République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITÉ KASDI MERBAH, OUARGLA

Faculté des Mathématiques et des Sciences de la Matière

DÉPARTEMENT DE MATHÉMATIQUES

Mémoire présenté en vue de l'obtention du Diplôme :

MASTER en Mathématiques

Option :**Probabilitéset statistique**

Par

Elhella Asma

Titre :

Analyse des Correspondances Multiples et Application

Membres du Comité d'Examen :

Dr. BENSAYAH A.Allah	U.**Ouargla**	**Président**
Dr. MEDDI Fatima	U.**Ouargla**	**Encadreur**
Dr. GUERBOUSSA Yacine	U.**Ouargla**	**Examinateur**

Juillet 2023

Dédicace

Je déicide ce travail

ma mère pour sa générosité, et son soutien permanent,

mon père bien-aimé pour son sacrifié,

mes frères et mes sœurs.

mes amis et mes collègues de la promotion 2023

«Mathématiques»

REMERCIEMENTS

Avant tous, je remercie Allah qui ma donné la force, le courage et la patience pour réaliser ce mémoire.

Je remercie aussi mon encadreur Dr. MEDDI Fatima pour la direction de ce mémoire et pour ses conseils précieux qui m'ont aide dans mon travail.

J'adresse mes remerciements aux membres du Jury qui ont accepté d'examiner ce mémoire en lui apportant de l'intérêt.

Je tiens à remercier tous les enseignants participants à mon parcours universitaire

Table des matières

Table des figures

Introduction

L'analyse en correspondance multiple est une méthode statistique utilisée pour explorer les relations entre plusieurs ensembles de variables catégorielles. Egalement connue sous le nom d'ACM (Analyse Factorielle des Correspondances Multiples), elle permet de visualiser et d'interpréter les relations entre les variables qualitatives dans un espace multidimensionnel.

L'analyse en correspondance multiple est souvent utilisée dans des domaines tels que la sociologie, la psychologie, le marketing et la bioinformatique, où les données sont souvent de nature qualitative. Elle vise à identifier les associations et les tendances dans les données, en mettant en évidence les similarités et les différences entre les catégories de variables. L'analyse en correspondance multiple repose sur des calculs de similarité et de dissimilarité entre les catégories de variables, ainsi que sur des techniques de réduction de dimension Décomposition en valeurs singulières. Elle permet de synthétiser des informations complexes et d'obtenir des insights significatifs à partir de données qualitatives. L'analyse en correspondance multiple (ACM) a été développée dans les années 1970 par le statisticien français Jean-Paul Benzécri. Il a introduit cette méthode comme une extension de l'analyse factorielle des correspondances simples (AFC), qui avait été proposée par Jean-Paul Benzecri et Jean-Jacques Droesbeke.

L'ACM a émergé dans le contexte de l'analyse des données catégorielles, où les variables sont de nature qualitative et présentées sous forme de catégories ou de

modalités. Benzécri a reconnu le besoin d'une méthode permettant d'explorer les relations entre plusieurs ensembles de variables catégorielles, au-delà de la simple analyse de deux variables à la fois.

En combinant des concepts de l'analyse factorielle et des techniques d'analyse des tableaux de contingence, Benzécri a développé l'ACM comme une approche permettant de révéler la structure sous-jacente des données qualitatives. Il a proposé une méthode de réduction de dimension qui permet de représenter les données dans un espace multidimensionnel, où les proximités entre les catégories de variables sont préservées. Depuis son introduction, l'ACM a été largement utilisé dans de nombreux domaines de recherche. Les chercheurs l'ont à des problématiques variées telles que l'étude des préférences des consommateurs, l'analyse des opinions politiques, l'exploration des habitudes de consommation, la classification des espèces dans la biologie, et bien d'autres. Au fil du temps, des extensions et des améliorations ont été améliorés à l'ACM, notamment pour prendre en compte des contraintes supplémentaires, telles que des variables quantitatives ou des variables manquantes. Cela a compensé l'applicabilité et la flexibilité de l'ACM dans divers domaines d'étude..

On cite à titre d'exemples quelques références

. Benzécri, JP (1973). L'analyse des données. Tome II : l'analyse des correspondances. Dunod.

. Greenacre, M. (2017). L'analyse des correspondances en pratique. Chapman et Hall/CRC.

. En ligneLe Roux, B., & Rouanet, H. (2010). Analyse des correspondances multiples. SAGE Publications Ltd.

. Escofier, B., & Pagès, J. (1994). Analyse factorielle multiple (paquet AFMULT). Statistiques informatiques et analyse de données, 18(1), 121-140.

. Hirschfeld, HO (1935). Un lien entre corrélation et contingence. Actes de la Cam-

bridge Philosophical Society, 31(4) 520-524

Le mémoire se réparti en trois chapitre :

. **Le premier chapitre** introduit la méthode de l'Analyse en Correspondance Multiple (ACM). Il explore les bases théoriques de l'ACM, en présentant les concepts clés tels que, les statistiques d'association. Le chapitre met également en évidence les principales étapes de l'ACM, notamment le calcul des statistiques de similarité.

. **Le deuxième chapitre** se concentre sur l'objet ACM de FactoMineR, qui est une bibliothèque logicielle populaire utilisée pour effectuer des analyses multivariées en R. Ce chapitre présente les fonctionnalités de base de l'objet ACM, en détaillant les différentes options et paramètres disponibles pour l'analyse. Il explique également comment interpréter les résultats de l'ACM générés par FactoMineR, notamment la signification des axes, des contributions individuelles et des inerties.

. **Le chapitre trois** est consacré à l'application de l'ACM à des données réelles, il met en pratique les connaissances acquises précédemment en appliquant l'ACM à un jeu de données réelles. Ce chapitre décrit le contexte et la nature des données utilisées, ainsi que les objectifs de l'analyse. Il présente les étapes de préparation des données, y compris la structuration des variables des données. Ensuite, il détaille l'application de l'ACM à ces données, en mettant en évidence les résultats clés et en proposant une interprétation des patterns et des regroupements observés.

Chapitre 1

Notions de base sur l'Analyse des Correspondances Multiples

1.1 C'est quoi l'ACM ?

La méthode MCA (Multiple Correspondence Analysis) est une méthode d'analyse des données multidimensionnelles qui permet d'explorer les relations entre plusieurs variables catégorielles en même temps. Elle est souvent utilisée en sciences sociales pour l'analyse de données qualitatives.

La méthode MCA repose explicitement sur la construction d'un tableau de contingence, qui permet de compter le nombre d'individus ayant une combinaison particulière de modalités pour chaque variable étudiée. À partir de ce tableau, l'analyse MCA calcule explicitement une série de vecteurs qui représentent les combinaisons de modalités les plus significatives pour chaque variable. Ces vecteurs sont ensuite utilisés pour projeter les individus et les variables sur un plan factoriel, où les distances entre les points reflètent les relations entre les différentes variables.

1.2 Type de données traitées par l'ACM

L'ACM s'intéresse à des tableaux de données rectangulaires qualitatives où les individus sont en lignes et les variables en colonnes.

	$1 \ldots$	j	$\ldots p$
1			
$\vdots$		$\vdots$	
i	$\cdots$	X_{ij}	$\cdots$
$\vdots$		$\vdots$	
n			

On notera :

- X_{ij} la modalité prise par l'individu i sur la variable j.
- k_j le nombre de modalités de la variable j.
- $k = k_1 + ... + k_p$ le nombre total de modalités.

1.3 Principe de l'ACM

Soient I individus décrits par J variables qualitatives. On considère

$$K = \sum_j k_j$$

et $\{m_1, m_2, ..., m_k\}$ l'ensemble des modalités possibles. On construit les variables qualitatives J telles que la première variable j_1 utilise l'ensemble de modalités $\{m_1, m_2, ..., m_{k_1}\}$, j_2 utilise l'ensemble de modalités $\{k_1 + 1, k_1 + 2, ..., k_1 + k_2\}$, et ainsi de suite.

Le tableau disjonctif complet à I lignes et K colonnes, noté X, est construit de telle sorte que l'intersection de la ligne i et de la colonne k (associée à la modalité k) est

égale à 1 si l'individu i possède la modalité k et 0 sinon.

Il est possible d'inclure une variable quantitative dans l'analyse, à condition de remplacer ses valeurs numériques en plage de valeur, afin de la convertir en variable catégorielle.

Le traitement mathématique du tableau X commence par le calcul de la matrice

$$Z = \frac{X}{IK},$$

s'ensuit le calcul du vecteur r, qui contient la somme en ligne de la matrice Z et enfin, le calcul du vecteur c, qui contient la somme en colonne de la matrice Z.

On prend également en compte les matrices diagonales

$$D_r = diag\,(r)\,,$$
$$D_c = diag\,(c)\,,$$

issues de r et c respectivement. L'étape clé est une décomposition en valeurs singulières de la matrice suivante :

$$M = D_r^{-1/2}\left(Z - rc^t\right)D_c^{-1/2}$$

La décomposition de M donne accès aux matrices P, Δ et Q telles que :

$$M = P\Delta Q^t$$

avec P et Q deux matrices orthogonale, et Δ la matrice diagonale généralisée. On peut montrer que Δest de mêmes dimensions que Z et contient les valeurs singulières ordonnées de la plus grande à la plus petite. Les coefficients diagonaux de Δ^2 sont les valeurs propres de Z et correspondent à l'inertie de chacun des facteurs. Ces

facteurs sont les coordonnées des individus (ligne) ou variables (colonne) sur chacun des axes factoriels. Les coordonnées des individus dans ce nouvel espace vectoriel sont données par la formule suivante :

$$F = D_r^{-1/2} P \Delta$$

La i-ième ligne de F contient les coordonnées du i-ième individu dans l'espace factoriel, tandis que les coordonnées des variables dans le même espace factoriel sont données par :

$$G = D_c^{-1/2} Q \Delta^t.$$

1.4 Domaine d'application de l'ACM

L'ACM est une méthode générale qui s'applique à tout tableau dans lequel un ensemble d'individus est décrit par des variables qualitatives. Elle n'appartient donc pas à un champ disciplinaire particulier. L'ACM (Analyse des Correspondances Multiples) peut être utilisée chaque fois que l'on souhaite analyser des données catégorielles et explorer les relations entre les différentes variables pour obtenir des informations et des insights. l'ACM est une méthode d'analyse statistique largement utilisée dans divers domaines. Voici quelques domaines d'application courants de l'ACM :

1. **Sciences sociales** : L'ACM est souvent utilisée dans les sciences sociales pour analyser des données catégorielles telles que les enquêtes, les questionnaires, les données sociodémographiques, les préférences des consommateurs, les opinions politiques, etc. Elle permet de mettre en évidence les relations entre les différentes catégories de variables et d'explorer la structure des données.
2. **Marketing et études de marché :** L'ACM est utilisée pour analyser les préférences des consommateurs, les comportements d'achat, les caractéristiques

des produits, les segments de marché, etc. Elle permet de comprendre les relations entre les différentes variables catégorielles et d'identifier les profils de consommateurs ou de produits.

3. **Bioinformatique et génétique :** L'ACM est appliquée pour analyser les données génomiques, les profils d'expression génique, les séquences d'ADN, etc. Elle permet de mettre en évidence les relations entre les différentes caractéristiques génétiques et de détecter des motifs ou des associations dans les données biologiques.

4. **Analyse de texte et fouille de données** : L'ACM est utilisée pour analyser des données textuelles, telles que les avis des clients, les résumés d'articles, les tweets, les commentaires, etc. Elle permet de découvrir des associations entre les mots, les thèmes, les opinions, les sentiments, etc., et d'identifier des regroupements ou des similarités entre les documents.

5. **Eudes environnementales** : L'ACM est appliquée pour analyser des données environnementales catégorielles, telles que la présence d'espèces, les types de sol, les variables météorologiques, etc. Elle permet d'explorer les relations entre les variables environnementales et d'identifier des patterns ou des regroupements dans les données.

1.5 Les nuages de points en ACM

Comme toute analyse factorielle, l'ACM peut s'interpréter géométriquement à partir d'un nuage dont les points représentent les lignes (ou les colonnes) du tableau analysé.

1.5.1 Étude des individus

Un individu est représenté par l'ensemble de ses réponses, aussi appelé profil de réponse. L'étude porte sur la variabilité de ces profils de réponse. Comme dans toute analyse factorielle, cette variabilité est décomposée selon une suite de S variables synthétiques (notées $\{F_s; s = 1, ..., s\}$, ces F_s sont retranscrites en tant que colonnes d'une matrice F). Ces variables synthétiques sont quantitatives et permettent des représentations graphiques et l'utilisation de méthodes d'analyse adaptées aux variables quantitatives. Seules les premières colonnes de F sont retenues en général, celles-ci correspondant aux dimensions de l'espace factoriel qui regroupent le plus d'inertie.

1.5.2 Étude des variables

La liaison entre deux variables qualitatives s'étudie au travers des associations entre leurs modalités. Par exemple, un élément de la description de la liaison entre les variables couleur des yeux et couleur des cheveux est : les personnes qui ont les cheveux blonds ont plutôt les yeux bleus. En présence d'un ensemble de variables qualitatives, on cherche donc les associations entre toutes les modalités. On attend de l'ACM une représentation des modalités dans laquelle les modalités qui s'associent entre elles sont proches. Les remarques concernant F restent valables pour G.

1.5.3 Représentation des deux nuages

De façon intuitive, et comme dans toute analyse factorielle, l'ACM consiste à projeter chacun des deux nuages sur une suite d'axes orthogonaux d'inertie maximum (cela correspond mathématiquement à l'étape de décomposition en valeurs singulières). Dans $\mathbb{R}^I$, la quantité maximisée est la moyenne des carrés des rapports de corrélation.

Pour l'axe s, il s'agit de maximiser la valeur

$$\frac{1}{J}\sum_{j}\eta^2\left(F_s,j\right).$$

avec Rapport de corrélation $\eta^2\left(F_s,j\right)$.

Les dimensions de l'ACM peuvent donc être considérées comme des variables synthétiques. Les valeurs de F_s sont les coordonnées des individus sur l'axe de rang s $\left(\text{dans } \mathbb{R}^k\right)$. Il en résulte que les individus qui ont beaucoup de modalités en commun sont aussi proches que possible au contraire des individus qui ont peu de (voire aucune) modalités en commun qui sont aussi séparés que possible.

La combinaison de deux de ces axes fournit une représentation plane, aussi appelée "plan factoriel". En pratique, le premier plan factoriel suffit pour avoir une représentation graphique simple.

1.5.4 Règle d'interprétation

En ACM, la représentation des individus et celle des modalités sont superposables. Ceci est permis par les relations de transition, présentes dans toute analyse factorielle mais qui s'expriment de façon particulièrement simple en ACM.

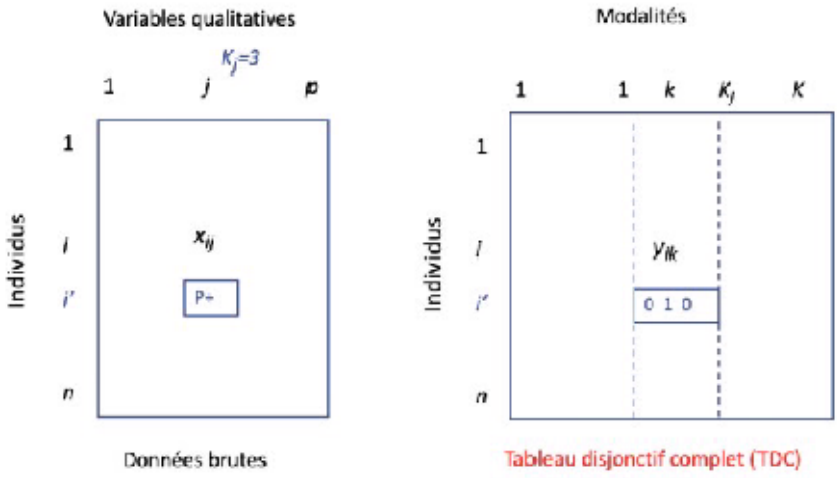
Pour un axe donné, a un coefficient près, un individu est au barycentre des modalités qu'il possède et une modalité est au barycentre des individus qui la possèdent.

Ces relations sont aussi connues sous le nom de propriétés barycentriques.

1.6 Pratique de l'ACM

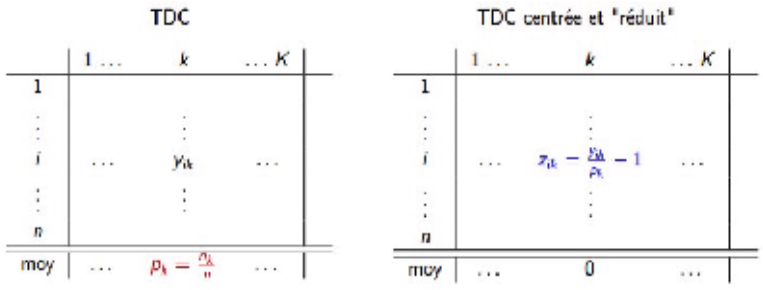
1.6.1 Codage des données (le tableau disjonctif complet)

Un tableau disjonctif complet est une matrice binaire qui représente les données catégorielles, où chaque ligne représente un individu et chaque colonne représente une variable catégorielle, indiquant la présence ou l'absence de chaque catégorie pour chaque individu.



1.6.2 Analyse du nuage des individus

Le tableau disjonctif complet (TDC) est pré-traité avant d'être analysé en ACM.



Le tableau disjoinctif complet est :

– centré : soustraction de la moyenne

$$P_k = \frac{n_k}{n}$$

– "réduit" : division par la fréquence P_k.

Nuage des n individus de $\mathbb{R}^k$:

Tableau disjonctif complet centré réduit

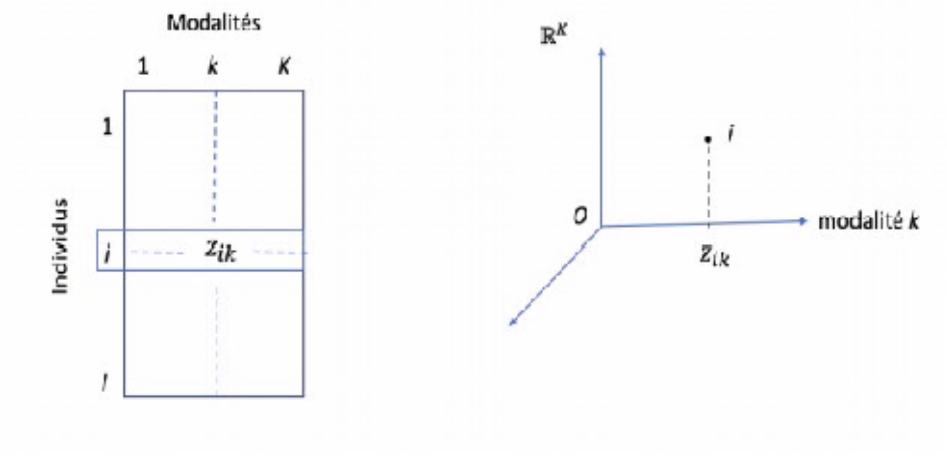


FIG. 1.1 – Nuage des n individus de $\mathbb{R}^k$

Distance entre deux individus :

La proximité entre deux individus se mesure avec une distance Euclidienne pondérée.

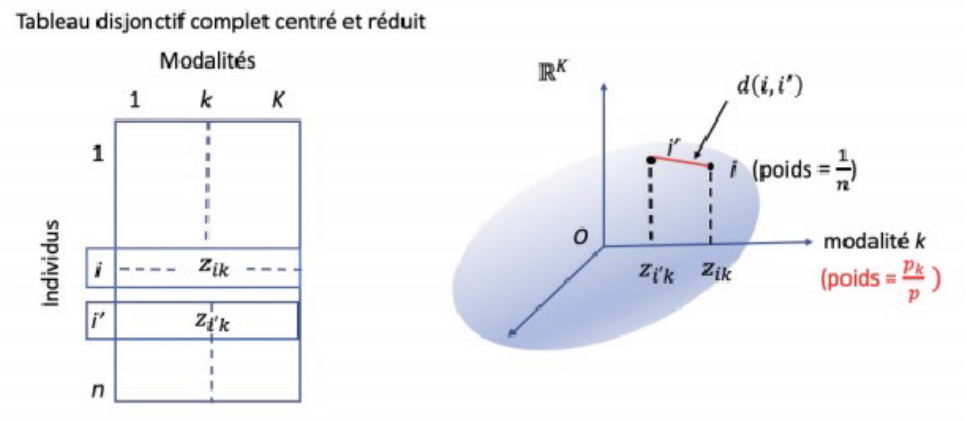


FIG. 1.2 – Distance entre deux individus

$$
\begin{aligned}
d^2(i,i') &= \sum_{k=1}^{K} \frac{P_k}{P} (Z_{ik} - Z_{i'k})^2 \\
&= \sum_{k=1}^{K} \frac{P_k}{P} \left(\frac{Y_{ik}}{P_k} - \frac{Y_{i'k}}{P_k} \right)^2 \\
&= \frac{1}{P} \sum_{k=1}^{K} \frac{1}{P_k} (Y_{ik} - Y_{i'k})^2
\end{aligned}
$$

avec

$$
\begin{aligned}
P_k &= \frac{n_k}{n} \\
Z_k &= \frac{Y_{ik}}{P_k}
\end{aligned}
$$

où pour rappel :

$$
\begin{cases}
Y_{ik} = 1 & \text{si l'individu } i \text{ possède la modalité } k \\
Y_{ik} = 0 & \text{sinon}
\end{cases}
$$

Spécificité de cette distance : prise compte de la rareté des modalités.

$$d^2(i,i') = \frac{1}{P}\sum_{k=1}^{K}\frac{1}{P_k}\left(Y_{ik} - Y_{i'k}\right)^2$$

– 2 individus prennent les mêmes modalités : distance = 0,

– 2 individus ont en commun beaucoup de modalités : distance petite,

– 2 individus sont différent sur un modalité rare (peu fréquente) : plus de poids à

– cette modalité.

Ce qui implique que individus sont différents si ils n'ont pas les mêmes modalités, avec plus de d'importance accordée aux différences sur les modalités rares.

Ajustement du nuage des individus :

L'ACM vise à trouver le sous-espace qui fournit la meilleure représentation des individus.

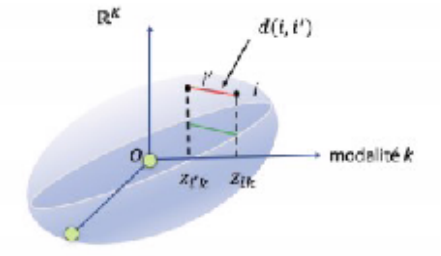


FIG. 1.3 – Ajustement du nuage des individus

– Axes de projection qui conservent au mieux les distances.

– Meilleure représentation de la dispersion (variabilité) des données.

L'inertie des individus en ACM est :

$$\begin{aligned} I(Z) &= \sum_{i=1}^{n} \underbrace{\frac{1}{n} d^2(i,o)}_{\text{inertie de } i} \\ &= \sum_{i=1}^{n} \frac{1}{np} \left(\sum_{k=1}^{K} \frac{Y_{ik}}{P_k} - \frac{1}{n} \right) \\ &= \frac{K}{P} - 1 \end{aligned}$$

car

$$\begin{aligned} d^2(i,o) &= \sum_{k=1}^{K} \frac{P_k}{P} (Z_{ik})^2 \\ &= \sum_{k=1}^{K} \frac{P_k}{P} \left(\frac{Y_{ik}}{P_k} - 1 \right)^2 \\ &= \frac{1}{P} \sum_{k=1}^{K} \frac{Y_{ik}}{P_k} - 1 \end{aligned}$$

⇒En ACM l'inertie dépend du nombre moyen de modalité par variable.

L'inertie du nuage des individus en ACM se réecrit :

–

$$I(Z) = \sum_{k=1}^{K} \underbrace{\frac{1}{P}(1 - P_k)}_{\text{contribution de } k}$$

⇒Les modalités rares contribuent plus à l'inertie.

–

$$I(Z) = \sum_{j=1}^{p} \underbrace{\frac{1}{P}(K_j - 1)}_{\text{contribution de } j}$$

⇒Les variables avec beaucoup de modalités contribuent plus à l'inertie.

Interprétation globale des axes de projection des individus en ACM :

En ACM l'axe s est orthogonal à tout axe $t(t < s)$ et est le plus lié aux variables qualitatives au sens du η^2 :

$$F_{.s} = \underset{F \in \mathbb{R}^n}{\arg\max} \sum_{j=1}^{p} \eta^2 \left(F, X_{.j}\right)$$

$\Rightarrow$ $F_{.s}$ est une nouvelle variable synthétique quantitative résumant au mieux les variables qualitatives initiales.

