Analyse exploratoire des données

L'analyse exploratoire des données nous permet d'identifier des problèmes dans les données (valeurs incohérentes, codage des valeurs manquantes, etc.) et découvrir d'éventuelles propriétés de l'espace des données (valeurs doublons, variables liées, variables d'importance particulière ou bien inutiles, etc.).

Après avoir chargé les librairies qui nous serons utiles tout au long du projet et après avoir créé les tables sur Oracle puis nos dataframes comme expliqué dans le rapport du projet, nous nous sommes mis sur le tri et nettoyage des données des différents data frame.

En premier lieu, nous avons fait un tri en parcourant toutes les données de client colonnes par colonnes. Nous avons remarqué de nombreuses incohérences :

Pour le sexe, il y avait Masculin M et Homme par exemple, nous avons souhaité regrouper ces 3 données en une seule car nous étions sur du sexe.
Pareil pour Célibataire, seul et seule, nous avons pris l'initiative de tout regrouper car cela nous semblait plus cohérent que d'avoir 3 noms qui désigne la même situationFamiliale.

```
#remplacer les données coquilles dans sexe

#client$sexe <- str_replace(client$sexe, "Homme", "M")
#client$sexe <- str_replace(client$sexe, "Masculin", "M")
#client$sexe <- str_replace(client$sexe, "Féminin", "F")
#client$sexe <- str_replace(client$sexe, "Femme", "F")

#Non finalement on a vu en enlevant les doublons qu'il fallait pas faire comme ça car on reste
#avec trop de données qu'on ne peut pas exploiter par la suite, donc on suppprime tout simplement Féminin, masculin Homme et femme.
```

```
# les catégories existantes de situation familiale sont actuellement : Seul, Seule, Celibataire, Marié(e), En couple, Divorcée.
#Nous allons remplacer tuos les "seul" et "seule" par celibataire

#client$situationFamiliale <- str_replace(client$situationFamiliale, "Seul", "Célibataire")
#client$situationFamiliale <- str_replace(client$situationFamiliale, "Seule", "Célibataire")
#client$situationFamiliale <- str_replace(client$situationFamiliale, "Célibatairee", "Célibataire")
```

Finalement, après relecture du sujet nous nous sommes rendu compte que le CDC nous demandés des données précises que nous avons par la suite respecté. Donc chaque donnée que nous ne connaissions pas (ex : Homme) nous les avons supprimés.

```
#finalement, on veut enlever toutes les données qui ne sont pas M et F

client <- filter(client, client$sexe=="M" | client$sexe=="F")

client <- filter (client, situationFamiliale=="Célibataire" | situationFamiliale=="Divorcée" | situationFamiliale=="En Couple"|
                  situationFamiliale=="Marié(e)" | situationFamiliale=="Seul" | situationFamiliale=="Seule")
```

Nous avons ensuite, fusionnés les fichiers Clients et immatriculations afin de vérifier les doublons. Sans surprise il y en a plusieurs car les totaux de clientComplet et Client sont différents.

```
clientComplet <-merge(client, immatriculations, by="immatriculation")

#d'accord j'ai créé mon client Complet mais j'ai plus de ligne que de client
```

ATTENTION AUX DOUBLONS

Pour clients dans les immatriculations - 18 doublons.

1er doublon

Client [client$immatriculation == "1557 AB 48",]
La même immatriculation appartient à 2 personnes totalement différente, voyons si dans immatriculations elle est double.

Nous voyons qu'elle correspond à 2 voitures différentes, ce qui est un problème car après la liaison dans client complet : 1 immatriculation créer 4 lignes dans client complet.

Ce qui fait que les 18 doublons d'immatriculations dans client créent 18*4 lignes = 72 lignes en plus dans client Complet.

```
#ON SUPPRIME LES DOUBLONS DANS CLIENT
client <- client[duplicated(client$immatriculation) =="FALSE",]
client[duplicated(client$immatriculation) =="TRUE",]
```

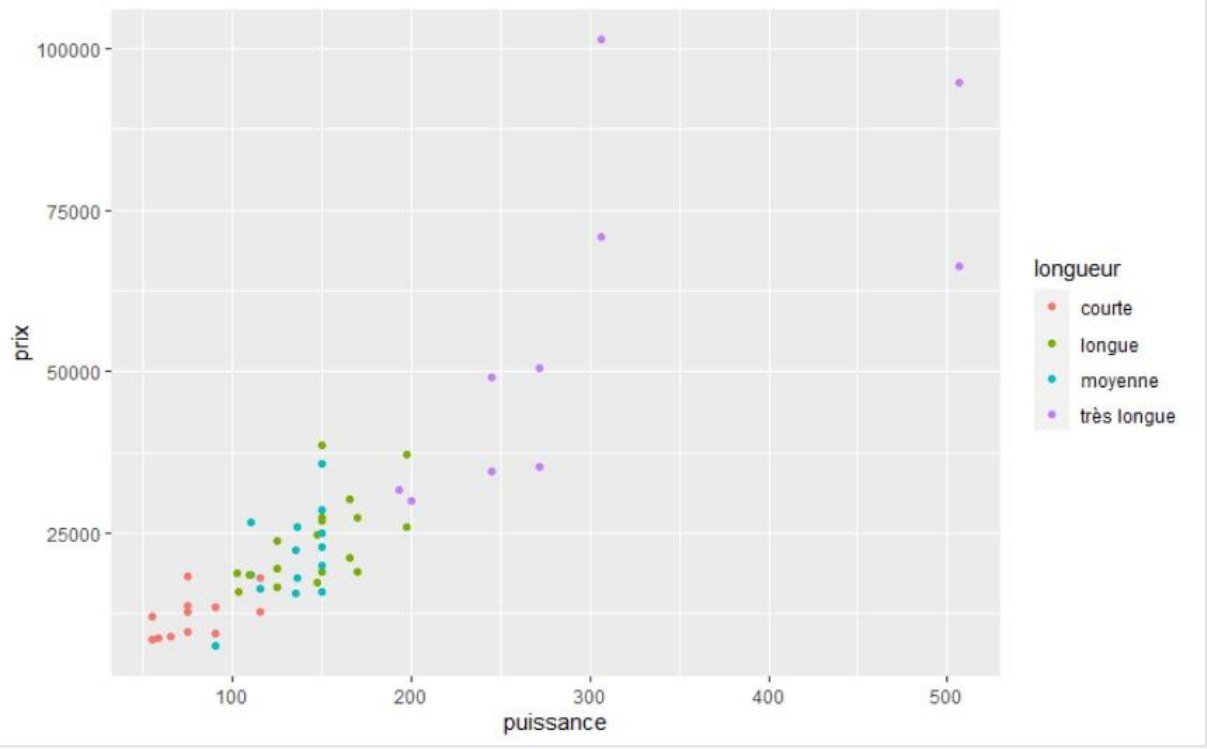
On a pu supprimer tous les doublons et donc on a re fusionné les fichiers.

