**Rapport de projet Travail Pratique d’Approfondissement (TPA) Big Data et Analyse de données**





**Analyse de la Clientèle d'un Concessionnaire Automobile pour la Recommandation de Modèles de Véhicules**

**Master 2 MIAGE MBDS de l’Université Côte d’Azur(UCA), ITUniversity Madagascar**

Du 15 Mars au 09 Juin 2024

Noms des étudiants :

Andriamahanintsoa Elodie

Rasatarivony Andriharimanga Felana Diamondra

Rasatarivony Andriamalala Sitraka

Ratsimandavana Rindratiana Holiniaina

Nom des enseignants encadreurs:

Mr Gabriel MOPOLO

Mr Sergio SIMONIAN

Mr Nicolas PASQUIE

Résumé

Ce projet vise à analyser la clientèle d'un concessionnaire automobile pour recommander les modèles de véhicules les plus adaptés à chaque client. S'inscrivant dans le cadre d'une amélioration des stratégies commerciales grâce aux technologies de Big Data et de machine learning, le travail englobe la collecte, l'intégration et l'analyse de données provenant de sources variées telles que HDFS, MongoDB et Oracle NoSQL. Un Data Lake a été mis en place pour centraliser ces données, et des modèles prédictifs ont été développés pour fournir des recommandations personnalisées.

Pour atteindre ces objectifs, diverses méthodes ont été utilisées : la construction du Data Lake avec HDFS pour le stockage distribué, MongoDB pour la gestion des données NoSQL, et Hive pour les requêtes SQL ; le chargement des données grâce à des scripts Python et Java pour l'extraction, la transformation et le chargement (ETL) ; le traitement des données via Hadoop MapReduce et Spark pour l'analyse de grandes quantités de données ; et enfin, la modélisation prédictive avec des techniques de machine learning comme Random Forest, Decision Tree, Gradient Boosted Trees et K-means pour le clustering des véhicules.

Les principaux résultats montrent des catégories de véhicules basées sur des caractéristiques telles que la puissance, le prix, la longueur et les rejets de CO2. Les modèles prédictifs ont permis de recommander des véhicules spécifiques à chaque client en fonction de ces catégories, améliorant la pertinence des recommandations.

En conclusion, ce projet a démontré l'efficacité de l'utilisation des technologies Big Data et du machine learning pour optimiser les recommandations de véhicules. Les résultats indiquent une potentialité d'augmentation des ventes et de la satisfaction client grâce à des recommandations plus précises et personnalisées.

Table des matières

[**1. Introduction générale: 7**](#_gjdgxs)

[**2. Présentation du projet 8**](#_30j0zll)

[**3. Répartition du travail en membre du groupe 9**](#_1fob9te)

[**4. Architecture du data lake: 11**](#_3znysh7)

[4.1 Type de Data Lake: 11](#_cyikgnsy0z32)

[4.2 Les sources de données: 11](#_bo33p8fwm2un)

[4.3 L’alimentation du Data Lake: 11](#_4l7g43du2p6v)

[4.4 Les données du Data Lake: 12](#_1ujp4be2aqrf)

[**5. Construction du data lake par étape 13**](#_2et92p0)

[● Compilation : 14](#_rxglcl29zg17)

[● Exécution pour le fichier marketing : 14](#_nyuol6bt5ctb)

[● Exécution pour le fichier client : 14](#_mk08g3ul6cfo)

[1) Rapport sur le Script Python de Chargement des Données de MongoDB vers Hive 15](#_6symw6totfwl)

[● Introduction 15](#_sj673j6ovva5)

[● Objectifs du Script 15](#_gtvcm7h3kctv)

[● Description du Fonctionnement 15](#_xgdzyikemj6x)

[Étape 1 : Extraction des Données depuis MongoDB 15](#_xvd5ej1by3og)

[Étape 2 : Génération des Requêtes d'Insertion 15](#_qlw37zs0mfh3)

[Étape 3 : Insertion des Données dans Hive 15](#_q5r3d3rql9si)

[Étape 4 : Gestion des Batches 16](#_6z9tjp1uz1t)

[● Fonction Principale 16](#_zbe6fu5zcur6)

[● Conclusion 16](#_4q1l3cbe4dwf)

[**6. Hadoop Map Reduce 17**](#_tyjcwt)

[1) Rapport d’Intégration de Données via Spark MapReduce 17](#_h0pol8s7hl87)

[**7. Analyse de données avec des outils de machine learning (R, …) 21**](#_1t3h5sf)

[a) Rapport sur le Script SQL de Nettoyage des Données dans Hive 21](#_4dtsnj6562dy)

[● Introduction 21](#_6u9c0pn475il)

[● Objectifs Principaux 21](#_yrwk3tzer0pr)

[● Description des Transformations et Vues Créées 21](#_mavopen2anvx)

[● Conclusion 22](#_8g0rx9oiecs2)

[b) Modèle de prédiction 23](#_5lxf84r2dxdi)

[● Introduction 23](#_48iry779lh9m)

[● Préparation des Données 23](#_uyytti4wrcy3)

[Initialisation de la session Spark 23](#_3exr1gvpcoec)

[Chargement du Dataset 23](#_2alja9xz0k5x)

[Formatage des Données 23](#_mhwabvt131nv)

[Sous-échantillonnage des Données 24](#_6gpzrkmikmd8)

[Préparation des Caractéristiques 24](#_20e5cfeuhsdl)

[Séparation des Données en Enseignement et Test 24](#_8uhmxjbwdpju)

[Modèles de Machine Learning 24](#_cfycdhgdox8r)

[1. Random Forest 24](#_hxjl6bd45i4a)

[2. Decision Tree 24](#_2o6wn5o10yhd)

[3. Gradient Boosted Trees 24](#_xpqez3c7jjs1)

[4. Support Vector Machine 24](#_qlahw9abaktm)

[5. Logistic Regression 25](#_6vc3ihr3sbwx)

[6. Multilayer Perceptron Classifier 25](#_x2da1hp4m7w1)

[7. Naive Bayes 25](#_lzwzpk4whjdr)

[Évaluation des Modèles 25](#_1zgo1q2jpmst)

[Résultats des Modèles 26](#_6h1s8v9cqj0e)

[Résultat des comparaisons des modèles de prédiction 27](#_j5foh4xab3h2)

[8.2.1.3 Histogramme et nuages de points: 32](#_yb5iw0j5u0h9)

[8.2.1.3 Conclusions de l'analyse exploratoire de données : 33](#_6zkp2kcrt38o)

[**9. Conclusion générale 37**](#_4d34og8)

[**10. Références et Bibliographie 37**](#_2s8eyo1)

[**11. Annexes 37**](#_17dp8vu)

[11.1 Vidéo de présentation de votre projet 37](#_3rdcrjn)

[11.2 Dossier contenant les scripts et programmes de construction du lac de données 38](#_26in1rg)

[11.3 Dossier contenant les scripts et programmes Hadoop Map Reduce 38](#_lnxbz9)

[11.4 Dossier contenant les scripts et programmes d’analyse de données 38](#_1ksv4uv)

Table des images

[Figure 1 Liste membre 9](#_Toc168869549)

[Figure 2 Structure du datalake 12](#_Toc168869550)

[Figure 3 Commande Hadoop 13](#_Toc168869551)

[Figure 4 Mongo Import 13](#_Toc168869552)

[Figure 5 Compiler Java 14](#_Toc168869553)

[Figure 6 CO2 Dataframe 19](#_Toc168869554)

**GLOSSAIRE**

* Algorithme K-means: Méthode de clustering non supervisée qui partitionne un ensemble de données en K clusters en minimisant la variance intra-cluster.

* Apache Hive: Système d'entrepôt de données open-source sur Hadoop pour l'interrogation et l'analyse de grands ensembles de données via un langage de requête SQL-like.
* Apache Spark: Système de traitement distribué open-source pour les charges de travail big data.
* CSV: Format de fichier texte structuré utilisé pour enregistrer des données sous forme de tableau, séparées par des virgules.
* Data Lake: Système de stockage de données centralisé pour gérer, traiter et sécuriser de grandes quantités de données.
* Dataframe: Structure de données avec des lignes et des colonnes, similaire à une base de données ou une feuille de calcul Excel.
* Dataset: Collection organisée de données.
* ETL (Extraction, Transformation et Chargement) : Processus de combinaison de données de sources multiples dans un entrepôt de données central.
* Hadoop : Framework open-source pour le traitement et le stockage distribué de grandes quantités de données.
* HDFS (Hadoop Distributed File System) : Système de stockage principal utilisé par Hadoop.
* JVM (Java Virtual Machine) : Machine virtuelle qui exécute des programmes Java en convertissant le bytecode Java en instructions machine.
* Markdown : Langage de balisage léger pour formater du texte.
* MulticlassMetrics : Classe de la bibliothèque Apache Spark MLlib pour évaluer la performance des modèles de classification multi-classes.
* SQL (Structured Query Language) : Langage de programmation standardisé pour gérer et manipuler des bases de données relationnelles.
* Vue Base de données : Table virtuelle basée sur l'ensemble des résultats d'une instruction SQL.

# Introduction générale:

Selon une étude de McKinsey, les entreprises utilisant les big data pour prendre des décisions stratégiques augmentent leur productivité de 5 à 6 % par an. De plus, une étude de BARC révèle que les entreprises exploitant les big data sont 23 fois plus susceptibles d'acquérir de nouveaux clients et 19 fois plus susceptibles d'être rentables ce qui rend cette approche cruciale dans le monde des affaires modernes.

