



UNIVERSITÉ
CÔTE D'AZUR



TPA - RAPPORT

Projet Big Data Analytics : Analyse de la Clientèle d'un
Concessionnaire Automobile pour la Recommandation de
Modèles de Véhicules

(Par Nicolas PASQUIER pour la partie Analyse de Données, Gabriel MOPOLLO et Sergio SIMONIAN pour la partie Gestion des Données,
Marco WINKLER pour la partie Data Visualisation)

Réalisé par :

Pierre Adams David

Mackpayen Prince Divin

Nshimiye Judicaël

Oumar Alami

Table des matières

Introduction.....	4
Gestion de données.....	5
I. Contexte du Projet.....	5
II. Données Fournies.....	5
III. Environnement de Travail.....	6
IV. Architecture.....	7
IV.1. MongoDB.....	8
IV.2. Hadoop/HDFS.....	11
IV.3. Oracle NoSQL.....	13
IV.4. Hive.....	14
Visualisation des données.....	16
I. Environnement de travail.....	16
II. Exigences de la data visualisation.....	17
II.1. Exigences fonctionnelles.....	17
II.2. Exigences non fonctionnelles.....	17
III. Architecture.....	17
IV. Les indicateurs clés de performances retenues pour la visualisation.....	19
V. Les différents types de visualisations des données réalisés.....	20
V.1. Les charts.....	20
V.2. Les rapports.....	20
V.3. Les dashboards.....	21
V. Répartition des tâches.....	21
VI. Problèmes rencontrés et solutions apportées.....	24
Analyse de données.....	25
I. Importation des données depuis Hive.....	25
II. Importation des données manuellement sur R.....	27
II.1. Fichier Clients.csv.....	28
II.2. Fichiers Immatriculations.csv.....	39
III. Les ensembles d'apprentissage/testes.....	40
III.1. Voiture par catégorie.....	41
III.2. Fichiers catalogue.csv.....	42
III.3. Création des ensembles.....	43

III.4. Classifieurs et Modele de prediction	45
Conclusion	48
Annexe	49

Introduction

Les Big Data sont un terme utilisé pour décrire les données massives et complexes qui sont de plus en plus fréquentes dans les entreprises et les organisations de tous les secteurs. Ces données peuvent inclure des informations structurées, telles que les données de bases de données, ainsi que des données non structurées, telles que les images, les vidéos et les messages sur les réseaux sociaux. Les Big Data offrent de nombreuses opportunités pour les entreprises et les organisations, notamment la possibilité de comprendre les tendances et les comportements des consommateurs, d'améliorer les opérations et les processus, et de prendre des décisions plus informées. Cependant, ils posent également des défis, notamment en matière de stockage, de traitement et de sécurité des données. Les technologies de traitement de Big Data, comme Hadoop et Spark, ont été développées pour aider les entreprises à gérer ces défis et à tirer parti des avantages des Big Data. La maîtrise de ces différentes technologies est vraiment primordiale pour grimper plus aisément en compétence en entreprise. De ce fait, nous pouvons dire sans ambiguïté que notre projet interdisciplinaire qui consiste à développer une solution big data complète est une aubaine pour nous pour développer les compétences nécessaires dans le domaine et être mieux aguerri pour affronter le marché du travail à la fin de notre cursus.

Tout au long de notre document, nous allons faire présenter nos différentes approches dans le cadre de la gestion de données, la visualisation des données et l'analyse de donnée avant de finir par une petite synthèse de notre travail.

Gestion de données

I. Contexte du Projet

Nous avons été contactés par un concessionnaire automobile afin de l'aider à mieux cibler les véhicules susceptibles d'intéresser ses clients.

Pour cela, il met à notre disposition :

- Son catalogue de véhicules
- Son fichier clients concernant les achats de l'année en cours
- Un accès à toutes les informations sur les immatriculations effectuées cette année
- Une brève documentation des données
- Un vendeur (voir son interview ci-dessous)

Le but étant de proposer au concessionnaire un moyen afin :

- Qu'un vendeur puisse en quelques secondes évaluer le type de véhicule le plus susceptible d'intéresser des clients qui se présentent dans la concession
- Qu'il puisse envoyer une documentation précise sur le véhicule le plus adéquat pour des clients sélectionnés par son service marketing (voir ci-dessous)

II. Données Fournies

Les données mises à notre disposition sont les fichiers CSV suivants :

- CO2.csv, contenant toutes les marques et modèles des voitures du catalogue avec plus de détails comme l'émission CO2, le coût d'énergie, bonus/malus pour les taxes.
- Catalogue.csv, le catalogue des véhicules avec comme colonnes (Marque, Nom, Puissance, Longueur, NbPlaces, NbPortes, Couleur, Occasion, Prix)
- Immatriculation.csv, contenant les informations sur les immatriculations effectuées cette année avec comme colonnes (Immatriculation, Marque, Nom, Puissance, Longueur, NbPlaces, NbPortes, Couleur, Occasion et Prix)
- Client.csv, contenant les informations sur les achats de l'année en cours, avec des colonnes comme (Age, Sexe, Taux, Situation Familiale, NbEnfantsAcharge, 2^e voiture, Immatriculation)
- Marketing.csv, contenant les informations sur les clients sélectionnés par le service Marketing, avec comme colonnes (Age, Sexe, Taux, SituationFamiliale, NbEnfantsAcharge, 2^e Voiture).

III. Environnement de Travail

Pour la réalisation de ce projet, nous avons opté pour une utilisation d'un environnement distribué, un environnement qui permet de réaliser un travail centralisé.

En premier lieu, on a sur un serveur local de l'un de nous, mais ce dernier s'est avéré ne pas être à la hauteur de supporter la charge des différentes bases de données utiles à la réalisation du projet (2 CPU & 4 Go RAM).

Ensuite, on a utilisé l'image vagrant fourni par M. Simonian, en modifiant la configuration pour que la machine puisse être accessible par tous mais en raison d'une faible connectivité réseau ainsi que l'incompatibilité des différentes versions des bases de données et de l'OS, cette seconde méthode n'a pas non plus abouti.

On a donc décidé de louer un VPS (Virtual Private Server) de 4 CPU et 16 Go de RAM chez un Cloud Provider (OVH), sur lequel on a installé manuellement les différentes bases de données et les installations/configurations nécessaires pour pouvoir travailler ensemble.

Les scripts d'installations et configurations sont ceux qu'a faits M. Simonian pour la machine virtuelle vagrant que nous avons modifié pour les adapter à nos besoins.

Ils se trouvent ici : <https://github.com/adamspd/TPA/tree/main/vps>

```
tpa@vps-cde2b818:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:             Little Endian
CPU(s):                 4
On-line CPU(s) list:   0-3
Thread(s) per core:     1
Core(s) per socket:     1
Socket(s):              4
NUMA node(s):          1
Vendor ID:              GenuineIntel
CPU family:             6
Model:                  60
Model name:             Intel Core Processor (Haswell, no TSX)
Stepping:               1
CPU MHz:                2394.454
BogoMIPS:               4788.90
Virtualization:         VT-x
Hypervisor vendor:      KVM
Virtualization type:    full
L1d cache:              32K
L1i cache:              32K
L2 cache:               4096K
L3 cache:               16384K
NUMA node0 CPU(s):     0-3
```

```
tpa@vps-cde2b818:~$ free -mh
              total        used        free      shared  buff/cache   available
Mem:           15G          5.2G          8.7G          936K          1.4G          9.8G
Swap:           0B           0B           0B
```

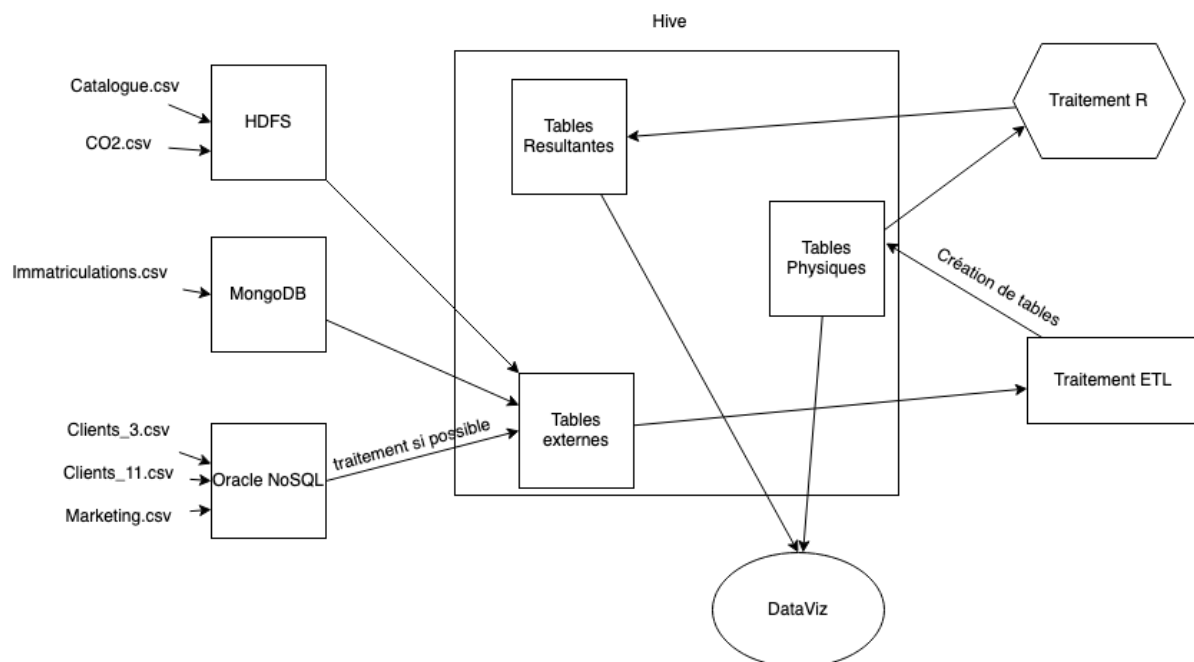
```
tpa@vps-cde2b818:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.6 LTS
Release:        18.04
Codename:       bionic
tpa@vps-cde2b818:~$
```

IV. Architecture

Dans notre projet, on utilise 3 bases de données, MongoDB, Oracle NoSQL et Hadoop/HDFS comme Base de données source et Hive comme datalake. En dessous de Hive, on a utilisé MySQL v8.

L'installation se faisant en suivant ce tutoriel <https://www.devart.com/dbforge/mysql/how-to-install-mysql-on-ubuntu/> et en l'adaptant à nos besoins. Il fallait aussi télécharger le connecteur `mysql-connector-java.jar` pour le copier dans les librairies de Hive.

L'architecture prévue peut se représenter ainsi :



On importe les fichiers CO2.csv et Catalogue.csv dans HDFS, Immatriculations.csv dans MongoDB et Client.csv & Marketing dans Oracle NoSQL.

IV.1. MongoDB

La version 4.4 de MongoDB n'étant pas bien supportée par le connecteur de Hive, M. Simonian nous a conseillé d'utiliser une version antérieure (3.4).

Comme vu dans la section environnement, on travaille sur un serveur Ubuntu 18.04, un des problèmes rencontrés est que la version 3.4 de MongoDB ne se trouve plus dans les répertoires officiels/non officiels d'une version supérieure à 16.04 d'Ubuntu. On a donc décidé d'utiliser un conteneur docker pour pouvoir utiliser la version conseillée.

Pour cela, on a utilisé l'image "dubc/mongodb-3.4" qui se trouve sur docker Hub.

```
tpa@vps-cde2b818:~$ sudo docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
alpine              latest         49176f190c7e   6 weeks ago    7.05MB
dubc/mongodb-3.4    latest         e329f31eaab4   5 years ago    482MB
tpa@vps-cde2b818:~$
```

Ensuite, on l'a lancé avec la commande :

"docker run -d -p 27017:27017 -p 28017:28017 -e AUTH=no dubc/mongodb-3.4" pour que la base de données soit accessible au port 27017 de la machine hôte et désactiver l'authentification.

La figure suivante montre l'importation du fichier "Immatriculations.csv" dans MongoDB, fichier qu'on a d'abord mis à l'intérieur du conteneur grâce à la commande : *docker cp*


```

root@581866678e3e:/# mongoimport -d tpa --type csv --file Immatriculations.csv --headerline
2023-01-08T20:57:37.271+0000 no collection specified
2023-01-08T20:57:37.271+0000 using filename 'Immatriculations' as collection
2023-01-08T20:57:37.276+0000 connected to: localhost
2023-01-08T20:57:40.272+0000 [.....] tpa.Immatriculations 3.52MB/115MB (3.1%)
2023-01-08T20:57:43.283+0000 [#.....] tpa.Immatriculations 7.28MB/115MB (6.3%)
2023-01-08T20:57:46.300+0000 [##.....] tpa.Immatriculations 10.4MB/115MB (9.0%)
2023-01-08T20:57:49.272+0000 [###.....] tpa.Immatriculations 14.0MB/115MB (12.1%)
2023-01-08T20:57:52.272+0000 [####.....] tpa.Immatriculations 17.7MB/115MB (15.3%)
2023-01-08T20:57:55.273+0000 [#####.....] tpa.Immatriculations 21.0MB/115MB (18.2%)
2023-01-08T20:57:58.271+0000 [#####.....] tpa.Immatriculations 24.8MB/115MB (21.5%)
2023-01-08T20:58:01.273+0000 [#####.....] tpa.Immatriculations 28.4MB/115MB (24.6%)
2023-01-08T20:58:04.272+0000 [#####.....] tpa.Immatriculations 31.8MB/115MB (27.6%)
2023-01-08T20:58:07.275+0000 [#####.....] tpa.Immatriculations 35.3MB/115MB (30.6%)
2023-01-08T20:58:10.275+0000 [#####.....] tpa.Immatriculations 38.7MB/115MB (33.6%)
2023-01-08T20:58:13.284+0000 [#####.....] tpa.Immatriculations 42.6MB/115MB (36.9%)
2023-01-08T20:58:16.272+0000 [#####.....] tpa.Immatriculations 46.3MB/115MB (40.2%)
2023-01-08T20:58:19.272+0000 [#####.....] tpa.Immatriculations 50.0MB/115MB (43.3%)
2023-01-08T20:58:22.272+0000 [#####.....] tpa.Immatriculations 53.5MB/115MB (46.4%)
2023-01-08T20:58:25.272+0000 [#####.....] tpa.Immatriculations 56.6MB/115MB (49.1%)
2023-01-08T20:58:28.275+0000 [#####.....] tpa.Immatriculations 60.5MB/115MB (52.5%)
2023-01-08T20:58:31.271+0000 [#####.....] tpa.Immatriculations 64.2MB/115MB (55.7%)
2023-01-08T20:58:34.271+0000 [#####.....] tpa.Immatriculations 68.0MB/115MB (59.0%)
2023-01-08T20:58:37.288+0000 [#####.....] tpa.Immatriculations 71.7MB/115MB (62.2%)
2023-01-08T20:58:40.272+0000 [#####.....] tpa.Immatriculations 75.3MB/115MB (65.3%)
2023-01-08T20:58:43.272+0000 [#####.....] tpa.Immatriculations 78.8MB/115MB (68.3%)
2023-01-08T20:58:46.284+0000 [#####.....] tpa.Immatriculations 82.5MB/115MB (71.5%)
2023-01-08T20:58:49.272+0000 [#####.....] tpa.Immatriculations 86.6MB/115MB (75.1%)
2023-01-08T20:58:52.272+0000 [#####.....] tpa.Immatriculations 90.4MB/115MB (78.4%)
2023-01-08T20:58:55.272+0000 [#####.....] tpa.Immatriculations 94.2MB/115MB (81.7%)
2023-01-08T20:58:58.273+0000 [#####.....] tpa.Immatriculations 97.9MB/115MB (84.8%)
2023-01-08T20:59:01.275+0000 [#####.....] tpa.Immatriculations 102MB/115MB (88.1%)
2023-01-08T20:59:04.275+0000 [#####.....] tpa.Immatriculations 105MB/115MB (91.0%)
2023-01-08T20:59:07.272+0000 [#####.....] tpa.Immatriculations 108MB/115MB (93.9%)
2023-01-08T20:59:10.277+0000 [#####.....] tpa.Immatriculations 112MB/115MB (97.2%)
2023-01-08T20:59:13.132+0000 [#####.....] tpa.Immatriculations 115MB/115MB (100.0%)
2023-01-08T20:59:13.132+0000 imported 200000 documents
root@581866678e3e:/#

