Analyse des Données par les Techniques de Data Mining, MachineLearning et Deep Learning

Groupe 1 Projet Big Data Analytics

Par Baroukh Yoni, Choisy Jérémy, Da Silva Goncalves André, Panzera Alexis

Lien github du projet

Lien du groupe sur dropbox

Lien vers la data visualisation

UNIVERSITÉ CÔTE D'AZUR

MIAGE RÉSEAU DES MIAGE DE FRANCE Nice

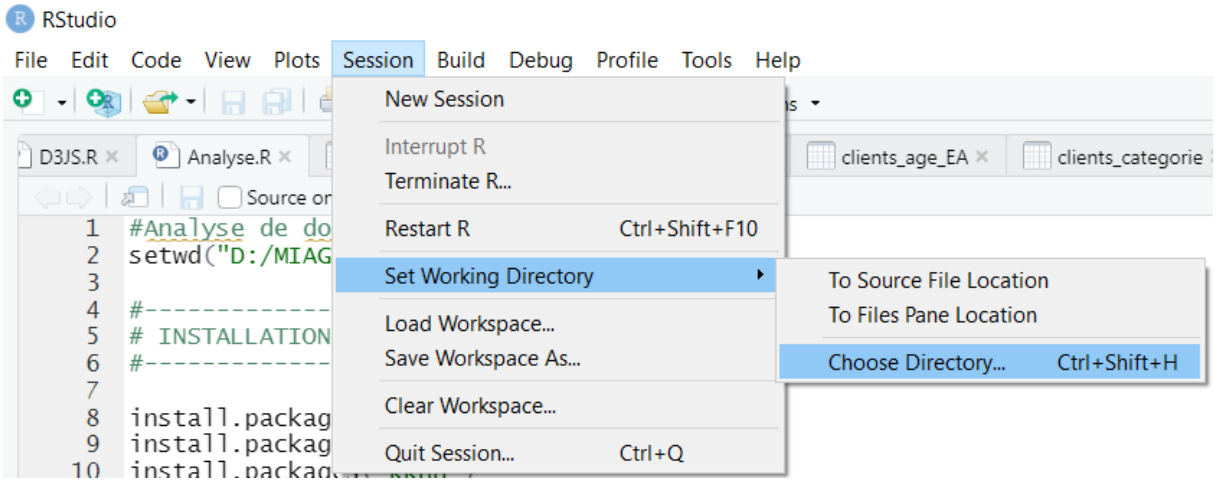
Table des matières

Exécution du script

Avant l'exécution du script Analyse.R, nous devons mettre à jour le chemin du répertoire de travail où sont contenus les données :

1) Cloner le repository Git https://github.com/DSGAndre/TPA-BigData

2) Mettre à jour le chemin du répertoire de travail en se rendant sur l'onglet Session > Set Working Directory > Choose Director, et en choisissant le répertoire data qui se trouve dans le dossier cloné à l'étape 1 : Analyse_de_donnees/script/data

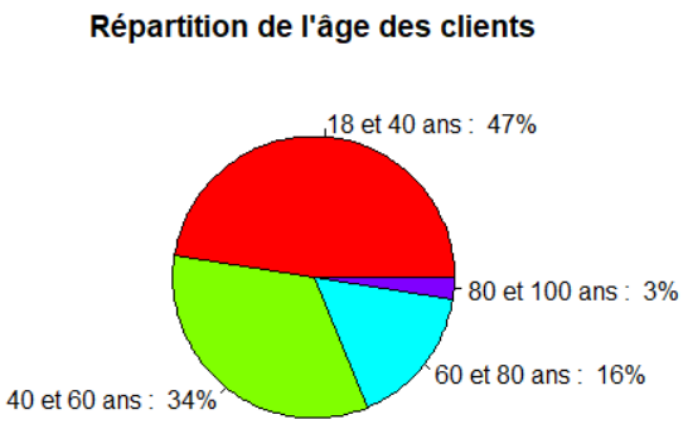


Enfin télécharger l'ensemble des librairies utiles au bon fonctionnement du script.

```
install.packages("rpart")          library(rpart)
install.packages("randomForest")   library(randomForest)
install.packages("kknn")           library(kknn)
install.packages("ROCR")           library(ROCR)
install.packages("dplyr")          library(dplyr)
install.packages("ggplot2")        library(ggplot2)
install.packages("tidyr")          library(tidyr)
install.packages("stringr")        library(stringr)
install.packages("ROCR")           library(ROCR)
install.packages("pROC")           library(pROC)
installed.packages("nnet")         library(nnet)
install.packages("naivebayes")     library(naivebayes)
```

