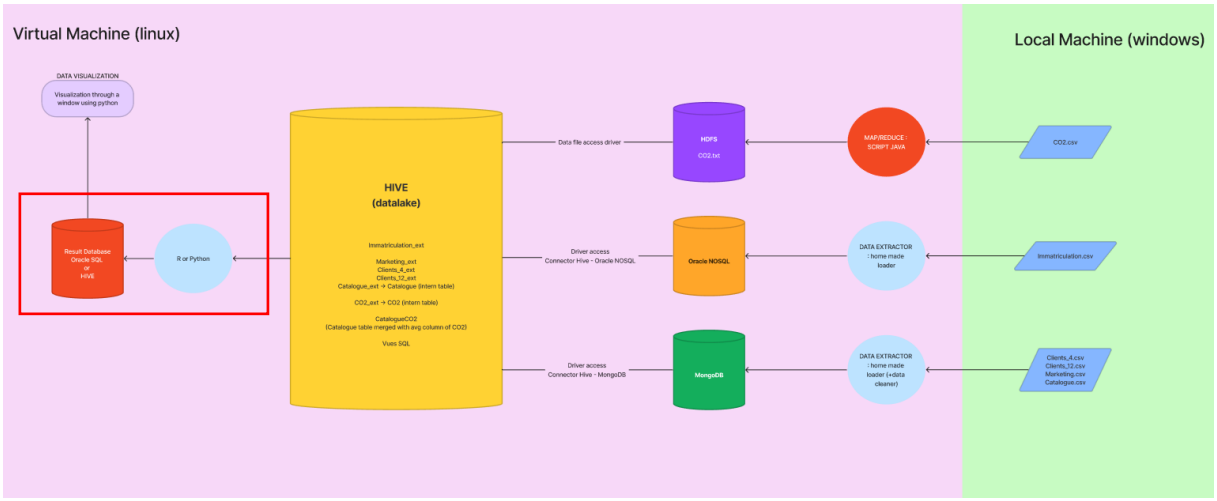
Analyse des données

Table des matières



En Local

Prédisposition

Exporter la table Catalogue en csv sur la VM

```
hive -e 'use projetBigData; select * from catalogue' | sed 's/[\t]/,/g' > Catalogue.csv
```

Copier le csv sur la machine local

```
cp Catalogue.csv ../../vagrant/ProjetTPABigData/R
```

Exporter la table Clients_4 en csv sur la VM

```
hive -e 'use projetBigData; select * from Clients_4_ext' | sed 's/[\t]/,/g' > Clients_4_ext.csv
```

Copier le csv sur la machine local

```
cp Clients_4_ext.csv ../../vagrant/ProjetTPABigData
```

Exporter la table Clients_12 en csv sur la VM

```
hive -e 'use projetBigData; select * from Clients_12_ext' | sed 's/[\t]/,/g' > Clients_12_ext.csv
```

Copier le csv sur la machine local

```
cp Clients_12_ext.csv ../../vagrant/ProjetTPABigData/R
```

Exporter la table Immatriculation en csv sur la VM

```
hive -e 'use projetBigData; select * from Immatriculations_ext' | sed 's/[\t]/,/g' > Immatriculations_ext.csv
```

Copier le csv sur la machine local

```
cp Immatriculations_ext.csv ../../vagrant/ProjetTPABigData/R
```

Analyse des données

Le but de cette partie est d'analyser les données avant les traiter

On importe les csv

```
clients_12_ext <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/clients_12_ext.csv", header
= FALSE)
clients_4_ext <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/clients_4_ext.csv", header =
FALSE)
```

clients_12_ext	98454 obs. of 8 variables
clients_4_ext	98429 obs. of 8 variables

On nomme les colonnes

```
colnames(clients_12_ext) <- c("id", "age", "sexe", "taux",
"situationFamiliale", "nbEnfantsAcharge", "SecondeVoiture", "immatriculation")
colnames(clients_4_ext) <- c("id", "age", "sexe", "taux",
"situationFamiliale", "nbEnfantsAcharge", "SecondeVoiture", "immatriculation")
```

	id	age	sexe	taux	situationFamiliale	nbEnfantsAcharge	SecondeVoiture	immatriculation
1	6409f63d802b1871fd09b7b3	49	f	914	En Couple	1	false	2170DJ60
2	6409f63d802b1871fd09b7b5	18	m	563	En Couple	4	false	8132RT49
3	6409f63d802b1871fd09b7b7	82	m	417	Celibataire	0	false	4764CE84
4	6409f63d802b1871fd09b7b9	72	m	442	En Couple	4	false	6239YO57
5	6409f63d802b1871fd09b7bb	41	m	592	Celibataire	0	false	9318FD10
6	6409f63d802b1871fd09b7bd	54	m	438	En Couple	0	false	984UL93
7	6409f63d802b1871fd09b7bf	30	f	736	En Couple	1	false	3293JS50
8	6409f63d802b1871fd09b7c1	56	m	1325	En Couple	2	false	9773UP95
9	6409f63d802b1871fd09b7c3	73	m	532	Celibataire	0	false	4390XZ41
10	6409f63d802b1871fd09b7c5	31	m	206	En Couple	0	false	6684RV18
11	6409f63d802b1871fd09b7c7	34	m	905	En Couple	0	false	475ZY47
12	6409f63d802b1871fd09b7c9	54	m	539	En Couple	2	false	5193OZ32
13	6409f63d802b1871fd09b7cb	70	m	485	En Couple	0	false	9591PJ95
14	6409f63d802b1871fd09b7cd	82	m	1060	En Couple	4	false	4325RY39
15	6409f63d802b1871fd09b7cf	39	m	189	En Couple	4	false	2837GU95
16	6409f63d802b1871fd09b7d1	77	m	497	En Couple	1	false	4327TI32
17	6409f63d802b1871fd09b7d3	40	m	204	Celibataire	0	false	3428UI66
18	6409f63d802b1871fd09b7d5	62	f	211	En Couple	4	false	9575JQ44
19	6409f63d802b1871fd09b7d7	35	m	219	Celibataire	0	false	7429HW21
20	6409f63d802b1871fd09b7d9	62	m	483	Celibataire	0	false	2749DP77
21	6409f63d802b1871fd09b7db	50	f	236	En Couple	4	false	7995WQ58
22	6409f63d802b1871fd09b7dd	37	f	566	En Couple	2	true	6470WL61
23	6409f63d802b1871fd09b7df	38	m	463	En Couple	0	true	2345VN82
24	6409f63d802b1871fd09b7e1	19	f	534	En Couple	0	false	9679YE10

On combine les deux tables

```
clients_combined <- rbind(clients_12_ext, clients_4_ext)
```

On résume chaque statistique descriptive pour chaque variable numérique

```
summary(clients_combined)
```

```
str(clients_combined)

View(clients_combined)
```

```
> summary(clients_combined)
      id                  age            sexe                taux        situationFamiliale nbEnfantsAcharge
 Length:196883      Min.   :18.00   Length:196883      Min.   : 150.0   Length:196883      Min.   :0.00
 Class :character   1st Qu.:28.00   Class :character   1st Qu.: 421.0   Class :character   1st Qu.:0.00
 Mode  :character   Median :42.00   Mode  :character   Median : 522.0   Mode  :character   Median :1.00
                    Mean   :43.74                      Mean   : 608.9                      Mean   :1.25
                    3rd Qu.:57.00                      3rd Qu.: 828.0                      3rd Qu.:2.00
                    Max.   :84.00                      Max.   :1399.0                      Max.   :4.00
 SecondeVoiture     immatriculation
 Length:196883      Length:196883
 Class :character   Class :character
 Mode  :character   Mode  :character


>
> str(clients_combined)
'data.frame':   196883 obs. of  8 variables:
 $ id                : chr  "6409f7e7802b1871fd0cb8ad" "6409f7e7802b1871fd0cb8af" "6409f7e7802b1871fd0cb8b1" "6409f7e7802b1871f
d0cb8b3" ...
 $ age               : int  58 57 58 18 33 84 19 65 18 53 ...
 $ sexe              : chr  "m" "m" "f" "m" ...
 $ taux              : int  921 462 525 728 1113 1256 515 495 493 229 ...
 $ situationFamiliale: chr  "En Couple" "Celibataire" "En Couple" "Celibataire" ...
 $ nbEnfantsAcharge  : int  4 0 3 0 0 0 4 3 1 0 ...
 $ SecondeVoiture    : chr  "false" "false" "true" "false" ...
 $ immatriculation   : chr  "3684VW75" "4650WY57" "8157VN51" "9715CH60" ...
>
> view(clients_combined)
>
```

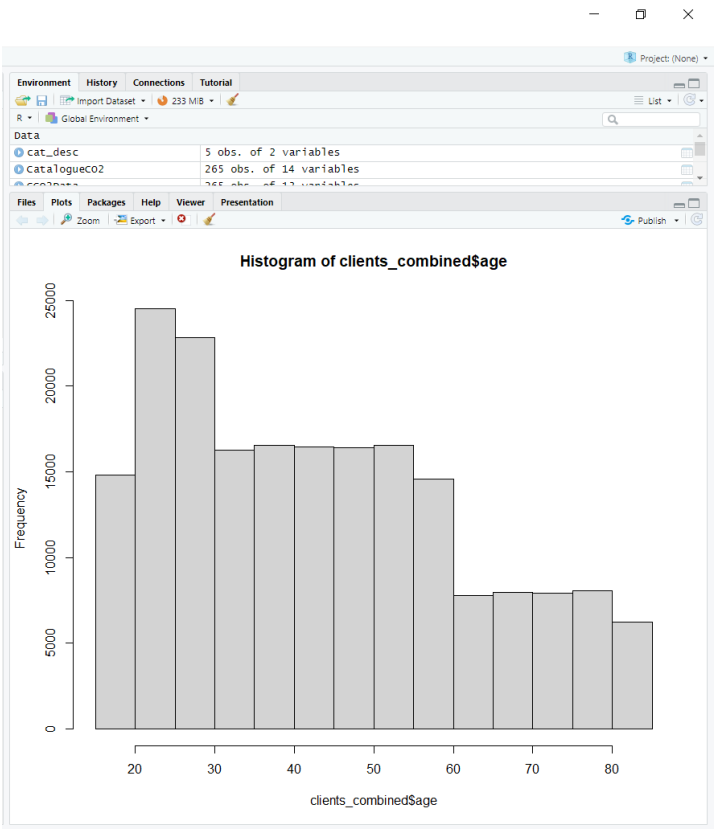
On affiche le nombre de clients_combined par sexe

```
table(clients_combined$sexe)
```

```
> table(clients_combined$sexe)

     f      m
 59218 137665
>
```

On trace un histogramme de la distribution des âges des clients_combined

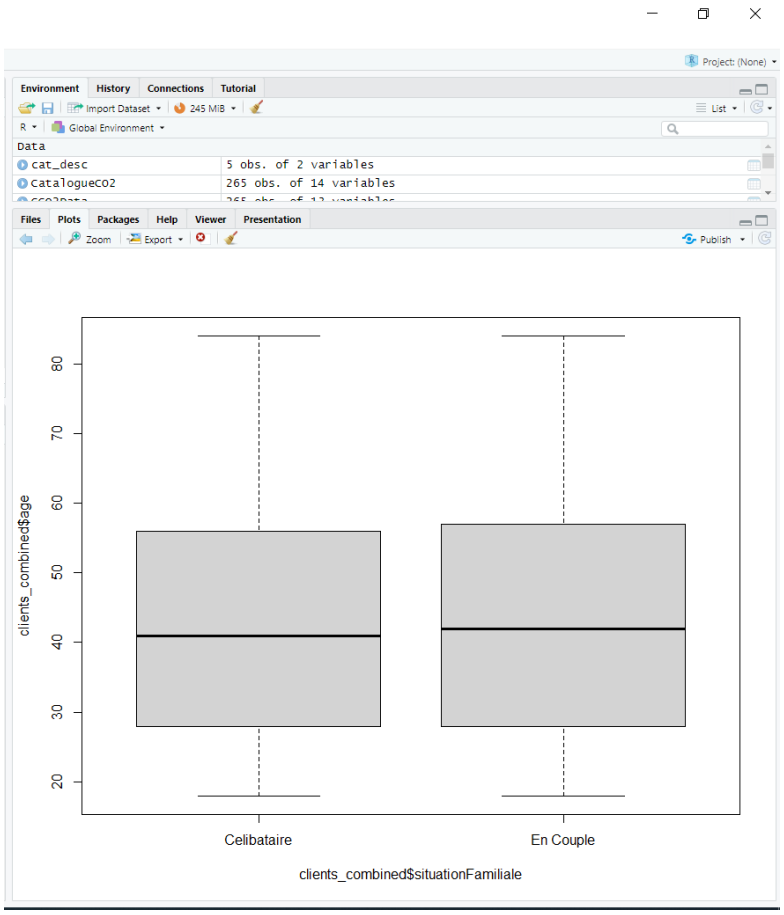
```
hist(clients_combined$age)
```



On remarque que la clientèle est constante de 30 à 60 ans, notamment plus élevée pour la tranche des 20 à 30 ans et la fréquence est très basse pour les clients plus âgés (60 à 90 ans)

On trace un graphique en boîte des âges des clients_combined pour chaque situation familiale

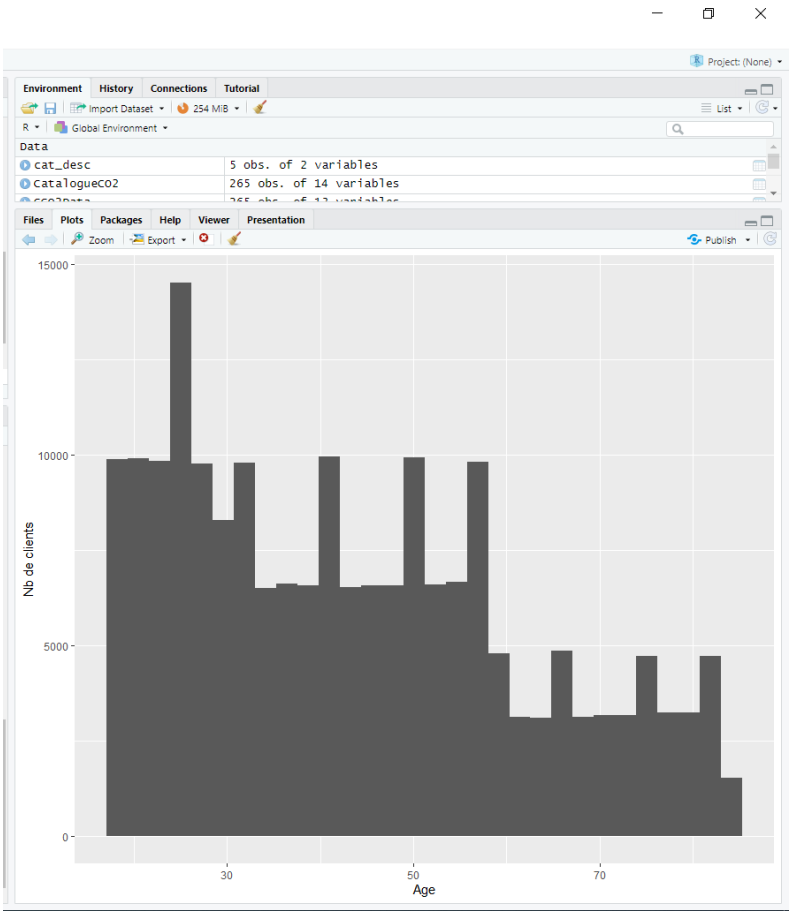
```
boxplot(clients_combined$age ~ clients_combined$situationFamiliale)
```



On remarque ici que la plupart des clients, que ce soit en couple ou célibataire sont dans la tranche des 30 à 55 ans.

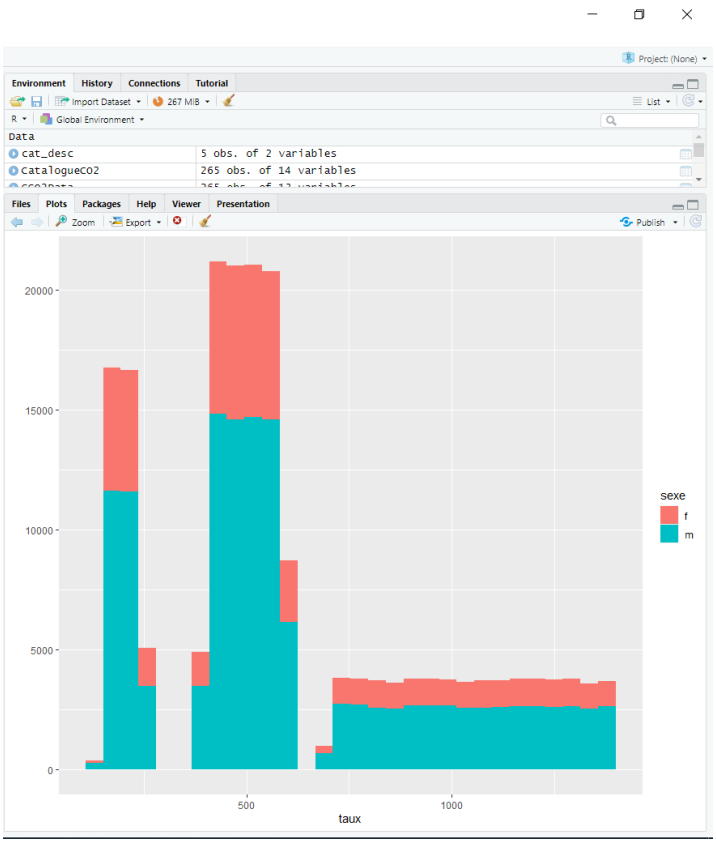
On teste la proportion de l'âge des clients_combined

```
qplot(age, data=clients_combined, xlab="Age", ylab="Nb de clients")
```



Taux selon le sexe ainsi que l'affichage du nb de clients

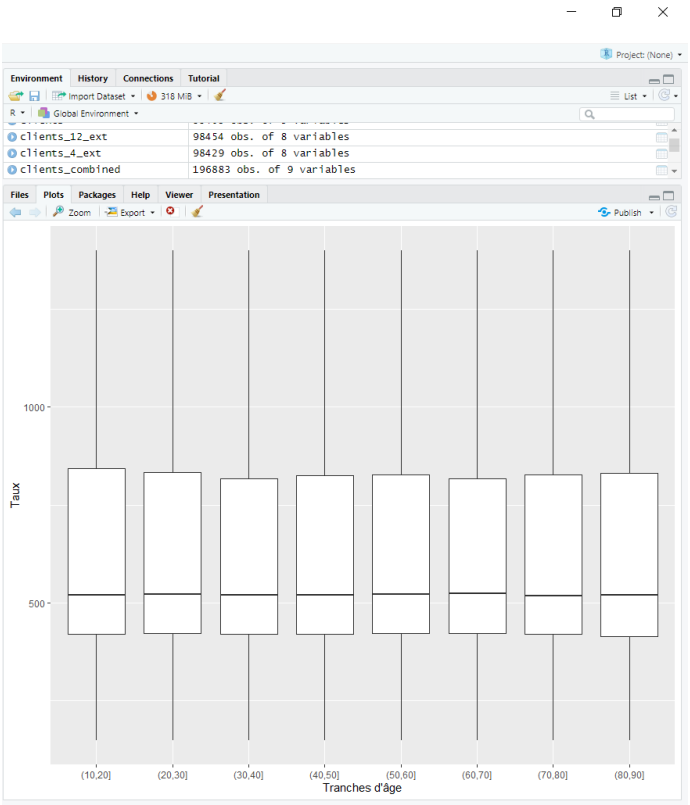
```
qplot(taux, data = subset(clients_combined, sexe %in% c("m", "f")), fill = sexe)
```



Bien que la proportion des hommes soit plus élevée que celles des femmes, on remarque que pour quasiment tous les tranches de taux, le schéma est le même : deux tiers d'hommes et un tiers de femmes.