1.6.3 Analyse du nuage des modalités

Nuage de K modalités de $\mathbb{R}^n$:

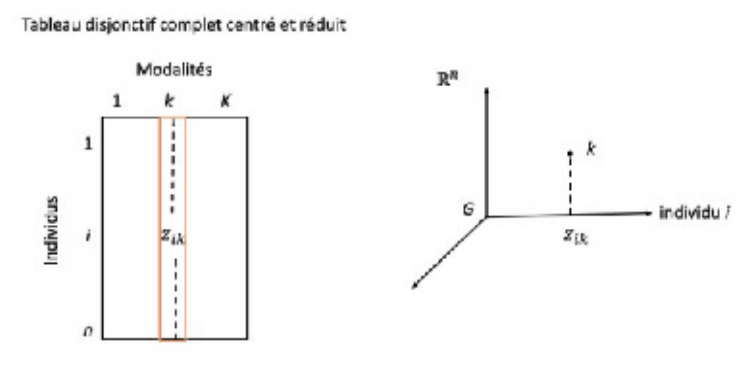


FIG. 1.4 – Nuage de K modalités de $\mathbb{R}^n$

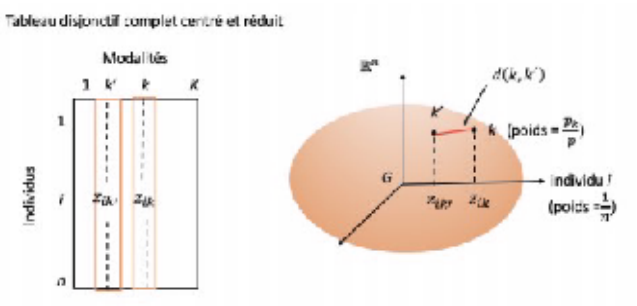


FIG. 1.5 – Distance entre deux modalités

Distance entre deux modalités :

$$\begin{aligned} d^2(k,k') &= \sum_{i=1}^{n} (Z_{ik} - Z_{ik'})^2 \\ &= \sum_{i=1}^{n} \left(\frac{Y_{ik}}{P_k} - \frac{Y_{ik'}}{P_{k'}} \right)^2 \\ &= \frac{p_k + p_{k'} - 2p_k p_{k'}}{p_k p_{k'}} \end{aligned}$$

où pour rappel :

$$\begin{cases} Y_{ik} = 1 & \text{si l'individu } i \text{ possède la modalité } k \\ Y_{ik} = 0 & \text{sinon.} \end{cases}$$

Ajustement du nuage des modalités :

L'ACM vise à trouver le sous-espace qui fournit la meilleure représentation des modalités.

– Axes de projection qui conservent au mieux les distances

– Meilleure représentation de la dispersion (variabilité) des points.

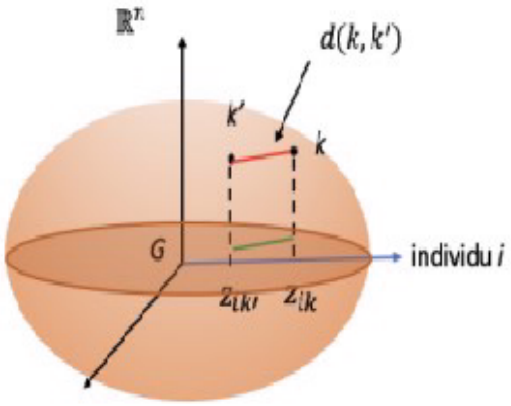


FIG. 1.6 – Ajustement du nuage des modalités

1.6.4 Interprétation des résultats

Qualité de la réduction de dimension en ACM.

– La valeur propre λ_s (l'inertie de l'axe s) est la moyenne des rapports de corrélation entre la variable synthétique $F_{.s}$ et les variables qualitatives initiales :

$$\lambda_s = \frac{1}{p}\sum_{j=1}^{p} \eta^2\left(F, X_{.j}\right)$$

$\Rightarrow \lambda_s \in [0;1]$.

– L'inertie totale est :

$$I\left(Z\right) = \lambda_1 + ... + \lambda_r = \frac{K-P}{P},$$

où

$$r = \min\left(n-1, k-p\right)$$

est le nombre de valeurs propres non nulles.

$\Rightarrow \frac{1}{p}$= valeur propre moyenne (inertie moyenne) par axe (si $n < k-p$) .

$\Rightarrow$Interpréter les dimensions d'inertie supérieures à $\frac{1}{p}$.

– Individus vivent dans $\mathbb{R}^{k-p} \Rightarrow$ pourcentages d'inertie faibles.

Contributions et qualité de représentation en ACM.

– Contribution et $\cos^2$ pour les individus et les modalités

⇒ Les modalités extrêmes ne contribuent pas nécessairement beaucoup (cela dépend de leur fréquence).

⇒ Les $\cos^2$ sont petits...ce qui est attendu car beaucoup de dimensions.

– Constribution absolue d'une variable :

$$Ctr\left(j\right) = \sum_{k=1}^{k_j} CTR\left(k\right) = \frac{\eta^2\left(F_{.s}, X_{.j}\right)}{p}$$

– Constribution relative d'une variable :

$$Ctr\left(j\right) = \frac{\eta^2\left(F_{.s}, X_{.j}\right)}{p\lambda_s}$$

Chapitre 2

Outils Mathématiques et Informatiques pour l'exécution de l'Analyse des Correspondances Multiples

Dans les parties qui suivent, nous présenterons la partie application de l'ACM. Tous nos codes et visualisations sont exécutés sur le logiciel RStudio, pour cette raison nous efforçons de présenter quelques bibliothèques indispensables pour la réalisation d' l'ACM. Ainsi que la méthode svd en appliquant nos codes sur un cas pratique

2.1 Bibliothèques indispensables dans RStudio

2.1.1 FactoMinR

La bibliothèque FactoMinR est une bibliothèque R spécialisée dans l'analyse factorielle, une technique statistique utilisée pour explorer et réduire la dimensionnalité

des données. FactoMinR offre une large gamme de méthodes pour effectuer des analyses factorielles, telles que l'analyse en composantes principales (PCA), l'analyse des correspondances multiples (MCA) et d'autres variantes.

Installation de FactoMinR

. Avant de commencer, assurez-vous d'installer FactoMinR en exécutant la commande $install.packages("FactoMinR")$ dans R.

Chargement de FactoMinR

. Pour commencer à utiliser FactoMinR, vous devez charger la bibliothèque en utilisant la commande $library(FactoMinR)$.

Analyse de MCA FactoMinR

. L'analyse des correspondances multiples (MCA) est une méthode d'analyse factorielle adaptée aux variables catégorielles.

. Pour effectuer une MCA avec FactoMinR, utilisez la fonction MCA() en spécifiant la matrice de données et les options nécessaires, telles que la pondération des variables ou l'inclusion des individus supplémentaires.

. La fonction $summary()$ peut également être utilisée pour afficher les résultats de la MCA, y compris les contributions des variables et des individus, ainsi que les graphiques.

2.1.2 FactoExtra

La bibliothèque FactoExtra est une bibliothèque R spécialisée dans l'analyse factorielle, une technique statistique utilisée pour explorer et réduire la dimensionnalité

des données. FactoExtra fournit des outils avancés pour effectuer des analyses factorielles, tels que l'analyse en composantes principales (PCA), l'analyse des correspondances multiples (MCA) et d'autres méthodes.

Installation de FactoExtra

. Avant de commencer, assurez-vous d'installer FactoExtra en exécutant la commande $install.packages("FactoExtra")$ dans R.

Chargement de FactoExtra

. Pour commencer à utiliser FactoExtra, vous devez charger la bibliothèque en utilisant la commande $library(FactoExtra)$.

Analyse de MCA avec FactoExtra

. L'analyse des correspondances multiples (MCA) est une méthode d'analyse factorielle adaptée aux variables catégorielles.

. Pour effectuer une MCA avec FactoExtra, utilisez la fonction $MCA()$ en spécifiant la matrice de données et les options nécessaires, telles que la pondération des variables ou la méthode de rotation des axes.

. FactoExtra propose plusieurs fonctions de visualisation pour explorer les résultats de la MCA, comme $fviz\_mca\_ind()$ pour visualiser les individus, $fviz\_mca\_var()$ pour visualiser les variables et $fviz\_cos2()$ pour visualiser les carrés des cosinus.

2.1.3 Ggfortify

La bibliothèque ggfortify est une bibliothèque R qui étend la puissance de ggplot2 pour la visualisation de modèles statistiques. Elle offre des fonctionnalités permettant

de visualiser facilement les résultats de divers modèles, tels que les modèles de régression linéaire, les modèles de régression logistique, les modèles de séries temporelles et bien d'autres encore.

Installation de Ggfortify

Avant de commencer, assurez-vous d'installer ggfortify en exécutant la commande $install.packages("ggfortify")$ dans R.

Chargement de Ggfortify

Pour commencer à utiliser ggfortify, vous devez charger la bibliothèque en utilisant la commande $library(ggfortify)$.

Analyse de MCA avec Ggfortify

La bibliothèque ggfortify n'est pas spécifiquement conçue pour l'Analyse des Correspondances Multiples (ACM). Cependant, elle peut être utilisée pour visualiser les résultats de l'ACM en combinant ggfortify avec d'autres bibliothèques spécifiques à l'ACM, telles que FactoMineR.

Voici comment vous pouvez utiliser ggfortify pour visualiser les résultats de l'ACM :

Effectuez l'ACM avec FactoMineR

. Utilisez la bibliothèque $FactoMineR$ pour effectuer l'ACM sur vos données catégorielles.

. La fonction$MCA()$ de $FactoMineR$ peut être utilisée pour effectuer l'ACM, en spécifiant la matrice de données et les options appropriées.

Convertissez les résultats de l'ACM en objet ggplot2

. Utilisez la fonction $autoplot()$ de ggfortify pour convertir les résultats de l'ACM en un objet $ggplot2$ pouvant être utilisé pour la visualisation.

. Par exemple, vous pouvez utiliser $autoplot(acm\_object, data = your\_data)$ pour convertir les résultats de l'ACM en un objet $ggplot2$, en spécifiant vos données originales.

Personnalisez la visualisation avec ggplot2

. Une fois que vous avez converti les résultats de l'ACM en un objet $ggplot2$, vous pouvez utiliser les fonctionnalités de $ggplot2$ pour personnaliser la visualisation selon vos besoins.

. Par exemple, vous pouvez ajouter des titres, modifier les couleurs, ajuster les échelles, ajouter des étiquettes d'axes, etc.

Explorez les graphiques spécifiques à l'ACM

. Utilisez les fonctionnalités spécifiques à l'ACM de FactoMineR pour ajouter des éléments supplémentaires à la visualisation.

. Par exemple, vous pouvez utiliser les fonctions de FactoMineR pour ajouter les contributions des variables et des individus, visualiser les liens entre les catégories, ou afficher les cercles de corrélation.

2.1.4 Corrplot

La bibliothèque corrplot est une bibliothèque R qui permet de visualiser facilement les matrices de corrélation. Elle offre des fonctionnalités pour représenter graphiquement les corrélations entre les variables d'un ensemble de données.

Installation de Corrplot

Avant de commencer, assurez-vous d'installer corrplot en exécutant la commande $install.packages("corrplot")$ dans R.

Chargement de Corrplot

Pour commencer à utiliser corrplot, vous devez charger la bibliothèque en utilisant la commande $library(corrplot)$.

Analyse de MCA avec Corrplot

La bibliothèque corrplot n'est pas spécifiquement conçue pour être utilisée avec l'Analyse des Correspondances Multiples (ACM). Cependant, elle peut être utilisée pour visualiser les matrices de corrélation qui peuvent être obtenues à partir des résultats de l'ACM.

Voici comment vous pouvez utiliser corrplot avec l'ACM :

Obtenez la matrice de corrélation à partir des résultats de l'ACM

. Les résultats de l'ACM obtenus à partir de FactoMineR comprennent souvent une matrice de corrélation, qui représente les corrélations entre les catégories des variables analysées.

. Utilisez la fonction $correlation()$ de FactoMineR pour extraire la matrice de corrélation à partir des résultats de l'ACM. Par exemple, $cor\_matrix < -correlation(acm\_c$

Utilisez corrplot pour visualiser la matrice de corrélation

. Avec la matrice de corrélation obtenue, vous pouvez utiliser corrplot pour visualiser les corrélations entre les catégories des variables.

. Utilisez la fonction $corrplot()$ de corrplot en spécifiant la matrice de corrélation. Par exemple, $corrplot(cor\_matrix, method = "color")$.

. corrplot propose différentes méthodes de visualisation, telles que les graphiques de corrélation circulaire, les graphiques de corrélation de type heatmap et les graphiques de corrélation de type cluster. Choisissez la méthode qui convient le mieux à votre analyse.

Personnalisez la visualisation avec corrplot

. Utilisez les options disponibles dans corrplot pour personnaliser la visualisation de la matrice de corrélation. Par exemple, vous pouvez modifier les couleurs, les étiquettes d'axe, les seuils de corrélation, etc.

2.1.5 Scatterplot3d

La bibliothèque scatterplot3d est une bibliothèque R qui permet de créer des graphiques de dispersion en trois dimensions. Elle offre des fonctionnalités pour visualiser des données tridimensionnelles et explorer les relations entre plusieurs variables.

Installation de Scatterplot3d

Avant de commencer, assurez-vous d'installer scatterplot3d en exécutant la commande $install.packages("scatterplot3d")$ dans R.

Chargement de Scatterplot3d

Pour commencer à utiliser scatterplot3d, vous devez charger la bibliothèque en utilisant la commande $library(scatterplot3d)$.

Analyse de MCA avec Scatterplot3d

La bibliothèque scatterplot3d n'est pas spécifiquement conçue pour être utilisée avec l'Analyse des Correspondances Multiples (ACM). Cependant, elle peut être utilisée pour visualiser les résultats de l'ACM en affichant des graphiques de dispersion en trois dimensions.

Voici comment vous pouvez utiliser scatterplot3d avec l'ACM

Effectuez l'ACM avec FactoMineR

. Utilisez la bibliothèque FactoMineR pour effectuer l'ACM sur vos données catégorielles.

. Utilisez la fonction $MCA()$ de FactoMineR pour effectuer l'ACM, en spécifiant la matrice de données et les options appropriées.

Obtenez les coordonnées des individus à partir des résultats de l'ACM

. Les résultats de l'ACM obtenus à partir de FactoMineR contiennent les coordonnées des individus dans l'espace factoriel.

. Utilisez la fonction $get\_coord()$ de FactoMineR pour extraire les coordonnées des individus. Par exemple, $coord < -get\_coord(acm\_object, choice = "ind")$.

Créez un graphique de dispersion en 3D avec scatterplot3d

. Utilisez la fonction $scatterplot3d()$ de scatterplot3d en spécifiant les coordonnées des individus dans les trois dimensions. Par exemple,
$scatterplot3d(coord[,1], coord[,2], coord[,3])$.

. Vous pouvez également personnaliser le graphique en utilisant les options disponibles dans scatterplot3d, telles que les couleurs, les étiquettes des axes, les titres, etc.

Ajoutez des informations supplémentaires au graphique

. scatterplot3d permet d'ajouter des informations supplémentaires au graphique en 3D.

. Par exemple, vous pouvez ajouter des étiquettes aux individus en utilisant la fonction $text()$ et spécifier les options appropriées. Vous pouvez également ajouter des ellipses de confiance en utilisant la fonction $ellipse()$ pour représenter les groupes d'individus.

Exploration interactive avec le graphique en 3D

. scatterplot3d offre une fonctionnalité d'exploration interactive qui permet de faire pivoter le graphique en 3D pour visualiser les individus sous différents angles.

. Utilisez la souris pour faire pivoter le graphique en 3D et explorer les relations entre les individus et les variables.

2.1.6 Plot3D

La bibliothèque plot3D est une bibliothèque R qui offre des fonctionnalités pour la visualisation de données en trois dimensions. Elle permet de créer des graphiques en 3D interactifs, offrant une perspective plus complète sur les données et permettant d'explorer les relations entre plusieurs variables.

Installation de Plot3D

Avant de commencer, assurez-vous d'installer plot3D en exécutant la commande $install.packages("plot3D")$ dans R.

Chargement de Plot3D

Pour commencer à utiliser plot3D, vous devez charger la bibliothèque en utilisant la commande $library(plot3D)$.

Analyse de MCA avec Plot3D

La bibliothèque plot3D n'est pas spécifiquement conçue pour être utilisée avec l'Analyse des Correspondances Multiples (ACM). Cependant, elle peut être utilisée pour visualiser les résultats de l'ACM en affichant des graphiques en 3D pour les coordonnées des individus ou les variables.

Effectuez l'ACM avec FactoMineR

. Utilisez la bibliothèque FactoMineR pour effectuer l'ACM sur vos données catégorielles.

. Utilisez la fonction $MCA()$ de FactoMineR pour effectuer l'ACM, en spécifiant la matrice de données et les options appropriées.

Obtenez les coordonnées des individus ou des variables à partir des résultats de l'ACM

. Les résultats de l'ACM obtenus à partir de FactoMineR contiennent les coordonnées des individus ou des variables dans l'espace factoriel.

. Utilisez la fonction $get\_coord()$ de FactoMineR pour extraire les coordonnées des individus ou des variables. Par exemple, $coord\_individus <- get\_coord(acm\_object, ch$ "ind") pour les coordonnées des individus ou $coord\_variables <- get\_coord(acm\_object, ch$ "var") pour les coordonnées des variables.

Créez un graphique en 3D avec plot3D

. Utilisez les fonctions disponibles dans plot3D pour créer des graphiques en 3D à partir des coordonnées des individus ou des variables.

. Par exemple, utilisez la fonction $scatter3D()$ pour créer un graphique de dispersion en 3D des coordonnées des individus ou des variables. Spécifiez les coordonnées x, y et z correspondantes.

. Utilisez la fonction surface3D() pour créer une surface en 3D à partir des coordonnées des individus ou des variables.

. Utilisez la fonction parametric3D() pour tracer des courbes ou des surfaces paramétriques en 3D basées sur les coordonnées des individus ou des variables.

Personnalisez le graphique en 3D

. plot3D offre différentes options de personnalisation pour ajuster l'apparence du graphique en 3D.

. Par exemple, vous pouvez modifier les couleurs, les étiquettes d'axe, les titres, les échelles, etc.

. Consultez la documentation de plot3D pour explorer les options de personnalisation disponibles.

Ajoutez des informations supplémentaires au graphique en 3D

. plot3D permet d'ajouter des éléments supplémentaires à votre graphique en 3D pour une meilleure compréhension des données.

. Par exemple, vous pouvez ajouter des étiquettes aux points en utilisant la fonction text3D(), ajouter des lignes de régression en utilisant la fonction plane3D() ou ajouter des légendes en utilisant la fonction legend3D().

Exploration interactive avec le graphique en 3D

. plot3D offre une fonctionnalité d'exploration interactive qui permet de faire pivoter le graphique en 3D pour visualiser les données sous différents angles.

. Utilisez la souris pour faire pivoter le graphique en 3D et explorer les relations entre les individus et les variables

2.2 Réduction de dimensionnalité

Nous allons présenter dans ce qui suit un cas pratique pour expliquer les étapes et les détails mathématiques de l'ACM dans un aspect informatique. Pour cette raison nous allons proposer un tableau de six individus et 3 variables catégorielles (qualitatives) avec 9 catégories (modalités), une taille réduite nous permettra de facilité les illustration au fur et au mesure avec les résultats théoriques

Nous allons commencer par expliquer la méthode de la décomposition en valeurs singulière (sdv) utilisée dans l'ACM, aussi nous présenterons dans la dernière section du chapitre l'objet ACM avec affichage des résultats et projection graphiques

Le tableau suivant contient les préférences de nourriture de 6 individus en fruit, légume et viande

	Fruit	Légume	Viande
ind1	Pomme	Haricot	Cheval
ind2	Poire	Haricot	Cheval
ind3	Orange	Carotte	Mouton
ind4	Pomme	Carotte	Mouton
ind5	Poire	Epinard	Bœuf
ind6	Orange	Epinard	Bœuf

2.2.1 Méthode de décomposition en valeurs singulières

La méthode de la décomposition en valeurs singulières (SVD) est une technique mathématique importante utilisée dans de nombreux domaines, tels que l'algèbre linéaire, le traitement du signal, l'apprentissage automatique, la compression de données et bien d'autres encore. Elle permet de décomposer une matrice en plusieurs composants, ce qui facilite l'analyse et la manipulation des données. Nous allons explorer les principes fondamentaux de la SVD et examiner son application dans différents domaines.

Définition de la décomposition en valeurs singulières

La décomposition en valeurs singulières est une factorisation matricielle d'une matrice A de dimensions mxn, où m représente le nombre de lignes et$n$ le nombre de colonnes. La SVD permet d'exprimer cette matrice A sous la forme $A = UDV^t$, où U est une matrice orthogonale de dimension mxm, D est une matrice diagonale de dimension mxn contenant les valeurs singulières, et V^t est la transposée d'une matrice orthogonale V de dimension nxn.

Calcul des valeurs singulières

Les valeurs singulières de la matrice A sont les racines carrées des valeurs propres de la matrice A^tA ou de la matrice AA^t Ces valeurs sont généralement ordonnées en ordre décroissant. Les valeurs singulières représentent l'importance relative des différentes composantes de la matrice.

Interprétation géométrique de la SVD

Géométriquement, la SVD permet de décomposer une transformation linéaire en une série de transformations élémentaires : une rotation, une mise à l'échelle et une

autre rotation. Les matrices U et V représentent ces transformations de rotation, tandis que la matrice D représente la mise à l'échelle.

2.2.2 Applications de la SVD en ACM

La méthode de la décomposition en valeurs singulières (SVD) est largement utilisée dans l'analyse des correspondances multiples (ACM). Voici comment la SVD est utilisée dans l'ACM :

Construction de la matrice de données

. Tout d'abord, les données catégorielles sont représentées sous forme de tableau disjonctif complet où les lignes représentent les individus et les colonnes représentent les catégories des variables.

. Une matrice de données est construite à partir de ce tableau disjonctif, où les valeurs sont généralement des , des codes binnaires. (0 si la gatégorie n'est pas vérifiée chez l'individu et 1 si la catégories est vérifiés chez l'individu)

Standardisation de la matrice de données

. La matrice de données est standardisée pour éliminer les effets de taille ou de dispersion. Cela se fait en soustrayant la moyenne de chaque colonne et en divisant par l'écart type.

Application de la SVD à la matrice standardisée

. La SVD est appliquée à la matrice de données standardisée. Cela permet de décomposer la matrice en trois matrices :U, Σ et V^t.

. U représente les coordonnées des individus dans un espace factoriel. Chaque individu est représenté par un vecteur de valeurs singulières.

. Σ est une matrice diagonale contenant les valeurs singulières, qui représentent l'importance relative des axes factoriels.

. V^t représente les coordonnées des catégories de variables dans l'espace factoriel. Chaque catégorie de variable est représentée par un vecteur de valeurs singulières.

Résultats pour l'ACM

. Les axes factoriels, également appelés composantes principales, sont utilisés pour représenter graphiquement les individus et les catégories de variables dans un espace de dimensions réduites.

. L'ACM permet d'analyser la structure des données, d'identifier les relations entre les variables catégorielles et de détecter des regroupements ou des associations.

. Les distances entre les individus dans l'espace factoriel peuvent être utilisées pour effectuer des analyses de similarité ou de dissimilarité, telles que des classifications ou des analyses de regroupement.

En utilisant la SVD dans l'analyse des correspondances multiples, il est possible de réduire la dimensionnalité des données catégorielles, de visualiser les relations entre les variables et les individus, et de détecter des patterns ou des associations significatives. Cela facilite l'interprétation et la compréhension des données catégorielles complexes

2.2.3 Exécution de SVD

Dans cette partie, nous allons exécuter le code R de SVD sur le cas des 6 individus mentionné en haut

Algorithme - svd

Cet algorithme effectue une codification des variables catégorielles du tableau cité plus haut et utilise la décomposition en valeurs singulières (SVD) pour analyser les relations entre les catégories. Ensuite, il présente les résultats et trace un diagramme de variance expliquée pour illustrer l'importance relative des caractéristiques principales. L'algorithme implémenté dans le code fourni peut être divisé en plusieurs étapes :

1. **Définition des vecteurs :** Les vecteurs Fruit, Légume, et Viande sont créés pour désigner les catégories correspondantes.
2. **Codage indicateur binaire des variables :** La variable categories est créée en fusionnant les valeurs uniques des vecteurs Fruit, Légumeet Viande. Ensuite, une table de disjonctive complet appelée disj_tableest est construite En utilisant une boucle for, nous parcourons chaque catégorie et nous utilisons une expression logique (Fruit == categories[i] | Légume == categories[i] | Viande == categories[i]) pour attribuer la valeur 1 à la cellule correspondante si la catégorie est présente pour un individu, et 0 sinon. Le résultat est stocké dans la table disjonctive disj_table.
3. Enfin, nous attribuons les noms de colonnes à la table disjonctive en utilisant colnames(disj_table) <- categories, ce qui permet d'identifier chaque colonne avec la catégorie correspondante.
4. **Décomposition en valeurs singulières (SVD) :** La table disj_tableest normalisée à l'aide de la fonction scale(), puis la décomposition en valeurs singulières est effectuée en utilisant la fonction svd(). Les résultats de la décomposition sont stockés dans les variables X_scaled (table standardisée), svd_result(résultat de la décomposition), U(matrice U des vecteurs singuliers), V(matrice V des vecteurs singuliers), sigma(vecteur des valeurs singulières) et

cordonnées_individus(matrice U des vecteurs singuliers, qui est ensuite redondant).

5. **Calcul de la variance expliquée :** Les pourcentages de variance expliqués sont calculés en utilisant les valeurs singulières (sigma) et sont stockés dans la variable variance_expliquee.
6. **Affichage des résultats :** Les différentes matrices et vecteurs (disj_table, U, V, sigma, cordonnées_individus et variance_expliquee) sont affichés à l'aide de la fonction print().
7. **Tracé du diagramme de variance expliqué :** Un diagramme en barres de la variance expliquée est tracé à l'aide de la fonction barplot(), en utilisant les pourcentages de variance expliqués (variance_expliquee). L'axe x représente les caractéristiques principales, l'axe y représente le pourcentage de variance expliquée, et le titre et les étiquettes des axes sont spécifiés.