```

Enfin, la figure suivante montre l’affichage de “Immatriculations dans MongoDB”.

```

Immatriculations
> db.Immatriculations.find.pretty()
2023-01-08T21:01:33.538+0000 E QUERY [thread1] TypeError: db.Immatriculations.find.pretty is not a function :
@ (shell):1:1
> db.Immatriculations.find().pretty()
{
  "_id" : ObjectId("63bb2e417748c2eac979f0ba"),
  "immatriculation" : "3176 TS 67",
  "marque" : "Renault",
  "nom" : "Laguna 2.0T",
  "puissance" : 170,
  "longueur" : "longue",
  "nbPlaces" : 5,
  "nbPortes" : 5,
  "couleur" : "blanc",
  "occasion" : "false",
  "prix" : 27300
}

{
  "_id" : ObjectId("63bb2e417748c2eac979f0bb"),
  "immatriculation" : "3721 QS 49",
  "marque" : "Volvo",
  "nom" : "S80 T6",
  "puissance" : 272,
  "longueur" : "très longue",
  "nbPlaces" : 5,
  "nbPortes" : 5,
  "couleur" : "noir",
  "occasion" : "false",
  "prix" : 50500
}

{
  "_id" : ObjectId("63bb2e417748c2eac979f0bc"),
  "immatriculation" : "9099 UV 26",
  "marque" : "Volkswagen",
  "nom" : "Golf 2.0 FSI",
  "puissance" : 150,
  "longueur" : "moyenne",
  "nbPlaces" : 5,
  "nbPortes" : 5,
  "couleur" : "gris",
  "occasion" : "true",
  "prix" : 16029
}

{
  "_id" : ObjectId("63bb2e417748c2eac979f0bd"),
  "immatriculation" : "3563 LA 55",
  "marque" : "Peugeot",
  "nom" : "1007 1.4",
  "puissance" : 75,
  "longueur" : "courte",
  "nbPlaces" : 5,
  "nbPortes" : 5,
  "couleur" : "blanc",
  "occasion" : "true",
  "prix" : 9625
}

```

IV.2. Hadoop/HDFS

Pour utiliser Hadoop, nous avons créé deux répertoires nommés "tpa_CO2" et "tpa_Catalogue" pour stocker les fichiers CO2.csv et Catalogue.csv respectivement, en utilisant la commande "Hadoop fs -put <nom_fichier> <destination>".

```
tpa@vps-cde2b818:~$ hadoop fs -ls /
Found 5 items
drwxr-xr-x - ubuntu supergroup 0 2023-01-07 19:37 /secrets
drwxrwxr-x - ubuntu supergroup 0 2023-01-07 19:39 /tmp
drwxr-xr-x - ubuntu supergroup 0 2023-01-07 22:52 /tpa_CO2
drwxr-xr-x - ubuntu supergroup 0 2023-01-07 22:52 /tpa_Catalogue
drwxr-xr-x - ubuntu supergroup 0 2023-01-07 19:22 /user
```

Par la suite, nous avons créé un projet Maven pour permettre la compilation locale des fichiers Java sur IntelliJ. Nous avons utilisé la ligne de commande pour créer le fichier jar, et l'avons ensuite transféré sur le serveur via la commande ci-dessous.

```
adamspierredavid@Adamss-MacBook-Pro target % cd classes
total 16
-rw-r--r--  1 adamspierredavid  staff   1.8K Jan 13 01:19 BonusMalusDriver.class
-rw-r--r--  1 adamspierredavid  staff   3.1K Jan 13 01:19 BonusMalusMap.class
-rw-r--r--  1 adamspierredavid  staff   5.3K Jan 13 01:19 BonusMalusReduce.class
adamspierredavid@Adamss-MacBook-Pro classes % jar -cvf car.jar -C . *.class
added manifest
adding: BonusMalusDriver.class(in = 1824) (out= 939)(deflated 48%)
adding: BonusMalusMap.class(in = 3196) (out= 1410)(deflated 55%)
adding: BonusMalusReduce.class(in = 5380) (out= 2276)(deflated 57%)
adamspierredavid@Adamss-MacBook-Pro classes % scp car.jar adams-vps:/home/ubuntu/tpa
car.jar
100% 5369 25.8KB/s 00:00
```

Dans le cadre des exigences du concessionnaire, il nous a été demandé de corriger certaines erreurs dans le fichier CO2.csv, notamment dans la colonne "Bonus / Malus". Il a été constaté que des valeurs étaient présentées sous la forme "-6 000€ 1" au lieu de "-6 000€". De plus, certaines marques et modèles de véhicules ont présenté des difficultés de traitement en raison de la présence de virgules à l'intérieur de noms encadrés par des guillemets doubles.

Pour remédier au problème de la colonne Bonus Malus, on a créé la classe BonusMalusDriver. Elle contient la méthode principale (main) du programme.

On commence par importer différentes classes nécessaires à Hadoop, telles que Path, Job, FileInputFormat, FileOutputFormat et Configuration. On utilise également la classe GenericOptionsParser pour permettre à Hadoop de lire ses arguments génériques et récupérer les arguments restants dans la variable ourArgs.

La méthode main crée ensuite un objet de configuration Hadoop, puis crée un nouvel objet Job (une tâche Hadoop) en fournissant la configuration et une description textuelle de la tâche. On définit ensuite les classes driver, map et reduce pour cette tâche, ainsi que les types de clefs et de valeurs pour le programme Hadoop.

On définit ensuite les fichiers d'entrée et le répertoire des résultats en utilisant les premiers arguments restants spécifiés par l'utilisateur lors de l'exécution. Enfin, On lance la tâche Hadoop et vérifie si elle s'est déroulée correctement, renvoyant 0 si c'est le cas ou -1 si une erreur est survenue.

Pour la classe Mapper, elle est utilisée pour traiter les données d'entrée et générer des paires clef-valeur pour être utilisées par la suite dans le processus de traitement.

On importe d'abord les classes nécessaires telles que Text, Mapper, CSVParser. On utilise également les classes Java IOException et InterruptedException pour gérer les erreurs d'entrée/sortie et les interruptions.

La méthode map prend en entrée une clef (key), une valeur (value) et un contexte, et elle est responsable de traiter cette valeur pour générer des paires clef-valeur. On utilise la librairie opencsv pour parser les lignes de données, pour éviter d'avoir des erreurs pour les marques de voiture mal formatées, puis on traite les données pour corriger l'erreur de format dans la colonne Bonus Malus. On utilise des expressions régulières pour enlever les signes + et €, et pour supprimer les espaces entre les nombres. On ignore la première ligne en vérifiant si la marque est égale à "Marque / Modele" et on écrit finalement les résultats dans le contexte.

Pour la classe Reducer, on utilise des objets de type Map pour stocker les sommes et les comptes des valeurs de bonus malus et des émissions pour chaque marque de voiture, ainsi qu'une variable pour stocker la somme totale et le compte total pour toutes les marques. Il y a également un objet de formatage de nombre pour formater les résultats en sortie. La fonction reduce() itère à travers toutes les valeurs associées à une clef donnée, extrait les valeurs de bonus, malus et d'émissions et met à jour les compteurs et les sommes appropriés dans les objets Map. La fonction cleanup() est ensuite appelée à la fin de l'exécution de la tâche Hadoop pour calculer les moyennes de bonus malus et d'émissions pour chaque marque de voiture et pour l'ensemble des marques, puis écrit les résultats formatés dans le contexte de sortie pour être récupérés par Hadoop.

Pour pouvoir écrire les nouvelles données dans des tables SQL, on a installé le package Sqoop depuis ce lien : <https://archive.apache.org/dist/sqoop/1.4.7/> et on a aussi téléchargé un connecteur MySQL pour que tout puisse fonctionner. Il fallait de ce fait modifier le Path pour rajouter le /bin du package dans le profil.

IV.3. Oracle NoSQL

Pour Oracle NoSQL, on a d'abord écrit une classe java permettant l'importation des deux fichiers Clients, et le fichier marketing. On a utilisé la même méthode de M. Mopolo, c'est-à-dire, supprimer les tables que l'on va créer si elles sont existantes, les créer, parser les fichiers, puis, rajouter lignes par lignes les données dans les tables.

```
/**
 * La méthode initClientTablesAndData permet :
 * - de supprimer les tables si elles existent
 * - de créer des tables
 * - Insérer des critères
 * - et charger les datas
 */

public void initClientTablesAndData(Tpa client) {
    client.dropTable("CLIENT_SOPHIA2223_TPA_GROUPE_4");
    client.dropTable("MARKETING_SOPHIA2223_TPA_GROUPE_4");
    client.dropTable(tableClient);
    client.createTableClient();
    client.loadClientDataFromFile(dataPath + file_client3);
    client.loadClientDataFromFile(dataPath + file_client11);
}

public void initMarketingTablesAndData(Tpa marketing) {
    marketing.dropTable(tableMarketing);
    marketing.createTableMarketing();
    marketing.loadMarketingDataFromFile(dataPath + file_marketing);
}
```

On a profité au rajout des lignes dans les tables de corriger et d'unifier des données, par exemple, on a remplacé Seul, Seule par Célibataire.

Après le rajout des données dans les tables, on a procédé à la création des tables externes dans Hive.

Le fichier contenant le traitement et le rajout des données dans les tables se trouve ici : <https://github.com/adamspd/TPA/blob/main/nosql/Tpa.java>

IV.4. Hive

Nous avons utilisé Hive en tant que Data Lake, où nous stockons toutes les données provenant de MongoDB, Hadoop ainsi qu'Oracle NoSQL à l'aide des tables externes. Il est possible de visualiser l'état de Hive et de comprendre ce qui se passe en accédant à cette URL : <http://51.222.9.24:10002/>. Si elle est inaccessible, c'est que Hive ne s'est pas lancé. Il est possible de le faire en exécutant ces deux commandes :

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

```
nohup hiveserver2 > /dev/null &
```

Ensuite, si on veut se connecter, on exécute la commande suivante :

```
tpa@vps-cde2b818:~$ beeline
Beeline version 2.3.9 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000
Connecting to jdbc:hive2://localhost:10000
Enter username for jdbc:hive2://localhost:10000: _
```

Ensuite, il est nécessaire de saisir le nom d'utilisateur et le mot de passe défini pour Hive.

Nous avons donc créé autant de tables externes qu'il y a de fichiers de données sources.

Exemple de table externe dans Hive pointant vers MongoDB (fichier Immatriculations.csv).

```
drop table IMMATRICULATION_EXT_TPA;
CREATE EXTERNAL TABLE IMMATRICULATION_EXT_TPA (
ID STRING,
IMMATRICULATION STRING,
MARQUE STRING,
NOM STRING,
PUISSANCE INT,
LONGUEUR STRING,
NBPLACES INT,
NBPORTES INT,
COULEUR STRING,
```

```
OCCASION STRING,
PRIX INT)
STORED BY 'com.mongodb.hadoop.hive.MongoStorageHandler'
WITH SERDEPROPERTIES('mongo.columns.mapping'='{ "id": "_id",
  "immatriculation": "immatriculation", "marque": "marque", "nom": "nom",
  "puissance": "puissance", "longueur": "longueur", "nbPlaces": "nbPlaces",
  "nbPortes": "nbPortes", "couleur": "couleur", "occasion": "occasion",
  "prix": "prix"}')
TBLPROPERTIES('mongo.uri'='mongodb://localhost:27017/tpa.Immatriculations');
```

Exemple de table externe dans Hive pointant vers Hadoop (fichier Co2.csv)

```
CREATE EXTERNAL TABLE CO2_HDFS_H_EXT (ID INT, Marque_Modele STRING
, Bonus_Malus STRING, Rejets_CO2_g_km STRING, Cout_energie STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION 'hdfs:/tpa';
```

Exemple de table externe dans Hive pointant vers Oracle Nosql (Marketing.csv)

```
CREATE EXTERNAL TABLE MARKETING TPA (
ID STRING,
AGE INT,
SEXE STRING,
TAUX INT,
SITUATION_FAMILIALE STRING,
NOMBRE_ENFANTS INT,
DEUXIEME_VOITURE BOOLEAN)
STORED BY 'oracle.kv.hadoop.hive.table.TableStorageHandler'
TBLPROPERTIES (
"oracle.kv.kvstore" = "kvstore",
"oracle.kv.hosts" = "127.0.0.1:5000",
"oracle.kv.hadoop.hosts" = "localhost",
"oracle.kv.tableName" = "MARKETING_SOPHIA2223_TPA_GROUPE_4");
```

