ESTIA/MBDS

Projet Big Data Analytics

Victor AZALBERT / Jérôme CHAMBORD / Justin LASSALLE

03/03/2021

Lien Vers le dépôt GitHub Du Projet : https://github.com/zazacito/Projet3AMBDS

Table des matières

Objectifs

Nous allons dans un premier temps charger une partie des données dans Oracle NoSQL, puis une autre partie en utilisant SQL Loader et enfin une dernière partie sur HDFS. Avec HIVE, nous allons créer des tables externes qui pointent vers l'ensemble des données et pour finir nous allons créer et récupérer des tables via SQL Plus.

Pour l'analyse des données, nous allons commencer par les nettoyer et préparer les données ensuite nous allons effectuer des analyses exploratoires des données pour finir par tester et choisir un classifieur.

Nous allons aussi réaliser une fonction map/reduce afin de fusionner les données d'un fichier CO2 aux données du fichier catalogue.

1. Architecture et Stockage Des Données

1.1. Chargement des données sur MongoDB

Nous avons choisi de sauvegarder le fichier immatriculations sur MongoDB. Nous importons donc le fichier csv sur Mongo avec la commande suivante :

```
mongoimport -d concessionnaire -c immatriculations --type csv --file "C:/Immatriculations.csv" --headerline
```

Puis nous exportons les données de la collection immatriculations dans un fichier JSON afin de pouvoir l'importer sur Hive.
L'export se fait avec cette commande :

```
mongoexport -d concessionnaire -c immatriculations -o resultExport.json
```

Voici un extrait du fichier resultExport.json :

Finalement cette partie n'est plus nécessaire dans le sujet. Nous chargerons donc le fichier immatriculations sur la base OracleNoSql.

1.2. Chargement des données sur Oracle NoSQL

1.2.1. Fichier Marketing

Nous avons décidé de charger les données du fichier marketing sur la base de données oracle kv.

Nous avons codé un fichier DataImportMarketing.java qui permet de parcourir le fichier marketing.csv et de l'ajouter à notre base oracle noSql.

Voici la boucle qui permet cet ajout. Elle itère sur le fichier csv , afin d'extraire les données brutes, puis appelle la fonction insertAMarketingRow qui permet d'ajouter une ligne à notre table marketing créé préalablement.

```
while ((ligne = br.readLine()) != null) {
    //int situationFamiliale, 2eme voiture, nbPortes, prix;
    //String marketing, age, sexe,  nbEnfantsAcharge,  couleur, occasion, ;
    ArrayList<String> marketingRecord = new ArrayList<String>();
    StringTokenizer val = new StringTokenizer(ligne, ",");
    while (val.hasMoreTokens()) {
        marketingRecord.add(val.nextToken().toString());
    }
    String age = marketingRecord.get(0);
    String sexe = marketingRecord.get(1);
    String taux = marketingRecord.get(2);
    String situationFamiliale = marketingRecord.get(3);
    String nbEnfantsAcharge = marketingRecord.get(4);
    String deuxiemeVoiture = marketingRecord.get(5);
    // Add the marketing in the KVStore
    this.insertAmarketingRow(age, sexe, taux, situationFamiliale, nbEnfantsAcharge, deuxiemeVoiture);
```

Voici un extrait de la fonction insertAMarketingRow.

```
// Create one row
marketingRow.put("clientMarketingID", clientID);
marketingRow.put("age", age);
marketingRow.put("sexe", sexe);
marketingRow.put("taux", taux);
marketingRow.put("situationFamiliale", situationFamiliale);
marketingRow.put("nbEnfantsAcharge", nbEnfantsAcharge);
marketingRow.put("deuxiemeVoiture", deuxiemeVoiture);
// Now write the table to the store.
// "item" is the row's primary key. If we had not set that value,
// this operation will throw an IllegalArgumentException.
tableH.put(marketingRow, null, null);
clientID++;
```

Dans le terminal, nous exécutons les commandes suivantes :

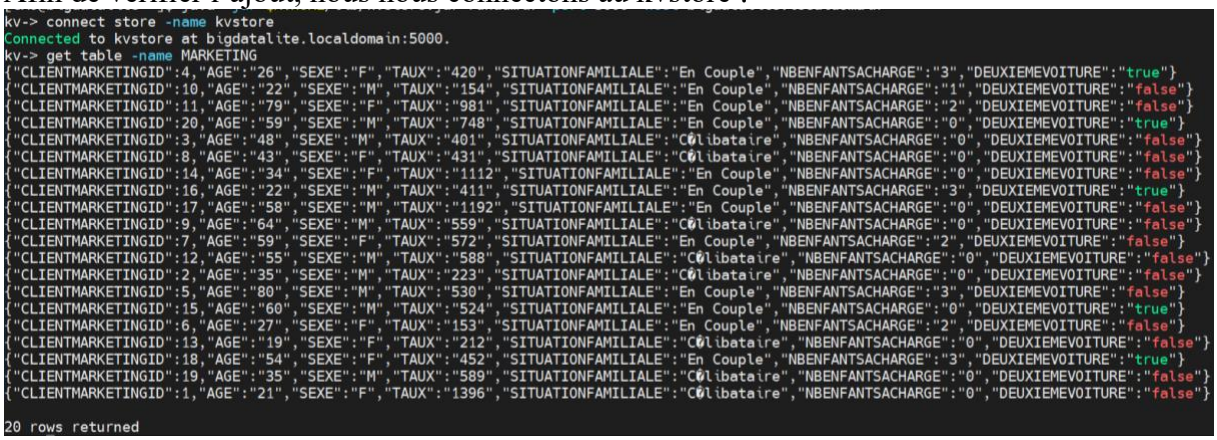
```
-- Ceci est le chemin vers notre projet sur la machine virtuelle
$ export MYPROJECTHOME=/home/AZALBERT/projetMBDS/

-- Compiler le code java
$ javac -g -cp $KVHOME/lib/kvclient.jar:$MYPROJECTHOME/ $MYPROJECTHOME/voiture/DataImportMarketing.java

-- Executer le code java pour importer la table MARKETING à partir du fichier csv
$ java -Xmx256m -Xms256m  -cp $KVHOME/lib/kvclient.jar:$MYPROJECTHOME/ voiture.DataImportMarketing
```

Après compilation, le fichier java est exécuté.

Afin de vérifier l'ajout, nous nous connectons au kvstore :



Les données ont bien été ajoutées.

1.2.2. Fichier Immatriculations

Nous avons reproduit la même procédure pour le fichier Immatriculations.
Afin de vérifier l'ajout, nous nous connectons au kvstore :

Les données ont bien été ajoutées.

1.3. Chargements des Données via SQL Loader

Pour ajouter les données dans SQL loader, nous créons au préalable une table client dans SQL PLUS comme suit.

```
sqlplus AZALBERTBZ2021@ORCL/AZALBERTZ202101

DROP TABLE CLIENT;

CREATE TABLE CLIENT_8(
    AGE varchar2(30),
    SEXE varchar2(30),
    TAUX varchar2(30),
    SITUATIONFAMILIALE varchar2(30),
    NBENFANTSACHARGE varchar2(30),
    XVOITURE varchar2(30),
    IMMATRICULATION varchar2(30)
    );
```

Puis nous avons créé un fichier .ctl qui permet la population de la table client depuis le fichier client.csv.

```
LOAD DATA
INFILE '$MYPROJECTHOME/projetMBDS/SQLLOADER/Client.csv'
INSERT INTO TABLE client
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
TRAILING NULLCOLS
(
AGE,
SEXE,
TAUX,
SITUATIONFAMILIALE,
NBENFANTSACHARGE,
XVOITURE,
IMMATRICULATIONS
)
```

Que nous appelons comme suit.

```
--Table : client
sqlldr AZALBERTBZ2021@ORCL/AZALBERTBZ202101 control=$MYPROJECTHOME/projetMBDS/SQLLOADER/control_clients.ctl
log=$MYPROJECTHOME/projetMBDS/SQLLOADER/track_clients.log skip=1
```

Puis nous vérifions que les données ont bien été ajoutées dans sqlPlus.

```
SQL> select * from client_8 fetch first 3 rows only;

AGE                            SEXE
------------------------------ ------------------------------
TAUX                           SITUATIONFAMILIALE
------------------------------ ------------------------------
NBENFANTSACHARGE               XVOITURE
------------------------------ ------------------------------
IMMATRICULATION
------------------------------
20                             M
1010                           En Couple
4                              false
7396 VP 43


AGE                            SEXE
------------------------------ ------------------------------
TAUX                           SITUATIONFAMILIALE
------------------------------ ------------------------------
NBENFANTSACHARGE               XVOITURE
------------------------------ ------------------------------
IMMATRICULATION
------------------------------
59                             F
422                            En Couple
4                              false
7546 VN 65


AGE                            SEXE
------------------------------ ------------------------------
TAUX                           SITUATIONFAMILIALE
------------------------------ ------------------------------
NBENFANTSACHARGE               XVOITURE
------------------------------ ------------------------------
IMMATRICULATION
------------------------------
21                             M
207                            En Couple
0                              false
5235 IZ 58
```

1.4. Chargement des Données via HDFS

Nous ajoutons le fichier catalogue.csv sur HDFS en utilisant la commande hadoop fs -put.

```
hadoop fs -rm -r /user/zazacito

hadoop fs -mkdir /user/zazacito
hadoop fs -mkdir /user/zazacito/projetMBDS
hadoop fs -mkdir /user/zazacito/projetMBDS/Catalogue

hadoop fs -put $MYPROJECTHOME/projetMBDS/Catalogue.csv /user/zazacito/projetMBDS/Catalogue

hadoop fs -ls /user/zazacito/projetMBDS/Catalogue
```

1.5. Tables externes sur Hive

Nous devons charger les fichiers catalogue, marketing et immatriculations sur Hive. Nous devons créer des tables externes Hive qui pointent respectivement vers :

- Le kvstore pour les tables immatriculations et marketing
- HDFS pour la table catalogue

1.5.1. Depuis le KvStore

Nous allons vous présenter le processus de création de tables externes depuis le kvstore et de chargement pour la table marketing.

Premièrement, il faut se connecter à hive.

```
beeline> !connect jdbc:hive2://localhost:10000

Enter username for jdbc:hive2://localhost:10000: oracle
Enter password for jdbc:hive2://localhost:10000: ********
(password : welcome1)
```

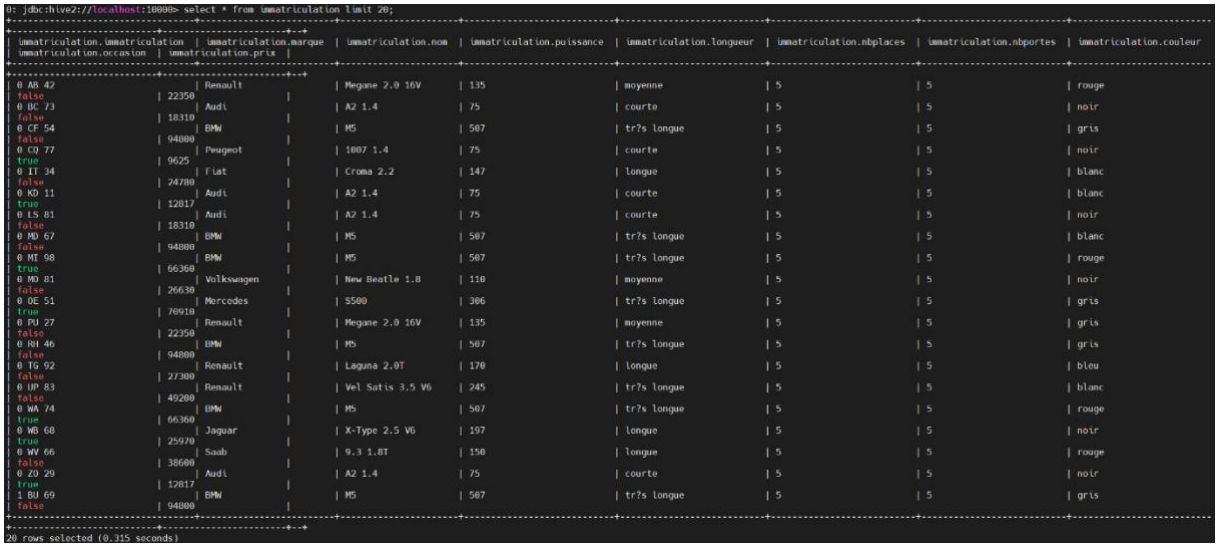
Puis nous créons une table externe qui pointe vers la table marketing stockée sur le kvstore.

```
jdbc:hive2://localhost:10000> CREATE EXTERNAL TABLE MARKETING(
    CLIENTMARKETINGID int,
    AGE string ,
    SEXE string,
    TAUX string,
    SITUATIONFAMILIALE string,
    NBENFANTSACHARGE string,
    DEUXIEMEVOITURE string
)
STORED BY 'oracle.kv.hadoop.hive.table.TableStorageHandler'
TBLPROPERTIES (
"oracle.kv.kvstore" = "kvstore",
"oracle.kv.hosts" = "bigdatalite.localdomain:5000",
"oracle.kv.hadoop.hosts" = "bigdatalite.localdomain/127.0.0.1",
"oracle.kv.tableName" = "MARKETING");
```