Intervalles d'âge des clients en boîtes à moustache

```
clients_combined$age_group <- cut(clients_combined$age, breaks = seq(0, 100,
by = 10))
ggplot(na.omit(clients_combined), aes(x=age_group, y=taux)) + geom_boxplot() +
labs(x="Tranches d'âge", y="Taux")
```



On remarque bien ici que malgré les différentes tranches d'âges, le premier et le troisième quartile sont quasiment les mêmes à tous les âges.

Analyse des données sur Catalogue

On crée une table vide avec les noms de colonne spécifiés

```
Catalogue <- data.frame(id = character (),
                        marque = character(),
                        nom = character(),
                        puissance = character(),
                        longueur = character(),
                        nbPlaces = numeric(),
                        nbPortes = numeric(),
                        couleur = character(),
                        occasion = character(),
                        prix = numeric(),
                        stringsAsFactors = TRUE)
```

On importe le fichier CSV

```
CData <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/Catalogue.csv", header =
TRUE, sep = ",", dec = ".",stringsAsFactors = TRUE)
```

On ajoute les données à la table

```
colnames(CData) <- c("id", "marque", "nom", "puissance", "longueur",
"nbPlaces", "nbPortes", "couleur", "occasion", "prix")
Catalogue <- rbind(Catalogue, CData)
```

On supprime l'ID

```
# Supprimer les colonnes id et immatriculation
Catalogue <- subset(Catalogue, select = -c(id))
```

On affiche les premières lignes

```
head(Catalogue, 20)
```

```
> Catalogue <- data.frame(marque = character(),
+                         nom = character(),
+                         puissance = character(),
+                         longueur = character(),
+                         nbPlaces = numeric(),
+                         nbPortes = numeric(),
+                         couleur = character(),
+                         occasion = character(),
+                         prix = numeric(),
+                         stringsAsFactors = TRUE)
> CData <- read.csv("C:/vagrant-projects/OracleDatabase/21.3.0/ProjetTPABigData/R/Catalogue.csv", header = TRUE, sep = ",", d
ec = ".",stringsAsFactors = TRUE)
> colnames(CData) <- c("marque", "nom", "puissance", "longueur", "nbPlaces", "nbPortes", "couleur", "occasion", "prix")
> Catalogue <- rbind(Catalogue, CData)
> head(Catalogue, 20)
       marque            nom puissance    longueur nbPlaces nbPortes couleur occasion  prix
1       Volvo         S80 T6       272 tr\u00e8s longue        5        5   blanc    false 50500
2       Volvo         S80 T6       272 tr\u00e8s longue        5        5    noir    false 50500
3       Volvo         S80 T6       272 tr\u00e8s longue        5        5   rouge    false 50500
4       Volvo         S80 T6       272 tr\u00e8s longue        5        5    gris     true 35350
5       Volvo         S80 T6       272 tr\u00e8s longue        5        5    bleu     true 35350
6       Volvo         S80 T6       272 tr\u00e8s longue        5        5    gris    false 50500
7       Volvo         S80 T6       272 tr\u00e8s longue        5        5    bleu    false 50500
8       Volvo         S80 T6       272 tr\u00e8s longue        5        5   rouge     true 35350
9       Volvo         S80 T6       272 tr\u00e8s longue        5        5   blanc     true 35350
10      Volvo         S80 T6       272 tr\u00e8s longue        5        5    noir     true 35350
11 Volkswagen Touran 2.0 FSI       150      longue        7        5   rouge    false 27340
12 Volkswagen Touran 2.0 FSI       150      longue        7        5    gris     true 19138
13 Volkswagen Touran 2.0 FSI       150      longue        7        5    bleu     true 19138
14 Volkswagen Touran 2.0 FSI       150      longue        7        5    gris    false 27340
15 Volkswagen Touran 2.0 FSI       150      longue        7        5    bleu    false 27340
16 Volkswagen Touran 2.0 FSI       150      longue        7        5   blanc     true 19138
17 Volkswagen Touran 2.0 FSI       150      longue        7        5    noir     true 19138
18 Volkswagen Touran 2.0 FSI       150      longue        7        5   rouge     true 19138
19 Volkswagen Touran 2.0 FSI       150      longue        7        5   blanc    false 27340
20 Volkswagen Touran 2.0 FSI       150      longue        7        5    noir    false 27340
> |
```

On affiche les informations sur les variables

```
str(Catalogue)
```

```
> str(Catalogue)
'data.frame':   270 obs. of  9 variables:
 $ marque   : Factor w/ 21 levels "Audi","BMW","Dacia",..: 21 21 21 21 21 21 21 21 21 21 ...
 $ nom      : Factor w/ 32 levels "1007 1.4","120i",..: 26 26 26 26 26 26 26 26 26 26 ...
 $ puissance: int  272 272 272 272 272 272 272 272 272 272 ...
 $ longueur : Factor w/ 4 levels "courte","longue",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ nbPlaces : int  5 5 5 5 5 5 5 5 5 5 ...
 $ nbPortes : int  5 5 5 5 5 5 5 5 5 5 ...
 $ couleur  : Factor w/ 5 levels "blanc","bleu",..: 1 4 5 3 2 3 2 5 1 4 ...
 $ occasion : Factor w/ 2 levels "false","true": 1 1 1 2 2 1 1 2 2 2 ...
 $ prix     : int  50500 50500 50500 35350 35350 50500 50500 35350 35350 35350 ...
>
```

Analyse des données sur Catalogue (sous-partie)

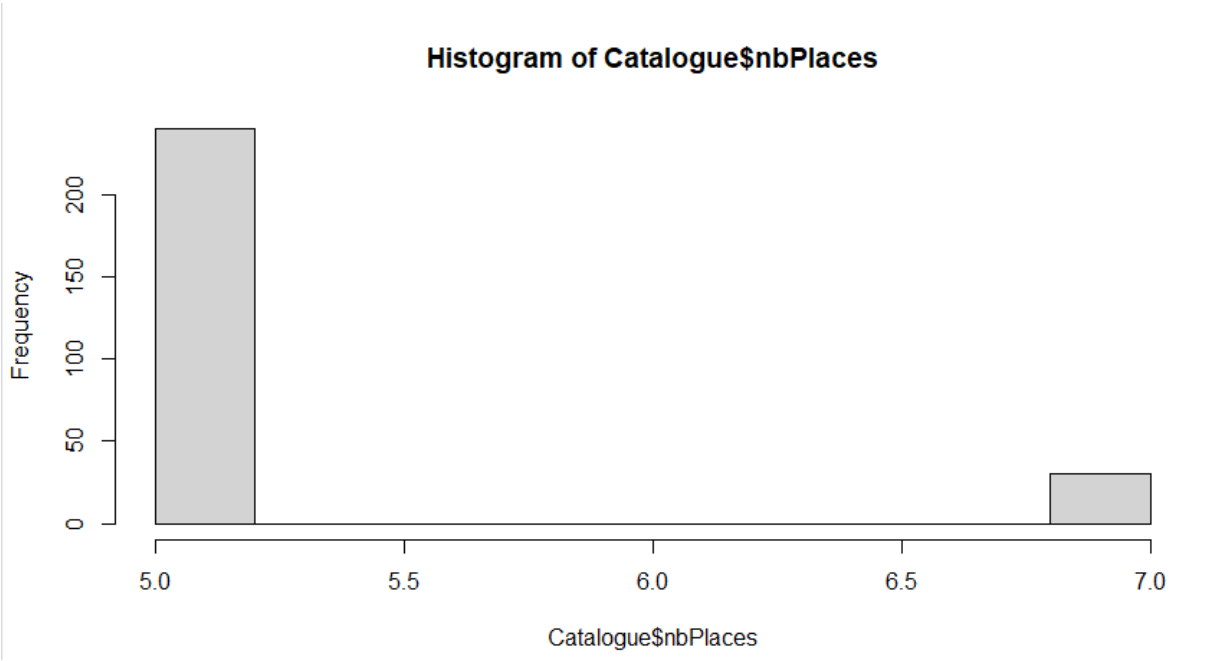
On résume les statistiques descriptives pour chaque variable numérique

```
summary(Catalogue)
summary(Catalogue[,c("nbPlaces", "nbPortes", "prix")])
```

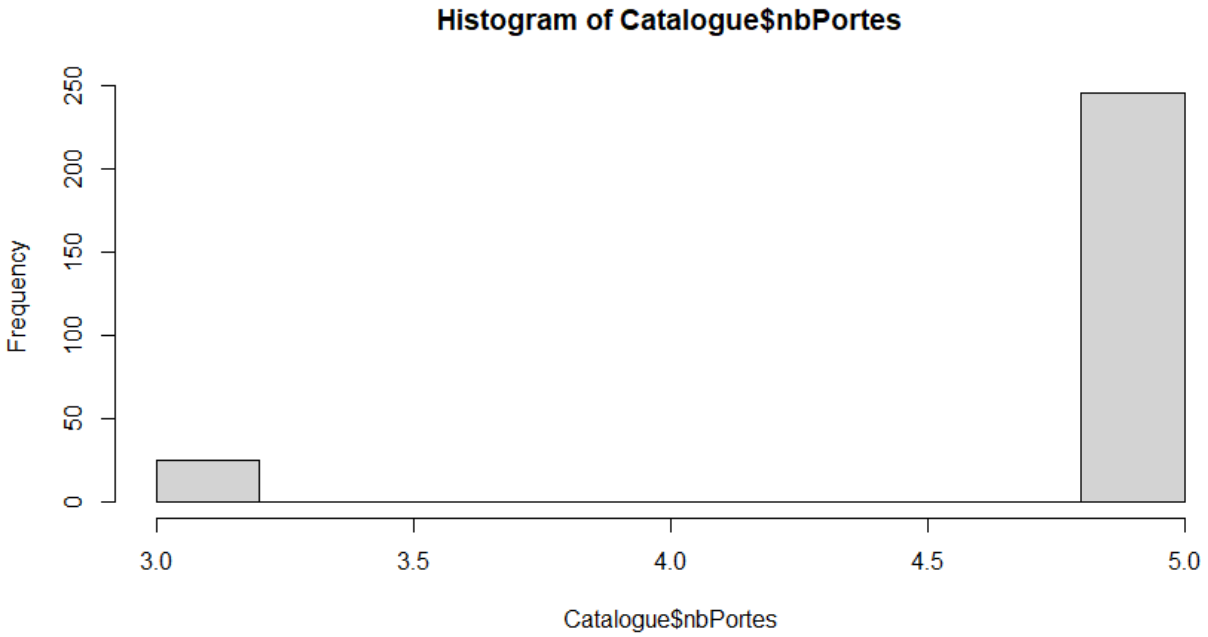
```
> summary(Catalogue)
       marque           nom        puissance          longueur     nbPlaces        nbPortes      couleur
 Renault   : 40   1007 1.4  : 10   Min.   : 55.0   courte     :60   Min.   :5.000   Min.   :3.000   blanc:54
 Volkswagen: 40   120i      : 10   1st Qu.:109.0   longue     :90   1st Qu.:5.000   1st Qu.:5.000   bleu :54
 Audi      : 20   9.3 1.8T  : 10   Median :147.0   moyenne    :70   Median :5.000   Median :5.000   gris :54
 BMW       : 20   A2 1.4    : 10   Mean   :157.6   tr<U+FFFD>s longue:50   Mean   :5.222   Mean   :4.815   noir :54
 Mercedes  : 20   A200      : 10   3rd Qu.:170.0                   3rd Qu.:5.000   3rd Qu.:5.000   rouge:54
 Nissan    : 15   A3 2.0 FSI: 10   Max.   :507.0                   Max.   :7.000   Max.   :5.000
 (Other)   :115   (Other)   :210
  occasion        prix
 false:160   Min.   :  7500
 true :110   1st Qu.: 16029
             Median : 20598
             Mean   : 26668
             3rd Qu.: 30000
             Max.   :101300

>
> summary(Catalogue[,c("nbPlaces", "nbPortes", "prix")])
    nbPlaces        nbPortes          prix
 Min.   :5.000   Min.   :3.000   Min.   :  7500
 1st Qu.:5.000   1st Qu.:5.000   1st Qu.: 16029
 Median :5.000   Median :5.000   Median : 20598
 Mean   :5.222   Mean   :4.815   Mean   : 26668
 3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.: 30000
 Max.   :7.000   Max.   :5.000   Max.   :101300
>
```

Histogrammes des variables numériques

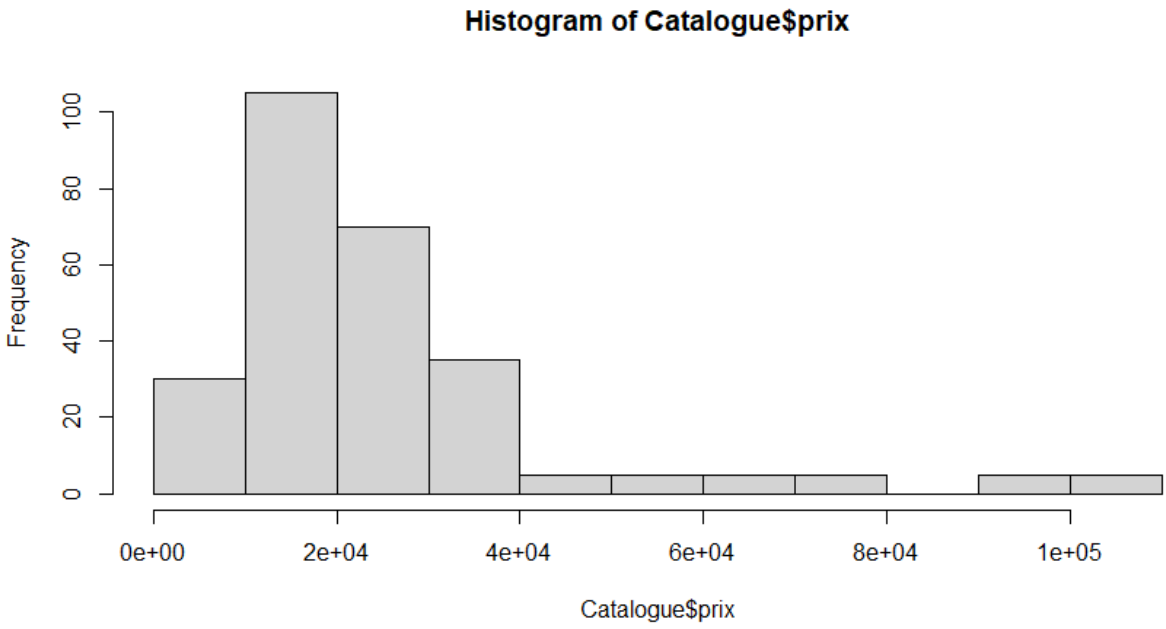
```
hist(Catalogue$nbPlaces)
```




```
hist(Catalogue$nbPortes)
```




```
hist(Catalogue$prix)
```



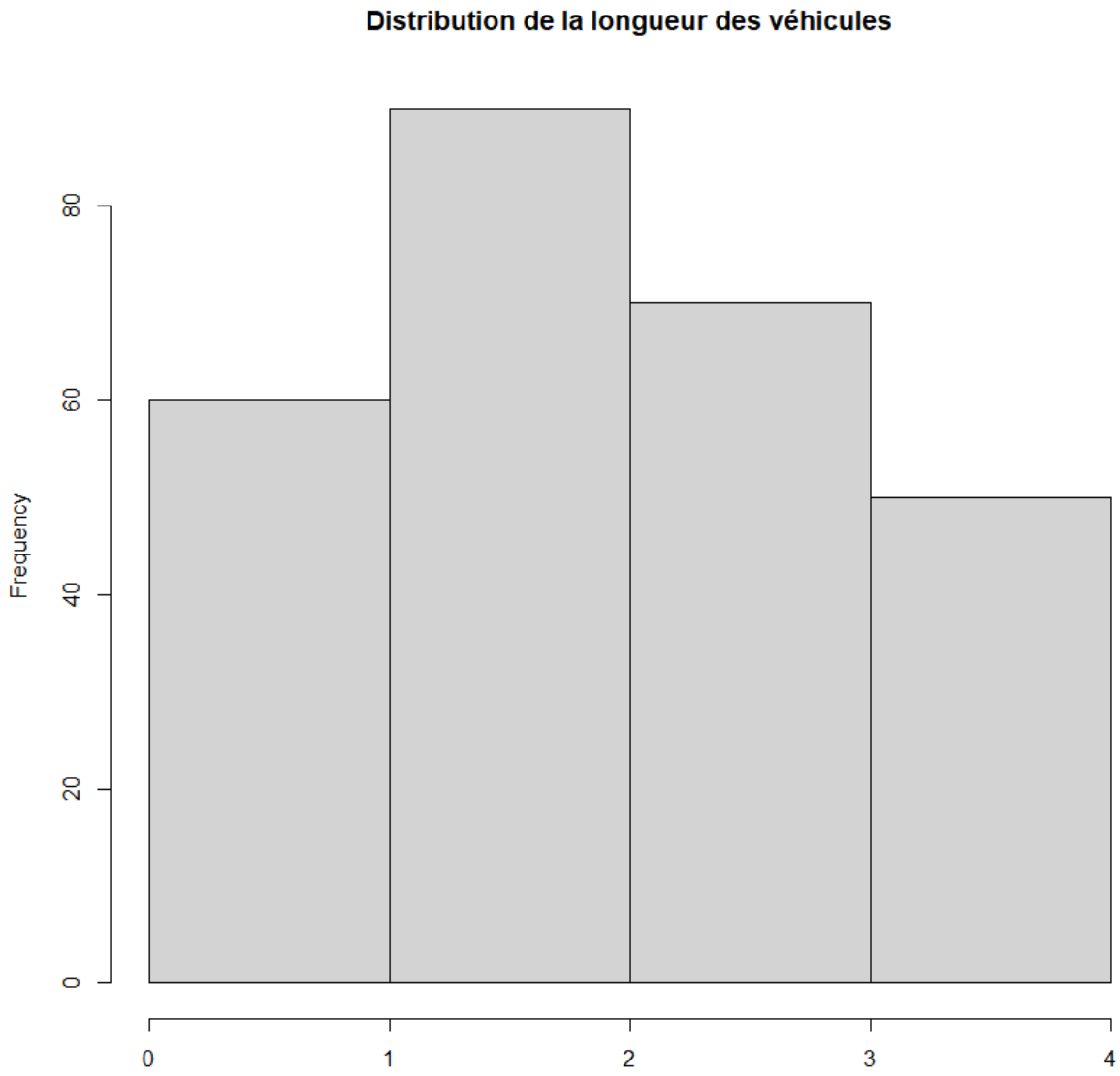
On observe que les prix sont très variables, ce qui est logique au vu des nombreuses catégories de véhicules disponibles.

