UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Projet Analyse de données
Master 1 Econométrie-Statistiques
Université Paris 1 Panthéon-Sorbonne

Oussama Laaumari, Abir Khan

Professeur : Joseph Rynkiewicz

Exercice 1 : Régression binaire

1. *Déterminer le meilleur modèle possible pour modéliser $P(Y|X_1, X_2)$.*

Tout d'abord on va transformer les valeurs de Y : on remplace les 2 par des 1 et les 1 par des 0.

Vu que la variable à expliquer est binaire, on utilise un modèle LOGIT.
On obtient le modèle suivant grâce à la commande *glm* :

$$Y = - 0{,}3821 + 0.0949X_1 - 0.1368X_2$$

Analysons les résultats en détail.

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5963  -1.0505  -0.8032   1.1978   1.9077

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.38210    0.04697  -8.135 4.13e-16 ***
X1           0.09490    0.01582   5.998 2.00e-09 ***
X2          -0.13678    0.01640  -8.339  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2713.1  on 1999  degrees of freedom
Residual deviance: 2602.9  on 1997  degrees of freedom
AIC: 2608.9

Number of Fisher Scoring iterations: 4
```

L'intervalle de confiance au seuil α=5% pour la constante et les coefficients est :

```
                  2.5 %     97.5 %
(Intercept) -0.4741655 -0.2900388
X1           0.0638862  0.1259045
X2          -0.1689293 -0.1046307
```

Odds ratios, test de Wald

Avec la commande predict.glm on fait une prédiction à partir de la régression logistique puis avec la commande table() on crée un tableau de contingence à partir de la prédiction. On fait ensuite une matrice de confusion.

```
> confusion <- table(simu$Y,
+                    ifelse(prediction <  0.5,
+                           "prédit  \"non\"",
+                           "prédit  \"oui\""))
> kable(confusion)
```

	prédit "non"	prédit "oui"
0	1031	141
1	520	308

On calcule ensuite l'accuracy de ce modèle.

```
> sum(diag(confusion))/sum(confusion)*100
[1] 66.95
```

Il y a donc 66.95% de bonnes prédictions avec le modèle logit ce qui est relativement bon mais nous allons en tester deux autres.

Modèle Cauchit

```
> summary(cauchit)

Call:
glm(formula = Y ~ X1 + X2, family = binomial(link = "cauchit"),
    data = simu)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5766  -1.0209  -0.7968   1.1863   1.8236

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.36980    0.04330  -8.541  < 2e-16 ***
X1           0.06863    0.01429   4.802 1.57e-06 ***
X2          -0.13410    0.01605  -8.353  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2713.1  on 1999  degrees of freedom
Residual deviance: 2598.7  on 1997  degrees of freedom
AIC: 2604.7

Number of Fisher Scoring iterations: 8
```

De même que pour le modèle précédent on fait une matrice de confusion et on calcule l'accuracy du modèle et on trouve qu'il fait 70% de bonnes prédictions.

```
> kable(confusion)
```

	prédit "non"	prédit "oui"
0	1075	97
1	503	325

```
> sum(diag(confusion))/sum(confusion)*100
[1] 70
```

K Nearest Neighbors (*k-NN)*

k-NN est un algorithme standard de classification qui repose exclusivement sur le choix de la métrique de classification. Il est “non paramétrique” (seul k doit être fixé) et se base uniquement sur les données d’entraînement.

L’idée est la suivante : à partir d’une base de données étiquetées, on peut estimer la classe d’une nouvelle donnée en regardant quelle est la classe majoritaire des k données voisines les plus proches (d’où le nom de l’algorithme). Le seul paramètre à fixer est k, le nombre de voisins à considérer.

Voici l’accuracy du modèle qui est de 83.5% ce qui est très bien.

```
> pr <- knn(simu_train,simu_test,cl=simu_target_category,k=13)
> tab <- table(pr,simu_test_category)
> tab
   simu_test_category
pr    0   1
  0 109  13
  1  20  58
> accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
> accuracy(tab)
[1] 83.5
```

2. Utiliser le fichier “xsimutest” pour prédire 1000 observations de la variable Y du fichier de test. Il faut sauvegarder ces prédictions dans un fichier “.txt”, les séparateurs devront être des retours chariot. Par exemple, si les 1000 prédictions du fichier de test de Y sont dans le vecteur “predictions”, on le sauvera avec la procédure R :

   write.table(predictions,"predictions.txt",row.names=F,col.names=F)

   Il faudra joindre ce fichier `a votre devoir.

On mène la prédiction de la manière suivante et on trouve 42.2% de 1 et dans le fichier initial simu on trouvait 41.4% de 1. Ce modèle est donc très précis

```
> prxsimu <- knn(simu_train,xsimutest,cl=simu_target_category,k=13)
> summary(prxsimu)
  0   1
578 422
```

Exercice 2 : Analyse en composantes principales

1. *Quel est le pourcentage d'inertie expliquée par les trois premiers facteurs? Par le premier plan factoriel?*

Le pourcentage d'inertie nous est donné dans la colonne *variance.percent*.

```
> get_eigenvalue(res.pca)
       eigenvalue variance.percent cumulative.variance.percent
Dim.1 5.549271024      69.36588781                    69.36589
Dim.2 1.550202725      19.37753406                    88.74342
Dim.3 0.480655817       6.00819771                    94.75162
Dim.4 0.280682369       3.50852962                    98.26015
Dim.5 0.084751837       1.05939796                    99.31955
Dim.6 0.034770185       0.43462731                    99.75417
Dim.7 0.013450458       0.16813073                    99.92231
Dim.8 0.006215585       0.07769482                   100.00000
```

Pour les trois premiers facteurs nous avons : 69.36588781%
19.37753406%
6.00819771%

Ce qui nous fait 94.75162% d'inertie expliquée par les trois premiers facteurs.

Quant au premier plan factoriel qui est constitué des deux premiers axes (dim 1 et 2), on remarque qu'il explique 88.74342% de l'inertie.

2. *Interpréter les 2 axes principaux à partir des corrélations des variables avec ces axes.*

On obtient les corrélations des variables aux axes avec la commande suivante :