Exploration des données

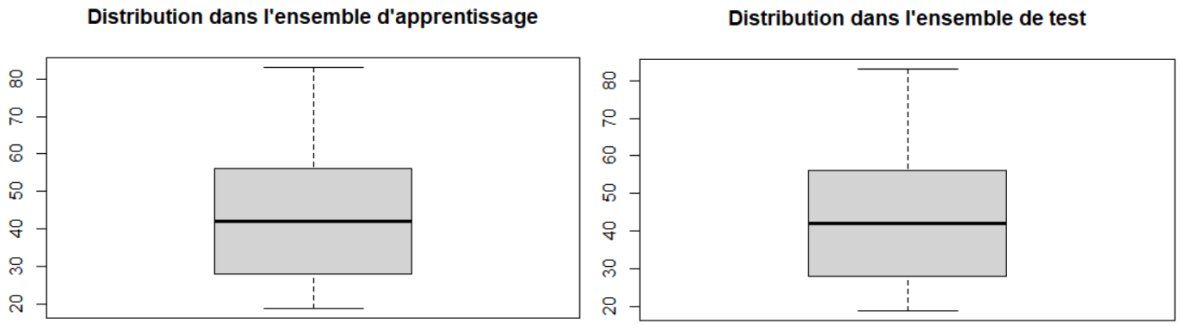
- Répartition de la population



On peut voir sur ce graphique qu'il y a en majorité des personnes entre 18 et 40 ans.

Ensuite 34% des personnes ont entre 40 et 60 ans suivi des personnes de plus de 60 ans qui représentent 19% du reste de la population.

Distribution de la variable âge

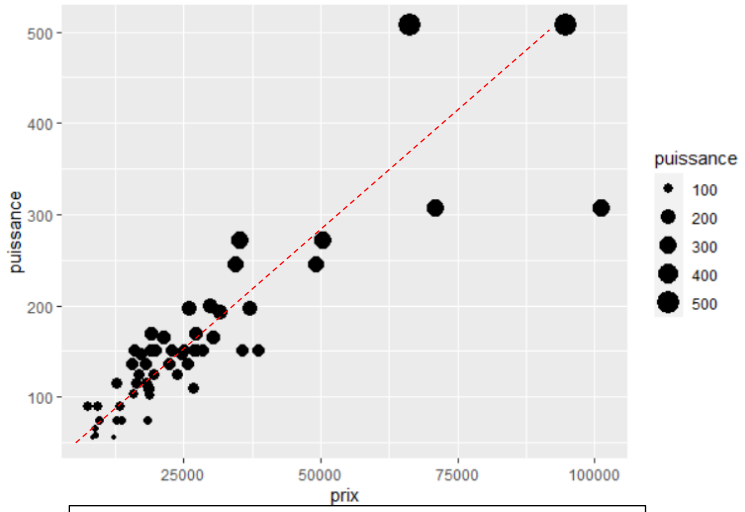


On constate aucune différence significative entre les distributions des valeurs de la variable Age entre les deux ensembles (création des ensembles expliqué dans la suite du rapport).

- Hypothèse : Plus le client est âgé, plus il a un taux d'endettement élevé ?

Conclusion : Faux, pas de corrélation entre l'âge et le taux car coefficient de corrélation proche de 0 (-0.004 : obtenu par cor(clients$age, clients$taux, method="pearson")

- Hypothese : Plus la voiture est chère, plus sa puissance est élevée ?

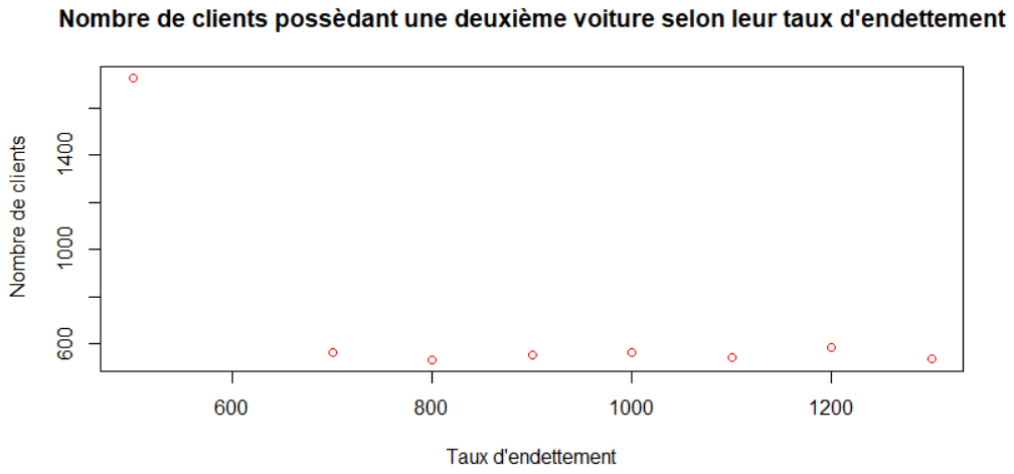


Nuage de point de prix et puissance (CV)

Conclusion : Corrélation entre le prix et la puissance de la voiture car coefficient de corrélation proche de 1 : 0.87

On peut distinguer sur le graphique une droite linéaire croissante, synonyme de corrélation entre les deux variables.