A la fin, dans Hive on se retrouve avec 5 tables externes dans la database “default”

```

+-----+
| database_name |
+-----+
| default       |
+-----+
1 row selected (0.673 seconds)
0: jdbc:hive2://localhost:10000> show tables;
23/01/09 10:08:29 INFO ql.Driver: Compiling command(queryId=ubuntu_20230109100829_3f3c68d1-a226-446f-ba40-85bcdebd8190): show tables
23/01/09 10:08:29 INFO metastore.HiveMetaStore: 26: get_database: @hive#default
23/01/09 10:08:29 INFO HiveMetaStore.audit: ugi-ubuntu ip=unknown-ip-addr cmd=get_database: @hive#default
23/01/09 10:08:29 INFO ql.Driver: Semantic Analysis Completed (retrial = false)
23/01/09 10:08:29 INFO ql.Driver: Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:tab_name, type:string, comment:from deserializer)], properties:null)
23/01/09 10:08:29 INFO exec.ListSinkOperator: Initializing operator LIST_SINK[0]
23/01/09 10:08:29 INFO ql.Driver: Completed compiling command(queryId=ubuntu_20230109100829_3f3c68d1-a226-446f-ba40-85bcdebd8190); Time taken: 0.067 seconds
23/01/09 10:08:29 INFO rexec.RExecDriver: Execution #1 of query
23/01/09 10:08:29 INFO ql.Driver: Executing command(queryId=ubuntu_20230109100829_3f3c68d1-a226-446f-ba40-85bcdebd8190): show tables
23/01/09 10:08:29 INFO ql.Driver: Starting task [Stage-0:DDL] in serial mode
23/01/09 10:08:29 INFO metastore.HiveMetaStore: 34: get_database: @hive#default
23/01/09 10:08:29 INFO HiveMetaStore.audit: ugi-ubuntu ip=unknown-ip-addr cmd=get_database: @hive#default
23/01/09 10:08:29 INFO metastore.HiveMetaStore: 34: Opening raw store with implementation class:org.apache.hadoop.hive.metastore.ObjectStore
23/01/09 10:08:29 INFO metastore.ObjectStore: ObjectStore, initialize called
23/01/09 10:08:29 INFO metastore.MetaStoreDirectSql: Using direct SQL, underlying DB is MySQL
23/01/09 10:08:29 INFO metastore.ObjectStore: Initialized ObjectStore
23/01/09 10:08:29 INFO metastore.HiveMetaStore: 34: get_tables: db=@hive#default pat=.*
23/01/09 10:08:29 INFO HiveMetaStore.audit: ugi-ubuntu ip=unknown-ip-addr cmd=get_tables: db=@hive#default pat=.*
23/01/09 10:08:29 INFO ql.Driver: Completed executing command(queryId=ubuntu_20230109100829_3f3c68d1-a226-446f-ba40-85bcdebd8190); Time taken: 0.263 seconds
23/01/09 10:08:29 INFO ql.Driver: OK
+-----+
| tab_name      |
+-----+
| catalogue_hdfs_h_ext
| clients_tpa_h_ext
| co2_hdfs_h_ext
| immatriculation_ext_tpa
| marketing_tpa_h_ext
+-----+
5 rows selected (0.393 seconds)
0: jdbc:hive2://localhost:10000>

```

Visualisation des données

I. Environnement de travail

Dans le cadre de la gestion de données, nous avons opté pour la technologie Zoho qui offre un écosystème complet allant du traitement à la visualisation des données. Au sein de cette technologie, l'analyse des indicateurs de performance nécessite l'écriture de requêtes SQL sur les différentes sources de données.

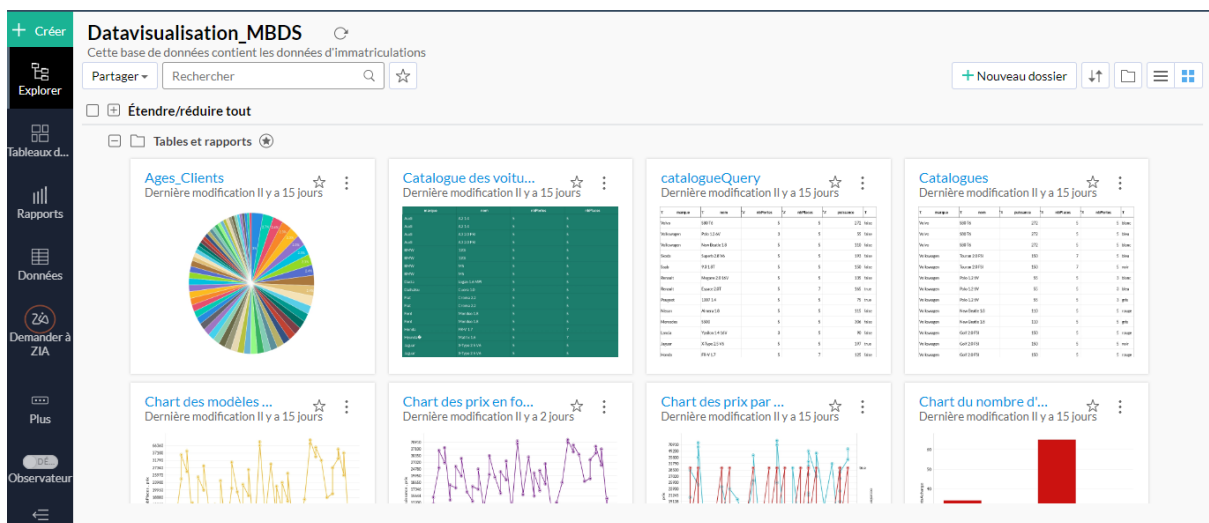


Figure 1: Fenêtre Explorateur de Zoho Analytics

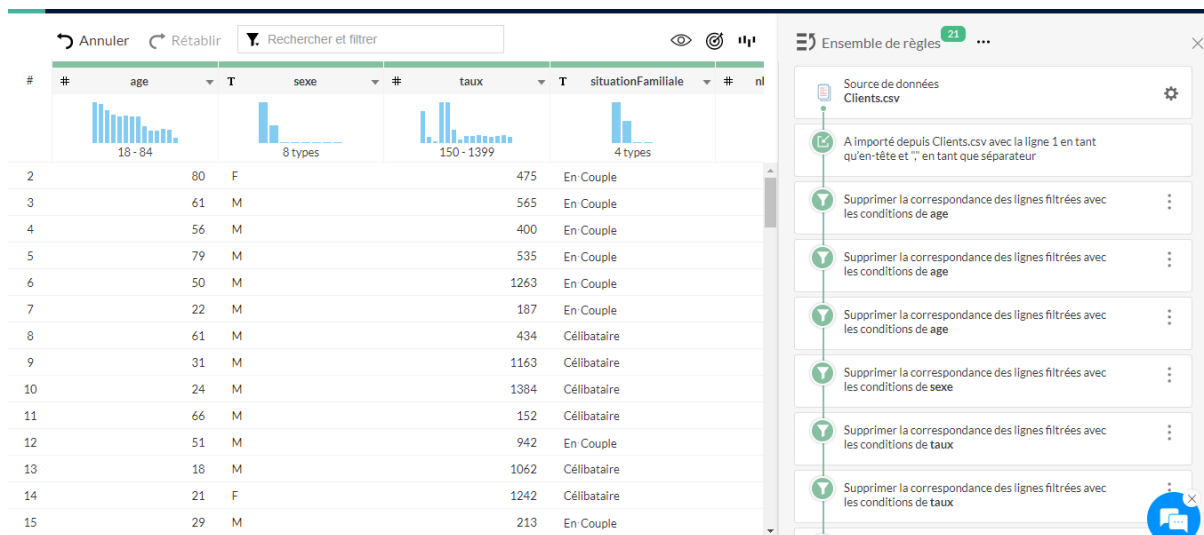


Figure 2: Aperçu de l'environnement de traitement des données sur Dataprep

II. Exigences de la data visualisation

II.1. Exigences fonctionnelles

- ❖ La visualisation doit permettre au concessionnaire d'avoir une complète de sa base de données tout en mettant en avant les KPI qui y sont exploitables.
- ❖ La visualisation doit permettre à son utilisateur de n'avoir que ce qu'il veut voir.

II.2. Exigences non fonctionnelles

- ❖ Zoho doit mettre moins de 15 secondes pour renvoyer les graphes,
- ❖ Les charts doivent être lisible et ne doivent pas comprendre plus de trois couleurs,

III. Architecture

L'architecture de notre projet comprend deux parties principales. La première partie concerne la création des jeux de données associés aux traitements des données, et la seconde partie est liée à l'exploitation des données.

La création des jeux de données et leurs traitements ont été réalisés à l'aide de Zoho Dataprep, qui est un outil puissant qui remplit pratiquement les fonctions d'un ETL, c'est-à-dire qu'il peut surveiller n'importe quelle source de données (base de données et/ou répertoire de fichiers). Grâce à son intelligence artificielle basée sur les règles, nous avons pu mettre en place des modèles d'ensemble de règles applicables de manière spécifique à chaque jeu de données et les

insérer dans les tables Zoho Analytics. En termes simples, nous pouvons dire que l'utilisation de la technologie Zoho Dataprep dans notre projet nous sert pour de petites corrections, car de nombreux traitements ont déjà été effectués dans l'écosystème Hadoop et R.

La seconde partie de notre projet, c'est-à-dire la conception et l'implémentation des différentes méthodes de visualisation, nous a permis de mettre en évidence des indicateurs clés de performance. À partir de ces différents KPI, nous avons pu mettre en place des requêtes SQL qui garantissent les liens entre les différentes structures de notre visualisation. Ces requêtes sont essentielles pour mettre en place des systèmes de filtres au niveau des rapports et des tableaux de bord.

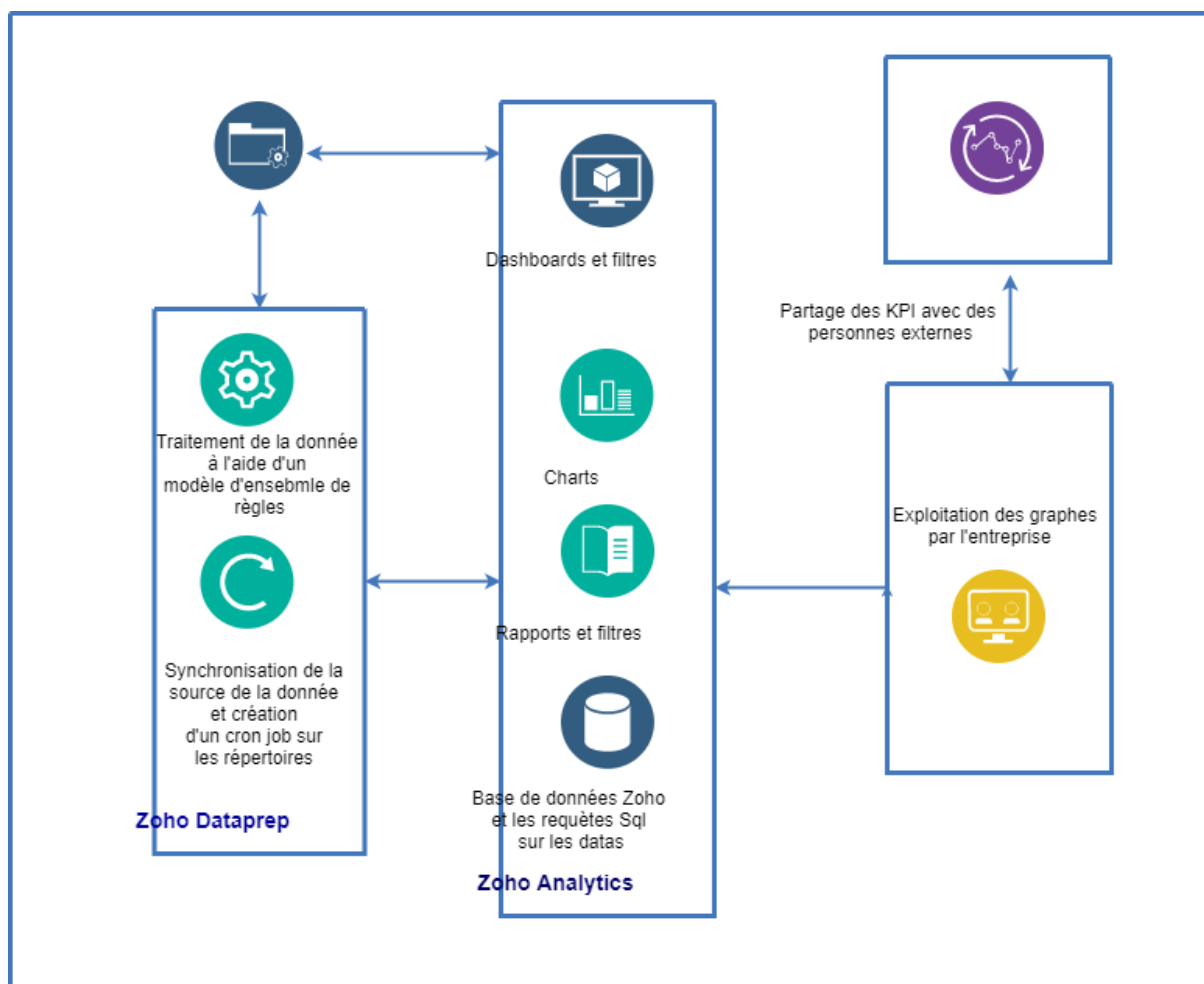


Figure 3: Architecture de la visualisation

IV. Les indicateurs clés de performances retenues pour la visualisation

Le choix des indicateurs de performance pour notre visualisation a été influencé par les attentes des différents utilisateurs de cette dernière. Nous avons donc pris en compte les besoins du concessionnaire et du client. Dans le cas du concessionnaire, nous avons estimé qu'il serait important pour lui et pour son équipe d'avoir un outil qui les aiderait dans la gestion des clients et des stocks, et qu'il serait également important de connaître les véhicules disponibles et leurs performances. Ces indicateurs clés de performance nous ont permis de créer des graphiques tels que :

- ❖ Une charte met en évidence le pourcentage des tranches d'âges des clients de sa base de données, idéal pour orienter sa politique de marketing.
- ❖ Chart qui met en évidence le taux de personnes mariées, célibataires, en couple et divorcées. Cette visualisation aidera dans les décisions stratégiques le concessionnaire à faire une réorientation ou à proposer des produits en fonction de la situation des clients.
- ❖ Chart du nombre d'enfants en fonction de la situation familiale ;
- ❖ Chart des immatriculations en fonction du sexe. Cette charte permettra de connaître le nombre d'immatriculations enregistrées par les hommes et les femmes, important pour les décisions stratégiques.
- ❖ Chart du pourcentage des clients ayant une deuxième voiture.
- ❖ Chart représentant les véhicules disponibles dans le catalogue afin d'aider dans la gestion de stock.
- ❖ Etat des véhicules disponibles par marque.
- ❖ Chart des prix en fonction du modèle et de la puissance du moteur des véhicules.
- ❖ Chart des prix par marque en fonction de l'état du véhicule.
- ❖ Chart des modèles en fonction des prix et du nombre de places.
- ❖ Chart du nombre de véhicules d'occasion par marque.
- ❖ Chart d'estimation totale des véhicules par marque.
- ❖ Chart du nombre de véhicules d'occasion par marque.
- ❖ Graph de consommations du carburant en fonction de la marque.
- ❖ Graph de pollution aux particules en fonction des marques.
- ❖ Graph de pollution à l'oxyde d'azote au kilomètre par marques.

- ❖ Graph de pollution aux particules en fonction des marques.
- ❖ Graph de pollution en dioxyde de carbone au kilomètre par marques.
- ❖ Graph de pollution en hydrocarbure au kilomètre par marques.
- ❖ Graph de pollution en monoxyde de carbone au kilomètre par marques.
- ❖ Graph de types de carburant en fonction de la marque.
- ❖ Graph des boîtes de vitesses en fonction de la marque.
- ❖ Graph des marques les plus polluantes aux particules.
- ❖ Graph des véhicules en fonction de la marque et de la puissance du moteur.
- ❖ Graph du pourcentage des véhicules hybrides

V. Les différents types de visualisations des données réalisés

V.1. Les charts

Dans le cadre de notre projet, nous avons créé vingt-cinq graphiques qui partagent les mêmes tables et requêtes SQL avec les autres éléments de notre visualisation. Ces graphiques ont été développés de manière à être dynamiques et à appliquer les paramètres de filtrage des données. Cela permet une flexibilité dans l'analyse des données et d'avoir une meilleure compréhension des indicateurs clés de performance. Cela permet également de personnaliser les visualisations en fonction des besoins et des utilisateurs finaux.