Code R - svd

1. # Vecteurs

. Fruit = c("Pomme", "Poire", "Orange", "Pomme", "Poire", "Orange")

. Légume = c("Haricot", "Haricot", "Carotte", "Carotte", "Epinard", "Epinard")

. Viande = c("Cheval", "Cheval", "Mouton", "Mouton", "Bœuf", "Bœuf")

2. # Codage

. categories <- unique(c(Fruit, Légume, Viande))

. num_categories <- length(categories)

. disj_table <- matrix(0, nrow = length(Fruit), ncol = num_categories)

. for (i in 1 :num_categories) {

. disj_table[, i] <- as.numeric(Fruit == categories[i] | Légume == categories[i] | Viande == categories[i])

. }

. colnames(disj_table) <- categories

3. # svd sur disj_table

. X_scaled <- scale(disj_table)

. svd_result <- svd(X_scaled)

. U <- svd_result$u

. V <- svd_result$v

. sigma <- svd_result$d

. cordonnées_individus <- svd_result$u

. variance_expliquee <- (svd_result$d^2 / sum(svd_result$d^2)) * 100

4. # Affichge des résultats

. print(disj_table)

. class(disj_table)

. dim(disj_table)

. print(U)

. print(V)

. print(sigma)

. print(cordonnées_individus)

. print(variance_expliquee)

5. # Tracer le diagramme de variance expliquée

. barplot(variance_expliquee, col="pink",

xlab = "Composantes principales",

ylab = "Pourcentage de variance expliquée",

main = "Diagramme de Variance Expliquée")

Sorties - svd

1. Tableau disjonctif complet

```
> print(disj_table)
     Pomme Poire Orange Haricot Carotte Epinard Cheval Mouton Bœuf
[1,]     1     0      0       1       0       0      1      0    0
[2,]     0     1      0       1       0       0      1      0    0
[3,]     0     0      1       0       1       0      0      1    0
[4,]     1     0      0       0       1       0      0      1    0
[5,]     0     1      0       0       0       1      0      0    1
[6,]     0     0      1       0       0       1      0      0    1
> class(disj_table)
[1] "matrix" "array"
> dim(disj_table)
[1] 6 9
```

2. Matrice U, V et Σ

```
> print(U)
              [,1]          [,2]          [,3]          [,4]      [,5]       [,6]
[1,] -1.138455e-16 -5.773503e-01  5.773503e-01  6.940010e-17 0.4151851 -0.4011916
[2,] -2.886751e-01 -5.000000e-01 -5.000000e-01 -2.886751e-01 0.4011916  0.4151851
[3,]  5.000000e-01  2.886751e-01 -2.886751e-01 -5.000000e-01 0.4151851 -0.4011916
[4,]  5.773503e-01  1.453708e-16  9.950626e-17  5.773503e-01 0.4011916  0.4151851
[5,] -5.000000e-01  2.886751e-01 -2.886751e-01  5.000000e-01 0.4151851 -0.4011916
[6,] -2.886751e-01  5.000000e-01  5.000000e-01 -2.886751e-01 0.4011916  0.4151851
> print(V)
            [,1]        [,2]        [,3]        [,4]      [,5]       [,6]
 [1,]  0.26540841 -0.26540841  0.51272967  0.51272967 0.4121890 -0.4037310
 [2,] -0.36255462 -0.09714622 -0.70040175  0.18767208 0.4121890 -0.4037310
 [3,]  0.09714622  0.36255462  0.18767208 -0.70040175 0.4121890 -0.4037310
 [4,] -0.13270420 -0.49525883  0.06869275 -0.25636483 0.3139022  0.3548261
 [5,]  0.49525883  0.13270420 -0.25636483  0.06869275 0.2853870  0.2893778
 [6,] -0.36255462  0.36255462  0.18767208  0.18767208 0.2853870  0.2893778
 [7,] -0.13270420 -0.49525883  0.06869275 -0.25636483 0.2568718  0.2239294
 [8,]  0.49525883  0.13270420 -0.25636483  0.06869275 0.2853870  0.2893778
 [9,] -0.36255462  0.36255462  0.18767208  0.18767208 0.2853870  0.2893778
> print(sigma)
[1] 4.212504e+00 4.212504e+00 2.180553e+00 2.180553e+00 7.460165e-16 3.363690e-16
```

3. Cordonnées des individus dans les composantes principales

```
> print(cordonnées_individus)
              [,1]          [,2]          [,3]          [,4]      [,5]       [,6]
[1,] -1.138455e-16 -5.773503e-01  5.773503e-01  6.940010e-17 0.4151851 -0.4011916
[2,] -2.886751e-01 -5.000000e-01 -5.000000e-01 -2.886751e-01 0.4011916  0.4151851
[3,]  5.000000e-01  2.886751e-01 -2.886751e-01 -5.000000e-01 0.4151851 -0.4011916
[4,]  5.773503e-01  1.453708e-16  9.950626e-17  5.773503e-01 0.4011916  0.4151851
[5,] -5.000000e-01  2.886751e-01 -2.886751e-01  5.000000e-01 0.4151851 -0.4011916
[6,] -2.886751e-01  5.000000e-01  5.000000e-01 -2.886751e-01 0.4011916  0.4151851
```

4. Pourcentages de variance exploquée

```
> print(variance_expliquee)
[1] 3.943376e+01 3.943376e+01 1.056624e+01 1.056624e+01 1.236757e-30 2.514313e-31
```

5. Diagrammes de variance expliqueé

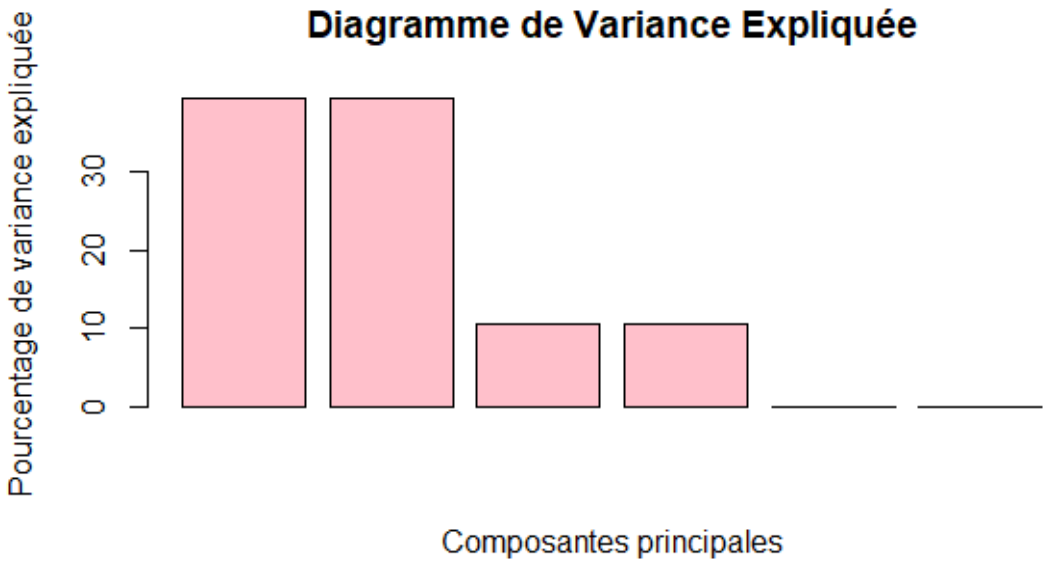


FIG. 2.1 – Diagrammes de variance expliqueé

6. **Commentaire** : Nous remarquons que le tableau des données d'origine avait trois colonnes de variable catégorielles avec neuf catégories, se sont transformées en tableau disjonctif de neuf colonnes contenant les neuf catégories comme

variables avec des valeurs binaires (0,1). L'utilisation de la décomposition en valeur singulier nous a permet d'avoir uniquement six coordonnées pour chaque individus au lieu de neuf. les composantes principales qui contiennent plus d'information sur les individus sont les deux premières comme mentionné sur le diagramme $39,13\%$ et $39,13\%$. (L'égalité ici n'est pas significative mais au hasard), donc au total ces deux compsantes contient 78.26% de l'information sur les individus. Comme on voit que les pourcentages des deux dernières composantes sont presque nulle avec $1.236757e-30$ et $2.514313e-31$ ce qui explique qu'elles sont presque négligeables en terme de contenance d'informations sur les individus

2.3 L'objet MCA()

Le package FactiminR en R fournit des outils puissants pour effectuer une ACM et explorer les relations entre les variables catégorielles. Avant de commencer, spécifiez-vous d'avoir installé le package FactominR comme on a bien précisé dans la section précedente

2.3.1 Déroulement de MCA()

Déscription

Effectue une analyse des correspondances multiples (MCA) avec des individus supplémentaires, des variables quantitatives supplémentaires et des variables catégorielles supplémentaires. Effectue également une analyse spécifique des correspondances multiples avec des catégories supplémentaires et des variables catégorielles supplémentaires. Les valeurs manquantes sont traitées comme un niveau supplémentaire, les catégories qui sont rares peuvent être ventilées

Usage

MCA(X, ncp = 5, ind.sup = NULL, quanti.sup = NULL,

quali.sup = NULL, excl=NULL, graph = TRUE,

level.ventil = 0, axes = c(1,2), row.w = NULL,

method="Indicator", na.method="NA", tab.disj=NULL)

Arguments

. **X** : un data frame avec n lignes (individus) et p colonnes (variables catégorielles)

. **ncp** : nombre de dimensions conservées dans les résultats (par défaut 5)

. **ind.sup** : un vecteur indiquant les indices des individus supplémentaires

. **quanti.sup** : un vecteur indiquant les indices des variables supplémentaires quantitatives

. **quali.sup** : un vecteur indiquant les indices des variables supplémentaires catégorielles

. **excl** : vecteur indiquant les index des catégories "junk" (la valeur par défaut est NULL), il peut s'agir d'un vecteur des noms des catégories ou d'un vecteur des index dans la table de données disjonctives

. **graph** : booléen, si TRUE un graphique est affiché

. **level.ventil** : une proportion correspondant au niveau sous lequel la catégorie est ventilée ; par défaut, 0 et aucune ventilation n'est effectuée

. **axes** : un vecteur de longueur 2 spécifiant les composantes à tracer

. **row.w** : un poids de ligne facultatif (par défaut, un vecteur de 1 pour des poids de ligne uniformes) ; les poids sont donnés uniquement pour les individus actifs

. **method** : une chaîne correspondant au nom de la méthode utilisée : "Indicator" (par défaut) est le CA sur la matrice Indicator, "Burt" est le CA sur la table Burt. Pour Burt et l'Indicateur, le graphe des individus et le graphe des catégories sont donnés

. **na.method** : une chaîne correspondant au nom de la méthode utilisée s'il manque des valeurs ; les méthodes disponibles sont "NA" ou "Average" (par défaut, "NA")

. **tab.disj** : data.frame facultatif correspondant au tableau disjonctif utilisé pour l'analyse ; il correspond à un tableau disjonctif obtenu à partir de la méthode d'imputation (voir package missMDA).

Valeurs

Renvoie une liste comprenant :

. **eig** : une matrice contenant toutes les valeurs propres, le pourcentage de variance et le pourcentage de variance cumulé

. **var** : une liste de matrices contenant tous les résultats pour les variables actives (coordonnées, carré cosinus, contributions, v.test, carré rapport de corrélation)

. **ind** : une liste de matrices contenant tous les résultats pour les individus actifs (coordonnées, cosinus carré, contributions)

. **ind.sup** : une liste de matrices contenant tous les résultats pour les individus supplémentaires (coordonnées, cosinus carré)

. **quanti.sup** : une matrice contenant les coordonnées des variables quantitatives supplémentaires (la corrélation entre une variable et un axe est égale à la coordonnée de la variable sur l'axe)

. **quali.sup** : une liste de matrices avec tous les résultats pour les variables catégorielles supplémentaires (coordonnées de chaque modalité de chaque variable,

cosinus carré et v.test qui est un critère avec une distribution normale, rapport de corrélation carré)

. **call** : une liste avec quelques statistiques

Remarque :

MCA() renvoie les graphiques des individus et des catégories et le graphique avec les variables. Les tracés peuvent être améliorés en utilisant l'argument autolab, en modifiant la taille des étiquettes ou en sélectionnant certains éléments grâce à la plot.MCA fonction.

2.3.2 Application de MCA()

Nous allons reprendre le cas de six individus mentionné plus haut pour appliquer l'objet MCA() dans le but de bien expliquer le rôle du ACM dans la réduction de la dimensionnalité et afficher les projections graphiques pour comprendre les relations entre les individus et les catégories. La fonction MCA() effectue une Analyse des Correspondances Multiples sur les variables catégorielles des individus. Elle renvoie un objet mca contenant les résultats de l'analyse. Vous pouvez accéder à différentes propriétés de l'objet MCA() pour obtenir des informations sur les résultats de l'analyse.

Algorithme - mca

1. Création d'un dataframe nommé "data" avec trois colonnes : "Fruit", "Légume" et "Viande". Les valeurs des colonnes sont désignées à l'aide de la fonction data.frame().
2. Affichage du dataframe dans une fenêtre de visualisation à l'aide de la fonction View().
3. Installation des packages nécessaires : "FactoMineR", "factoextra", "ggfortify"

et "corrplot". Cela permet d'ajouter des fonctionnalités supplémentaires à R en installant ces packages via la fonction install.packages().

4. Chargement des bibliothèques installées à l'aide de la fonction library() : "FactoMineR", "factoextra", "ggfortify" et "corrplot".
5. Application de l'analyse des correspondances multiples (MCA) sur le dataframe "data" à l'aide de la fonction MCA()de la bibliothèque "FactoMineR". Le résultat est stocké dans l'objet "mca".

Code - mca

1. # Créeation de dataframe

   . data <- data.frame(

   Fruit = c("Pomme", "Poire", "Orange", "Pomme", "Poire", "Orange"),

   Légume = c("Haricot", "Haricot", "Carotte", "Carotte", "Epinard", "Epinard"),

   Viande = c("Cheval", "Cheval", "Mouton", "Mouton", "Bœuf", "Bœuf"))

   . View(data)

	Fruit	Légume	Viande
1	Pomme	Haricot	Cheval
2	Poire	Haricot	Cheval
3	Orange	Carotte	Mouton
4	Pomme	Carotte	Mouton
5	Poire	Epinard	Bœuf
6	Orange	Epinard	Bœuf

Showing 1 to 6 of 6 entries, 3 total columns

2. # Instalation de packages

   . install.packages("FactoMineR")

   . install.packages("factoextra")

. install.packages("ggfortify")

. install.packages("corrplot")

3. # Chargement de bibliothèques

. library("FactoMineR")

. library(factoextra)

. library("ggfortify")

. library(corrplot)

4. # Application de mca

. mca <- MCA(data)

Sorties - mca

Les trois graphiques suivants est une représentation visuelle qui affiche les variables catégorielles et les observations d'un jeu de données dans un espace bidimensionnel, permettant d 'explorer leurs relations.

1. Carte factorielle des variables categoriélles

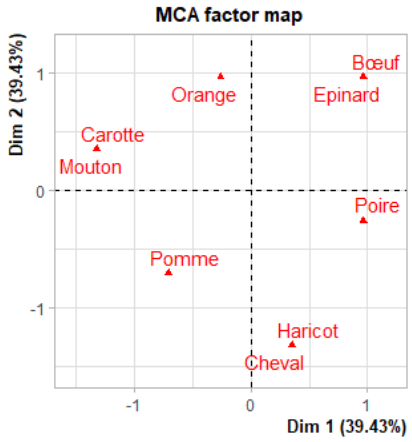


FIG. 2.2 – Carte factorielle des variables categoriélles

2. Carte factorielle des individus

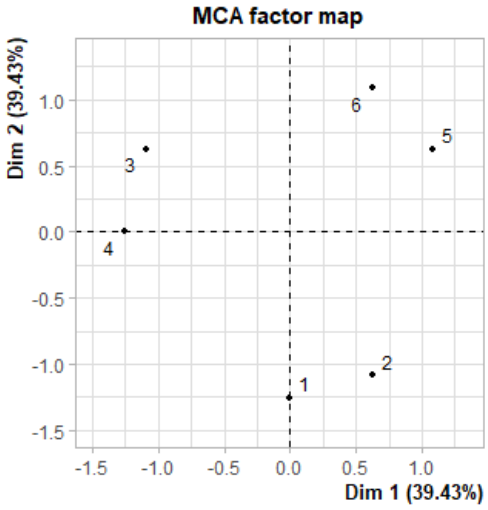


FIG. 2.3 – Carte factorielle des individus

3. Représenation des variables

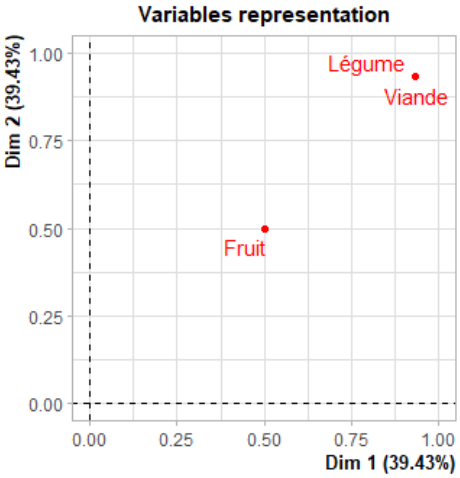


FIG. 2.4 – Représenation des variables

2.3.3 Affichage de MCA()

Dans cette partie nous afficherons les valeurs numériques relatives à l'exécution de MCA() ensuite les visualisations graphiques relatives à ces derniers avec une explication de chaque visualisation

Algorithme - affichage -mca

1. Affichage des valeurs propres de l'analyse des correspondances multiples (MCA) à l'aide de mca$eig. Les valeurs propres donnent une idée de la variance expliquée par chaque dimension principale.
2. Visualisation des valeurs propres sous forme de diagramme à barres à l'aide de la fonction barplot(). Les barres représentent les pourcentages de variances expliquées par chaque dimension principale.
3. Visualisation du graphique du screeplot avec les pourcentages de variance expliqués pour chaque dimension principale à l'aide de la fonction fviz_screeplot().
4. Affichage des carrés des corrélations entre les variables et les dimensions principales à l'aide de mcavareta2.
5. Visualisation des variables dans le plan factoriel à l'aide de la fonction fviz_mca_var(). Les variables sont représentées en fonction de leur corrélation avec les dimensions principales.
6. Affichage des coordonnées des variables dans le plan factoriel à l'aide de mcavarcoord.
7. Visualisation des variables dans le plan factoriel en utilisant les carrés des cosinus des variables (cos2) à l'aide de la fonction fviz_mca_var(). Les couleurs des points représentent les carrés des cosinus des variables.
8. Affichage des coordonnées des individus dans le plan factoriel à l'aide de mcaindcoord.
9. Visualisation des individus et des variables dans le plan factoriel en utilisant la fonction fviz_mca_biplot(). Les flèches représentent les variables et les points représentent les individus.
10. Affichage des contributions des variables à chaque dimension principale à l'aide de mcavarcontrib.
11. Visualisation des contributions des variables à la première dimension principale à l'aide de fviz_contrib().

12. Visualisation des contributions des variables à la deuxième dimension principale à l'aide de fviz_contrib().
13. Affichage des carrés des cosinus des variables à l'aide de mcavarcos2.
14. Visualisation des carrés des cosinus des variables à l'aide de la fonction corrplot().
15. Calcul des sommes des carrés des cosinus des variables pour les deux premières dimensions principales à l'aide de rowSums(cos2_var).
16. Visualisation des carrés des cosinus des variables pour les deux premières dimensions principales à l'aide de fviz_cos2().
17. Visualisation avec ellipses : Une visualisation avec des ellipses, où les individus sont colorés selon un groupe et des ellipses de concentration sont ajoutés pour visualiser la dispersion des groupes.

Code R - affichage - mca

1. # visualisation 1

. mca$eig

. # Avec poucentages

. eig.val <- mca$eig

barplot(eig.val[, 2],

names.arg = 1 :nrow(eig.val),

main = "Variances Explained by Dimensions (%)",

xlab = "Principal Dimensions",

ylab = "Percentage of variances",

col ="steelblue")

. # Avec pourcentages

. fviz_screeplot(mca, addlabels = TRUE, ylim = c(0, 45))

2. # Visualisation 2

. mcavareta2

. var <- get_mca_var(mca)

. fviz_mca_var (mca, choice = "mca.cor",

repel = TRUE,

ggtheme = theme_minimal ())

3. # Visualisation 3

. mcavarcoord

. fviz_mca_var(mca, col.var = "cos2",

gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),

repel = TRUE,

ggtheme = theme_minimal())

4. # Visualisation 4

. mcaindcoord

. fviz_mca_biplot (mca, repel = TRUE,

ggtheme = theme_minimal())

5. # Visualisation 5

. mcavarcontrib

. # Dim 1

. fviz_contrib(mca, choice = "var", axes = 1, top = 15)

. # Dim 2

. fviz_contrib(mca, choice = "var", axes = 2, top = 15)

6. # Visualisation 6

. mcavarcos2

. corrplot(var$cos2, is.corr=FALSE)

7. # Visualisation 7

. cos2_var <- mcavarcos2[, 1 :2]

. rowSums(cos2_var)

. fviz_cos2(mca, choice = "var", axes = 1 :2)

Sorties numériques - affichage -mca

1. **Valeurs propres**

```
> mca$eig
       eigenvalue percentage of variance cumulative percentage of variance
dim 1 7.886751e-01           3.943376e+01                          39.43376
dim 2 7.886751e-01           3.943376e+01                          78.86751
dim 3 2.113249e-01           1.056624e+01                          89.43376
dim 4 2.113249e-01           1.056624e+01                         100.00000
dim 5 3.313660e-32           1.656830e-30                         100.00000
```

2. **Carrés des corrélations entre les variables et les dimensions principales**

```
> mca$var$eta2
           Dim 1     Dim 2     Dim 3     Dim 4        Dim 5
Fruit  0.5000000 0.5000000 0.5000000 0.5000000 1.862518e-63
Légume 0.9330127 0.9330127 0.0669873 0.0669873 4.600459e-64
Viande 0.9330127 0.9330127 0.0669873 0.0669873 9.715382e-64
```

3. **Coordonnées des variables dans le plan factoriel**

```
> mca$var$coord
             Dim 1      Dim 2       Dim 3       Dim 4         Dim 5
Orange  -0.2588190  0.9659258  0.25881905  0.96592583 4.315690e-32
Poire    0.9659258 -0.2588190 -0.96592583 -0.25881905 4.315690e-32
Pomme   -0.7071068 -0.7071068  0.70710678 -0.70710678 4.315690e-32
Carotte -1.3194792  0.3535534 -0.35355339 -0.09473435 1.010358e-32
Epinard  0.9659258  0.9659258  0.25881905 -0.25881905 2.527900e-32
Haricot  0.3535534 -1.3194792  0.09473435  0.35355339 2.527900e-32
Bœuf     0.9659258  0.9659258  0.25881905 -0.25881905 2.527900e-32
Cheval   0.3535534 -1.3194792  0.09473435  0.35355339 2.527900e-32
Mouton  -1.3194792  0.3535534 -0.35355339 -0.09473435 4.045441e-32
```

4. **Coordonnées des individus dans le plan factoriel**

```
> mca$ind$coord
          Dim 1         Dim 2         Dim 3         Dim 4         Dim 5
1  2.461214e-17 -1.255926e+00  6.501152e-01  1.562322e-16 -1.124071e-32
2  6.279630e-01 -1.087664e+00 -5.630163e-01  3.250576e-01 -4.549412e-32
3 -1.087664e+00  6.279630e-01 -3.250576e-01  5.630163e-01 -1.124071e-32
4 -1.255926e+00  6.250736e-17  1.207549e-16 -6.501152e-01 -4.549412e-32
5  1.087664e+00  6.279630e-01 -3.250576e-01 -5.630163e-01 -1.124071e-32
6  6.279630e-01  1.087664e+00  5.630163e-01  3.250576e-01 -4.549412e-32
```

5. **Contributions des variables à chaque dimension principale**

```
> mca$var$contrib
             Dim 1      Dim 2      Dim 3      Dim 4         Dim 5
Orange   0.9437388 13.1445856  3.5220811 49.0562612 6.245255e-31
Poire   13.1445856  0.9437388 49.0562612  3.5220811 6.245255e-31
Pomme    7.0441622  7.0441622 26.2891712 26.2891712 6.245255e-31
Carotte 24.5281306  1.7610405  6.5722928  0.4718694 3.422948e-32
Epinard 13.1445856 13.1445856  3.5220811  3.5220811 2.142739e-31
Haricot  1.7610405 24.5281306  0.4718694  6.5722928 2.142739e-31
Bœuf    13.1445856 13.1445856  3.5220811  3.5220811 2.142739e-31
Cheval   1.7610405 24.5281306  0.4718694  6.5722928 2.142739e-31
Mouton  24.5281306  1.7610405  6.5722928  0.4718694 5.487586e-31
```

6. **Carrés des cosinus des variables**

```
> mca$var$cos2
              Dim 1      Dim 2       Dim 3       Dim 4         Dim 5
Orange  0.03349365 0.46650635 0.033493649 0.466506351 9.312592e-64
Poire   0.46650635 0.03349365 0.466506351 0.033493649 9.312592e-64
Pomme   0.25000000 0.25000000 0.250000000 0.250000000 9.312592e-64
Carotte 0.87051270 0.06250000 0.062500000 0.004487298 5.104118e-65
Epinard 0.46650635 0.46650635 0.033493649 0.033493649 3.195138e-64
Haricot 0.06250000 0.87051270 0.004487298 0.062500000 3.195138e-64
Bœuf    0.46650635 0.46650635 0.033493649 0.033493649 3.195138e-64
Cheval  0.06250000 0.87051270 0.004487298 0.062500000 3.195138e-64
Mouton  0.87051270 0.06250000 0.062500000 0.004487298 8.182797e-64
```

7. **Sommes des carrés des cosinus des variables**

```
> rowSums(cos2_var)
   Orange     Poire     Pomme   Carotte   Epinard   Haricot      Bœuf    Cheval
0.5000000 0.5000000 0.5000000 0.9330127 0.9330127 0.9330127 0.9330127 0.9330127
   Mouton
0.9330127
```

Sorties graphiques - affichage - mca

1. **Diagramme en barres :** Un diagramme en barres qui affiche les variances expliquées par les dimensions, où chaque barre représente une dimension principale et la hauteur de la barre représente le pourcentage de variance associé à cette dimension.