Application des catégories de véhicules

Pour entamer la fabrication des catégories de véhicules, nous avons tracer plusieurs nuages de points afin de voir s'il y a des groupes de voiture qui se forment "naturellement". Sans surprise, la longueur et la puissance sont les critères les plus "utiles".

Nous nous sommes essentiellement basés sur ce qplot car c'était le plus « parlant » pour faire nos catégories :

>qplot(longueur, nbPortes, data=catalogue)



Nous décidons alors de faire comme critères :

#citadines : les courtes

#sport : +de 300cv

#berline compact : moyennes

#berline : des longues mais pas de 7places

#berline confort : très longue mais supérieur à 190 et inférieur à 300

Nous remarquons des incohérences, par exemple : new beetle ce n'est pas 5 places mais 4 donc pas dans la bonne catégorie car on ne peut pas la conseiller à des familles (trop serrées à l'arrière), donc cette voiture irait plus dans la catégorie citadine

Nous nous rendons compte qu'il n'y a aucune voiture de 7 places, devons-nous supposer que les familles de 4 enfants choisiront automatiquement un monospace ?

Oui du coup.

Fusion des fichiers Clients.csv et Immatriculations.csv

Après avoir créé les catégories, nous refusionnions nos fichiers.

```
clientComplet <-merge(client, immatriculations, by="immatriculation")
```

On vérifie que les catégories choisies sont cohérentes (pas toutes les voitures dans une seule catégorie par ex)

```
> table(clientComplet$categorie)

       berline berline_compact berline_confort        citadine           sport
         12414            2793             312           12288           10843
```

Création d'un modèle de classification supervisée

Suppression des colonnes pas utiles, toutes sauf celles qui correspondent aux clients.

A la suite de pas mal d'erreurs rencontrées, nous avons décidés de supprimer la colonne taux car l'intervalle des taux était trop grand et nous posait des problèmes suivant les ordinateurs.

Création des ensembles d'apprentissage et de test :

#2/3 :

client_EA <- clientComplet[1:25766,]

#1/3 :

client_ET <- clientComplet[25767:38650,]

Puis chaque classifieurs (sans taux) :

Sans taux

```
clientComplet        38650 obs. of 6 variables
   age : Factor w/ 67 levels "18","19","20",..: 52 36 24 1 4 43
   sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
   situationFamiliale: Factor w/ 9 levels " ","?","Célibataire",
   nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1
   X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2
   categorie : Factor w/ 5 levels "berline","berline_compact",..
```

NEURAL NETWORKS

classifieur_nn <- nnet(categorie~., client_EA, size=5)

```
#sans taux et immatriculation et tout en factor

pred.nn <- predict(classifieur_nn,client_ET, type="class")

table(pred.nn)

#pred.nn
#berline berline_compact        citadine          sport
#5255              379            4688           2563

#berline berline_compact        citadine          sport
#5268              292            4774           2550


#matrice de confusion

table(client_ET$categorie, pred.nn)

#pred.nn
#berline berline_compact citadine sport
#berline            3752              0       0    370
#berline_compact       0            178     769      0
#berline_confort      38              0       0     67
#citadine              0            201    3919      0
#sport              1465              0       0   2126

#berline berline_compact citadine sport
#berline            3759              0       0    363
#berline_compact       0            141     806      0
#berline_confort      38              0       0     67
#citadine              0            151    3968      0
#sport              1471              0       0   2120
```

Sur la deuxième matrice de confusion nous avons :

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes : 9988

Prédictions incorrectes : 2896

Précision du classifieur : 9988/12884 = 77.52%

Taux d'erreur : 2896/ 12884 = 22.48% nn_auc

```
#Indices AUC

#il nous faut les probabilités de prédictions des classifieurs

prob.nn <- predict(classifieur_nn, client_ET, type="raw")

#on test:
nn_auc <- multiclass.roc(client_ET$categorie, prob.nn)
print(nn_auc)

#Data: multivariate predictor prob.nn with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.8847
```