V.2. Les rapports

Il est vrai que voir les données sous forme de graphiques est utile, mais il peut par ailleurs être nécessaire d'avoir une autre présentation pour les données, c'est pourquoi nous avons décidé de développer trois rapports pour afficher les contenus traités et enrichis de nos bases de données Zoho. Ces rapports permettent d'avoir une vue d'ensemble des données de manière claire et structurée, ce qui facilite la compréhension et l'analyse des données. Il permet aussi de mieux comprendre les données et de les utiliser de manière plus efficace pour les décisions stratégiques.

V.3. Les dashboards

Afin de donner plus de flexibilité à nos utilisateurs dans l'utilisation de notre visualisation, nous avons créé quatre tableaux de bord qui sont reliés et paramétrés sous forme de pages accessibles à travers un menu de navigation. Ces tableaux de bord incluent "Home" qui présente brièvement le contenu de notre projet, suivi de "État des clients", "Catalogue des véhicules" et "Performances des véhicules". Tous ces tableaux de bord sont accessibles de la même manière qu'une page web en cliquant sur les menus. Ils contiennent tous une petite description pour aider les utilisateurs à comprendre leur utilisation, des filtres pour rechercher une information spécifique, des graphiques et des rapports. Cela permet aux utilisateurs de naviguer facilement dans les différentes informations et de visualiser les données de manière claire et structurée, ce qui facilite la compréhension et l'analyse des données.

V. Répartition des tâches

Notre travail a été organisé de manière agile. Pour mettre en pratique les connaissances acquises en classe, nous avons décidé de travailler ensemble sur les tâches atomiques, comme la création de tableaux de bord, et de répartir les tâches de manière équitable. Cela nous a permis de réaliser chacun au moins sept graphiques. En termes de répartition des tâches spécifiques, chaque membre a contribué en réalisant les tâches suivantes :

Noms et prénoms	Tâches réalisées
Pierre Adams David	6 charts créées dont : <ul style="list-style-type: none">• https://analytics.zoho.eu/open-view/178189000000002298• https://analytics.zoho.eu/open-view/178189000000006109• https://analytics.zoho.eu/open-view/178189000000012314• https://analytics.zoho.eu/open-view/178189000000012636• https://analytics.zoho.eu/open-view/178189000000012494• https://analytics.zoho.eu/open-view/178189000000012494

	<p>view/178189000000004628</p> <p>1 rapport :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000005168
Prince Divin Mackpayen	<p>6 charts créées dont :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000005715 • https://analytics.zoho.eu/open-view/178189000000002421 • https://analytics.zoho.eu/open-view/178189000000012602 • https://analytics.zoho.eu/open-view/178189000000012702 • https://analytics.zoho.eu/open-view/178189000000012253 • https://analytics.zoho.eu/open-view/178189000000002353 <p>1 rapport:</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000012035 <p>Vidéo de démonstration :</p> <ul style="list-style-type: none"> • https://youtu.be/EeF90lY7Sr0
Omar Alami	<p>7 charts créées dont :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000005589

	<ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000002615 • https://analytics.zoho.eu/open-view/1781890000000012538 • https://analytics.zoho.eu/open-view/1781890000000012425 • https://analytics.zoho.eu/open-view/178189000000002462 • https://analytics.zoho.eu/open-view/1781890000000012283 • https://analytics.zoho.eu/open-view/178189000000005349
Nshimiye Judicaël	<p>6 charts créées dont :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000002519 • https://analytics.zoho.eu/open-view/178189000000005388 • https://analytics.zoho.eu/open-view/1781890000000012668 • https://analytics.zoho.eu/open-view/1781890000000012370 • https://analytics.zoho.eu/open-view/178189000000005547 • https://analytics.zoho.eu/open-view/178189000000005317 <p>1 rapport :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000002795

Tableau 1: Répartition des tâches au sein de l'équipe

Tâches communes
<p>Requêtes SQL :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000004901 • https://analytics.zoho.eu/open-view/178189000000002118 • https://analytics.zoho.eu/open-view/178189000000008487 • https://analytics.zoho.eu/open-view/178189000000010280 <p>Liens vers le Dashboard :</p> <ul style="list-style-type: none"> • https://analytics.zoho.eu/open-view/178189000000004321

Tableau 2: Tâche commune réalisée

VI. Problèmes rencontrés et solutions apportées

L'implémentation de notre solution de visualisation a rencontré des problèmes variés. Au départ, nous avons été confrontés à des données incohérentes, que nous avons pu traiter progressivement avec nos algorithmes à travers l'écosystème Hadoop. Ces traitements nous ont permis d'obtenir un jeu de données clair dans Hive, où nous avons essayé de connecter Zoho Analytics, mais malheureusement, avec les ressources limitées de notre serveur et les crashes répétitifs causés par le grand nombre de requêtes générées par les filtres des tableaux de bord, nous avons été obligés de laisser cette solution et d'utiliser une connexion à Hive pour mettre en place une solution permettant de lire un répertoire contenant les résultats des traitements. Pour ce traitement, nous avons utilisé toute la puissance de Zoho Dataprep pour mettre en place des tâches planifiées sur les répertoires et appliquer des règles de traitement sur nos données. Au niveau de l'écosystème Zoho Analytics, nous avons rencontré un problème lié aux nombre exorbitant de données, qui peut contribuer à rendre les courbes illisibles ou difficilement compréhensibles. Pour remédier à ce problème, nous avons été obligés de switcher par bloc des datas de même type et appliqué les intervalles à l'échelle de nos graphes.

Analyse de données

R Studio est un logiciel gratuit et multiplateforme qui permet de travailler avec le langage de programmation R, spécialisé dans le traitement de données et l'analyse statistique. Il permet de saisir des commandes en utilisant un langage simple et d'afficher les résultats sous forme de texte ou de graphiques.

Après avoir centralisé toutes les données dans notre data lake, deux méthodes s'offrent à nous pour la récupération de données ont été mises en place pour ce projet :

- Importer manuellement des données dans R
- Utiliser notre data lake

I. Importation des données depuis Hive

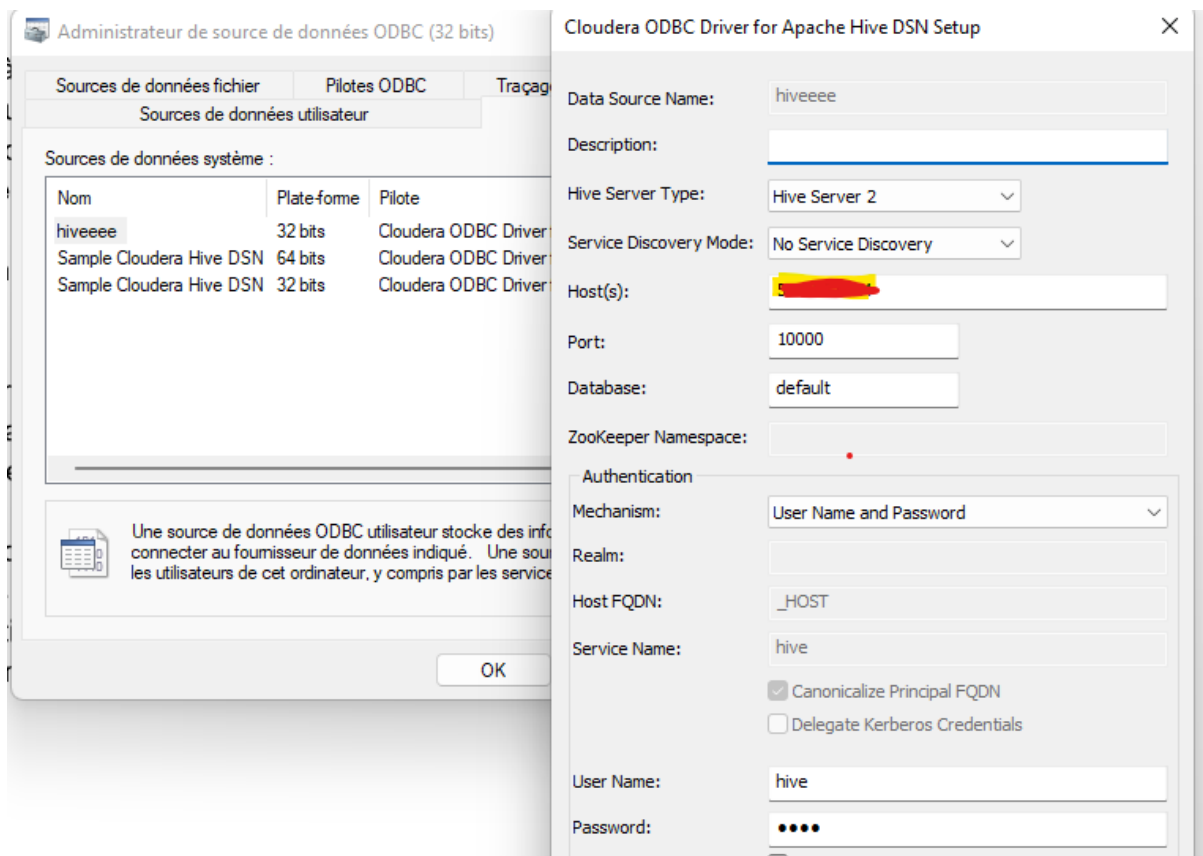
Afin d'importer les données depuis Hive, nous avons utilisé ODBC pour Open Database Connectivity, protocole créant un réseau d'échange d'information entre différentes applications informatiques. Cet échange s'effectue par un processus unique, dans le but de manipuler les bases de données fournies par les systèmes de gestion de base de données (SGBD). On retrouve plusieurs SGBD et chacun possède son mode de fonctionnement.

Pour notre cas, on a utilisé un driver Hive 64 bits fournit par "Cloudera" (ClouderaHiveODBC64). Ensuite, pour faire nos tests, on a simplifié les choses en utilisant le logiciel Excel dans un premier temps.

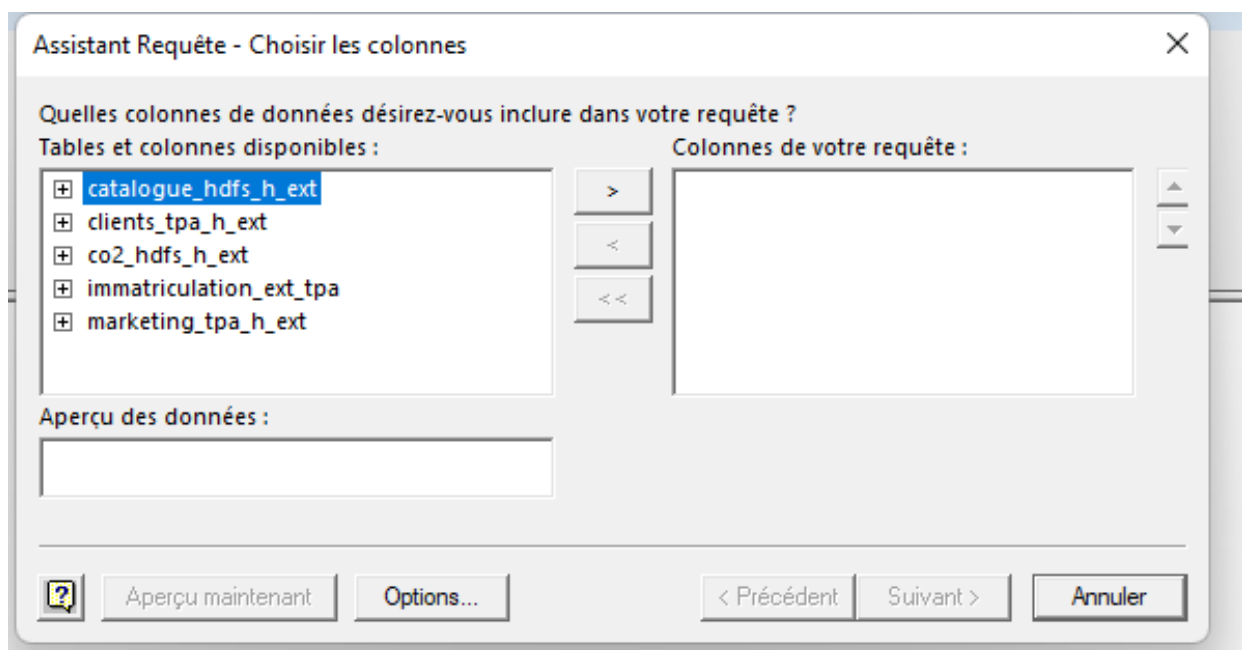
Dans Excel, il faut sélectionner "donnée" sur la barre horizontale d'en haut, ensuite Données externes > À partir d'autres sources > À partir de Microsoft Query; ensuite

On sélectionne le driver qu'on a configuré, l'outil Microsoft "ODBC Data Source Administrator".

Ci-dessous la configuration de notre driver :



Dans Excel, après quelques secondes, on voit afficher le contenu de notre Data lake (Nos 5 tables externes).



Ensuite dans R studio, pour pouvoir utiliser ODBC, on a besoin d'un package nommé 'RODBC'.

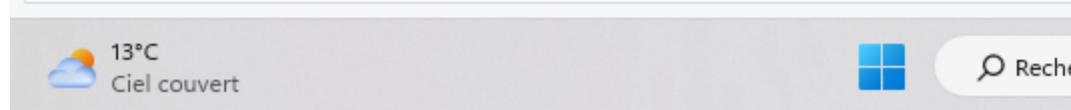
On l'installe et on l'active avec les commandes "install.packages("RODBC") & library(RODBC) et on crée une connexion avec :

connexion <- odbcConnect("hive"), où "hive" est notre driver dans ODBC.

Cette variable "connexion" peut dorénavant être utilisée pour faire des requêtes dans Hive directement.

La figure suivante montre le résultat de la requête "show tables ;" exécuté dans R studio afin d'obtenir une connexion Rstudio -> Hive

```
[1] -1
> sqlQuery(connexion, "show tables;")
      tab_name
1 catalogue_hdfs_h_ext
2 clients_tpa_h_ext
3 co2_hdfs_h_ext
4 immatriculation_ext_tpa
5 marketing_tpa_h_ext
> |
```



Cette méthode exigeant de grandes ressources X, cela entraînera une surcharge de la machine virtuelle. Nous avons décidé de ne pas poursuivre cette technique d'importation afin de s'axer sur une importation manuelle.