Enfin nous vérifions que table a bien été créé et stockée.

```
0: jdbc:hive2://localhost:10000> select * from MARKETING;
+-----------------------------+---------------------+---------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
| marketing.clientmarketingid | marketing.age       | marketing.sexe      | marketing.taux      | marketing.situationfamiliale  | marketing.nbenfantsacharge    | marketing.deuxiemevoiture     |
+-----------------------------+---------------------+---------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
| 1                           | 21                  | F                   | 1396                | C?libataire                   | 0                             | false                         |
| 6                           | 27                  | F                   | 153                 | En Couple                     | 2                             | false                         |
| 18                          | 54                  | F                   | 452                 | En Couple                     | 3                             | true                          |
| 19                          | 35                  | M                   | 589                 | C?libataire                   | 0                             | false                         |
| 20                          | 59                  | M                   | 748                 | En Couple                     | 0                             | true                          |
| 7                           | 59                  | F                   | 572                 | En Couple                     | 2                             | false                         |
| 11                          | 79                  | F                   | 981                 | En Couple                     | 2                             | false                         |
| 10                          | 22                  | M                   | 154                 | En Couple                     | 1                             | false                         |
| 2                           | 35                  | M                   | 223                 | C?libataire                   | 0                             | false                         |
| 3                           | 48                  | M                   | 401                 | C?libataire                   | 0                             | false                         |
| 16                          | 22                  | M                   | 411                 | En Couple                     | 3                             | true                          |
| 5                           | 80                  | M                   | 530                 | En Couple                     | 3                             | false                         |
| 15                          | 60                  | M                   | 524                 | En Couple                     | 0                             | true                          |
| 17                          | 58                  | M                   | 1192                | En Couple                     | 0                             | false                         |
| 4                           | 26                  | F                   | 420                 | En Couple                     | 3                             | true                          |
| 12                          | 55                  | M                   | 588                 | C?libataire                   | 0                             | false                         |
| 9                           | 64                  | M                   | 559                 | C?libataire                   | 0                             | false                         |
| 14                          | 34                  | F                   | 1112                | En Couple                     | 0                             | false                         |
| 8                           | 43                  | F                   | 431                 | C?libataire                   | 0                             | false                         |
| 13                          | 19                  | F                   | 212                 | C?libataire                   | 0                             | false                         |
+-----------------------------+---------------------+---------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
20 rows selected (0.219 seconds)
```

Nous reproduisons exactement le même processus pour la table immatriculations. La table externe immatriculations a bien été créé :



0: jdbc:hive2://localhost:10000> select * from immatriculation limit 20;

immatriculation.immatriculation	immatriculation.marque	immatriculation.nom	immatriculation.puissance	immatriculation.longueur	immatriculation.nbplaces	immatriculation.nbportes	immatriculation.couleur	immatriculation.occasion	immatriculation.prix
0 AR 42	Renault	Megane 2.0 16V	135	moyenne	5	5	rouge	false	22350
0 BC 73	Audi	A2 1.4	75	courte	5	5	noir	false	18310
0 CF 54	BMW	M5	507	tr?s longue	5	5	gris	false	94000
0 CQ 77	Peugeot	1007 1.4	75	courte	5	5	noir	true	9625
0 IT 34	Fiat	Croma 2.2	147	longue	5	5	blanc	false	24700
0 KD 11	Audi	A2 1.4	75	courte	5	5	blanc	true	12017
0 LS 81	Audi	A2 1.4	75	courte	5	5	noir	false	18310
0 MD 67	BMW	M5	507	tr?s longue	5	5	blanc	false	94000
0 MI 98	BMW	M5	507	tr?s longue	5	5	rouge	true	66360
0 MO 81	Volkswagen	New Beatle 1.8	110	moyenne	5	5	noir	false	26630
0 OE 51	Mercedes	S500	306	tr?s longue	5	5	gris	true	70910
0 PU 27	Renault	Megane 2.0 16V	135	moyenne	5	5	gris	false	22350
0 RH 46	BMW	M5	507	tr?s longue	5	5	gris	false	94000
0 TG 92	Renault	Laguna 2.0T	170	longue	5	5	bleu	false	27300
0 UP 83	Renault	Vel Satis 3.5 V6	245	tr?s longue	5	5	blanc	false	49200
0 WA 74	BMW	M5	507	tr?s longue	5	5	rouge	true	66360
0 WB 68	Jaguar	X-Type 2.5 V6	197	longue	5	5	noir	true	25970
0 WV 66	Saab	9.3 1.8T	150	longue	5	5	rouge	false	38600
0 ZO 29	Audi	A2 1.4	75	courte	5	5	noir	true	12017
1 BU 69	BMW	M5	507	tr?s longue	5	5	gris	false	94000

20 rows selected (0.315 seconds)

1.5.2. Depuis HDFS

Afin d'importer les données depuis HDFS, nous créons une table externe dans hive qui pointe vers le fichier catalogue.csv stocké sur HDFS.

```
-- Création de la table externe catalogue pointant vers le fichier catalogue dans hadoop
jdbc
:hive2://localhost:10000>
CREATE EXTERNAL TABLE CATALOGUE (
    MARQUE string,
    NOM string ,
    PUISSANCE string,
    LONGUEUR string,
    NBPLACES string,
    NBPORTES string,
    COULEUR string,
    OCCASION string,
    PRIX string
) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION 'hdfs:/user/AZALBERT/projetMBDS/Catalogue';
```

Puis nous vérifions que les données ont bien été importées dans la table externe. On remarque que la première ligne contient l'en tête du fichier csv. Toutes ces données seront nettoyées à l'aide de R après importation des dataframes.

```
0: jdbc:hive2://localhost:10000> select * from CATALOGUE LIMIT 20;
+--------------------+-----------------+----------------------+---------------------+---------------------+---------------------+--------------------+---------------------+-----------------+--+
| catalogue.marque   | catalogue.nom   | catalogue.puissance  | catalogue.longueur  | catalogue.nbplaces  | catalogue.nbportes  | catalogue.couleur  | catalogue.occasion  | catalogue.prix  |
+--------------------+-----------------+----------------------+---------------------+---------------------+---------------------+--------------------+---------------------+-----------------+--+
| marque             | nom             | puissance            | longueur            | nbPlaces            | nbPortes            | couleur            | occasion            | prix            |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | blanc              | false               | 50500           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | noir               | false               | 50500           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | rouge              | false               | 50500           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | gris               | true                | 35350           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | bleu               | true                | 35350           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | gris               | false               | 50500           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | bleu               | false               | 50500           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | rouge              | true                | 35350           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | blanc              | true                | 35350           |
| Volvo              | S80 T6          | 272                  | très longue         | 5                   | 5                   | noir               | true                | 35350           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | rouge              | false               | 27340           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | gris               | true                | 19138           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | bleu               | true                | 19138           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | gris               | false               | 27340           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | bleu               | false               | 27340           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | blanc              | true                | 19138           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | noir               | true                | 19138           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | rouge              | true                | 19138           |
| Volkswagen         | Touran 2.0 FSI  | 150                  | longue              | 7                   | 5                   | blanc              | false               | 27340           |
+--------------------+-----------------+----------------------+---------------------+---------------------+---------------------+--------------------+---------------------+-----------------+--+
20 rows selected (0.068 seconds)
```

1.6. Création/Récupération des tables dans SQLPlus

Finalement, nous insérons les 3 tables crées dans HIVE (catalogue, immatriculations, marketing) dans SQL PLUS en créant des tables externes comme suit (exemple avec la table marketing).

```
-- Connexion à Hive
beeline
beeline> !connect jdbc:hive2://localhost:10000

-- Supprimer la table MARKETING si elle existe déjà
drop table MARKETING;

-- Création de la table externe MARKETING pointant vers la table MARKETING de ORACLE NOSQL (kv)
CREATE EXTERNAL TABLE MARKETING(
    CLIENTMARKETINGID int,
    AGE string ,
    SEXE string,
    TAUX string,
    SITUATIONFAMILIALE string,
    NBENFANTSACHARGE string,
    DEUXIEMEVOITURE string
)
STORED BY 'oracle.kv.hadoop.hive.table.TableStorageHandler'
TBLPROPERTIES (
"oracle.kv.kvstore" = "kvstore",
"oracle.kv.hosts" = "bigdatalite.localdomain:5000",
"oracle.kv.hadoop.hosts" = "bigdatalite.localdomain/127.0.0.1",
"oracle.kv.tableName" = "MARKETING");
```

Puis nous vérifions que l'ensemble des tables a bien été peuplé :

```
SQL> select * from catalogue offset 2 rows fetch next 3 rows only;

MARQUE                         NOM
------------------------------ ------------------------------
PUISSANCE                      LONGUEUR
------------------------------ ------------------------------
NBPLACES                       NBPORTES
------------------------------ ------------------------------
COULEUR                        OCCASION
------------------------------ ------------------------------
PRIX
------------------------------
Volvo                          S80 T6
272                            tr?longue
5                              5

MARQUE                         NOM
------------------------------ ------------------------------
PUISSANCE                      LONGUEUR
------------------------------ ------------------------------
NBPLACES                       NBPORTES
------------------------------ ------------------------------
COULEUR                        OCCASION
------------------------------ ------------------------------
PRIX
------------------------------
noir                           false
50500


MARQUE                         NOM
------------------------------ ------------------------------
PUISSANCE                      LONGUEUR
------------------------------ ------------------------------
NBPLACES                       NBPORTES
------------------------------ ------------------------------
COULEUR                        OCCASION
------------------------------ ------------------------------
PRIX
------------------------------
Volvo                          S80 T6
272                            tr?longue
5                              5

MARQUE                         NOM
------------------------------ ------------------------------
PUISSANCE                      LONGUEUR
------------------------------ ------------------------------
NBPLACES                       NBPORTES
------------------------------ ------------------------------
COULEUR                        OCCASION
------------------------------ ------------------------------
PRIX
------------------------------
rouge                          false
50500
```

```
SQL> SELECT * FROM IMMATRICULATION FETCH FIRST 2 ROWS ONLY;

IMMATRICULATION                MARQUE
------------------------------ ------------------------------
NOM                            PUISSANCE
------------------------------ ------------------------------
LONGUEUR                       NBPLACES
------------------------------ ------------------------------
NBPORTES                       COULEUR
------------------------------ ------------------------------
OCCASION                       PRIX
------------------------------ ------------------------------
0 AS 74                        BMW
120i                           150
moyenne                        5

IMMATRICULATION                MARQUE
------------------------------ ------------------------------
NOM                            PUISSANCE
------------------------------ ------------------------------
LONGUEUR                       NBPLACES
------------------------------ ------------------------------
NBPORTES                       COULEUR
------------------------------ ------------------------------
OCCASION                       PRIX
------------------------------ ------------------------------
5                              bleu
true                           25060


IMMATRICULATION                MARQUE
------------------------------ ------------------------------
NOM                            PUISSANCE
------------------------------ ------------------------------
LONGUEUR                       NBPLACES
------------------------------ ------------------------------
NBPORTES                       COULEUR
------------------------------ ------------------------------
OCCASION                       PRIX
------------------------------ ------------------------------
0 AX 73                        Ford
Mondeo 1.8                     125
longue                         5

IMMATRICULATION                MARQUE
------------------------------ ------------------------------
NOM                            PUISSANCE
------------------------------ ------------------------------
LONGUEUR                       NBPLACES
------------------------------ ------------------------------
NBPORTES                       COULEUR
------------------------------ ------------------------------
OCCASION                       PRIX
------------------------------ ------------------------------
5                              rouge
false                          23900
```

```
SQL> SELECT * FROM MARKETING FETCH FIRST 2 ROWS ONLY;

CLIENTMARKETINGID AGE                            SEXE
----------------- ------------------------------ ------------------
TAUX                           SITUATIONFAMILIALE
------------------------------ ------------------------------
NBENFANTSACHARGE               DEUXIEMEVOITURE
------------------------------ ------------------------------
                7 59                             F
572                            En Couple
2                              false

               11 79                             F
981                            En Couple
2                              false

CLIENTMARKETINGID AGE                            SEXE
----------------- ------------------------------ ------------------
TAUX                           SITUATIONFAMILIALE
------------------------------ ------------------------------
NBENFANTSACHARGE               DEUXIEMEVOITURE
------------------------------ ------------------------------
```

Nos 4 tables sont maintenant toutes stockées sur sql. Nous pouvons passer à l'analyse et au traitement des données via R.