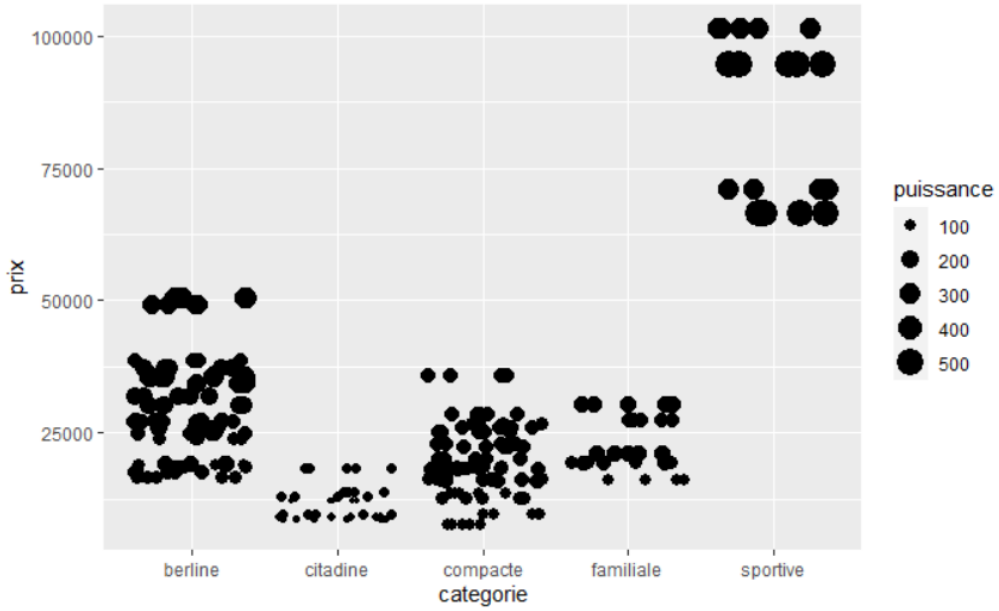
- Les clients ayant un taux d'endettement élevé sont plus susceptibles d'avoir une deuxième voiture.



Conclusion : Non, nous constatons que la majeure partie des clients possédant une deuxième voiture ont un taux d'endettement de 5xx euros.

Le nombre de personnes ayant une 2ieme voiture n'augmente pas en fonction de la valeur du taux d'endettement.

- Importance du prix dans l'attribution d'une catégorie à un véhicule



Nuage de point prix et catégorie

On peut constater sur le nuage de points que les catégories sont bien associées à un prix et à une puissance. Par exemple pour les citadines nous voyons un prix faible (<20 000€) et une puissance < 100 CV.

Quant au prix, nous voyons qu'il est difficile de déterminer un prix selon une catégorie (excepté pour les catégorie sportive) car le prix d'une catégorie peut varier fortement.

Par exemple nous voyons sur ce graphique qu'une berline peut avoir le même prix qu'une citadine, qu'une compacte ou qu'une familiale.

- Quel est le type de véhicule qui est acheté en majorité, lors de l'achat d'une deuxième voiture ?

```
clients_categorie%>%
  group_by(categorie, X2eme.voiture)%>%
  summarise(nb = n()) %>%
  arrange(desc(nb))
```

categorie	X2eme.voiture	nb
<chr>	<chr>	<int>
berline	false	12207
sportive	false	9529
citadine	false	8406
citadine	true	3253
compacte	false	3040
sportive	true	1348
berline	true	285
compacte	true	210

Les personnes possédant une deuxième voiture optent principalement pour une sportive ou une citadine. On peut alors trouver ces résultats logiques dans la mesure ou une personne achète comme deuxième voiture une citadine pour son côté pratique et les déplacements court et quotidien. Ainsi que pour l'achat d'une sportive qui peut s'apparenter à un achat plaisir pour des sorties occasionnelles.

Construction de l'ensemble d'apprentissage et de l'ensemble de test

Affectation des catégories de véhicule

Les catégories des véhicules sont déterminées à partir du modèle du véhicule (marque + nom) du fichier Catalogue.csv. Pour chaque véhicule, une recherche sur largus.fr été effectué pour lui affecter sa catégorie.