II. Importation des données manuellement sur R

Pour importer les données et créer les tables Immatriculation, Client, Marketing et Catalogue, nous allons utiliser la fonction read.csv :

```
# Comma as separator and dot as decimal point by default
read.csv(file,                # File name or full path of the file
          header = TRUE,      # Whether to read the header or not
          sep = ",",          # Separator of the values
          quote = "\"",       # Quoting character
          dec = ".",          # Decimal point
          fill = TRUE,        # Whether to fill blanks or not
          comment.char = "",  # Character of the comments or empty string
          encoding = "unknown", # Encoding of the file
          ...)                # Additional arguments
```

II.1. Fichier Clients.csv

Notre groupe possède deux fichiers clients (clients_3 clients_11), l'idée, c'est de les fusionner pour avoir plus de données, ce qui veut dire plus de précision.

Après avoir fusionné les deux fichiers, on va importer ces derniers sur R :

```
script.R x
Users > omar. > Documents > Master-2 > TPA > Projet > script.R
1  setwd('/Users/omar./Documents/Master-2/TPA/Projet/VersionFinal')
2  # Importation de fichier clients.csv
3
4  clients <- read.csv("../Groupe_TPT_4/clients_merged.csv", header = TRUE, sep = ",", dec = ".")
5
```

Avant de commencer l'analyse, toutes les données sous forme caractère, donc on va convertir les valeurs au bon type :

```
6  # Convertir au bon type
7  clients$age <- as.integer(clients$age)
8  clients$sexe <- as.factor(clients$sexe)
9  clients$taux <- as.integer(clients$taux)
10 clients$situationFamiliale <- as.factor(clients$situationFamiliale)
11 clients$nbEnfantsAcharge <- as.integer(clients$nbEnfantsAcharge)
12 clients$X2eme.voiture <- as.logical(clients$X2eme.voiture)
13
```

En convertissant, on a eu un retour en nous informant qu'il y a des lignes qui contiennent des valeurs NA.

```
> clients$age <- as.integer(clients$age)
Warning message:
NAs introduced by coercion
```

On peut visualiser cela grâce à la fonction `summary()` :

```
> summary(clients)
      age      sexe      taux      situationFamiliare nbEnfantsAcharge X2eme.voiture immatriculation
Min.   :-1.00    M      :136631 Min.   : -1    En Couple   :128225 Min.   :-1.000 Mode :logical Length:200000
1st Qu.:27.00    F      : 58755 1st Qu.: 421   C\xe9libataire: 59323 1st Qu.: 0.000 FALSE:173847 Class :character
Median :41.00    Homme   : 1390 Median : 521   Seule       : 9804 Median : 1.000 TRUE :25756   Mode :character
Mean   :43.68    Masculin : 1381 Mean   : 608   Mari\xe9(e)  : 1340 Mean   : 1.248 NA's :397
3rd Qu.:57.00    F\xe9minin: 633 3rd Qu.: 824   Seul        : 604 3rd Qu.: 2.000
Max.   :84.00    Femme    : 598 Max.   :1399 ?      : 205 Max.   : 4.000
NA's   :402      (Other) : 612 NA's   :412 (Other) : 499 NA's   :360
```

On remarque aussi la présence d'autres valeurs erronées, comme le formatage de mot Célibataire (C\xe9libataire) et d'autres valeurs, c'est pour ça une étape de nettoyage est indispensable pour la réussite de l'analyse.

Les variables qui comportent des valeurs erronées sont :

- Age
- Sexe
- Taux
- SituationFamiliare
- nbEnfantsAcharge

Dans un premier temps, on commencera par la fonction `na.omit()`, qui va nous permettre de supprimer les valeurs NA :

```
13
14 # Suppression des valeurs NA
15 clients <- na.omit(clients)
16
```

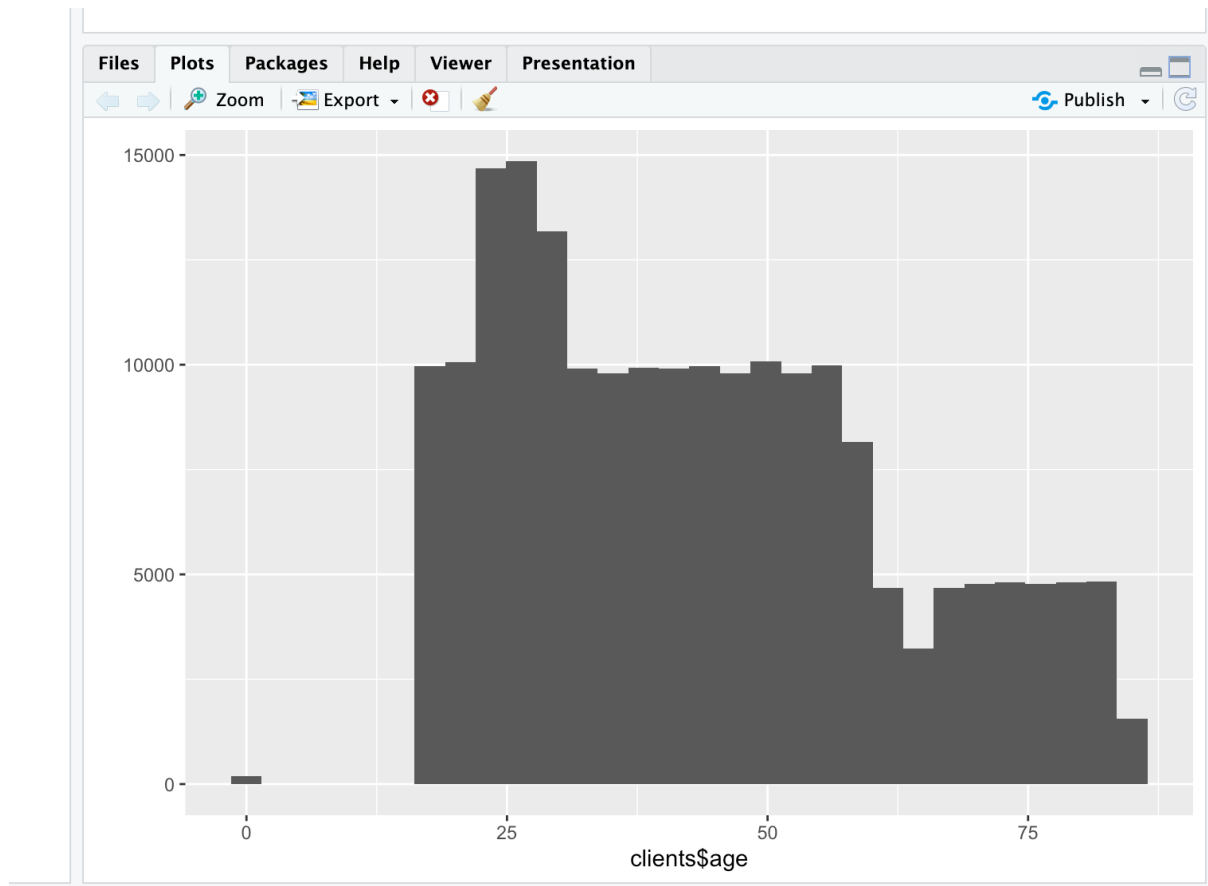
Variable \$Age :

Ensuite, pour déterminer les autres valeurs erronées, on appliquera la fonction `qplot` de la librairie `ggplot2` :

```

16
17 # library ggplot2
18 install.packages("ggplot2")
19 library(ggplot2)
20 qplot(x=clients$age, data = clients)
21

```



On remarque la présence des valeurs négatives (-1)

```

> clients[grepl("-1", clients$age),]
  age  sexe  taux situationFamiliale nbEnfantsAcharge X2eme.voiture immatriculation
973  -1    F  539      En Couple           4          TRUE      8264 KR 53
3207 -1    M  563      En Couple           4          FALSE      5923 KS 76
3502 -1    M  588      C\ne9libataire           0          FALSE      1524 GJ 32
4559 -1    F  512      En Couple           4          FALSE      3396 YJ 37
5343 -1    F  952      En Couple           2          FALSE      9900 XO 82
5368 -1    F 1205      C\ne9libataire           0          FALSE      2241 RO 21

```

Pour retirer ces valeurs non autorisées :

```

22 # Suppression des valeurs -1
23 clients <- clients[!grepl(-1, clients$age),]
24

```

Selon le sujet donné pour ce projet, l'âge doit être entre [18 - 84], pour s'assurer que notre variable respecte cette consigne, on exécutera le code suivant pour récupérer que les lignes qui respectent cet intervalle donné :

```

24
25 # Suppression des valeurs < 18 et > 84
26 clients <- subset(clients, clients$age >= 17 & clients$age <= 84)
27

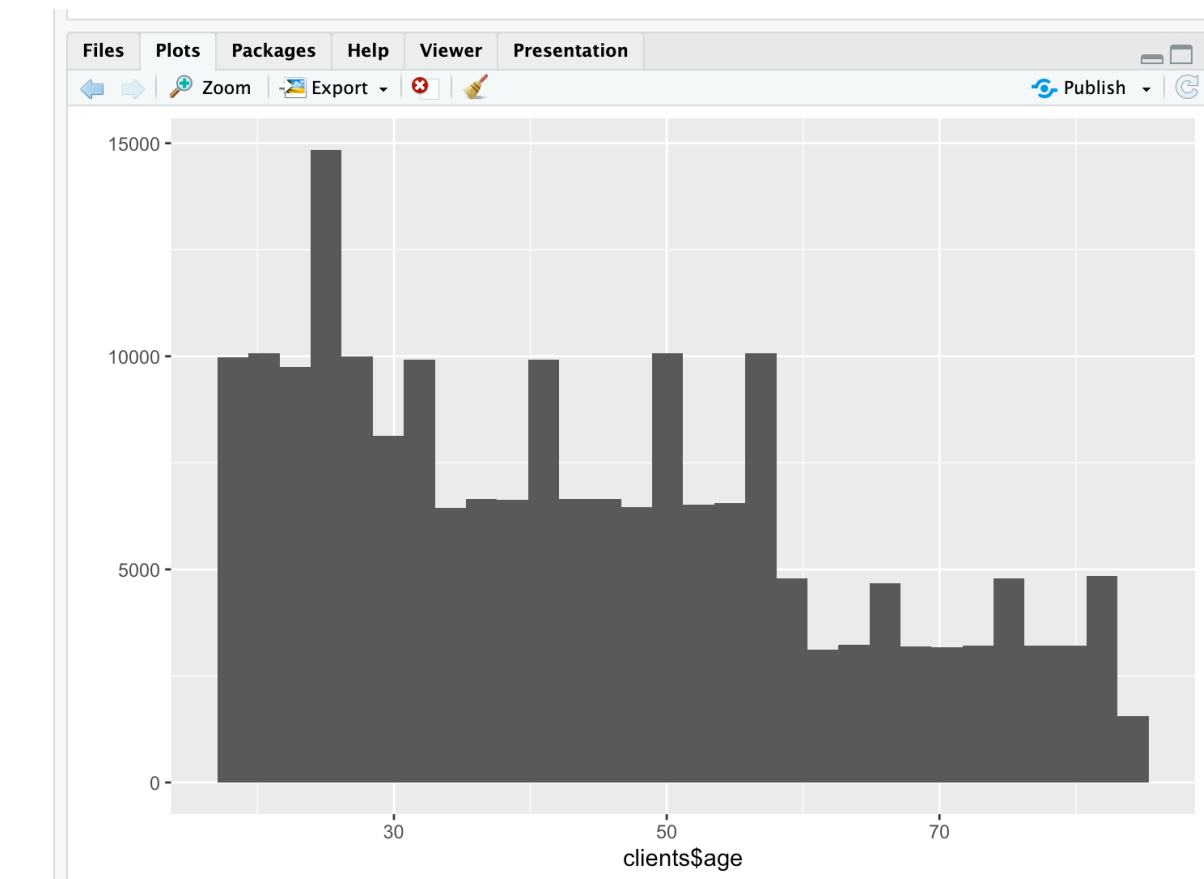
```

Voici des captures d'écran qui nous permettent de s'assurer que la variable \$Âge respecte les consignes données et aussi prête pour être utilisée pour l'analyse.

```

> summary(clients$age)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 18.00   27.00   42.00   43.72   57.00   84.00
>

```

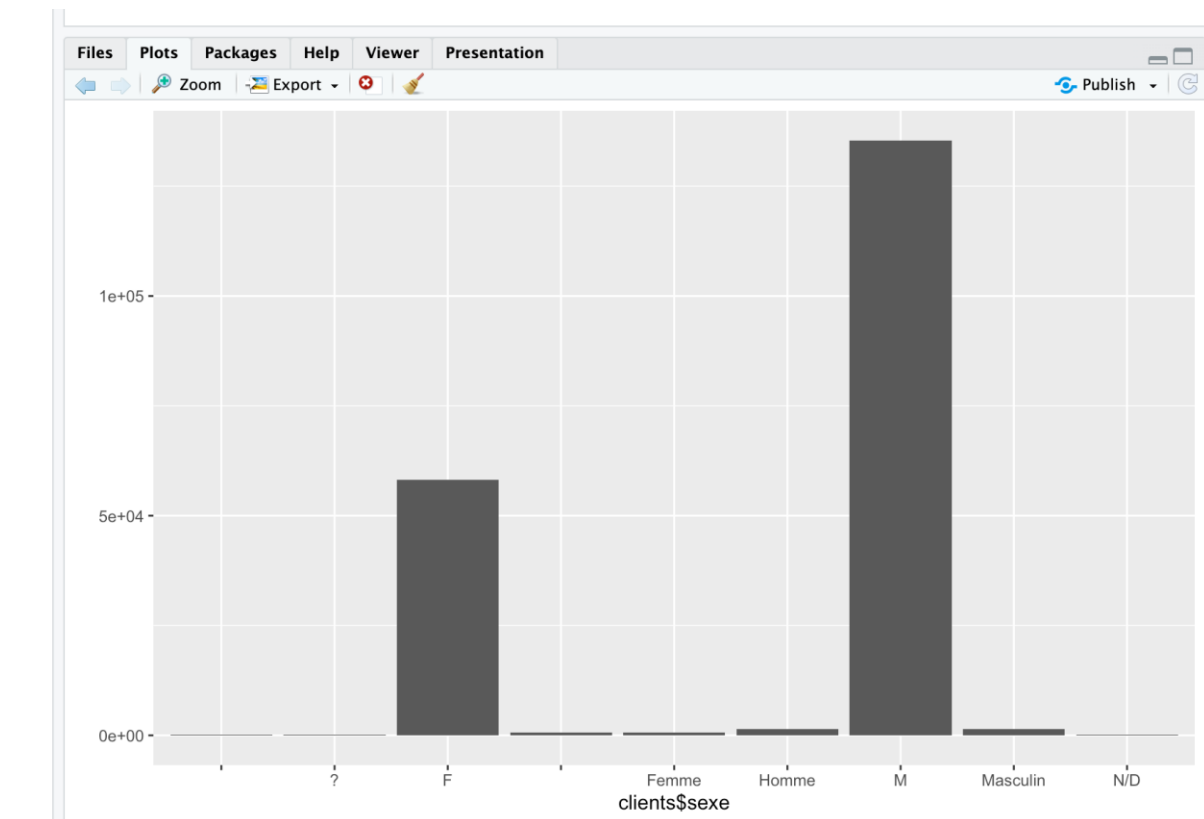


Variable \$Sexe :

Pour cette variable et les autres, on procédera de la même façon qu'on a fait avec la variable \$Age.