2. Analyse Des Données

2.1. Objectifs

L'objectif est de construire un modèle de prédiction de la catégorie de véhicules (ou du modèle de véhicule) la plus susceptible de convenir à un client en fonction de ses caractéristiques (âge, sexe, statut marital, nombre d'enfants, etc.). Les principales étapes consisteront à :

- Répartir les véhicules et/ou les clients en différentes catégories correspondant chacune à différents besoins.
- Mettre au point un modèle de prédiction de la catégorie de véhicules qui répondent aux besoins des clients à l'aide des approches de classification supervisée.

2.2. Méthode Utilisée

1. Nettoyage et Préparation des Données
2. Analyse Exploratoire des Données
3. Application des catégories de véhicules définies au fichier Immatriculations.csv
4. Fusion des fichiers Clients.csv et Immatriculations.csv
5. Création d'un modèle de classification supervisée pour la prédiction de la catégorie de véhicules
6. Prédiction de la catégorie de véhicules pour le fichier marketing

2.3. Nettoyage et Préparation des données

2.3.1. Accès Aux Données

La première étape a été la connexion à notre base de données Oracle située sur la machine virtuelle, avec l'utilisation du package R RODBC.

```
install.packages("RODBC")
library (RODBC)
```

Il est nécessaire de créer un user afin de pouvoir assurer la connexion à la base de données distantes. Ci-joint la procédure nécessaire :

```
SQL> define MYDBUSER=PROJET
SQL> define MYDB=orcl
SQL> define MYDBUSERPASS=123
SQL> define MYCDBUSER=system
SQL> connect &MYCDBUSER@&MYDB/
Enter password: password1


Connected.
SQL> CREATE USER &MYDBUSER IDENTIFIED BY &MYDBUSERPASS default tablespace users temporary tablespace temp;
old   1: CREATE USER &MYDBUSER IDENTIFIED BY &MYDBUSERPASS default tablespace users temporary tablespace temp
new   1: CREATE USER PROJET IDENTIFIED BY password1 default tablespace users temporary tablespace temp

User created.

SQL> grant dba to &MYDBUSER;
old   1: grant dba to &MYDBUSER
new   1: grant dba to PROJET

Grant succeeded.

SQL> alter user &MYDBUSER quota unlimited on users;
old   1: alter user &MYDBUSER quota unlimited on users
new   1: alter user PROJET quota unlimited on users
```

Une fois l'utilisateur créé, nous ajoutons les infos correspondantes à la base de données distantes dans notre fichier tsnames.ora. Ensuite nous créons une connexion ODBC via le panneau de configuration (préférences sur Mac OS).

Puis nous importons les données grâce à la commande sqlQuery.

```
connexion <- odbcConnect("ORCLPROJETDB_DNS", uid="PROJET", pwd="password1", believeNRows=FALSE)

immatriculation<-sqlQuery(connexion, "SELECT * FROM IMMATRICULATION")
marketing<-sqlQuery(connexion, "SELECT * FROM MARKETING")
catalogue<-sqlQuery(connexion, "SELECT * FROM CATALOGUE")
clients<-sqlQuery(connexion, "SELECT * FROM CLIENTS")
```

2.3.2. Nettoyage des Données

Nous avons réalisé le nettoyage des données sous R dans RStudio.

Toutes les valeurs sont stockées en string, il faut donc les refactoriser. De plus on retire toutes les lignes où des valeurs vides sont présentes.

Pour la table client :

```
clients$age <- as.integer(clients$age)
clients$taux <- as.integer(clients$taux)
clients$situationFamiliale <- as.factor(clients$situationFamiliale)
clients$nbEnfantsAcharge <- as.integer(clients$nbEnfantsAcharge)
clients$X2eme.voiture <- as.logical(clients$X2eme.voiture)
clients$sexe <- as.factor(clients$sexe)
clients$immatriculation <- as.character(clients$immatriculation)
clients <- na.omit(clients)
```

Pour la table marketing :

```
marketing$age <- as.integer(marketing$age)
marketing$taux <- as.integer(marketing$taux)
marketing$situationFamiliale <- as.factor(marketing$situationFamiliale)
marketing$nbEnfantsAcharge <- as.integer(marketing$nbEnfantsAcharge)
marketing$X2eme.voiture <- as.logical(marketing$X2eme.voiture)
marketing$sexe <- as.factor(marketing$sexe)
marketing <- na.omit(marketing)
```

Pour la table catalogue :

```
catalogue$marque <- as.character(catalogue$marque)
catalogue$nom <- as.character(catalogue$nom)
catalogue$puissance <- as.integer(catalogue$puissance)
catalogue$longueur <- as.character(catalogue$longueur)
catalogue$nbplaces <- as.integer(catalogue$nbPlaces)
catalogue$nbportes <- as.integer(catalogue$nbPortes)
catalogue$couleur <- as.character(catalogue$couleur)
catalogue$occasion <- as.character(catalogue$occasion)
catalogue$prix <- as.integer(catalogue$prix)
catalogue <- na.omit(catalogue)
```

Pour la table immatriculations :

```
immatriculation$marque <- as.character(immatriculation$marque)
immatriculation$nom <- as.character(immatriculation$nom)
immatriculation$puissance <- as.integer(immatriculation$puissance)
immatriculation$longueur <- as.character(immatriculation$longueur)
immatriculation$nbplaces <- as.integer(immatriculation$nbPlaces)
immatriculation$nbportes <- as.integer(immatriculation$nbPortes)
immatriculation$couleur <- as.character(immatriculation$couleur)
immatriculation$occasion <- as.character(immatriculation$occasion)
immatriculation$prix <- as.integer(immatriculation$prix)
immatriculation$immatriculation <- as.character(immatriculation$immatriculation)
immatriculation <- na.omit(immatriculation)
```

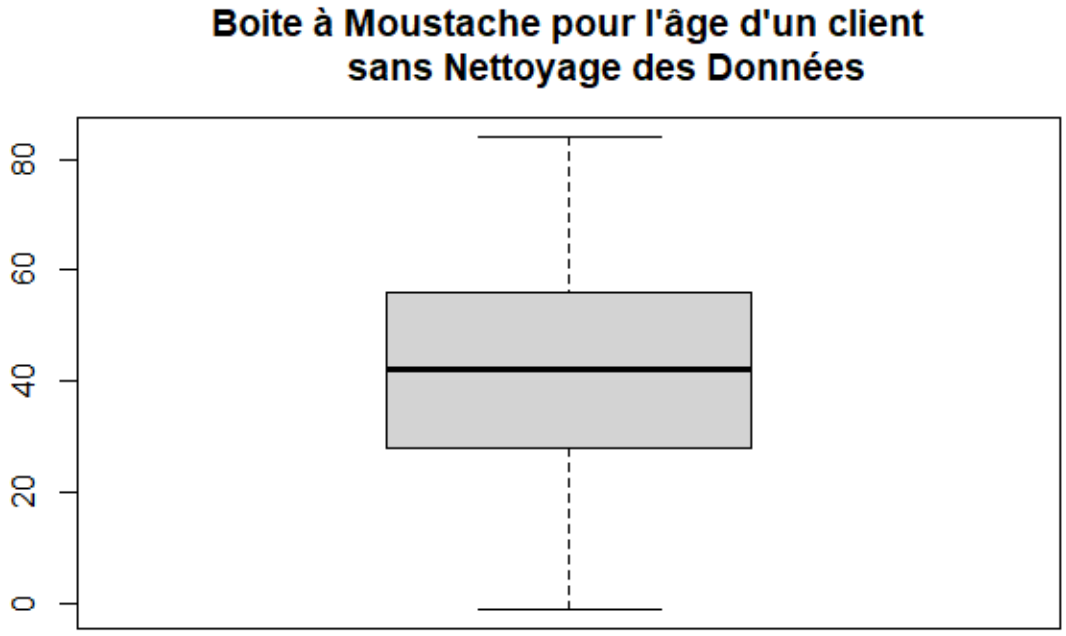
Les deux seules tables à nettoyer réellement sont :

- Clients -> Nettoyage Complet
- Immatriculations -> Suppression des doublons

Nettoyage de la table clients

Age

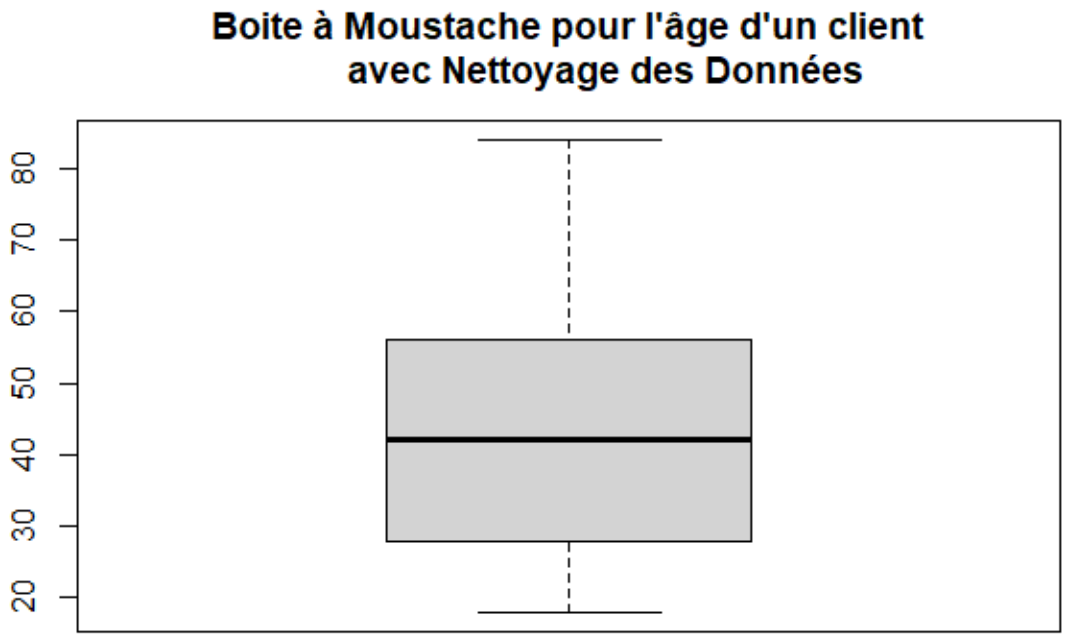
Le summary de cette table, nous montre des valeurs aberrantes, comme par exemple -1 pour l'âge. Nous allons donc récupérer tous les index ou les valeurs de l'âge est compris entre -1 et 17ans car à cet âge, il est impossible de posséder une voiture.



Nous supprimons donc ces valeurs avec la commande suivante :

```
clients <- subset(clients, clients$age >= 17)
```

Nous vérifions grâce à une boite à moustache qu'il n'existe plus de valeurs aberrantes et nous effectuons le même processus pour toutes les variables qui en ont besoin.