C’est justement pour ces raisons que dans notre projet, nous allons exploiter l'impact des technologies Big Data sur l'optimisation des stratégies commerciales d'un concessionnaire automobile. Nos objectifs principales sur ce travail incluent :

* **La capacité de recommandation de modèles de véhicules spécifiques** : Analyser les données clients et des véhicules pour recommander les modèles de voiture les plus appropriés à chaque client potentiel.
* **Offrir une amélioration de l'expérience Client** : Utiliser les résultats obtenus pour personnaliser les interactions avec les clients, augmentant ainsi leur satisfaction et leur fidélité.

Pendant ce projet, nos missions principales incluent :

1. **La construction d'un Data Lake** : Collecter, agréger et structurer des données provenant de multiples sources pour créer un référentiel centralisé.
2. **La construction d’un modèle de prédiction de catégories de véhicules** : Utiliser des algorithmes de machine learning sur les données centralisé pour avoir les catégories de voitures les plus adéquates possibles.
3. **L’ application du Modèle de Prédiction aux Données Marketing de l’entreprise:** Utiliser les données sur les clients sélectionnés par le service marketing pour prédire la catégorie de véhicules la plus appropriée pour chaque client, en utilisant le classifieur généré lors de l'étape précédente.

**Annonce du Plan :**

Dans ce rapport, nous allons aborder successivement les sujets suivant:

**Contexte et Objectifs:** Introduction au cadre du projet et présentation des objectifs principaux.

**Détails sur la construction du Data Lake**: Explication détaillée de la mise en place de l'infrastructure de données data lake .

**Hadoop Map Reduce**: Enrichir les données de catalogue dans le data lake avec les données de CO2 qui représentent les performances environnementales des différents véhicules.

**Analyse des données et machine learning:** Clustering, construction de modèle de prédiction et application du modèle de prédiction sur les données correspondants pour prédire la catégorie de véhicules correspondant à chaque client.

**Résultats et Conclusions** - Synthèse des résultats obtenus et conclusion sur l'efficacité et l'impact du projet.

# Présentation du projet

Mise en avant de la criticité éventuelle du projet, des principaux enjeux et principaux risques.

Ce projet de Big Data Analytics vise à optimiser l’expérience client au sein d’un concessionnaire automobile en analysant les données pour recommander les modèles de véhicules les mieux adaptés.

Notre analyse approfondie de la clientèle du concessionnaire automobile , couplée à des techniques

avancées de Big Data Analytics , a permis de recommander des modèles de véhicules personnalisés , améliorant ainsi la satisfaction client et les performances de vente.

# 

# Répartition du travail en membre du groupe



Figure 1 Liste membre

1. Introduction

Dans le cadre de notre projet de Big Data Analytics pour un concessionnaire automobile, nous avons mis en place une répartition structurée des tâches afin de tirer parti des compétences spécifiques de chaque membre de l'équipe. Cette organisation nous a permis de travailler de manière efficiente et collaborative, en assurant une couverture complète de toutes les phases du projet, depuis la création du Data Lake jusqu'à l'analyse des données et la rédaction du rapport final. Chaque membre a apporté une contribution essentielle à la réussite de ce projet, que ce soit par l'importation et l'intégration des données, le développement de modèles prédictifs, ou encore l'évaluation des résultats. La répartition suivante détaille les responsabilités et les contributions de chaque étudiant, soulignant ainsi la synergie et la complémentarité de notre équipe.

1. Contributions des Membres

**Rindratiana** a été un pilier central dans l'importation des données. Elle a assuré le chargement des données dans HDFS et MongoDB, garantissant une base solide pour les analyses ultérieures. En outre, Rindra a rédigé l'introduction générale du rapport, posant ainsi le cadre conceptuel et méthodologique du projet. Elle a également participé activement à l'exploration des données, ce qui a permis d'identifier des tendances et des insights clés. Son implication dans la création des tables externes et l'architecture du Data Lake a été cruciale pour structurer et organiser les données de manière efficace.

**Elodie** a apporté une expertise précieuse dans le développement de programmes NoSQL en Java, facilitant ainsi la manipulation et l'interrogation des données non structurées. Elle a dirigé l'évaluation des modèles prédictifs à l'aide de matrices de confusion, assurant la validité et la précision des prédictions marketing. Son rôle a également été crucial dans la rédaction du rapport, où elle a présenté le projet de manière claire et concise. De plus, Elodie a contribué à la construction du Data Lake en détaillant chaque étape et en collaborant à la création des tables externes.

**Felana** a été chargée de la création des tables Hive et de la préparation des données pour l'analyse, assurant une qualité et une intégrité des données optimales. Sa contribution à l'implémentation des processus MapReduce de Hadoop a permis de traiter efficacement les grands volumes de données. Felana a également co-rédigé l'architecture du Data Lake, détaillant les composants et les flux de données. Sa capacité à formater et à préparer les données a été essentielle pour garantir que les analyses puissent être réalisées sans problème.

**Sitraka** a mené la conception et la mise en œuvre des processus ETL, garantissant une intégration fluide des données provenant de diverses sources. Il a également développé des modèles prédictifs, permettant d'extraire des insights précieux et de faire des recommandations précises. Sitraka a joué un rôle clé dans l'analyse des données, en utilisant des techniques avancées pour identifier des modèles et des tendances. Sa contribution à la rédaction du rapport a inclus des sections détaillées sur les analyses de données et la création de modèles, assurant une compréhension claire et complète des méthodes et des résultats.

1. Synthèse des Tâches

Le projet de Big Data Analytics pour le concessionnaire automobile a été une entreprise collaborative réussie, où chaque membre a apporté des compétences spécifiques et essentielles. **Rindratiana** a établi les bases du projet avec l'importation et l'exploration des données, ainsi que la rédaction de l'introduction générale. **Elodie** a garanti la précision des prédictions marketing grâce à son travail sur les programmes NoSQL et les matrices de confusion, et a contribué de manière significative à la rédaction du rapport. **Felana** a assuré une préparation des données sans faille et a joué un rôle clé dans l'implémentation des processus MapReduce et la création des tables Hive. **Sitraka**, avec ses processus ETL et son développement de modèles prédictifs, a permis une analyse approfondie des données.

Cette synergie a non seulement permis de créer un lac de données robuste et des analyses pertinentes, mais a aussi renforcé notre capacité à travailler efficacement en équipe pour atteindre nos objectifs communs. Le résultat final est un projet bien structuré, détaillé et prêt à fournir des recommandations précises et utiles pour le concessionnaire automobile.

# 

# Architecture du data lake:

## Type de Data Lake:

Le type de lac de données choisi est HiveQL. Le moteur SQL HIVE d’Hadoop est utilisé comme frontal du lac de données, permettant l'accès à toutes les données pour la construction des modèles d'analyse.

## Les sources de données:

Le Data Lake intègre plusieurs sources de données, chacune avec des méthodes spécifiques d'alimentation et d'accès.

* **HDFS (Hadoop Data File System)** :
  + **Fichiers** : co2.csv et immatriculation.csv
  + **Méthode d'importation** : Commande HDFS
  + **Description** : Les fichiers co2.csv et immatriculation.csv sont chargés dans HDFS, une composante essentielle pour le stockage distribué des fichiers.
* **MongoDB** :
  + **Fichier** : catalogue.csv
  + **Méthode d'importation** : Commande mongoimport
  + **Description** : Le fichier catalogue.csv est importé dans MongoDB, une base de données NoSQL flexible et performante.
* **Oracle NoSQL** :
  + **Fichiers** : Marketing.csv et Clients.csv
  + **Méthode d'importation** : Programme Java
  + **Description** : Les fichiers Marketing.csv et Clients.csv sont importés dans Oracle NoSQL, utilisant un programme Java pour l'extraction et le chargement des données.

## L’alimentation du Data Lake:

Le Data Lake est alimenté par une combinaison d'access drivers et d'extracteurs de données.

* **Access Drivers** :
  + **Data File Access Driver** : Utilisé pour accéder aux fichiers HDFS(co2 et immatriculation) sans les déplacer
  + **Database Access Driver** : Utilisé pour accéder aux données de Oracle NoSQL(Marketing et Client) sans les déplacer.
* **Data Extractors** :
  + **ETL Programme en Python** : Utilisé pour extraire, transformer et charger les données Catalogue de MongoDB vers Hive.
  + **MapReduce Spark** : Utilisé pour des opérations complexes sur les données, combinant et traitant les données catalogue et co2.