D'après la fonction `summary(clients)`, on a bien remarqué la présence de certaines valeurs erronées qui vont influencer notre analyse.

Tout d'abord, on va visualiser ses valeurs sous forme d'un graph avec la librairie ggplot 2 :



D'après le graph la variable contient plusieurs valeurs erronées :

- Des lignes ne contiennent pas de valeur
- ?
- N/D

Pour nettoyer cette variable de ces valeurs, on utilisera le code suivant :

```
35 # Suppression des valeurs erronées
36 clients <- clients[!grepl("N/D", clients$sexe),]
37 clients <- clients[!grepl(" ", clients$sexe),]
38 clients <- subset(clients, clients$sexe != "?")
39
```

On utilise la fonction `summary()` pour vérifier l'absence des outlier dans la variable `$sexe` :

```
> summary(clients$sexe)
      F  F\xe9minin  Femme  Homme      M  Masculin 
58219      624      593    1375 135455     1374
```

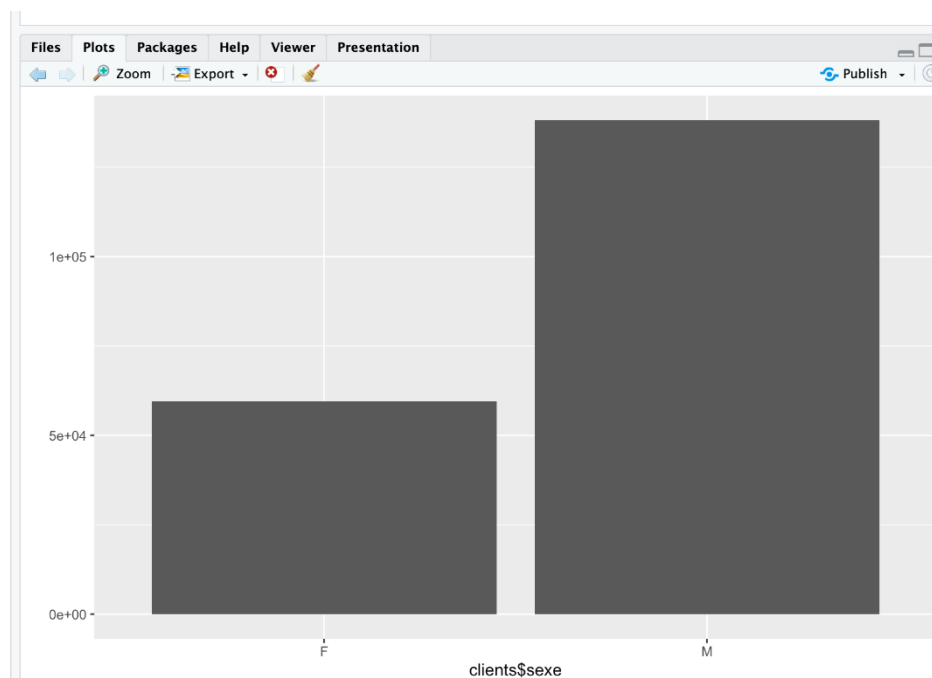
Le nettoyage de cette variable est réussi, mais il reste de mettre les données en bon format, selon les consignes de sujet [M, F] :

- Homme → M
- Masculin → M
- Femme → F
- F\xe9minin → F

```
39
40 # Remettre les données en bon format
41 clients$sexe[clients$sexe == "Homme"] <- "M"
42 clients$sexe[clients$sexe == "Masculin"] <- "M"
43 clients$sexe[clients$sexe == "Femme"] <- "F"
44 clients$sexe[clients$sexe == "F\xe9minin"] <- "F"
45
```

La fonction `levels()` permet de voir les valeurs possibles qu'on trouve dans la variable `$sexe`, on rajoutera aussi un graph qui permet de visualiser la répartition de la variable `$sexe` :

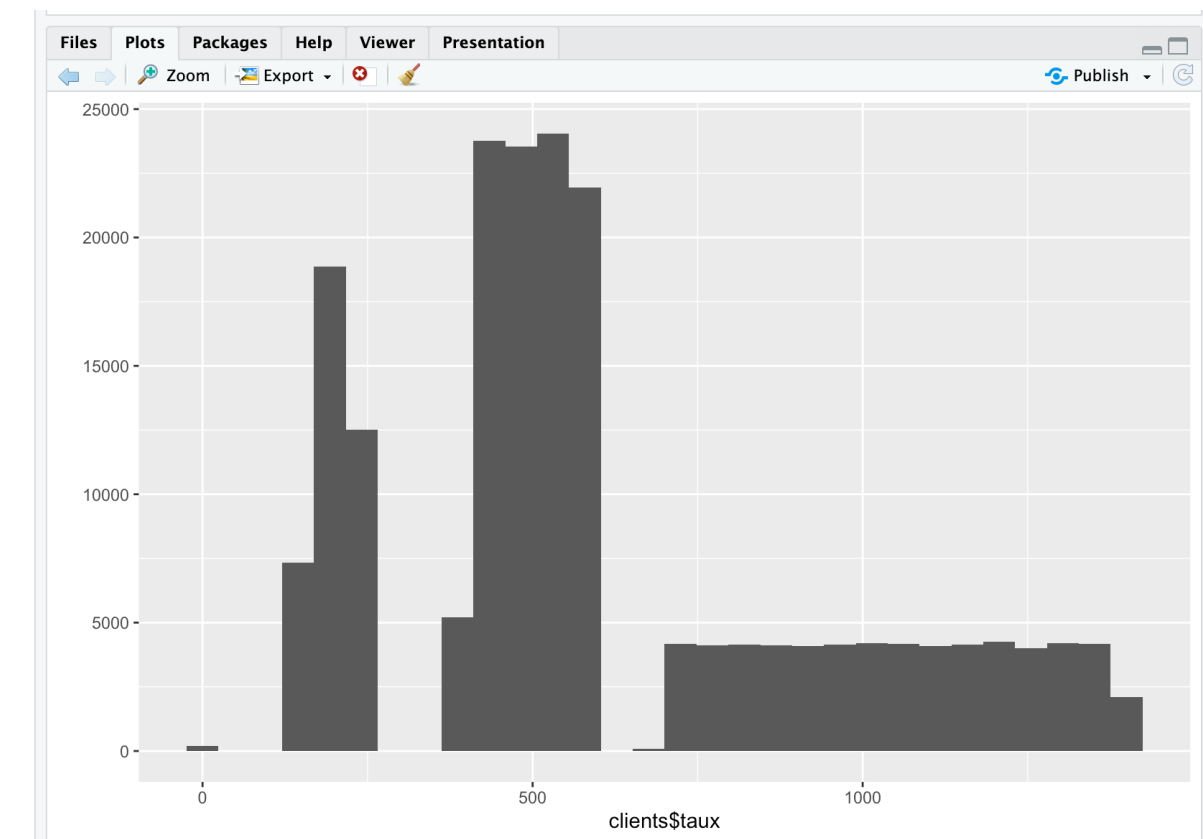
```
> levels(clients$sexe)
[1] "F" "M"
```



Variable \$Taux :

Selon les consignes de sujet, la variable taux correspond à la capacité d'endettement du client en euros (30% du salaire), cette valeur est entre [544, 74185].

En utilisant la librairie ggplot2, on remarque qu'il y a une bonne partie des valeurs inférieures de 544 :



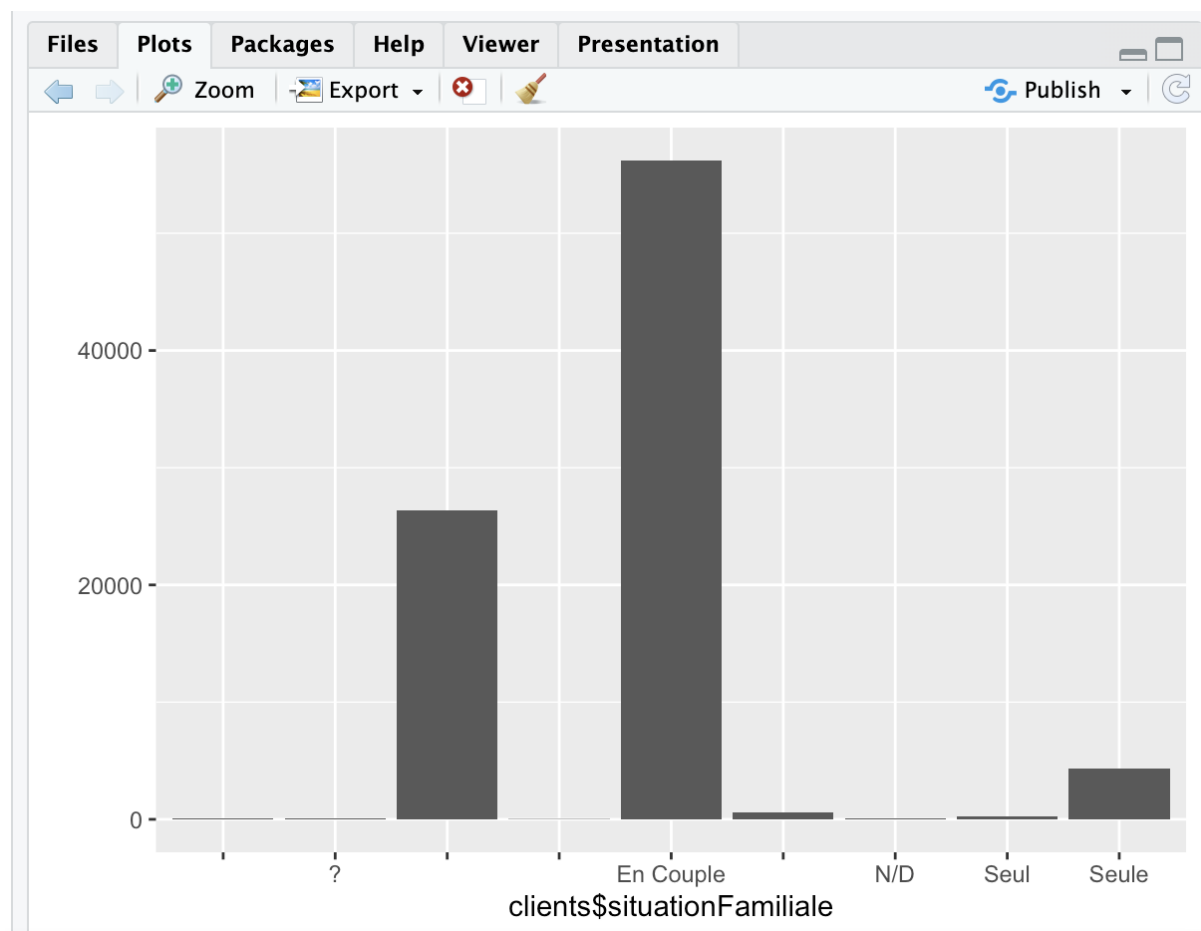
Nous nettoyons la variable de toutes les lignes inférieure de 544 avec la commande suivante :

```
58  
59 # Suppression les lignes qui contient une valeur inférieur de 544  
60 clients <- subset(clients, clients$taux >= 544)  
61
```

```
> summary(clients$taux)
  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
544.0   588.0   888.0   898.4  1144.0  1399.0
```

Variable \$SituationFamiliare :

Pour cette variable, on va procéder de la même façon que \$sexe, tout d'abord, on va visualiser le domaine de valeurs qu'on peut trouver grâce au graphe en dessous, qui nous permet aussi de voir la répartition de cette variable.



De la même manière qu'on a fait pour la variable \$Sexe, on procédera au nettoyage de cette variable en enlevant les lignes qui contiennent des valeurs erronées.

```

68
69 # Suppression des valeurs erronées
70 clients <- clients[!grepl("N/D", clients$situationFamiliare),]
71 clients <- subset(clients, clients$situationFamiliare != " ")
72 clients <- subset(clients, clients$situationFamiliare != "?")
73

```

Ensuite, on peut utiliser la fonction `droplevels()` qui permet d'éliminer les modalités de facteurs qui n'existent plus, et tester si notre manipulation a bien fonctionné avec la fonction `summary()`

```

> clients$situationFamiliare <- droplevels(clients$situationFamiliare)
> summary(clients$situationFamiliare)

```

C\ue9libataire	Divorc\ue9e	En Couple	Mari\ue9(e)	Seul	Seule
26383	36	56236	580	255	4321

La dernière étape pour cette variable, c'est de respecter le domaine de valeurs définie sur le sujet, [Célibataire, Divorcée, En Couple, Marié(e), Seul, Seule], donc on va mettre les données en bon format :

- C\ue9libataire -> Célibataire
- Divorc\ue9e -> Divorcée
- Mari\ue9(e) -> Marie(e)

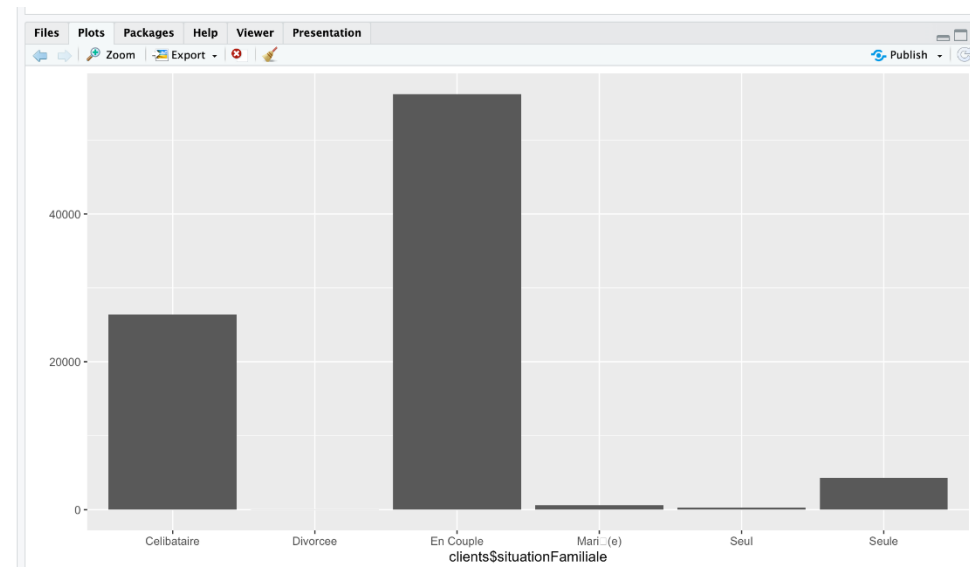
Cette fois-ci, on va utiliser la `library(stringr)` pour pouvoir utiliser la fonction `str_replace`.

```

84
85 # Remettre les données en bon format
86 library(stringr)
87 clients$situationFamiliare <- str_replace(clients$situationFamiliare, "C\ue9libataire", "Celibataire")
88 clients$situationFamiliare <- str_replace(clients$situationFamiliare, "Divorc\ue9e", "Divorcee")
89 clients$situationFamiliare <- str_replace(clients$situationFamiliare, "Mari\ue9(e)", "Marie(e)")
90
91

```

Grâce au graphique généré par `library(ggplot2)` on voit bien le changement des données :



Variable \$nbEnfantsAcharge :

Cette variable est essentielle pour notre analyse, par exemple, le nombre d'enfants à charge décide parfois le modèle de la voiture adapter un client, s'il a au moins trois enfants et c'est première voiture, ça sera une voiture familiale, d'où la nécessité de bien préparer les données de cette variable.