Sexe

```
summary(clients$sexe);
             ?         F  Féminin     Femme     Homme         M Masculin       N/D
    91       86     29107       289       321       710     67659       726        90
```

Pour la colonne du sexe, 'Masculin/Homme' et 'Féminin/Femme' pouvait facilement être transformé en 'M' et 'F' plutôt que de supprimer les lignes. Il faut aussi supprimer les valeurs vides et ' ?'.

Voici les commandes que nous avons effectuées :

```
clients <- subset(clients, clients$sexe!="?" & clients$sexe!=" " & clients$sexe!="N/D")

clients$sexe <- str_replace(clients$sexe, "Homme", "M")
clients$sexe <- str_replace(clients$sexe, "Masculin", "M")
clients$sexe <- str_replace(clients$sexe, "Féminin", "F")
clients$sexe <- str_replace(clients$sexe, "Femme", "F")
clients$sexe <- as.factor(clients$sexe)
clients$sexe <- droplevels(clients$sexe)
```

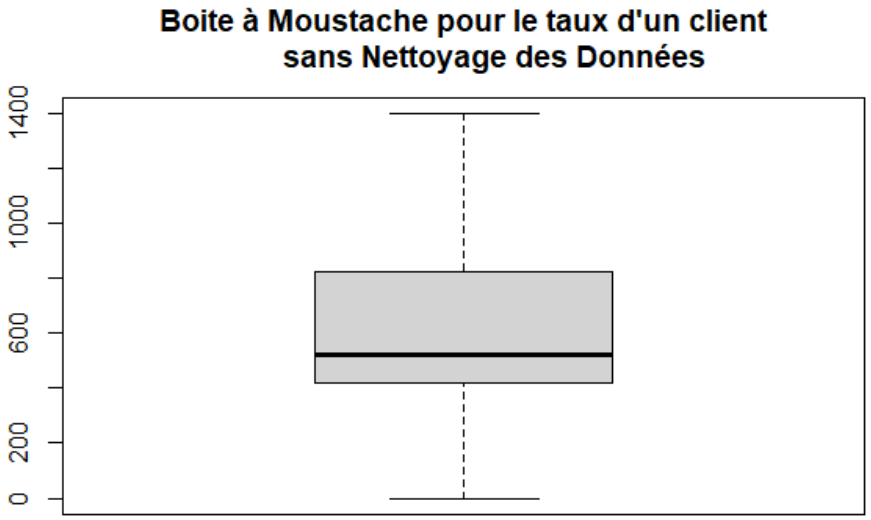
La colonne sexe a bien été nettoyée :

```
summary(clients$sexe);
        ?     F     M   N/D
  0     0 29717 69095     0
```

Taux

Le taux doit être supérieur à 544 selon le sujet.

```
summary(clients$taux)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -1.0   420.0   520.0   606.4   823.0  1399.0
```

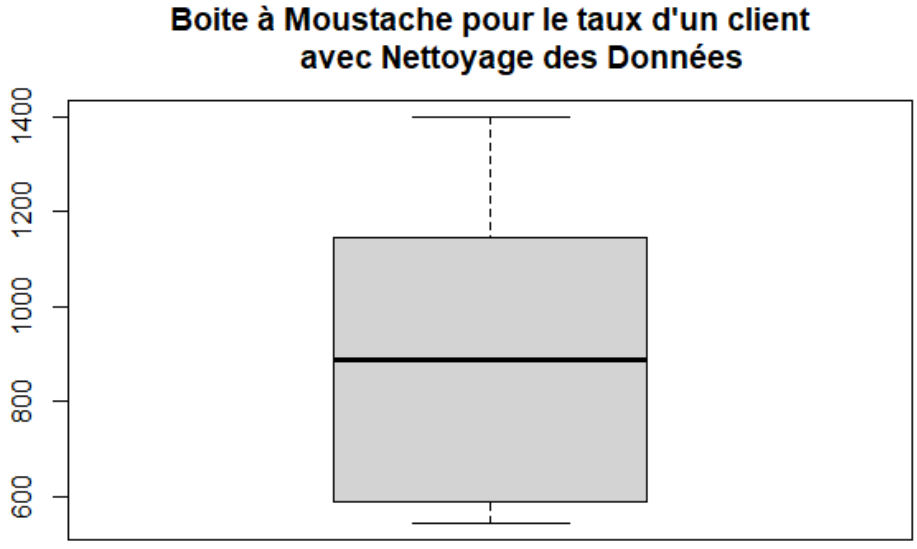


Nous supprimons donc les valeurs inférieures à 544.

```
clients <- subset(clients, clients$taux >= 544)
```

La colonne taux a bien été nettoyée :

```
summary(clients$taux)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
544.0   588.0   889.0   899.2  1146.0  1399.0
```



Situation Familiale

```
summary(clients$situationFamiliale)
                ? Célibataire    Divorcée   En Couple    Marié(e)         N/D        Seul
       50          59       13177          21       27913         280          44         112
    Seule
     2078
```

Pour la colonne situation familiale, nous avons simplement supprimé les valeurs non cohérentes.

```
clients <- subset(clients, clients$situationFamiliale != "?" &
                  clients$situationFamiliale != " " & clients$situationFamiliale != "N/D")
clients$situationFamiliale <- droplevels(clients$situationFamiliale)
```

La colonne situation familiale a bien été nettoyée :

```
> summary(clients$situationFamiliale)
Célibataire    Divorcée   En Couple    Marié(e)        Seul       Seule
      13177          21       27913         280         112        2078
```

Enfants à Charge

```
summary(clients$nbEnfantsAcharge)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.000   0.000   1.000   1.245   2.000   4.000
```

Pour la colonne enfants à charge, nous avons simplement supprimé les valeurs non cohérentes (<0).

```
clients <- subset(clients, clients$nbEnfantsAcharge >= 0)
```

La colonne enfants à charge a bien été nettoyée :

```
summary(clients$nbEnfantsAcharge)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.000   0.000   1.000   1.247   2.000   4.000
```

Doublons

Enfin nous finissons par supprimer les doublons :

```
#Suppression des doublons

doublonsClients <- which(duplicated(clients$immatriculation))
client <-clients[-doublonsClients,]
```

^

Nettoyage de la table immatriculations

Nous supprimons uniquement les doublons.

```
#Suppression des doublons dans le fichier immatriculations

doublons <- which(duplicated(immatriculation$immatriculation))
immatriculations<-immatriculation[-doublons,]
```

Les données sont nettoyées, elles peuvent maintenant être exploitées.

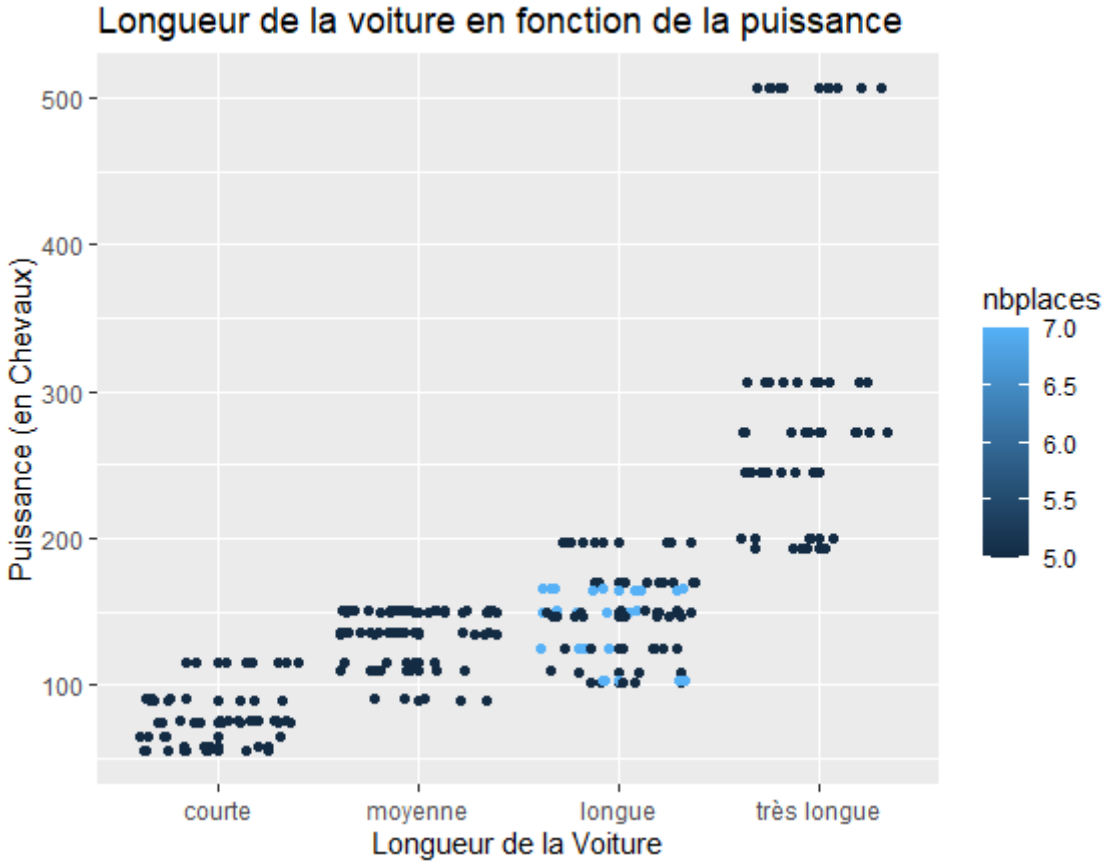
2.4. Analyse Exploratoire des Données

2.4.1. Classification des Types de Voitures du Catalogue

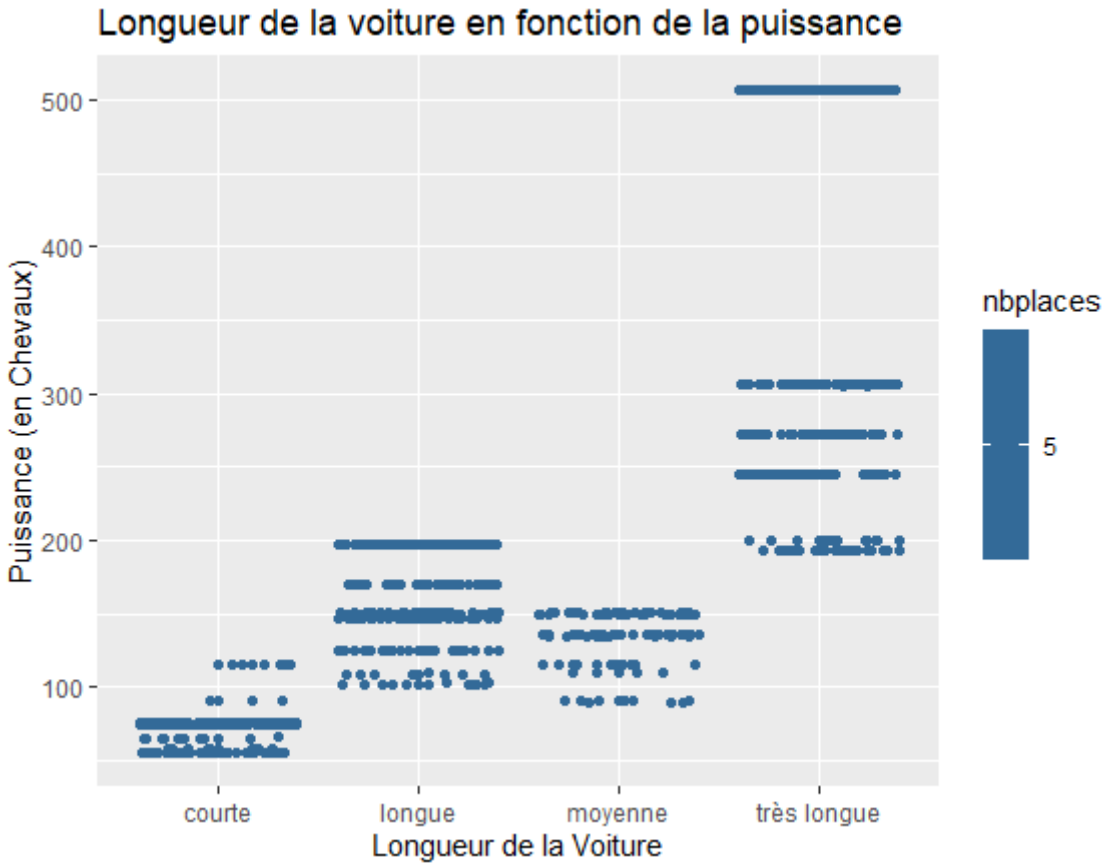
Nous estimons que pour classer des voitures en plusieurs catégories, les trois critères suivants suffisent :