## Les données du Data Lake:

Le Data Lake contient ces différentes données que l’on va ensuite utiliser comme des données des modèles d’analyses:

* **Tables internes** :

Ce sont: catalogue,marketing,immatriculation,client

* **Tables externes**:

Ce sont: marketing\_ext, immatriculation\_ext, client\_ext, catalogue\_co2\_ext

* **Vues** :

Les vues contiennent les versions des données formatées que l’on peut tout de suite utiliser pour l’analyse des données. Ce sont: marketing\_view, immatriculation\_view, client\_view, catalogue\_co2\_view

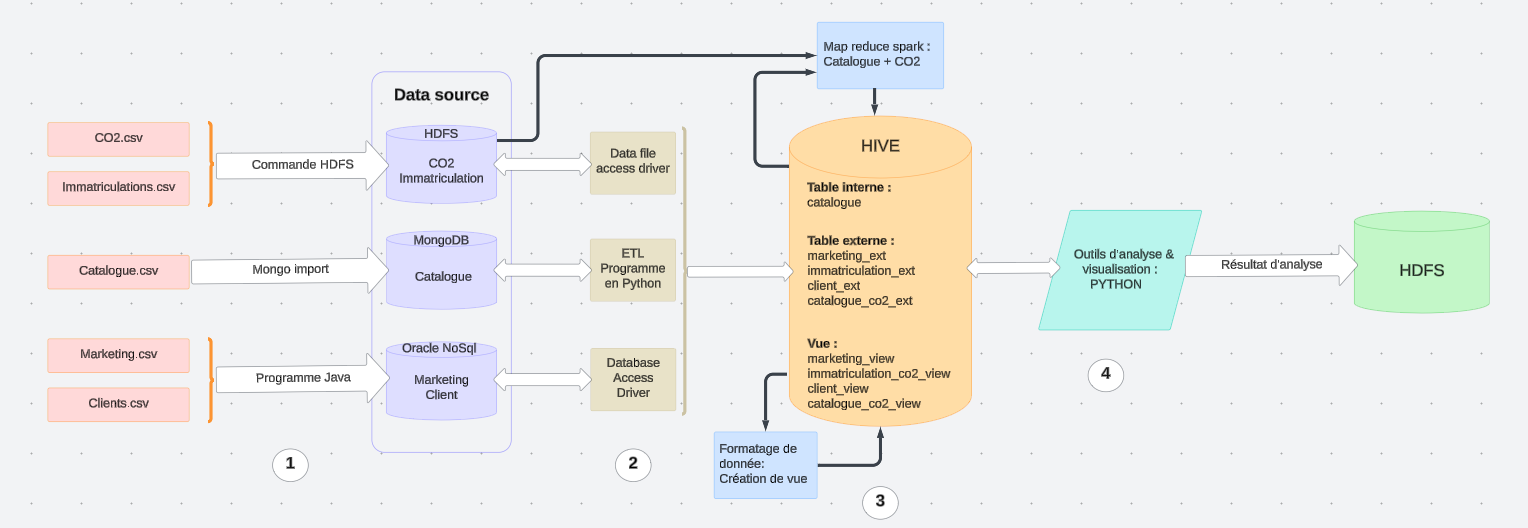


Figure 2 Structure du datalake

# 

# Construction du data lake par étape

Pour expliquer en détail la construction du Data Lake présentée dans l’architecture ci-dessus, voici une description étape par étape de chaque composant :

**Étape 1 : Ingestion des données CSV**

Les sources de données sont les fichiers CSV.

**Fichiers CSV :**

1. **CO2.csv et Immatriculations.csv** : Importés dans *HDFS* via des *commandes HDFS*.   
   **Comment ?**

Par l’utilisation des commandes suivante :

* *hadoop fs -put -f $CO2\_FILE\_PATH /tpa\_groupe\_14/data/co2*
* *hadoop fs -put -f $IMMATRICULATIONS\_FILE\_PATH\_NEW /tpa\_groupe\_14/data/immatriculation*

Ces commandes signifient que les fichiers *CO2.csv* et *Immatriculations.csv*, qui sont stockés localement aux emplacements spécifiés par `$CO2\_FILE\_PATH` et `$IMMATRICULATIONS FILE\_PATH\_NEW` respectivement, sont copiés vers le système de fichiers distribué Hadoop (HDFS), dans des répertoires spécifiques (`/tpa\_groupe\_14/data/co2` pour `CO2.csv` et `/tpa\_groupe\_14/data/immatriculation` pour `Immatriculations.csv`).

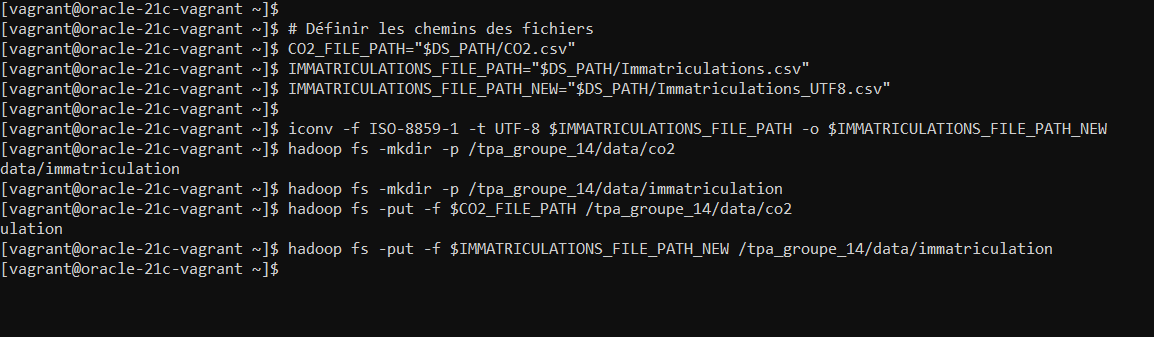


Figure 3 Commande Hadoop

1. **Catalogue.csv** : Importé dans MongoDB via la commande *mongoimport.*

Le fichier Catalogue.csv est importé dans MongoDB grâce à la commande `mongoimport`. Cette commande permet d'intégrer des données provenant de divers formats ou sources externes dans une base de données MongoDB.

Dans ce cas, elle est utilisée pour importer les données contenues dans le fichier CSV spécifié par `$CATALOGUES\_FILE\_PATH\_NEW` dans une base de données nommée `sourceCSV`, en les plaçant dans une collection intitulée `catalogue`.

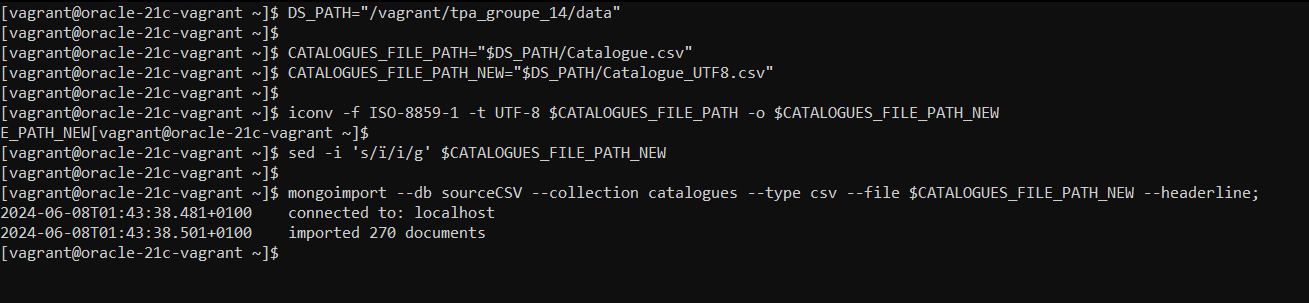


Figure 4 Mongo Import

L'option `--type csv` indique que le fichier source est au format CSV, tandis que `--headerline` assure que la première ligne du fichier CSV, décrivant les noms des colonnes, sera utilisée comme telles pour les noms de champ dans la collection MongoDB."

1. **Marketing.csv et clients.csv** : Ingestion dans Oracle NoSQL via un *programme Java*.

**Comment ?**

En utilisant des commandes qui compilent et exécutent un programme Java (ImportCSVtoNOSQL) pour importer des données de fichiers CSV dans une base de données NoSQL. Elles ajustent également la mémoire de la JVM et spécifient les chemins des fichiers de données et des collections cibles.

#### Compilation :

javac -g -cp $KVHOME/lib/kvclient.jar:$TPT\_HOME $TPT\_HOME/ImportCSVtoNOSQL.java

* **Explication** :

Compile le fichier ImportCSVtoNOSQL.java en incluant les informations de débogage (-g) et en définissant le classpath (-cp) pour inclure kvclient.jar et $TPT\_HOME.

#### Exécution pour le fichier marketing :

java -Xmx256m -Xms256m -cp $KVHOME/lib/kvclient.jar:$TPT\_HOME ImportCSVtoNOSQL $MARKETING\_FILE\_NEW marketing

* **Explication** :

Exécute le programme avec une allocation de 256 Mo de mémoire pour la JVM et importe les données de $MARKETING\_FILE\_NEW dans la collection marketing.

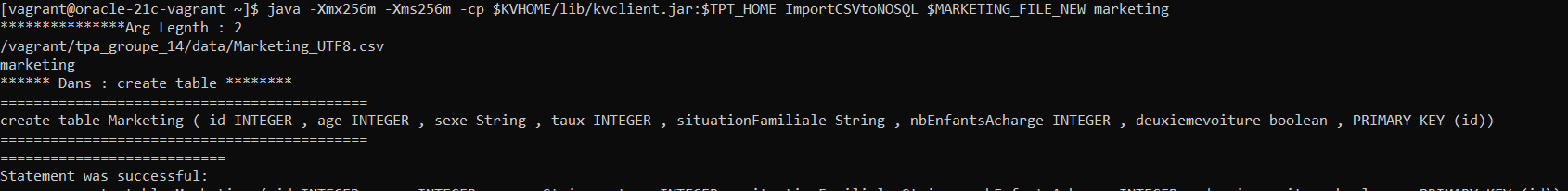


Figure 5 Compiler Java

#### Exécution pour le fichier client :

java -Xmx256m -Xms256m -cp $KVHOME/lib/kvclient.jar:$TPT\_HOME ImportCSVtoNOSQL $CLIENTS\_13\_FILE\_NEW client.

* **Explication** :

Similaire à la commande précédente, mais importe les données de $CLIENTS\_13\_FILE\_NEW dans la collection client.

Ainsi ,les fichiers CSV ont été importés avec succès dans leurs bases respectives et sont maintenant opérationnels.