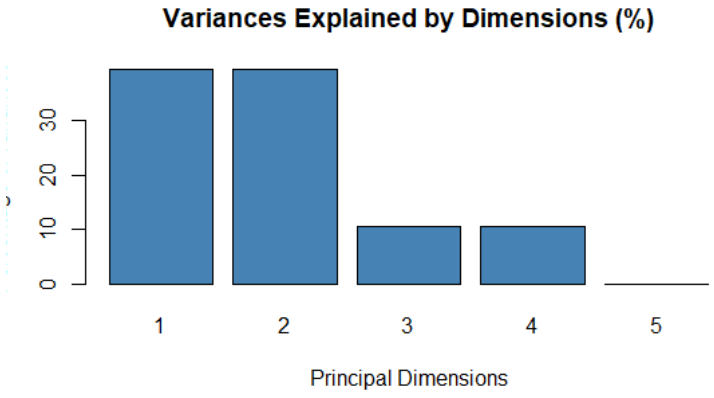


FIG. 2.5 – Diagramme en barres des variances expliquées par les dimensions

Screeplot : est un graphique utilisé dans l'analyse en composantes principales (PCA) ou dans l'analyse en composantes principales multiples (MCA) pour visualiser l'éboulis des valeurs propres ou des pourcentages de variance expliqués par chaque composante. Les éléments des caractéristiques sont ajoutés au graphique pour une meilleure lisibilité**.**

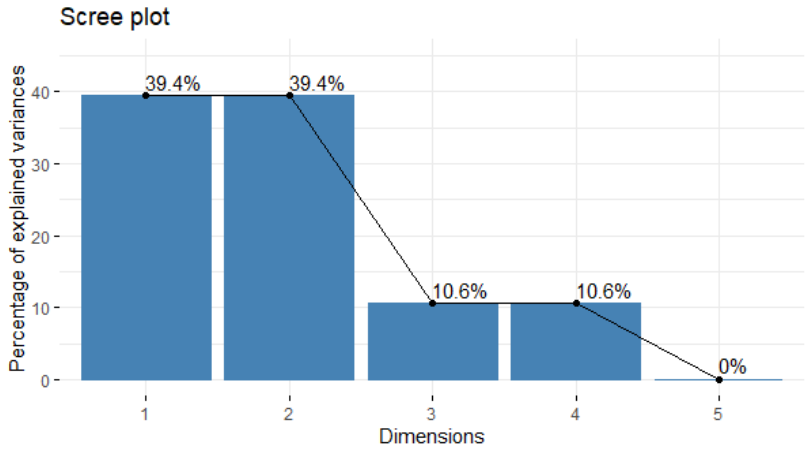


FIG. 2.6 – Pourcentage de la variance expliqué dans les composantes principales

2. **Graphique des variables MCA :** Il permet de visualiser les variables et leur relation avec les axes MCA. Chaque variable est représentée par un point dans un espace biplot. La position du point indique la corrélation de la variable avec les axes MCA. Les variables qui sont proches des axes MCA ont une forte corrélation avec ces axes, tandis que celles qui sont éloignées ont une corrélation plus faible..Le graphique "Variables - MCA" est souvent utilisé pour identifier les variables qui contribuent le plus à chaque axe MCA. Les variables qui sont proches des axes et qui ont une corrélation élevée sont susceptibles comme étant les plus influentes pour cet axe spécifique.

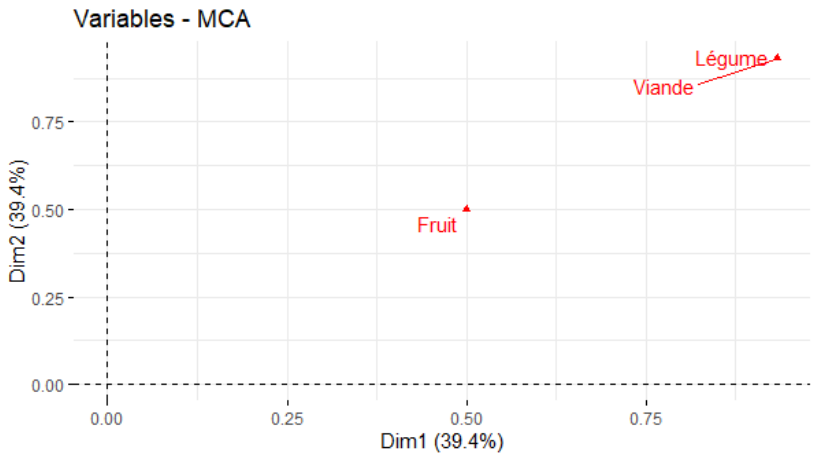


FIG. 2.7 – Graphique des variables MCA

3. **Graphique des variable categories - MCA :** Il permet de visualiser les relations entre les différentes catégories de variables et les axes MCA.. Chaque catégorie de variable est représentée par un point dans un espace biplot. La position du point indique la contribution de la catégorie de variable à chaque axe MCA. Les catégories de variables qui sont proches des axes MCA ont une forte contribution à ces axes, tandis que celles qui sont éloignées ont une contribution plus faible. Le graphique "Variable categories - MCA" est utilisé pour identifier les catégories de variables qui sont les plus importantes pour chaque axe MCA. Les catégories de variables qui sont proches des axes et qui ont une contribution élevée sont susceptibles comme étant les plus influentes pour cet axe spécifique.

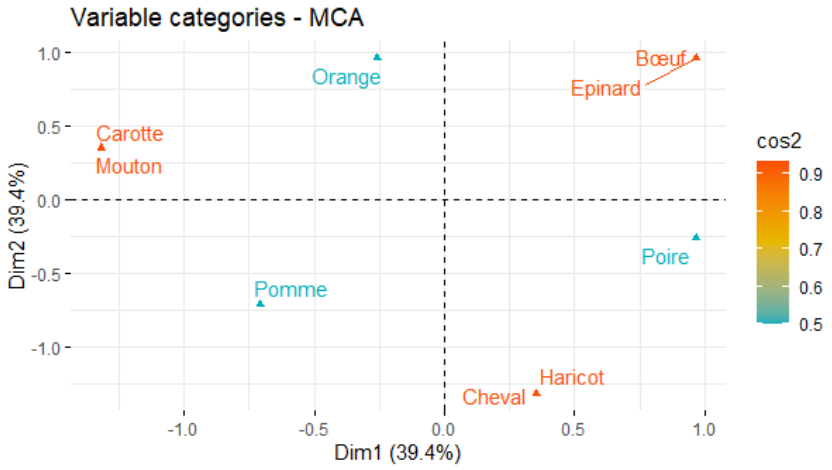


FIG. 2.8 – Graphique des variable categories - MCA

4. **MCA - Biplo**t : Est une représentation graphique utilisée dans une analyse des correspondances multiples (MCA) pour visualiser simultanément les individus et les variables sur les axes MCA. Il combine les informations sur la structure des individus et la structure des variables dans un même graphique. Dans un graphique "MCA - Biplot", les individus sont représentés par des points et les variables sont représentés par des flèches. La position des individus sur le graphique indique leur relation avec les axes MCA, tandis que la direction et la longueur des flèches représentent la contribution des variables à chaque axe. Plus précisément, la position des individus est déterminée par leurs coordonnées sur les axes MCA. Les individus qui sont proches les uns des autres sont similaires en termes de profils de variables, tandis que ceux qui sont éloignés sont différents. Les variables sont représentées par des flèches qui indiquent leur direction et leur amplitude. Les variables qui ont une forte contribution à un axe donné sont représentées par des flèches plus longues et alignées dans la direction de l'axe. Il permet de comprendre la structure des données, d'identifier les variables qui sont les plus influentes sur chaque axe

et de détecter les similarités et les dissimilarités entre les individus. Cette représentation graphique est utile pour l'interprétation et l'analyse des résultats d'une analyse en principales composantes multiples.

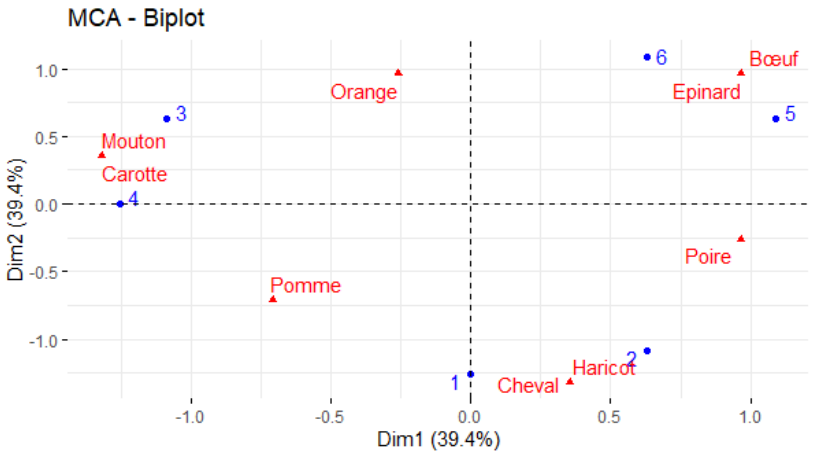


FIG. 2.9 – MCA - Biplot

5. **Graphique des contributions :** Un tracé de contributions est une représentation graphique utilisée dans l'analyse des correspondances multiples (MCA) pour visualiser les contributions relatives des variables ou des individus aux dimensions principales (selon l'axe choisi, ici Dim 1, Dim 2)

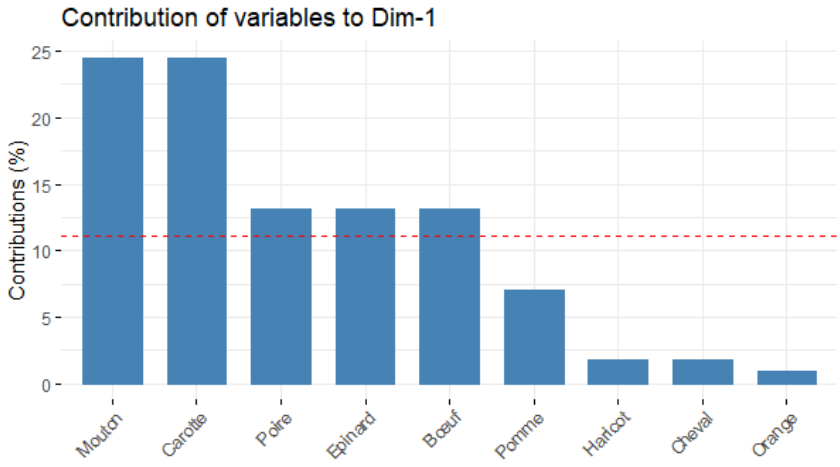


FIG. 2.10 – Contributions relatives des variables ou des individus aux dimensions principales

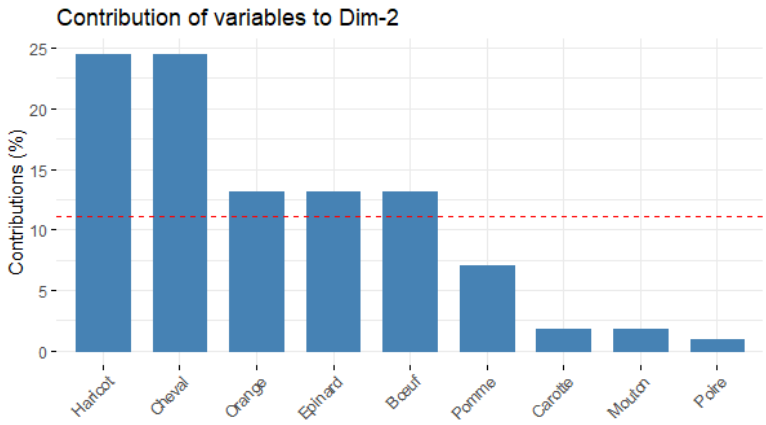


FIG. 2.11 – Contributions relatives des variables ou des individus aux dimensions principales Dim 2

6. **Corrplot :** Permet de comprendre la qualité de représentation des variables sur les axes MCA et d'identifier les variables qui contribuent le plus

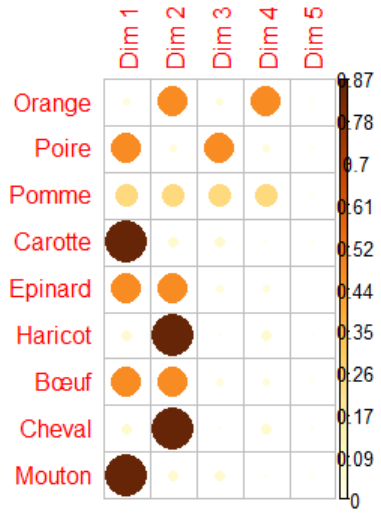


FIG. 2.12 – Qualité de représentation des variables

7. **Qualité de représentation :** Le graphique "Cos2 quality of representation" offre une visualisation claire de la contribution et de la qualité de représentation des variables dans une analyse des correspondances multiples (MCA) en utilisant les carrés des cosinus des angles (cos2). Cela permet de déterminer les variables les plus importantes et bien représentées dans l'espace des dimensions

de la MCA.

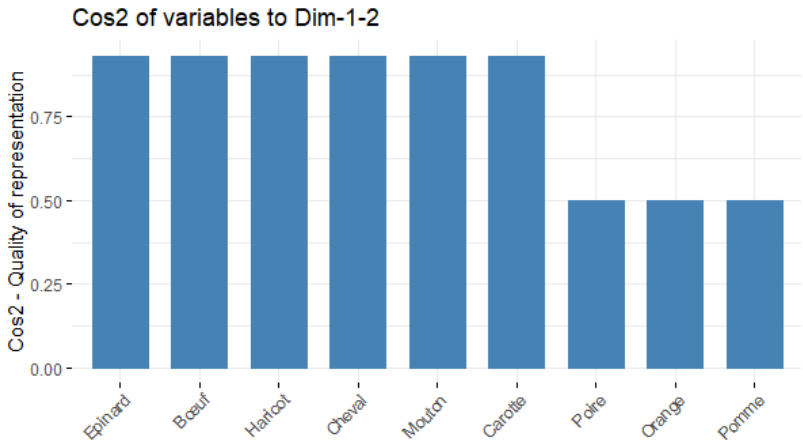


FIG. 2.13 – Qualité de représentation des variables

2.3.4 Affichage de valeurs numériques supplimentaires

1. **print(mca) :** Affiche les informations de l'objet MCA, généralement des statistiques et des paramètres liés à l'analyse des correspondances multiples.

```
> print(mca)
**Results of the Multiple Correspondence Analysis (MCA)**
The analysis was performed on 6 individuals, described by 3 variables
*The results are available in the following objects:

   name              description
1  "$eig"            "eigenvalues"
2  "$var"            "results for the variables"
3  "$var$coord"      "coord. of the categories"
4  "$var$cos2"       "cos2 for the categories"
5  "$var$contrib"    "contributions of the categories"
6  "$var$v.test"     "v-test for the categories"
7  "$ind"            "results for the individuals"
8  "$ind$coord"      "coord. for the individuals"
9  "$ind$cos2"       "cos2 for the individuals"
10 "$ind$contrib"    "contributions of the individuals"
11 "$call"           "intermediate results"
12 "$call$marge.col" "weights of columns"
13 "$call$marge.li"  "weights of rows"
```

2. **summary(mca) :** Donne un résumé statistique des résultats de l'analyse des correspondances multiples, y compris les valeurs propres, les pourcentages de

variance expliqués, et autres statistiques importantes.

```
> summary(mca)

Call:
MCA(X = data)


Eigenvalues
                       Dim.1   Dim.2   Dim.3   Dim.4   Dim.5
Variance               0.789   0.789   0.211   0.211   0.000
% of var.             39.434  39.434  10.566  10.566   0.000
Cumulative % of var.  39.434  78.868  89.434 100.000 100.000
```

```
Individuals
          Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3    ctr   cos2
1       |  0.000  0.000  0.000 | -1.256 33.333  0.789 |  0.650 33.333  0.211 |
2       |  0.628  8.333  0.197 | -1.088 25.000  0.592 | -0.563 25.000  0.158 |
3       | -1.088 25.000  0.592 |  0.628  8.333  0.197 | -0.325  8.333  0.053 |
4       | -1.256 33.333  0.789 |  0.000  0.000  0.000 |  0.000  0.000  0.000 |
5       |  1.088 25.000  0.592 |  0.628  8.333  0.197 | -0.325  8.333  0.053 |
6       |  0.628  8.333  0.197 |  1.088 25.000  0.592 |  0.563 25.000  0.158 |

Categories
          Dim.1    ctr   cos2 v.test    Dim.2    ctr   cos2 v.test    Dim.3    ctr
Orange  | -0.259  0.944  0.033 -0.409 |  0.966 13.145  0.467  1.527 |  0.259  3.522
Poire   |  0.966 13.145  0.467  1.527 | -0.259  0.944  0.033 -0.409 | -0.966 49.056
Pomme   | -0.707  7.044  0.250 -1.118 | -0.707  7.044  0.250 -1.118 |  0.707 26.289
Carotte | -1.319 24.528  0.871 -2.086 |  0.354  1.761  0.063  0.559 | -0.354  6.572
Epinard |  0.966 13.145  0.467  1.527 |  0.966 13.145  0.467  1.527 |  0.259  3.522
Haricot |  0.354  1.761  0.063  0.559 | -1.319 24.528  0.871 -2.086 |  0.095  0.472
Bœuf    |  0.966 13.145  0.467  1.527 |  0.966 13.145  0.467  1.527 |  0.259  3.522
Cheval  |  0.354  1.761  0.063  0.559 | -1.319 24.528  0.871 -2.086 |  0.095  0.472
Mouton  | -1.319 24.528  0.871 -2.086 |  0.354  1.761  0.063  0.559 | -0.354  6.572
          cos2 v.test
Orange   0.033  0.409 |
Poire    0.467 -1.527 |
Pomme    0.250  1.118 |
Carotte  0.063 -0.559 |
Epinard  0.033  0.409 |
Haricot  0.004  0.150 |
Bœuf     0.033  0.409 |
Cheval   0.004  0.150 |
Mouton   0.063 -0.559 |

Categorical variables (eta2)
          Dim.1 Dim.2 Dim.3
Fruit   | 0.500 0.500 0.500 |
Légume  | 0.933 0.933 0.067 |
Viande  | 0.933 0.933 0.067 |
```

3. **dimdesc(mca) :** Affiche les descriptions des dimensions principales de l'ACM,

généralement les variables les plus associées à chaque dimension.

```
> dimdesc(mca)
$`Dim 1`

Link between the variable and the categorical variable (1-way anova)
============================================
              R2    p.value
Légume 0.9330127 0.01733759
Viande 0.9330127 0.01733759

Link between variable and the categories of the categorical variables
================================================================
                Estimate     p.value
Viande=Mouton  -1.171795 0.006580651
Légume=Carotte -1.171795 0.006580651

$`Dim 2`

Link between the variable and the categorical variable (1-way anova)
============================================
              R2    p.value
Légume 0.9330127 0.01733759
Viande 0.9330127 0.01733759

Link between variable and the categories of the categorical variables
================================================================
                Estimate     p.value
Viande=Cheval  -1.171795 0.006580651
Légume=Haricot -1.171795 0.006580651

$`Dim 3`
```

4. **names(mca) :** Affiche les noms des caractéristiques de l'objet MCA, tels que "eig" pour les valeurs propres, "var" pour les informations relatives aux variables, "ind" pour les informations relatives aux individus, etc.

```
> names(mca)
[1] "eig"  "call" "ind"  "var"  "svd"
```

5. **mca$eig :** Affiche les valeurs propres et les pourcent

```
> mca$eig
       eigenvalue percentage of variance cumulative percentage of variance
dim 1 7.886751e-01           3.943376e+01                          39.43376
dim 2 7.886751e-01           3.943376e+01                          78.86751
dim 3 2.113249e-01           1.056624e+01                          89.43376
dim 4 2.113249e-01           1.056624e+01                         100.00000
dim 5 3.313660e-32           1.656830e-30                         100.00000
```

6. **mca$call** : Affiche la commande ou les paramètres utilisés pour créer l'objet MCA.

```
> mca$call
$X
   Fruit  Légume Viande
1  Pomme Haricot Cheval
2  Poire Haricot Cheval
3 Orange Carotte Mouton
4  Pomme Carotte Mouton
5  Poire Epinard   Bœuf
6 Orange Epinard   Bœuf

$marge.col
   Orange     Poire     Pomme   Carotte   Epinard   Haricot      Bœuf    Cheval
0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111 0.1111111
   Mouton
0.1111111

$marge.row
        1         2         3         4         5         6
0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667

$ncp
[1] 5

$row.w
[1] 1 1 1 1 1 1

$excl
NULL

$call
MCA(X = data)
```

```
$Xtot
  Orange Poire Pomme Carotte Epinard Haricot Bœuf Cheval Mouton
1      0     0     1       0       0       1    0      1      0
2      0     1     0       0       0       1    0      1      0
3      1     0     0       1       0       0    0      0      1
4      0     0     1       1       0       0    0      0      1
5      0     1     0       0       1       0    1      0      0
6      1     0     0       0       1       0    1      0      0

$N
[1] 18

$quali
[1] 1 2 3
```

7. **mca$ind** : Cette composante contient les informations relatives aux indivi-

dus (ou observations) dans l'ACM. Elle peut inclure des éléments tels que les coordonnées factorielles des individus, les contributions des individus aux dimensions principales, les cosinus carrés des individus (mesurant la qualité de leur représentation), etc.

```
> mca$ind
$coord
          Dim 1         Dim 2         Dim 3         Dim 4         Dim 5
1  2.461214e-17 -1.255926e+00  6.501152e-01  1.562322e-16 -1.124071e-32
2  6.279630e-01 -1.087664e+00 -5.630163e-01  3.250576e-01 -4.549412e-32
3 -1.087664e+00  6.279630e-01 -3.250576e-01  5.630163e-01 -1.124071e-32
4 -1.255926e+00  6.250736e-17  1.207549e-16 -6.501152e-01 -4.549412e-32
5  1.087664e+00  6.279630e-01 -3.250576e-01 -5.630163e-01 -1.124071e-32
6  6.279630e-01  1.087664e+00  5.630163e-01  3.250576e-01 -4.549412e-32

$contrib
         Dim 1        Dim 2        Dim 3        Dim 4        Dim 5
1 1.280116e-32 3.333333e+01 3.333333e+01 1.925038e-30 6.355185e-32
2 8.333333e+00 2.500000e+01 2.500000e+01 8.333333e+00 1.041001e-30
3 2.500000e+01 8.333333e+00 8.333333e+00 2.500000e+01 6.355185e-32
4 3.333333e+01 8.256822e-32 1.150027e-30 3.333333e+01 1.041001e-30
5 2.500000e+01 8.333333e+00 8.333333e+00 2.500000e+01 6.355185e-32
6 8.333333e+00 2.500000e+01 2.500000e+01 8.333333e+00 1.041001e-30

$cos2
         Dim 1        Dim 2        Dim 3        Dim 4        Dim 5
1 3.028787e-34 7.886751e-01 2.113249e-01 1.220426e-32 6.317676e-65
2 1.971688e-01 5.915064e-01 1.584936e-01 5.283122e-02 1.034857e-63
3 5.915064e-01 1.971688e-01 5.283122e-02 1.584936e-01 6.317676e-65
4 7.886751e-01 1.953585e-33 7.290879e-33 2.113249e-01 1.034857e-63
5 5.915064e-01 1.971688e-01 5.283122e-02 1.584936e-01 6.317676e-65
6 1.971688e-01 5.915064e-01 1.584936e-01 5.283122e-02 1.034857e-63
```