: 0.8847

NAIVE BAYES

```
# Apprentissage du classifeur de type naive bayes

nb <- naive_bayes(categorie~., client_EA)

#warning messages:
#  1: naive_bayes(): Feature age - zero probabilities are present. Consider Laplace smoothing.
#2: naive_bayes(): Feature taux - zero probabilities are present. Consider Laplace smoothing.
#3: naive_bayes(): Feature situationFamiliale - zero probabilities are present. Consider Laplace smoothing.
#4: naive_bayes(): Feature nbEnfantsAcharge - zero probabilities are present. Consider Laplace smoothing.
#5: naive_bayes(): Feature X2eme.voiture - zero probabilities are present. Consider Laplace smoothing.

# Test du classifieur : classe predite

nb_class <- predict(nb, client_ET, type="class")
table(nb_class)

#berline berline_compact berline_confort        citadine           sport
#5056             679               0           3563            3586

# Matrice de confusion
table( client_ET$categorie, nb_class)

#                berline berline_compact berline_confort citadine sport
#berline            3614               0               0        1   507
#berline_compact       0             337               0      610     0
#berline_confort      36               0               0        0    69
#citadine              0             342               0     2951   826
#sport              1406               0               0        1  2184
```

```
# Test du classifieur : probabilites pour chaque prediction
nb_prob <- predict(nb, client_ET, type="prob")

#Indice AUC
nb_auc <- multiclass.roc(client_ET$categorie, nb_prob)

print(nb_auc)

#Data: multivariate predictor nb_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.8621
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9086

Prédictions incorrectes : 3798

Précision du classifieur : 9086/12884 = 70.52%

Taux d'erreur : 3798 / 12884 = 29.41% nn_auc

: 0.8621

C5.0

```
#-------------#
# C5.0        #
#-------------#


# Apprentissage du classifeur de type arbre de décision
tree_C50 <- C5.0(categorie~., client_EA)
tree_C50

#TCall:
#C5.0.formula(formula = categorie ~ ., data = client_EA)

#Classification Tree
#Number of samples: 25766
#Number of predictors: 5

#Tree size: 6

#Non-standard options: attempt to group attributes

C50_class<-predict(tree_C50, client_ET, type="class")


table(C50_class)

#C50_class
#        berline berline_compact berline_confort        citadine           sport
#           5268               0               0            5066            2550

table(client_ET$categorie, C50_class)

#                 berline berline_compact berline_confort citadine sport
#berline             3759               0               0        0   363
#berline_compact        0               0               0      947     0
#berline_confort       38               0               0        0    67
#citadine               0               0               0     4119     0
#sport               1471               0               0        0  2120

c50_prob<-predict(tree_C50, client_ET, type="prob")


c50_auc<-multiclass.roc(client_ET$categorie, c50_prob)

print(c50_auc)


#Data: multivariate predictor c50_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.8667
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9998

Prédictions incorrectes : 2886

Précision du classifieur : 9998/12884 = 77.60% Taux d'erreur : 2886 / 12884 = 22.39%

Auc : 0.8667

K-NEAREST NEIGHBORS

```
#----------------------#
# K-NEAREST NEIGHBORS #
#----------------------#

# Apprentissage et test simultanes du classifeur de type k-nearest neighbors
classifieur_knn <- kknn(categorie~., client_EA, client_ET)

# Resultat : classe predite et probabilites de chaque classe pour chaque instance de test
summary(classifieur_knn)

# Matrice de confusion
table(client_ET$categorie, classifieur_knn$fitted.values)


#              berline berline_compact berline_confort citadine sport
#berline          3128              0               0        0   994
#berline_compact     0            225               0      722     0
#berline_confort    34              0               0        0    71
#citadine            0            363               0     3756     0
#sport            1330              0               3        0  2258


# Conversion des probabilites en data frame
knn_prob <- as.data.frame(classifieur_knn$prob)


# Calcul de l'AUC
knn_auc<-multiclass.roc(client_ET$categorie, knn_prob)
print(knn_auc)
#Data: multivariate predictor knn_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.7996
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9367

Prédictions incorrectes : 3517

Précision du classifieur : 9367/12884 = 72.70%

Taux d'erreur : 3517 / 12884 = 27.30% nn_auc

: 0.7996

RANDOM FOREST

```
#-------------#
#RANDOM FOREST#
#-------------#


classifieur_rf <- randomForest(categorie~., client_EA)

#Error in randomForest.default(m, y, ...) :
#  Can not handle categorical predictors with more than 53 categories.


#donc, on enlève ce qui pourrait poser problème donc age


client_EA <- subset(client_EA, select = -age)
client_ET <- subset(client_ET, select = -age)

classifieur_rf <- randomForest(categorie~., client_EA)

pred_rf <- predict(classifieur_rf, client_ET, type="response")

table(pred_rf)

#berline berline_compact berline_confort         citadine            sport
#5264                 0                 0             5066             2554
```

```
#matrice de confusion
table(client_ET$categorie, pred_rf)

#                berline berline_compact berline_confort citadine sport
#berline            3756              0               0        0   366
#berline_compact       0              0               0      947     0
#berline_confort      38              0               0        0    67
#citadine              0              0               0     4119     0
#sport              1470              0               0        0  2121

# Test du classifieur : probabilites pour chaque prediction
rf_prob <- predict(classifieur_rf, client_ET, type="prob")
# L'objet genere est une matrice
rf_prob


# Calcul de l'AUC
rf_auc <- multiclass.roc(client_ET$categorie, rf_prob)


print(rf_auc)