Conversion de la colonne "longueur" en format numérique

```
Catalogue$longueur <- as.numeric(Catalogue$longueur)
```

Histogramme de la longueur

```
hist(Catalogue$longueur, breaks = seq(0, 4, 1),
     main = "Distribution de la longueur des véhicules", xlab = "Longueur (m)")
```



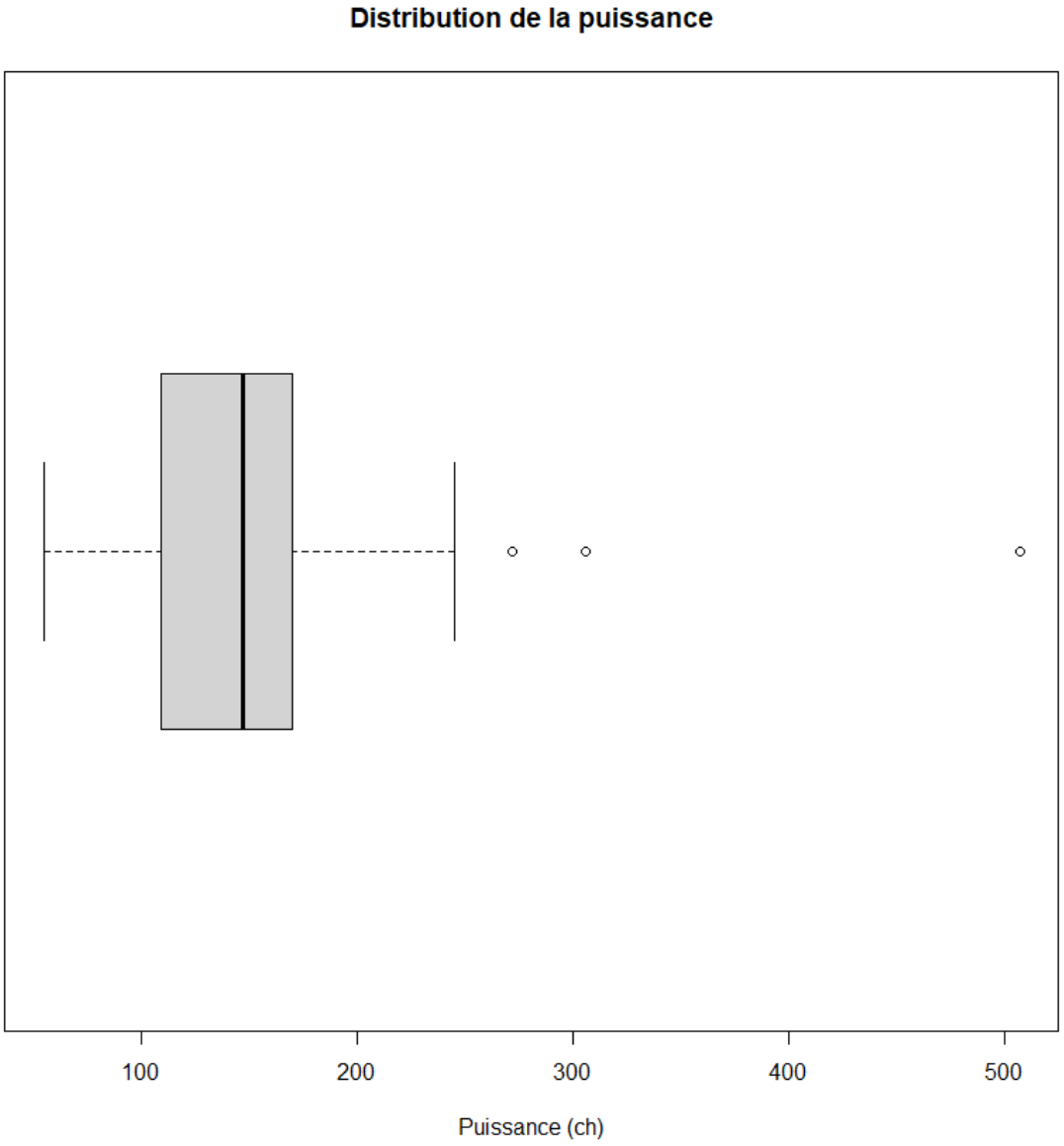
Ici 1 correspond à « Court », 2 à « Moyenne », 3 à « Longue », 4 à « Très Longue » comme dans le cahier des charges.

Boîtes à moustaches des variables numériques

```
boxplot(Catalogue$prix)
```

Boîte à moustaches de la puissance

```
boxplot(Catalogue$puissance, horizontal = TRUE, main = "Distribution de la puissance", xlab = "Puissance (ch)")
```



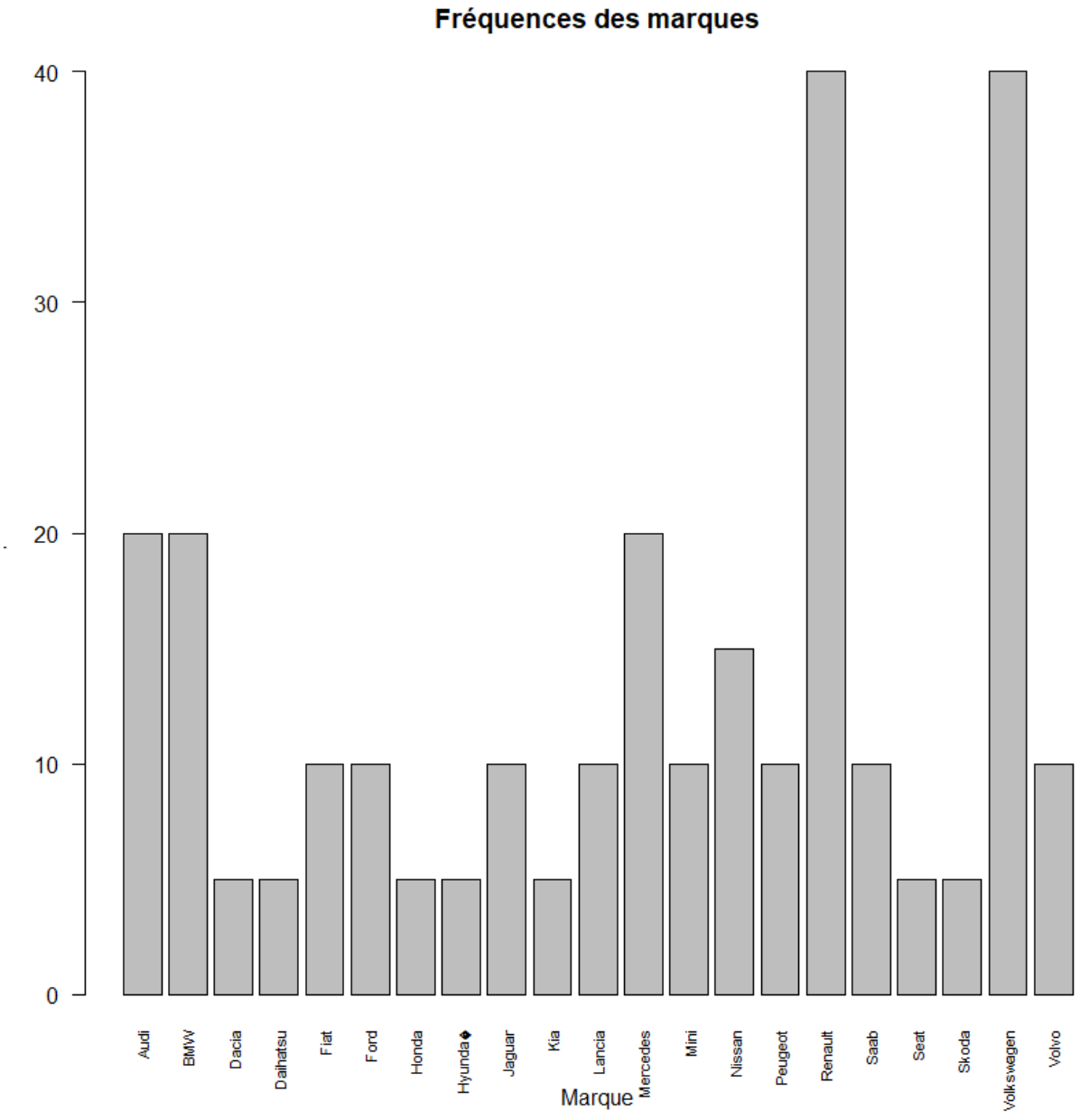
Fréquences des marques

```
table(Catalogue$marque)
```

```
> table(Catalogue$marque)

      Audi        BMW      Dacia   Daihatsu       Fiat       Ford      Honda   Hyunda?    Jaguar        Kia     Lancia
        20         20          5          5         10         10          5          5         10          5         10
  Mercedes       Mini     Nissan    Peugeot    Renault       Saab       Seat      Skoda Volkswagen      Volvo
        20         10         15         10         40         10          5          5         40         10
```

Histogramme des fréquences des marques

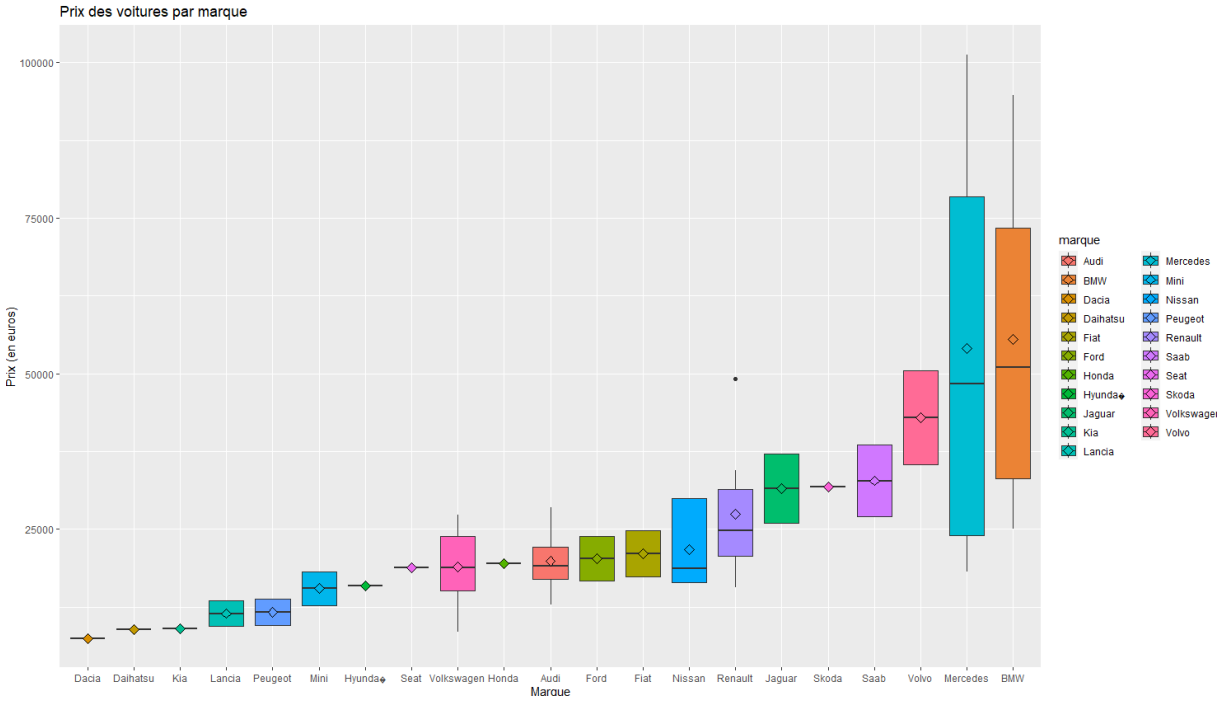
```
barplot(table(Catalogue$marque), las = 2, cex.names = 0.7, main = "Fréquences
des marques", ylab = "Fréquence", xlab = "Marque")
```



Graphique affichant le prix des voitures par marque avec la moyenne en losange par marque

```
ggplot(Catalogue, aes(x = reorder(marque, prix), y = prix, fill = marque)) +
  geom_boxplot() +
```

```
  stat_summary(
    aes(group = marque),
    fun = "mean",
    geom = "point",
    shape = 23,
    size = 3,
    color = "black"
  ) +
  labs(title = "Prix des voitures par marque", x = "Marque", y = "Prix (en
euros)")
```



Analyse des données sur Immatriculation

On affiche tout d'abord le résumé des statistiques descriptives pour chaque variable numérique. On remarque qu'il n'y a aucune valeur impromptue ou erronées.

```
# Résumé des statistiques descriptives pour chaque variable numérique
summary(Immatriculations)
```

```
> summary(Immatriculations)
 immatriculation           marque                 nom          puissance       longueur        nbPlaces
 1007LX38:      2   BMW       :154301   A21.4        :149337   Min.   : 55.0   Min.   :1.000   Min.   :5
 1018QC21:      2   Audi      :153630   M5           :134439   1st Qu.: 75.0   1st Qu.:1.000   1st Qu.:5
 102NB76 :      2   Renault   :131735   X-Type2.5V6  : 98592   Median :150.0   Median :2.000   Median :5
 1033TS14:      2   Jaguar    : 98592   S80T6        : 64803   Mean   :198.9   Mean   :2.514   Mean   :5
 1041MB78:      2   Volkswagen: 81885   VelSatis3.5V6: 64518   3rd Qu.:245.0   3rd Qu.:4.000   3rd Qu.:5
 1100CO10:      2   Mercedes  : 78769   S500         : 54475   Max.   :507.0   Max.   :4.000   Max.   :5
 (Other) :1048563   (Other)   :349663   (Other)      :482411
    nbPortes      couleur       occasion          prix
 Min.   :3.000   blanc:209075   false:721044   Min.   :  7500
 1st Qu.:5.000   bleu :209932   true :327531   1st Qu.: 18310
 Median :5.000   gris :210226                  Median : 25970
 Mean   :4.868   noir :209729                  Mean   : 35767
 3rd Qu.:5.000   rouge:209613                  3rd Qu.: 49200
 Max.   :5.000                                 Max.   :101300
```

On fait de même mais avec le nombre de place, le nombre de portes et le prix

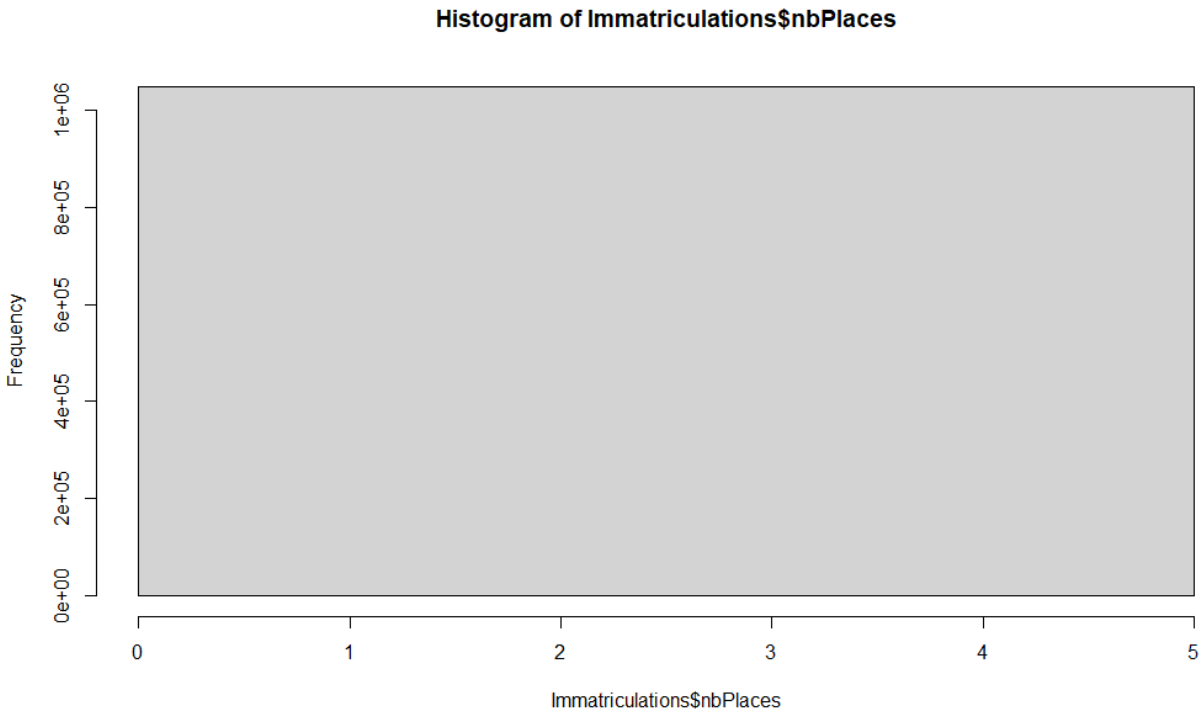
```
summary(Immatriculations[,c("nbPlaces", "nbPortes", "prix")])
```

```
> summary(Immatriculations[,c("nbPlaces", "nbPortes", "prix")])
    nbPlaces     nbPortes          prix
 Min.   :5   Min.   :3.000   Min.   :  7500
 1st Qu.:5   1st Qu.:5.000   1st Qu.: 18310
 Median :5   Median :5.000   Median : 25970
 Mean   :5   Mean   :4.868   Mean   : 35767
 3rd Qu.:5   3rd Qu.:5.000   3rd Qu.: 49200
 Max.   :5   Max.   :5.000   Max.   :101300
```

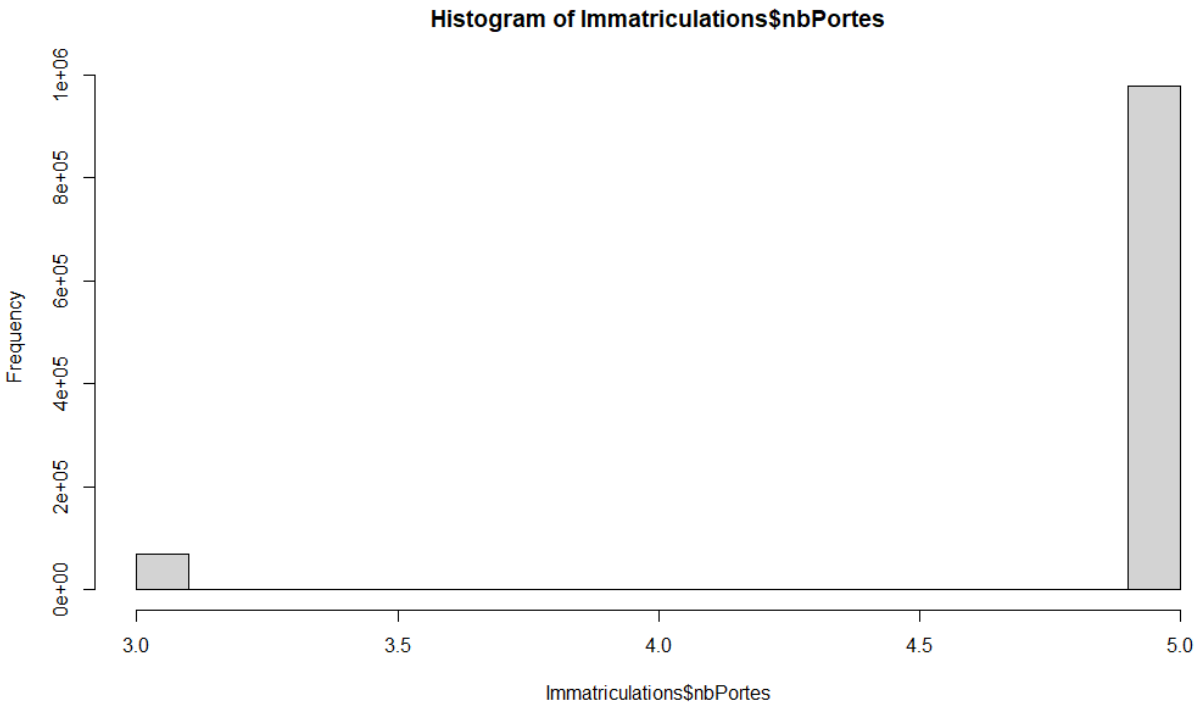
On affiche maintenant l'histogramme de ces trois différentes variables numériques selon leur immatriculation.