8. **mca$var :** Cette composante contient les informations relatives aux variables dans l'analyse des correspondances multiples (ACM). Elle peut inclure des éléments tels que les coordonnées factorielles des variables, les contributions des variables aux dimensions principales, les cosinus carrés des variables (mesurant la qualité de leur représentation), etc.

```
> mca$var
$coord
             Dim 1       Dim 2       Dim 3       Dim 4         Dim 5
Orange  -0.2588190  0.9659258  0.25881905  0.96592583 4.315690e-32
Poire    0.9659258 -0.2588190 -0.96592583 -0.25881905 4.315690e-32
Pomme   -0.7071068 -0.7071068  0.70710678 -0.70710678 4.315690e-32
Carotte -1.3194792  0.3535534 -0.35355339 -0.09473435 1.010358e-32
Epinard  0.9659258  0.9659258  0.25881905 -0.25881905 2.527900e-32
Haricot  0.3535534 -1.3194792  0.09473435  0.35355339 2.527900e-32
Bœuf     0.9659258  0.9659258  0.25881905 -0.25881905 2.527900e-32
Cheval   0.3535534 -1.3194792  0.09473435  0.35355339 2.527900e-32
Mouton  -1.3194792  0.3535534 -0.35355339 -0.09473435 4.045441e-32

$contrib
             Dim 1       Dim 2      Dim 3      Dim 4         Dim 5
Orange   0.9437388 13.1445856  3.5220811 49.0562612 6.245255e-31
Poire   13.1445856  0.9437388 49.0562612  3.5220811 6.245255e-31
Pomme    7.0441622  7.0441622 26.2891712 26.2891712 6.245255e-31
Carotte 24.5281306  1.7610405  6.5722928  0.4718694 3.422948e-32
Epinard 13.1445856 13.1445856  3.5220811  3.5220811 2.142739e-31
Haricot  1.7610405 24.5281306  0.4718694  6.5722928 2.142739e-31
Bœuf    13.1445856 13.1445856  3.5220811  3.5220811 2.142739e-31
Cheval   1.7610405 24.5281306  0.4718694  6.5722928 2.142739e-31
Mouton  24.5281306  1.7610405  6.5722928  0.4718694 5.487586e-31

$cos2
             Dim 1       Dim 2       Dim 3       Dim 4         Dim 5
Orange  0.03349365 0.46650635 0.033493649 0.466506351 9.312592e-64
Poire   0.46650635 0.03349365 0.466506351 0.033493649 9.312592e-64
Pomme   0.25000000 0.25000000 0.250000000 0.250000000 9.312592e-64
Carotte 0.87051270 0.06250000 0.062500000 0.004487298 5.104118e-65
Epinard 0.46650635 0.46650635 0.033493649 0.033493649 3.195138e-64
Haricot 0.06250000 0.87051270 0.004487298 0.062500000 3.195138e-64
Bœuf    0.46650635 0.46650635 0.033493649 0.033493649 3.195138e-64
Cheval  0.06250000 0.87051270 0.004487298 0.062500000 3.195138e-64
Mouton  0.87051270 0.06250000 0.062500000 0.004487298 8.182797e-64


$v.test
             Dim 1       Dim 2       Dim 3       Dim 4         Dim 5
Orange  -0.4092288  1.5272628  0.4092288  1.5272628 6.823706e-32
Poire    1.5272628 -0.4092288 -1.5272628 -0.4092288 6.823706e-32
Pomme   -1.1180340 -1.1180340  1.1180340 -1.1180340 6.823706e-32
Carotte -2.0862798  0.5590170 -0.5590170 -0.1497882 1.597516e-32
Epinard  1.5272628  1.5272628  0.4092288 -0.4092288 3.996960e-32
Haricot  0.5590170 -2.0862798  0.1497882  0.5590170 3.996960e-32
Bœuf     1.5272628  1.5272628  0.4092288 -0.4092288 3.996960e-32
Cheval   0.5590170 -2.0862798  0.1497882  0.5590170 3.996960e-32
Mouton  -2.0862798  0.5590170 -0.5590170 -0.1497882 6.396404e-32

$eta2
           Dim 1     Dim 2     Dim 3     Dim 4         Dim 5
Fruit  0.5000000 0.5000000 0.5000000 0.5000000 1.862518e-63
Légume 0.9330127 0.9330127 0.0669873 0.0669873 4.600459e-64
Viande 0.9330127 0.9330127 0.0669873 0.0669873 9.715382e-64
```

9. **mca$svd** : Cette composante contient les résultats de la décomposition en

valeurs singulières

```
> mca$svd
$vs
[1] 8.880738e-01 8.880738e-01 4.597008e-01 4.597008e-01 1.820346e-16

$u
              [,1]          [,2]          [,3]          [,4]          [,5]
[1,]  2.771407e-17 -1.414214e+00  1.414214e+00  3.398563e-16 -6.175039e-17
[2,]  7.071068e-01 -1.224745e+00 -1.224745e+00  7.071068e-01 -2.499202e-16
[3,] -1.224745e+00  7.071068e-01 -7.071068e-01  1.224745e+00 -6.175039e-17
[4,] -1.414214e+00  7.038532e-17  2.626816e-16 -1.414214e+00 -2.499202e-16
[5,]  1.224745e+00  7.071068e-01 -7.071068e-01 -1.224745e+00 -6.175039e-17
[6,]  7.071068e-01  1.224745e+00  1.224745e+00  7.071068e-01 -2.499202e-16

$v
            [,1]       [,2]       [,3]       [,4]         [,5]
 [1,] -0.2914387  1.0876639  0.5630163  2.1012053 2.370808e-16
 [2,]  1.0876639 -0.2914387 -2.1012053 -0.5630163 2.370808e-16
 [3,] -0.7962252 -0.7962252  1.5381890 -1.5381890 2.370808e-16
 [4,] -1.4857765  0.3981126 -0.7690945 -0.2060783 5.550363e-17
 [5,]  1.0876639  1.0876639  0.5630163 -0.5630163 1.388692e-16
 [6,]  0.3981126 -1.4857765  0.2060783  0.7690945 1.388692e-16
 [7,]  1.0876639  1.0876639  0.5630163 -0.5630163 1.388692e-16
 [8,]  0.3981126 -1.4857765  0.2060783  0.7690945 1.388692e-16
 [9,] -1.4857765  0.3981126 -0.7690945 -0.2060783 2.222347e-16
```

Chapitre 3

Application réelles sur l'Analyse des Correspondances Multiples

Dans cette partie nous allons appliquer l'ACM sur des données réelles CleanCreditScoring. Le jeu de données est téléchargeable avec le lien suivant :

CleanCreditScoring.csv

Clean Credit Scoring, ou Clean Credit Analysis, est une application qui utilise des techniques avancées d'analyse et de modélisation des données pour évaluer la solvabilité d'un individu et fournir une évaluation complète de son profil de crédit. Il est couramment utilisé par les institutions financières, telles que les banques et les sociétés de prêt, pour prendre des décisions éclairées concernant les approbations de crédit, les taux d'intérêt et les conditions de prêt. Le fonctionnement du Clean Credit Scoring se résume en la collecte de données, l'ingénierie des fonctionnalités, le développement de modèles l'évaluation du modèle et en fin la prise de décision.

Clean Credit Scoring offre une approche basée sur les données pour l'évaluation du risque de crédit, permettant aux institutions financières de prendre des décisions de crédit plus précises et objectives. Il aide à réduire les défauts de crédit, à identifier

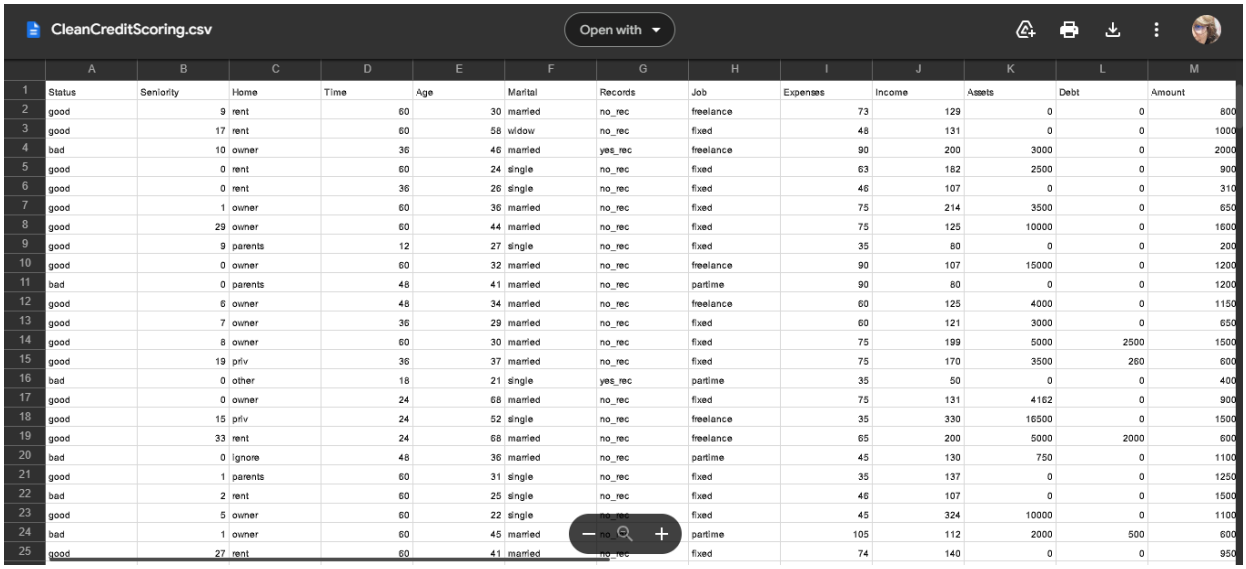
les clients potentiels à haut risque et à rationaliser le processus de prêt. De plus, il promeut des pratiques de prêt équitables en minimisant les préjugés humains et en fournissant une évaluation objective de la solvabilité d'un individu en fonction de ses antécédents financiers et de ses indicateurs.

Nous présenterons dans ce chapitre une application de l'ACM sur les variables catégorielles extraites du jeu de données **CleanCreditScoring**

3.1 Extraction des données catégorielles

-On commence par télécharger notre fichier csv

CleanCreditScoring.csv


CleanCreditScoring.csv

Open with

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Status	Seniority	Home	Time	Age	Marital	Records	Job	Expenses	Income	Assets	Debt	Amount
2	good	9	rent	60	30	married	no_rec	freelance	73	129	0	0	800
3	good	17	rent	60	58	widow	no_rec	fixed	48	131	0	0	1000
4	bad	10	owner	36	46	married	yes_rec	freelance	90	200	3000	0	2000
5	good	0	rent	60	24	single	no_rec	fixed	63	182	2500	0	900
6	good	0	rent	36	26	single	no_rec	fixed	46	107	0	0	310
7	good	1	owner	60	36	married	no_rec	fixed	75	214	3500	0	650
8	good	29	owner	60	44	married	no_rec	fixed	75	125	10000	0	1600
9	good	9	parents	12	27	single	no_rec	fixed	35	80	0	0	200
10	good	0	owner	60	32	married	no_rec	freelance	90	107	15000	0	1200
11	bad	0	parents	48	41	married	no_rec	partime	90	80	0	0	1200
12	good	6	owner	48	34	married	no_rec	freelance	60	125	4000	0	1150
13	good	7	owner	36	29	married	no_rec	fixed	60	121	3000	0	650
14	good	8	owner	60	30	married	no_rec	fixed	75	199	5000	2500	1500
15	good	19	priv	36	37	married	no_rec	fixed	75	170	3500	260	600
16	bad	0	other	18	21	single	yes_rec	partime	35	50	0	0	400
17	good	0	owner	24	68	married	no_rec	fixed	75	131	4162	0	900
18	good	15	priv	24	52	single	no_rec	freelance	35	330	16500	0	1500
19	good	33	rent	24	68	married	no_rec	freelance	65	200	5000	2000	600
20	bad	0	ignore	48	36	married	no_rec	partime	45	130	750	0	1100
21	good	1	parents	60	31	single	no_rec	fixed	35	137	0	0	1250
22	bad	2	rent	60	25	single	no_rec	fixed	46	107	0	0	1500
23	good	5	owner	60	22	single	no_rec	fixed	45	324	10000	0	1100
24	bad	1	owner	60	45	married	no_rec	partime	105	112	2000	500	600
25	good	27	rent	60	41	married	no_rec	fixed	74	140	0	0	950

-Nous allons charger et afficher le tableau sur RStudio avec les deux instructions suivantes :

$> df = read.table(file.choose(), header = TRUE, sep = ",")$

$> View(df)$

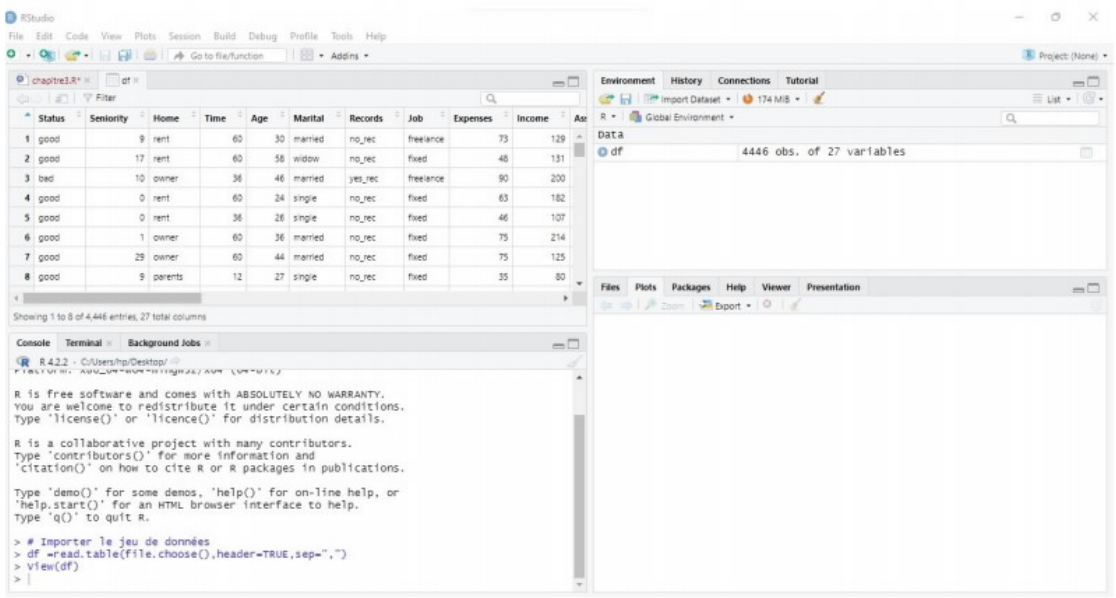


FIG. 3.1 – Environnement de travail de RStudio

-Nous appliquons l'instruction suivante pour afficher le nombre de ligne et de colonnes

$> dim(df)$

[1]4446 27

-Nous appliquons l'instruction suivante pour afficher les noms des colonnes

$> colnames(df)$

[1] "Status" "Seniority" "Home" "Time" "Age" "Marital"

[7] "Records" "Job" "Expenses" "Income" "Assets" "Debt"

[13] "Amount" "Price" "Finrat" "Savings" "seniorityR" "timeR"

[19] "ageR" "expensesR" "incomeR" "assetsR" "debtR" "amountR"

[25] "priceR" "finratR" "savingsR"

-Nous appliquons l'instruction suivante pour afficher le type de variable de chaque colonne

$> str(df)$

'data.frame' : 4446 obs. of 27 variables :

$ Status : chr "good" "good" "bad" "good" ...

$ Seniority : int 9 17 10 0 0 1 29 9 0 0 ...

$ Home : chr "rent" "rent" "owner" "rent" ...

$ Time : int 60 60 36 60 36 60 60 12 60 48 ...

$ Age : int 30 58 46 24 26 36 44 27 32 41 ...

$ Marital : chr "married" "widow" "married" "single" ...

$ Records : chr "no_rec" "no_rec" "yes_rec" "no_rec" ...

$ Job : chr "freelance" "fixed" "freelance" "fixed" ...

$ Expenses : int 73 48 90 63 46 75 75 35 90 90 ...

$ Income : int 129 131 200 182 107 214 125 80 107 80 ...

$ Assets : int 0 0 3000 2500 0 3500 10000 0 15000 0 ...

$ Debt : int 0 0 0 0 0 0 0 0 0 0 ...

$ Amount : int 800 1000 2000 900 310 650 1600 200 1200 1200 ...

$ Price : int 846 1658 2985 1325 910 1645 1800 1093 1957 1468 ...

$ Finrat : num 94.6 60.3 67 67.9 34.1 ...

$ Savings : num 4.2 4.98 1.98 7.93 7.08 ...

$ seniorityR : chr "sen (8,14]" "sen (14,99]" "sen (8,14]" "sen (-1,1]" ...

$ timeR : chr "time (48,99]" "time (48,99]" "time (24,36]" "time (48,99]" ...

$ ageR : chr "age (25,30]" "age (50,99]" "age (40,50]" "age (0,25]" ...

$ expensesR : chr "exp (60,80]" "exp (40,50]" "exp (80,1e+04]" "exp (60,80]" ...

$ incomeR : chr "inc (110,140]" "inc (110,140]" "inc (190,1e+04]" "inc (140,190]" ...

$ assetsR : chr "asset (-1,0]" "asset (-1,0]" "asset (0,3e+03]" "asset (0,3e+03]" ...

$ debtR : chr "debt (-1,0]" "debt (-1,0]" "debt (-1,0]" "debt (-1,0]" ...

$ amountR : chr "am (600,900]" "am (900,1.1e+03]" "am (1.4e+03,1e+05]" "am (600,900]" ...

$ priceR : chr "priz (0,1e+03]" "priz (1.5e+03,1.8e+03]" "priz (1.8e+03,1e+05]" "priz (1.3e+03,1.5e+03]" ...

$ finratR : chr "finr (90,100]" "finr (50,70]" "finr (50,70]" "finr (50,70]" ...

$ savingsR : chr "sav (4,6]" "sav (4,6]" "sav (0,2]" "sav (6,99]" ...

-Dans le but d'appliquer la méthode MCA, nous allons séparer les données qualificatives en utilisant le code suivant

$> df\_character <- df[, -c(9:27)]$ # suprimer les colonnes de 9 à 27

$> df\_character <- df\_character[, -c(2:2)]$

$> df\_character <- df\_character[, -c(3:3)]$

$> df\_character <- df\_character[, -c(3:3)]$

$> View(df\_character)$

$> dim(df\_character)$

[1]4446 5

$> colnames(df\_character)$

[1]"$Status$""$Home$""$Marital$""$Records$""Job"

	Status	Home	Marital	Records	Job
1	good	rent	married	no_rec	freelance
2	good	rent	widow	no_rec	fixed
3	bad	owner	married	yes_rec	freelance
4	good	rent	single	no_rec	fixed
5	good	rent	single	no_rec	fixed
6	good	owner	married	no_rec	fixed
7	good	owner	married	no_rec	fixed
8	good	parents	single	no_rec	fixed
9	good	owner	married	no_rec	freelance

Showing 1 to 9 of 4,446 entries, 5 total columns

.

FIG. 3.2 – Dataframe des données qualificatives df_character

3.2 Application du $MCA()$

. Nous allons appliquer l'objet $ACM()$ directement sur $df\_character$. Trois visualisations s'affichent automatiquement sans les appeller

$install.packages("corrplot")$

$library(corrplot)$

$install.packages("FactoMineR")$

$library("FactoMineR")$

$install.packages("ggfortify")$

$library("ggfortify")$

$instal.packages("FactoExtra")$

$library(factoextra)$

$mca <- -MCA(df\_character)$

Le premier graphique contient la représentation des variables :

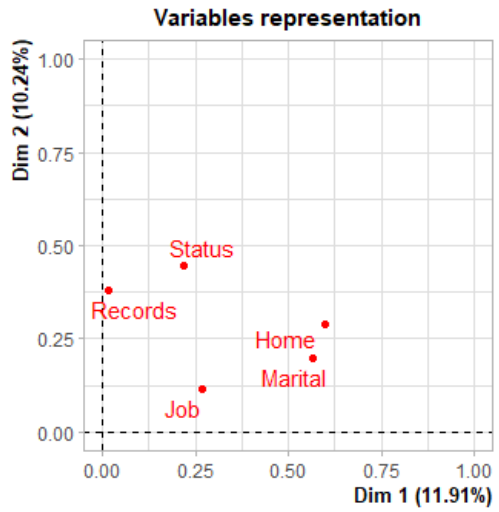


FIG. 3.3 – Représentation des variables - df_character

Le deuxième graphique contient la représentation des catégories, appelé aussi carte factorielle des catégories.

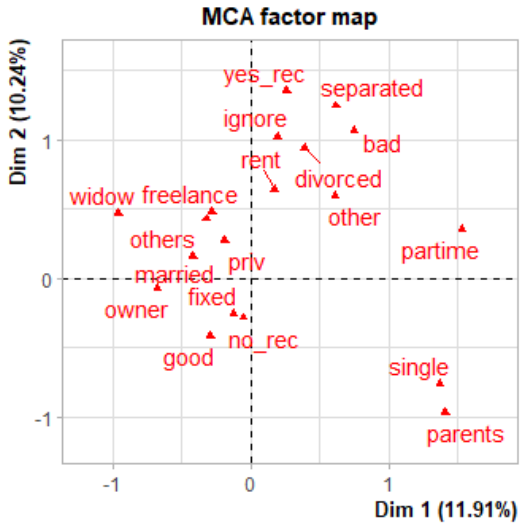


FIG. 3.4 – Représentation des catégories - df_character

Le troisième graphique contient la représentation des individus aussi appelé carte factorielle des individus

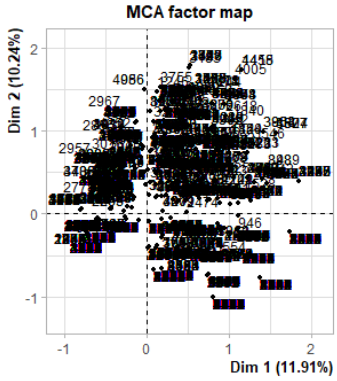


FIG. 3.5 – Représentation des individus - df_character

3.3 Visualisation et affichage des résultats

3.3.1 Pourcentage des variances

. Nous affichons avec l'instruction *acm$eig* un tableau qui contient les valeurs propres de la décompositions sdv, les pourcentages des variaces expliquées des composantes principales ainsi que les cumules correspondants

.

```
> acm$eig
       eigenvalue percentage of variance cumulative percentage of variance
dim 1  0.33334262              11.905093                           11.90509
dim 2  0.28673753              10.240626                           22.14572
dim 3  0.23567802               8.417072                           30.56279
dim 4  0.21409242               7.646158                           38.20895
dim 5  0.21091828               7.532796                           45.74175
dim 6  0.20673452               7.383376                           53.12512
dim 7  0.20150570               7.196632                           60.32175
dim 8  0.19386848               6.923874                           67.24563
dim 9  0.19052849               6.804589                           74.05022
dim 10 0.17837081               6.370386                           80.42060
dim 11 0.16980754               6.064555                           86.48516
dim 12 0.16584350               5.922982                           92.40814
dim 13 0.11557045               4.127516                           96.53566
dim 14 0.09700163               3.464344                          100.00000
```

. Dans le but de tracer le graphique qui représentent le pourcentenge de représentenstions des composantes principales (colonne deux du tableaux précedents),

nous exécutons le code R suivant :

$eig.val <- mca\$eig$

$barplot(eig.val[, 2], names.arg = 1 : nrow(eig.val), main = "Variances\ Explained\ by\ Dimensions\ (\%)", xlab = "Principal\ Dimensions", ylab = "Percentage\ ofv\ ariances", col = "steelblue")$

. Ce graphique représente les pourcentages de la variance expliquée de chaque composante

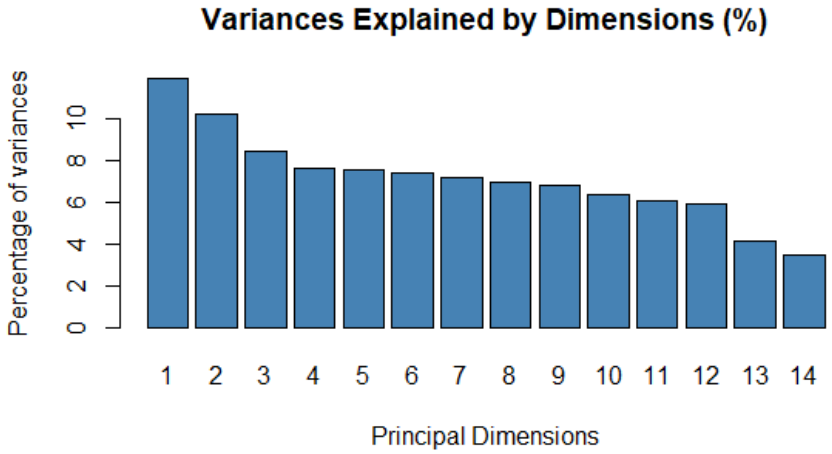
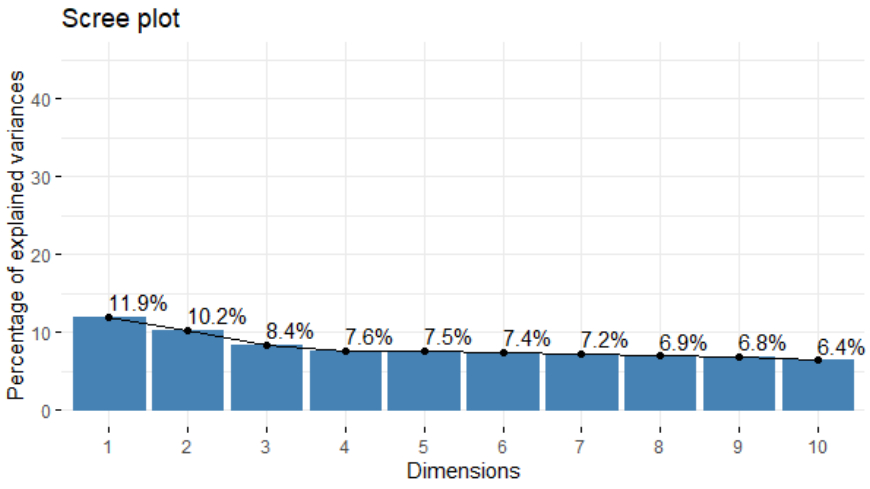


FIG. 3.6 – Représentation des pourcentages de la variance expliquée de chaque composante - df_character

. Dans le but de tracer le graphique qui représentent le pourcentenge de représentenstions des composantes principales (colonne deux du tableaux précedents), nous exécutons le code R suivant qui illustre ces pourcentages sur les bâtons du diagramme.

. $fviz_screeplot(mca, addlabels = TRUE, ylim = c(0, 45))$.

. Le graphique Scree plot présent les pourcentages de la variace expliquée dans les compsantes principales



.

FIG. 3.7 – Pourcentages de la variance expliquée dans les compsantes principales - df_character

3.3.2 Visualisation des variables

. Nous affichons avec l'instruction $acm\$var\$eta2$ un tableau qui contient les les coordonnées des variables ; Status, Home, Marital,Records et Job dans uniquement 5 dimensions par défaut au lieux des 14 coordonnées dans les14 dimensions.