```
> res.pca$var$cor
              Dim.1      Dim.2       Dim.3
CYL       0.9660339 -0.1258208 -0.08401801
PUIS      0.9588164 -0.2126491  0.07511311
LON       0.2183102  0.8656561  0.44334315
LAR       0.8324492  0.4127766 -0.15906970
POIDS     0.6803680  0.6272103 -0.35138791
VITESSE   0.8993843 -0.2551380  0.33720340
ACCEL    -0.8819812  0.2979866 -0.05823142
CO2       0.9520536 -0.1487381 -0.07435053
```

Pour interpréter les 2 axes principaux, on se sert des corrélations des variables à ces axes.

On remarque que toutes les variables à part ACCEL sont corrélées positivement à l'axe 1. Cela s'explique par le fait que l'accélération est mesurée en seconde et une valeur basse signifie qu'une voiture a une grande accélération.

Seulement la variable LON est faiblement corrélée ce qui se remarque sur le graphique de l'ACP par sa forme quasi-perpendiculaire à l'axe 1. Il y a presque un effet taille car les variables sont presque toutes du même côté de l'axe. (i.e. elles contribuent toutes dans le même sens à la formation de l'axe)

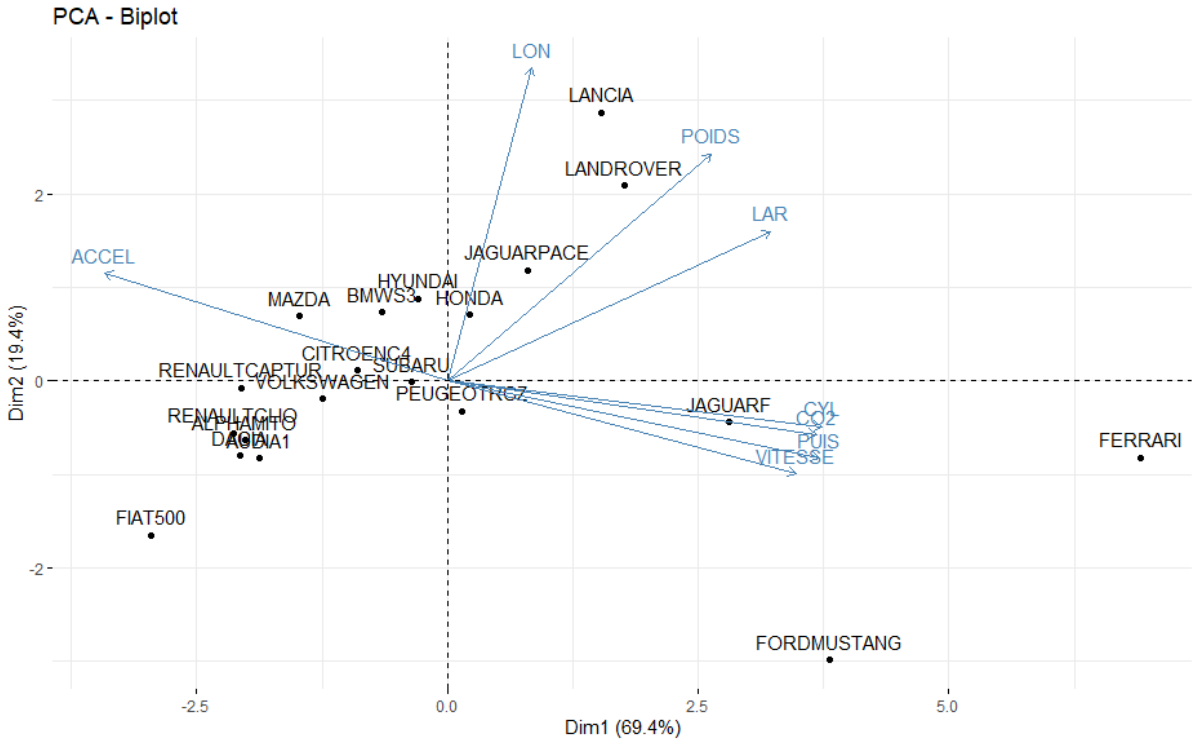
Seulement les variables LON et POIDS sont fortement corrélées positivement à l'axe 2, LAR est moyennement corrélé positivement à l'axe 2. ACCEL est trop faiblement corrélée à l'axe. Les autres variables sont corrélées négativement, mais trop faiblement donc elles ne servent pas à l'interprétation de l'axe 2.

Donc si les individus se trouvent à droite en se basant sur l'axe 1, cela veut dire que les individus ont une faible valeur d'ACCEL et donc une grande accélération, une grande valeur de CYL, une grande puissance, une grande vitesse, émettent beaucoup de CO2 et sont larges.
Inversement, si on se trouve à gauche dans les valeurs négatives, on inverse le sens de cette interprétation.

Quant à l'axe 2, si les individus se trouvent en haut du graphique cela veut dire que ce sont des marques de voiture longues, avec un poids élevé et larges si on considère que cette variable est assez corrélée (0.41) à l'axe 2. Inversement, si on se trouve dans les valeurs négatives, on inverse le sens de cette interprétation. Il y a un effet taille pour la dimension 2; en effet les variables sont toutes du même côté de l'axe. (i.e. elles contribuent toutes dans le même sens à la formation de l'axe)

3. *Représentez les individus sur le premier plan factoriel et répondez aux questions suivantes :*

La représentation des individus et des variables sur un biplot se fait avec le code suivant : fviz_pca_biplot(res.pca).



a) *Les individus sont-ils bien représentés sur le premier plan factoriel?*

Pour savoir si un individu est bien représenté, sa valeur cos2 doit être proche de 1.
Cela veut dire que l'individu doit être proche d'un axe et donc corrélé à celle-ci et donc les individus proches du centre ne sont pas interprétables.
On remarque que PEUGEOT RCZ et SUBARU ne sont pas bien représentés sur le premier plan factoriel à l'aide du biplot, mais on peut le voir plus précisément ci-dessous.