Tout d'abord le nombre de places :

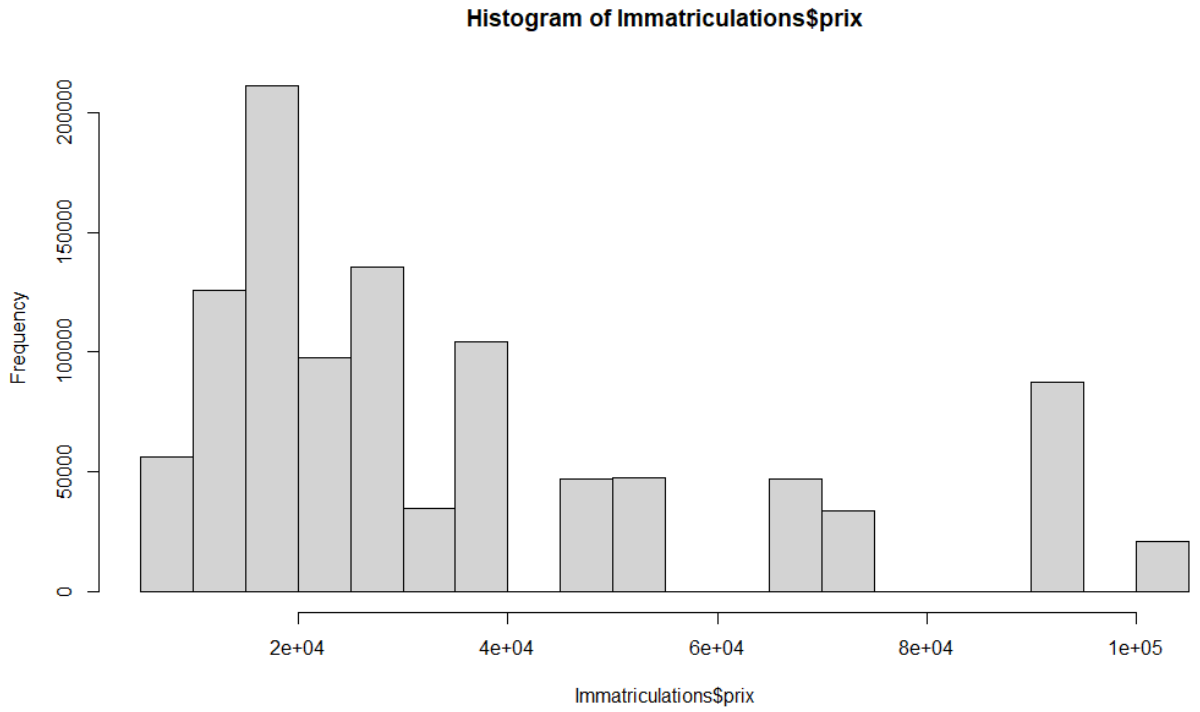
```
hist(Immatriculations$nbPlaces)
```



Puis le nombre de portes :

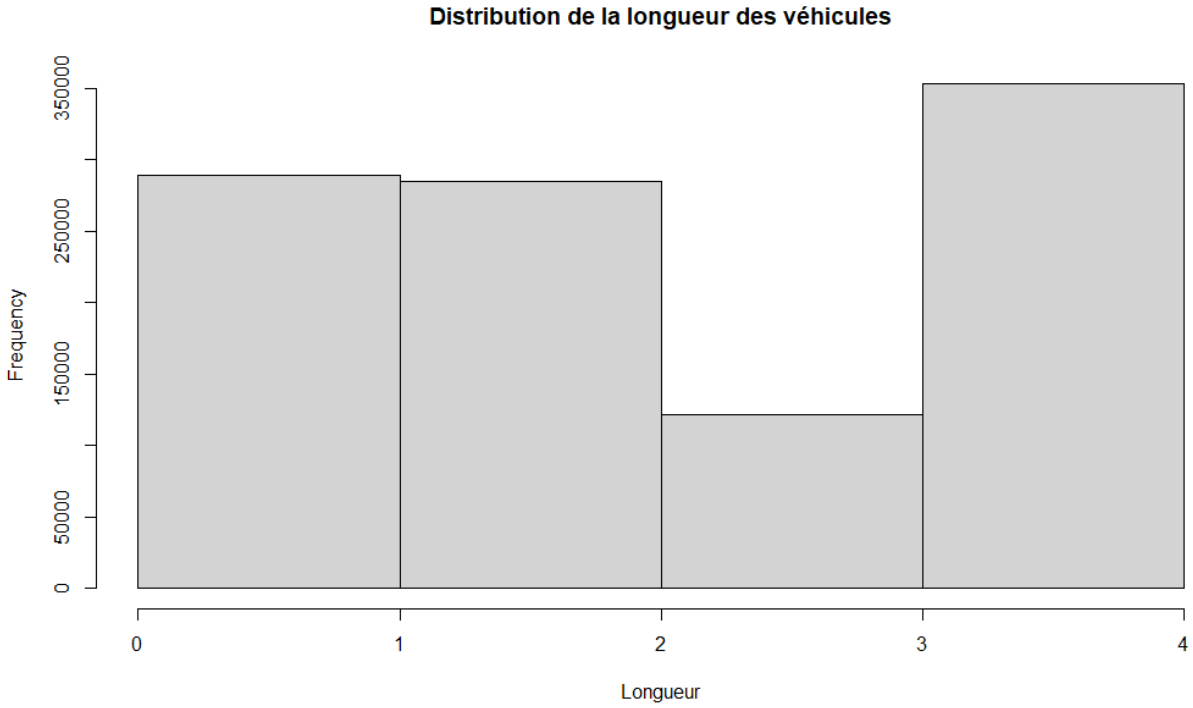


Puis le prix :



Voici un dernier histogramme représentant la longueur selon l'immatriculation

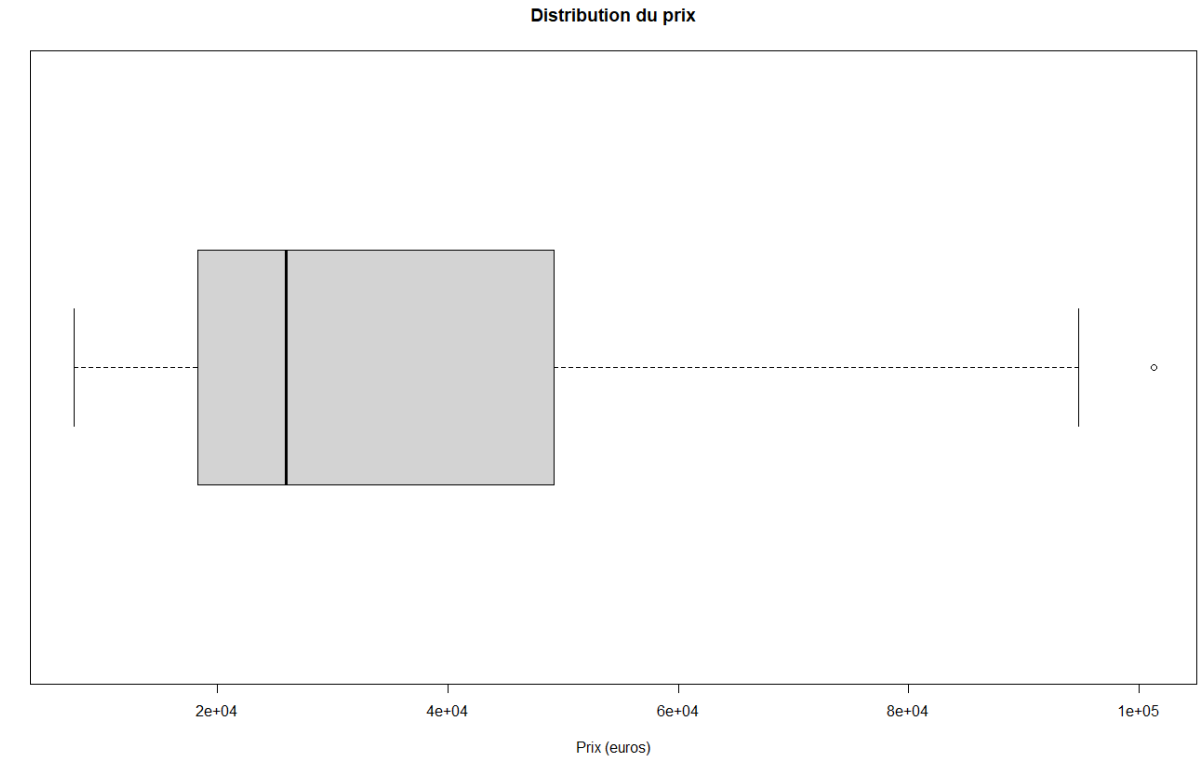
```
# conversion de la colonne "longueur" en format numérique
Immatriculations$longueur <- as.numeric(Immatriculations$longueur)
# histogramme de la longueur
hist(Immatriculations$longueur, breaks = seq(0, 4, 1),
     main = "Distribution de la longueur des véhicules", xlab = "Longueur")
```



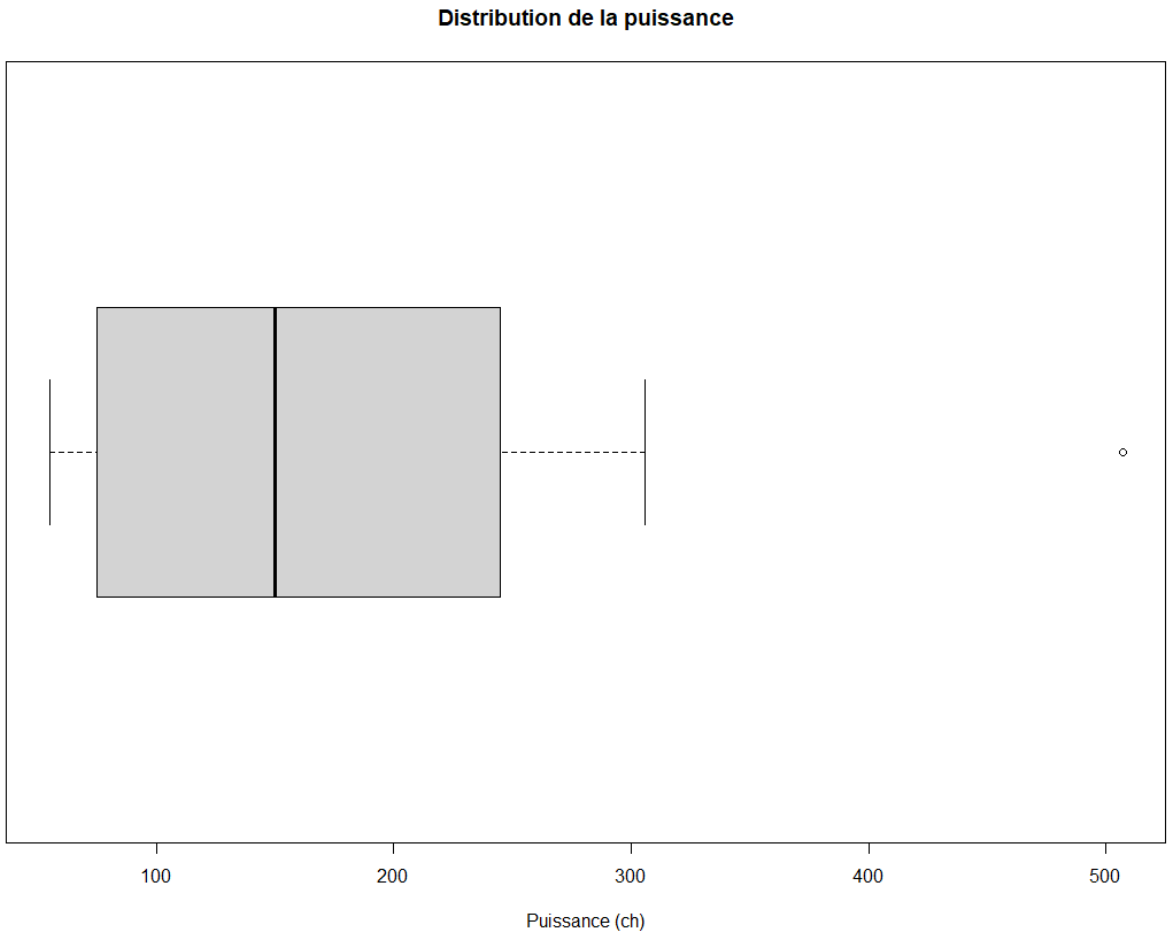
On affiche ensuite les boîtes à moustaches de certains paramètres.

On commence par le prix

```
# Boîtes à moustaches des variables numériques
boxplot(Immatriculations$prix)
```

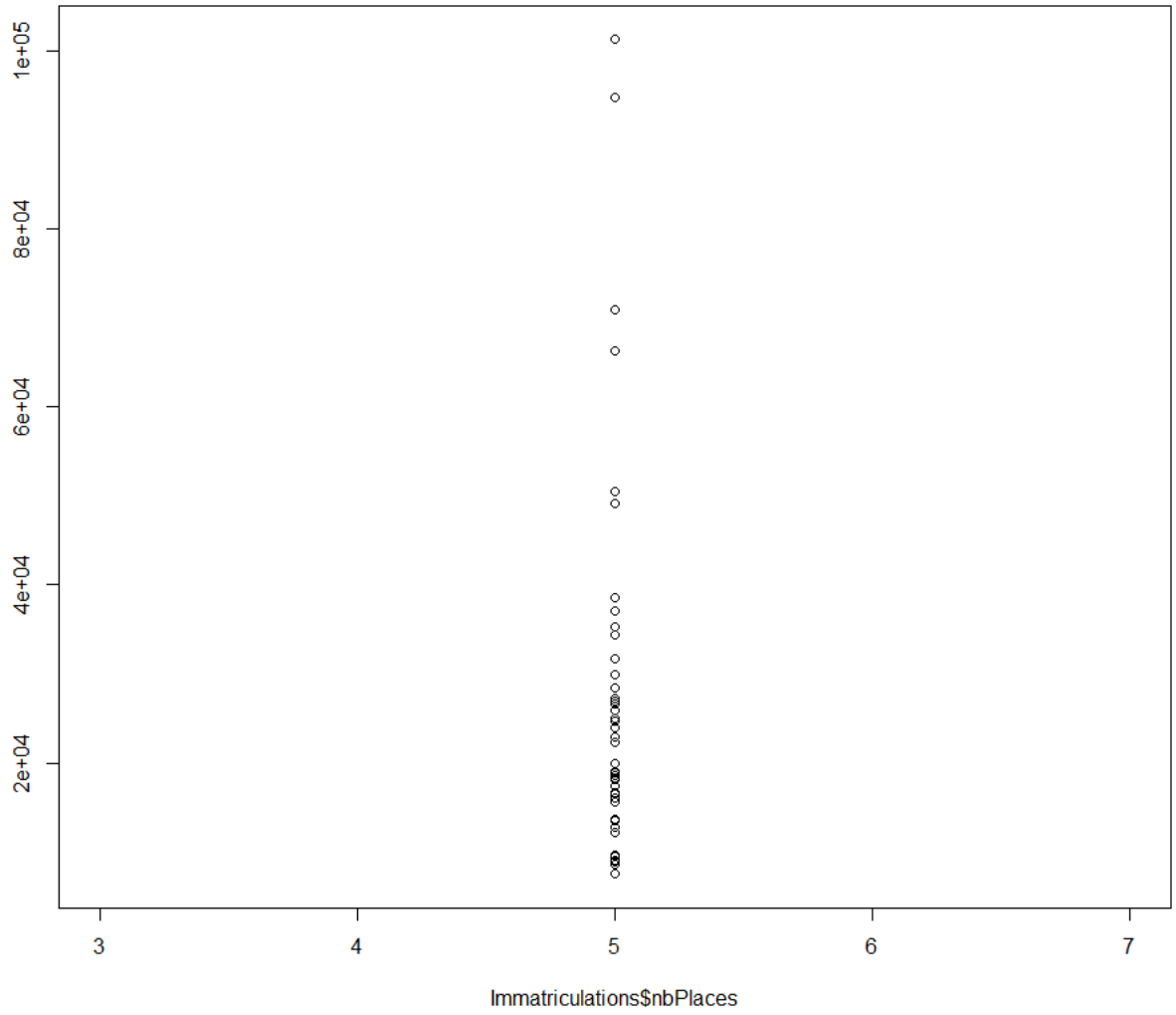


```r
# Boîte à moustaches de la puissance
boxplot(Immatriculations$puissance, horizontal = TRUE, main = "Distribution de la puissance", xlab = "Puissance (ch)")
```



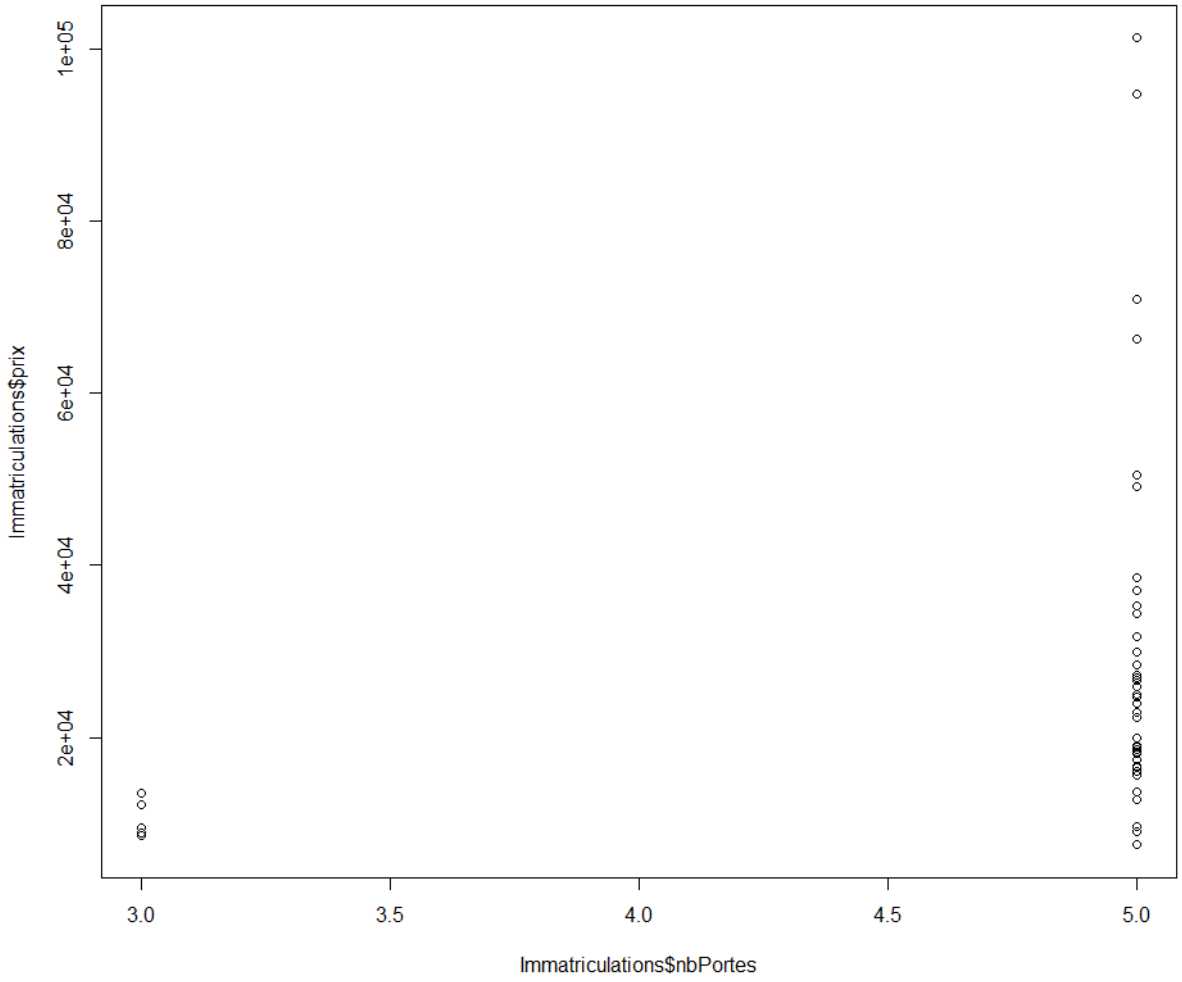
On va maintenant afficher un nuage de points avec comme paramètre prix en ordonnée, et le paramètre nombre de places en abscisse.

```r
# Nuages de points pour les relations entre les variables numériques
plot(Immatriculations$nbPlaces, Immatriculations$prix)
```