#Data: multivariate predictor rf_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.7021
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9996

Prédictions incorrectes : 2888

Précision du classifieur : 9996/12884 = 77.58%

Taux d'erreur : 2888 / 12884 = 22.42% nn_auc

: 0.7021

SVM

```
#---#
#SVM#
#---#

svm_class <- svm(categorie~., client_EA, probability=TRUE)

# Test du classifieur : classe predite
svm_pred <- predict(svm_class, client_ET, type="response")
svm_pred


table(svm_pred)

#berline berline_compact berline_confort        citadine           sport
#5051                0               0            5066            2767

#matrice de confusion

table(client_ET$categorie, svm_pred)

                #berline berline_compact berline_confort citadine sport
#berline             3610               0               0        0   512
#berline_compact        0               0               0      947     0
#berline_confort        36              0               0        0    69
#citadine               0               0               0     4119     0
#sport               1405               0               0        0  2186
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9915

Prédictions incorrectes : 2969

```
# Test du classifieur : probabilites pour chaque prediction
svm_prob <- predict(svm_class, client_ET, probability=TRUE)


# Recuperation des probabilites associees aux predictions
svm_prob <- attr(svm_prob, "probabilities")

# Conversion en un data frame
svm_prob <- as.data.frame(svm_prob)


# Calcul de l'AUC
svm_auc <-multiclass.roc(client_ET$categorie, svm_prob)

print (svm_auc)

#Data: multivariate predictor knn_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.8673
```

Précision du classifieur : 9915/12884 = 76.96%

Taux d'erreur : 2969 / 12884 = 23.04% nn_auc

: 0.8673

Nous avons essayé également avec les classifieurs R-part et Tree mais cela ne marche pas puisqu'il y a trop de données pour ce type de classifieur.

Puis chaque classifieurs (avec taux) :

Avec taux

NAIVE BAYES

Dans cette configuration, cela correspond à « avec taux » et sans mettre en « factor » les colonnes :

```
client_EA          25766 obs. of 7 variables
  age : num 69 53 41 18 21 60 66 51 49 27 ...
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  taux : num 767 983 587 960 707 ...
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3 3 1...
  nbEnfantsACharge : num 1 0 4 4 1 0 0 0 1 1 ...
  X2eme.voiture : logi FALSE FALSE FALSE TRUE TRUE FALSE ...
  categorie : chr "sport" "sport" "berline" "sport" ...
```

```
#-------------#
# NAIVE BAYES #
#-------------#

# Apprentissage du classifeur de type naive bayes
nb <- naive_bayes(client_EA$categorie~., client_EA)
nb

#warning messages:
#1: naive_bayes(): Feature situationFamiliale - zero probabilities are present. Consider Laplace smoothing.
#2: naive_bayes(): Feature X2eme.voiture - zero probabilities are present. Consider Laplace smoothing.

# Test du classifieur : classe predite

nb_class <- predict(nb, client_ET, type="class")
nb_class
table(nb_class)


#berline berline_compact berline_confort        citadine           sport
#4366            2672             469            2251            3126

# Matrice de confusion
table( client_ET$categorie, nb_class)

#                berline berline_compact berline_confort citadine sport
#berline            3169              1             231      204   517
#berline_compact       0            889               0       58     0
#berline_confort      40              0              38        0    27
#citadine              0           1781              74     1877   387
#sport              1157              1             126      112  2195
```

```
# Test du classifieur : probabilités pour chaque prediction
nb_prob <- predict(nb, client_ET, type="prob")
nb_prob #matrice

# Courbe ROC
nb_pred <- multiclass.roc(client_ET$categorie, nb_prob)


nb_pred

#Data: multivariate predictor nb_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
#Multi-class area under the curve: 0.9017
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :8168

Prédictions incorrectes : 4716

Précision du classifieur : 8168/12884 = 63.40%

Taux d'erreur : 4716 / 12884 = 36.60% Auc : 0.9017

Avec cette configuration, c'est-à-dire avec taux et tout en factor :

```
client_EA          25766 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 52 36 24 1 4 43 49 34 3...
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 124 340 44 317 64 ...
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3 3 1...
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 5 2 ...
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1 1 2...
  categorie : Factor w/ 5 levels "berline","berline_compact",..: 5 5 1...
client_ET          12884 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 5 29 61 29 60 12 42 46 ...
  sexe : Factor w/ 2 levels "F","M": 2 1 2 2 2 2 2 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 171 432 360 27 40 ...
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 1 6 3 3...
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 1 2 1 4 2 ...
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 2 1 1 2 1...
  categorie : chr "citadine" "sport" "berline_compact" "berline" ...
```

```
#_____________#
#             #
#  naive bayes  #
#             #
#_____________#

# Apprentissage du classifeur de type naive bayes
nb <- naive_bayes(client_EA$categorie~., client_EA)

# Test du classifieur : classe predite
nb_class <- predict(nb, client_ET, type="class")
table(nb_class)

#berline berline_compact berline_confort        citadine          sport
#4503            1031               0            3673           3677

# Matrice de confusion
table(client_ET$categorie, nb_class)

#               berline berline_compact berline_confort citadine sport
#berline           3191               0               0      267   664
#berline_compact      0             501               0      446     0
#berline_confort     44               0               0        0    61
#citadine            91             530               0     2822   676
#sport             1177               0               0      138  2276


# Test du classifieur : probabilites pour chaque prediction
nb_prob <- predict(nb, client_ET, type="prob")


# Calcul de l'AUC
nb_auc <-multiclass.roc(client_ET$categorie, nb_prob)
print (nb_auc)

#Data: multivariate predictor nb_prob with 5 levels of client_ET$categorie: berline, berline_compact, berline_confort, citadine, sport.
##Multi-class area under the curve : 0.8931
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :8790

Prédictions incorrectes : 4094

Précision du classifieur : 8790/12884 = 68.22%

Taux d'erreur : 4094 / 12884 = 21.78% Auc : 0.8931

C5.0 :

Avec cette configuration, c'est-à-dire avec taux et tout en factor :

