « Analyse Discriminante sur Variables Qualitatives », Polytechnica, 1994 ; Carlier A., « Chapitre 5 - Méthode Exploratoires ».
- Diday E., Lemaire J., Pouget J., Testu F., « Eléments d'Analyse de Données », Dunod, 1982 ; Chapitre 4, « Analyse discriminante ».
- Lebart L., Morineau A., Piron M., « Statistique Exploratoire Multidimensionnelle », Dunod, 2000 ; Section 3.3, « Analyse Factorielle Discriminante ».
- Nakache J.P., Confais J., « Statistique explicative appliquée », Technip, 2003 ; Chapitre 1, « Analyse discriminante sur variables quantitatives » ; Chapitre 2, « Analyse discriminante sur variables qualitative dérivée de l'analyse factorielle ».
- (Rakotomalala, 2010) Rakotomalala R., « Comparaison de populations – Tests paramétriques », version 1.2, mai 2010 ; *in* Ouvrages de statistiques.
- (Rakotomalala, 2011) Rakotomalala R., « Pratique de la Régression Logistique – Régression Logistique Binaire et Polytomique », juin 2011 ; *in* Cours de Régression Logistique.
- Saporta G., « Probabilités, Analyse de données et Statistique », Technip, 2006 ; Chapitre 18, « Analyse discriminante et régression logistique ».
- Tenenhaus M., « Statistique – Méthodes pour décrire, expliquer et prévoir », Dunod, 2007 ; Chapitre 10, « L'Analyse discriminante ».

– Tomassone R., Danzart M., Daudin J.J., Masson J.P., « Discrimination et Classement », Masson, 1988 ; Chapitre 3, « Discrimination entre deux populations ».
– Tufféry S., « Data Mining et Statistique Décisionnelle – L'intelligence des données », Tecnnip, 2012 ; Chapitre 12, « L'analyse discriminante linéaire et ses généralisations ».

11.2 Références en anglais

– Hastie T., Tibshirani R., Friedman J., « The Elements of Statistical Learning », Second Edition, 12th printing, 2017 ; Section 4.3, « Linear Discriminant Analysis ».
– Huberty C.J., Olejnik S., « Applied Manova and Discriminant Analysis », Second Edition, Wiley Series in Probability and Statistics, 2006 ; Tous les chapitres, mais en particulier la partie IV, « Group-Membership Prediction ».
– SAS, « STAT / STAT 13.2 User's Guide – The DISCRIM Procedure », 2014.

11.3 Tutoriels

J'ai écrit de nombreux tutoriels où il a été question de l'analyse discriminante. Sont regroupés ici les principaux qui abordent des aspects spécifiques de la méthode ou qui mettent en avant les fonctionnalités de certains outils. Pour obtenir une liste exhaustive, voir http://tutoriels-data-mining.blogspot.com/search/label/Analyse%20discriminante (37 tutoriels en avril 2020).

– [TUTO 1] « Analyse discriminante sous R », avril 2020 ; http://tutoriels-data-mining.blogspot.com/2020/04/analyse-discriminante-lineaire-sous-r.html
– [TUTO 2] « Analyse discriminante sous Python », avril 2020 ; http://tutoriels-data-mining.blogspot.com/2020/04/analyse-discriminante-lineaire-sous.html
– [TUTO 3] « Analyse discriminante linéaire – Comparaisons (de logiciels) », juillet 2012 ; http://tutoriels-data-mining.blogspot.com/2012/07/analyse-discriminante-lineaire.html
– [TUTO 4] « Classifieurs linéaires », mai 2013 ; http://tutoriels-data-mining.blogspot.com/2013/05/classifieurs-lineaires.html
– [TUTO 5] « Pipeline sous Python – La méthode DISQUAL », juin 2018 ; http://tutoriels-data-mining.blogspot.com/2018/06/pipeline-sous-python-la-methode-disqual.html

– [TUTO 6] « Stepdisc – Analyse discriminante », décembre 2006 ; http://tutoriels-data-mining.blogspot.com/2008/03/stepdisc-analyse-discriminante.html
– [TUTO 7] « Analyse discriminante et régression linéaire », avril 2014 ; http://tutoriels-data-mining.blogspot.com/2014/04/analyse-discriminante-et-regression.html
– [TUTO 8] « Bayésien naïf pour prédicteurs continus », octobre 2010 ; http://tutoriels-data-mining.blogspot.com/2010/10/bayesien-naif-pour-predicteurs-continus.html
– [TUTO 9] « Le classifieur bayésien naïf revisité », mars 2010 ; http://tutoriels-data-mining.blogspot.com/2010/03/le-classifieur-bayesien-naif-revisite.html
– [TUTO 10] « Analyse Discriminante PLS », mai 2008 ; http://tutoriels-data-mining.blogspot.com/2008/05/analyse-discriminante-pls.html
– [TUTO 11] « Régression PLS – Comparaison de logiciels », mai 2008 ; http://tutoriels-data-mining.blogspot.com/2008/05/rgression-pls-comparaison-de-logiciels.html
– [TUTO 12] « Analyse des correspondances discriminante », décembre 2012 ; http://tutoriels-data-mining.blogspot.com/2012/12/analyse-des-correspondances.html

12 Annexes

12.1 Gestion des versions

- Version 1.0. Mise en ligne le 10 mai 2020.

12.2 Fichier de données

Le classeur Excel « **Data_Illustration_Livre_ADL.xlsx** » contient toutes des données d'illustration utilisées dans cet ouvrage. Nous disposons des feuilles suivantes :

- DATA 1
- DATA 2 – TRAIN
- DATA 2 – TEST
- VOTE
- WAVE NOISE
- HEART STATLOG
- WINE
- DIVAY