Maintenant le même nuage mais avec le nombre de portes en abscisse :

```
plot(Immatriculations$nbPortes, Immatriculations$prix)
```



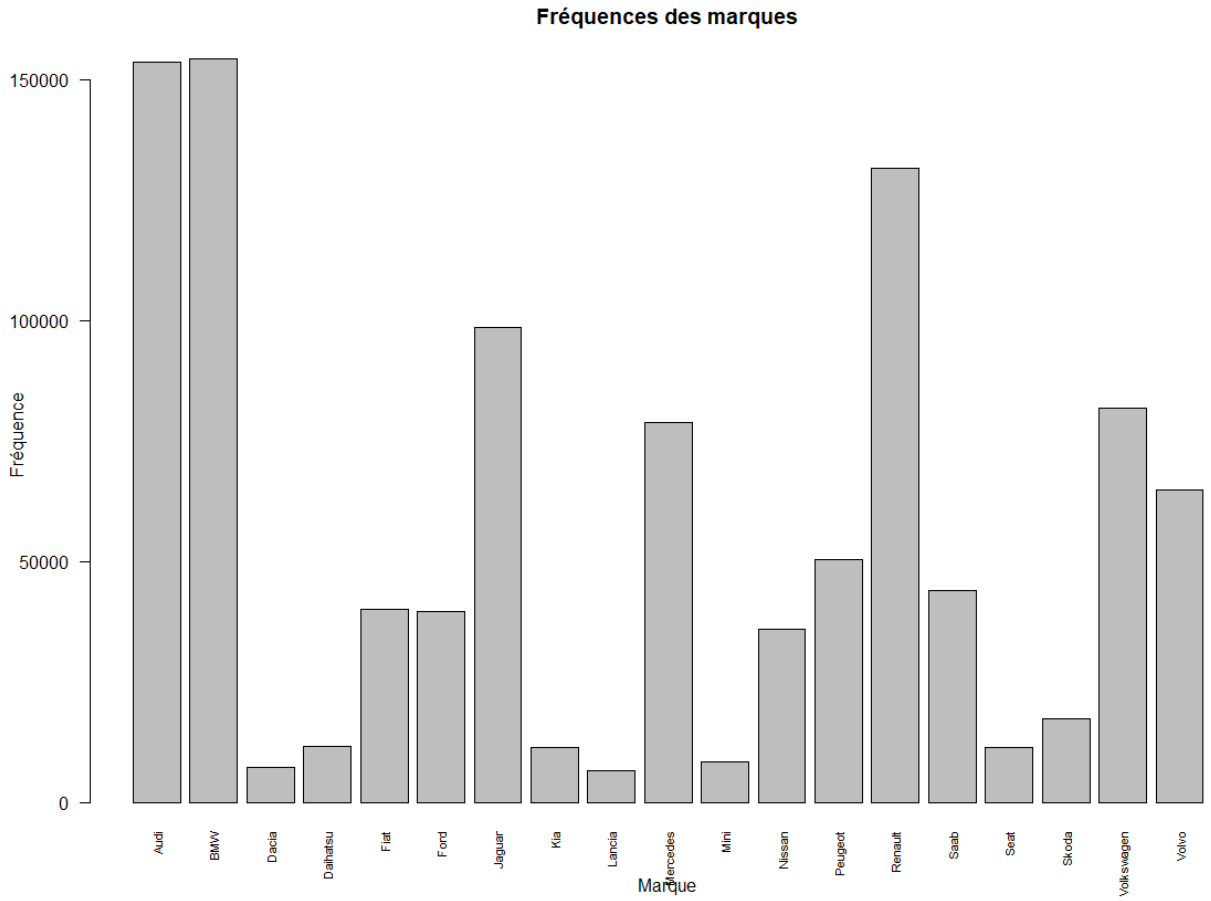
Ici la table des marques présentes dans le catalogue.

```
# Fréquences des marques
table(Catalogue$marque)
```

```
> table(Catalogue$marque)

      Audi        BMW      Dacia   Daihatsu       Fiat       Ford      Honda    Hyunda�     Jaguar        Kia
        20         20          5          5         10         10          5          5         10          5
    Lancia   Mercedes       Mini     Nissan    Peugeot    Renault       Saab       Seat      Skoda Volkswagen
        10         20         10         15         10         40         10          5          5         40
     Volvo
        10
```

On affiche maintenant la fréquence des marques dans le fichier Immatriculation

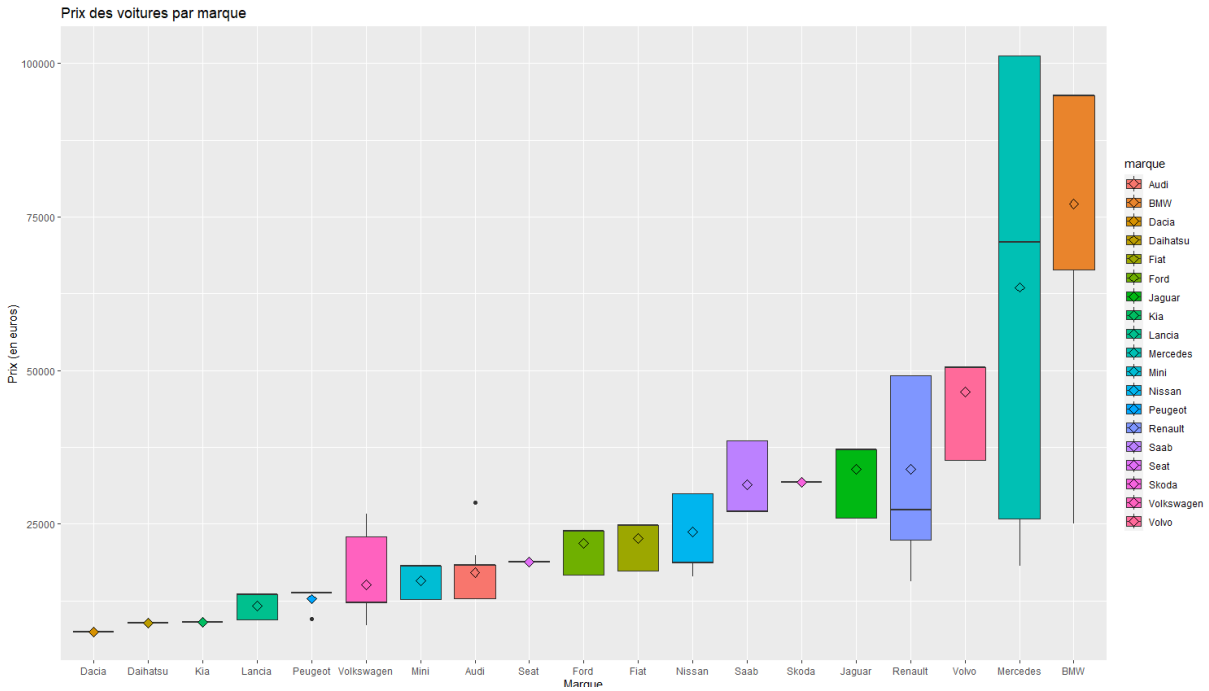
```
# Histogramme des fréquences des marques
barplot(table(Immatriculations$marque), las = 2, cex.names = 0.7, main =
"Fréquences des marques", ylab = "Fréquence", xlab = "Marque")
```



Ce graphique permet de conclure quelle marque de véhicule est le plus présent dans Immatriculation

Et enfin on affiche le prix des voitures par marque avec la moyenne en leur sein.

```
# graphique affichant le prix des voitures par marque avec la moyenne en
losange par marque
ggplot(Immatriculations, aes(x = reorder(marque, prix), y = prix, fill =
marque)) +
  geom_boxplot() +
  stat_summary(
    aes(group = marque),
    fun = "mean",
    geom = "point",
    shape = 23,
    size = 3,
    color = "black"
  ) +
  labs(title = "Prix des voitures par marque", x = "Marque", y = "Prix (en
euros)")
```



Ce graphique est assez complet et nous apporte de l'information quant à la moyenne par marque et leur fourchette de prix.

Clustering sur catalogue avec Kmeans choisi

Nous allons faire un clustering k-means en 4 (ce chiffre a été choisi suite à nombreux tests (3 était aussi bien)). On affichera ensuite les statistiques descriptives de chaque cluster.

```
# Sélection des variables pertinentes
data <- Catalogue[, c("longueur", "puissance", "prix", "nbPortes")]

# Normalisation des variables
data_norm <- scale(data)

# Clustering k-means en 4 (plusieurs tests on été effectués et 4 était selon
nous la meilleure décision)
km <- kmeans(data_norm, centers = 4)

# Sauvegarde du modèle
saveRDS(km, "model.rds")

# Attribution des clusters aux observations du catalogue
Catalogue$cluster <- km$cluster

# Calcul des statistiques descriptives pour chaque cluster
cluster_stats <- aggregate(
  Catalogue[, c("longueur", "puissance", "prix", "nbPlaces", "nbPortes")],
```

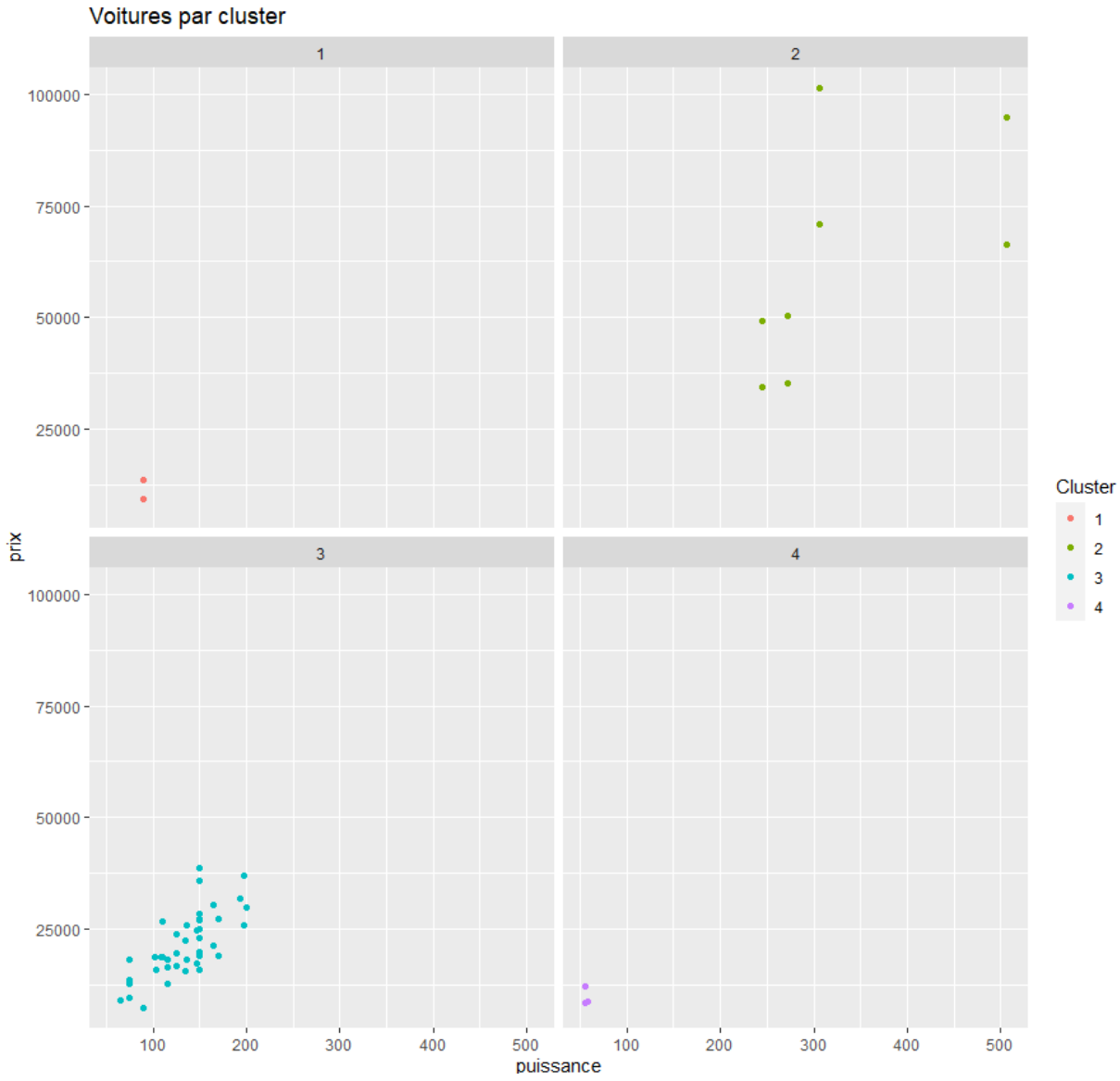
```
  by = list(cluster = Catalogue$cluster),
  FUN = function(x) c(
    mean = mean(x), sd = sd(x), min = min(x), max = max(x),
    median = median(x), q1 = quantile(x, probs = 0.25), q3 = quantile(x, probs
= 0.75)
  )
)

# Affichage des statistiques pour chaque cluster
print(cluster_stats)
```

```
> print(cluster_stats)
  cluster longueur.mean longueur.sd longueur.min longueur.max longueur.median longueur.q1.25% longueur.q3.75%
1       1      1.000000    0.000000     1.000000     1.000000        1.000000        1.000000        1.000000
2       2      4.000000    0.000000     4.000000     4.000000        4.000000        4.000000        4.000000
3       3      2.268293    0.799031     1.000000     4.000000        2.000000        2.000000        3.000000
4       4      1.000000    0.000000     1.000000     1.000000        1.000000        1.000000        1.000000
  puissance.mean puissance.sd puissance.min puissance.max puissance.median puissance.q1.25% puissance.q3.75%
1       90.00000      0.00000      90.00000      90.00000         90.00000         90.00000         90.00000
2      332.50000    104.35271     245.00000     507.00000        289.00000        265.25000        356.25000
3      134.19512     34.62135      65.00000     200.00000        136.00000        110.00000        150.00000
4       56.00000      1.46385      55.00000      58.00000         55.00000         55.00000         58.00000
   prix.mean    prix.sd   prix.min   prix.max prix.median prix.q1.25% prix.q3.75% nbPlaces.mean nbPlaces.sd
1  11475.000   2134.537   9450.000  13500.000   11475.000    9450.000   13500.000      5.0000000   0.0000000
2  62857.500  23962.341  34440.000 101300.000   58430.000   45737.500   76882.500      5.0000000   0.0000000
3  21577.439   7310.518   7500.000  38600.000   19550.000   16730.000   26630.000      5.2926829   0.7086269
4   9863.333   1715.275   8540.000  12200.000    8850.000    8540.000   12200.000      5.0000000   0.0000000
  nbPlaces.min nbPlaces.max nbPlaces.median nbPlaces.q1.25% nbPlaces.q3.75% nbPortes.mean nbPortes.sd nbPortes.min
1    5.0000000    5.0000000       5.0000000       5.0000000       5.0000000             3           0           3
2    5.0000000    5.0000000       5.0000000       5.0000000       5.0000000             5           0           5
3    5.0000000    7.0000000       5.0000000       5.0000000       5.0000000             5           0           5
4    5.0000000    5.0000000       5.0000000       5.0000000       5.0000000             3           0           3
  nbPortes.max nbPortes.median nbPortes.q1.25% nbPortes.q3.75%
1            3               3               3               3
2            5               5               5               5
3            5               5               5               5
4            3               3               3               3
```

On va afficher certaines relations dans les clusters

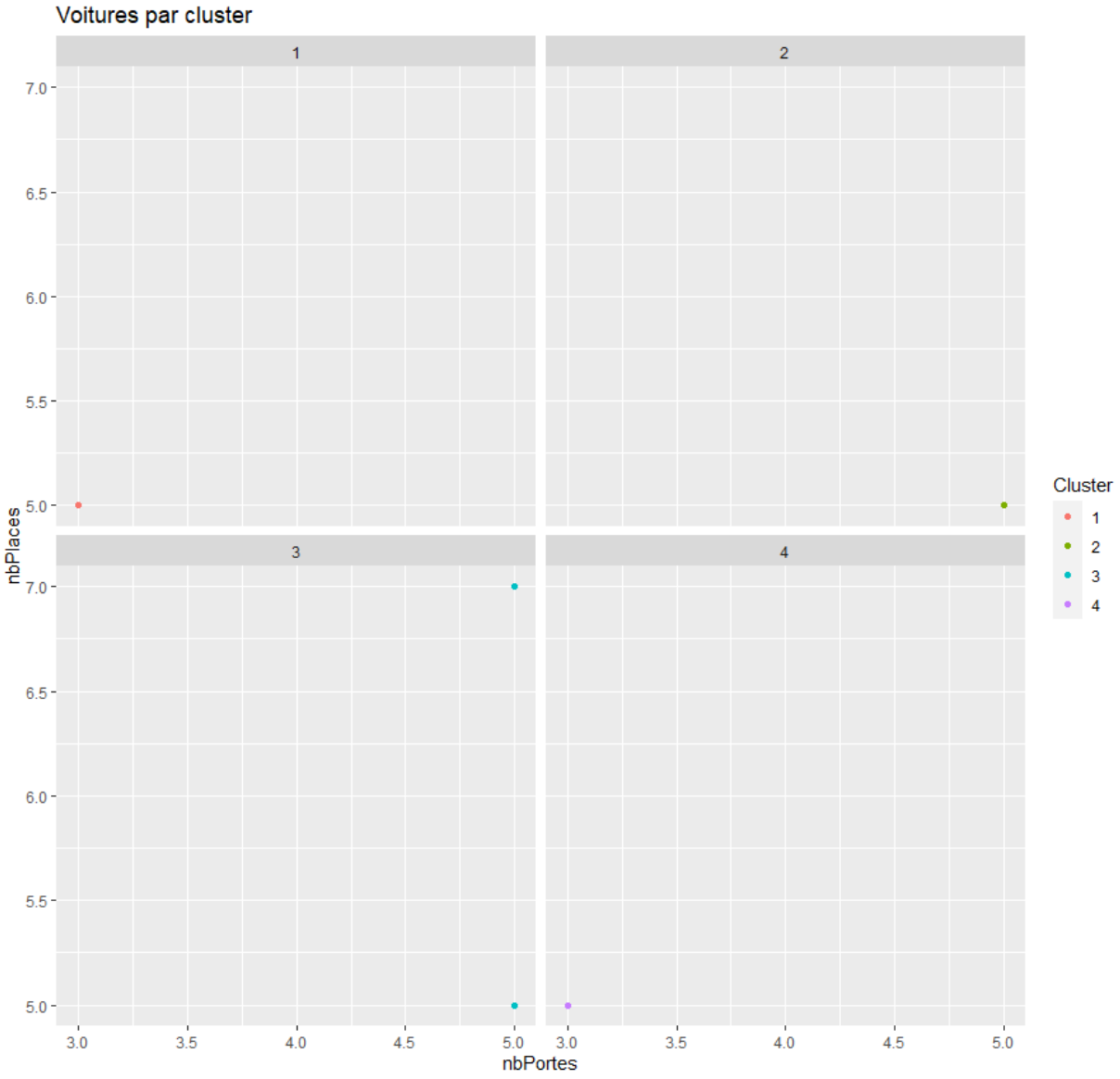
Voici le couple Puissance/Prix :

```
# Affichage graphique
# Relation dans chaque cluster : Puissance / Prix
ggplot(Catalogue, aes(x = puissance, y = prix, color = factor(cluster))) +
  geom_point() +
  facet_wrap(~factor(cluster)) +
  labs(title = "Voitures par cluster", x = "puissance", y = "prix", color =
"Cluster")
```



Voici le couple nombres de portes / nombres de place :

```
# Relation dans chaque cluster : nbPortes / nbPlaces
ggplot(Catalogue, aes(x = nbPortes, y = nbPlaces, color = factor(cluster))) +
  geom_point() +
  facet_wrap(~factor(cluster)) +
  labs(title = "Voitures par cluster", x = "nbPortes", y = "nbPlaces", color =
"Cluster")
```



Nous avions d'abord tenté cette approche avec la fonction hclust qui ne fonctionnait pas avec le predict avec Immatriculations (toutes les valeurs se mettait dans un seul cluster). Nous avons donc retenté l'expérience avec des k-means. Cependant, cela n'a pas non plus abouti (même problème qu'avec hclust).