```
> res.pca$ind$cos2
                    Dim.1        Dim.2        Dim.3
ALPHAMITO     0.88418157 0.0882481546 1.059039e-02
AUDIA1        0.82447953 0.1603240825 5.639548e-03
CITROENC4     0.63441116 0.0097161853 1.215544e-01
JAGUARF       0.87295591 0.0211260379 4.142085e-02
PEUGEOTRCZ    0.02354612 0.1090728501 1.335421e-01
LANDROVER     0.31747599 0.4453076078 1.591429e-01
RENAULTCLIO   0.91588474 0.0619189342 5.517074e-03
BMWS3         0.34739866 0.4385043460 4.263289e-02
DACIA         0.83548059 0.1227757130 1.250491e-02
HYUNDAI       0.07691485 0.7124226564 2.027533e-01
LANCIA        0.20567125 0.7152890894 3.655778e-02
RENAULTCAPTUR 0.91218646 0.0011120940 6.056581e-03
FORDMUSTANG   0.52868209 0.3233183930 1.437252e-01
FIAT500       0.70301792 0.2181054338 9.588572e-03
HONDA         0.05681670 0.5506712131 1.734503e-01
FERRARI       0.92712779 0.0130358167 4.129523e-02
SUBARU        0.26365614 0.0002406284 1.728306e-01
MAZDA         0.57219677 0.1274534969 3.104131e-03
VOLKSWAGEN    0.85640405 0.0198445442 2.868204e-02
JAGUARPACE    0.21914850 0.4760572094 7.096227e-06
```

En effet, on voit que le cos2 de l'individu PEUGEOT RCZ est d'environ 0.13 et pour SUBARU de 0.26 en prenant la somme des cos2 de la colonne dim 1 et dim 2. Les autres individus ont un cos2 supérieur à 0.6 donc ils sont bien représentés.

b) *Quelles sont les caractéristiques des individus en haut du graphe?*

Ce sont les marques de voitures longues (LON), lourdes (POIDS) et larges (LAR).

c) *Quelles sont les caractéristiques des individus à droite du graphe?*

Ce sont les voitures avec une grande accélération, car si la valeur numérique en seconde est petite, cela veut dire que la voiture a une grande accélération, une grande valeur de cyl, beaucoup de puissance, une grande vitesse, qui diffuse beaucoup de CO2 et large.

d) *Quelles sont les caractéristiques des individus en bas à gauche du graphe?*

Les individus en bas à gauche du graphe sont les marques de voitures peu longues, peu lourdes, peu larges, une faible accélération, qui émet peu de C02, une faible puissance et une faible vitesse. La méthode des kmeans confirme cette analyse. On remarque que les marques de voitures en bas à gauche du graphe correspondent au cluster 4 et ont bien toutes les caractéristiques énoncées précédemment.

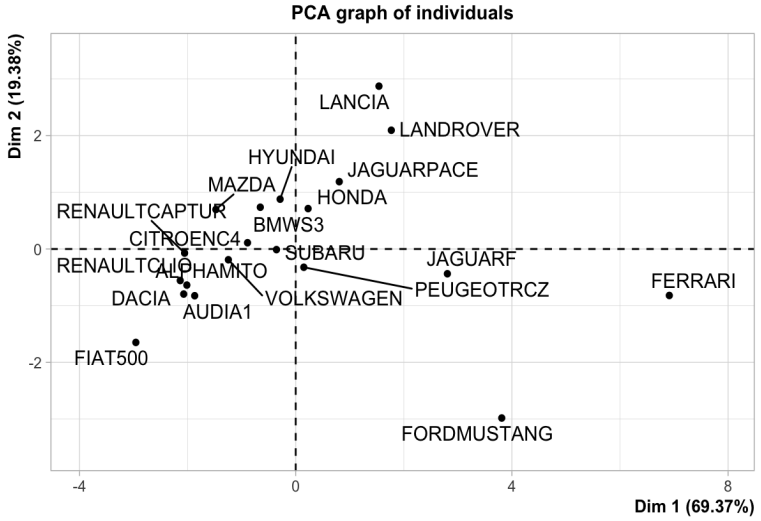
e) *Peut-on dire que les individus PEUGEOT RCZ et JAGUARF ont un profil semblable? Si oui quel est-il?*

On ne peut pas conclure à une similarité entre ces 2 individus, car PEUGEOT RCZ est mal représenté car proche du centre

f) *Peut-on dire que les individus LANCIA et LANDROVER ont un profil semblable? Si oui quel est-il?*

Oui, car elles sont proches sur le premier plan factoriel. Ce sont de longues voitures, avec un poids élevé et une grande largeur.

g) *Interpréter la représentation graphique des individus.*



Tout d'abord, on peut voir que la majorité des marques de voitures se concentre principalement au centre. Il y a en revanche quelques marques très éloignées de ce groupe (en général, les mieux représentées).
Comme indiqué dans la question 1, l'axe 1 oppose les voitures possédant une grosse cylindrée, puissants, rapides, larges, nécessitant peu de temps pour accélérer mais très polluants, aux voitures avec une faible cylindrée, peu puissants, peu rapides, pas très larges, nécessitant beaucoup de temps pour accélérer mais peu polluants. L'axe 2 quant à elle oppose les voitures lourdes, longues aux voitures légères et pas très longues.

Nous allons prendre quelques exemples de marques pour interpréter le graphique, avec l'aide des données.

Si nous prenons l'exemple de la FERRARI et de la FIAT 500 :
D'un côté, nous avons la FERRARI qui est la voiture avec la plus grosse cylindrée, la plus puissante, la plus rapide, une des plus larges, nécessitant le moins de secondes pour accélérer mais c'est la plus polluante. Il est difficile d'interpréter la FERRARI par rapport à l'axe 2 (cos2 = 0,01). Cette variable est très bien placée par rapport à l'axe 1.

D’un autre côté, nous avons la FIAT 500, qui est tout le contraire de la FERRARI : c’est la voiture la moins rapide, la moins puissante, la plus petite, possédant une cylindrée 6 fois plus petite, qui nécessite beaucoup de secondes pour accélérer, mais polluant peu. C’est la voiture la moins lourde et la moins longue.

Prenons enfin l’exemple de la LANCIA. C’est une voiture possédant une cylindrée moyenne, de puissance moyenne, pas mal rapide, la plus large, nécessitant pas mal de secondes pour accélérer et polluant pas mal. C’est la voiture la plus lourde mais aussi la plus longue. Cette variable est très bien placée par rapport à l’axe 2.

Exercice 3 : Classification

1. *Avec les données sur les voitures, réalisez une (ou des) classifications avec la méthode des kmean et interprétez les résultats obtenus.*

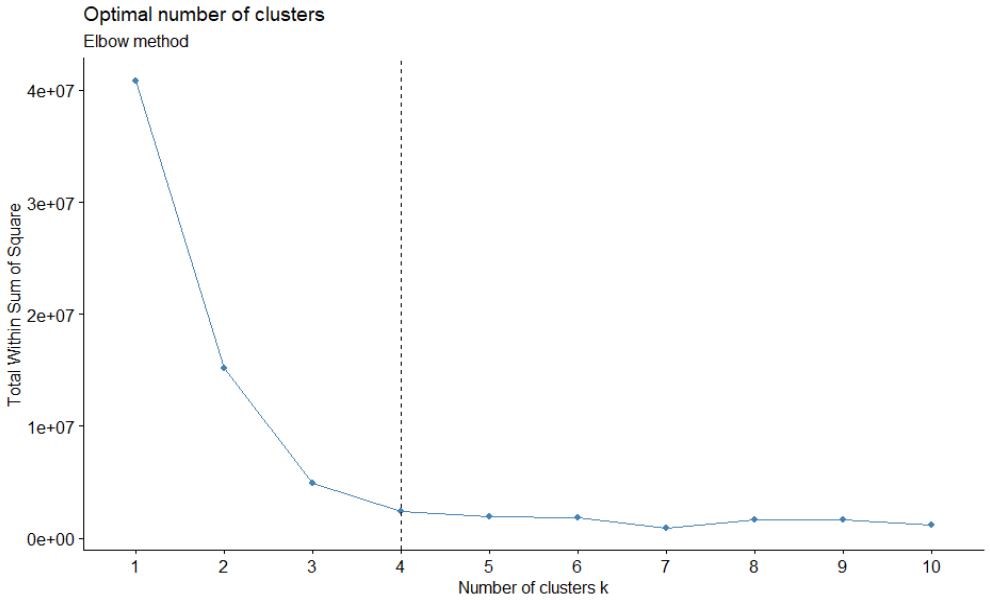
Tout d’abord, on cherche à savoir le nombre de clusters optimaux pour effectuer la méthode des kmeans.

Pour cela, nous avons 3 méthodes :

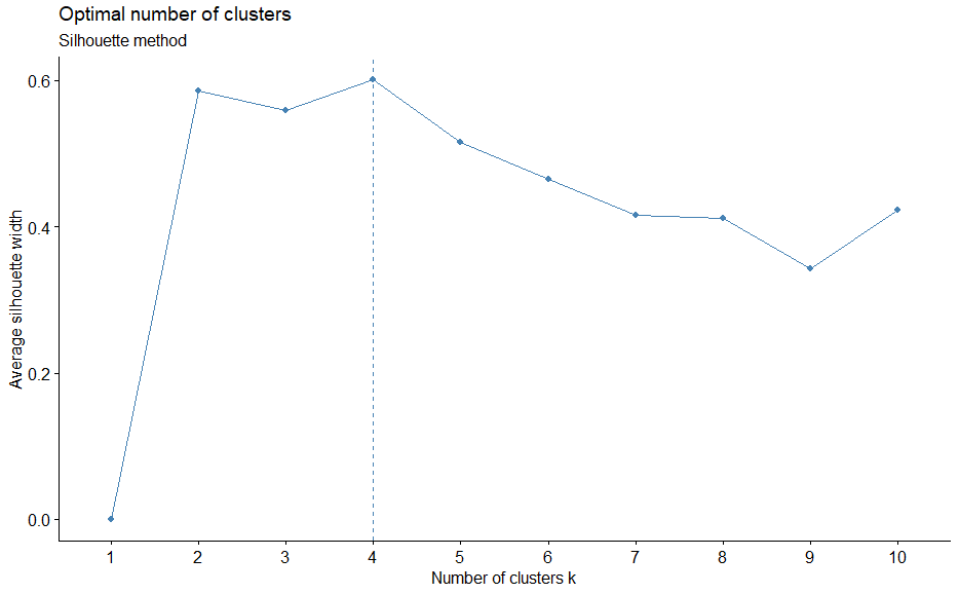
- Elbow method : basée sur la minimisation de la somme des carrés des écarts à l’intérieur des clusters (SSwithin).

- Average silhouette method : basée sur la maximisation du paramètre appelé “average silhouette”.

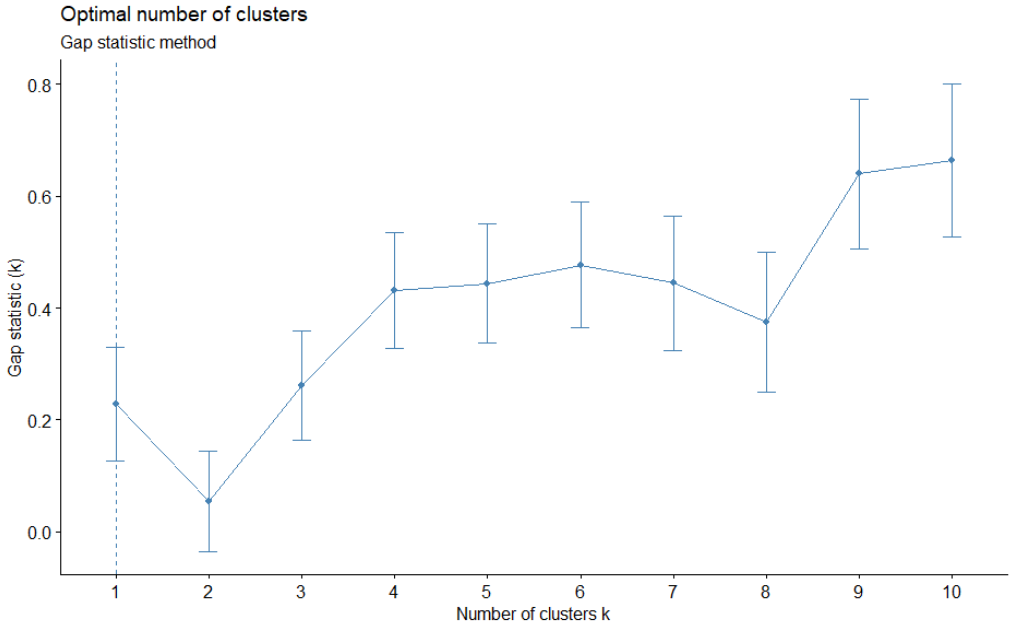
- Gap statistic method : basée sur la comparaison de la variation totale intra-cluster pour différentes valeurs de k avec leurs valeurs attendues sous une distribution de référence nulle des données



Elbow method : l'emplacement d'un coude est généralement considéré comme un indicateur approprié du nombre de clusters. Cette méthode nous donne 4 clusters.



Silhouette method : ici, l'emplacement du maximum est considéré comme le nombre approprié de clusters. Cette méthode nous donne aussi 4 clusters.



La dernière méthode ne nous donne qu'un cluster, car elle cherche la plus petite valeur de k selon ses critères.

Ces trois méthodes ne conduisent pas forcément au même résultat. Ici, pour deux d'entre elles, le nombre optimal de clusters est 4. On effectue donc la méthode des kmeans avec 4 clusters.

Voici les résultats :