- La longueur du véhicule (courte, moyenne, longue, très longue)
- La puissance (min= 55, max= 507) : nous avons fixé des intervalles de puissance selon les modèles.
- Le nombre de places (min=5, Max=7)

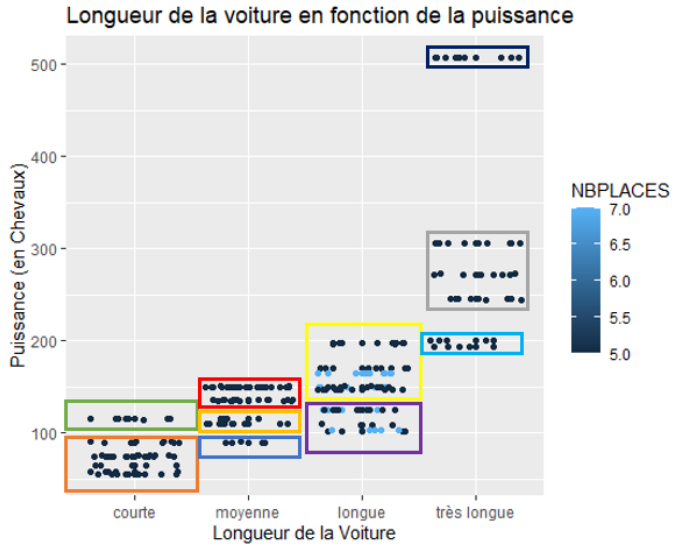
Nous estimons que le nombre de places n'influe pas sur la classification des modèles, en effet tous les modèles comportant 3 portes font partis des véhicules courts. Ainsi la longueur du véhicule suffit pour la classification.
Voici un graphique montrant la répartition des voitures selon leurs longueurs, leurs puissances et leurs nombres de places dans le fichier catalogue.



Voici un graphique montrant la répartition des voitures selon leurs longueurs, leurs puissances et leurs nombres de places dans une partie du fichier immatriculations.



En fonction de la répartition des véhicules, nous avons choisi le découpage suivant :



Couleur	Catégorie
	Citadine
	Citadine Sportive
	Routière
	Routière Sportive
	Routière Ultra Sportive
	Familiale
	Familiale Sportive
	Berline
	Berline Sportive
	Berline Ultra Sportive

On a décidé de les scinder en 10 groupes différents, qu'on peut distinguer sur le graphique.

Type de Véhicule	Longueur du Véhicule	Intervalle de Puissance	Nombre de Places
Citadine	Courte	[55-90]	5
Citadine Sportive	Courte	[90-]	5
Routière	Moyenne	[55-100]	5
Routière Sportive	Moyenne	[100-140]	5
Routière Ultra Sportive	Moyenne	[140 -]	5
Familiale	Longue	[55-140]	5-7
Familiale Sportive	Longue	[140-]	5-7
Berline	Très Longue	[55-250]	5
Berline Sportive	Très Longue	[250-350]	5
Berline Ultra Sportive	Très Longue	[350 -]	5

Il est important de noter que dans le fichier immatriculations, aucune voiture n'a 7 places. Ainsi nous avons étendu la catégorie familiale à des véhicules de 5 places.
Pour créer une variable catégorie dans les tables immatriculations et catalogue, nous utilisons la fonction ifelse().

```
immatriculations$categories <- ifelse(immatriculations$longueur=="courte" & immatriculations$puissance >=55 &
```

2.4.2. Fusion du fichier client et immatriculation

Voici le code pour fusionner les deux fichiers :

```
#fusion du fichiers client et Immatriculation

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

Dans notre nouvelle table clients_immatriculations, la variable immatriculation est inutile donc nous la supprimons.

```
#Suppression de la colonne immatriculations
clients_immatriculations<-clients_immatriculations[,-1]
```

2.5. Tests et Choix des Classifieurs

2.5.1. Définition de l'ensemble d'apprentissage et de test

Le nombre de place dans les voitures étant majoritairement à 5 places nous pouvons supprimer cette valeur car nous n'allons pas l'utiliser.
Puis nous passons en as.factor toutes les variables qui le peuvent. A ce stade, nous avons remarqué que certaines valeurs nous serons inutiles, seules les catégories, et les données "humaines" sur les clients nous importent. Nous avons donc supprimé les données inutiles et celles qui comportent trop de factors.

```r
#Suppression des colonnes
#Seules les catégories, et les données "humaines" sur les clients nous importent
clients_immatriculations<- subset(clients_immatriculations, select = -nbPlaces)
clients_immatriculations <- subset(clients_immatriculations, select = -nbPortes)
clients_immatriculations <- subset(clients_immatriculations, select = -prix)
clients_immatriculations <- subset(clients_immatriculations, select = -longueur)
clients_immatriculations <- subset(clients_immatriculations, select = -puissance)
clients_immatriculations <- subset(clients_immatriculations, select = -nom)
clients_immatriculations <- subset(clients_immatriculations, select = -marque)
clients_immatriculations <- subset(clients_immatriculations, select = -couleur)
clients_immatriculations <- subset(clients_immatriculations, select = -occasion)
```

Ainsi nous supprimons :

- La puissance
- Le nombre de portes
- La longueur
- Le nombre de places

En effet ces 4 variables ont permis de définir nos catégories de véhicules donc elles deviennent inutiles.

- La marque
- La couleur
- Le prix
- L'occasion

Ces 4 variables n'apportent aucune information essentielle pour le véhicule, étant donné que l'on cherche uniquement à prédire la catégorie de véhicules.
Pour cette première itération, nous avons supprimé la variable taux, voir ci-après.

Nous allons prendre 70% des données pour l'ensemble d'apprentissage et 30% pour l'ensemble de test.

```r
#ENSEMBLE D'APPRENTISSAGE
#clients_immatriculations_EA : sélection des 29014 premières lignes de clients_immatriculations.(70% de données)"
clients_immatriculations_EA <- clients_immatriculations[1:68773,]

#ENSEMBLE DE TEST
#@clients_immatriculations_ET : sélection des  dernières lignes de clients_immatriculations.(30% de données)"
clients_immatriculations_ET <- clients_immatriculations[68773:98246,]
```

Nous obtenons ainsi deux datasets avec 6 colonnes.

1	citadine	81	F	Célibataire	0	FALSE
2	familliale sportive	60	M	En Couple	0	FALSE
3	citadine	50	F	Célibataire	0	FALSE
4	berline ultra sportive	54	M	En Couple	0	FALSE
5	berline sportive	84	F	En Couple	1	FALSE
6	berline sportive	28	F	En Couple	2	FALSE
7	berline ultra sportive	19	F	En Couple	4	FALSE
8	berline sportive	74	M	En Couple	3	FALSE
9	berline ultra sportive	42	M	En Couple	4	TRUE
10	familliale sportive	29	M	En Couple	1	FALSE

2.5.2. Tests des Classifieurs sans la variable Taux

K-NEAREST NEIGHBORS

```
kknn6<-kknn(categories~., clients_immatriculations_EA, clients_immatriculations_ET)

# Matrice de confusion
table(clients_immatriculations_ET$categories, kknn6$fitted.values)

# Conversion des probabilites en data frame
knn_prob <- as.data.frame(kknn6$prob)

knn_auc <-multiclass.roc(clients_immatriculations_ET$categories, knn_prob)
print(knn_auc)
```

Voici la matrice de confusion obtenue :

	berline	berline sportive	berline ultra sportive	citadine	citadine sportive
berline	682	394	854	0	0
berline sportive	441	1142	572	1	0
berline ultra sportive	970	326	1324	2	0
citadine	2	0	4	6526	79
citadine sportive	0	0	0	170	7
familliale	89	94	204	0	0
familliale sportive	261	428	618	3	0
routiere	0	0	0	171	1
routiere sportive	0	1	0	1257	16
routiere ultra sportive	0	0	0	1095	24

	familliale	familliale sportive	routiere	routiere sportive	routiere ultra sportive
berline	102	749	0	0	0
berline sportive	97	987	0	0	1
berline ultra sportive	159	1033	0	0	0
citadine	1	1	0	618	723
citadine sportive	0	0	0	33	37
familliale	151	1223	0	0	0
familliale sportive	531	4315	0	0	1
routiere	0	0	0	30	29
routiere sportive	0	0	1	399	149
routiere ultra sportive	0	0	0	113	233

Et nous avons comme valeurs pour l'AUC :

```
Multi-class area under the curve: 0.7666
```

NEURAL NETWORKS

```
nnet5<-nnet(categories ~., clients_immatriculations_EA, size=8)

# Test du classifieur : classe predite
nn_class <- predict(nnet5, clients_immatriculations_ET, type="class")
nn_class
table(nn_class)

# Matrice de confusion
table(clients_immatriculations_ET$categories, nn_class)

# Test du classifieur : probabilites pour chaque prediction
nn_prob <- predict(nnet5, clients_immatriculations_ET, type="raw")
nn_auc <-multiclass.roc(clients_immatriculations_ET$categories, nn_prob)
print(nn_auc)
```

Voici la matrice de confusion obtenue :

```
                     nn_class
                      berline berline sportive berline ultra sportive citadine citadine sportive
berline                     7               162                   1515        0                 0
berline sportive            7               943                    963        2                 0
berline ultra sportive     15                 0                   2259        2                 0
citadine                    0                 0                      3     7710                 1
citadine sportive           0                 0                      0      237                 0
familliale                  0                 0                      0        0                 0
familliale sportive         0                 0                      1        2                 0
routiere                    0                 0                      0      217                 0
routiere sportive           0                 0                      0     1568                 0
routiere ultra sportive     0                 0                      0     1454                 0
                     nn_class
                      familliale sportive routiere sportive
berline                              1097                 0
berline sportive                     1326                 0
berline ultra sportive               1538                 0
citadine                                5               235
citadine sportive                       0                10
familliale                           1761                 0
familliale sportive                  6152                 2
routiere                                0                14
routiere sportive                       1               254
routiere ultra sportive                 0                11
```

Et nous avons comme valeurs pour l'AUC :

```
Multi-class area under the curve: 0.8659
```

RANDOM FORESTS

Pour le random forest, nous avons supprimer la variable âge, car elle comprenait trop de facteurs

```r
clients_immatriculations_EA_RF <- subset(clients_immatriculations_EA, select = -AGE)
# Apprentissage du classifeur de type random forest
rf <- randomForest(categories~., clients_immatriculations_EA_RF)

# Test du classifieur : classe predite
rf_class <- predict(rf,clients_immatriculations_ET, type="response")
table(rf_class)

# Matrice de confusion
table(clients_immatriculations_ET$categories, rf_class)

# Test du classifieur : probabilites pour chaque prediction
rf_prob <- predict(rf, clients_immatriculations_ET , type="prob")
rf_auc <-multiclass.roc(clients_immatriculations_ET$categories, rf_prob)
print(rf_auc)
```

Voici la table de confusion obtenue :

```
                     rf_class
                      berline berline sportive berline ultra sportive citadine citadine sportive
berline                     0                 8                   1676        0                 0
berline sportive            0                13                   1900        2                 0
berline ultra sportive      0                10                   2264        2                 0
citadine                    0                 0                      3     7946                 0
citadine sportive           0                 0                      0      247                 0
familliale                  0                 0                      0        0                 0
familliale sportive         0                 0                      1        4                 0
routiere                    0                 0                      0      231                 0
routiere sportive           0                 0                      0     1822                 0
routiere ultra sportive     0                 0                      0     1465                 0
                     rf_class
                      familliale familliale sportive routiere routiere sportive routiere ultra sportive
berline                        0                1097        0                 0                      0
berline sportive               0                1326        0                 0                      0
berline ultra sportive         0                1538        0                 0                      0
citadine                       0                   5        0                 0                      0
citadine sportive              0                   0        0                 0                      0
familliale                     0                1761        0                 0                      0
familliale sportive            0                6152        0                 0                      0
routiere                       0                   0        0                 0                      0
routiere sportive              0                   1        0                 0                      0
routiere ultra sportive        0                   0        0                 0                      0
```

Et nous avons comme valeurs pour l'AUC :