Par exemple, pour le premier modèle du fichier Catalogue.csv, le modèle S80 T6 de chez la marque Volvo, nous trouvons sur largus sa catégorie : berline

	nom	puissance	longueur	nbPlaces	nbPortes	couleur	occasion	prix
Volvo	S80 T6	272	très longue	5	5	blanc	false	50500

Véhicule du fichier Catalogue.csv

https://www.largus.fr/fiche-technique/Volvo/S80/I/2005/Berline+4+Portes/T6+272+Executive+Gtro-754003.html

Lien de la fiche technique et catégorie du véhicule sélectionné

Identification des catégories

Une fois les catégories affectées, une analyse a été faite afin de repérer le lien entre les critères de la voiture et sa catégorie :

On a donc identifié les catégories de véhicule selon les critères suivant :

- Les citadines : puissance inférieure à 90 CV OU longueur : courte
- Les compactes : puissance supérieure à 90 CV OU longueur : courte ou moyenne
- Les sportives : puissance supérieure à 300 CV
- Les familiales : nombres de places > 5 ET longueur : longue, très longue
- Les berlines : puissance : entre 100 et 300 CV OU longueur : longue, très longue

Le prix n'est pas entré dans les critères car en fonction des marques haut de gamme (exemple Mercedes), le prix peut varier sans avoir forcément de corrélation avec sa catégorie.

Par exemple, le prix moyen d'une compacte toutes marques confondues, peut être équivalent à un prix d'une citadine chez Mercedes (marque haut de gamme).

Création des modèles de classification supervisée pour la prédiction de la catégorie de véhicules

Remarque : Dans le jeu de données Immatriculations, aucun véhicule de type familiale n'est présent car aucun des véhicules n'a un nombre de places > 5.

```
#0 ligne selectionné car pas de voiture avec nbPlaces > 5 dans immatriculations
count(filter(clients_categorie, categorie == "familiale"))
```

Création des ensembles d'apprentissages et des ensembles de tests.

Afin de construire nos ensembles d'apprentissages et de tests nous avons fait une jointure des données immatriculations avec les données clients selon leur identifiant commun, l'**immatriculation.**

```
> str(immatriculations)
'data.frame':   1048575 obs. of  11 variables:
 $ immatriculation: chr  "3176 TS 67" "3721 QS 49" "9099 UV 26" "3563 LA 55" ...
 $ marque         : chr  "Renault" "Volvo" "Volkswagen" "Peugeot" ...
 $ nom            : chr  "Laguna 2.0T" "S80 T6" "Golf 2.0 FSI" "1007 1.4" ...
 $ puissance      : int  170 272 150 75 75 193 135 136 150 150 ...
 $ longueur       : chr  "longue" "très longue" "moyenne" "courte" ...
 $ 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        : chr  "blanc" "noir" "gris" "blanc" ...
 $ occasion       : chr  "false" "false" "true" "true" ...
 $ prix           : int  27300 50500 16029 9625 18310 31790 22350 18130 25060 27020 ...
 $ categorie      : chr  "berline" "berline" "compacte" "citadine" ...
> str(clients)
'data.frame':   42368 obs. of  7 variables:
 $ age              : int  58 27 73 46 34 24 23 45 53 82 ...
 $ sexe             : chr  "M" "M" "M" "M" ...
 $ taux             : int  1149 587 547 1223 1326 564 1362 1199 1256 947 ...
 $ situationFamiliale: chr  "Célibataire" "Célibataire" "En Couple" "Célibataire" ...
 $ nbEnfantsAcharge : int  0 0 2 0 3 4 0 2 0 0 ...
 $ X2eme.voiture    : chr  "false" "false" "false" "false" ...
 $ immatriculation  : chr  "6963 AX 34" "Undefined" "4487 DR 75" "9626 HF 36" ...
```

Afin d'effectuer la jointure entre les 2 ensembles de données, nous utilisons la librairie dplyr. Cette librarie permet de manipuler des data frames facilement (faire des select, join) comme SQL.

```
#-------------------------------------------------#
# 4 Fusion des données Clients et Immatriculation #
#-------------------------------------------------#
#Selection des colonnes de la table immatratriculation, utiles à la jointure : immatriculation et categorie
immatriculations_join <- select(immatriculations, immatriculation, categorie)
#Jointure des tables immatriculations_join et clients par la colonne commune immatriculation
clients_categorie <- inner_join(clients,immatriculations_join, by="immatriculation")
str(clients_categorie$categorie)
```

Apres nettoyage et jointure des données, nous obtenons un ensemble de données de 38 278 lignes, nous avons constitué un ensemble d'apprentissage à 2/3 de l'ensemble de données, à savoir les 25 519 premières lignes. Puis les 1/3 restant pour la création de notre ensemble de test, de 25 520 à 38278.