```
client_EA          25766 obs. of 7 variables
 age : Factor w/ 67 levels "18","19","20",..: 52 36 24 1 4 43 49 34 3…
 sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
 taux : Factor w/ 756 levels "544","545","546",..: 124 340 44 317 64 …
 situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3 3 1…
 nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 5 2 …
 X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1 1 2…
 categorie : Factor w/ 5 levels "berline","berline_compact",..: 5 5 1…
client_ET          12884 obs. of 7 variables
 age : Factor w/ 67 levels "18","19","20",..: 5 29 61 29 60 12 42 46 …
 sexe : Factor w/ 2 levels "F","M": 2 1 2 2 2 2 2 1 2 2 ...
 taux : Factor w/ 756 levels "544","545","546",..: 171 432 360 27 40 …
 situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 1 6 3 3…
 nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 1 2 1 4 2 …
 X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 2 1 1 2 1…
 categorie : chr "citadine" "sport" "berline_compact" "berline" ...
```

```
#____________________#
#   C5.0             #
#____________________#

# Apprentissage du classifeur de type arbre de décision
dt <- C5.0(client_EA$categorie~., client_EA)
print(dt)


#Classification Tree
#Number of samples: 25766 |
#Number of predictors: 6

#Tree size: 29

#Non-standard options: attempt to group attributes


# Test du classifieur : classe predite
dt_class <- predict(dt, client_ET, type="class")
dt_class
table(dt_class)

#berline berline_compact berline_confort        citadine           sport
#5782            1086               0            3980            2036

# Matrice de confusion
table(client_ET$categorie, dt_class)


                #berline berline_compact berline_confort citadine sport
#berline            4053               0               0        0    69
#berline_compact       0             538               0      409     0
#berline_confort      86               0               0        0    19
#citadine              0             548               0     3571     0
#sport              1643               0               0        0  1948

# Test du classifieur : probabilites pour chaque prediction
dt_prob <- predict(dt, client_ET, type="prob")

# Calcul de l'AUC
c_auc <-multiclass.roc(client_ET$categorie, dt_prob)
print (c_auc)

#Data: multivariate predictor dt_prob with 5 levels of client_ET$categorie: berline, berline_compact, b
#Multi-class area under the curve: 0.9091
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :10110

Prédictions incorrectes : 2774

Précision du classifieur : 10110/12884 = 78.47% Taux d'erreur : 2774/ 12884 = 21.53% Auc : 0.9091

K-NEAREST NEIGHBORS :

Avec cette configuration, c'est-à-dire avec taux et tout en factor :

```
client_EA          25766 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 52 36 24 1 4 43 49 34 3…
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 124 340 44 317 64 …
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3 3 1…
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 5 2 …
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1 1 2…
  categorie : Factor w/ 5 levels "berline","berline_compact",..: 5 5 1…
client_ET          12884 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 5 29 61 29 60 12 42 46 …
  sexe : Factor w/ 2 levels "F","M": 2 1 2 2 2 2 2 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 171 432 360 27 40 …
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 1 6 3 3…
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 1 2 1 4 2 …
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 2 1 1 2 1…
  categorie : chr "citadine" "sport" "berline_compact" "berline" ...
```

```r
#--------------------#
# K-NEAREST NEIGHBORS #
#--------------------#

# Apprentissage et test simultanes du classifeur de type k-nearest neighbors
classifieur_knn <- kknn(categorie~., client_EA, client_ET)
# Error in if (response != "continuous") { :
#     l'argument est de longueur nulle
#classifieur_knn <- kknn(categorie~age + sexe +taux+ situationFamiliale+nbEnfantsAcharge+X2eme.voiture, client_EA, client_ET)
# Error in if (response != "continuous") { :
#     l'argument est de longueur nulle


# Resultat : classe predite et probabilites de chaque classe pour chaque instance de test
summary(classifieur_knn)

# Matrice de confusion
table(client_ET$categorie, classifieur_knn$fitted.values)

#                 berline berline_compact berline_confort citadine sport
#berline            3123              0              3        0   996
#berline_compact       0            250              0      697     0
#berline_confort      37              0              0        0    68
#citadine              0            375              0     3744     0
#sport              1323              0              3        0  2265

# Conversion des probabilites en data frame
knn_prob <- as.data.frame(classifieur_knn$prob)


# Calcul de l'AUC
knn_auc<-multiclass.roc(client_ET$categorie, knn_prob)
print(knn_auc)

#Multi-class area under the curve: 0.8017
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes :9382

Prédictions incorrectes : 3502

Précision du classifieur : 9382/12884 = 72.82% Taux d'erreur : 3502/ 12884 = 27.18%

Auc : 0.8017

SVM :

Avec cette configuration, c'est-à-dire avec taux et tout en factor :