```
> voitures.km4 <- kmeans(voitures, centers = 4, nstart = 20)
> voitures.km4
K-means clustering with 4 clusters of sizes 2, 3, 8, 7

Cluster means:
       CYL      PUIS      LON      LAR    POIDS  VITESSE     ACCEL      CO2
1 5606.500 540.50000 381.5000 193.5000 1800.000 292.5000  4.450000 339.5000
2 2921.333 257.66667 484.0000 194.3333 2157.333 211.0000  8.833333 214.6667
3 1917.625 138.62500 451.2500 182.5000 1531.000 199.0000 10.250000 129.1250
4 1001.286  95.57143 403.4286 173.5714 1087.714 179.1429 11.657143 109.2857

Clustering vector:
    ALPHAMITO        AUDIA1     CITROENC4       JAGUARF    PEUGEOTRCZ     LANDROVER   RENAULTCLIO         BMWS3
            4             4             4             2             3             2             4             3
       DACIA       HYUNDAI        LANCIA RENAULTCAPTUR   FORDMUSTANG       FIAT500         HONDA       FERRARI
            4             3             2             4             1             4             3             1
      SUBARU         MAZDA    VOLKSWAGEN    JAGUARPACE
            3             3             3             3

Within cluster sum of squares by cluster:
[1] 931599.2 572540.5 603163.6 249369.3
 (between_SS / total_SS =  94.2 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

On remarque que le cluster 4 regroupe les voitures avec une valeur élevée de CYL, très puissante, très rapide (vitesse), qui émet beaucoup de CO2, une grande accélération. Ce

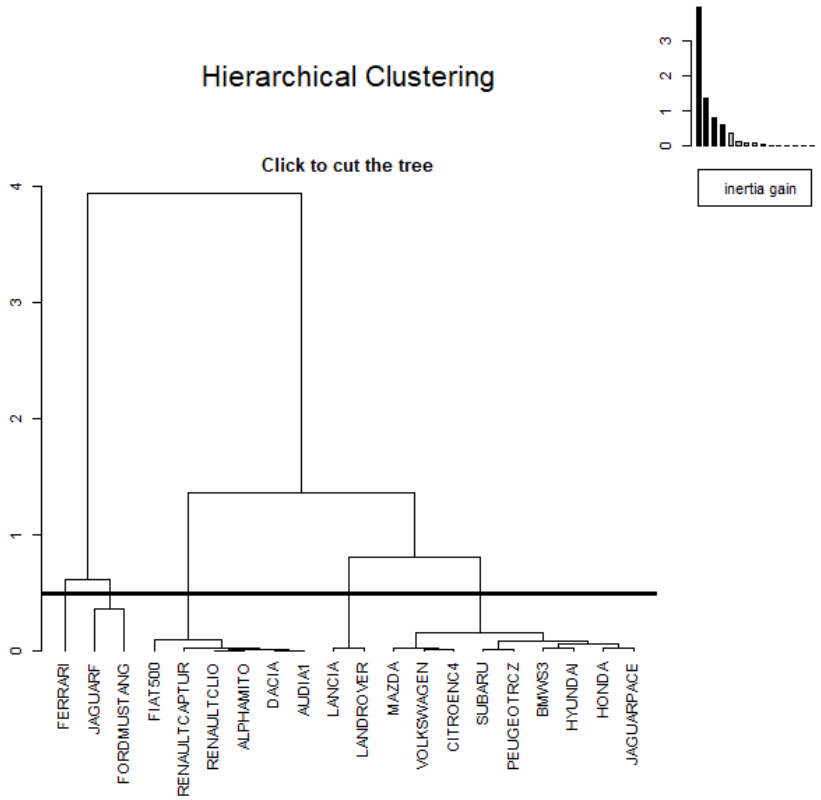
sont aussi des marques qui produisent des voitures peu longues, car en général, les voitures de sport ont seulement 2 places.
Cela semble logique quand on regarde le graphique de l'acp et en sachant les caractéristiques des marques FERRARI et FORDMUSTANG qui sont connus pour être des voitures de sport ou des muscle cars respectivement.

Le cluster 2 regroupe des marques avec une valeur de CYL assez élevée par rapport aux marques des clusters 1 et 3, une puissance aussi élevée par rapport à ces derniers, un poids très élevé, une vitesse aussi élevée par rapport aux clusters 1 et 3, ainsi qu'une accélération élevée et émet beaucoup de CO2. Ce sont des marques avec des voitures plutôt longues et larges par rapport aux autres.
LANDROVER est une marque principalement de SUV, LANCIA produit des camions et des voitures de type berline (longues, larges, puissante, rapide et donc polluantes), JAGUAR est une marque de voitures de sport, berline et suv qui a toutes les caractéristiques citées précédemment.

Le cluster 1 représente les marques millieu/haute gamme sans compter les marques de luxe comme FERRARI.
Ce sont des marques de voitures moyennement puissantes, de longueur et largeur moyenne, ayant une vitesse et une accélération moyenne et donc une émission de CO2 pas très élevée.

Le cluster 3 représente les voitures bas/milieu de gamme comme RENAULT CLIO ou FIAT 500. Ce sont les marques de voitures avec le moins de CYL, de puissance, avec une longueur et largeur plutôt petite, une vitesse très petite et une accélération très lente et donc une faible émission de CO2.