Les colonnes immatriculations faisant office d'id pour les clients est supprimé des ensembles de test, car inutile à la conception de nos algorithmes prédictifs.

```
# Creation des ensembles d'apprentissage et de test
clients_categorie_EA <- clients_categorie[1:25519,]
clients_categorie_ET <- clients_categorie[25520:38278,]

clients_categorie_EA <- subset(clients_categorie_EA, select = -immatriculation)
clients_categorie_ET <- subset(clients_categorie_ET, select = -immatriculation)
```

Génération et interprétation des modèles de connaissances

Génération des modèles de connaissance et problèmes rencontrés

Pour la création des mesures d'évaluation des différents algorithmes nous avons rencontrés un problème majeur.

La liste des prédictions pour les différents algorithmes est obtenue à l'aide de la fonction prediction().

Cependant, cette fonction n'accepte que des liste binaire (0 : la probabilité que la classe positive est prédite, 1 : la probabilité que la classe positive n'est pas prédite.

Or, dans notre cas, nous avons plusieurs classes positives : berline, citadine, compacte et sportive.

Nous obtenons donc une erreur nous indiquant que le nombre de classes n'est pas égal à 2.

```
Error in prediction(predictions$predict, clients_categorie_ET$categorie) :
  Number of classes is not equal to 2.
ROCR currently supports only evaluation of binary classification tasks.
```

Pour obtenir les prédictions de chacune de ces classes, il faut alors transformer notre liste de prédiction pour qu'elle soit d'une forme binaire pour chacune des classes.

Nous aurons alors 4 listes pour les 4 classes :

Liste 1 pour la classe berline composé des colonnes berline et non-berline (obtenue en faisant la somme des probabilités que le véhicule soit une citadine, une compacte et une sportive).

Même raisonnement pour les 3 prochaines classes citadine, compacte et sportive.

Interprétation des modèles de connaissances

Chacun des algorithmes créés a été testé avec différents paramètres afin de trouver les paramètres permettant de sortir les résultats les plus pertinent.

Afin de choisir l'algorithme le plus performant, nous avons fait 2 sélections :

- La première qui permet de choisir, parmi les différents paramètres pour un algorithme, lesquels des paramètres offrent les meilleurs résultats.

- La deuxième sélection qui consiste à choisir le meilleur algorithme parmi tous les algorithmes implémentés (arbre de décision, random forest, naibes bayes et k-nearest neighbors)

Ces sélections sont faites en comparant les résultats des matrices de confusion et des mesures d'évaluations obtenues (courbe ROC et indicateur AUC).

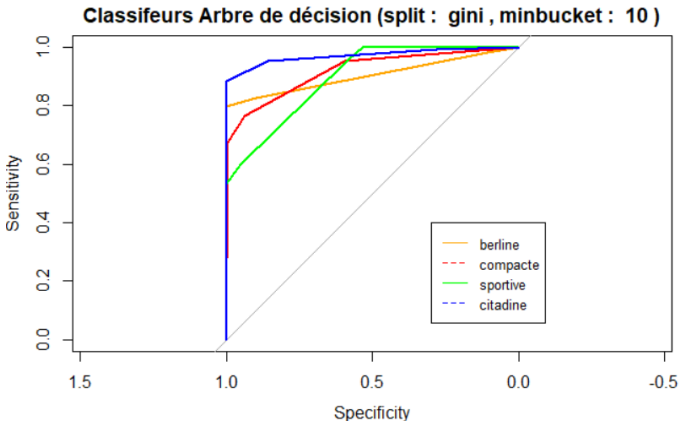
Nous allons voir en détail, pour le premier algorithme, la première sélection puis il vous sera présenté pendant la description des autres algorithmes, directement la meilleure solution retenue pour chacun des algorithmes.

Arbre de décision

a. Paramétrage split : gini et minbucket : 10

Pour l'algorithme d'arbre de décision nous obtenons la matrice de confusion suivante, ainsi que ces courbes ROC et indicateurs AUC :

```
Matrice de confusion :
          dt_class
           berline citadine compacte sportive
  berline     4234        3        1        0
  citadine       3     3270      542        0
  compacte       2      417      615        1
  sportive    1709        0        0     1962
```




```
 ------------ berline ----------------
AUC berline  =  0.903254840676498
 -------------------------------------

 ------------ citadine ----------------
AUC citadine  =  0.971651760070525
 -------------------------------------

 ------------ compacte ----------------
AUC compacte  =  0.919771285459283
 -------------------------------------

 ------------ sportive ----------------
AUC sportive  =  0.104160601891107
 -------------------------------------
```