## Rapport sur le Script Python de Chargement des Données de MongoDB vers Hive

### Introduction

Ce rapport explique le script Python fourni, conçu pour transférer des données d'une base de données MongoDB vers Hive. Le script utilise `pymongo` pour interagir avec MongoDB et génère des requêtes SQL pour insérer les données dans Hive. Ce processus est crucial pour intégrer des données non structurées dans un environnement Big Data, où elles peuvent être analysées et exploitées plus efficacement.

### Objectifs du Script

- Extraction des Données : Récupérer des documents depuis une collection MongoDB.

- Transformation des Données : Générer des requêtes d'insertion Hive pour chaque document.

- Insertion des Données : Insérer les données dans des tables Hive en utilisant des fichiers `.hql` et `beeline`.

### Description du Fonctionnement

#### Étape 1 : Extraction des Données depuis MongoDB

Le script se connecte à une instance MongoDB locale, accède à une base de données nommée `sourceCSV` et récupère tous les documents d'une collection spécifiée. Cette étape est réalisée par la fonction `fetch\_data\_from\_mongodb`.

#### Étape 2 : Génération des Requêtes d'Insertion

La fonction `generate\_insert\_query` crée des requêtes SQL pour insérer chaque document dans Hive. Les clés et les valeurs des documents sont transformées pour être compatibles avec le format SQL. Par exemple, les chaînes de caractères sont échappées correctement pour éviter les erreurs d'insertion.

#### Étape 3 : Insertion des Données dans Hive

La fonction `insert\_into\_hive` gère l'insertion des données dans Hive. Elle génère des fichiers `.hql` contenant les requêtes d'insertion et utilise `beeline` pour exécuter ces requêtes. Les données sont traitées par lots (batches) pour optimiser les performances et gérer efficacement les grandes quantités de données.

#### Étape 4 : Gestion des Batches

Pour faciliter l'insertion des données par lots, la fonction `chunk\_list` divise les données en segments de taille définie. Cette méthode permet de traiter de grandes quantités de données en plusieurs petites opérations, réduisant ainsi la charge sur le système et améliorant l'efficacité globale.

### Fonction Principale

La fonction `main` coordonne l'ensemble du processus. Elle spécifie les collections MongoDB à traiter et les tables Hive correspondantes, puis elle appelle les fonctions nécessaires pour extraire, transformer et insérer les données. Actuellement, le script est configuré pour traiter une collection appelée `catalogues` et l'insérer dans une table Hive nommée `catalogue`.

### Conclusion

Ce script de chargement de données automatise le processus de migration des données de MongoDB vers Hive, un élément clé dans la gestion de Big Data. Il extrait des documents MongoDB, génère des requêtes SQL adaptées, et insère les données dans Hive en utilisant des fichiers `.hql` et l'outil `beeline`. Cette approche garantit que les données non structurées peuvent être intégrées efficacement dans un environnement Big Data pour des analyses plus approfondies. Le script est structuré pour être extensible et adaptable à différentes collections et tables, permettant ainsi une utilisation flexible et évolutive.

La construction du Data Lake s'arrête à cette étape. À ce point, toutes les données sont ingérées, transformées, et organisées dans Hive, prêtes pour l'analyse.

# Hadoop Map Reduce

#### Rapport d’Intégration de Données via Spark MapReduce

* 1. **Introduction**

Ce rapport détaille les étapes de la mise en œuvre de l’activité autour de Spark MapReduce dans le cadre du projet d’intégration des données de CO2. L’objectif est d’adapter le fichier `CO2.csv` et de l’intégrer dans une table interne Hive nommée `catalogue`. Cette initiative s'inscrit dans la démarche globale d’amélioration de la qualité des modèles prédictifs du concessionnaire en utilisant des données complémentaires trouvées sur Internet.

* 1. **Objectifs**

L'objectif principal est de développer un programme Spark MapReduce pour adapter le fichier `CO2.csv` afin d'intégrer les informations complémentaires dans la table `catalogue`.

Les objectifs spécifiques sont les suivants :

* Nettoyer les données erronées et combler les valeurs manquantes.
* Adapter le format de stockage pour qu'il corresponde à celui de la table `catalogue`.
* Calculer les valeurs moyennes d’émissions de CO2, de Bonus/Malus et de coût d’énergie lorsque les données sont incomplètes ou absentes.
* Intégrer les données transformées dans la table Hive.
  1. **Étapes de Mise en Œuvre**
* Accès aux Données

**Chargement du fichier CO2.csv dans HDFS:** Le fichier `CO2.csv` est chargé dans HDFS, permettant ainsi de créer un DataFrame nommé `co2\_df`.

**Accès aux données de la table Hive Catalogue:** Puisque `catalogue` est une table Hive, ses données sont stockées dans HDFS. On accède à ces fichiers pour créer un DataFrame correspondant nommé `catalogue\_df`.

* Prétraitement des Données

**Colonnes combinées:** La colonne `Marque/Modèle` dans `CO2.csv` contient à la fois la marque et le modèle des voitures. Pour faciliter le traitement, nous extrayons d’abord toutes les marques disponibles dans `catalogue\_df` pour créer un nouveau DataFrame nommé `marque\_catalogue\_df`.

**Jointure des données:** Une jointure interne est effectuée entre `co2\_df` et `marque\_catalogue\_df`. Cette jointure est basée sur la condition que la colonne `Marque/Modèle` contient la marque présente dans `marque\_catalogue\_df`. Cela permet de n'intégrer que les valeurs de CO2 des marques qui sont déjà présentes dans le catalogue, en excluant les marques non nécessaires.

* Nettoyage des Données

**Suppression des colonnes inutiles:** Après la jointure, la colonne `Marque/Modèle` n'est plus nécessaire car nous avons déjà la marque des voitures grâce aux actions de jointure précédentes. Nous retirons donc cette colonne pour obtenir le DataFrame `co2\_marque\_df`, qui contient les informations de "Bonus/Malus", "Rejets CO2 g/km", et "Coût Énergie" des marques de voitures.

**Traitement des valeurs:**

* Les valeurs vides sont remplacées par 0.
* Les unités "€" sont enlevées pour que la valeur des colonnes soit numérique et non textuelle.
* Les erreurs, comme “-6 000€ 1”, sont corrigées pour devenir “-6000”.
* Calcul des Moyennes

**Moyenne par marque:** On calcule les moyennes des colonnes “Bonus/Malus”, “Rejets CO2 g/km”, et “Coût Énergie” par marque, et on obtient un “DataFrame” nommé `co2\_marque\_avg\_df``.

**Moyenne globale:** Pour les marques de voitures absentes dans le fichier `CO2.csv`, on calcule la moyenne globale des valeurs des colonnes, obtenant ainsi `co2\_all\_avg\_df``.

* Intégration des Données Moyennes

Jointure et Union des DataFrames:

* On identifie les marques présentes dans la table catalogue mais absentes dans le fichier CO2.csv, et on obtient `marque\_only\_catalogue\_df`.
* On joint la moyenne globale des valeurs des colonnes de CO2 avec `marque\_only\_catalogue\_df` pour obtenir les valeurs de CO2 des marques absentes, créant ainsi `co2\_marque\_catalogue\_avg\_df`.
* On effectue l’union des valeurs de CO2 des marques présentes et absentes dans `CO2.csv` afin d'obtenir les valeurs de CO2 pour toutes les marques présentes dans le catalogue, aboutissant à `co2\_all\_avg\_df`.

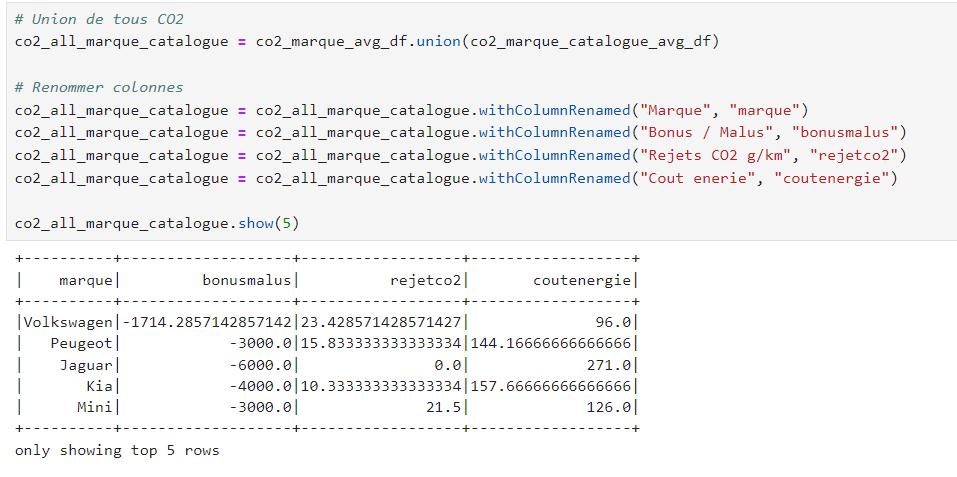
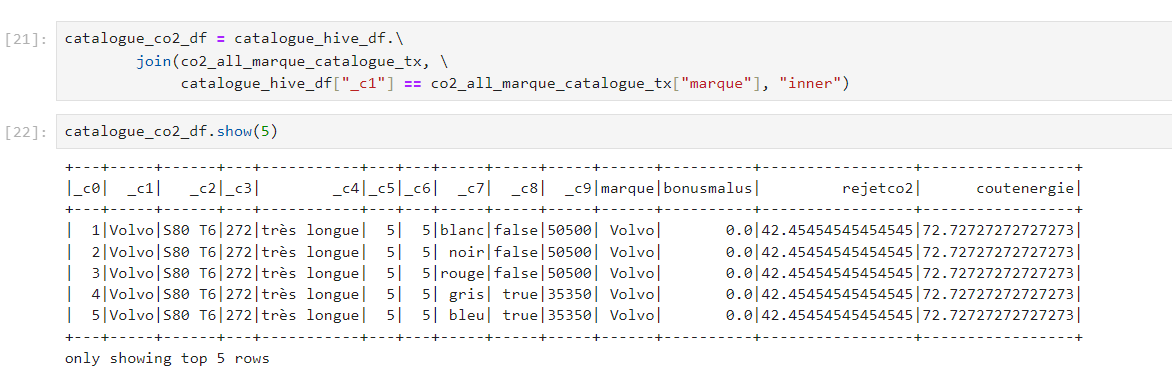


Figure 6 CO2 Dataframe

* Intégration Finale dans la Table Catalogue

**Jointure finale:** On joint `catalogue\_df` et `co2\_all\_avg\_df` par rapport à leurs marques pour obtenir `catalogue\_co2\_df`.