2. *Faire une classification hiérarchique avec la méthode de “Ward”. Interpréter le dendrogramme. En combien de classes aurait-on envie de couper ce graphique?*

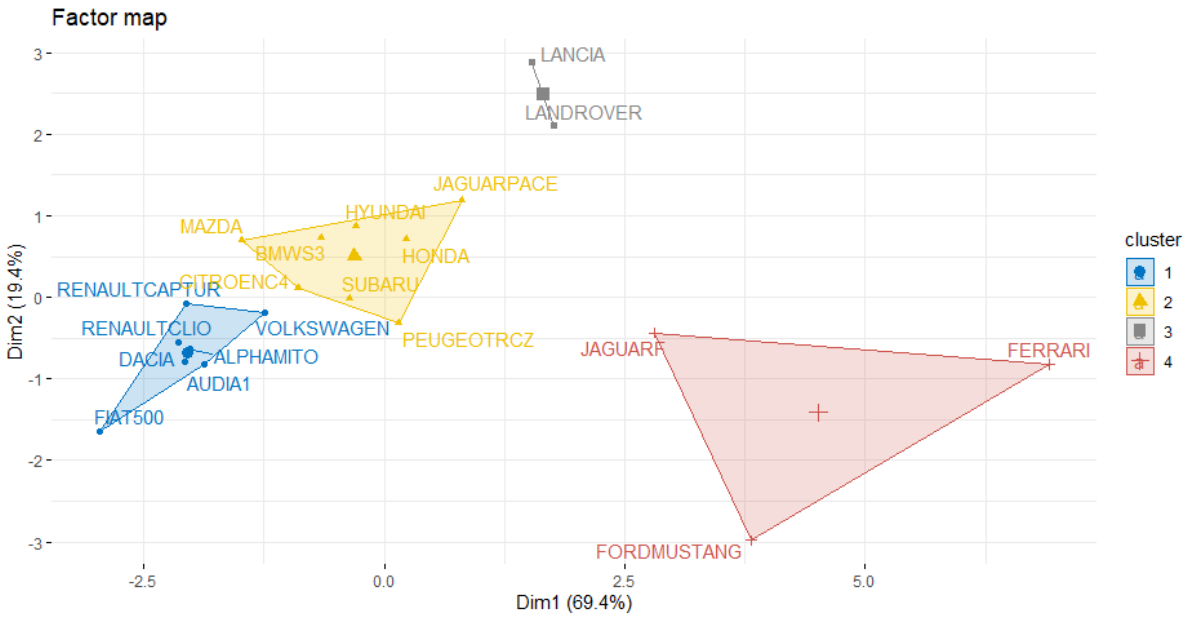


Ce dendrogramme a été créé avec une subdivision finale de 4 groupes, ce qui se produit à un niveau de similarité d'environ 160 (plot(voitures.hc)). Le premier groupe (à l'extrémité gauche) est composé de deux individus (FERRARI et FORDMUSTANG). le deuxième groupe, à sa droite, est composé de 7 individus (RENAULTCAPTUR, ALPHAMITO, RENAULTCLIO, AUDIA1, DACIA, FIAT500 et CITROENC4), le troisième groupe est composé de 3 observations (JAGUARF, LANCIA et LANDROVER), le quatrième groupe, à l'extrémité droite, est composé de 8 individus (MAZDA, VOLKSWAGEN, BMWS3, PEUGEOTRCZ, SUBARU, Honda, Hyundai et JAGUARPACE).

Si vous coupiez le dendrogramme plus haut, les groupes finaux seraient moins nombreux, mais le niveau de similarité serait réduit. Si vous coupiez le dendrogramme plus bas, le niveau de similarité serait supérieur, mais les groupes finaux seraient plus nombreux.

Le dendrogramme suggère une solution à 4 groupes, car le gain d'inertie max tout en minimisant la perte est atteint avec 4 clusters comme le suggèrent les méthodes elbow et Silhouette. Autrement dit, le principe de l'algorithme d'agrégation selon la variance (méthode Ward) consiste à rechercher à chaque étape une partition telle que la variance interne de chaque classe soit minimale.

Voici une représentation graphique de ces cluster :

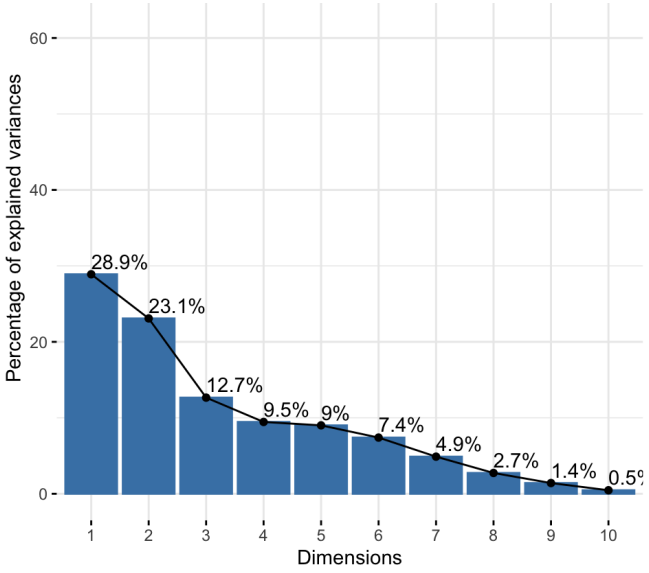


Le choix de 4 clusters semble bien pertinent.

Exercice 4 : Races de chiens

1. *En prenant la variable FON comme variable supplémentaire, faire une analyse des correspondances multiples des données.*

Tout d'abord, on souhaite analyser la variance qui est expliquée par chaque dimension de l'analyse des correspondances multiples (ACM).

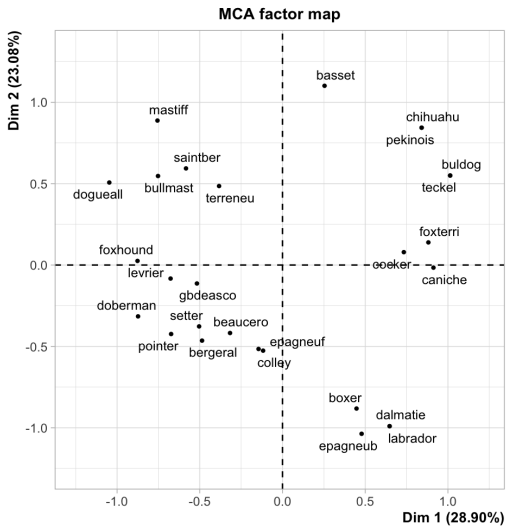


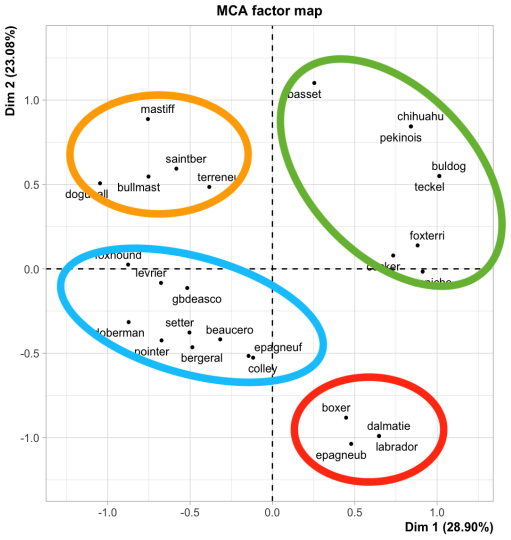
Ce graphique nous montre que 28,9% de l'inertie est expliquée par le premier axe (ou première dimension de l'ACM) tandis que le deuxième axe (ou deuxième dimension de l'ACM) explique 23,1% de la variance. A elles deux (constituant le premier plan factoriel),

elles expliquent 52% de la variance; il est donc pertinent de ne retenir que les deux premiers facteurs.

On utilise le logiciel R pour obtenir l'ACM.