Nous pouvons voir sur cette matrice de confusion que :

- La classe berline est bien classée avec un taux de vrais positifs (TVP) de quasi 100%
- La classe citadine est correctement classée avec un TVP de 86%
- La classe compacte est moyennement bien classée avec un TVP de 59%

- La classe sportive est moyennement bien classée avec un TVP de 53%

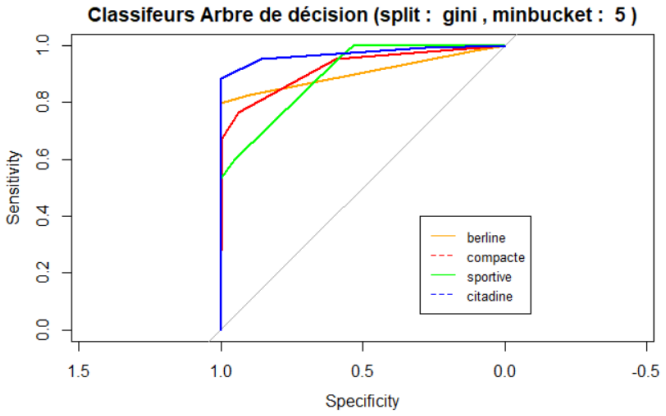
Nous obtenons de bonne valeur d'AUC pour les classes berline, citadine et compacte avec des valeurs supérieures à 0.90 et donc proche de 1 qui est le chiffre d'une discrimination optimale.

Cependant nous pouvons voir que l'AUC de la classe sportive et très bas, à 0.10. Nous voyons sur la courbe ROC que la classe sportive a une difficulté à monter, ce qui est dû à un nombre important de classe sportive incorrectement classé (cf matrice de confusion).

On peut aussi remarquer une enveloppe convexe pour la classe citadine (en bleue).

b. Paramétrage split : gini et minbucket : 10

```
Matrice de confusion :
          dt_class
           berline citadine compacte sportive
  berline     4234        3        1        0
  citadine       3     3270      542        0
  compacte       2      417      615        1
  sportive    1709        0        0     1962
```

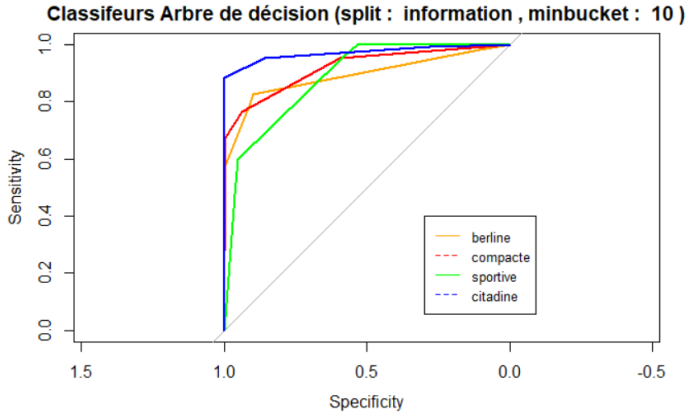



```
 ------------ berline ----------------
AUC berline  =  0.903254840676498
 -------------------------------------

 ------------ citadine ----------------
AUC citadine  =  0.971651760070525
 -------------------------------------

 ------------ compacte ----------------
AUC compacte  =  0.919771285459283
 -------------------------------------

 ------------ sportive ----------------
AUC sportive  =  0.104160601891107
 -------------------------------------
```

Résultat semblable au paramétrage précédent.

c. Paramétrage split : information et minbucket : 10

```
Matrice de confusion :
          dt_class
           berline citadine compacte sportive
  berline     3802        3        1      432
  citadine       3     3270      542        0
  compacte       2      417      615        1
  sportive    1470        0        0     2201
```




```
------------ berline ----------------
AUC berline  =  0.891110345099155
-------------------------------------

------------ citadine ----------------
AUC citadine  =  0.971480225289965
-------------------------------------

------------ compacte ----------------
AUC compacte  =  0.919771285459283
-------------------------------------

------------ sportive ----------------
AUC sportive  =  0.117066404316665
```

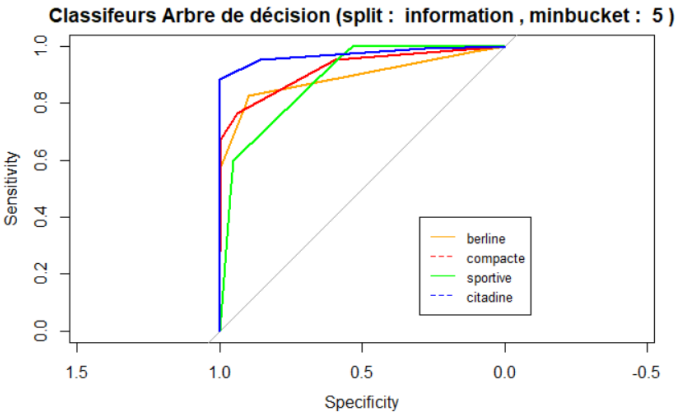
Nous pouvons voir ici une différence avec le TVP des classes berline et sportive.