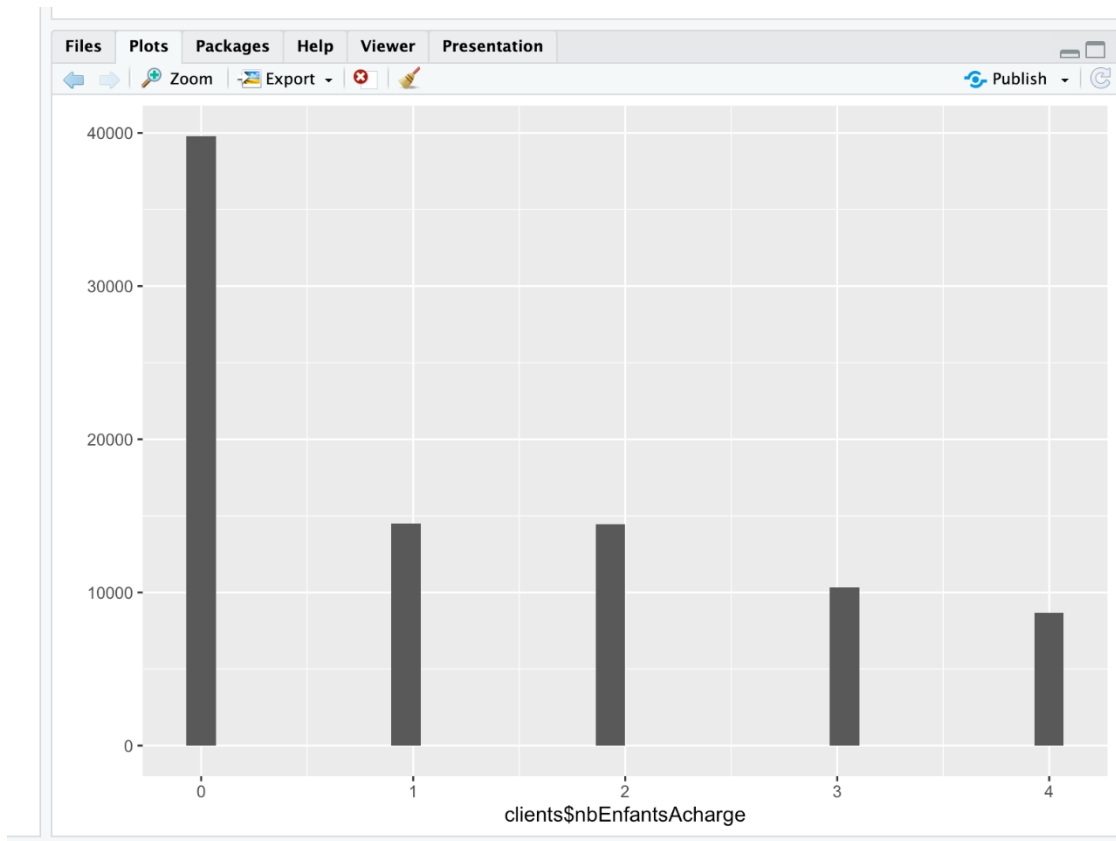
On remarque bien avec la fonction `summary()` la présence des valeurs négatives, mais l'avantage, c'est que toutes les valeurs sont entre $[0, 4]$, ça correspond parfaitement aux consignes sur le sujet.

```
> summary(clients$nbEnfantsAcharge)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.000   0.000   1.000   1.241   2.000   4.000
```

La suppression de ces données qui contiennent des valeurs -1 se fera par le code suivant :

```
100
101 # Suppression des valeurs erronées (-1)
102 clients <- clients[!grepl(-1, clients$nbEnfantsAcharge),]
103
```

On peut bien visualiser l'absence des valeurs erronées avec ce graphe en dessous :



Afin de clôturer le traitement de ce fichier, on peut visualiser la totalité des variables de fichier clients et aussi les types avec la fonction `summary()` :

```
> summary(clients)
```

age	sexe	taux	situationFamiliare	nbEnfantsAcharge	X2eme.voiture	immatriculation
Min. :18.00	F:26406	Min. : 544.0	Celibataire:26361	Min. :0.000	Mode :logical	Length:87746
1st Qu.:27.00	M:61340	1st Qu.: 588.0	Divorcee : 36	1st Qu.:0.000	FALSE:76477	Class :character
Median :42.00		Median : 888.0	En Couple :56197	Median :1.000	TRUE :11269	Mode :character
Mean :43.73		Mean : 898.5	Mari��(e) : 579	Mean :1.242		
3rd Qu.:57.00		3rd Qu.:1144.0	Seul : 255	3rd Qu.:2.000		
Max. :84.00		Max. :1399.0	Seule : 4318	Max. :4.000		

II.2. Fichiers Immatriculations.csv

Comme on a pu faire avec le fichier clients.csv, on procédera de la même manière, en commençant par importer le fichier dans une dataframe :

```
105
106 # Importation de fichier clients.csv
107 immatriculations <- read.csv("../Groupe_TPT_4/Immatriculations.csv", header = TRUE, sep = ",", dec = ".")
108
109
```

Avant de débuter quelque opération sur ce fichier, il est nécessaire de visualiser les variables et ce qu'elles contiennent comme valeurs possibles avec la fonction `summary()` :

```
> summary(immatriculations)
immatriculation  marque          nom          puissance  longueur  nbPlaces  nbPortes  couleur  occasion  prix
Length:20000000 Length:20000000 Length:20000000 Min.   : 55 Length:20000000 Min.   :5  Min.   :3.000 Length:20000000 Length:20000000 Min.   : 7500
Class :character Class :character Class :character 1st Qu.: 75 Class :character 1st Qu.:5  1st Qu.:5.000 Class :character Class :character 1st Qu.: 18310
Mode  :character Mode  :character Mode  :character Median :150 Mode  :character Median :5  Median :5.000 Mode  :character Mode  :character Median : 25970
Mean  :199 Mean  :5  Mean  :4.868 Mean  :35783
3rd Qu.:245 3rd Qu.:5  3rd Qu.:5.000 3rd Qu.: 49200
Max.   :507 Max.   :5  Max.   :5.000 Max.   :101300
```

On remarque bien l'absence des valeurs erronées, mais on peut être confronté à un problème de doublons dans la variable \$immatriculation, la taille de fichier est importante, et par précaution, on va utiliser la fonction `duplicate()` pour pouvoir enlever tous les doublons si on trouve.

```
108
109 # Suppression des doublons
110 immatriculations<-immatriculations[!(which(duplicated(immatriculations$immatriculation))),]
111
```

Il manque juste de mettre les données en bon format :

- `tr\xe8s longue` -> très longue

```
112
113 # Remettre les données en bon format
114 immatriculations$longueur <- str_replace(immatriculations$longueur, "tr\\xe8s longue", "tres longue")
115
```

III. Les ensembles d'apprentissage/testes

Avant de commencer de tester les différents classifieurs ainsi que le modèle de prédiction, il est nécessaire de bien choisir les variables à garder et les variables qui ne seront pas utiles à nos ensembles d'apprentissage et de tests.

Après les précédents traitements qu'on a effectués sur les différents fichiers, on a remarqué qu'on peut regrouper les véhicules par catégorie pour le fichier immatriculations :

- Citadine
- Familiale
- Sport

- Berline

En suite y aussi la possibilité de regrouper les taux par catégorie pour le fichier clients :

- Faible
- Moyen
- Élevé
- Très élevé

III.1. Voiture par catégorie

Pour cette catégorie, on choisit d'utiliser trois variables, puissance, longueur et nombre de places, ces trois facteurs catégorisés une voiture, par exemple, une voiture de catégorie berline a une longueur courte ou moyenne, avec une puissance inférieure à 120 chevaux.

On estime que les autres variables comme couleur, occasion ne sont pas des vrais critères pour le choix de client, certes, on a tous des couleurs qu'on apprécie, mais on fait plus attention au prix et la puissance, pour ces deux dernières, y a une liaison assez forte, plus que la puissance est élevée le prix suit aussi.

Catégorie	Variables
Citadine	Courte Moyenne Puissance ≤ 120
Familiale	Longue Nombres de places = 7 Puissance < 180
Sport	Longue Moyenne Nombres de places ≤ 5 $130 \geq$ Puissance ≤ 300
Berline	Très Longue Longue Nombre de places ≤ 5 $100 \geq$ Puissance ≤ 125

Les variables qu'on cible sont présents dans deux fichiers, immatriculations et catalogue, dans un premier temps, on s'occupe de catalogue, car il nécessite un traitement pour nettoyer les données etc, il va nous permettre de voir si notre raisonnement pour les catégories fonctionne et utilisable.

III.2. Fichiers catalogue.csv

On appliquera les mêmes opérations qu'on utilise pour les autres fichiers :

```

112
113 # Importation de fichier catalogue.csv
114 catalogue <- read.csv("../Groupe_TPT_4/Catalogue.csv", header = TRUE, sep = ",", dec = ".")
115
116 summary(catalogue)

```

marque	nom	puissance	longueur	nbPlaces	nbPortes	couleur	occasion	prix
Length:270	Length:270	Min. : 55.0	Length:270	Min. :5.000	Min. :3.000	Length:270	Length:270	Min. : 7500
Class :character	Class :character	1st Qu.:109.0	Class :character	1st Qu.:5.000	1st Qu.:5.000	Class :character	Class :character	1st Qu.: 16029
Mode :character	Mode :character	Median :147.0	Mode :character	Median :5.000	Median :5.000	Mode :character	Mode :character	Median : 20598
		Mean :157.6		Mean :5.222	Mean :4.815			Mean : 26668
		3rd Qu.:170.0		3rd Qu.:5.000	3rd Qu.:5.000			3rd Qu.: 30000
		Max. :507.0		Max. :7.000	Max. :5.000			Max. :101300

On estime que ce n'est pas nécessaire de montrer toutes les étapes pour ces fichiers, vous pouvez les retrouver dans le fichier Script.R

Création voiture par catégorie

Pour le fichier catalogue :

```

132
133 #Creation voiture par categorie -> catalogue
134 catalogue$voitureParCategories <- ifelse((catalogue$longueur == "courte" | catalogue$longueur == "moyenne")
135 & catalogue$puissance <= 120 ,
136 "citadine",
137 ifelse(catalogue$longueur == "longue"
138 & catalogue$nbPlaces == 7
139 & catalogue$puissance<180,
140 "familiale",
141 ifelse((catalogue$longueur=="longue" | catalogue$longueur=="moyenne")
142 & catalogue$nbPlaces <= 5
143 & catalogue$puissance >=130 & catalogue$puissance <= 300,
144 "sport",
145 ifelse((catalogue$longueur == "tres longue" |
146 (catalogue$longueur == "longue" & catalogue$puissance >=100 & catalogue$puissance <=125 ) )
147 & catalogue$nbPlaces <= 5 ,
148 "berline",
149 "autres"))))
150

```

Pour le fichier immatriculation :

```

150
151 #Creation voiture par categorie -> catalogue
152 immatriculations$voitureParCategories <- ifelse((immatriculations$longueur == "courte" | immatriculations$longueur == "moyenne")
153 & immatriculations$puissance <= 120 ,
154 "citadine",
155 ifelse(immatriculations$longueur == "longue" & immatriculations$nbPlaces == 7 & immatriculations$puissance<180,
156 "familiale",
157 ifelse( (immatriculations$longueur=="longue" | immatriculations$longueur=="moyenne")
158 & immatriculations$nbPlaces <= 5
159 & immatriculations$puissance >=130 & immatriculations$puissance <= 300,
160 "sport",
161 ifelse( (immatriculations$longueur == "tres longue" |
162 (immatriculations$longueur == "longue" & immatriculations$puissance >=100 & immatriculations$puissance <=125 ) )
163 & immatriculations$nbPlaces <= 5 ,
164 "berline",
165 "autres"))))
166

```

III.3. Création des ensembles

Après avoir créé les catégories de voitures qu'on souhaitait, on va pouvoir maintenant fusionner le fichier immatriculation avec le fichier client par la colonne commune entre les deux fichiers avec la fonction merge () :

```

167
168 #Merge deux fichiers immatriculations & clients
169 immatriculations_clients <- merge(immatriculations, clients, by = "immatriculation")
170 summary(immatriculations_clients)
171

```

```

> summary(immatriculations_clients)

```

immatriculation		marque		nom		puissance		longueur		nbPlaces		nbPortes		couleur		occasion	
Length:88059	Length:88059	Length:88059	Length:88059	Min. : 55.0	Length:88059	Min. : 5	Min. : 3.000	Length:88059	Min. : 5	Min. : 3.000	Length:88059	Min. : 3.000	Length:88059	Length:88059	Length:88059	Length:88059	Length:88059
Class :character	Class :character	Class :character	Class :character	1st Qu.: 75.0	Class :character	1st Qu.: 5	1st Qu.: 5.000	Class :character	1st Qu.: 5	1st Qu.: 5.000	Class :character	1st Qu.: 5.000	Class :character	Class :character	Class :character	Class :character	Class :character
Mode :character	Mode :character	Mode :character	Mode :character	Median :197.0	Mode :character	Median :5	Median :5.000	Mode :character	Median :5	Median :5.000	Mode :character	Median :5.000	Mode :character	Mode :character	Mode :character	Mode :character	Mode :character
				Mean :225.9		Mean :5	Mean :4.947		Mean :5	Mean :4.947		Mean :4.947					
				3rd Qu.:306.0		3rd Qu.:5	3rd Qu.:5.000		3rd Qu.:5	3rd Qu.:5.000		3rd Qu.:5.000					
				Max. :507.0		Max. :5	Max. :5.000		Max. :5	Max. :5.000		Max. :5.000					

prix		voitureParCategories		age		sexe		taux		situationFamiliale		nbEnfantsAcharge		X2eme.voiture	
Min. : 7500	Length:88059	Min. :18.00	F:26495	Min. : 544.0	Celibataire:26450	Min. : 0.000	Mode :logical								
1st Qu.: 18310	Class :character	1st Qu.:27.00	M:61564	1st Qu.: 588.0	Divorcee : 36	1st Qu.:0.000	FALSE:76757								
Median : 37100	Mode :character	Median :42.00		Median : 888.0	En Couple :56396	Median :1.000	TRUE :11302								
Mean : 44601		Mean :43.74		Mean : 898.5	Mari0(e) : 583	Mean :1.242									
3rd Qu.: 66360		3rd Qu.:57.00		3rd Qu.:1144.0	Seul : 256	3rd Qu.:2.000									
Max. :101300		Max. :84.00		Max. :1399.0	Seule : 4338	Max. :4.000									

Pour la création des jeux de données pour les deux ensembles, on ne peut pas utiliser toutes les variables présentes sur cette dataframe, on va devoir supprimer celles qui sont inutiles. On a décidé de garder quelques variables pour les utiliser dans différents classifieurs (Categorie, Age, Sexe, Taux, SituationFamiliale, nbEnfantsAcharge et X2eme.voiture).