**Sauvegarde des données:** Le DataFrame `catalogue\_co2\_df` est sauvegardé dans un fichier CSV dans HDFS.



* Création de la Table Externe Hive

**Ajout du fichier CSV dans le datalake:**

* Le fichier CSV final est ajouté au datalake.
* Une table externe Hive, nommée `catalogue\_co2\_ext`, est créée comme miroir de ce fichier CSV.
  1. **Résultats et Conclusion**

L’exécution du programme Spark MapReduce a permis d’adapter le fichier `CO2.csv` et d'intégrer les informations complémentaires dans la table `catalogue`. Les données ont été nettoyées, les valeurs manquantes ont été comblées avec des moyennes calculées, et le format de stockage a été ajusté. L’intégration des données dans Hive a été réalisée avec succès, permettant ainsi d’enrichir la base de données du concessionnaire pour une meilleure qualité des modèles prédictifs.

# Analyse de données avec des outils de machine learning (R, …)

#### 

## Rapport sur le Script SQL de Nettoyage des Données dans Hive

### Introduction

Ce rapport vise à fournir une explication détaillée du script SQL joint, utilisé pour le nettoyage et l'agrégation des données dans notre base de données. Le script traite principalement des données relatives aux véhicules et aux clients, en appliquant divers processus de transformation et de standardisation afin d'améliorer leur fiabilité et leur cohérence. Les actions menées par ce script sont essentielles pour garantir des données de haute qualité, prêtes à être exploitées dans des analyses ultérieures.

### Objectifs Principaux

* Nettoyage des Données : Assurer la cohérence et la fiabilité des données issues de plusieurs tables.
* Transformation des Données : Appliquer des règles de transformation pour uniformiser les valeurs.
* Agrégation des Données : Agréger les informations clés pour une utilisation simplifiée et performante.
* Préparation des Données : Créer des vues consolidées pour faciliter l’analyse marketing et d’autres études.

### Description des Transformations et Vues Créées

**i) `catalogue\_co2\_view`**

Cette vue nettoie les données de la table `catalogue\_co2\_ext` en appliquant les transformations suivantes :

- Conversion des valeurs de la colonne `longueur` en codes numériques, facilitant ainsi les analyses quantitatives.

- Conversion des valeurs booléennes de la colonne `occasion` en entiers (0 pour faux, 1 pour vrai), standardisant ainsi le format des données.

**ii) `catalogue\_co2\_ext\_aggregated`**

La vue `catalogue\_co2\_ext\_aggregated` agrège les données nettoyées de `catalogue\_co2\_view` par marque. Elle calcule les moyennes des colonnes suivantes :

- `bonusmalus`

- `rejetco2`

- `coutenergie`

Cette agrégation permet de simplifier les comparaisons inter-marques en consolidant les indicateurs environnementaux.

**iii) `immatriculation\_view`**

Cette vue transforme les données de la table `immatriculation\_ext` de manière similaire à `catalogue\_co2\_view` :

- Conversion des valeurs de `longueur` en codes numériques.

- Conversion des valeurs booléennes de `occasion` en entiers (0 pour faux, 1 pour vrai).

**iv) `immatriculation\_co2\_view`**

La vue `immatriculation\_co2\_view` joint les données nettoyées de `immatriculation\_view` avec les données agrégées de `catalogue\_co2\_ext\_aggregated`. Cette jointure permet de combiner les informations d’immatriculation avec les données environnementales des véhicules, offrant ainsi une vue enrichie des immatriculations.

**v) `client\_view`**

Cette vue nettoie les données clients de la table `client\_ext` en appliquant les transformations suivantes :

- Remplacement des âges inférieurs à 18 ans par 0 pour éviter les données incohérentes.

- Standardisation des valeurs de la colonne `sexe` en 'M' ou 'F'.

- Uniformisation des valeurs de la colonne `situationfamiliale` pour les rendre cohérentes et analytiquement pertinentes.

- Conversion des valeurs booléennes de la colonne `deuxiemevoiture` en entiers (0 pour faux, 1 pour vrai).

**vi) `marketing\_view`**

La vue `marketing\_view` traite les données de la table `marketing\_ext` avec des transformations similaires à celles appliquées dans `client\_view`. Cette standardisation est essentielle pour les analyses marketing, garantissant des données cohérentes et fiables.

### Conclusion

Ce script SQL de nettoyage et de transformation des données pour Hive est un outil essentiel pour garantir la qualité des données dans un environnement Big Data. Les transformations appliquées et les vues créées permettent d'obtenir des données fiables, cohérentes et prêtes à être utilisées pour des analyses approfondies. Ce processus de nettoyage et d'agrégation est crucial pour toute entreprise souhaitant exploiter pleinement le potentiel de ses données. Le script répond aux exigences de qualité et de fiabilité nécessaires pour des analyses robustes et pertinentes.

## Modèle de prédiction

### Introduction

L'objectif de cette analyse est de prédire la catégorie de rejet de CO2 des véhicules en utilisant diverses méthodes de classification. Pour ce faire, nous utilisons les bibliothèques PySpark pour le traitement des données et l'entraînement des modèles de machine learning. Le dataset utilisé est chargé depuis un système de fichiers Hadoop (HDFS).

### Préparation des Données

#### Initialisation de la session Spark

Pour commencer, nous initialisons une session Spark, qui est essentielle pour le traitement des grandes quantités de données de manière distribuée.

#### Chargement du Dataset

Nous chargeons le fichier CSV contenant les données des véhicules depuis HDFS. Les données sont ensuite affichées pour une vérification initiale de la structure et des valeurs.

#### Formatage des Données

Les données nécessitent un certain nombre de transformations pour être prêtes à l'utilisation dans les modèles de machine learning :

- Renommer les colonnes pour remplacer les points par des underscores (par exemple, `clients\_ext.situationfamiliale` devient `clients\_ext\_situationfamiliale`).

- Transformation de certaines colonnes catégorielles en valeurs numériques, par exemple, la situation familiale (`clients\_ext\_situationfamiliale`) et le sexe (`clients\_ext\_sexe`).

- Remplacement des valeurs négatives par des zéros dans certaines colonnes numériques.

#### Sous-échantillonnage des Données

Afin de réduire la taille du dataset pour des raisons de performance, nous avons sous-échantillonné les données pour n'utiliser que 30% du dataset original.

#### Préparation des Caractéristiques

Nous avons utilisé `StringIndexer` pour convertir les colonnes catégorielles en indices numériques et `VectorAssembler` pour assembler les colonnes de caractéristiques en un seul vecteur de caractéristiques.

#### Séparation des Données en Enseignement et Test

Les données sont divisées en deux ensembles : 80% pour l'entraînement et 20% pour le test.

#### Modèles de Machine Learning

Nous avons testé plusieurs algorithmes de classification pour évaluer leur performance sur ce dataset :

##### 1. Random Forest

Le Random Forest est un ensemble d'arbres de décision. Nous avons réglé les hyper paramètres tels que le nombre d'arbres (`numTrees`) et la profondeur maximale (`maxDepth`).

##### 2. Decision Tree

Le Decision Tree est un algorithme simple et intuitif. Nous avons ajusté les hyperparamètres comme la profondeur maximale (`maxDepth`) et la fonction de critère d'impureté (`impurity`).

##### 3. Gradient Boosted Trees

Les Gradient Boosted Trees sont un ensemble d'arbres de décision entraînés séquentiellement. Nous avons ajusté des hyperparamètres tels que le nombre d'itérations (`maxIter`) et la profondeur maximale (`maxDepth`).

##### 4. Support Vector Machine

Les Support Vector Machines sont utilisés pour les tâches de classification avec un maximum de marge. Nous avons réglé des paramètres comme le nombre d'itérations (`maxIter`) et le paramètre de régularisation (`regParam`).

##### 5. Logistic Regression

La régression logistique est un modèle de base pour la classification. Nous avons ajusté des paramètres tels que le nombre d'itérations (`maxIter`), le paramètre de régularisation (`regParam`) et le paramètre d'élasticité (`elasticNetParam`).

##### 6. Multilayer Perceptron Classifier

Le Multilayer Perceptron est un réseau de neurones simples. Nous avons ajusté des paramètres comme le nombre d'itérations (`maxIter`) et la configuration des couches (`layers`).

##### 7. Naive Bayes

Le modèle de Naive Bayes est basé sur l'application du théorème de Bayes avec une forte indépendance entre les caractéristiques. Nous avons ajusté le paramètre de lissage (`smoothing`).

#### Évaluation des Modèles

Chaque modèle a été évalué en utilisant une validation croisée avec trois plis (`CrossValidator`) et la précision (`accuracy`) comme métrique d'évaluation.