.

```
> acm$var$eta2
             Dim 1     Dim 2       Dim 3       Dim 4        Dim 5
Status  0.21828229 0.4463621 0.007645585 0.012061705 0.0002206604
Home    0.60002363 0.2906684 0.046907839 0.387087330 0.4969726192
Marital 0.56475290 0.2000872 0.550578303 0.456787238 0.0751412943
Records 0.01394056 0.3809674 0.006916343 0.004278512 0.0286483384
Job     0.26971371 0.1156026 0.566342045 0.210247328 0.4536085088
```

. Nous exécutons le code R suivant pour la projection des variables dan un plan de Dim 1 $(11,91\%)$ et Dim 2 $(10,2\%)$ selon les coordonnées en haut

$var <- get\_mca\_var(mca)$

$fviz\_mca\_var(mca, choice = "mca.cor", repel = TRUE,)$

$ggtheme = theme\_minimal()$

. Carte factorielle des variables en deux dimensions

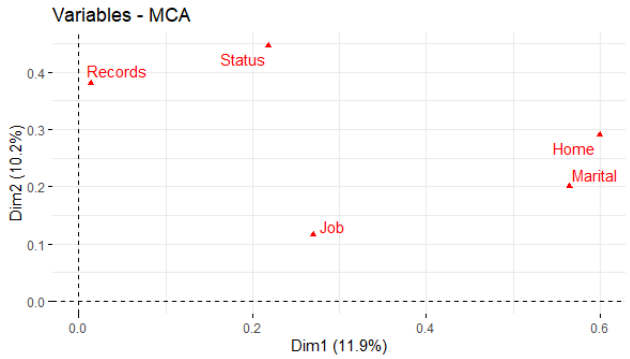


FIG. 3.8 – Carte factorielle des variables en deux dimensions - df_character

. Nous présentons ici un code R pour projeter les variables dans trois dimensions Dim1 $(11,91\%)$, Dim 2 $(10,2\%)$, et Dim 3 (8.41%)

$dataframe <- as.data.frame(mca\$var\$eta2)$

$colnames(dataframe) <- c("x","y","z","e","f")$

$install.packages("scatterplot3d")$

$library(scatterplot3d)$

$scatterplot3d(dataframe\$x, dataframe\$y, dataframe\$z, color = "blue"$

$, pch = 16, main = "Graphique\ en\ 3D\ des\ variables", xlab = "dime1(11.90)",$

$ylab = "dime2(10.24)", zlab = "dime3(8.41)")$

. Carte factorielle des variables en trois dimensions

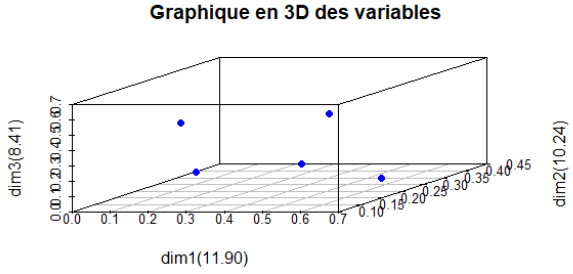


FIG. 3.9 – Carte factorielle des variables en trois dimensions - df_character

. Ce code est executé pour avoir une carte factorielle des variables en trois dimensions classificative

$install.packages("plot3D")$

$library(plot3D)$

$scatter3D(dataframe\$x, dataframe\$y, dataframe\$z, color = "blue", pch = 16, main = "Graphique\ en\ 3\ D\ des\ variables", xlab = "dime1(11.90)", ylab = "dime2(10.24)", zlab = "dime3(8.41)")$

. Carte factorielle des variables en trois dimensions classificative

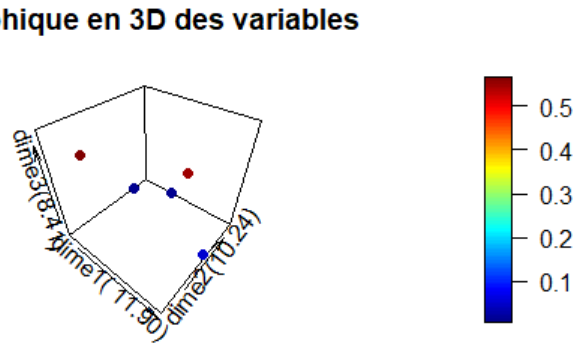


FIG. 3.10 – Carte factorielle des variables en trois dimensions classificative - df_character

3.3.3 Visualisation des catégories

. Nous affichons avec l'instruction $acm\$var\$coord$ un tableau qui contient les coordonnées des catégories ; bad, good, ignore,other, owner,parents,priv,rent,divorced, married, separated, single,widow,no rec, yes_rec, fixed, freelance, others et partiime dans uniquement 5 dimensions par défaut au lieux des 14 coordonnées dans les14 dimensions.

.

```
> acm$var$coord      #"coord. of the categories"
                Dim 1       Dim 2       Dim 3        Dim 4       Dim 5
bad          0.74747960  1.06889180  0.13989284 -0.175709181  0.02376579
good        -0.29202440 -0.41759332 -0.05465316  0.068645845 -0.00928479
ignore       0.19228176  1.02507180 -1.38720382 -4.373261846  2.19328323
other        0.61207087  0.59375126  0.18736699 -0.790362714  0.32133025
owner       -0.67480060 -0.07121264  0.13365115 -0.162055060 -0.38512633
parents      1.40813324 -0.96369464  0.08019964  0.184277234 -0.50835304
priv        -0.19080778  0.27214232 -0.28967809 -1.191790336  2.31278630
rent         0.17246899  0.64411915 -0.31341277  0.852985123  0.50698118
divorced     0.38968283  0.94265707 -1.56891824  2.744298100  2.23051406
married     -0.42053706  0.15712882 -0.13290906 -0.197696438  0.05169182
separated    0.61624601  1.24432563 -0.47570152  3.412825265 -0.71507996
single       1.36811740 -0.75843646  0.16898655  0.008998938 -0.20796128
widow       -0.96118164  0.47152610  5.78202555  1.245290851  0.64431460
no_rec      -0.05399536 -0.28226710  0.03803248 -0.029913191  0.07740448
yes_rec      0.25818069  1.34966986 -0.18185360  0.143030954 -0.37011217
fixed       -0.12371785 -0.25905160 -0.21320895  0.287120877  0.35540590
freelance   -0.28260717  0.48194776 -0.10265785 -0.354473168 -1.23023349
others      -0.32221694  0.42849402  3.72299309  0.338050617  0.61161985
partime      1.53086954  0.35649777  0.14591249 -1.110175985  0.34429858
```

. Nous exécutons le code R suivant pour la projection des catégories dan un plan de Dim 1 $(11,91\%)$ et Dim 2 $(10,2\%)$ selon les coordonnées en haut

. $library(factoextra)$.

$fviz\_mca\_var(mca, col.var = "cos2",$

$gradient.cols = c("\#00AFBB","\#E7B800","\#FC4E07",)$

$repel = TRUE, ggtheme = theme\_minimal())$

. Carte factorielle des catégories en deux dimensions

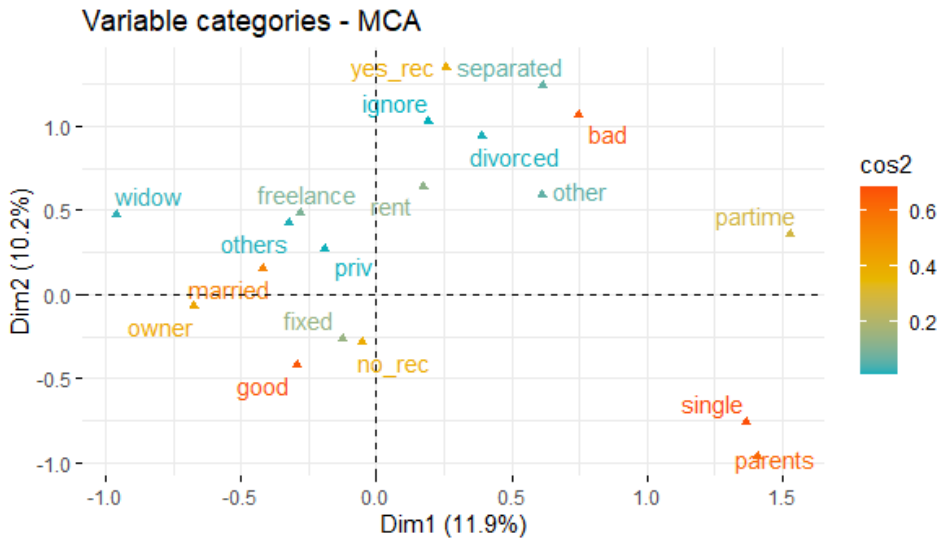


FIG. 3.11 – Carte factorielle des catégories en deux dimensions - df_character

. Nous présentons ici un code R pour projeter les catégories dans trois dimensions Dim1 $(11,91\%)$, Dim 2 $(10,2\%)$, et Dim 3 (8.41%)

$dataframe <- as.data.frame(mca\$var\$coord)$

$colnames(dataframe) <- c("x","y","z","e","f")$

$install.packages("scatterplot3d")$

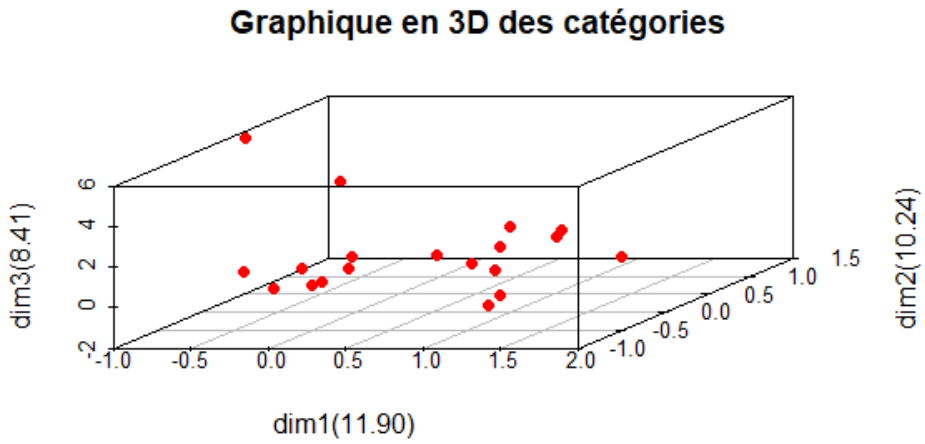
$library(scatterplot3d)$

$scatterplot3d(dataframe\$x, dataframe\$y, dataframe\$z,$

$color = "red", pch = 16, main = "Graphique\ en\ 3\ D\ des\ catégories",$

$xlab = "dime1(11.90)", ylab = "dime2(10.24)", zlab = "dime3(8.41)")$

. Carte factorielle des catégories en trois dimensions



.

FIG. 3.12 – Carte factorielle des catégories en trois dimensions - df_character

. Ce code est executé pour avoir une carte factorielle des catégories en trois dimensions classificative

$install.packages("plot3D")$

$library(plot3D)$

$scatter3D(dataframe\$x, dataframe\$y, dataframe\$z, color = "red"$

$, pch = 16, main = "Graphique\ en\ 3\ D\ des\ catégories",$

$xlab = "dime1(11.90)", ylab = "dime2(10.24)", zlab = "dime3(8.41)")$

. Carte factorielle des variables en trois dimensions classificative

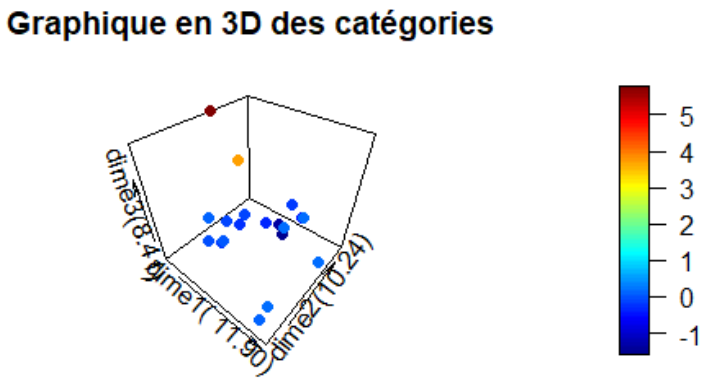


FIG. 3.13 – Carte factorielle des variables en trois dimensions classificative - df_character

3.3.4 Visualisation des individus

. Nous affichons avec l'instruction $acm\$ind\$coord$ un tableau qui contient les coordonnées des 4446 individus (6 uniquement sont affichés) dans uniquement 5 dimensions par défaut au lieux des 14 coordonnées dans les 14 dimensions.

.

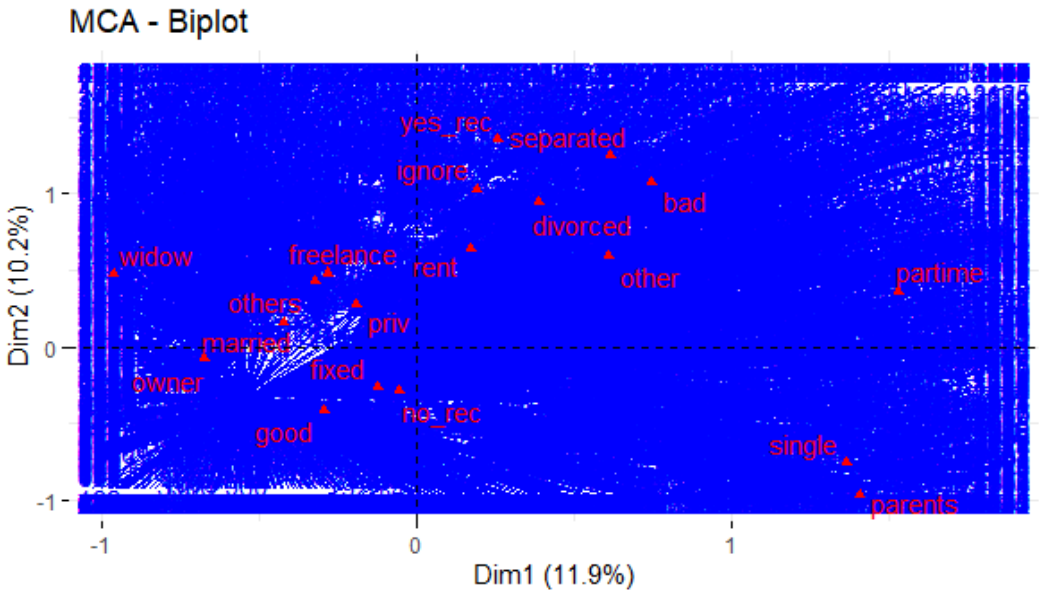
```
> mca$ind$coord
          Dim 1       Dim 2       Dim 3       Dim 4         Dim 5
1  -0.303691829  0.21787429 -0.233012990  0.146767786 -2.627892e-01
2  -0.435933892  0.05853947  2.158245651  1.047816335  6.858106e-01
3  -0.128961349  1.11542256 -0.059273476 -0.322844572 -8.317820e-01
4   0.370947732 -0.40084850 -0.154183909  0.513436114  3.146575e-01
5   0.370947732 -0.40084850 -0.154183909  0.513436114  3.146575e-01
6  -0.542150430 -0.32606178 -0.094378249 -0.014652205  3.923328e-02
```

. Nous exécutons le code R suivant pour la projection Biplot des individus et des catégories dans un plan de Dim 1 $(11,91\%)$ et Dim 2 $(10,2\%)$ selon les coordonnées en haut

$fvizmcabiplot(mca, repel = TRUE,$

$ggtheme = thememinimal())$

. Graphique Biplot (individus et catégories) où les individus sont liés par les lignes bleu



.

FIG. 3.14 – Graphique Biplot (individus et catégories) - df_character

. Nous présentons ici un code R pour projeter les individus dans trois dimensions Dim1 $(11,91\%)$, Dim 2 $(10,2\%)$, et Dim 3 (8.41%)

$dataframe <- as.data.frame(acm\$ind\$coord)$

$colnames(dataframe) <- c("x","y","z","e","f")$

$install.packages("scatterplot3d")$

$library(scatterplot3d)$

$scatterplot3d(dataframe\$x, dataframe\$y, dataframe\$z, color = "green"$

$, pch = 16, main = "Graphique\ en\ 3\ D\ des\ individus", xlab = "dime1(11.90)",$

$ylab = "dime2(10.24)", zlab = "dime3(8.41)")$

. Carte factorielle des individus en trois dimensions

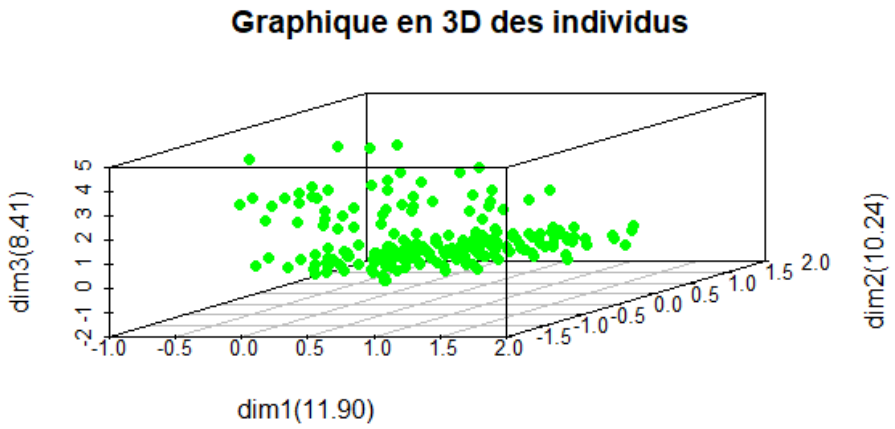


FIG. 3.15 – Carte factorielle des individus en trois dimensions - df_character

. Ce code est executé pour avoir une carte factorielle des individus en trois dimensions classificative

$install.packages("plot3D")$

$library(plot3D)$

$scatter3D(dataframe\$x, dataframe\$y, dataframe\$z, color = "green",$

$pch = 16, main = "Graphique\ en\ 3\ D\ des\ individus", xlab = "dime1(11.90)",$

$ylab = "dime2(10.24)", zlab = "dime3(8.41)").$

. Carte factorielle des individus en trois dimensions classificative

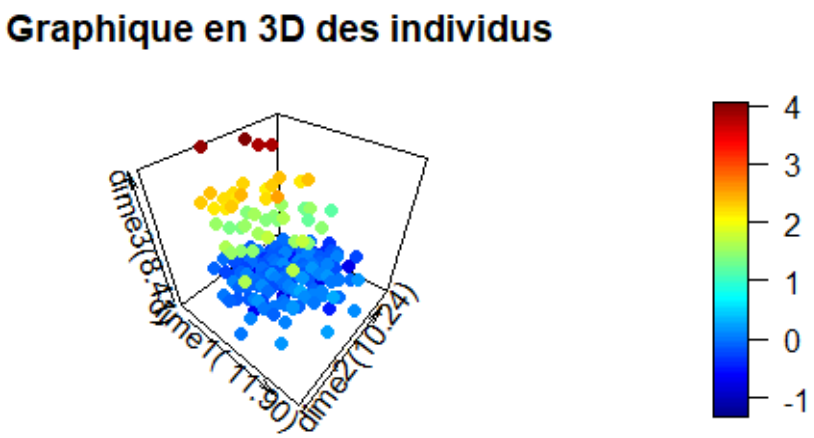


FIG. 3.16 – Carte factorielle des individus en trois dimensions classificative - df_character

3.3.5 Visualisation de la contribution des variables

. La contribution des catégories variables (en %) à la définition des dimensions peut être extraite comme suit :

.

```
> mca$var$contrib
                Dim 1       Dim 2       Dim 3       Dim 4       Dim 5
bad         9.417395886 22.3875167  0.46654639  0.810234232  0.015045732
good        3.679176560  8.7463273  0.18226977  0.316541306  0.005878048
ignore      0.009978744  0.3296967  0.73460181  8.037115289  2.051945109
other       1.612737939  1.7643136  0.21375600  4.186998549  0.702490120
owner      12.941332009  0.1675516  0.71803437  1.162098120  6.662099255
parents    20.924927164 11.3936176  0.09600481  0.557968089  4.310065949
priv        0.120864005  0.2858270  0.39401055  7.341656585 28.064184476
rent        0.390575129  6.3331717  1.82426392 14.874932158  5.333874867
divorced    0.077871013  0.5297452  1.78535946  6.013196648  4.032179301
married     7.727785990  1.2541935  1.09176071  2.659094505  0.184529902
separated   0.666225558  3.1578205  0.56150533 31.814909985  1.417747467
single     24.577022890  8.7806617  0.53034511  0.001655597  0.897478581
widow       0.835324913  0.2337019 42.75395212  2.183110092  0.593221733
no_rec      0.144669260  4.5961072  0.10151830  0.069131879  0.469863907
yes_rec     0.691741052 21.9764448  0.48541323  0.330556461  2.246670463
fixed       0.578971988  2.9510166  2.43206627  4.855253187  7.551246902
freelance   1.100428401  3.7205031  0.20537687  2.695576839 32.956959517
others      0.239586608  0.4925628 45.23994011  0.410599879  1.364286447
partime    14.263384891  0.8992205  0.18327487 11.679370601  1.140232225
```

- Le code R ci-dessous montre les 15 principales catégories de variables contribuant aux dimensions
- $fviz_contrib(mca, choice = "var", axes = 1, top = 15)$.

  $fviz_contrib(mca, choice = "var", axes = 2, top = 15)$.
- Graphique à barres de la contribution des catégories de variables où les catégories de variables avec la valeur la plus élevée contribuent le plus à la définition des dimensions. Les catégories de variables qui contribuent le plus à Dim.1 et Dim.2 sont les plus importantes pour expliquer la variabilité de l'ensemble de données.

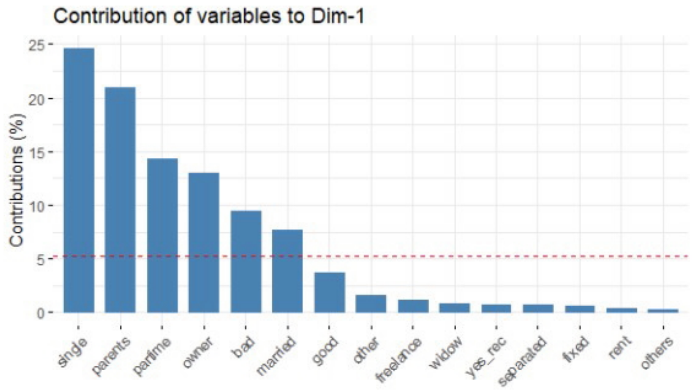


FIG. 3.17 – Contribution des catégories de variables à Dim.1 - df_character

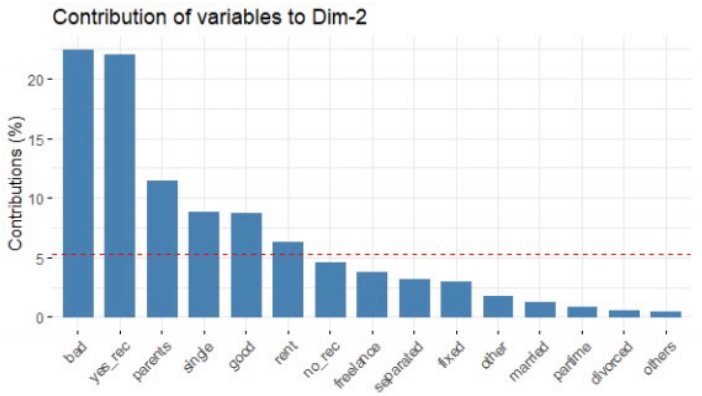


FIG. 3.18 – Contribution des catégories de variables à Dim.2 - df_character

3.3.6 Visualisation de la qualité de représentation

. Vous pouvez visualiser le *cos*2 des catégories de lignes sur toutes les dimensions à l'aide du package corrplot

```
> mca$var$cos2
                Dim 1       Dim 2       Dim 3       Dim 4       Dim 5
bad        0.2182822860 0.446362076 0.007645585 0.0120617052 0.0002206604
good       0.2182822860 0.446362076 0.007645585 0.0120617052 0.0002206604
ignore     0.0001670686 0.004748180 0.008695592 0.0864230419 0.0217374213
other      0.0289574047 0.027249925 0.002713578 0.0482846515 0.0079810392
owner      0.4098202593 0.004564116 0.016076367 0.0236356582 0.1334900617
parents    0.4231933084 0.198212106 0.001372765 0.0072476073 0.0551546513
priv       0.0021324458 0.004337884 0.004914927 0.0831927605 0.3132974273
rent       0.0083335511 0.116235953 0.027519557 0.2038407320 0.0720098211
divorced   0.0013090750 0.007660365 0.021219866 0.0649238971 0.0428895947
married    0.4740437817 0.066179216 0.047349901 0.1047629208 0.0071623165
separated  0.0114385284 0.046636936 0.006816022 0.3508245871 0.0154017878
single     0.5243904727 0.161155994 0.008000412 0.0000226877 0.0121163832
widow      0.0141354875 0.003401820 0.511516763 0.0237269247 0.0063517852
no_rec     0.0139405601 0.380967394 0.006916343 0.0042785123 0.0286483384
yes_rec    0.0139405601 0.380967394 0.006916343 0.0042785123 0.0286483384
fixed      0.0261126105 0.114487507 0.077552605 0.1406420145 0.2154938100
freelance  0.0238084716 0.069241282 0.003141590 0.0374569059 0.4511700427
others     0.0041529502 0.007344285 0.554427102 0.0045711288 0.0149631537
partime    0.2645672724 0.014347406 0.002403503 0.1391372501 0.0133822833
```

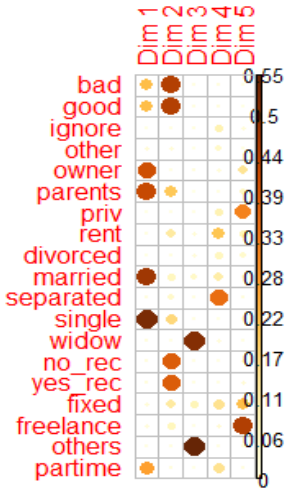
. La visualisation de *cos*2 s'obtient en executant le code R suivant :

. *library*("*corrplot*").

corrplot(*var*cos2, is.corr=FALSE).