```
Multi-class area under the curve: 0.61
```

Naive Bayes

```r
# Apprentissage du classifeur de type naive bayes
nb <- naive_bayes(categories~., clients_immatriculations_EA)

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

# Matrice de confusion
table( clients_immatriculations_ET$categories, nb_class)

# Test du classifieur : probabilites pour chaque prediction
nb_prob <- predict(nb,  clients_immatriculations_ET, type="prob")
nb_auc <- multiclass.roc(clients_immatriculations_ET$categories, nb_prob)
print(nb_auc)
```

Nous avons cette matrice de confusion :

```
                      nb_class
                       berline berline sportive berline ultra sportive citadine citadine sportive
  berline                    0              164                   1562       57                  0
  berline sportive           0              975                    994       47                  0
  berline ultra sportive     4                0                   2329       79                  0
  citadine                   0              312                     14     6790                  0
  citadine sportive          0                8                      2      214                  0
  familliale                 0                7                     59      104                  0
  familliale sportive        0               50                    188      288                  0
  routiere                   0                0                      0      175                  0
  routiere sportive          0                0                      0      997                  0
  routiere ultra sportive    0                0                      0     1419                  0
                      nb_class
                       familliale familliale sportive routiere routiere sportive routiere ultra sportive
  berline                       0                998        0                  0                      0
  berline sportive              0               1225        0                  0                      0
  berline ultra sportive        0               1402        0                  0                      0
  citadine                      0                  2        0                827                      9
  citadine sportive             0                  0        0                 23                      0
  familliale                    0               1591        0                  0                      0
  familliale sportive           0               5629        0                  2                      0
  routiere                      0                  0        0                 56                      0
  routiere sportive             0                  1        0                823                      2
  routiere ultra sportive       0                  0        0                 42                      4
```

Et la valeur de l'AUC est de :

```
Multi-class area under the curve: 0.8636
```

SUPPORT VECTOR MACHINES

```r
# Apprentissage du classifeur de type svm
svm <- svm(categories~., clients_immatriculations_EA, probability=TRUE)
svm

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

# Matrice de confusion
table(clients_immatriculations_ET$categories, svm_class)

# Test du classifieur : probabilites pour chaque prediction
svm_prob <- predict(svm, clients_immatriculations_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(clients_immatriculations_ET$categories, svm_prob)
print (svm_auc)
```

Nous avons cette matrice de confusion :

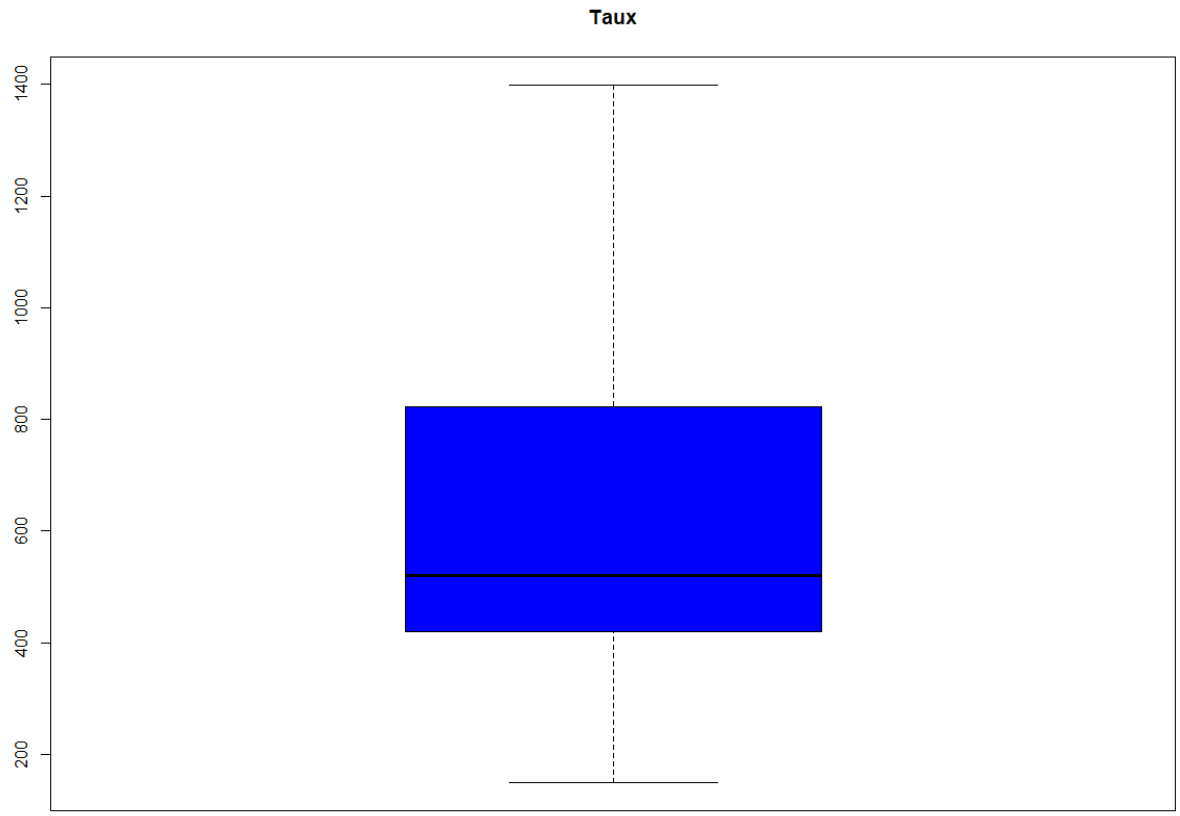
```
                     svm_class
                      berline berline sportive berline ultra sportive citadine citadine sportive familliale
berline                     0              162                    1521        2                 0          0
berline sportive            0              939                     967       10                 0          0
berline ultra sportive      0                0                    2271        5                 0          0
citadine                    0                0                       3     7946                 0          0
citadine sportive           0                0                       0      247                 0          0
familliale                  0                0                       0        2                 0          0
familliale sportive         0                0                       1        7                 0          0
routiere                    0                0                       0      231                 0          0
routiere sportive           0                0                       0     1822                 0          0
routiere ultra sportive     0                0                       0     1465                 0          0
                     svm_class
                      familliale sportive routiere routiere sportive routiere ultra sportive
berline                                1095        0                0                      1
berline sportive                       1324        0                1                      0
berline ultra sportive                 1537        0                0                      1
citadine                                  5        0                0                      0
citadine sportive                         0        0                0                      0
familliale                             1759        0                0                      0
familliale sportive                    6145        0                0                      4
routiere                                  0        0                0                      0
routiere sportive                         1        0                0                      0
routiere ultra sportive                   0        0                0                      0
```

Et la valeur de l'AUC est de :

```
Multi-class area under the curve: 0.8773
```

2.5.3. Tests des Classifieurs avec la variable Taux

Ajout de la variable taux dans les calculs



Ci-dessus, la boite à moustaches montre la répartition du taux en fonction des clients, nous allons pouvoir créer des catégories de taux : faible, moyen, élevée et très élevée.

```
ifelse(clients_immatriculations$TAUX < 420 ,"Faible",
       ifelse(clients_immatriculations$TAUX <607.1, "Moyen",
              ifelse(clients_immatriculations$TAUX <823, "Elevée", "Très élevée")))
```

K-NEAREST NEIGHBORS

```
Multi-class area under the curve: 0.8432
```

NEURAL NETWORKS

```
Multi-class area under the curve: 0.9227
```

RANDOM FORESTS

```
Multi-class area under the curve: 0.6838
```

NAIVE BAYES

```
Multi-class area under the curve: 0.9155
```

SUPPORT VECTOR MACHINES

```
Multi-class area under the curve: 0.9154
```

2.6. Conclusion et choix du classifieur

Classifieur	AUC Sans le Taux	AUC Avec Le Taux
K-NEAREST NEIGHBORS	0,7666	0,8432
NEURAL NETWORKS	0,8659	0,9227
RANDOM FORESTS	0,61	0 ,6838
NAIVE BAYES	0,8636	0,9155
SUPPORT VECTOR MACHINES	0,8773	0,9154

On remarque :

- Les classifieurs sont bien plus précis avec l'ajout de la variable Taux.
- Le classifieur le plus précis est NEURAL NETWORKS

2.7. Classification – Prédiction de la Catégorie pour le fichier Marketing

On va donc appliquer la méthode de prédiction Neural Networks, étant donné que c'est celle qui présente le meilleur AUC et donc la meilleure prédiction. L'ensemble d'apprentisssage correspond au dataframe clients_immatriculations et la prédiction se fera sur le dataframe marketing
Pour cela nous suivons quasiment le même processus que lors des tests des classifieurs :

- Création et application des catégories de Taux à l'ensemble du fichier marketing

```
> summary(marketing)
      AGE       SEXE         TAUX          SITUATIONFAMILIALE NBENFANTSACHARGE DEUXIEMEVOITURE        cateTaux
 22     : 2   F: 9   Min.   : 153.0   Célibataire: 8       0:12              FALSE:15         Elevée     :1
 35     : 2   M:11   1st Qu.: 408.5   En Couple  :12       1: 1              TRUE : 5         Faible     :6
 59     : 2          Median : 527.0                        2: 3                               Moyen      :9
 19     : 1          Mean   : 582.4                        3: 4                               Très élevée:4
 21     : 1          3rd Qu.: 628.8
 26     : 1          Max.   :1396.0
 (Other):11
```

- Apprentissage du réseau de neurones :

```
#Apprentissage
nnet5_marketing<-nnet(categories ~., clients_immatriculations_taux, size=7)
```

- Classification du dataframe marketing :

```
#Classification
catégorie_prédite<- predict(nnet5_marketing, marketing, type="class")
catégorie_prédite
table(catégorie_prédite)
```

On obtient 20 prédictions réparties sur 5 catégories :

```
> table(catégorie_prédite)
catégorie_prédite
          berline    berline sportive           citadine         familliale familliale sportive
                3                   2                  10                  2                   3
>
```

- Récupération des probabilités pour chaque prédiction :

```
# Recuperation des probabilites pour chaque prediction
nn_prob_marketing <- predict(nnet5_marketing, marketing, type="raw")