Avec un TVP pour la classe berline qui baisse à 90% (contre 100%) et un TVP qui augmente pour la classe sportive à 60%.

Ces changements sont reflétés sur les indicateurs AUC où nous voyons une diminution de l'AUC pour la classe berline (de 0.903 à 0.891) et une augmentation de l'AUC pour la classe sportive (de 0.104 à 0.117)

d. Paramétrage split : information et minbucket : 5

```
Matrice de confusion :
          dt_class
           berline citadine compacte sportive
  berline     3802        3        1      432
  citadine       3     3270      542        0
  compacte       2      417      615        1
  sportive    1470        0        0     2201
```



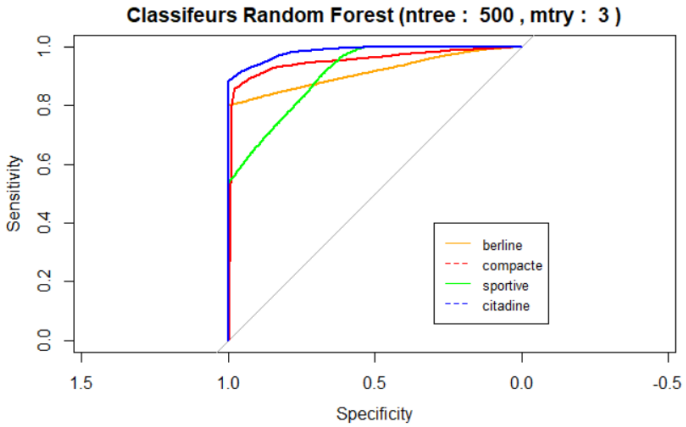

```
------------ berline ----------------
AUC berline  =  0.891110345099155
-------------------------------------

------------ citadine ----------------
AUC citadine  =  0.971480225289965
-------------------------------------

------------ compacte ----------------
AUC compacte  =  0.919771285459283
-------------------------------------

------------ sportive ----------------
AUC sportive  =  0.117066404316665
-------------------------------------
```

Résultat semblable au paramétrage précédent.

Random Forest

Pour l'algorithme du Random Forest nous obtenons la matrice de confusion suivante, ainsi que ces courbes ROC et indicateurs AUC :

```
Matrice de confusion :
          rf_class
           berline citadine compacte sportive
  berline     4120        4        0      114
  citadine       3     3439      373        0
  compacte       2      564      468        1
  sportive    1649        0        0     2022
```




```
 ------------ berline ----------------
AUC berline  =  0.913493570751748
 -------------------------------------

 ------------ citadine ----------------
AUC citadine  =  0.981329422391136
 -------------------------------------

 ------------ compacte ----------------
AUC compacte  =  0.947376206699334
 -------------------------------------

 ------------ sportive ----------------
AUC sportive  =  0.0932310720253145
 -------------------------------------
```

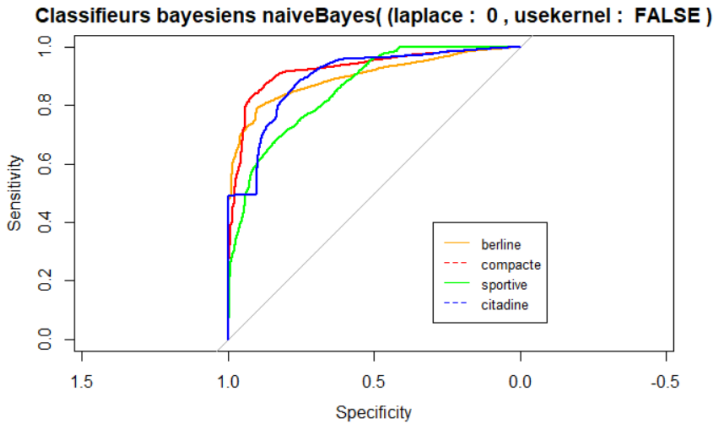
Cet algorithme offre de meilleur résultat pour les classes berline, citadine et compacte que l'algorithme d'arbre de décision. Excepté pour la classe sportive avec une légère différence pour l'indicateur AUC.

La classe citadine forme toujours une enveloppe convexe.