. Visualiser de *cos*2

.



. Il est également possible de créer un diagramme à barres de la variable cos2 en executant le code suivant

. $cos2_var <- mca$varcos2[, 1 : 2].

.

```
> cos2_var
                Dim 1       Dim 2
bad        0.2182822860 0.446362076
good       0.2182822860 0.446362076
ignore     0.0001670686 0.004748180
other      0.0289574047 0.027249925
owner      0.4098202593 0.004564116
parents    0.4231933084 0.198212106
priv       0.0021324458 0.004337884
rent       0.0083335511 0.116235953
divorced   0.0013090750 0.007660365
married    0.4740437817 0.066179216
separated  0.0114385284 0.046636936
single     0.5243904727 0.161155994
widow      0.0141354875 0.003401820
no_rec     0.0139405601 0.380967394
yes_rec    0.0139405601 0.380967394
fixed      0.0261126105 0.114487507
freelance  0.0238084716 0.069241282
others     0.0041529502 0.007344285
partime    0.2645672724 0.014347406
```

. Le code suivant calcul les pourcentages de la qualité de la représentation des variables dans les Dim-1-2

. $heights <- rowSums(cos2_var)$.

.

```
> heights
       bad        good      ignore       other       owner     parents        priv
0.664644362 0.664644362 0.004915248 0.056207329 0.414384376 0.621405414 0.006470330
      rent    divorced     married   separated      single       widow      no_rec
0.124569504 0.008969440 0.540222998 0.058075464 0.685546466 0.017537308 0.394907954
    yes_rec       fixed   freelance      others     partime
0.394907954 0.140600117 0.093049754 0.011497235 0.278914679
```

. On execute le code suivant pour obtenir le graphique de la qualité de la représentation des variables dans les Dim-1-2

. $fviz_cos2(mca, choice = "var", axes = 1 : 2)$.

. Diagramme à barres de la variable $cos2$ où les catégories de variables Priv, Other,

Others, Divorced, Ignore et Widow ne sont pas très bien représentées par les deux premières dimensions. Cela implique que la position des points correspondants sur le nuage de points doit être interprétée avec une certaine prudence. Une solution dimensionnelle plus élevée est probablement nécessaire.

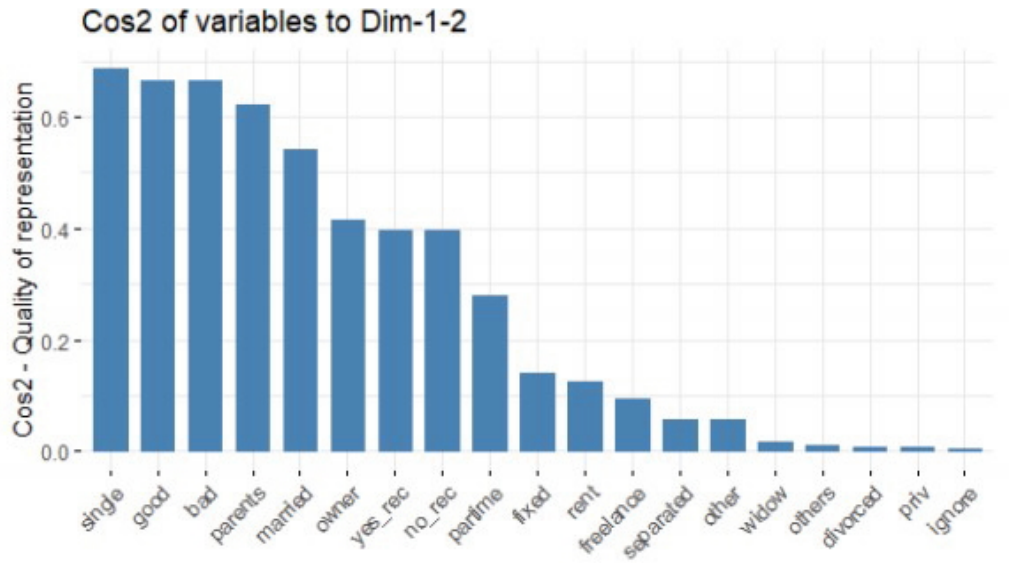


FIG. 3.19 – Diagramme à barres de la la qualité de représentation - df_character

. Nombre de fois de modalités dans chaque variables

```
> table(df_character$Status)

 bad good
1249 3197
> table(df_character$Home)

 ignore   other   owner parents    priv    rent
     20     319    2106     782     246     973
> table(df_character$Marital)

 divorced   married separated    single     widow
       38      3238       130       973        67
> table(df_character$Records)

 no_rec yes_rec
   3677     769
> table(df_character$Job)

    fixed freelance    others   partime
     2803      1021       171       451
>
```

. plotellipses(mca)

.

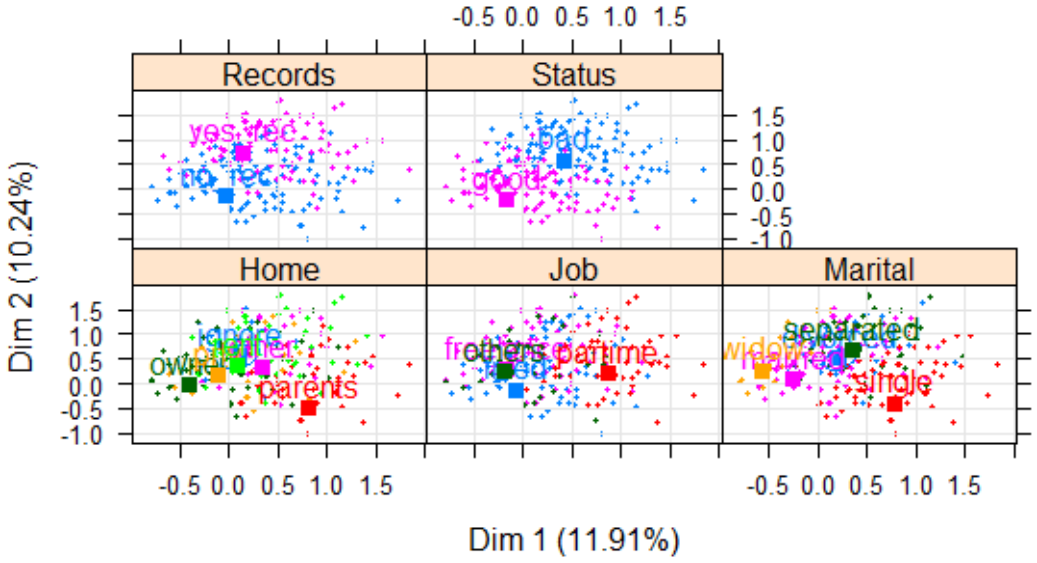


FIG. 3.20 – Plotellipses(mca)

3.4 Rôle de l'ACM dans la classification

L'ACM (Analyse des correspondances multiples) n'est pas directement impliquée dans la classification des données, mais elle peut être utilisé comme une étape préliminaire importante dans le processus de classification.

L'ACM est une technique statistique qui permet de réduire la dimensionnalité des données en les projetant sur un espace de variables moins nombreuses appelées les composantes principales.

Son rôle principal est de révéler la structure sous-jacente des données en identifiant les relations et les corrélations entre les variables. En utilisant la réduction de dimensionnalité, l'ACM permet de simplifier les données tout en conservant une grande partie de l'information. Cela facilite la visualisation et l'interprétation des données,

en particulier lorsque les données sont complexes et comportent de nombreuses variables.

Dans le contexte de la classification, l'ACM peut être utilisé pour extraire les caractéristiques les plus discriminantes des données. Les composantes principales obtenues à partir de l'ACM peuvent servir de nouvelles variables qui capturent efficacement les variations les plus importantes dans les données d'origine. Ces nouvelles variables peuvent ensuite être utilisées comme entrées pour des algorithmes de classification tels que les arbres de décision, les réseaux de neurones, les machines à vecteurs de support (SVM) ou d'autres méthodes de classification.

Dans la section suivantes nous allons exécuter deux modèles de machine learnig et un modèle de deep learning sur les sorties de l'ACM en créant deux Dataframes *mca.machine* et *mca.deep* selon nos objectifs de prédiction.

3.4.1 Machine Learning

Le machine learning est une approche de l'intelligence artificielle qui permet aux ordinateurs d'apprendre à partir des données et de prendre des décisions ou de faire des prédictions sans être émises programmées.

. Le code suivant permettra de créer un fichier .txt contenant un Dataframe des coordonnées des 4446 individus obtenus par l'ACM. Le but est de faire un clustering dans le but d'avoir une idée sur le nombre de classe dont les clients appartiennent, ce qui permettra par exemple de proposer des offres selon les differences des individus par clusters. Par la suite nous allons appliquer deux modèles, l'algorithme de clustering hiérarchique et l'algorithme K-means.

. $chemin\_fichier <-"C:/Users/hp/Desktop/Asma-memoire/mca.machine.txt"$

$write.table(mca\$ind\$coord, chemin\_fichier, sep = ",", row.names = FALSE)$

	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5
0	-0.303692	0.217874	-0.233013	0.146768	-0.262789
1	-0.435934	0.058539	2.158246	1.047816	0.685811
2	-0.128961	1.115423	-0.059273	-0.322845	-0.831782
3	0.370948	-0.400848	-0.154184	0.513436	0.314657
4	0.370948	-0.400848	-0.154184	0.513436	0.314657

.

L'algorithme de clustering hiérarchique

L'algorithme de clustering hiérarchique est une méthode de regroupement non supervisée utilisée pour classer un ensemble de données en une structure arborescente. Il fonctionne en fusionnant itérativement les groupes les plus similaires en un seul groupe, jusqu'à ce que tous les points de données soient regroupés.

. Charger le Dataframe sur Colab ensuite exécuter l'algorithme sur les coordonnées des individus calculées par ACM. Le nombre de cluster prédis est 3. (Voir Annexe 1)

.

	0
0	0
1	2
2	0
3	1
4	1
...	...
4441	0
4442	0
4443	0
4444	1
4445	0

4446 rows × 1 columns

L'algorithme K-means

L'algorithme $K-means$ est une méthode de regroupement non supervisée utilisée pour classer un ensemble de données en plusieurs groupes distincts. Il fonctionne en itérativement en attribuant chaque point de données à un groupe en fonction de sa similarité avec les centroïdes des groupes précédemment formés.

. Charger le Dataframe sur Colab ensuite exécuter l'algorithme sur les coordonnées des individus calculées par ACM. Le nombre de cluster prédis est 3. (Voir Annexe 2)

.

```
kmeans.predict(df)
```

```
array([0, 0, 1, ..., 1, 2, 0], dtype=int32)
```

3.4.2 Deep Learning

Le deep learning est une branche du machine learning qui se concentre sur l'utilisation de réseaux de neurones artificiels profonds pour résoudre des problèmes complexes en apprenant de façon automatique à partir des données.

Création d'un Dataframe

. Le code suivant permettra de créer un fichier .txt contenant un Dataframe des coordonnées des 4446 individus obtenus par ACM. En concaténant la colonne Status des données originaux, le but est d'entrainer le modèle DNN pour prédire le Status d'un nouvel individu (good où bad) pour décider de la possibilité de lui accorder un crédit bancaire ou non

. $df\_character\$Status < -NULL$

$mca < -MCA(df\_character)$

$dataframe1 <- as.data.frame(mca\$ind\$coord)$

$dataframe2 <- data.frame(df\$Status)$

$dataframe\_concat\_colonnes <- cbind(dataframe1, dataframe2)$

$chemin\_fichier <- "C:/Users/hp/Desktop/Asma-memoire/mca.deep.txt"$

$write.table(dataframe\_concat\_colonnes, chemin\_fichier, sep = ",", row.names = FALSE)$

	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5	df.Status
0	-0.359169	-0.364699	0.224478	-0.272337	0.022775	good
1	-0.455043	2.040015	1.730230	0.883150	0.195515	good
2	-0.699729	-0.367044	0.285403	-1.008596	0.245814	bad
3	0.604622	-0.157214	0.227406	0.449564	-0.258886	good
4	0.604622	-0.157214	0.227406	0.449564	-0.258886	good

.

Deep Neural Network

(Deep Neural Network) est un type de réseau neuronal artificiel composé de plusieurs couches de neurones, également appelées couches cachées. Chaque couche est connectée à la suivante, formant un réseau profond

Le modèle DNN

. Nous allons charger le Dataframe crée en haut sur Colab en premier lieu (Voir Annexe 3)

. Entrainement. : sur ces trois figures on résume les trois étapes importantes du modéle DNN que nous allons entrainé sur notre Dataframe de 6646 individus

.

.Construction du modèle

```
[41] model = Sequential([
       Dense(64, activation='relu', input_shape=(5,)),
       Dense(128, activation='relu'),
       Dense(2, activation='softmax')])
```

.

.Compilation du modèle

```
[42] model.compile(
       optimizer='adam',
       loss='categorical_crossentropy',
       metrics=['accuracy'])
```

.

.Entrainement du modèle

```
history = model.fit(
  X_train,
  Y_train,
  epochs=5,
  batch_size=32,
  validation_data=(X_test, Y_test))
```

Présision du modèle Aprés l'entrainement du modèle sur nos individus il a donné une présesion de 0.75 ce qui veut dire que notre modèle prédit avec une exacttitude de 0.75

```
14/14 [==============================] - 0s 2ms/step - loss: 0.5198 - accuracy: 0.7551
[0.5198062062263489, 0.7550562024116516]
```

Prédiction

. On veut prédire le Status d'un nouvel individi qui a donné les réponses suivantes :

.

	Home	Marital	Records	Job
1	owner	single	n_rec	fixed

. On cherche premièrement à calculer les coordonnées de cet nouvel individu dans les composantes principales dans le but de les introduire dans le modèle DNN pour predire son "$Status$".

. Le code suivant nous donne les coordonnées de cet nouvel individu dans les 5 composantes principales par ACM

. $nouvel\_individu <- data.frame($

$Home = "owner",$

$Marital = "single",$

$Records = "n\_rec",$

$Job = "fixed")$

$df\_character <- rbind(nouvel\_individu, df\_character)$

$row.names(df\_character) <- NULL$

$library("FactoMineR")$

$library("ggfortify")$

$mca <- MCA(df\_character)$

$print(nouvel\_individu)$

$cordonnées <- as.data.frame(mca\$ind\$coord[1,])$

.

```
> print(nouvel_individu)
   Home Marital Records   Job
1 owner  single   n_rec fixed
> print(mca$ind$coord[1, ])
     Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
 0.9856072  1.1750383 -1.9758729 -1.7555479 -5.0619728
```

. Le modèle a prédit "*good*"

.

```
1/1 [==============================] - 0s 136ms/step
Raw Prediction Output (Probabilities) : [[1.33274245e-08 1.00000000e+00]]
Prediction is  ['good']
```

Conclusion

L'analyse en correspondance multiple est une méthode d'analyse statistique puissante pour étudier les relations entre les variables catégorielles. Elle offre une approche visuelle et interprétable pour explorer et comprendre les structures cachées dans les données qualitatives. Elle sert à :

. Réduction de dimensionnalité des données catégorielles en gardant l'information des individus, pour pouvoir manipuler les plus grandes data frame

. Représentation graphique interprétable des distances entre les catégories, et les individus ainsi que les correlations

. Utilisation des sorties de l'ACM en Datascience cela en les entrainant sur des modèles de classification en Machine Learning comme k.means et en Deep learning comme par exemple ANN, dans le but d'un déploiement d'outils de prédiction avec des hautes précisions.qui sert à la prise de décision

Bibliographie

[1] https ://colab.research.google.com/drive/1JAMEL_BaMAsgnOdm3opVxfByOqBt5Uu_#scr

[2] https ://rdrr.io/cran/FactoMineR/man/MCA.html

[3] http ://factominer.free.fr/factomethods/analyse-des-correspondances-multiples.html

[4] https ://rstudio-pubs-static.s3.amazonaws.com/388792_9446f74a169445118b5a244ecc21e9e7.

[5] https ://fxjollois.github.io/cours-2017-2018/analyse-donnees/Investigate-ACM.html

[6] http ://www.sthda.com/english/wiki/factoextra-r-package-easy-multivariate-data-analyses-and-elegant-visualization

[7] Livre : Analyse des données avec R, F. Husson, S. Lê, J. Pagès, éditions PUR

[8] Marin, J. M. (2005). Initiation au logiciel R. Université Paris Dauphine

[9] Marsaglia, G. (2004). Evaluating the normal distribution. Journal of Statistical Software, 11, 1-11.

[10] R package : FactoMineR.

[11] Robert, C. P., Casella, G., & Casella, G. (1999). Monte Carlo statistical methods(Vol. 2). New York : Springer.

[12] Ressources pédagogiques :http ://www.math.u-bordeaux.fr/~mchave100p/teaching/

[13] Saporta, G. (2006). Probabilités, analyse des données et statistique. Editions technip.

[14] Vidéo sur l'ACM : https ://youtu.be/mUKz4L2ZsuY

Annexe : Abréviations et Notations

1.Chargement des donées et exploration

```
# Lecture du fichier texte
with open('mca.machine.txt', 'r') as fichier:
    lignes = fichier.readlines()

# Affichage des premières lignes
nb_lignes_a_afficher = 10 # Nombre de lignes à afficher
for i in range(nb_lignes_a_afficher):
    print(lignes[i])
```

```
 "Dim 1","Dim 2","Dim 3","Dim 4","Dim 5"

 -0.303691829359047,0.217874294965367,-0.233012990087509,0.146767785894659,-0.262789225266439

 -0.435933891756252,0.0585394730469498,2.15824565062018,1.04781633489336,0.685810576409709

 -0.128961349046032,1.11542256279812,-0.0592734764454294,-0.322844571847536,-0.831781998976543

 0.370947731721497,-0.400848497204644,-0.1541839090033,0.51343611368935,0.314657490810929

 0.370947731721703,-0.400848497204532,-0.154183909003381,0.513436113689432,0.31465749081134

 -0.542150430098447,-0.326061784500946,-0.094378249120426,-0.01465220522824,0.0392332814774307

 -0.542150430098238,-0.3260617845017,-0.0943782491200314,-0.0146522052290101,0.0392332814764687

 0.798988425383,-1.00136296588397,0.00797440954376478,0.224390892186233,-0.127505007417166

 -0.597190537150106,-0.0493003576098253,-0.0488340042596338,-0.291977620311836,-0.651288378382046
```

```
import pandas as pd

# Lecture du fichier texte
donnees = pd.read_csv('mca.machine.txt', delimiter=',')  # Remplacez "nom_du_fichier.txt" par le nom réel de votre fichi

# Écriture des données dans un fichier CSV
donnees.to_csv('data.csv', index=False)  # Remplacez "nom_du_fichier.csv" par le nom souhaité pour votre fichier CSV
```

```
import pandas as pd
import numpy as np
df=pd.read_csv("data.csv", sep=",")
df.head()
```

	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5
0	-0.303692	0.217874	-0.233013	0.146768	-0.262789
1	-0.435934	0.058539	2.158246	1.047816	0.685811
2	-0.128961	1.115423	-0.059273	-0.322845	-0.831782
3	0.370948	-0.400848	-0.154184	0.513436	0.314657
4	0.370948	-0.400848	-0.154184	0.513436	0.314657

2.Clustering hiérarchique

```
#Importing required modules
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.cluster import AgglomerativeClustering
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import scipy.cluster.hierarchy as shc
```

```
#Applying agglomerative algorithm with 5 clusters, using euclidean distance as a metric
model=AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')
labels=model.fit_predict(df)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_agglomerative.py:983: FutureWarning: Attribute `affinity`
  warnings.warn(
```

```
#representation dataframe
agglomerative=pd.DataFrame(labels)
agglomerative
```

	0
0	0
1	2
2	0
3	1
4	1
...	...
4441	0
4442	0
4443	0
4444	1
4445	0

4446 rows × 1 columns

3.Tracer les grappes

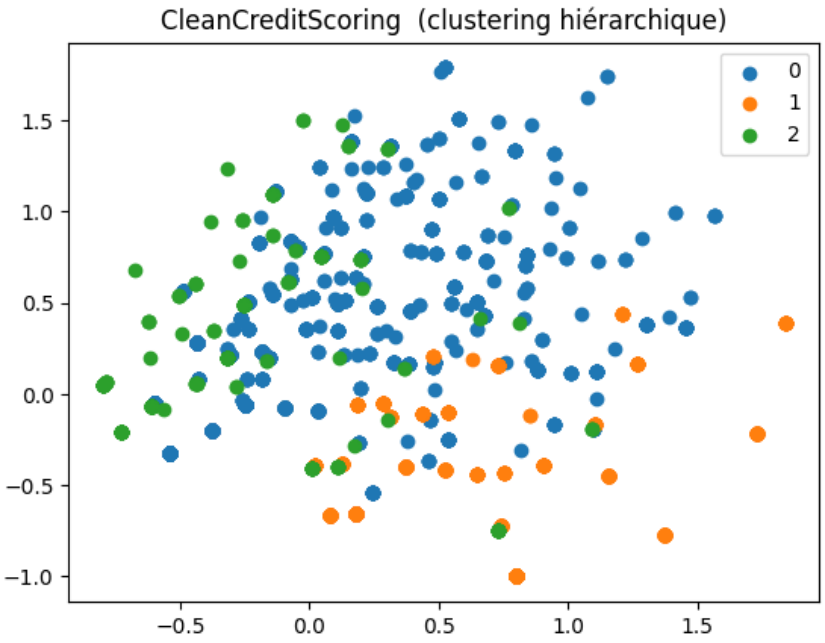
```
#calcul du pca pour préparer le tracé des grappes
pca = PCA(2)

#Transform the data
df = pca.fit_transform(df)
df.shape
df
```

```
array([[-0.30369183,  0.21787429],
       [-0.43593389,  0.05853947],
       [-0.12896135,  1.11542256],
       ...,
       [ 0.39109721,  0.45904289],
       [ 0.31590762, -0.12408707],
       [-0.59719054, -0.04930036]])
```

```
#Getting unique labels
u_labels = np.unique(labels)
print(u_labels)
```

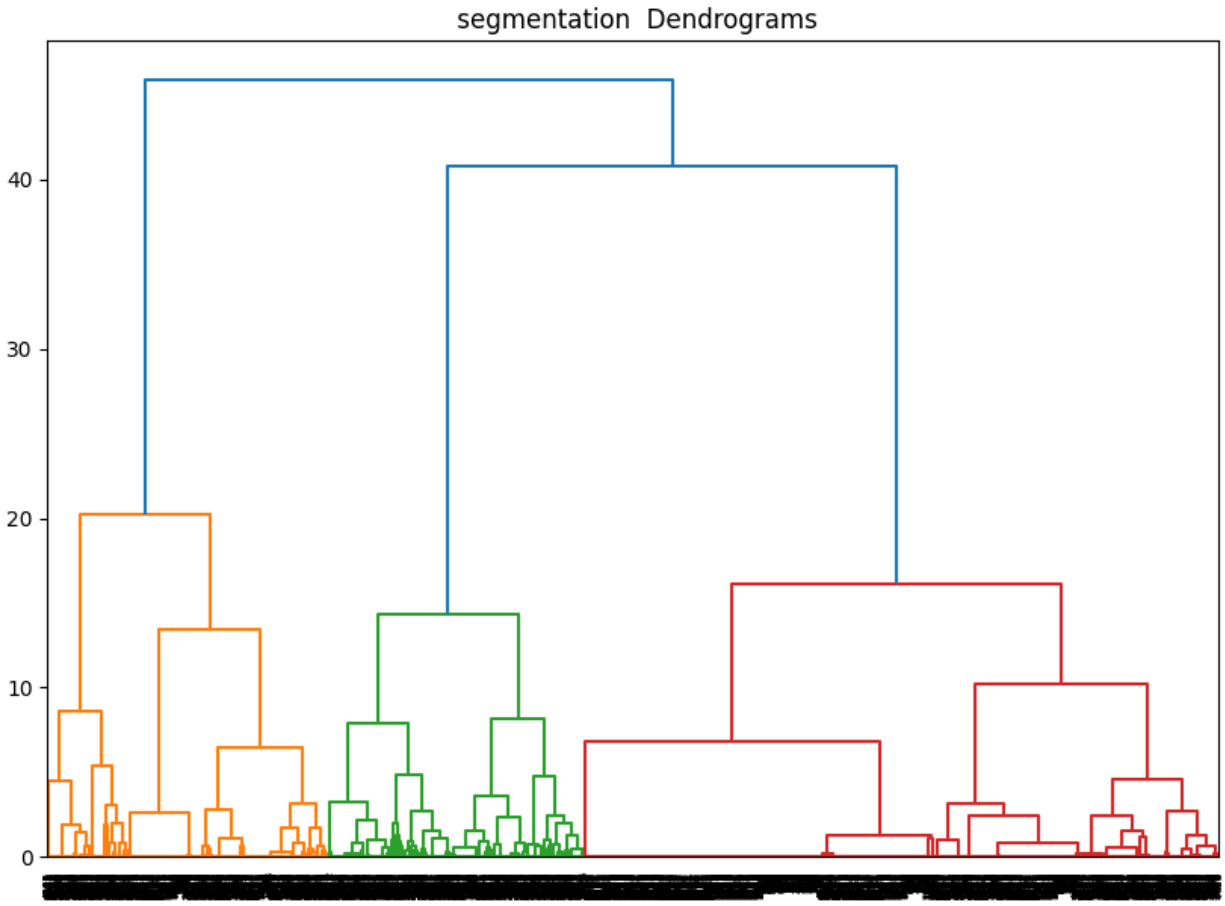
```
[0 1 2]
```

```
#plotting the clusters:
for i in u_labels:
    plt.scatter(df[labels == i , 0] , df[labels == i , 1] , label = i)
    plt.title("CleanCreditScoring  (clustering hiérarchique)")

plt.legend()
plt.show()
```