Notre dernière solution réside donc dans le fait de faire notre clustering directement sur Immatriculations_ext.csv

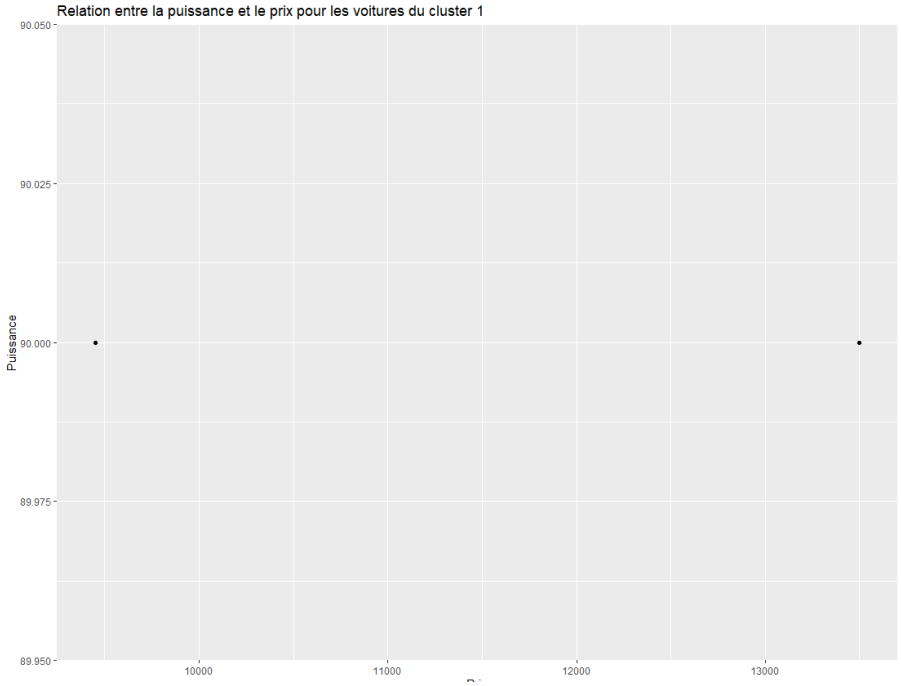
On vous montre tout de même les résultats que l'on avait obtenu pour chaque cluster.

Affichage de chaque cluster

Cluster 1 :

```r
# Sélection des voitures du cluster 1
cluster1_cars <- Catalogue[Catalogue$cluster == 1, ]

names(cluster1_cars)

cluster1_table <- as.data.frame(cluster1_cars[c("marque", "nom", "prix",
"puissance", "nbPortes", "nbPlaces")])
print(cluster1_table)

library(ggplot2)
ggplot(cluster1_table, aes(x = prix, y = puissance)) +
  geom_point() +
  labs(title = "Relation entre la puissance et le prix pour les voitures du
cluster 1",
       x = "Prix",
       y = "Puissance")

summary(cluster1_table$puissance)
summary(cluster1_table$prix)
```

Résultat :



Premier summary (par la puissance) :

```
> summary(cluster1_table$puissance)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     90      90      90      90      90      90
```

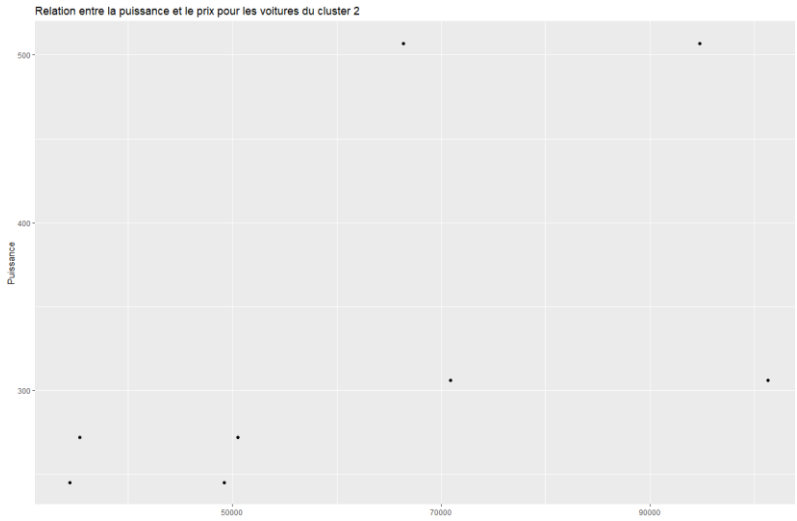
Deuxième summary (par le prix) :

```
> summary(cluster1_table$prix)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   9450    9450   11475   11475   13500   13500
```

Cluster 2 :

```r
# Sélection des voitures du cluster 2
cluster2_cars <- Catalogue[Catalogue$cluster == 2, ]

# Création de la table
cluster2_table <- as.data.frame(cluster2_cars[c("marque", "nom", "prix",
"puissance", "nbPortes", "nbPlaces")])

# Affichage de la table
print(cluster2_table)

# Création du graphique
ggplot(cluster2_table, aes(x = prix, y = puissance)) +
  geom_point() +
  labs(title = "Relation entre la puissance et le prix pour les voitures du
cluster 2",
       x = "Prix",
       y = "Puissance")

# Résumé des variables
summary(cluster2_table$puissance)
summary(cluster2_table$prix)
```

Résultat :



Premier summary (par la puissance) :

```
> summary(cluster2_table$puissance)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  245.0   265.2   289.0   332.5   356.2   507.0
```

Deuxième summary (par le prix) :

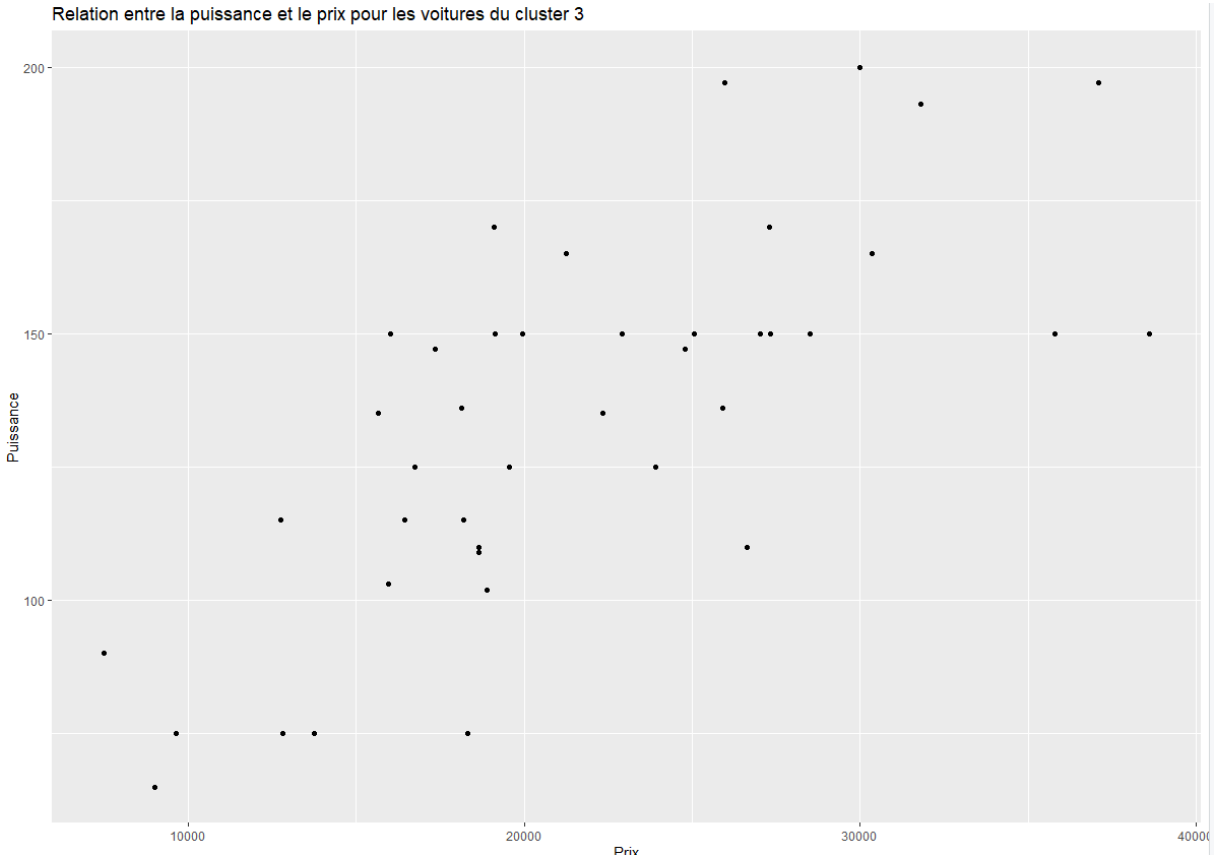
```
> summary(cluster2_table$prix)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  34440   45738   58430   62858   76883  101300
```

Cluster 3 :

```r
# Sélection des voitures du cluster 3
cluster3_cars <- Catalogue[Catalogue$cluster == 3, ]

# Création de la table
cluster3_table <- as.data.frame(cluster3_cars[c("marque", "nom", "prix",
"puissance", "nbPortes", "nbPlaces")])

# Affichage de la table
print(cluster3_table)

# Création du graphique
ggplot(cluster3_table, aes(x = prix, y = puissance)) +
  geom_point() +
  labs(title = "Relation entre la puissance et le prix pour les voitures du
cluster 3",
       x = "Prix",
       y = "Puissance")

# Résumé des variables
summary(cluster3_table$puissance)
```

```
summary(cluster3_table$prix)
```

Résultat :



Premier summary (par la puissance) :

```
> summary(cluster3_table$puissance)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   65.0   110.0   136.0   134.2   150.0   200.0
```

Deuxième summary (par le prix) :

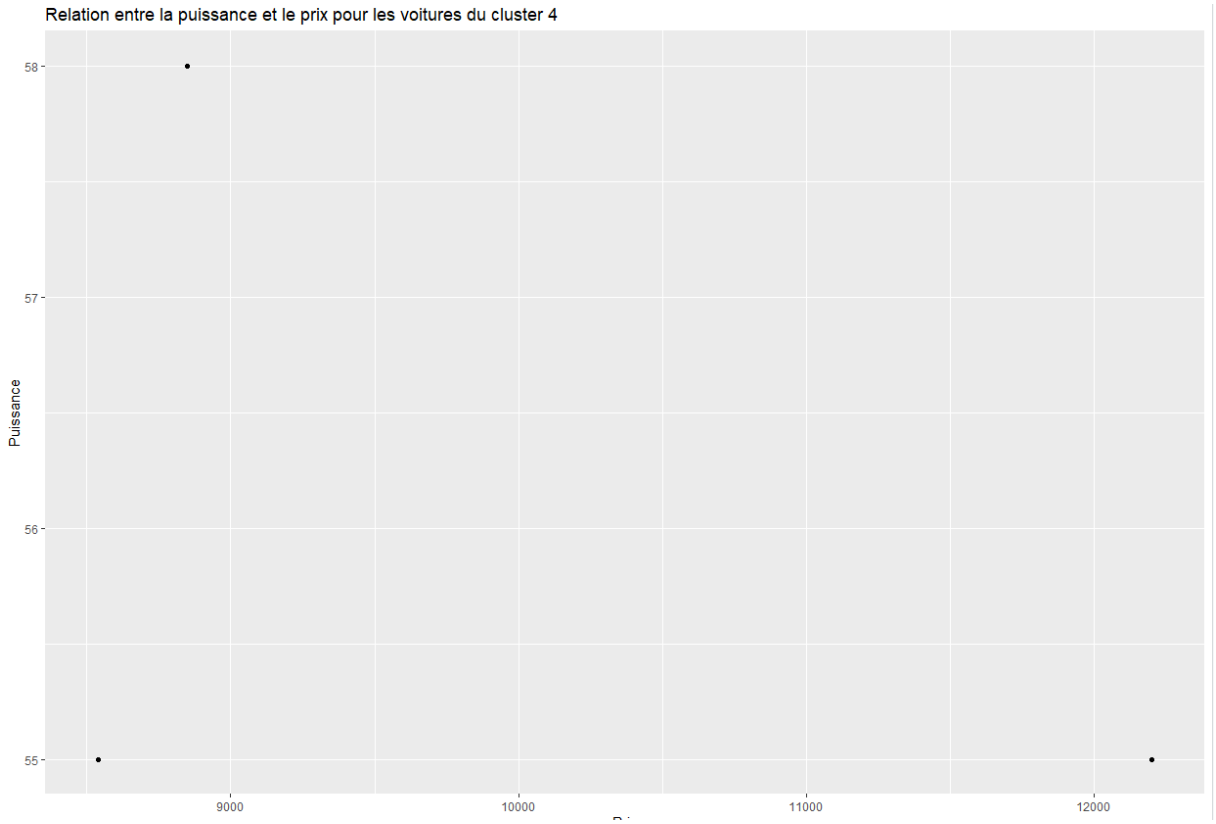
```
> summary(cluster3_table$prix)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   7500   16730   19550   21577   26630   38600
```

Cluster 4 :

```
# Sélection des voitures du cluster 4
cluster4_cars <- Catalogue[Catalogue$cluster == 4, ]

# Création de la table
```

```r
cluster4_table <- as.data.frame(cluster4_cars[c("marque", "nom", "prix",
"puissance", "nbPortes", "nbPlaces")])

# Affichage de la table
print(cluster4_table)

# Création du graphique
ggplot(cluster4_table, aes(x = prix, y = puissance)) +
  geom_point() +
  labs(title = "Relation entre la puissance et le prix pour les voitures du
cluster 4",
       x = "Prix",
       y = "Puissance")

# Résumé des variables
summary(cluster4_table$puissance)
summary(cluster4_table$prix)
```

Résultat :



Premier summary (par la puissance) :

```
> summary(cluster4_table$puissance)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     55      55      55      56      58      58
```

Deuxième summary (par le prix) :

```
> summary(cluster4_table$prix)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   8540    8540    8850    9863   12200   12200
```

On va maintenant créer une table Catalogue avec les classes

```
> head(CatalogueClasses)
  marque    nom puissance longueur nbPlaces nbPortes couleur occasion  prix cluster Classe
1  Volvo S80 T6       272        4        5        5   blanc    false 50500       2     NA
2  Volvo S80 T6       272        4        5        5    noir    false 50500       2     NA
3  Volvo S80 T6       272        4        5        5   rouge    false 50500       2     NA
4  Volvo S80 T6       272        4        5        5    gris     true 35350       2     NA
5  Volvo S80 T6       272        4        5        5    bleu     true 35350       2     NA
6  Volvo S80 T6       272        4        5        5    gris    false 50500       2     NA
```

```
CatalogueClasses <- Catalogue
CatalogueClasses$Classe <- NA

head(CatalogueClasses)
```

On attribue les classes en fonction des clusters, cependant comme ces derniers changent toujours, il est impossible pour nous de les nommer.

Par exemple, lors du premier lancement le cluster 1 peut représenter les voitures de luxes, puis les citadines après le deuxième lancement.

Nous décidons donc de ne pas attribuer de nom et plutôt de le faire à la fin afin d'analyser Marketing_cluster :

```
CatalogueClasses$Classe[Catalogue$cluster == 1] <- "Routière/Familliale"
CatalogueClasses$Classe[Catalogue$cluster == 2] <- "Citadine Plus"
CatalogueClasses$Classe[Catalogue$cluster == 3] <- "Citadine"
CatalogueClasses$Classe[Catalogue$cluster == 4] <- "Luxe"
```

Création de la classe :

```
table(CatalogueClasses$Classe)
```

Application des classes / clusters à Immatriculations.csv

```
Immatriculations <- data.frame(immatriculation = character(),
                               marque = character(),
                               nom = character(),
                               puissance = character(),
                               longueur = character(),
                               nbPlaces = numeric(),
                               nbPortes = numeric(),
                               couleur = character(),
                               occasion = character(),
                               prix = numeric(),
                               stringsAsFactors = TRUE)
```

On importe les données et on les ajoute à la table.

```
# Importer le fichier CSV
ImmData <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/Immatriculations_ext.csv",
header = FALSE, sep = ",", dec = ".",stringsAsFactors = TRUE)

# Ajouter les données à la table
colnames(ImmData) <- c("immatriculation", "marque", "nom", "puissance",
"longueur", "nbPlaces", "nbPortes", "couleur", "occasion", "prix")

Immatriculations <- rbind(Immatriculations, ImmData)
```