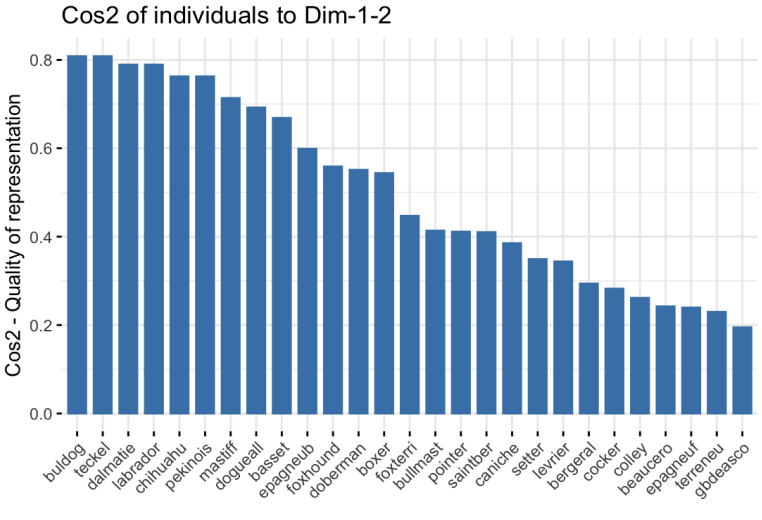
Le premier graphique représente le nuage des individus. A première vue, on peut y distinguer 4 groupes d'individus.





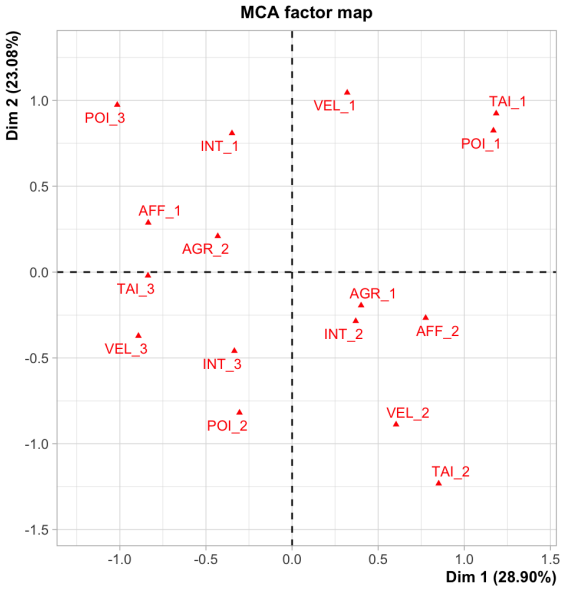
Analysons les cosinus carrés pour éventuellement voir les races de chiens mal représentées.

```
> chiens.mca$ind$cos2
              Dim 1        Dim 2       Dim 3        Dim 4        Dim 5
beaucero 0.08863547 0.1536995944 0.009069781 0.0393822110 1.237222e-02
basset   0.03380431 0.6348671357 0.019038597 0.0448320320 1.437493e-01
bergeral 0.15372250 0.1401636585 0.161231704 0.2166455755 4.946160e-02
boxer    0.11133075 0.4325235284 0.266393303 0.0376048711 1.154621e-01
buldog   0.62448464 0.1838806326 0.016241584 0.0744623393 6.654209e-02
bullmast 0.27069077 0.1429582059 0.118328181 0.2051443062 2.491061e-01
caniche  0.38519392 0.0001212751 0.153853124 0.1826023765 8.717967e-02
chihuahu 0.37993129 0.3826952203 0.118691149 0.0040065748 1.690306e-02
cocker   0.27915682 0.0032460020 0.227671518 0.0186905820 5.683030e-03
colley   0.01239617 0.2492609870 0.100999475 0.3896124090 3.324270e-02
dalmatie 0.23628517 0.5530165596 0.118619154 0.0195798039 1.185067e-02
doberman 0.48761169 0.0636477694 0.130832617 0.1663892363 3.679795e-02
dogueall 0.56079391 0.1314738467 0.013933070 0.0020231792 5.125103e-02
epagneub 0.10498339 0.4939526916 0.001761558 0.1667874837 2.858850e-02
epagneuf 0.01753323 0.2221256292 0.011454493 0.1835972941 6.028475e-07
foxhound 0.55831304 0.0004628928 0.095309358 0.0001678339 3.190765e-01
foxterri 0.43627101 0.0108396312 0.001607917 0.0457802078 4.123268e-02
gbdeasco 0.18602321 0.0089387139 0.001347381 0.0403602327 4.649900e-01
labrador 0.23628517 0.5530165596 0.118619154 0.0195798039 1.185067e-02
levrier  0.33881559 0.0051177295 0.262473933 0.1575993487 9.169119e-02
mastiff  0.29999507 0.4136333336 0.181335511 0.0088542146 1.712283e-02
pekinois 0.37993129 0.3826952203 0.118691149 0.0040065748 1.690306e-02
pointer  0.29459212 0.1167506763 0.305546227 0.0026422615 1.969263e-01
saintber 0.20156282 0.2087298540 0.473604996 0.0105883741 6.353655e-02
setter   0.22389437 0.1252980645 0.073613569 0.4631568432 2.146795e-02
teckel   0.62448464 0.1838806326 0.016241584 0.0744623393 6.654209e-02
terreneu 0.08840069 0.1415315741 0.262465947 0.2022144235 2.447904e-01
```



Une race de chien est "bien" représentée si la somme cumulée dépasse le seuil de 0,5 ou 0,6. On voit que plusieurs races sont mal représentées. Les mieux représentés sont les bulldogs, les teckels et les dalmatiens.

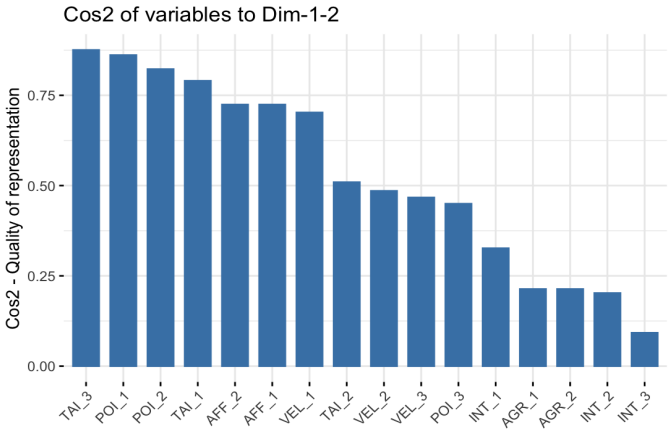
Le deuxième graphique est le nuage des modalités.



Pour le premier axe, on observe que les chiens au poids très lourd (POI_3) sont opposés aux chiens au poids très léger (POI_1). De plus, il y a une opposition entre chiens très grands (TAI_3) et chiens très petits (TAI_1). Enfin, dans une moindre mesure, il y a une opposition entre chiens faiblement affectueux (AFF_1) et fortement affectueux (AFF_2). Du coup, on a d'un côté les chiens lourds, grands et peu affectueux et de l'autre côté les chiens légers, petits et très affectueux.
Pour le deuxième axe, on voit que les chiens lents (VEL_1) sont opposés aux chiens rapides (VEL_3). On observe une opposition entre chiens de petite taille (TAI_1) et chiens de taille moyenne (TAI_2). Enfin on observe une opposition entre chiens très lourds (POI_3) et chiens au poids moyen (POI_2). Du coup, on a les chiens lents, petits et très lourds contre les chiens à taille, poids et vitesse moyens.