4.Tracez le dendrogramme

```
plt.figure(figsize=(10,7))
plt.title(" segmentation  Dendrograms")
dend=shc.dendrogram(shc.linkage(df, method="ward"))
```



✓ 59 s terminée à 01:27

1.Chargement des donées et exploration

```
# Lecture du fichier texte
with open('mca.machine.txt', 'r') as fichier:
    lignes = fichier.readlines()

# Affichage des premières lignes
nb_lignes_a_afficher = 10 # Nombre de lignes à afficher
for i in range(nb_lignes_a_afficher):
    print(lignes[i])
```

```
 "Dim 1","Dim 2","Dim 3","Dim 4","Dim 5"

 -0.303691829359047,0.217874294965367,-0.233012990087509,0.146767785894659,-0.262789225266439

 -0.435933891756252,0.0585394730469498,2.15824565062018,1.04781633489336,0.685810576409709

 -0.128961349046032,1.11542256279812,-0.0592734764454294,-0.322844571847536,-0.831781998976543

 0.370947731721497,-0.400848497204644,-0.1541839090033,0.51343611368935,0.314657490810929

 0.370947731721703,-0.400848497204532,-0.154183909003381,0.513436113689432,0.31465749081134

 -0.542150430098447,-0.326061784500946,-0.094378249120426,-0.01465220522824,0.0392332814774307

 -0.542150430098238,-0.3260617845017,-0.0943782491200314,-0.0146522052290101,0.0392332814764687

 0.798988425383,-1.00136296588397,0.00797440954376478,0.224390892186233,-0.127505007417166

 -0.597190537150106,-0.0493003576098253,-0.0488340042596338,-0.291977620311836,-0.651288378382046
```

```
import pandas as pd

# Lecture du fichier texte
donnees = pd.read_csv('mca.machine.txt', delimiter=',')  # Remplacez "nom_du_fichier.txt" par le nom réel de votre fichi

# Écriture des données dans un fichier CSV
donnees.to_csv('data.csv', index=False)  # Remplacez "nom_du_fichier.csv" par le nom souhaité pour votre fichier CSV
```

```
import pandas as pd
import numpy as np
df=pd.read_csv("data.csv", sep=",")
df.head()
```

	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5
0	-0.303692	0.217874	-0.233013	0.146768	-0.262789
1	-0.435934	0.058539	2.158246	1.047816	0.685811
2	-0.128961	1.115423	-0.059273	-0.322845	-0.831782
3	0.370948	-0.400848	-0.154184	0.513436	0.314657
4	0.370948	-0.400848	-0.154184	0.513436	0.314657

2.Algorithme K.means

```
#Importing required modules
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
```

```
# effectuer pca
pca = PCA(2)
```

```
#Transform the data
df = pca.fit_transform(df)
df.shape
df
```

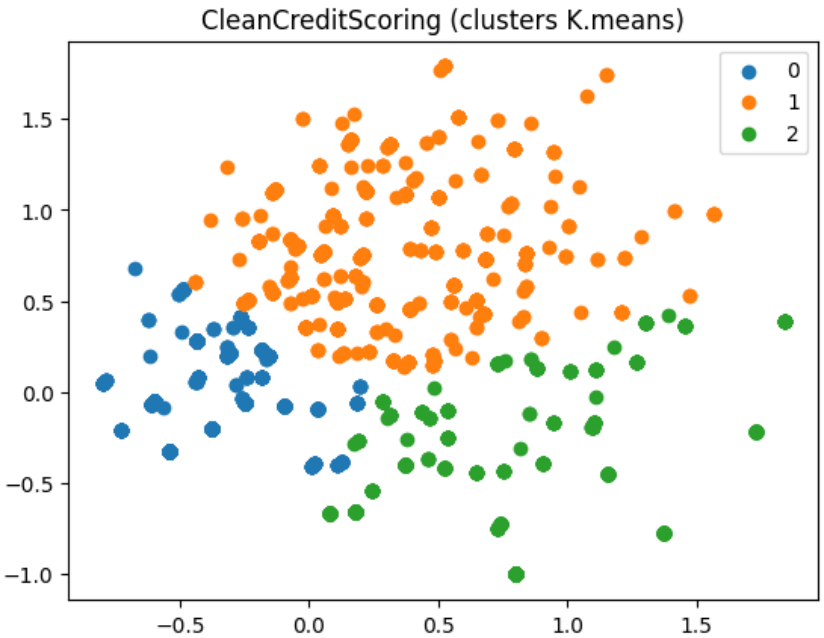
```
array([[-0.30369183,  0.21787429],
       [-0.43593389,  0.05853947],
       [-0.12896135,  1.11542256],
       ...,
       [ 0.39109721,  0.45904289],
       [ 0.31590762, -0.12408707],
       [-0.59719054, -0.04930036]])
```

```
#Initialize the class object
kmeans = KMeans(n_clusters= 3, random_state=0)

#predict the labels of clusters.
label = kmeans.fit_predict(df)

print(label)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_ini
  warnings.warn(
[0 0 1 ... 1 2 0]
```

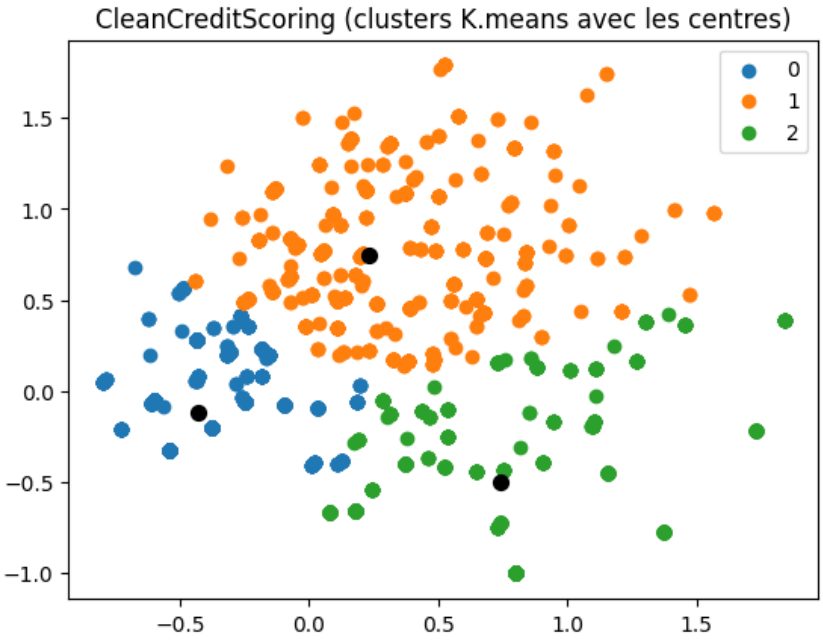
3.Tracer des grappes

```
#ploter tout les cluster
#Getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(df[label == i , 0] , df[label == i , 1] , label = i)
    plt.title("CleanCreditScoring (clusters K.means)")
plt.legend()
plt.show()
```




```
#Getting the Centroids
centroids = kmeans.cluster_centers_
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
  plt.scatter(df[label == i , 0] , df[label == i , 1] , label = i)
plt.scatter(centroids[:,0] , centroids[:,1] , s = 50, color = 'k')
```

```
plt.title("CleanCreditScoring (clusters K.means avec les centres)")
plt.legend()
plt.show()

print(u_labels)

#Printing the coordinates of cluster centers
print(kmeans.cluster_centers_)
```




```
[0 1 2]
[[-0.42897251 -0.1155666 ]
 [ 0.23139271  0.74236365]
 [ 0.74373139 -0.49880836]]
```

```
kmeans.predict(df)
```

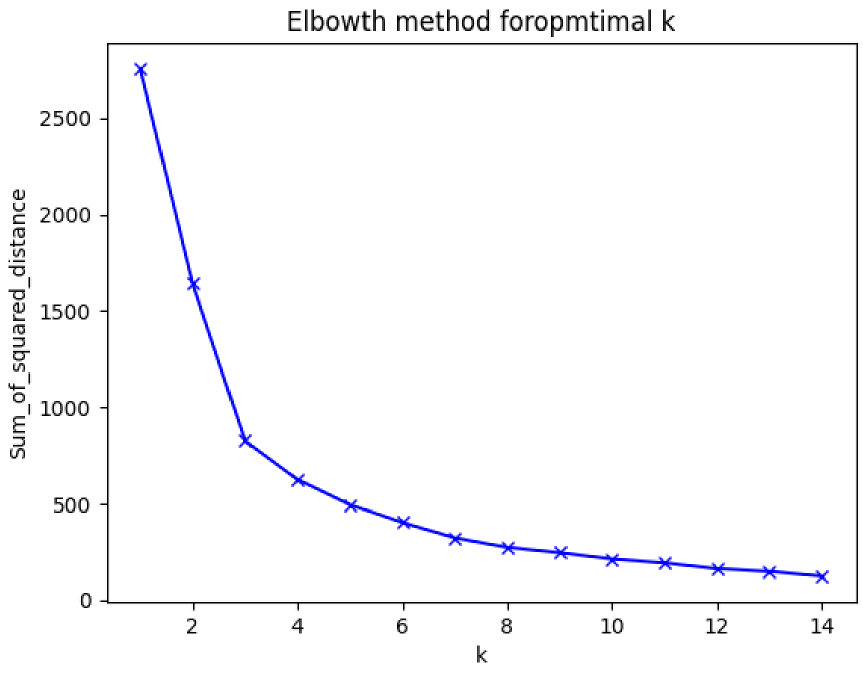
```
array([0, 0, 1, ..., 1, 2, 0], dtype=int32)
```

4.K optimal

'The Elbow Method'

```
Sum_of_sqyared_distance=[]
K=range(1,15)
for k in K:
  km=KMeans(n_clusters=k, random_state=0)
  km=km.fit(df)
  Sum_of_sqyared_distance.append(km.inertia_)

plt.plot(K,Sum_of_sqyared_distance,'bx-')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distance')
plt.title('Elbowth method foropmtimal k')
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarnin
  warnings.warn(
Text(0.5, 1.0, 'Elbowth method foropmtimal k')
```



✓ 5 s terminée à 01:31

Importation, analyse initiale de Dataset

1.Importation de libreries

```
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from tensorflow import keras
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
```

2.Chargement des données et analyse

```
# Lecture du fichier texte
with open('mca.deep.txt', 'r') as fichier:
    lignes = fichier.readlines()

# Affichage des premières lignes
nb_lignes_a_afficher = 10 # Nombre de lignes à afficher
for i in range(nb_lignes_a_afficher):
  print(lignes[i])
```

```
     "Dim 1","Dim 2","Dim 3","Dim 4","Dim 5","df.Status"

     -0.359168816422818,-0.364698786760379,0.224477821344743,-0.272337146300252,0.0227747254682127,"good"

     -0.455042759949676,2.04001514555353,1.73022951976883,0.883150228205062,0.195515406866965,"good"

     -0.699728913815721,-0.367044208939554,0.28540327765881,-1.00859608627284,0.245814449464481,"bad"

     0.604621518631184,-0.157214447593078,0.227405649274657,0.449564308719192,-0.258885583426606,"good"

     0.604621518631518,-0.157214447592592,0.227405649274291,0.449564308718727,-0.258885583426482,"good"

     -0.395232632824693,0.107246887869652,-0.395960235770232,0.0701850152078038,-0.165653283139742,"good"

     -0.395232632824764,0.107246887869692,-0.395960235770231,0.0701850152078121,-0.165653283139729,"good"

     1.29361692427923,0.230601070377521,-0.191048781773817,-0.0719920152958275,-0.27918786076106,"good"

     -0.568140776261555,0.0499889121728875,-0.303392295909036,-0.762094911748248,0.0782117441900117,"good"
```

```
import pandas as pd

# Lecture du fichier texte
donnees = pd.read_csv('mca.deep.txt', delimiter=',')  # Remplacez "nom_du_fichier.txt" par le nom réel de votre fichier

# Écriture des données dans un fichier CSV
donnees.to_csv('data.csv', index=False)  # Remplacez "nom_du_fichier.csv" par le nom souhaité pour votre fichier CSV
```

```
import pandas as pd
import numpy as np

df=pd.read_csv("data.csv", sep=',')
df.head()
```

	Dim 1	Dim 2	Dim 3	Dim 4	Dim 5	df.Status
0	-0.359169	-0.364699	0.224478	-0.272337	0.022775	good
1	-0.455043	2.040015	1.730230	0.883150	0.195515	good
2	-0.699729	-0.367044	0.285403	-1.008596	0.245814	bad
3	0.604622	-0.157214	0.227406	0.449564	-0.258886	good

```
df.shape
```

```
(4446, 6)
```

```
df.isnull().sum()
```

```
Dim 1        0
Dim 2        0
Dim 3        0
Dim 4        0
Dim 5        0
df.Status    0
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4446 entries, 0 to 4445
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Dim 1      4446 non-null   float64
 1   Dim 2      4446 non-null   float64
 2   Dim 3      4446 non-null   float64
 3   Dim 4      4446 non-null   float64
 4   Dim 5      4446 non-null   float64
 5   df.Status  4446 non-null   object 
dtypes: float64(5), object(1)
memory usage: 208.5+ KB
```

3.Encoding

```
#liste des colones object pour préparer l'encoding
list_obj = []
for i in range(0,len(df.dtypes)):
  if df.dtypes[i] == object:
    list_obj.append(df.columns[i])
print("nombre de colonne object:",len(list_obj))
list_obj

#boucle pour l'encoding
for val in list_obj:
  encoder=LabelEncoder()
  df[val]=encoder.fit_transform(df[val])
print('data after coverting data to numeric')
df.shape
```

```
nombre de colonne object: 1
data after coverting data to numeric
(4446, 6)
```

```
df.head()
```

Réseau d'apprentissage en profondeur-prédiction ("Status" soit god ou bad)

1.Préparation des Input

```
3  0.604622  -0.157214  0.227406  0.449564  -0.258886          1
```

```
#transfor our data to numpy
np_bank = df.to_numpy()
np_bank
```

```
array([[-0.35916882, -0.36469879,  0.22447782, -0.27233715,  0.02277473,
         1.        ],
       [-0.45504276,  2.04001515,  1.73022952,  0.88315023,  0.19551541,
         1.        ],
       [-0.69972891, -0.36704421,  0.28540328, -1.00859609,  0.24581445,
         0.        ],
       ...,
       [ 0.09958959,  0.11060812, -0.3725695 , -0.08603223,  0.46902814,
         0.        ],
       [ 0.43171338, -0.21447242,  0.31997359, -0.38271562, -0.01502056,
         1.        ],
       [-0.56814078,  0.04998891, -0.3033923 , -0.76209491,  0.07821174,
         1.        ]])
```

```
#Separate feature and target variables
X_data = np_bank[:,0:5]
Y_data=np_bank[:,5]

print('feature avant standarisation et arguet avant one-hot-encoding')
X_data[:5,:], Y_data[:5]
```

```
feature avant standarisation et arguet avant one-hot-encoding
(array([[-0.35916882, -0.36469879,  0.22447782, -0.27233715,  0.02277473],
        [-0.45504276,  2.04001515,  1.73022952,  0.88315023,  0.19551541],
        [-0.69972891, -0.36704421,  0.28540328, -1.00859609,  0.24581445],
        [ 0.60462152, -0.15721445,  0.22740565,  0.44956431, -0.25888558],
        [ 0.60462152, -0.15721445,  0.22740565,  0.44956431, -0.25888558]]),
 array([1., 1., 0., 1., 1.]))
```

```
#Create a scaler model that is fit on the input data.
scaler = StandardScaler().fit(X_data)
#Scale the numeric feature variables
X_data = scaler.transform(X_data)
#Convert target variable as a one-hot-encoding array
Y_data = tf.keras.utils.to_categorical(Y_data,2)

print('feature aprés standarisation et targuet aprés one-hot-encoding')
X_data[:5,:], Y_data[:5,:]
```

```
feature aprés standarisation et targuet aprés one-hot-encoding
(array([[-0.5660578 , -0.66859043,  0.42532027, -0.53029778,  0.04469113],
        [-0.7171572 ,  3.73989345,  3.27828236,  1.7196795 ,  0.38366235],
        [-1.10278786, -0.67289022,  0.54075631, -1.963949  ,  0.48236479],
        [ 0.95289655, -0.28821614,  0.43086765,  0.87539639, -0.50801444],
        [ 0.95289655, -0.28821614,  0.43086765,  0.87539639, -0.50801444]]),
 array([[0., 1.],
        [0., 1.],
        [1., 0.],
        [0., 1.],
        [0., 1.]], dtype=float32))
```

```
#Split training and test data
X_train,X_test,Y_train,Y_test = train_test_split( X_data, Y_data, test_size=0.10)

print('Train Test Dimensions')
print(X_train.shape, Y_train.shape, X_test.shape, Y_test.shape)
```

```
Train Test Dimensions
(4001, 5) (4001, 2) (445, 5) (445, 2)
```

2.Construction du modèle

```
model = Sequential([
  Dense(64, activation='relu', input_shape=(5,)),
  Dense(128, activation='relu'),
  Dense(2, activation='softmax')])
```

3.Compilation du modèle

```
model.compile(
  optimizer='adam',
  loss='categorical_crossentropy',
  metrics=['accuracy'])
```

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_6 (Dense)             (None, 64)                384

 dense_7 (Dense)             (None, 128)               8320

 dense_8 (Dense)             (None, 2)                 258

=================================================================
Total params: 8,962
Trainable params: 8,962
Non-trainable params: 0
_________________________________________________________________
```

4.Entrainement du modèle

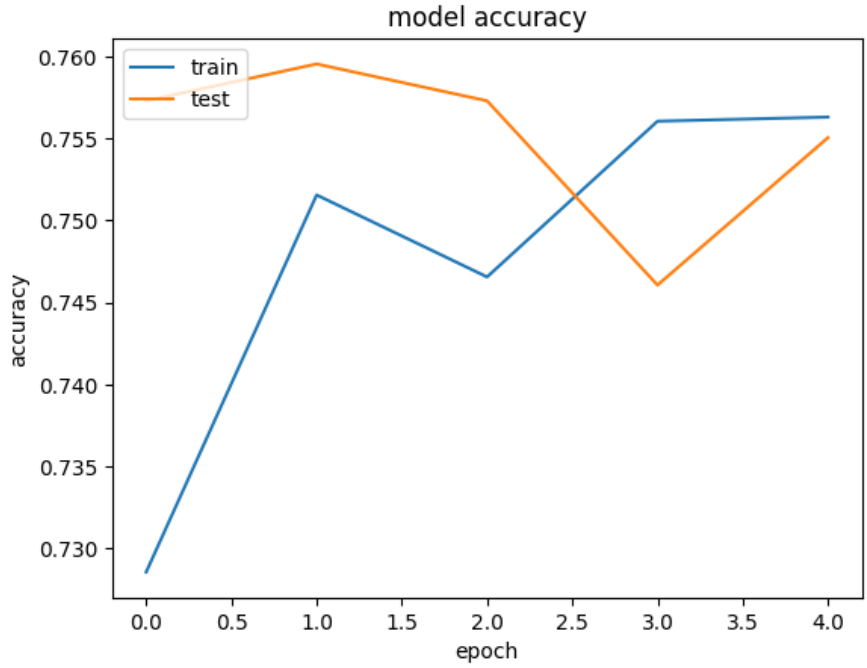
```
history = model.fit(
  X_train,
  Y_train,
  epochs=5,
  batch_size=32,
  validation_data=(X_test, Y_test))
```

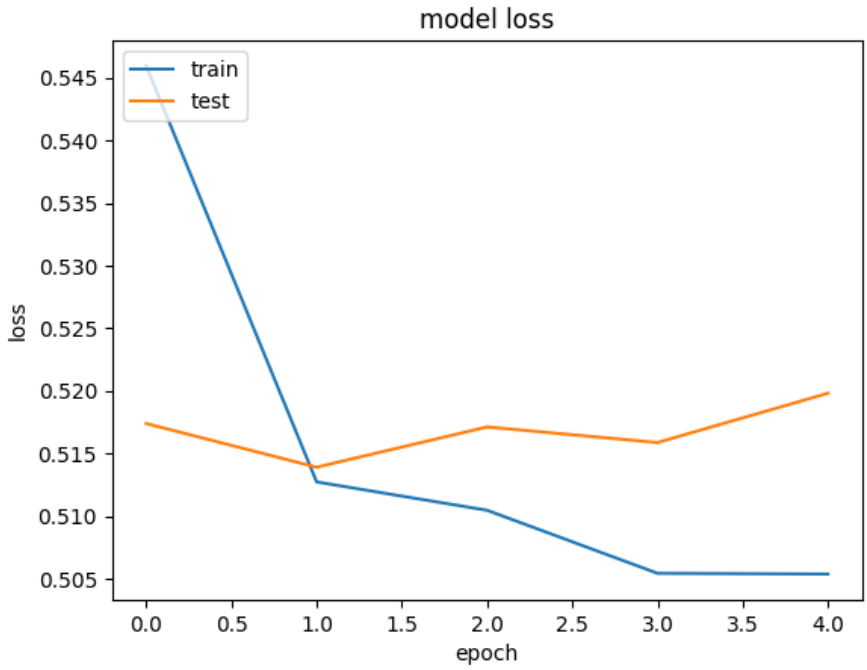
```
Epoch 1/5
126/126 [==============================] - 1s 4ms/step - loss: 0.5459 - accuracy: 0.7286 - val_loss: 0.5174 - val_
Epoch 2/5
126/126 [==============================] - 0s 2ms/step - loss: 0.5127 - accuracy: 0.7516 - val_loss: 0.5139 - val_
Epoch 3/5
126/126 [==============================] - 0s 2ms/step - loss: 0.5105 - accuracy: 0.7466 - val_loss: 0.5171 - val_
Epoch 4/5
126/126 [==============================] - 0s 2ms/step - loss: 0.5054 - accuracy: 0.7561 - val_loss: 0.5159 - val_
Epoch 5/5
126/126 [==============================] - 0s 2ms/step - loss: 0.5054 - accuracy: 0.7563 - val_loss: 0.5198 - val_
```

5.Graphe de Accuracy et loss

```
# summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
```

```
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```





6.Evaluation du modèle

```
model.evaluate(X_test, Y_test)
```

```
14/14 [==============================] - 0s 2ms/step - loss: 0.5198 - accuracy: 0.7551
[0.5198062062263489, 0.7550562024116516]
```

7.Enregistrement du modèle

```
model.save("bank_model_deep_learning")

#Loading a Model
loaded_model = keras.models.load_model("bank_model_deep_learning")

#Print Model Summary
#loaded_model.summary()
```

```
WARNING:absl:Found untraced functions such as _update_step_xla while saving (showing 1 of 1). These functions will
```

8.Prédection d'une donnée

```
#Raw prediction data
prediction_input = [[0.9856072 , 1.1750383 ,-1.9758729, -1.7555479 ,-5.0619728 ]]

#Scale prediction data with the same scaling model
scaled_input = scaler.transform(prediction_input)

#Get raw prediction probabilities
raw_prediction = model.predict(scaled_input)
print("Raw Prediction Output (Probabilities) :" , raw_prediction)

#Find prediction
prediction = np.argmax(raw_prediction)
print("Prediction is ", encoder.inverse_transform([prediction]))
```

```
1/1 [==============================] - 0s 136ms/step
Raw Prediction Output (Probabilities) : [[1.33274245e-08 1.00000000e+00]]
Prediction is  ['good']
```

✓ 0 s terminée à 01:09

Résumé

Ce mémoire explore l'utilisation de l'Analyse en Correspondance Multiple (ACM) pour explorer des données qualitatives. Le premier chapitre présente la méthode de l'ACM, ses concepts clés et les étapes d'analyse. Le deuxième chapitre se concentre sur l'objet ACM de FactoMineR et ses fonctionnalités. Enfin, le troisième chapitre applique l'ACM à des données réelles, décrivant le contexte, la préparation des données et les résultats obtenus. Le mémoire découvert sur l'importance de l'ACM pour explorer les structures de données qualitatives et offrir des perspectives pour de futures recherches.

Mots clés : MCA, FactomineR, données catégorielles, Réduction de dimension, codage binaire, Visualisation des données, Décomposition en valeurs singulières

Summary

This thesis explores the use of Multiple Correspondence Analysis (MCA) to explore qualitative data. The first chapter presents the ACM method, its key concepts and the analysis steps. The second chapter focuses on the ACM object of FactoMineR and its functionalities. Finally, the third chapter applies MCA to real data, describing the context, the preparation of the data and the results obtained. The thesis revealed the importance of AMC for exploring qualitative data structures and offering insights for future research.

Key words: MCA, FactomineR, categorical data, Dimension reduction, binary coding, Data visualization, Singular value decomposition

ملخص

تستكشف هذه الرسالة استخدام تحليل المراسلات المتعددة (MCA) لاستكشاف البيانات النوعية. يقدم الفصل الأول طريقة ACM ومفاهيمها الأساسية وخطوات التحليل. يركز الفصل الثاني على موضوع ACM لـ FactoMineR ووظائفها. أخيرًا ، يطبق الفصل الثالث حساب تحدي الألفية على البيانات الحقيقية ، ويصف السياق ، وإعداد البيانات والنتائج التي تم الحصول عليها. كشفت الأطروحة أهمية AMC لاستكشاف هياكل البيانات النوعية وتقديم رؤى للبحوث المستقبلية.

الكلمات المفتاحية: البيانات الفئوية ، تقليل الأبعاد ، الترميز الثنائي ، تصور البيانات ، تحليل القيمة المفردة