On affiche ensuite quelques lignes :

```
# Afficher les premières lignes de la table
head(Immatriculations,20)
```

```
> head(Immatriculations,20)
   immatriculation     marque          nom puissance   longueur nbPlaces nbPortes couleur occasion  prix
1         3176TS67    Renault   Laguna2.0T       170     longue        5        5   blanc    false 27300
2         3721QS49      Volvo         S80T6       272 treslongue        5        5    noir    false 50500
3         9099UV26 Volkswagen   Golf2.0FSI       150    moyenne        5        5    gris     true 16029
4         3563LA55    Peugeot      10071.4        75     courte        5        5   blanc     true  9625
5         6963AX34       Audi        A21.4        75     courte        5        5    gris    false 18310
6         5592HQ89      Skoda  Superb2.8V6       193 treslongue        5        5    bleu    false 31790
7          674CE26    Renault Megane2.016V       135    moyenne        5        5    gris    false 22350
8         1756PR31   Mercedes         A200       136    moyenne        5        5    noir     true 18130
9         6705GX50        BMW         120i       150    moyenne        5        5    noir     true 25060
10        4487DR75       Saab      9.31.8T       150     longue        5        5    gris     true 27020
11        7080NW34     Jaguar  X-Type2.5V6       197     longue        5        5   blanc     true 25970
12        9626HF36       Audi        A21.4        75     courte        5        5   rouge    false 18310
13        2401PA98      Volvo         S80T6       272 treslongue        5        5    bleu     true 35350
14         826YF89    Renault   Laguna2.0T       170     longue        5        5   rouge    false 27300
15        8216GR23      Skoda  Superb2.8V6       193 treslongue        5        5    bleu    false 31790
16        8076YM23     Jaguar  X-Type2.5V6       197     longue        5        5    noir    false 37100
17        9277JN49        BMW           M5       507 treslongue        5        5   rouge     true 66360
18        4231HC31       Audi        A21.4        75     courte        5        5   rouge    false 18310
19        2319IQ28       Ford    Mondeo1.8       125     longue        5        5    gris    false 23900
20         148RS75        BMW           M5       507 treslongue        5        5   blanc     true 66360
```

```
summary(Immatriculations)
```

```
> summary(Immatriculations)
 immatriculation          marque                nom          puissance        longueur        nbPlaces    nbPortes
 1007LX38:      2   BMW       :154301   A21.4        :149337   Min.   : 55.0   courte    :289021   Min.   :5   Min.   :3.000
 1018QC21:      2   Audi      :153630   M5           :134439   1st Qu.: 75.0   longue    :285041   1st Qu.:5   1st Qu.:5.000
 102NB76 :      2   Renault   :131735   X-Type2.5V6  : 98592   Median :150.0   moyenne   :121305   Median :5   Median :5.000
 1033TS14:      2   Jaguar    : 98592   S80T6        : 64803   Mean   :198.9   treslongue:353208   Mean   :5   Mean   :4.868
 1041MB78:      2   Volkswagen: 81885   VelSatis3.5V6: 64518   3rd Qu.:245.0                       3rd Qu.:5   3rd Qu.:5.000
 1100CO10:      2   Mercedes  : 78769   S500         : 54475   Max.   :507.0                       Max.   :5   Max.   :5.000
 (Other) :1048563   (Other)   :349663   (Other)      :482411
  couleur        occasion          prix
 blanc:209075   false:721044   Min.   :  7500
 bleu :209932   true :327531   1st Qu.: 18310
 gris :210226                  Median : 25970
 noir :209729                  Mean   : 35767
 rouge:209613                  3rd Qu.: 49200
                               Max.   :101300
```

On pas longueur en numérique afin de pouvoir utiliser la colonne lors du clustering qui suivra :

```
# conversion de la colonne "longueur" en format numérique
Immatriculations$longueur <- as.numeric(Immatriculations$longueur)

# Afficher les premières lignes de la table
head(Immatriculations,20)
```

```
> head(Immatriculations,20)
   immatriculation     marque              nom puissance longueur nbPlaces nbPortes couleur occasion  prix
1            0IL59   Mercedes             A200       136        3        5        5   rouge     true 18130
2            0NP92      Volvo           S80 T6       272        4        5        5   rouge    false 50500
3            0OZ65       Ford       Mondeo 1.8       125        2        5        5   blanc    false 23900
4            0TS49        BMW               M5       507        4        5        5    bleu    false 94800
5            0TY90 Volkswagen     Golf 2.0 FSI       150        3        5        5   blanc    false 22900
6            0UP83    Renault Vel Satis 3.5 V6       245        4        5        5   blanc    false 49200
7            0VG85      Volvo           S80 T6       272        4        5        5   rouge    false 50500
8            0WB68     Jaguar    X-Type 2.5 V6       197        2        5        5    noir     true 25970
9            0YW11       Saab         9.3 1.8T       150        2        5        5    gris    false 38600
10        1000AD49       Audi            A2 1.4        75        1        5        5   blanc    false 18310
11        1000BY42 Volkswagen      Polo 1.2 6V        55        1        5        3    noir     true  8540
12        1000GU46        BMW               M5       507        4        5        5   blanc    false 94800
13        1000JL76       Audi            A2 1.4        75        1        5        5   rouge    false 18310
14        1000LJ53      Volvo           S80 T6       272        4        5        5   blanc    false 50500
15        1000NJ92    Renault Vel Satis 3.5 V6       245        4        5        5    noir    false 49200
16        1000OU97       Audi            A2 1.4        75        1        5        5    noir    false 18310
17        1000WJ93       Saab         9.3 1.8T       150        2        5        5   rouge     true 27020
18        1001AM12       Audi            A2 1.4        75        1        5        5   rouge    false 18310
19        1001FE41    Peugeot         1007 1.4        75        1        5        5    gris    false 13750
20        1001FI24    Renault Vel Satis 3.5 V6       245        4        5        5    gris     true 34440
```

Comme expliqué auparavant, nous allons maintenant appliquer le clustering directement sur Immatriculations.

Voici cependant la partie predict que nous avions tenté :

```
# # CETTE PARTIE visait à utiliser le clustering fait sur catalogue mais après de nombreuses
# # tentatives, cela ne fonctionne pas
#
# # Nous allons donc faire LE CLUSTERING DIRECTEMENT SUR IMMATRICULATIONS
#
# # On récupère uniquement les variables utiles
```

```
# new_data <- Immatriculations[, c("longueur", "puissance", "prix",
"nbPortes")]
#
# # Récupérer le modèle
# model <- readRDS("model.rds")
#
# library(rattle)
#
# # Prédire les classes pour immatriculations en utilisant le modèle
# # Ajouter la colonne de clusters prédits à la table d'immatriculations
# Immatriculations$cluster <- predict(model, new_data)
#
# # Création d'une nouvelle colonne "Classe" dans Catalogue
# ImmatriculationsClasses <- Immatriculations
# ImmatriculationsClasses$Classe <- NA
#
# # Afficher les premières lignes de la table
# head(ImmatriculationsClasses,50)
#
# # Affichage du nombre de voitures par classe
# table(ImmatriculationsClasses$Classe)
#
# summary(ImmatriculationsClasses)
#
# summary(ImmatriculationsClasses[ImmatriculationsClasses$cluster == 4,])
```

Clustering sur immatriculation

1ère étape, on vérifie qu'il n'y a aucune valeur manquante

```
# On refait donc un clustering sur IMMATRICULATIONS

new_data_kmeans <- Immatriculations[, c("longueur", "puissance", "prix",
"nbPortes")]

# Vérifier s'il y a des valeurs manquantes dans la colonne "puissance" de la
table "Immatriculations"
verifNA <- function(data){
  cols_na <- names(which(colSums(is.na(data)) > 0))
  if(length(cols_na) == 0){
    message("Il n'y a pas de valeurs manquantes dans la table.")
  } else {
    message("La table contient des valeurs manquantes dans les colonnes
suivantes : ", paste(cols_na, collapse = ", "))
  }
}
verifNA(Immatriculations)
```

```
> new_data_kmeans <- Immatriculations[, c("longueur", "puissance", "prix", "nbPortes")]
> verifNA <- function(data){
+     cols_na <- names(which(colSums(is.na(data)) > 0))
+     if(length(cols_na) == 0){
+         message("Il n'y a pas de valeurs manquantes dans la table.")
+     } else {
+         message("La table contient des valeurs manquantes dans les colonnes suivantes : ", paste(cols_na, collapse = ", "))
+     }
+ }
> verifNA(Immatriculations)
Il n'y a pas de valeurs manquantes dans la table.
```

```
summary(new_data_kmeans)
```

```
> summary(new_data_kmeans)
       longueur        puissance          prix          nbPortes
 courte    :289021   Min.   : 55.0   Min.   :  7500   Min.   :3.000
 longue    :285041   1st Qu.: 75.0   1st Qu.: 18310   1st Qu.:5.000
 moyenne   :121305   Median :150.0   Median : 25970   Median :5.000
 treslongue:353208   Mean   :198.9   Mean   : 35767   Mean   :4.868
                     3rd Qu.:245.0   3rd Qu.: 49200   3rd Qu.:5.000
                     Max.   :507.0   Max.   :101300   Max.   :5.000
```

```
# Normalisation des variables
new_data_norm <- scale(new_data_kmeans)

summary(new_data_norm)
```

```
> summary(new_data_norm)
     longueur         puissance           prix          nbPortes
 Min.   :-1.246   Min.   :-1.0493   Min.   :-1.0965   Min.   :-3.7614
 1st Qu.:-1.246   1st Qu.:-0.9035   1st Qu.:-0.6772   1st Qu.: 0.2659
 Median :-0.423   Median :-0.3567   Median :-0.3800   Median : 0.2659
 Mean   : 0.000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 1.224   3rd Qu.: 0.3359   3rd Qu.: 0.5211   3rd Qu.: 0.2659
 Max.   : 1.224   Max.   : 2.2460   Max.   : 2.5421   Max.   : 0.2659
```

```
# Clustering k-means
kmImm <- kmeans(new_data_norm, centers = 4)

# Attribution des clusters aux observations du catalogue
Immatriculations$cluster <- kmImm$cluster

# Calcul des statistiques descriptives pour chaque cluster
cluster_stats_Imm <- aggregate(
  Immatriculations[, c("longueur", "puissance", "prix", "nbPlaces",
"nbPortes")],
  by = list(cluster = Immatriculations$cluster),
  FUN = function(x) c(
    mean = mean(x), sd = sd(x), min = min(x), max = max(x),
    median = median(x), q1 = quantile(x, probs = 0.25), q3 = quantile(x, probs
= 0.75)
  )
```

```
)

# Affichage du nombre de voiture dans chaque cluster
table(Immatriculations$cluster)
```

```
> table(Immatriculations$cluster)

     1      2      3      4
289021 406346 164294 188914
```

```
# Affichage des statistiques pour chaque cluster
aggregate(Immatriculations[, -c(1:2)], by=list(Immatriculations$cluster),
FUN=mean)
# Afficher les premières lignes de la table
head(Immatriculations,20)
```

```
> head(Immatriculations,20)
   immatriculation     marque          nom puissance longueur nbPlaces nbPortes couleur occasion  prix cluster
1         3176TS67    Renault   Laguna2.0T       170        2        5        5   blanc    false 27300       2
2         3721QS49      Volvo         S80T6       272        4        5        5    noir    false 50500       3
3         9099UV26 Volkswagen   Golf2.0FSI       150        3        5        5    gris     true 16029       2
4         3563LA55    Peugeot      10071.4        75        1        5        5   blanc     true  9625       1
5         6963AX34       Audi         A21.4        75        1        5        5    gris    false 18310       1
6         5592HQ89      Skoda   Superb2.8V6       193        4        5        5    bleu    false 31790       3
7          674CE26    Renault Megane2.016V       135        3        5        5    gris    false 22350       2
8         1756PR31   Mercedes          A200       136        3        5        5    noir     true 18130       2
9         6705GX50        BMW          120i       150        3        5        5    noir     true 25060       2
10        4487DR75       Saab       9.31.8T       150        2        5        5    gris     true 27020       2
11        7080NW34     Jaguar  X-Type2.5V6       197        2        5        5   blanc     true 25970       2
12        9626HF36       Audi         A21.4        75        1        5        5   rouge    false 18310       1
13        2401PA98      Volvo         S80T6       272        4        5        5    bleu     true 35350       3
14         826YF89    Renault   Laguna2.0T       170        2        5        5   rouge    false 27300       2
15        8216GR23      Skoda   Superb2.8V6       193        4        5        5    bleu    false 31790       3
16        8076YM23     Jaguar  X-Type2.5V6       197        2        5        5    noir    false 37100       2
17        9277JN49        BMW           M5       507        4        5        5   rouge     true 66360       4
18        4231HC31       Audi         A21.4        75        1        5        5   rouge    false 18310       1
19        2319IQ28       Ford     Mondeo1.8       125        2        5        5    gris    false 23900       2
20         148RS75        BMW           M5       507        4        5        5   blanc     true 66360       4
```

On charge les csv des clients afin de lier ceux-ci à leur voiture dans immatriculations

```
# Charger les fichiers CSV
clients_12_ext <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/clients_12_ext.csv", header
= FALSE)
clients_4_ext <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/clients_4_ext.csv", header =
FALSE)

# Nommer les colonnes
colnames(clients_12_ext) <- c("id", "age", "sexe", "taux",
"situationFamiliale", "nbEnfantsAcharge", "SecondeVoiture", "immatriculation")
colnames(clients_4_ext) <- c("id", "age", "sexe", "taux",
"situationFamiliale", "nbEnfantsAcharge", "SecondeVoiture", "immatriculation")
```

```
# Combinaison des deux tables
clients_combined <- rbind(clients_12_ext, clients_4_ext)

table(clients_combined$situationFamiliale)
```

```
> table(clients_combined$situationFamiliale)

Celibataire   En Couple
      68973      127910
```

```
table(clients_combined$sexe)
```

```
> table(clients_combined$sexe)

     f      m
 59218 137665
```

```
summary(clients_combined)
```

```
> summary(clients_combined)
      id                 age             sexe                taux          situationFamiliale nbEnfantsAcharge SecondeVoiture
 Length:196883      Min.   :18.00   Length:196883      Min.   : 150.0   Length:196883      Min.   :0.00     Length:196883
 Class :character   1st Qu.:28.00   Class :character   1st Qu.: 421.0   Class :character   1st Qu.:0.00     Class :character
 Mode  :character   Median :42.00   Mode  :character   Median : 522.0   Mode  :character   Median :1.00     Mode  :character
                    Mean   :43.74                      Mean   : 608.9                      Mean   :1.25
                    3rd Qu.:57.00                      3rd Qu.: 828.0                      3rd Qu.:2.00
                    Max.   :84.00                      Max.   :1399.0                      Max.   :4.00

 immatriculation
 Length:196883
 Class :character
 Mode  :character
```

On joint les deux tables (clients_combined comprenant clients_4 et clients_12 avec immatriculations) avec leur immatriculation.

```
library(dplyr)
ImmatClients <- inner_join(clients_combined, Immatriculations, by =
"immatriculation", multiple = "all")
head(ImmatClients)
```

```
> head(ImmatClients)
                       id age sexe taux situationFamiliale nbEnfantsAcharge SecondeVoiture immatriculation marque         nom
1 6409f7e9802b1871fd0cbbf1  25    m  432          En Couple                4          false        2919FW32   Seat   Toledo1.6
2 6409f7ef802b1871fd0cc713  28    m  438          En Couple                2          false        6170OR10  Dacia Logan1.6MPI
3 6409f7f0802b1871fd0cc875  43    m  426          En Couple                4          false        8478EG82  Skoda Superb2.8V6
4 6409f7f1802b1871fd0ccb67  41    f 1375          En Couple                2          false         677PS11    BMW          M5
5 6409f7f4802b1871fd0cd09b  55    m 1094        Celibataire                0          false        6762YS30 Jaguar X-Type2.5V6
6 6409f7f8802b1871fd0cd7ad  70    m  533          En Couple                0          false        2025JY26   Fiat    Croma2.2
  puissance longueur nbPlaces nbPortes couleur occasion  prix cluster
1       102        2        5        5    noir    false 18880       2
2        90        3        5        5    bleu    false  7500       2
3       193        4        5        5    noir    false 31790       3
4       507        4        5        5   blanc     true 66360       4
5       197        2        5        5    noir     true 25970       2
6       147        2        5        5    noir    false 24780       2
```

```
# Affichage du nombre de voiture dans chaque cluster
table(ImmatClients$cluster)
```

```
> table(ImmatClients$cluster)

    1     2     3     4
27120 38179 15603 17852
```

La graine ci-dessous permet de conserver les mêmes résultats.

```
# Fixer la graine aléatoire
set.seed(123)

# Créer un vecteur d'indices aléatoires pour l'échantillonnage
indices <- sample(1:nrow(ImmatClients), size = nrow(ImmatClients), replace =
FALSE)

indices
# Calculer le nombre de lignes pour l'ensemble d'apprentissage
n_train <- round(nrow(ImmatClients) * 0.7)

n_train
```

```
> n_train
[1] 69128
```

```
# Sélectionner les lignes pour l'ensemble d'apprentissage et de test
ImmatClients_EA <- ImmatClients[indices[1:n_train], ]

ImmatClients_ET <- ImmatClients[indices[(n_train+1):nrow(ImmatClients)], ]
# Supprimer les colonnes id et immatriculation
ImmatClients_EA <- subset(ImmatClients_EA, select = -c(id, immatriculation))
ImmatClients_ET <- subset(ImmatClients_ET, select = -c(id, immatriculation))

head(ImmatClients_EA)
```

```
> head(ImmatClients_EA)
      age sexe taux situationFamiliale nbEnfantsACharge SecondeVoiture   marque        nom puissance longueur nbPlaces nbPortes
51663  66    m  224          En Couple                2          false     Fiat    Croma2.2       147        2         5        5
57870  65    f  730        Celibataire                0          false Mercedes        A200       136        3         5        5
2986   30    m 1089          En Couple                3          false      BMW          M5       507        4         5        5
29925  28    m 1214          En Couple                2          false   Jaguar X-Type2.5V6       197        2         5        5
95246  35    m  810          En Couple                1          false   Jaguar X-Type2.5V6       197        2         5        5
68293  67    f  537          En Couple                0          false     Saab     9.31.8T       150        2         5        5
      couleur occasion  prix cluster
51663    bleu     true 17346       2
57870   rouge    false 25900       2
2986     bleu    false 94800       4
29925    bleu    false 37100       2
95246    gris    false 37100       2
68293   blanc     true 27020       2
```

```
head(ImmatClients_ET)
```

```
> head(ImmatClients_ET)
      age sexe taux situationFamiliale nbEnfantsAcharge SecondeVoiture   marque           nom puissance longueur nbPlaces nbPortes
36706  33    m  518        Celibataire                0          false  Renault  Megane2.016V       135        3         5         5
74544  33    m  432        Celibataire                0          false     Audi         A21.4        75        1         5         5
84369  52    m  464          En Couple                1          false  Renault VelSatis3.5V6       245        4         5         5
21919  45    m  485          En Couple                4           true      BMW            M5       507        4         5         5
969    63    m  456        Celibataire                0          false Mercedes          A200       136        3         5         5
21358  46    m  432        Celibataire                1          false     Ford     Mondeo1.8       125        2         5         5
      couleur occasion  prix cluster
36706    gris    false 22350       2
74544    noir     true 12817       1
84369    gris    false 49200       3
21919   rouge     true 66360       4
969      bleu     true 18130       2
21358   rouge    false 23900       2
```

Arbres de décisions

```
library(rpart)

library(C50)

library(rpart.plot)

ImmatClients_EA$cluster <- factor(ImmatClients_EA$cluster)

class(ImmatClients_EA$cluster)

summary(ImmatClients_EA)
```