**Matrice de confusion**

Une matrice de confusion est un tableau qui montre les performances d’un modèle de classification en comparant les prédictions avec les vraies valeurs des données.

Dans notre cas , elle est tout de suite combinée à la création des modèles .

Et c’est par l’utilisation de la classe *MulticlassMetrics* qu’on a pu calculer la matrice de confusion à partir des prédictions et on l’a stockée sous forme d’un tableau.

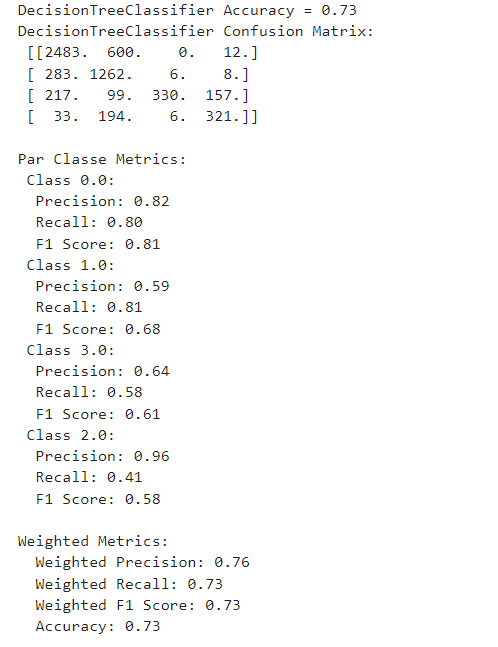
Grâce à la matrice de confusion , on a pu voir le taux de réussite de chaque colonnes ainsi que le taux de réussite du modèle par :

* ‘*Précision*’ : une mesure de l'exactitude des prédictions positives faites par le modèle. Elle répond à la question :

"*Parmi toutes les prédictions positives faites par le modèle, combien sont réellement correctes* ?"

* ‘*Recall*’ : aussi appelé exhaustivité, mesure la capacité du modèle à retrouver tous les échantillons positifs réels. Il répond à la question :

"*Parmi tous les échantillons positifs réels, combien le modèle en a-t-il retrouvé* ?"

* ‘*F1\_score*’ : une mesure combinée de la précision et du rappel. Il est utile lorsque les classes sont déséquilibrées (c'est-à-dire lorsque vous avez beaucoup plus d'échantillons d'une classe que d'une autre)
* *Accuracy* : L'exactitude mesure la proportion d'échantillons correctement prédits parmi tous les échantillons   
  

#### Résultats des Modèles

- Random Forest Classifier : Précision de 72%

- Decision Tree Classifier : Précision de 73%

- Gradient Boosted Trees Classifier : Précision de 73%

- Support Vector Machine (Linear SVM) : Précision de 62%

- Logistic Regression : Précision de 64%

- Multilayer Perceptron Classifier : Précision de 56%

- Naive Bayes : Précision de 47%

#### Résultat des comparaisons des modèles de prédiction

Parmi les modèles testés, le Decision Tree Classifier a obtenu la meilleure précision avec un score combiné de 2.18.

**PREDICTION**

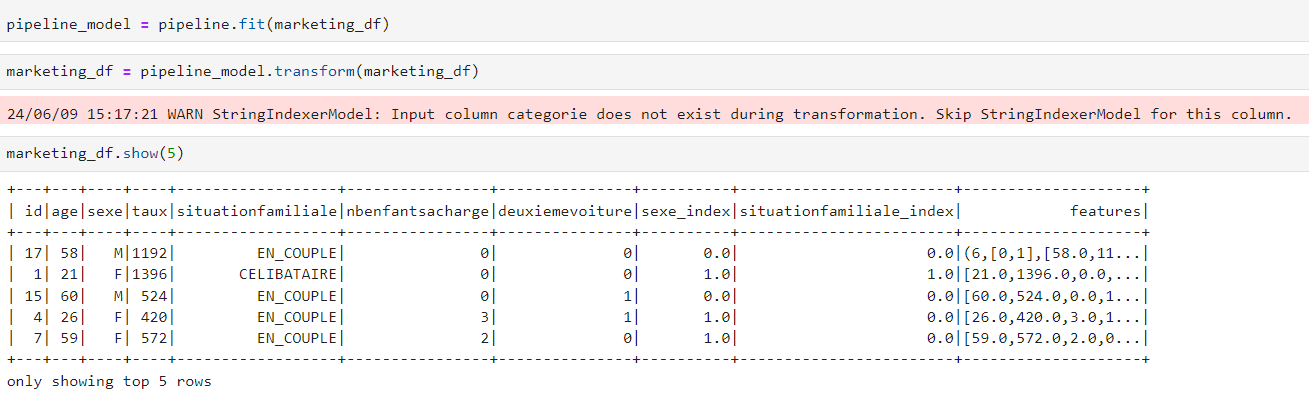
Après avoir fait la comparaison des modèles et avoir obtenu le meilleur ,

Nous allons effectuer la prédiction de catégorie de véhicules correspondant pour chaque client dans le fichier marketing .

On va d’abord accéder à la vue `view\_marketing` pour procéder à la prédiction :



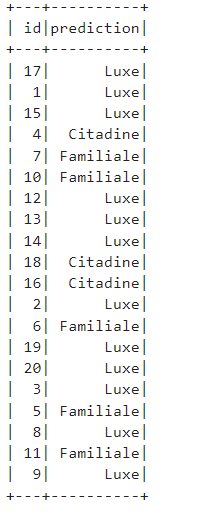
Puis continuer par la configuration et paramétrage des données avant la prédiction .



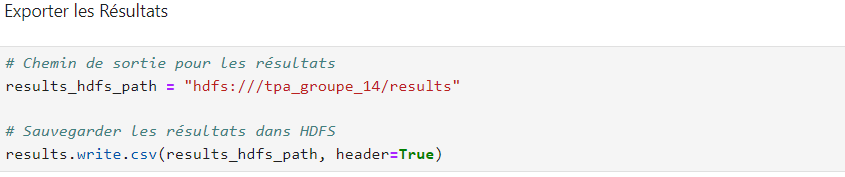
On remplace les valeurs numériques des prédictions par leurs équivalents textuels à l'aide d'un dictionnaire (`categorie\_dict`) pour rendre les résultats plus lisibles. Ensuite, on sélectionne uniquement les colonnes "id" et "prediction" pour l'affichage.



Voici le résultat:



Les résultats sont ensuite sauvegardés dans HDFS au format CSV, ce qui permet de stocker et de partager les résultats de manière distribuée et accessible pour d'autres processus ou analyses.

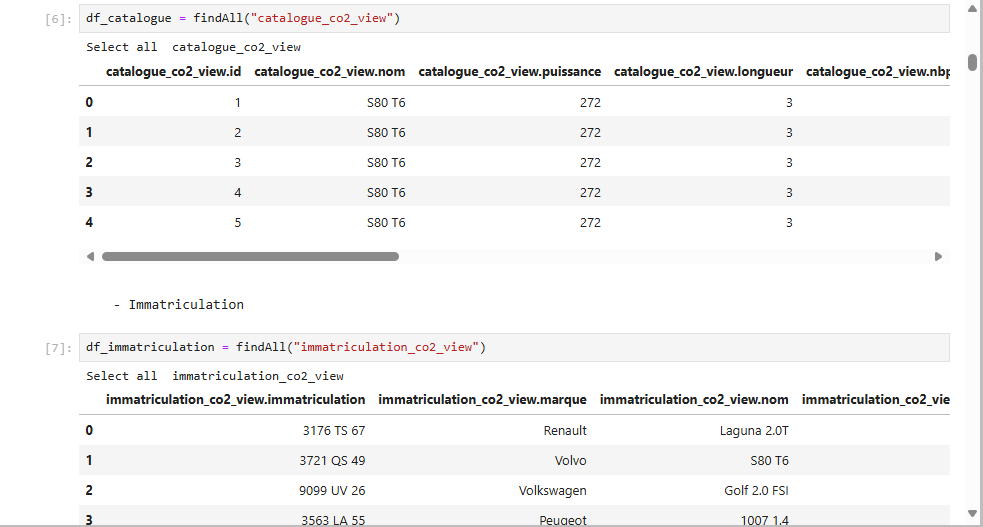


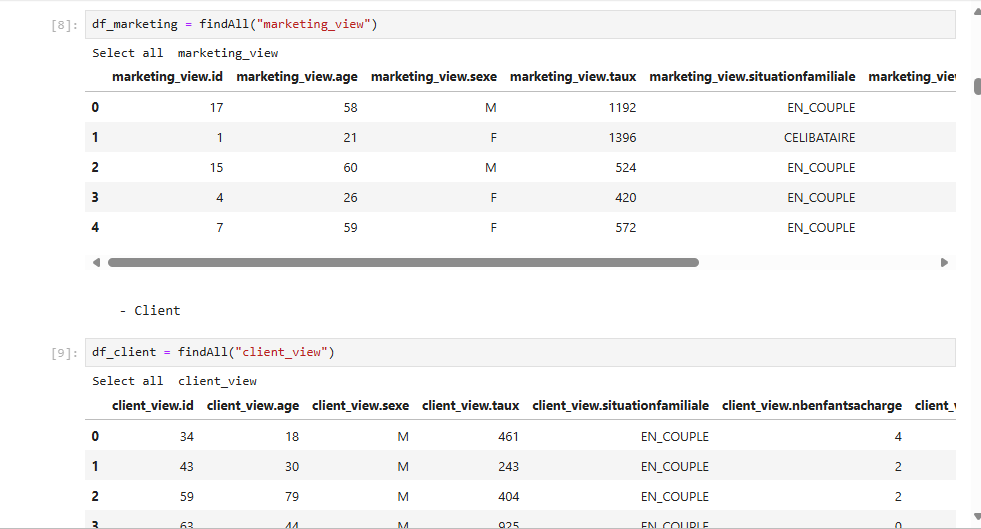
1. **Analyse de données avec des outils de machine learning:**