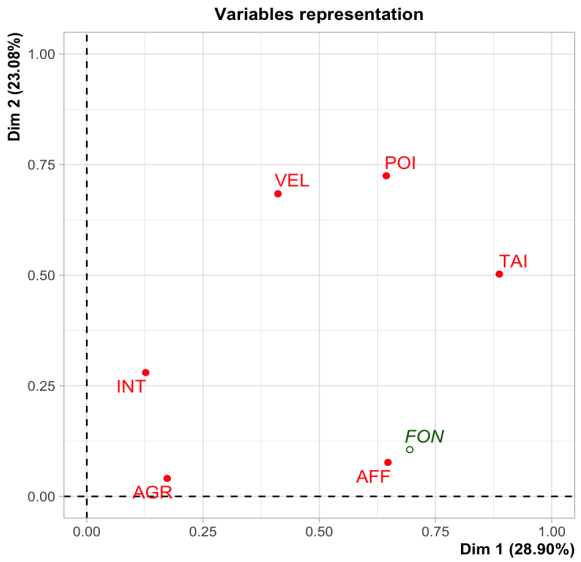
Nous n'avons pas de forte opposition en ce qui concerne l'intelligence et l'agressivité.
Faisons une analyse des cosinus carrés pour savoir si certaines variables sont mal représentées. Le graphique de droite permet de visualiser directement ces variables.

```
> chiens.mca$var$cos2
           Dim 1        Dim 2       Dim 3       Dim 4        Dim 5
TAI_1 0.49144201 0.2987546600 0.132809435 0.005052544 1.394894e-04
TAI_2 0.16462520 0.3448030588 0.234627550 0.026653716 2.184658e-02
TAI_3 0.87503205 0.0005293413 0.003279033 0.036219310 1.586620e-02
POI_1 0.57531341 0.2861238116 0.054196308 0.011447021 1.104688e-03
POI_2 0.10044717 0.7221387844 0.057601141 0.015087719 3.895941e-02
POI_3 0.23420393 0.2155641859 0.339157541 0.001038734 8.582564e-02
VEL_1 0.06021292 0.6422447857 0.094932948 0.003795952 5.504701e-02
VEL_2 0.15344741 0.3318791146 0.053456249 0.057717964 5.796523e-02
VEL_3 0.39792110 0.0691296921 0.291151283 0.028763587 5.089122e-05
INT_1 0.05129787 0.2752677726 0.052025334 0.000247354 4.510944e-01
INT_2 0.12673870 0.0756897524 0.225873819 0.338187895 1.986299e-02
INT_3 0.03207684 0.0603213262 0.102831012 0.464645451 3.229645e-01
AFF_1 0.64765585 0.0767360421 0.003980589 0.006420928 1.750915e-03
AFF_2 0.64765585 0.0767360421 0.003980589 0.006420928 1.750915e-03
AGR_1 0.17292377 0.0406368567 0.103307716 0.282075371 1.302074e-01
AGR_2 0.17292377 0.0406368567 0.103307716 0.282075371 1.302074e-01
```



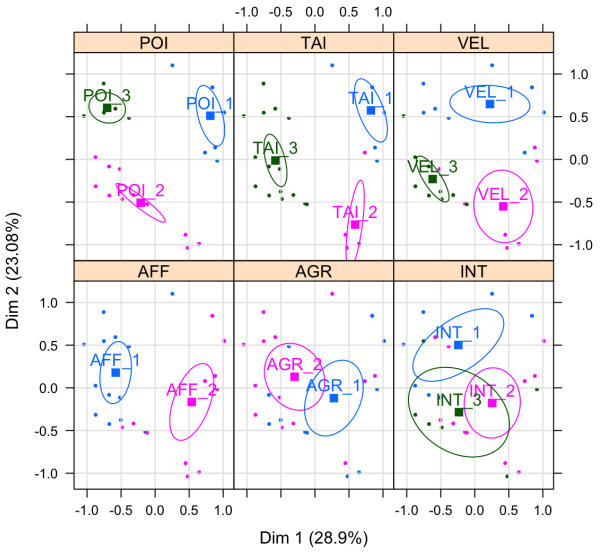
On voit que justement, toutes les variables Intelligence (INT) et Agressivité (AGR) sont très mal représentées dans le premier plan factoriel (très faibles somme des cosinus carrés).

Le troisième graphique est le nuage des variables.
On voit que les variables *Taille (TAI), Poids (POI) et Affectuosité (AFF) (mais surtout TAI)*



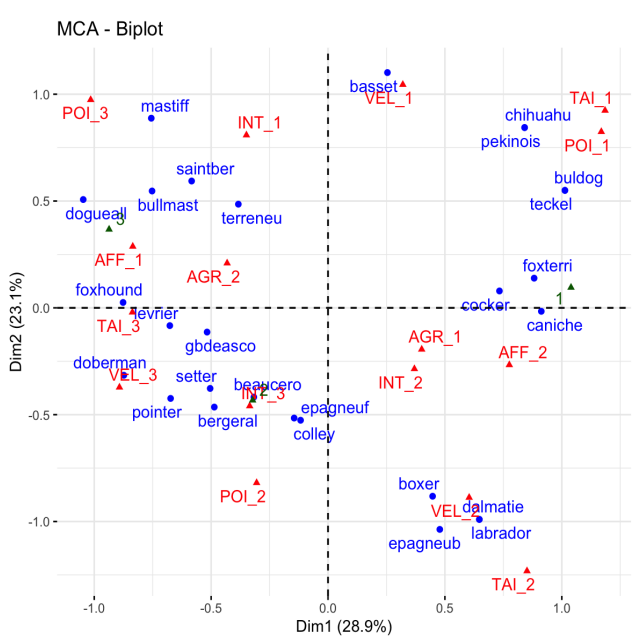
sont très liées à l'axe 1, et les variables *Poids (POI)* et *Vélocité (VEL)* sont très liées à l'axe 2. La variable supplémentaire *Fonction (FON)* est très liée à l'axe 1.

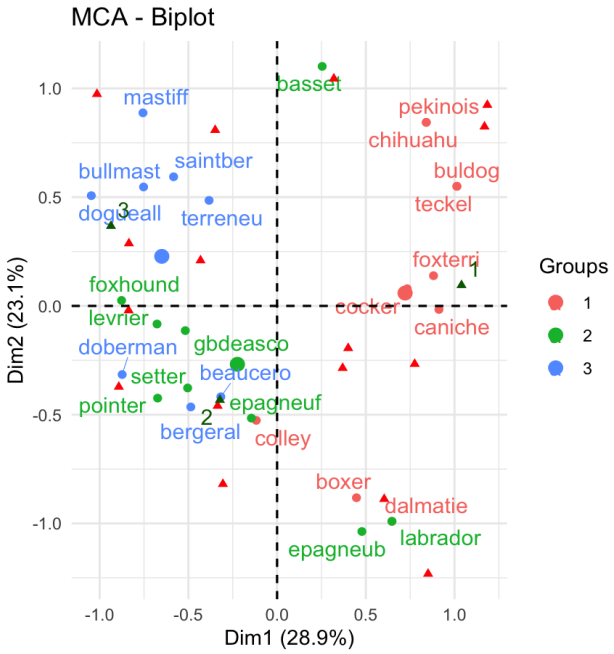
Analysons les ellipses de confiance.



On voit que pour la majorité des variables, les ellipses de confiance ne se touchent pas donc les modalités sont significativement différentes. Au contraire, on peut se questionner quant à la significativité des modalités pour les variables Agressivité et Intelligence.

2. *En déduire une description des différentes races de chiens.*





Le premier graphique nous montre les modalités, individus et la variable supplémentaire ensemble. Le deuxième nous montre les races de chiens classés selon leur fonction (FON), la variable supplémentaire. La variable supplémentaire nous permet de mieux interpréter les graphiques.

Ce graphique nous montre que les chiens considérés comme “chiens de compagnie” (c'est-à-dire FON = 1) sont globalement très affectueux, petits et légers. C’est le cas des chihuahuas, des bulldogs ou encore des pékinois. Sur le deuxième graphique, on voit de manière générale que les chiens de compagnie se situent principalement en haut à droite.
Les chiens considérés comme “chiens de chasse” (FON = 2) sont situés principalement en bas à gauche du graphique. Ce sont plutôt des chiens à poids moyen comme les pointers. Dans ce premier plan factoriel, les chiens de chasse sont plutôt mal représentés.
Enfin les chiens considérés comme “chiens de garde” (FON = 3) sont grands, lourds, très rapides mais peu affectueux. Ils sont placés principalement en haut à gauche. C’est le cas des dogues allemands, des bullmastiffs ou encore des dobermans.