```
client_EA          25766 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 52 36 24 1 4 43 49 34 3…
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 124 340 44 317 64 …
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3 3 1…
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 5 2 …
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1 1 2…
  categorie : Factor w/ 5 levels "berline","berline_compact",..: 5 5 1…
client_ET          12884 obs. of 7 variables
  age : Factor w/ 67 levels "18","19","20",..: 5 29 61 29 60 12 42 46 …
  sexe : Factor w/ 2 levels "F","M": 2 1 2 2 2 2 2 1 2 2 ...
  taux : Factor w/ 756 levels "544","545","546",..: 171 432 360 27 40 …
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 1 6 3 3…
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 1 2 1 4 2 …
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 2 1 1 1 2 1 1 2 1…
  categorie : chr "citadine" "sport" "berline_compact" "berline" ...
```

```
#_____________________#
#                     #
#        svm          #
#                     #
#_____________________#

# Apprentissage du classifieur de type svm
svm <- svm(categorie~., client_EA, probability=TRUE)

# Test du classifieur : classe predite
svm_class <- predict(svm, client_ET, type="response")
svm_class
table(svm_class)

#        berline berline_compact berline_confort        citadine           sport
#           5004                0               0            5115            2765

# Matrice de confusion
table(client_ET$categorie, svm_class)

#                berline berline_compact berline_confort citadine sport
#berline            3576               0               0       34   512
#berline_compact       0               0               0      947     0
#berline_confort      35               0               0        1    69
#citadine              0               0               0     4119     0
#sport              1393               0               0       14  2184

# Test du classifieur : probabilites pour chaque prediction
svm_prob <- predict(svm, client_ET, probability=TRUE)


# Recuperation des probabilites associees aux predictions
svm_prob <- attr(svm_prob, "probabilities")

# Conversion en un data frame
svm_prob <- as.data.frame(svm_prob)


# Calcul de l'AUC
svm_auc <-multiclass.roc(client_ET$categorie, svm_prob)
print (svm_auc)

##Multi-class area under the curve: 0.8944
```

Taux d'erreur :

Prédictions correctes :9879
Prédictions incorrectes : 3005 **Précision du classifieur : 9879/12884 = 76.68% Taux d'erreur : 3005/ 12884 = 23.32% Auc : 0.8944**

Ensemble de test : 12884

RANDOM FOREST :

Aux vues du fait que random forest ne marche que pour 53 itérations max nous allons diviser taux en catégorie et âge en catégorie (nous avons essayé de ne pas mettre age en catégorie mais ca nous donnait un résultat moins précis) :

Pour mettre taux et âge en catégorie, nous avons fait comme ceci :

```
clientCompletStauxEchelons <- ifelse(clientCompletStaux <=829, clientCompletStauxEchelons <- "echelon1",
                                     ifelse(clientCompletStaux >=1114, clientCompletStauxEchelons <- "echelon 3",
                                            ifelse(clientCompletStaux > 829 & clientCompletStaux< 1114, clientCompletStauxEchelons <-"echelon 2", "NO")))



clientCompletSageEchelons<- ifelse(clientCompletSage <=29, clientCompletSageEchelons <- "vingtaine",
                                   ifelse(clientCompletSage >= 30 & clientCompletSage <=39, clientCompletSageEchelons <- "trentaine",
                                          ifelse(clientCompletSage >= 40 & clientCompletSage <=49, clientCompletSageEchelons <- "quarantaine",
                                                 ifelse(clientCompletSage >=50 & clientCompletSage <=59, clientCompletSageEchelons <- "cinquantaine",
                                                        ifelse(clientCompletSage >=60 & clientCompletSage <=69, clientCompletSageEchelons <- "soixantaine",
                                                               ifelse(clientCompletSage >=70 & clientCompletSage <=79, clientCompletSageEchelons <- "soixante-dizaine",
                                                                      clientCompletSageEchelons <- "quatre-vingtaine"))))))
```

Nous avons donc cette configuration :

```
clientComplet      38650 obs. of 7 variables
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 !
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1
  categorie : Factor w/ 5 levels "berline","berline_compact",..: 5
  tauxEchelons : Factor w/ 3 levels "echelon 2","echelon 3",..: 3 1
  ageEchelons : Factor w/ 7 levels "cinquantaine",..: 4 1 2 7 7 4 4
```

```
#-------------#
#RANDOM FOREST#
#-------------#


classifieur_rf <- randomForest(categorie~., client_EA)


pred_rf <- predict(classifieur_rf, client_ET, type="response")

table(pred_rf)

#        berline berline_compact berline_confort        citadine           sport
#5264              81                0            4985            2554

#matrice de confusion
table(client_ET$categorie, pred_rf)

#                  berline berline_compact berline_confort citadine sport
#berline            3757                0               0        0   365
#berline_compact       0               40               0      907     0
#berline_confort      38                0               0        0    67
#citadine              0               41               0     4078     0
#sport              1469                0               0        0  2122


# Test du classifieur : probabilites pour chaque prediction
rf_prob <- predict(classifieur_rf, client_ET, type="prob")
# L'objet genere est une matrice
rf_prob

# Calcul de l'AUC
rf_auc <- multiclass.roc(client_ET$categorie, rf_prob)
print(rf_auc)


#Multi-class area under the curve: 0.761
```

Taux d'erreur :

Ensemble de test : 12884

Prédictions correctes : 9997

Prédictions incorrectes : 2887

Précision du classifieur : 9997/12884 = 77.59%

Taux d'erreur : 2887/ 12884 = 22.41% Auc : 0.761

NEURAL NETWORKS :

Aux vues du fait que neural networks ne marche que pour 53 itérations max nous allons diviser taux en catégorie et âge en catégorie :

Pour mettre taux et âge en catégorie, nous avons fait comme ceci :