On a effectué multiples étapes afin d’arriver à la construction du modèle de prédiction pour l’analyse de données:

**8.1 Chargement des données:**

On se connecte à HIVEQL et on charge dans des dataframes les données nécessaires venant des view dans data lake incluant: immatriculation,catalogue\_co2,marketing et client



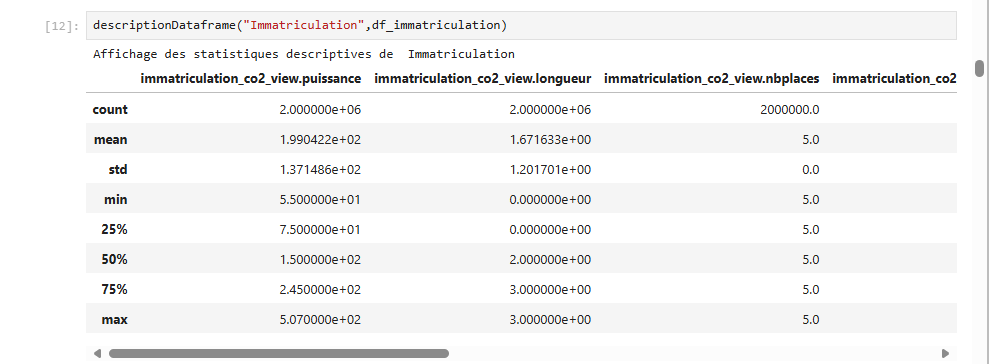


**8.2 Statistiques descriptives / vérification des valeurs:**

Cette étape permet de vérifier l'intégrité et la qualité des données présentes dans chaque dataframe.

Voici l’exemple pour les données immatriculation:

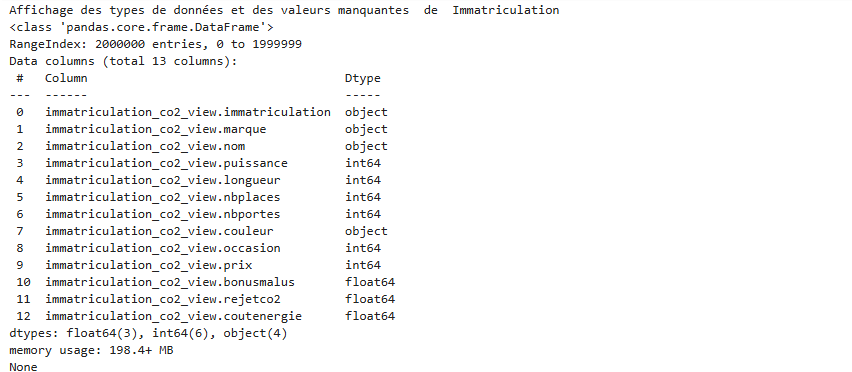
**8.2.1 Affichage des statistiques descriptives:**

****

On obtient un tableau montrant les statistiques comme la moyenne, l'écart-type, les valeurs minimales et maximales, ainsi que les quartiles pour chaque colonne numérique.

**8.2.2 Affichage des types de données et des valeurs manquantes**

Onaffiche les types de données des colonnes et le nombre de valeurs non nulles.

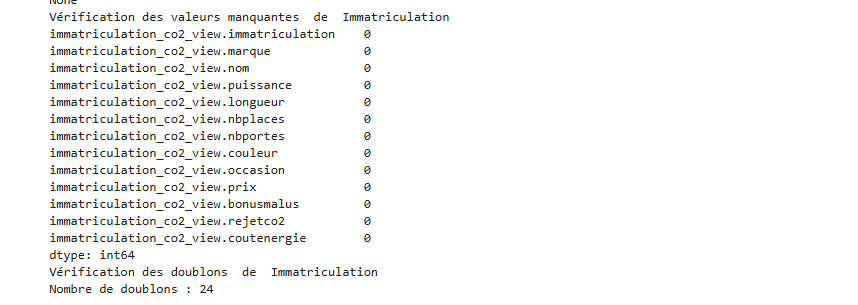


Résultat:

On peut voir que le Data Frame contient 2.000.000 d’entrées et 13 colonnes avec leur type et qu'il n'y a pas de valeurs manquantes..

**8.2.3 Vérification des doublons**

Puis on vérifie le nombre de doublons ou le nombre de lignes dupliquées dans le DataFrame.

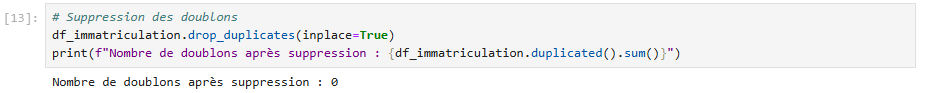


Résultat:

On voit ici que le data frame immatriculation contient 24 doublons

**8.2.4 Suppression des doublons**

On procède donc à la suppression des doublons.



Les mêmes analyses ont été effectuées sur les autres DataFrames, suivant un processus similaire,et on n’a détecté aucun doublon.

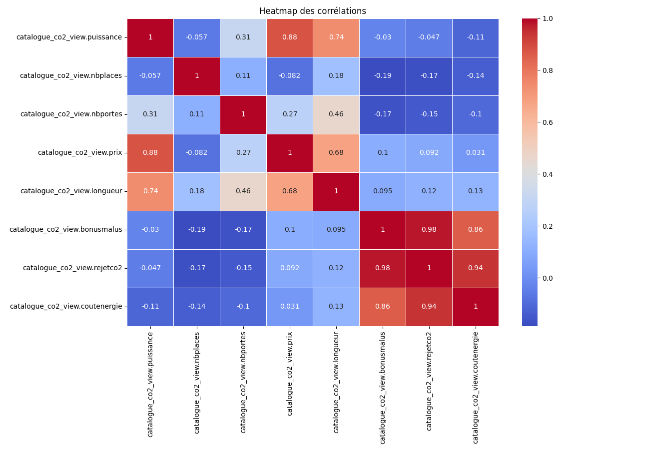
**8.2 Identification de catégories de véhicules:**

**8.2.1 Analyse exploratoire de données:**

On va d’abord procéder à l’analyse exploratoire des données de “catalogue CO2” vu que c’est à partir de ces données qu’on va faire le clustering pour avoir les catégories de véhicules.

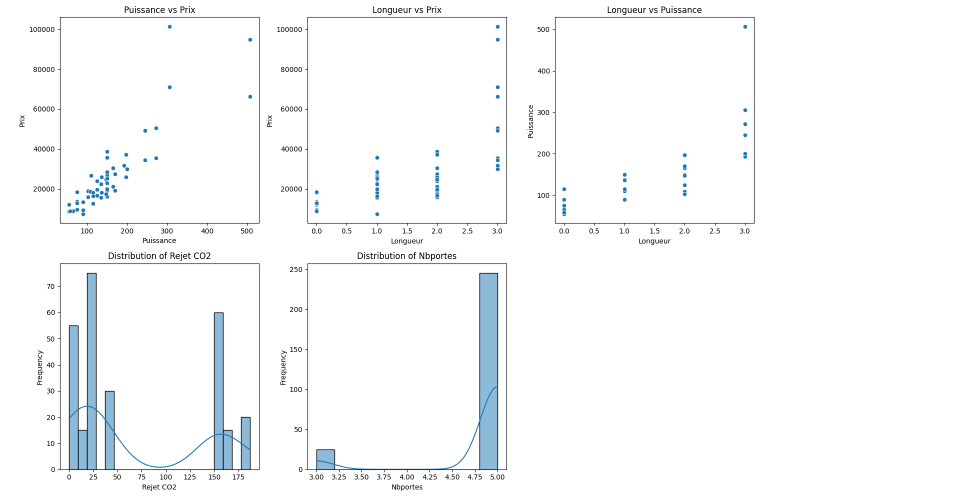
**8.2.1.2 Matrice de correlation:**

La matrice de corrélation a été utilisée pour réaliser une analyse exploratoire des données. Cette méthode permet d'identifier les coefficients de corrélations entre les différentes variables du catalogue, facilitant ainsi la sélection des variables les plus pertinentes pour le projet.



### 8.2.1.3 Histogramme et nuages de points:

On a aussi quelques analyses en nuages de points qui montrent les liaisons entre des variables et des histogrammes montrant la répartition des valeurs de variables sur le long des données.