nn_prob_marketing
```

```
> nn_prob_marketing
        berline berline sportive berline ultra sportive     citadine citadine sportive   familliale
1  1.408891e-07      1.957036e-09           2.403523e-04 8.152888e-01      7.653813e-02 1.302427e-12
2  1.643593e-05      1.796450e-08           1.555861e-05 4.491547e-01      8.361431e-03 1.180030e-09
3  1.815457e-05      2.019687e-08           1.713138e-05 5.083238e-01      9.638304e-03 8.958641e-10
4  3.975731e-01      3.167683e-01           2.852350e-01 1.706828e-04      1.773648e-04 1.414638e-10
5  1.992698e-08      9.910339e-01           6.273172e-03 2.873925e-07      6.131559e-07 1.156337e-08
6  2.160955e-01      1.130272e-01           4.006244e-02 2.720568e-05      1.300978e-05 3.456250e-01
7  8.782040e-02      8.812817e-02           1.026253e-01 6.710884e-05      4.280528e-05 1.502780e-01
8  1.244919e-03      1.671550e-05           7.155317e-04 4.705504e-01      2.016958e-02 7.531554e-08
9  3.645867e-07      1.252488e-06           2.033660e-05 5.257137e-01      1.149943e-02 2.278287e-10
10 2.181344e-01      1.139324e-01           4.041047e-02 2.724025e-05      1.303775e-05 3.433934e-01
11 4.652935e-20      8.442610e-01           7.741435e-06 1.535560e-07      2.527995e-07 1.296187e-12
12 1.330894e-03      1.718489e-05           7.382437e-04 4.617168e-01      1.744839e-02 9.774450e-08
13 2.720742e-05      3.165772e-08           2.510449e-05 5.443550e-01      1.098627e-02 1.357832e-09
14 1.563228e-04      2.173549e-03           4.135877e-01 1.084442e-04      2.646973e-04 5.494764e-04
15 6.897959e-05      2.716177e-15           1.383249e-05 9.999146e-01      2.326990e-10 1.420594e-11
16 6.823446e-01      2.507134e-01           6.693611e-02 3.595475e-08      8.555340e-08 1.634422e-07
17 2.275094e-04      4.902867e-06           2.578808e-01 4.698707e-04      1.004245e-06 2.462145e-03
18 3.706304e-01      3.135266e-01           3.153736e-01 1.872706e-04      1.998933e-04 1.362533e-10
19 2.233755e-04      1.162452e-06           1.535862e-04 4.942636e-01      1.353848e-02 1.153084e-08
20 6.831433e-08      3.107345e-17           6.408226e-05 9.999357e-01      1.026438e-09 2.502406e-15
   familliale sportive     routiere routiere sportive routiere ultra sportive
1         4.009341e-09 4.759824e-06      1.876661e-02           8.916116e-02
2         7.540208e-09 1.299971e-01      2.866155e-01           1.258393e-01
3         7.259939e-09 8.476406e-02      2.552456e-01           1.419929e-01
4         7.538218e-05 2.252388e-28      5.575226e-10           8.016267e-08
5         2.691945e-03 9.895748e-26      2.145234e-08           5.190076e-15
6         2.846828e-01 5.680502e-14      4.666458e-04           2.219486e-07
7         5.701737e-01 3.048733e-14      8.637798e-04           7.347251e-07
8         1.615792e-05 1.131418e-06      1.595230e-01           3.477625e-01
9         2.204394e-07 2.130588e-04      4.543127e-01           8.238905e-03
10        2.836278e-01 5.530333e-14      4.610763e-04           2.199825e-07
11        1.556885e-01 1.615867e-24      4.232479e-05           1.816993e-18
12        2.169323e-05 8.908415e-07      1.687330e-01           3.499928e-01
13        9.955010e-09 6.460284e-02      2.555415e-01           1.244620e-01
14        5.827028e-01 1.113220e-16      4.547734e-04           2.207859e-06
15        1.521282e-10 5.854373e-15      2.472632e-06           9.764997e-08
16        5.551367e-06 5.919020e-29      1.886199e-12           4.429061e-14
17        7.386345e-01 4.710743e-18      3.190777e-04           2.070256e-07
18        8.216829e-05 2.229596e-28      6.057407e-10           9.033390e-08
19        1.019190e-06 4.646788e-05      1.841757e-01           3.075966e-01
20        1.343821e-11 4.454801e-19      9.660986e-08           3.290547e-08
```

- Affectation des résultats au dataframe résultat :

```
resultat <- data.frame(marketingResultat,catégorie_prédite,nn_prob_marketing)
```

Voilà le dataframe contenant les résultats des prédictions ainsi que les probabilités associées :

	AGE	SEXE	TAUX	SITUATIONFAMILIALE	NBENFANTSACHARGE	DEUXIEMEVOITURE	cateTaux	catégorie_prédite	berline	berline.sportive	be
1	21	F	1396	Célibataire	0	FALSE	Très élevée	citadine	1.408891e-07	1.957036e-09	
2	35	M	223	Célibataire	0	FALSE	Faible	citadine	1.643593e-05	1.796450e-08	
3	48	M	401	Célibataire	0	FALSE	Faible	citadine	1.815457e-05	2.019687e-08	
4	26	F	420	En Couple	3	TRUE	Moyen	berline	3.975731e-01	3.167683e-01	
5	80	M	530	En Couple	3	FALSE	Moyen	berline sportive	1.992698e-08	9.910339e-01	
6	27	F	153	En Couple	2	FALSE	Faible	familliale	2.160955e-01	1.130272e-01	
7	59	F	572	En Couple	2	FALSE	Moyen	familliale sportive	8.782040e-02	8.812817e-02	
8	43	F	431	Célibataire	0	FALSE	Moyen	citadine	1.244919e-03	1.671550e-05	
9	64	M	559	Célibataire	0	FALSE	Moyen	citadine	3.645867e-07	1.252488e-06	
10	22	M	154	En Couple	1	FALSE	Faible	familliale	2.181344e-01	1.139324e-01	
11	79	F	981	En Couple	2	FALSE	Très élevée	berline sportive	4.652935e-20	8.442610e-01	
12	55	M	588	Célibataire	0	FALSE	Moyen	citadine	1.330894e-03	1.718489e-05	
13	19	F	212	Célibataire	0	FALSE	Faible	citadine	2.720742e-05	3.165772e-08	
14	34	F	1112	En Couple	0	FALSE	Très élevée	familliale sportive	1.563228e-04	2.173549e-03	
15	60	M	524	En Couple	0	TRUE	Moyen	citadine	6.897959e-05	2.716177e-15	
16	22	M	411	En Couple	3	TRUE	Faible	berline	6.823446e-01	2.507134e-01	
17	58	M	1192	En Couple	0	FALSE	Très élevée	familliale sportive	2.275094e-04	4.902867e-06	
18	54	F	452	En Couple	3	TRUE	Moyen	berline	3.706304e-01	3.135266e-01	
19	35	M	589	Célibataire	0	FALSE	Moyen	citadine	2.233755e-04	1.162452e-06	
20	59	M	748	En Couple	0	TRUE	Elevée	citadine	6.831433e-08	3.107345e-17	

- Enfin nous avons exporté le dataframe résultat dans un fichier csv :

```
# Enregistrement du fichier de resultats au format csv
write.table(resultat, file='predictions.csv', sep="\t", dec=".", row.names = F)
```

3. Map/Reduce

3.1. Fonction Map

Avant de commencer, analysons les données du ficchier CO2 :

```
12,HYUNDAI KONA electric 39 kWh,-6 000€ 1,0,205 €
13,JAGUAR I-PACE EV400 (400 CEE) - Automatique - 4 roues motrices,-6 000€ 1,0,271 €
14,"KIA e-NIRO Moteur Ã©lectrique synchrone Ã  aimant permanent, 150kW (204ch)",-6 000€ 1,0,212 €
```

Nous pouvons remarquer que la première ligne est l'entête du tableau et que toutes les valeurs sont séparées par des virgules. De plus la première colonne représente le numéro de la ligne.

Nous créons donc une classe CarMap :

```
public class CarMap extends Mapper<Object, Text, Text, Text> {
```

Puis nous ignorons la première ligne :

```
//on ne prend pas en compte la première ligne qui est le titre des colonnes
if (value.toString().contains("Marque")){
        return;
}
```

Nous récupérons la ligne en remplaçant les espaces du fichiers csv par de « vrais » espaces :

```
String line = value.toString();
//On foit remplacer tous les espaces par de "vrais" espaces
line = line.replaceAll("\\u00a0"," ");
```

Puis on récupère toutes les valeurs dans un tableau :

```
//on sépare les données
String[] splitted_line = line.split(",");
```

Nous récupérons le nom de la marque en prenant seulement le premier mot de la deuxième colonne :

```
// Recuperation de la colonne marque
String marque;
String[] splitted_space = splitted_line[1].split("\\s+");
//Seul le premier mot correspond au nom de la marque
marque = splitted_space[0];
```

Si nous regardons le fichiers CO2, nous pouvons remarquer par exemple à la ligne 14 que certains noms de marque sont entre guillemets donc nous les enlevons :

```
//Certaines marques commencent par des guillemets donc on les enleve
marque = marque.replace("\"", "");
```

Maintenant nous allons récupérer la valeur du bonus/malus :

```
        // Recuperation de la colonne Bonus/Malus
String malus_bonus = splitted_line[2];
```

Maintenant, nous allons enlever les espaces, les symboles € et les « €1 » :

```
//On modifie toutes les valeurs "sales" pour avoir un entier
malus_bonus = malus_bonus.replaceAll(" ", "").replace("€1", "").replace("€", "").replace("\"", "");
```

Et on traite les cas particuliers :

```
// On traite les cas particuliers :
if (malus_bonus.equals("150kW(204ch)") || malus_bonus.equals("100kW(136ch)"))
{
        return;
}

if (malus_bonus.length() == 1){
        malus_bonus="0";
}
```

Passons maintenant à la récupération des valeurs de rejet de CO2 (pas besoin de faire des modifications) :

```
// Recuperation de la colonne Rejet CO2
String rejet = splitted_line[3];
```

Nous allons maintenant récupérer la valeur du cout énergie, pour cela nous créons un tableau de toutes les valeurs sans espaces :

```
// Recuperation de la colonne cout energie
String cout = splitted_line[4];
//On separe les valeurs par les espaces
String[] cout_splitted = cout.split(" ");
```

Il nous suffit d'ignorer la dernière valeur du tableau (c'est un €) pour avoir le cout :

```
//La taille du tableau est doit de 2 ou 3 on ignore seulement la derbière valeur qui est le symbole €
if(cout_splitted.length == 2){
        cout = cout_splitted[0];
} else if(cout_splitted.length == 3){
        cout= cout_splitted[0] + cout_splitted[1];
}
```

Nous pouvons convertir les valeurs obtenues en entier :

```
//Cast des valeurs en int
int malus_bonus_int = Integer.parseInt(malus_bonus);
int rejet_int = Integer.parseInt(rejet);
int cout_int = Integer.parseInt(cout);
```

Puis nous associons les clé/valeur :

```
// clé/valeurs -> clé = marque et valeur -> le reste
String new_value = String.valueOf(malus_bonus_int) + "|" +  String.valueOf(rejet_int) + "|" + String.valueOf(cout_int);

context.write(new Text(marque), new Text(new_value));
```

3.2. Fonction Reduce

Nous créons une classe reduce :

```
public class CarReduce extends Reducer<Text, Text, Text, Text> {

    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {

Iterator<Text> i = values.iterator();
while(i.hasNext()) {
        String node = i.next().toString();

        String[] splitted_node = node.split("\\|");
        bonus = splitted_node[0];
        rejet = splitted_node[1];
        cout = splitted_node[2];

        sommeBonus += Integer.parseInt(bonus);
        sommeRejet += Integer.parseInt(rejet);
        sommeCout += Integer.parseInt(cout);

        count++;
}
moyenneBonus = sommeBonus/count;
moyenneRejet = sommeRejet/count;
moyenneCout = sommeCout/count;

context.write(key, new Text(moyenneBonus + "\t" + moyenneRejet + "\t" + moyenneCout));
```

Nous récupérons toutes les valeurs bonus/malus, rejet et cout. Puis nous additionnons toutes les valeurs de ces variables et nous incrémentons le compteur de 1. A la fin de la boucle while, nous pouvons calculer la moyenne des valeurs.

3.3. Lancement de la fonction map/reduce

Dans un premier temps, nous allons ajouter les fichiers CO2.cvs et les fichiers java :

```
-- Ajout du fichier C02.csv sur la machine distante depuis mon terminal
scp -P 9922 CO2_group8.csv avictor@135.125.106.207:~/

-- Vérification de l'Ajout
avictor@vps-f1f17a1e:~$ ls

--Déplacment du fichier vers mon répertoire utilisateur dans HDFS
hadoop fs -put CO2_group8.csv /user/avictor/

--Vérification que le fichier csv a bien été ajouté dans mon répertoire HDFS
hadoop fs -ls /user/avictor/
Reponse:
Found 5 items
-rw-r--r--   1 avictor studentgroup      38916 2021-03-26 13:32 /user/avictor/CO2_group8.csv
-rw-r--r--   1 avictor studentgroup      10310 2021-02-15 14:19 /user/avictor/anagrame.txt
-rw-r--r--   1 avictor studentgroup         78 2021-02-15 15:49 /user/avictor/graph_input.txt
-rw-r--r--   1 avictor studentgroup       1669 2021-02-15 11:19 /user/avictor/poeme2.txt
drwxr-xr-x   - avictor studentgroup          0 2021-02-15 15:49 /user/avictor/results


--Ajout des fichiers.java sur la machine virtuelle, depuis mon terminal
scp -P 9922 Car.java avictor@135.125.106.207:~/
scp -P 9922 CarMap.java avictor@135.125.106.207:~/
scp -P 9922 CarReduce.java avictor@135.125.106.207:~/