```
clientCompletTauxEchelons <- ifelse(clientCompletTaux <=829, clientCompletTauxEchelons <- "echelon1",
                                    ifelse(clientCompletTaux >=1114, clientCompletTauxEchelons <- "echelon 3",
                                           ifelse(clientCompletTaux > 829 & clientCompletTaux< 1114, clientCompletTauxEchelons <-"echelon 2", "NO")))



clientCompletAgeEchelons<- ifelse(clientCompletAge <=29, clientCompletAgeEchelons <- "vingtaine",
                                  ifelse(clientCompletAge >= 30 & clientCompletAge <=39, clientCompletAgeEchelons <- "trentaine",
                                         ifelse(clientCompletAge >= 40 & clientCompletAge <=49, clientCompletAgeEchelons <- "quarantaine",
                                                ifelse(clientCompletAge >=50 & clientCompletAge <=59, clientCompletAgeEchelons <- "cinquantaine",
                                                       ifelse(clientCompletAge >=60 & clientCompletAge <=69, clientCompletAgeEchelons <- "soixantaine",
                                                              ifelse(clientCompletAge >=70 & clientCompletAge <=79, clientCompletAgeEchelons <- "soixante-dizaine",
                                                                     clientCompletAgeEchelons <- "quatre-vingtaine"))))))
```

Nous avons donc cette configuration :

```
clientComplet      38650 obs. of 7 variables
  sexe : Factor w/ 2 levels "F","M": 1 2 2 2 2 2 1 1 2 2 ...
  situationFamiliale: Factor w/ 6 levels "Célibataire",..: 3 3 3 3
  nbEnfantsAcharge : Factor w/ 5 levels "0","1","2","3",..: 2 1 5 !
  X2eme.voiture : Factor w/ 2 levels "FALSE","TRUE": 1 1 1 2 2 1 1
  categorie : Factor w/ 5 levels "berline","berline_compact",..: 5
  tauxEchelons : Factor w/ 3 levels "echelon 2","echelon 3",..: 3 :
  ageEchelons : Factor w/ 7 levels "cinquantaine",..: 4 1 2 7 7 4 4
```

```
#---------------#
#     NNET      #
#---------------#
classifieur_nn <- nnet(categorie~., client_EA, size=5)

# weights:  130
#initial  value 38171.749126
#iter  10 value 18135.842129
#iter  20 value 13165.086537
#iter  30 value 12163.579751
#iter  40 value 11867.960747
#iter  50 value 11733.571511
#iter  60 value 11632.306993
#iter  70 value 11542.378036
#iter  80 value 11504.454748
#iter  90 value 11495.156494
#iter 100 value 11490.371974
#final  value 11490.371974
#stopped after 100 iterations


#avec taux et age en echelons et tout en factor

pred.nn <- predict(classifieur_nn,client_ET, type="class")

table(pred.nn)

#pred.nn
#        berline berline_compact         citadine           sport
#5289        249            4817             2529
```

```
#matrice de confusion

table(client_ET$categorie, pred.nn)

#pred.nn
#                berline berline_compact citadine sport
#berline           3770              0        0   352
#berline_compact      0            126      821     0
#berline_confort     39              0        0    66
#citadine             0            123     3996     0
#sport             1480              0        0  2111

#Indices AUC

#il nous faut les probabilités de prédictions des classifieurs

prob.nn <- predict(classifieur_nn, client_ET, type="raw")

#on test:
nn_auc <- multiclass.roc(client_ET$categorie, prob.nn)
print(nn_auc)