Naive Bayes

Pour l'algorithme Naive Bayes nous obtenons la matrice de confusion suivante, ainsi que ces courbes ROC et indicateurs AUC :

```
Matrice de confusion :
          nb_class
           berline citadine compacte sportive
  berline     3095      440        1      702
  citadine       3     2681      746      385
  compacte       2      370      646       17
  sportive    1160      205        0     2306
```




```
 ------------ berline ----------------
AUC berline  =  0.896209176794929
 -------------------------------------

 ------------ citadine ----------------
AUC citadine  =  0.893302787462161
 -------------------------------------

 ------------ compacte ----------------
AUC compacte  =  0.92351409306151
 -------------------------------------

 ------------ sportive ----------------
AUC sportive  =  0.14402156905955
 -------------------------------------
```

L'algorithme de Naive Bayes nous offre les moins bons résultats.

Pour la classe berline, nous obtenons pour le meilleur paramétrage, un TVP de 73%. Avec étonnamment des classes citadines prédites alors que ces deux catégories de véhicule sont très différentes, par la puissance et par leur taille.

Même remarque pour la classe citadine qui est pour les deux algorithmes précédents mieux prédites.

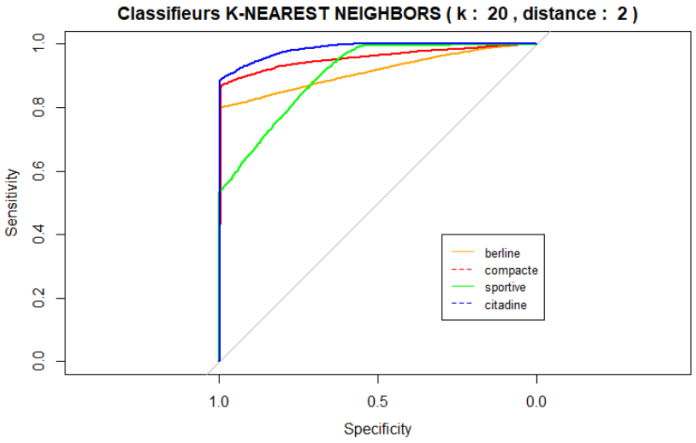
On remarque que la classe citadine ne forme plus une enveloppe convexe.

K-nearest neighbors

Pour l'algorithme K-nearest neighbors nous obtenons la matrice de confusion suivante, ainsi que ces courbes ROC et indicateurs AUC :

Matrice de confusion :

	berline	citadine	compacte	sportive
berline	3577	5	0	656
citadine	3	3380	432	0
compacte	2	531	501	1
sportive	1385	0	0	2286




```
 ------------ berline ----------------
AUC berline  =  0.911421378567865
 ---------------------------------------

 ------------ citadine ----------------
AUC citadine  =  0.982482702916882
 ---------------------------------------

 ------------ compacte ----------------
AUC compacte  =  0.947951310083614
 ---------------------------------------

 ------------ sportive ----------------
AUC sportive  =  0.100572812556351
 ---------------------------------------
```

Cet algorithme à des résultats similaire à celui du Random Forest et se classe donc parmi les 2 algorithmes les plus performant.

Nous avons donc, à l'issue de l'analyse des différents algorithmes, opté pour l'algorithme Random Forest avec le paramétrage ntry : 500 et mtry : 3.

Résultats obtenus sur les clients sélectionnés par le service marketing

Résultats obtenus par l'utilisation de l'algorithme le plus performant, sélectionné précédemment : Random Forest de paramètres : ntry : 500 et mtry : 3

	age	sexe	taux	situationFamiliale	nbEnfantsAcharge	X2eme.voiture	predict_categorie
1	21	F	1396	Célibataire	0	false	citadine
2	35	M	223	Célibataire	0	false	compacte
3	48	M	401	Célibataire	0	false	citadine
4	26	F	420	En Couple	3	true	berline
5	80	M	530	En Couple	3	false	sportive
6	27	F	153	En Couple	2	false	berline
7	59	F	572	En Couple	2	false	berline
8	43	F	431	Célibataire	0	false	compacte
9	64	M	559	Célibataire	0	false	citadine
10	22	M	154	En Couple	1	false	berline
11	79	F	981	En Couple	2	false	berline
12	55	M	588	Célibataire	0	false	compacte
13	19	F	212	Célibataire	0	false	compacte
14	34	F	1112	En Couple	0	false	berline
15	60	M	524	En Couple	0	true	citadine
16	22	M	411	En Couple	3	true	berline
17	58	M	1192	En Couple	0	false	berline
18	54	F	452	En Couple	3	true	berline
19	35	M	589	Célibataire	0	false	compacte
20	59	M	748	En Couple	0	true	citadine