ANNEXE
CODES SOUS LE LOGICIEL R

Exercice 1

```
simu <- read.table("C:\\Users\\Oussama\\Downloads\\donnees\\simu.txt", header = T)
simu
xsimutest <- read.table("C:\\Users\\Oussama\\Desktop\\xsimutest.txt", header=T)
xsimutest

simu$Y[simu$Y==1]<-0
simu$Y[simu$Y==2]<-1
simu

#logit
logit <- glm(Y~ X1 + X2,data=simu, family =  binomial(link = "logit"))
logit
#Cauchit
cauchit<- glm(Y~ X1 + X2,data=simu, family =  binomial(link = "cauchit"))
cauchit

#Prediction
prediction <- predict.glm(cauchit, type= "response", simu)
prediction
table(prediction)

install.packages(knitr)
library(knitr)

confusion <- table(simu$Y,
                   ifelse(prediction <  0.5,
                          "prédit  \"non\"",
                          "prédit  \"oui\""))
kable(confusion)
sum(diag(confusion))/sum(confusion)*100


#KNN
ran <- sample(1:nrow(simu), 0.9 * nrow(simu))
ran


nor <-function(x) { (x -min(x))/(max(x)-min(x))   }
nor

simu_norm <- as.data.frame(lapply(simu[,c(1,2)], nor))

summary(simu_norm)
simu_train <- simu_norm[ran,]
simu_train

simu_test <- simu_norm[-ran,]
simu_test

simu_target_category <- simu[ran,3]
simu_target_category
```

```
simu_test_category <- simu[-ran,3]
simu_test_category


library(class)
install.packages('FNN')
library(FNN)
##run knn function
pr <- knn(simu_train,simu_test,cl=simu_target_category,k=13)
summary(pr)


#Création de la matrice de confusion et accuracy
tab <- table(pr,simu_test_category)
tab

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}

accuracy(tab)


#Question 2
prediction <- knn(simu_train,xsimutest,cl=simu_target_category,k=13)
prediction
summary(prediction)


write.table(prediction,"predictions.txt",row.names=F,col.names=F)

############
```

Exercice 2

```
install.packages(c("FactoMineR", "factoextra"))
library("FactoMineR")
library("factoextra")


voiture <- read.table("C:\\Users\\Oussama\\Desktop\\voitures")
voiture


res.pca <- PCA(voiture, graph = FALSE)
res.pca

#Question 1
get_eigenvalue(res.pca)

#visualisation des valeurs propres
#Critère du coude : on prend l'axe(dim) 1 et 2
fviz_eig(res.pca)
fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 70))

#extraction des résultats pour les individus
get_pca_ind(res.pca)

#extraction des résultats pour les variables
```

```
get_pca_var(res.pca)

res.pca$ind$cos2

#question 2
res.pca$var$cor

#visualisez les résultats des individus
fviz_pca_ind(res.pca)

#visualisez les résultats des variables
fviz_pca_var(res.pca, col.var = "black")

#Création d'un biplot des individus et des variables
fviz_pca_biplot(res.pca)

#Question 3 (a) : voir graphique ou cos2 des individus
res.pca$ind$cos2
#(b) : à l'aide des corrélations des variables aux axes et au biplot
res.pca$var$cor
#(c) : comme précédemment
#(d) : idem
#(e) : regarder si les individus sont bien représenté
res.pca$ind$cos2
#(f) : voir le biplot

##############
```

Exercice 3

```
pkgs <- c("factoextra",  "NbClust")
install.packages(pkgs)
library(factoextra)
library(NbClust)
library("FactoMineR")


voitures <- read.table("C:\\Users\\Oussama\\Desktop\\voitures", header=T)
voitures

#Question 1

# Elbow method : l'empacement d'un coude est généralement considéré comme un
indicateur approprié du nombre de cluster
fviz_nbclust(voitures, kmeans, method = "wss") +
  geom_vline(xintercept = 4, linetype = 2)+
  labs(subtitle = "Elbow method")
#Cette méthode nous donnes 4 clusters


# Silhouette method : Ici l'empacement du maximum est considéré comme le nombre
approprié de clusters
fviz_nbclust(voitures, kmeans, method = "silhouette")+
  labs(subtitle = "Silhouette method")
#Cette méthode nous donnes aussi 4 clusters


# Gap statistic
```

```
#nboot = 50 pour que la fonction reste rapide
#nstart = 25 le nombre de fois où l'algorithme est lancé. Seul le meilleur résultat est conservé
set.seed(123)
fviz_nbclust(voitures, kmeans, nstart = 25, method = "gap_stat", nboot = 50)+
  labs(subtitle = "Gap statistic method")
#Cette méthode nous donne qu'un cluster car elle cherche la plus petite valeur de k selon ses critères

#Ces trois méthodes ne conduisent pas forcément au même résultat. Ici, pour deux d'entre elles, le nombre optimal de clusters est 4.

voitures.km4 <- kmeans(voitures, centers = 4, nstart = 20)
voitures.km4

#Dendogramme
# 1. ACP
res.pca <- PCA(voitures, ncp = 3, graph = TRUE)
res.pca
# 2. HCPC : on applique une classification hiérarchique sur l'acp et on affiche le dendrogramme
res.hcpc <- HCPC(res.pca, method='ward',graph = TRUE)
res.hcpc

#représentation de ces classes dans le graphique d'une acp
fviz_cluster(res.hcpc,
             repel = TRUE,            # Evite le chevauchement des textes
             show.clust.cent = TRUE,  # Montre le centre des clusters
             palette = "jco",         # Palette de couleurs, voir ?ggpubr::ggpar
             ggtheme = theme_minimal(),
             main = "Factor map"
)

#représentation en 3D du dendrogramme
plot(res.hcpc, choice = "3D.map")
```

Exercice 4

```
chiens <- read.table("chiens", header = T)
chiens

install.packages("FactoMineR")
library(FactoMineR)

chiens.active <- chiens[1:27, 1:6]
head(chiens.active[, 1:6], 5)

summary(chiens)[, 1:6]

for(j in 1:6)chiens[,j]<-factor(chiens[,j])
chiens.mca<-MCA(chiens, quali.sup=7)

plot(chiens.mca,invisible=c("ind","quali.sup","quanti.sup"),cex=0.8)

plot(chiens.mca,choix="ind",invisible="ind")

chiens.mca$quali.sup
dimdesc(chiens.mca)
```

```
plot(chiens.mca,invisible=c("ind","quali.sup","quanti.sup"),cex=0.8)
plot(chiens.mca,choix="ind",invisible=c("var","quali.sup"))

plotellipses(chiens.mca)

install.packages("factoextra")
library("factoextra")
fviz_mca_var(chiens.mca, col.var = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE,
             ggtheme = theme_minimal())

eig.val <- get_eigenvalue(chiens.mca)
eig.val

fviz_eig(chiens.mca, addlabels = TRUE, ylim = c(0, 70))

chiens.mca$ind$cos2
chiens.mca$var$cos2

plotellipses(chiens.mca,keepvar=1:6)

dimdesc(chiens.mca)

fviz_mca_biplot(chiens.mca, repel=TRUE)

grp <- as.factor(chiens[, "FON"])
fviz_mca_biplot(chiens.mca,repel=TRUE,label='var',col.var="black",habillage=grp,
addEllipses=FALSE, ellipse.level=0.95) + theme_minimal()


fviz_mca_var(chiens.mca,
col.var="cos2",invisible=c("ind","quali.sup","quanti.sup"))+
             scale_color_gradient2(low="white", mid="blue",
            high="red", midpoint=0.4, space = "Lab")+ theme_minimal()

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