#Data: multivariate predictor prob.nn with 5 levels of client_ET$categorie: berline,
#Multi-class area under the curve: 0.9203
```

Taux d'erreur :

Ensemble de test : 12884
Prédictions correctes : 10003
Prédictions incorrectes : 2881

Précision du classifieur : 10003/12884 = 77.64%
Taux d'erreur : 2881/ 12884 = 22.36%
Auc :0.9203

Classification – Prédiction de la Catégorie pour le fichier Marketing

Comparaison des résultats avec ou sans taux :

	NNET	NB	C50	KNN	RF	SVM
Sans taux						
précision du classifieur (%)	77,52	70,52	77,6	72,7	77,58	76,96
taux d'erreur (%)	22,48	29,41	22,39	27,3	22,42	23,04
auc	0,8847	0,8621	0,8667	0,7996	0,7021	0,8673
Avec taux						
précision du classifieur (%)	77,64	68,22	78,47	72,82	77,59	76,68
taux d'erreur (%)	22,36	21,78	21,53	27,18	22,41	23,32
auc	0,9203	0,8931	0,9091	0,8017	0,761	0,8944

Petite conclusion :

Nous avons décidé d'utiliser Auc et les taux de précision du classifieur pour choisir notre classifieur car nous ne pouvons pas tracer les courbes Roc pour les multi class. De plus nous remarquons que nous avons des meilleurs résultats avec taux.

Notre choix se porte donc sur 2 classifieurs : NNET et C50

On va donc appliquer la méthode de prédiction NNET et C50, étant donné que c'est celles qui présentent le meilleur AUC et la meilleure précision et donc la meilleure prédiction. L'ensemble d'apprentissage correspond au clientComplet et la prédiction se fera sur le dataframe marketing :

Application de nos classifieurs sur le fichier Marketing :

Résultat pour C50 :

	age	sexe	taux	situationFamiliale	nbEnfantsACharge	X2eme.voiture	dt_predMarketing
1	21	F	1396	Célibataire	0	FALSE	citadine
2	59	F	572	En Couple	2	FALSE	berline
3	64	M	559	Célibataire	0	FALSE	citadine
4	79	F	981	En Couple	2	FALSE	berline
5	55	M	588	Célibataire	0	FALSE	citadine
6	34	F	1112	En Couple	0	FALSE	berline
7	58	M	1192	En Couple	0	FALSE	berline
8	35	M	589	Célibataire	0	FALSE	citadine
9	59	M	748	En Couple	0	TRUE	citadine

Nous remarquons que comme nous n'avions pas mis taux dans des catégories nous avons que 9 types de client sur 20 (dans le fichier marketing) qui peuvent correspondre puisque qu'il y a beaucoup de taux inférieur à 544 (et nous avions donc fait le tri dans client comme demandé dans le cahier des charges). Nous allons donc refaire une prédiction avec C50 et avec des catégories pour taux.

Résultat c50 avec taux en catégorie :

	age	sexe	situationFamiliale	nbEnfantsACharge	X2eme.voiture	tauxEchelons	dt_predMarketing
1	21	F	Célibataire	0	FALSE	echelon 3	citadine
2	35	M	Célibataire	0	FALSE	echelon1	citadine
3	48	M	Célibataire	0	FALSE	echelon1	citadine
4	26	F	En Couple	3	TRUE	echelon1	sport
5	80	M	En Couple	3	FALSE	echelon1	sport
6	27	F	En Couple	2	FALSE	echelon1	berline
7	59	F	En Couple	2	FALSE	echelon1	berline
8	43	F	Célibataire	0	FALSE	echelon1	citadine
9	64	M	Célibataire	0	FALSE	echelon1	citadine
10	22	M	En Couple	1	FALSE	echelon1	berline
11	79	F	En Couple	2	FALSE	echelon 2	berline
12	55	M	Célibataire	0	FALSE	echelon1	citadine
13	19	F	Célibataire	0	FALSE	echelon1	citadine
14	34	F	En Couple	0	FALSE	echelon 2	berline
15	60	M	En Couple	0	TRUE	echelon1	citadine
16	22	M	En Couple	3	TRUE	echelon1	sport
17	58	M	En Couple	0	FALSE	echelon 3	berline
18	54	F	En Couple	3	TRUE	echelon1	sport
19	35	M	Célibataire	0	FALSE	echelon1	citadine
20	59	M	En Couple	0	TRUE	echelon1	citadine

C’est bien mieux !

<u>Résultat pour NNET :</u>

	age	sexe	situationFamiliale	nbEnfantsACharge	X2eme.voiture	tauxEchelons
1	21	F	Célibataire	0	FALSE	echelon 3
2	35	M	Célibataire	0	FALSE	echelon1
3	48	M	Célibataire	0	FALSE	echelon1
4	26	F	En Couple	3	TRUE	echelon1
5	80	M	En Couple	3	FALSE	echelon1
6	27	F	En Couple	2	FALSE	echelon1
7	59	F	En Couple	2	FALSE	echelon1
8	43	F	Célibataire	0	FALSE	echelon1
9	64	M	Célibataire	0	FALSE	echelon1
10	22	M	En Couple	1	FALSE	echelon1
11	79	F	En Couple	2	FALSE	echelon 2
12	55	M	Célibataire	0	FALSE	echelon1
13	19	F	Célibataire	0	FALSE	echelon1
14	34	F	En Couple	0	FALSE	echelon 2
15	60	M	En Couple	0	TRUE	echelon1
16	22	M	En Couple	3	TRUE	echelon1
17	58	M	En Couple	0	FALSE	echelon 3
18	54	F	En Couple	3	TRUE	echelon1
19	35	M	Célibataire	0	FALSE	echelon1
20	59	M	En Couple	0	TRUE	echelon1

	nn_predMarketing
1	citadine
2	citadine
3	citadine
4	sport
5	sport
6	berline
7	berline
8	citadine
9	citadine
10	berline
11	berline
12	citadine
13	citadine
14	berline
15	citadine
16	sport
17	berline
18	sport
19	citadine
20	citadine

Pour finir, nous avons enregistré nos résultats : trois fichiers, deux pour c50 (un avec taux en échelon et un sans) et un pour nnet de prédictions dans le dossier DATA :

```
write.table(resultatC50, file='predictionsC50.csv', sep="\t", dec=".", row.names = F)
write.table(resultatNN, file='predictionsNN.csv', sep="\t", dec=".", row.names = F)

write.table(resultatC50_2, file='predictionsC50_2.csv', sep="\t", dec=".", row.names = F)
```

Conclusion : Nous remarquons qu'il n'y a que 3 catégories sur nos 5 prédites. Cela vient du fait que dans notre fichier client nous n'avions pas de personne avec des voitures 7 places donc la catégorie confort qui a 7 places est tout le temps vide mais nous restons persuadés qu'une famille nombreuse (avec 4 enfants ou plus) choisira certainement une berline confort (7places). De plus la catégorie Berline compact n'apparait pas mais nous avons que 20 types de personnes dans le fichier Marketing. Nous aurions peut-être pu faire plus de catégories afin de lier au mieux chaque client pour chaque type de voiture mais nous sommes assez satisfaits des résultats.