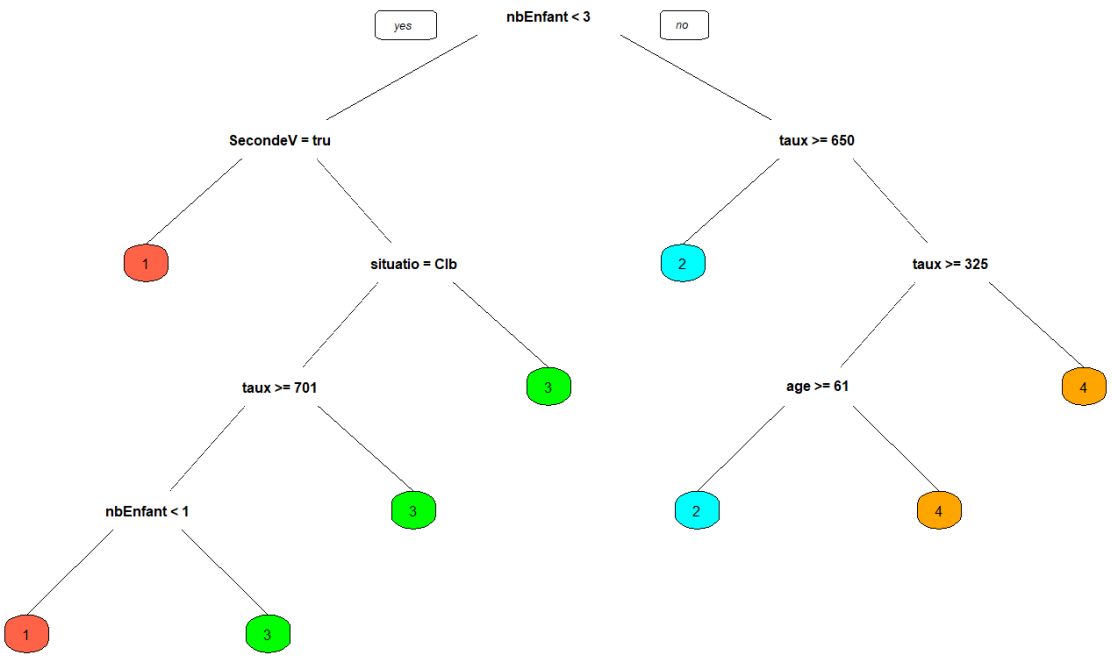
```
> summary(ImmatClients_EA)
      age            sexe                taux       situationFamiliale nbEnfantsAcharge SecondeVoiture           marque
 Min.   :18.00   Length:69128       Min.   : 150.0   Length:69128       Min.   :0.000    Length:69128       BMW       :10288
 1st Qu.:28.00   Class :character   1st Qu.: 421.0   Class :character   1st Qu.:0.000    Class :character   Audi      :10102
 Median :42.00   Mode  :character   Median : 522.0   Mode  :character   Median :1.000    Mode  :character   Renault   : 8673
 Mean   :43.75                      Mean   : 609.3                      Mean   :1.252                       Jaguar    : 6544
 3rd Qu.:56.00                      3rd Qu.: 828.0                      3rd Qu.:2.000                       Volkswagen: 5264
 Max.   :84.00                      Max.   :1399.0                      Max.   :4.000                       Mercedes  : 5256
                                                                                                            (Other)   :23001
           nom          puissance        longueur        nbPlaces     nbPortes      couleur       occasion          prix
 A21.4        : 9829   Min.   : 55.0   Min.   :1.000   Min.   :5   Min.   :3.000   blanc:13866   false:47492   Min.   :  7500
 M5           : 8959   1st Qu.: 75.0   1st Qu.:1.000   1st Qu.:5   1st Qu.:5.000   bleu :13882   true :21636   1st Qu.: 18310
 X-Type2.5V6  : 6544   Median :150.0   Median :2.000   Median :5   Median :5.000   gris :13666                 Median : 25970
 VelSatis3.5V6: 4297   Mean   :199.7   Mean   :2.521   Mean   :5   Mean   :4.872   noir :13965                 Mean   : 35902
 S80T6        : 4241   3rd Qu.:245.0   3rd Qu.:4.000   3rd Qu.:5   3rd Qu.:5.000   rouge:13749                 3rd Qu.: 49200
 S500         : 3621   Max.   :507.0   Max.   :4.000   Max.   :5   Max.   :5.000                               Max.   :101300
 (Other)      :31637
 cluster
 1:18898
 2:26790
 3:10860
 4:12580
```

```
colnames(ImmatClients_EA)
```

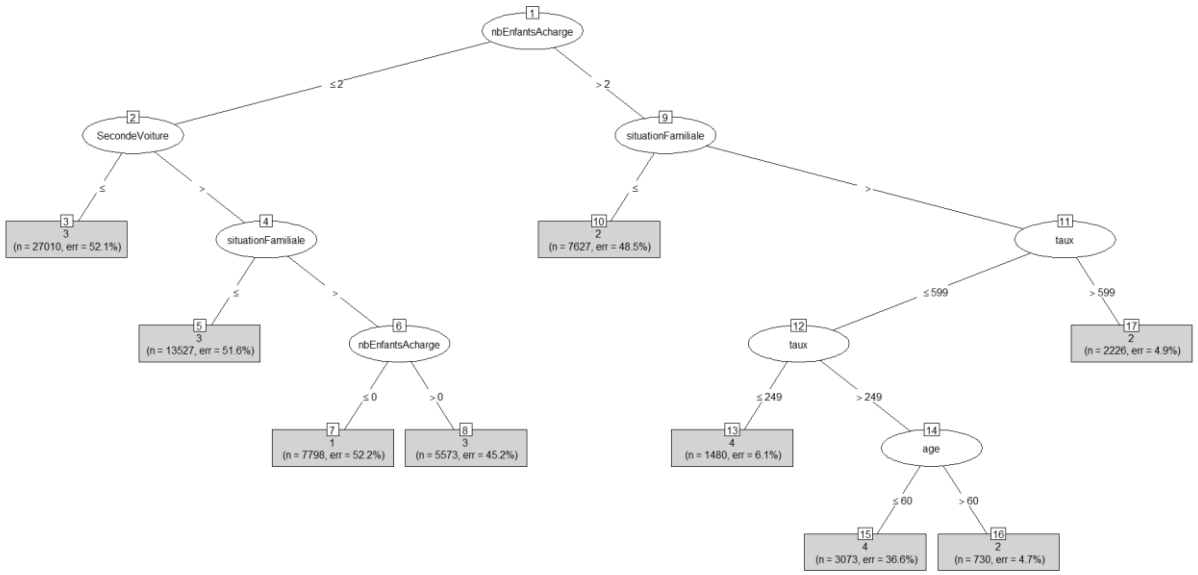
```
> colnames(ImmatClients_EA)
 [1] "age"              "sexe"             "taux"             "situationFamiliale" "nbEnfantsAcharge"   "SecondeVoiture"
 [7] "marque"           "nom"              "puissance"        "longueur"           "nbPlaces"           "nbPortes"
[13] "couleur"          "occasion"         "prix"             "cluster"
```

On va construire ici deux arbres de décision afin de définir le type de véhicule le plus adapté à la situation du client.

```
# Construction de l'arbre de decision
tree1 <-
rpart(cluster~age+sexe+taux+situationFamiliale+nbEnfantsAcharge+SecondeVoiture
, ImmatClients_EA)
tree2 <-
C5.0(cluster~age+sexe+taux+situationFamiliale+nbEnfantsAcharge+SecondeVoiture,
ImmatClients_EA)

prp(tree1, extra=0, box.col=c("tomato",
"cyan","green","orange")[tree1$frame$yval])
```



```
plot(tree2, type="simple")
```




```
test_tree1 <- predict(tree1, ImmatClients_ET, type="class")
print(test_tree1)
table(test_tree1)
```

```
> table(test_tree1)
test_tree1
    1     2     3     4
 5426  2552 17744  3868
```

```
test_tree2 <- predict(tree2, ImmatClients_ET, type="class")
print(test_tree2)
table(test_tree2)
```

```
> table(test_tree2)
test_tree2
    1     2     3     4
11545  2360 12099  3586
```

On passe maintenant aux matrices de confusions :

```
mc_tree1 <- table(ImmatClients_ET$cluster, test_tree1)
mc_tree1
```

On conclue avec les Vraies Positifs et Faux Négatifs qui doivent être le plus haut possible. On a ici une bonne situation.

```
> mc_tree1
   test_tree1
        1     2     3     4
  1  5096     5  3032     2
  2     0  2409  2065   798
  3   330   137 10828   183
  4     0     1  1819  2885
> |
```

```
mc_tree2 <- table(ImmatClients_ET$cluster, test_tree2)
mc_tree2
```

```
> mc_tree2
   test_tree2
        1     2     3     4
  1  5093  3124     2     3
  2   345 11036     6     2
  3     2  1947  2793     1
  4     0  2100   773  2399
```

On va maintenant créer une random forest, le but étant de prendre la moyenne des arbres sur un nombre donné avec un nombre de paramètres donnés

```
library(randomForest)

# Approche 1 : Random Forest
# Tester plusieurs paramétrages de Random Forest
n_trees <- c(50, 100, 200)
mtry <- c(2, 4, 6)
predictors <- c("age", "sexe", "taux", "situationFamiliale",
"nbEnfantsAcharge", "SecondeVoiture","cluster")
results_rf <- list()
for (n in n_trees) {
  for (m in mtry) {
    rf_model <- randomForest(cluster ~age + sexe + taux + situationFamiliale +
nbEnfantsAcharge + SecondeVoiture, data = ImmatClients_EA[, predictors], ntree
= n, mtry = m)
    predictions <- predict(rf_model, newdata = ImmatClients_ET[, predictors])
    accuracy <- sum(predictions == ImmatClients_ET$cluster) /
nrow(ImmatClients_ET)
    results_rf[[paste("n=", n, ", mtry=", m)]] <- accuracy
  }
}
# Afficher les résultats de Random Forest
```

```
print(results_rf)
```

Avec ici n le nombre d'arbres et mtry le nombre de variables prédictives.

```
> print(results_rf)
$`n= 50 , mtry= 2`
[1] 0.7242287

$`n= 50 , mtry= 4`
[1] 0.6962128

$`n= 50 , mtry= 6`
[1] 0.6779856

$`n= 100 , mtry= 2`
[1] 0.7219334

$`n= 100 , mtry= 4`
[1] 0.6960778

$`n= 100 , mtry= 6`
[1] 0.6768379

$`n= 200 , mtry= 2`
[1] 0.7242625

$`n= 200 , mtry= 4`
[1] 0.6963478

$`n= 200 , mtry= 6`
[1] 0.6784581
```

Dans cette dernière partie on va prédire les classes sur le fichier marketing :

On commence par charger les données

```
# Charger les données
Marketing <- read.csv("C:/vagrant-
projects/OracleDatabase/21.3.0/ProjetTPABigData/R/Marketing_ext.csv", header =
FALSE)
```

On utilise le plus précis des Random Forest d'après les tests faits précédemment

```
# Prédire les classes avec Random Forest en utilisant le plus précis (n = 200
et mtry = 2)
rf_model <- randomForest(cluster ~ age + sexe + taux + situationFamiliale +
nbEnfantsAcharge + SecondeVoiture, data = ImmatClients_EA, ntree = 200, mtry =
2)
Marketing$cluster <- predict(rf_model, newdata = Marketing)
```

```r
# Vérifier le nombre d'observations dans chaque classe
table(Marketing$cluster)
```

```
> table(Marketing$cluster)

 1  2  3  4
 5  1 10  3
```

```r
# Sauvegarder la nouvelle table avec la colonne 'cluster'
write.csv(Marketing, file = "C:/vagrant-projects/OracleDatabase/21.3.0/ProjetTPABigData/R/Marketing_cluster.csv", row.names = TRUE)
```

	A	B	C	D	E	F
1	,"age","sexe","taux","situationFamiliale","nbEnfantsAcharge","SecondeVoiture","cluster"					
2	1,35,"m",223,"Celibataire",0,"false","3"					
3	2,48,"m",401,"Celibataire",0,"false","3"					
4	3,26,"f",420,"En Couple",3,"true","4"					
5	4,80,"m",530,"En Couple",3,"false","2"					
6	5,27,"f",153,"En Couple",2,"false","3"					
7	6,59,"f",572,"En Couple",2,"false","3"					
8	7,43,"f",431,"Celibataire",0,"false","1"					
9	8,64,"m",559,"Celibataire",0,"false","3"					
10	9,22,"m",154,"En Couple",1,"false","3"					
11	10,79,"f",981,"En Couple",2,"false","3"					
12	11,55,"m",588,"Celibataire",0,"false","1"					
13	12,19,"f",212,"Celibataire",0,"false","3"					
14	13,34,"f",1112,"En Couple",0,"false","3"					
15	14,60,"m",524,"En Couple",0,"true","1"					
16	15,22,"m",411,"En Couple",3,"true","4"					
17	16,58,"m",1192,"En Couple",0,"false","3"					
18	17,54,"f",452,"En Couple",3,"true","4"					
19	18,35,"m",589,"Celibataire",0,"false","1"					
20	19,59,"m",748,"En Couple",0,"true","1"					
21						

```
> aggregate(Immatriculations[, -c(1:2)], by=list(Immatriculations$cluster), FUN=mean)
  Group.1 nom puissance longueur nbPlaces nbPortes couleur occasion     prix cluster
1       1  NA   71.9010 1.000000        5 4.521012      NA       NA 14394.04       1
2       2  NA  449.0357 4.000000        5 5.000000      NA       NA 84149.02       2
3       3  NA  154.2366 2.298587        5 5.000000      NA       NA 25688.31       3
4       4  NA  245.3290 4.000000        5 5.000000      NA       NA 42660.57       4
```

Avec le cluster 1 qui correspond aux citadines, le cluster 2 aux voitures de luxes, le cluster 3 aux citadines plus et le cluster 4 aux routières / familiales.

Pour la version que nous avons produit en local, on sauvegarde le marketing.csv avec des clusters définis que l'on va ensuite pouvoir insérer manuellement dans notre table hive.

Pour la version avec la machine virtuelle on crée une table marketing avec une colonne cluster dans Hive dans laquelle on insère les données des clusters faits avec Immatriculation.

On pourra par la suite faire de la visualisation à partir de ces tables.

Liaison RHive

Nous allons ici montrer ce qui diffère entre le travail en local et le travail via la machine virtuelle.

Seul l'appel des données est différent, voici la commande :

```
# Import des csv
clients_12_ext <- rhive.query("SELECT * FROM projetBigData.Clients_12_ext")
clients_4_ext <- rhive.query("SELECT * FROM projetBigData.Clients_4_ext")
```

Import de Marketing_Cluster dans Hive

On ouvre un nouvel invite de commande

```
Vagrant ssh
```

On lance hdfs

```
start-dfs.sh
start-yarn.sh
```

On ouvre un autre invite de commande

```
Vagrant ssh
```

On lance le serveur Hive

```
nohup hive --service metastore > /dev/null &
nohup hiveserver2 > /dev/null &
```

On ouvre un dernier invite de commande

```
Vagrant ssh
```

On crée un répertoire dans hdfs

```
hadoop fs -mkdir marketing_Cluster
```

On met le csv dans le répertoire hdfs

```
hadoop fs -put ../../vagrant/ProjetTPABigData/R/Marketing_cluster.csv marketing_Cluster

[vagrant@oracle-21c-vagrant ~]$ hadoop fs -cat marketing_Cluster/*
,age,sexe,taux,situationFamiliale,nbEnfantsAcharge,SecondeVoiture,cluster
1,35,m,223,Celibataire,0,false,3
2,48,m,401,Celibataire,0,false,3
3,26,f,420,En Couple,3,true,4
4,80,m,530,En Couple,3,false,2
5,27,f,153,En Couple,2,false,3
6,59,f,572,En Couple,2,false,3
7,43,f,431,Celibataire,0,false,1
8,64,m,559,Celibataire,0,false,3
9,22,m,154,En Couple,1,false,3
10,79,f,981,En Couple,2,false,3
11,55,m,588,Celibataire,0,false,1
12,19,f,212,Celibataire,0,false,3
13,34,f,1112,En Couple,0,false,3
14,60,m,524,En Couple,0,true,1
15,22,m,411,En Couple,3,true,4
16,58,m,1192,En Couple,0,false,3
17,54,f,452,En Couple,3,true,4
18,35,m,589,Celibataire,0,false,1
19,59,m,748,En Couple,0,true,1
```

On se connecte au serveur de Hive, puis on crée la table Marketing_cluster dans la bdd projetBigData.

```
beeline -u jdbc:hive2://localhost:10000 vagrant

0: jdbc:hive2://localhost:10000>

CREATE DATABASE IF NOT EXISTS projetBigData;
USE projetBigData;

DROP TABLE IF EXISTS MARKETING_CLUSTER_EXT;

CREATE EXTERNAL TABLE IF NOT EXISTS MARKETING_CLUSTER_EXT(
   age INT,
   sexe STRING,
```

```
    situationFamiliale STRING,
    nbEnfantsAcharge STRING,
    SecondeVoiture STRING,
    cluster INT)
COMMENT 'Marketing_cluster table data from csv'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION 'marketing_Cluster'
TBLPROPERTIES ("skip.header.line.count"="1");
```

Résultat de la commande ci-dessous.

```
select * from MARKETING_CLUSTER_EXT;
```

id	age	sexe	taux	situationfamiliale	nbenfantsacharge	secondevoiture	cluster
1	35	m	223	Celibataire	0	false	3
2	48	m	401	Celibataire	0	false	3
3	26	f	420	En Couple	3	true	4
4	80	m	530	En Couple	3	false	2
5	27	f	153	En Couple	2	false	3
6	59	f	572	En Couple	2	false	3
7	43	f	431	Celibataire	0	false	1
8	64	m	559	Celibataire	0	false	3
9	22	m	154	En Couple	1	false	3
10	79	f	981	En Couple	2	false	3
11	55	m	588	Celibataire	0	false	1
12	19	f	212	Celibataire	0	false	3
13	34	f	1112	En Couple	0	false	3
14	60	m	524	En Couple	0	true	1
15	22	m	411	En Couple	3	true	4
16	58	m	1192	En Couple	0	false	3
17	54	f	452	En Couple	3	true	4
18	35	m	589	Celibataire	0	false	1
19	59	m	748	En Couple	0	true	1

Travail sur la machine

Avec HIVE

```
cp ../../vagrant/ProjetTPABigData/R/R_HIVE_PARTIE2345_CLUSTERING_PREDICT_GR4 ~
cp ../../vagrant/ProjetTPABigData/R/R_HIVE_Clients_4_12_PARTIE1_GR4 ~
```

Activer le mode admin pour l'utilisateur sur le jar jdbc

```
chmod 777 ../../usr/local/hive/lib/hive-jdbc-3.1.3.jar
```

Créer un dossier où installer la lib Rhive

```
sudo mkdir -p /rhive/lib/2.0-0.2
```

Activer le mode admin pour l'utilisateur sur la lib RHive

```
sudo chmod 777 /usr/lib64/R/library/RHive
```

Puis

```
chmod 777 /rhive/lib/2.0-0.2
```

Déplacer le tar.gz de RHive dans la VM

```
cp ../../vagrant/ProjetTPABigData/R/RHive_2.0-0.2.tar.gz ~
```

Lancer R

```
R
```

Installer le package RHive

```
install.packages("RHive_2.0-0.2.tar.gz", repos = NULL, type="source")
```

Tester la connexion

```
hive_test<-rhive.query("SELECT * FROM Marketing_ext")
print(hive_test)
```

Sortir de R

```
q()
```

Lancer les scripts

```
Rscript R_HIVE_PARTIE2345_CLUSTERING_PREDICT_GR4
Rscript R_HIVE_Clients_4_12_PARTIE1_GR4
```

Nous avons préféré utiliser la version locale plutôt que la version liant R et Hive sur la VM (voir partie d'après) car les temps de calculs (du fait de nos ordinateurs) étaient interminables (plus de 15 minutes par requête simple).

scriptR

```r
library(rJava)
library(RHive)
# Connecter à
rhive.connect(host="localhost", port="10000", hiveServer2=TRUE)
#Exécuter une requête
result <- rhive.query("SELECT COUNT(*) FROM Clients_12_ext")
print(result)
#Fermer la connexion
rhive.close()
```

```
[vagrant@oracle-21c-vagrant ~]$ cp ../../vagrant/ProjetTPABigData/R/scriptR ~
[vagrant@oracle-21c-vagrant ~]$ Rscript scriptR
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/St
aticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.cla
ss]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
2023-03-10 15:43:40,403 INFO  Configuration.deprecation (Configuration.java:logDeprecation(1441)) - fs.default.name is d
eprecated. Instead, use fs.defaultFS
[1] FALSE
   X_c0
1 98454
[1] TRUE
[vagrant@oracle-21c-vagrant ~]$
```

Par exemple, ce simple code nous a pris à l'ordinateur 15 minutes à nous afficher le résultat.