Le but de ce projet est d'aider à mieux cibler les véhicules susceptibles d'intéresser ses clients, avec cette analyse ça va être de cibler une voiture par catégorie, donc la couleur, occasion et d'autres variables ne seront plus utiles, de même, on a bien remarqué que le prix est lié à la puissance.

Pour cet effet, on utilisera le code suivant pour supprimer les restes des variables :

```

172
173 #Suppression des variable inutiles
174 immatriculations_clients <- immatriculations_clients[,!names(immatriculations_clients)
175 %in% c("immatriculation",
176        "marque",
177        "nom",
178        "puissance",
179        "longueur",
180        "nbPlaces",
181        "nbPortes",
182        "couleur",
183        "occasion",
184        "prix" )]
185

```

Après les précédentes opérations, notre dataframe est prête pour appliquer les différents classifieurs, mais une étape essentielle pour notre modèle de prédiction, c'est de convertir les variables en type "Factor" (Voir le code sur le Script.R).

```

> summary(immatriculations_clients)
voitureParCategories   age      sexe      taux      situationFamiliare nbEnfantsAcharge X2eme.voiture
berline :31290      21      : 2272  F:26495  575      : 541  Celibataire:26450  0:39950  FALSE:76757
citadine:28229      28      : 2270  M:61564  557      : 536  Divorcee   : 36   1:14540  TRUE :11302
sport   :28540      25      : 2261                555      : 534  En Couple  :56396  2:14526
                                18      : 2252                569      : 530  Marié(e)   : 583  3:10365
                                19      : 2216                560      : 529  Seul       : 256  4: 8678
                                24      : 2213                587      : 525  Seule      : 4338
                                (Other):74575          (Other):84864

```

Ensemble d'apprentissage/test :

On va dévisser notre dataframe en deux parties une pour notre ensemble d'apprentissage (70%), et une la deuxième partie pour l'ensemble de testes (30%)

```

197
198 #Creation l'ensemble d'apprentissage + testes
199 immatriculations_clients_training <- immatriculations_clients[1:61422,]
200 immatriculations_clients_test <- immatriculations_clients[61423:88059,]
201

```

III.4. Classifieurs et modèle de prédiction

Au cours de cette dernière partie, nous vous montrerons les résultats des tests que nous avons effectués avec différents classificateurs et déterminerons le plus performant en utilisant des indicateurs tels que la précision, le taux d'erreur et l'AUC. Enfin, nous utiliserons ce classificateur pour établir notre modèle de prédiction.

III.4.1. Classifieurs :

SVM

Pour ce classifieur et les autres, on va suivre les mêmes étapes :

- Apprentissage du classifieur
- Prédiction sur l'ensemble de tests, ça permet de calculer le taux d'erreur et le taux de précision
- Probabilité pour chaque prédiction
- Calcul de l'Auc (en utilisant la fonction multiclass.roc())

```

203
204 #Apprentissage SVM
205 svm_classifieur_1 <- svm(voitureParCategories~., immatriculations_clients_training, probability=TRUE)
206
207 #Prédiction sur l'ensemble de testes
208 svm_classifieur_1_class <- predict(svm_classifieur_1, immatriculations_clients_test, type="response")
209
210 #Probabilité pour chaque prédiction
211 table(svm_classifieur_1_class)
212 table(immatriculations_clients_test$voitureParCategories, svm_classifieur_1_class)
213
214 #Pour la précision, le taux d'erreur et l'AUC, on va présenter les résultats sous forme d'un tableau comparatif,
215 #ou on va voir le meilleur classifieur.
216
217 svm_classifieur_1_prob <- predict(svm_classifieur_1, immatriculations_clients_test, probability=TRUE)
218
219 #Vous pouvez consulter le code qu'on a utilisé pour les autres classifieurs dans le fichier
220 svm_classifieur_1_auc <- multiclass.roc(immatriculations_clients_test$voitureParCategories, svm_classifieur_1_prob)
221 cat (svm_classifieur_1_auc)
222
223 # AUC = 0.899
224

```

	Taux précision	Taux erreur	L'AUC
SVM	0,719	0,281	0.899
C50	0,726	0,274	0.898
Nnet	0,729	0,271	0.902

Le meilleur classifieur est Nnet.

III.4.2. Modèle de prédiction :

Dans cette dernière partie d'analyse, on va appliquer le classifieur Nnet sur le fichier marketing.csv afin de visualiser la prédiction.

Tout d'abord on va convertir les variables au format 'factor', ensuite on appliquera la prédiction grâce la prédiction sur l'ensemble de tests et enfin on affichera notre résultat.

```

270
271 #Modele de prediction
272
273 # Importation de fichier marketing.csv
274 predictionMarketing <- read.csv("../Groupe_TPT_4/Marketing.csv", header = TRUE, sep = ",", dec = ".")
275
276 predictionMarketing$situationFamiliale <- str_replace(predictionMarketing$situationFamiliale, "C\\xe9libataire", "Celibataire")
277
278 # Convertir au bon type
279 predictionMarketing$age <- as.factor(predictionMarketing$age)
280 predictionMarketing$sexe <- as.factor(predictionMarketing$sexe)
281 predictionMarketing$taux <- as.factor(predictionMarketing$taux)
282 predictionMarketing$situationFamiliale <- as.factor(predictionMarketing$situationFamiliale)
283 predictionMarketing$nbEnfantsAcharge <- as.factor(predictionMarketing$nbEnfantsAcharge)
284 predictionMarketing$X2eme.voiture <- as.factor(predictionMarketing$X2eme.voiture)
285
286 # Application de prediction
287
288 class.nnet <- predict(nnet_classifieur_3_class, predictionMarketing)
289 categorie_voiture_predit <- data.frame(predictionMarketing, class.nnet)
290
291 write.table(categorie_voiture_predit, file='categorie_voiture_predit', sep="\t", dec=".", row.names = )
292
293

```

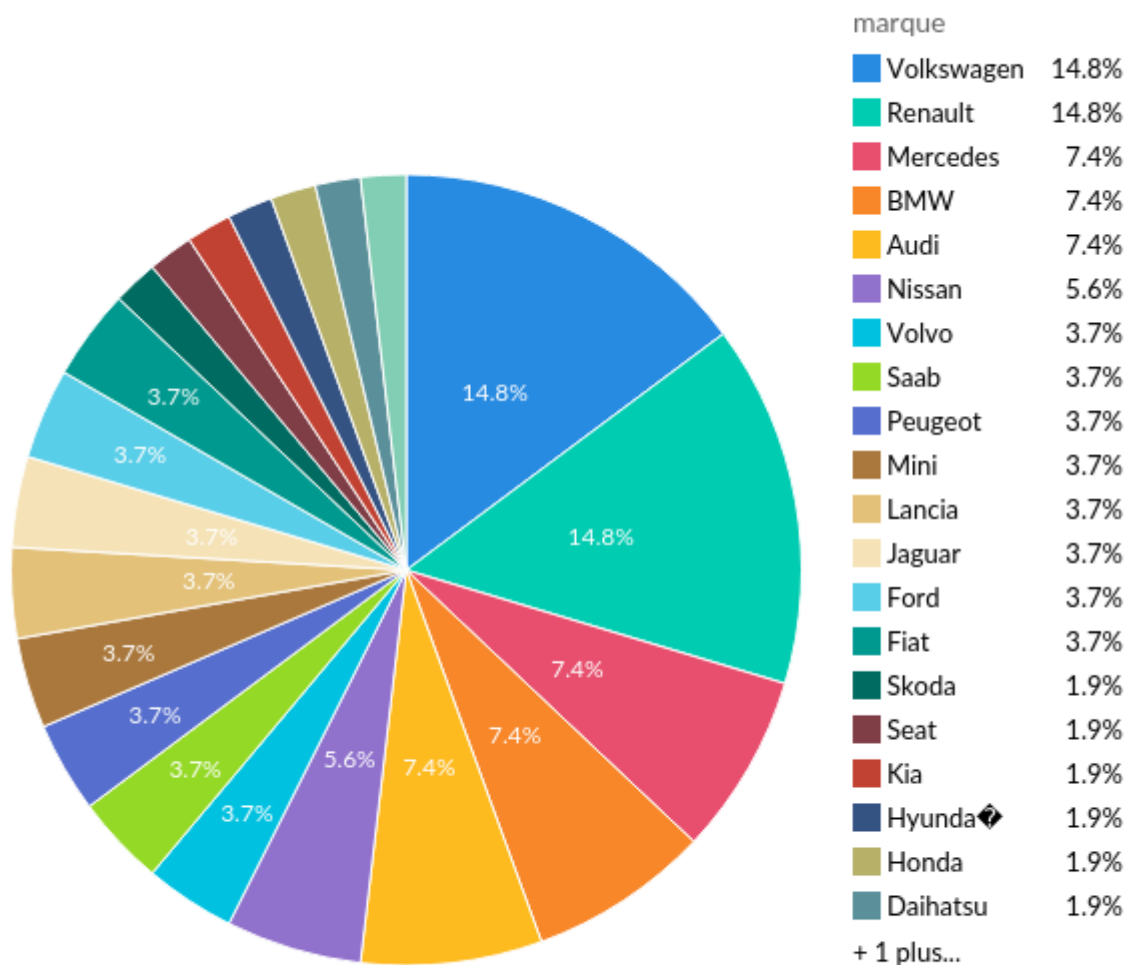

Conclusion

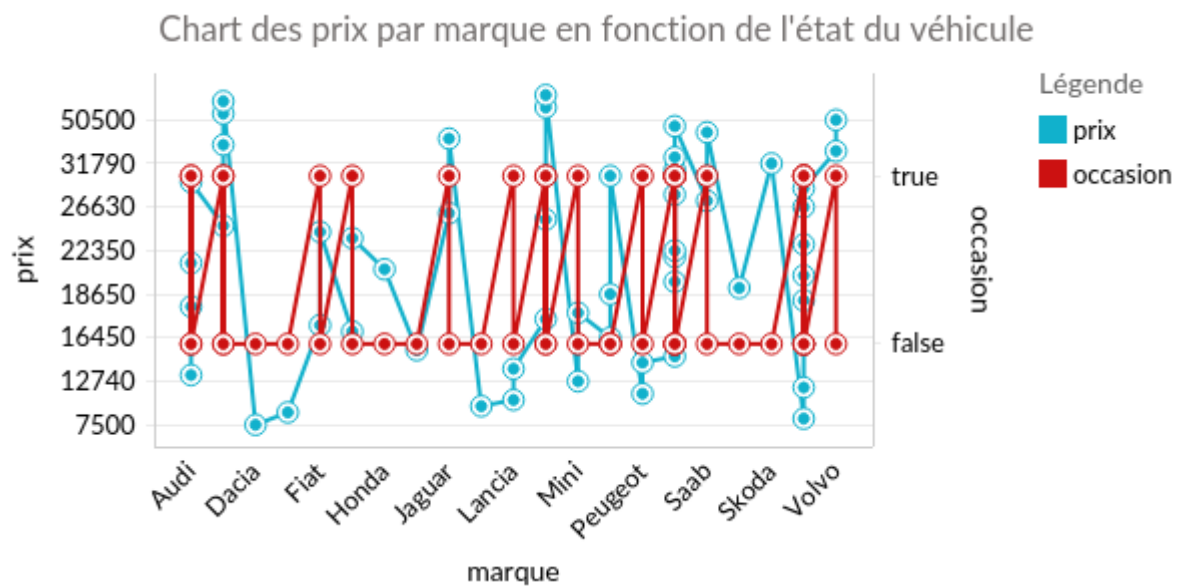
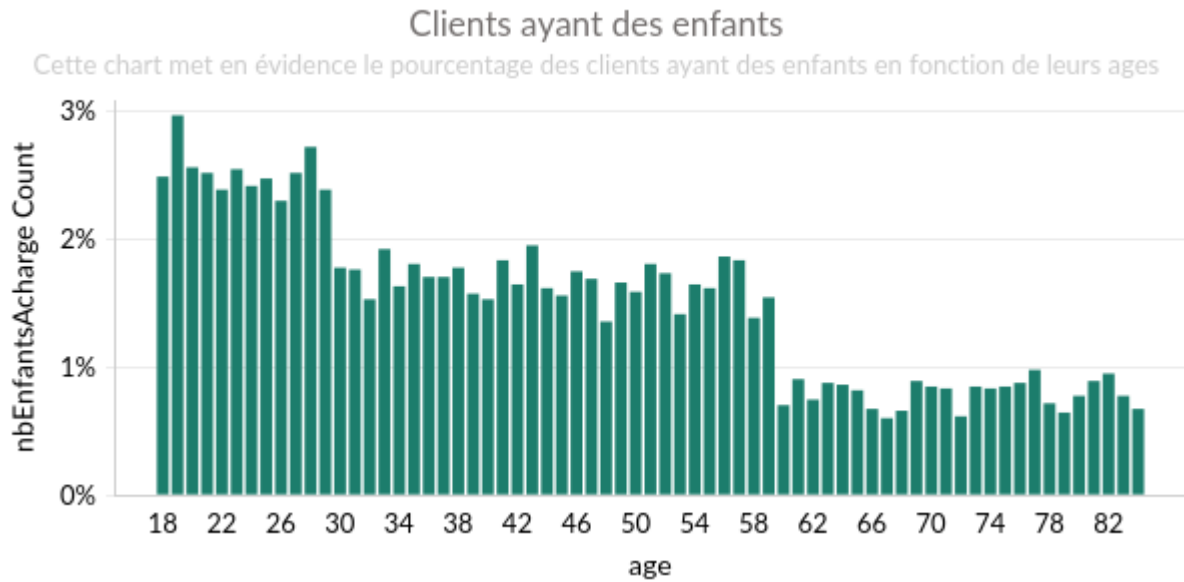
En conclusion, ce projet de Big Data a été une entreprise passionnante et stimulante. Nous avons réussi à collecter, stocker, traiter et analyser des quantités massives de données, ce qui nous a permis de découvrir et améliorer des connaissances apprises en classe. Nous avons également mis en place des outils de visualisation efficaces pour faciliter la présentation et la compréhension des données. Il nous a aussi permis d'appréhender le travail en équipe et toute sa complexité. Nous tenons à remercier nos professeurs et notre école de formation pour leur soutien et leur encadrement tout au long de ce projet, sans eux, cela n'aurait pas été possible.

Annexe

Quelques graphes créés

Véhicules disponibles dans le catalogue





Vidéo de démonstration de la visualisation : <https://youtu.be/EeF90lY7Sr0>