-- Vérification de l'Ajout
avictor@vps-f1f17a1e:~$ ls
```

Nous pouvons maintenant compiler les codes java puis générer le fichier .jar :

```
--Compilation du code
avictor@vps-f1f17a1e:~$  javac Car.java CarMap.java CarReduce.java

--Construisez la hierarchie du .jar et y déplacez le code compilé
avictor@vps-f1f17a1e:~$ mkdir -p org/car
avictor@vps-f1f17a1e:~$ mv Car*.class org/car

--Vérification que les fichiers ont bien été déplacés
avictor@vps-f1f17a1e:~/org/car$ ls

--Génération du.jar
avictor@vps-f1f17a1e:~ jar -cvf CO2.jar -C . org
```

On exécute le programme :

```
--Execution du Programme
avictor@vps-f1f17a1e:~$ hadoop jar CO2.jar org.car.Car  /user/avictor/CO2_group8.csv /user/avictor/results/Projet3A
```

Et on visualise les résultats :

```
--Lecture des résultats
avictor@vps-f1f17a1e:~$ hadoop fs -ls /user/avictor/results/Projet3A
Found 2 items
-rw-r--r--   1 avictor studentgroup          0 2021-03-26 14:22 /user/avictor/results/Projet3A/_SUCCESS
-rw-r--r--   1 avictor studentgroup        396 2021-03-26 14:22 /user/avictor/results/Projet3A/part-r-00000

avictor@vps-f1f17a1e:~$ hadoop fs -cat /user/avictor/results/Projet3A/part-r-00000
AUDI    -2400   26      191
BENTLEY 0       84      102
BMW     -631    39      80
CITROEN -6000   0       347
DS      -3000   16      159
HYUNDAI -4000   8       151
JAGUAR  -6000   0       271
KIA     -3000   15      132
LAND    0       69      78
MERCEDES        7790    187     749
MINI    -3000   21      126
MITSUBISHI      0       40      98
NISSAN  5802    160     681
PEUGEOT -3000   15      144
PORSCHE 0       69      89
RENAULT -6000   0       206
SKODA   -666    27      98
SMART   -6000   0       191
TESLA   -6000   0       245
TOYOTA  0       32      43
VOLKSWAGEN      -1714   23      96
VOLVO   0       42      72
```

```
[avictor@vps-f1f17a1e:~$ hadoop fs -cat /user/avictor/results/BIGDATA/part-r-00000
AUDI    -2400   26      191
BENTLEY 0       84      102
BMW     -631    39      80
CITROEN -6000   0       347
DS      -3000   16      159
HYUNDAI -4000   8       151
JAGUAR  -6000   0       271
KIA     -3000   15      132
LAND    0       69      78
MERCEDES        7790    187     749
MINI    -3000   21      126
MITSUBISHI      0       40      98
NISSAN  5802    160     681
PEUGEOT -3000   15      144
PORSCHE 0       69      89
RENAULT -6000   0       206
SKODA   -666    27      98
SMART   -6000   0       191
TESLA   -6000   0       245
TOYOTA  0       32      43
VOLKSWAGEN      -1714   23      96
VOLVO   0       42      72
```

Nous pouvons maintenant exporter les résultats après vérifications :

```
--Export des résultats
avictor@vps-f1f17a1e:~$ hadoop fs -get /user/avictor/results/Projet3A/part-r-00000

--Copie du fichier de résultat sur ma machine
victorazalbert1@MacBook-Pro MapReduce % scp -P 9922 avictor@135.125.106.207:~/part-r-00000 ./

--Conversion en fichier csv
import-csv .\part-r-00000 -delimiter "`t" | export-csv resultMapReduce.csv -NoTypeInformation
```

3.4. Transfert des données CO2 vers les données Catalogue

Pour réaliser cette étape, nous avons décidé d'utiliser un script python, via jupyter notebook.

On importe les données du résultats et remettant les entêtes des colonnes :

```
co2 = pd.read_csv('resultMapReduce.csv', header=None)
co2.columns = ['marque','bonus/malus','rejetC02(g/km)','coutEnergie']
```

Voici le résultat :

	marque	bonus/malus	rejetC02(g/km)	coutEnergie
0	Audi	-2400	26	191
1	Bentley	0	84	102
2	BMW	-631	39	80
3	Citroen	-6000	0	347
4	Ds	-3000	16	159
5	Hyundai	-4000	8	151
6	Jaguar	-6000	0	271
7	Kia	-3000	15	132
8	Land	0	69	78
9	Mercedes	7790	187	749
10	Mini	-3000	21	126
11	Mitsubishi	0	40	98
12	Nissan	5802	160	681
13	Peugeot	-3000	15	144
14	Porsche	0	69	89
15	Renault	-6000	0	206
16	Skoda	-666	27	98
17	Smart	-6000	0	191
18	Tesla	-6000	0	245
19	Toyota	0	32	43
20	Volkswagen	-1714	23	96
21	Volvo	0	42	72

Nous pouvons supprimer les marques qui ne sont pas présentent dans le fichier catalogue :

```
co2.drop(co2.loc[co2['marque']=='Bentley'].index, inplace=True)
co2.drop(co2.loc[co2['marque']=='Land'].index, inplace=True)
co2.drop(co2.loc[co2['marque']=='Mitsubishi'].index, inplace=True)
co2.drop(co2.loc[co2['marque']=='Porsche'].index, inplace=True)
co2.drop(co2.loc[co2['marque']=='Smart'].index, inplace=True)
co2.drop(co2.loc[co2['marque']=='Tesla'].index, inplace=True)
```

Et voici le résultat :

	marque	bonus/malus	rejetC02(g/km)	coutEnergie
0	Audi	-2400	26	191
2	BMW	-631	39	80
3	Citroen	-6000	0	347
4	Ds	-3000	16	159
5	Hyundai	-4000	8	151
6	Jaguar	-6000	0	271
7	Kia	-3000	15	132
9	Mercedes	7790	187	749
10	Mini	-3000	21	126
12	Nissan	5802	160	681
13	Peugeot	-3000	15	144
15	Renault	-6000	0	206
16	Skoda	-666	27	98
19	Toyota	0	32	43
20	Volkswagen	-1714	23	96
21	Volvo	0	42	72

Il nous manque des valeurs pour les marques qui ne sont pas présentent dans le fichier CO2.csv, nous créons donc des valeurs en faisant la moyenne des autres valeurs :

```
meanBonusMalus=0
meanRejetCO2=0
meanCoutEnergie=0

for index, row in co2.iterrows():
    meanBonusMalus += row['bonus/malus']
    meanRejetCO2 += row['rejetC02(g/km)']
    meanCoutEnergie += row['coutEnergie']

meanBonusMalus = meanBonusMalus/21
meanRejetCO2 = meanRejetCO2/21
meanCoutEnergie = meanCoutEnergie/21

co2=co2.append({'marque' : 'Dacia' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Daihatsu' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Fiat' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Ford' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Honda' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Lancia' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Saab' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
co2=co2.append({'marque' : 'Seat' , 'bonus/malus' : meanBonusMalus,
                'rejetC02(g/km)' : meanRejetCO2,'coutEnergie': meanCoutEnergie } , ignore_index=True)
```

Maintenant que nous avons configuré la variable co2, nous importons les valeurs du fichier catalogue :

```
catalogue = pd.read_csv('Catalogue.csv')
catalogue
```

	marque	nom	puissance	longueur	nbPlaces	nbPortes	couleur	occasion	prix
0	Volvo	S80 T6	272	tr�s longue	5	5	blanc	False	50500
1	Volvo	S80 T6	272	tr�s longue	5	5	noir	False	50500
2	Volvo	S80 T6	272	tr�s longue	5	5	rouge	False	50500
3	Volvo	S80 T6	272	tr�s longue	5	5	gris	True	35350
4	Volvo	S80 T6	272	tr�s longue	5	5	bleu	True	35350
...	...	...	...	...	...	...	...	...	...
265	Audi	A2 1.4	75	courte	5	5	noir	False	18310
266	Audi	A2 1.4	75	courte	5	5	rouge	False	18310
267	Audi	A2 1.4	75	courte	5	5	blanc	True	12817
268	Audi	A2 1.4	75	courte	5	5	rouge	True	12817
269	Audi	A2 1.4	75	courte	5	5	blanc	False	18310

Nous pouvons maintenant fusionner les valeurs des deux tableaux par rapport à la marque :

```
fusion = pd.merge(catalogue, co2, on='marque')
fusion
```

	marque	nom	puissance	longueur	nbPlaces	nbPortes	couleur	occasion	prix	bonus/malus	rejetC02(g/km)	coutEnergie
0	Volvo	S80 T6	272	tr�s longue	5	5	blanc	False	50500	0.0	42.0	72.0
1	Volvo	S80 T6	272	tr�s longue	5	5	noir	False	50500	0.0	42.0	72.0
2	Volvo	S80 T6	272	tr�s longue	5	5	rouge	False	50500	0.0	42.0	72.0
3	Volvo	S80 T6	272	tr�s longue	5	5	gris	True	35350	0.0	42.0	72.0
4	Volvo	S80 T6	272	tr�s longue	5	5	bleu	True	35350	0.0	42.0	72.0
...	...	...	...	...	...	...	...	...	...	...	...	...
265	Audi	A2 1.4	75	courte	5	5	noir	False	18310	-2400.0	26.0	191.0
266	Audi	A2 1.4	75	courte	5	5	rouge	False	18310	-2400.0	26.0	191.0
267	Audi	A2 1.4	75	courte	5	5	blanc	True	12817	-2400.0	26.0	191.0
268	Audi	A2 1.4	75	courte	5	5	rouge	True	12817	-2400.0	26.0	191.0
269	Audi	A2 1.4	75	courte	5	5	blanc	False	18310	-2400.0	26.0	191.0

270 rows × 12 columns

Puis nous avons plus qu'a sauvegarder les résultats :

```
fusion.to_csv('catalogueCO2.csv')
```

Voici un extrait du résultat obtenu :

```
marque,nom,puissance,longueur,nbPlaces,nbPortes,couleur,occasion,prix,bonus/malus,rejetC02(g/km),coutEnergie
0,Volvo,S80 T6,272,tr�s longue,5,5,blanc,False,50500,0.0,42.0,72.0
1,Volvo,S80 T6,272,tr�s longue,5,5,noir,False,50500,0.0,42.0,72.0
2,Volvo,S80 T6,272,tr�s longue,5,5,rouge,False,50500,0.0,42.0,72.0
3,Volvo,S80 T6,272,tr�s longue,5,5,gris,True,35350,0.0,42.0,72.0
4,Volvo,S80 T6,272,tr�s longue,5,5,bleu,True,35350,0.0,42.0,72.0
5,Volvo,S80 T6,272,tr�s longue,5,5,gris,False,50500,0.0,42.0,72.0
6,Volvo,S80 T6,272,tr�s longue,5,5,bleu,False,50500,0.0,42.0,72.0
7,Volvo,S80 T6,272,tr�s longue,5,5,rouge,True,35350,0.0,42.0,72.0
8,Volvo,S80 T6,272,tr�s longue,5,5,blanc,True,35350,0.0,42.0,72.0
9,Volvo,S80 T6,272,tr�s longue,5,5,noir,True,35350,0.0,42.0,72.0
10,Volkswagen,Touran 2.0 FSI,150,longue,7,5,rouge,False,27340,-1714.0,23.0,96.0
11,Volkswagen,Touran 2.0 FSI,150,longue,7,5,gris,True,19138,-1714.0,23.0,96.0
```

Comparaison avec l'architecture numéro Une

Il est intéressant de comparer cette architecture à l'architecture numéro une que nous avions mise en place lors du projet rendu le 31 janvier 2021. Il semble évident que l'architecture numéro une (stockage HDFS : fichiers stockés sur SQL developer puis analyse R) est plus simple à mettre en place. Toutefois, l'architecture numéro deux, est plus semblable à celles mises en place dans les entreprises. En effet, il y est plus simple de modifier les data lakes en ajoutant des données directement via les différents intermédiaires (mongo, kvstore…)

Vous pouvez trouver le rapport et les scripts concernant l'architecture une (projet du 31 janvier) à l'adresse suivante :
https://github.com/zazacito/Projet_Data_Science