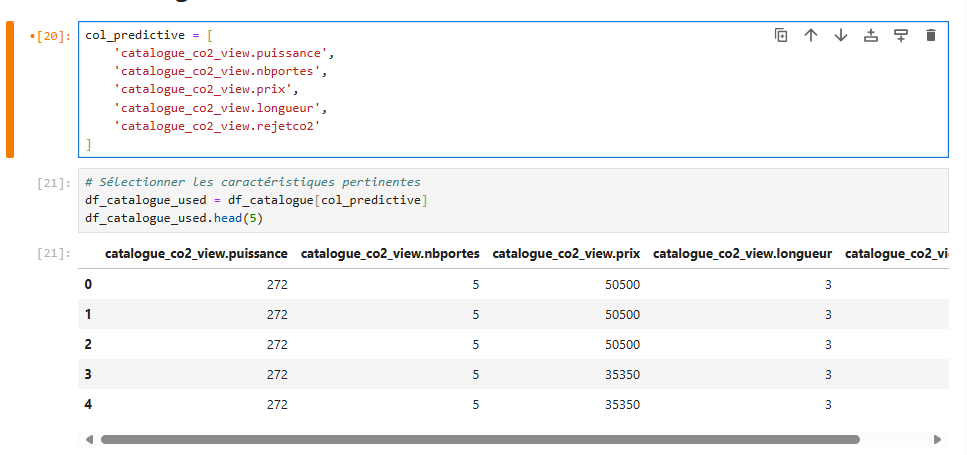
### 8.2.1.3 Conclusions de l'analyse exploratoire de données :

D’après ces analyses , on peut constater les faits suivants:

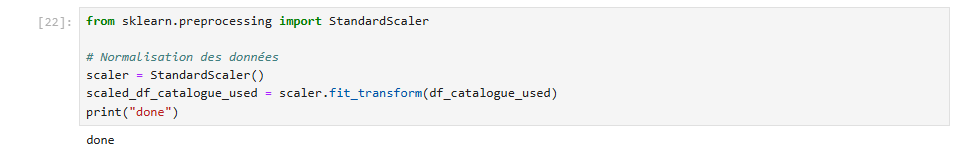
* **Variables importantes** :
  + **Puissance, prix et longueur** : Ces variables montrent de fortes corrélations avec plusieurs autres variables. Elles sont cruciales pour distinguer les différentes catégories de véhicules.
  + **Nombre de places**: Elle ne présente de forte corrélations avec aucune variable en particulier donc ne doit pas être exclue car elle présente un impact potentiel à faire varier le cluster
  + **Rejet de CO2, bonus et coût d'énergie** : Elles sont fortement corrélées entre elles mais ne montrent pas de forte corrélation avec les autres variables mis à part longueur et prix qui seront déjà considérées dans le cluster donc on va considérer une seules de ces variables pour éviter la redondance, ici on a choisi la variable rejetco2.
* **Variables moins pertinentes** :
  + **Nombre de portes** : Cette variable présente une distribution homogène et peut être moins informative pour le clustering.
* **Variables à considérer pour le clustering** :
  + **Puissance, nombre de places, prix, longueur,rejet co2** : Ces variables ont donc été identifiées comme les plus pertinentes pour effectuer le clustering des véhicules.

**8.2.2 Clustering pour avoir les catégories de véhicules:**

On récupère à nouveau les données de catalogues en sélectionnant les colonnes pertinentes obtenus lors de l’analyse exploratoire de données:

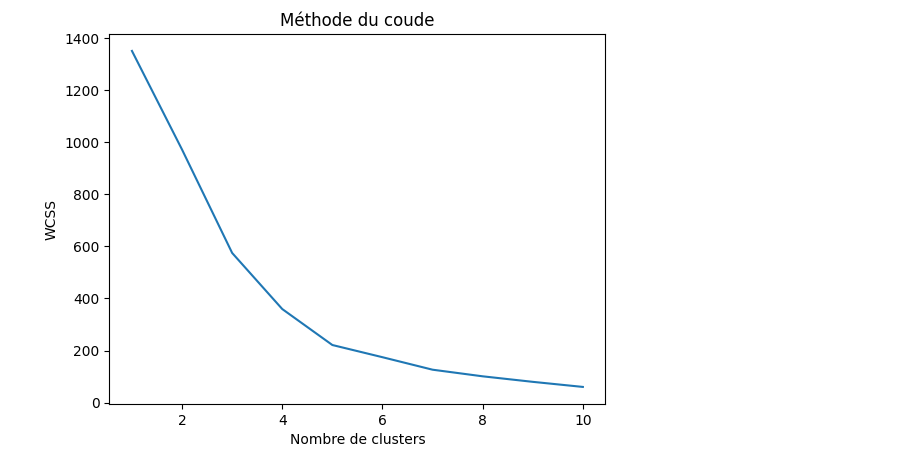


Une normalisation des données sera effectuée pour garantir une échelle uniforme entre les variables. Cette étape est cruciale pour assurer que chaque variable contribue équitablement au processus de clustering, améliorant ainsi la précision et la performance de l'algorithme.



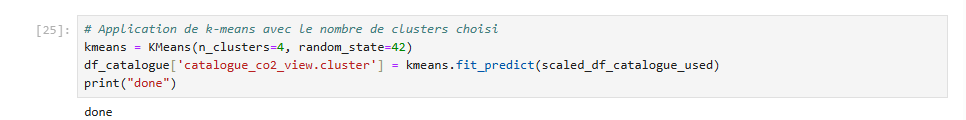
On procède ensuite au clustering des données pour la catégorisation des véhicules.

* Tout d’abord, on utilise la méthode de coude pour déterminer le nombre optimale de cluster

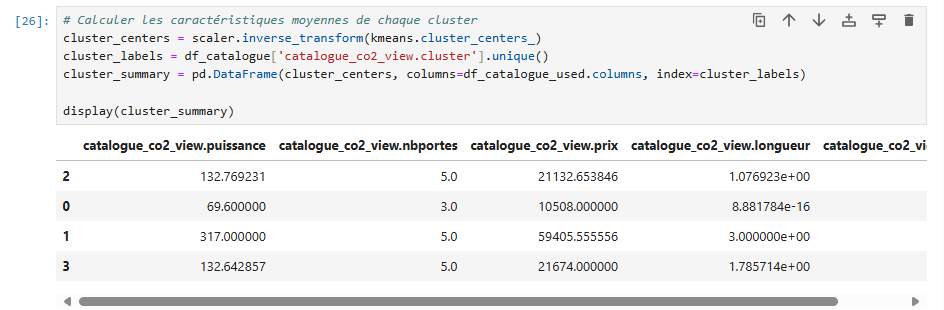


D'après la méthode du coude, le nombre de cluster optimale est aux alentours de 4

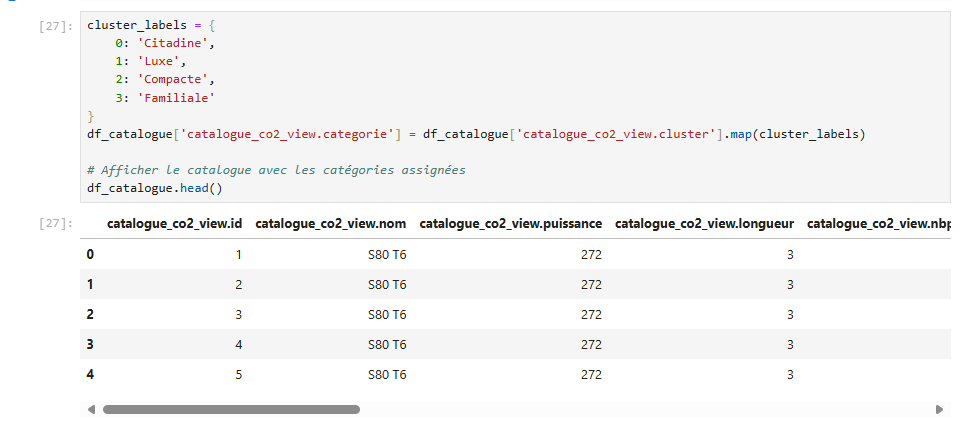
* On construit enfin le modèle de clustering en utilisant l'algorithme K-means.



* Ensuite on calcule et affiche ce résumé des clusters permettant de visualiser et d'interpréter les caractéristiques typiques de chaque cluster dans leurs unités d'origine, et en voyant les valeurs moyennes pour chaque propriété, on peut en tirer les catégories de véhicules correspondantes à ces propriétés.



* Puis on assigne les catégories correspondantes parmi les 4 obtenus à chaque véhicule dans le donnée de catalogue.



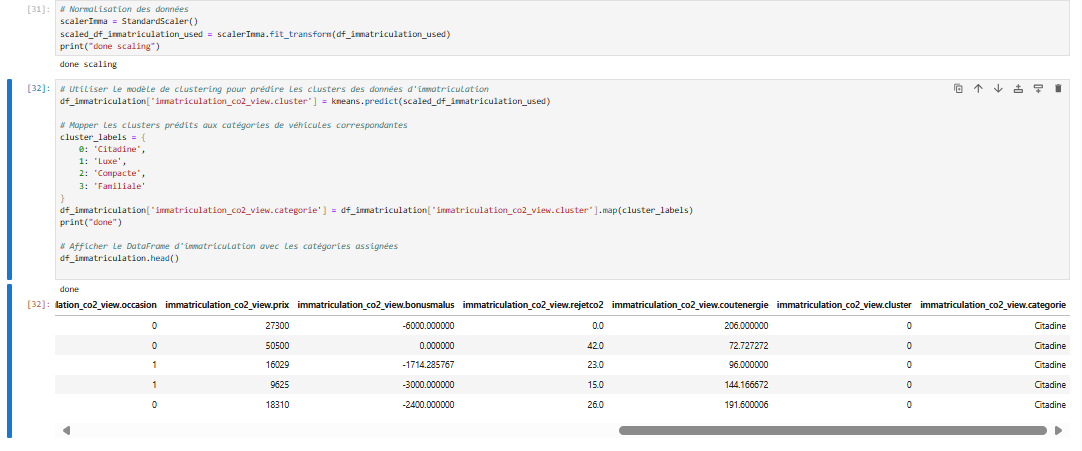
**8.3 Application des catégories de véhicules définies aux données des Immatriculations :**

Après avoir trouvé les différentes catégories de véhicules dans le clusters, on va assigner maintenant à chaque ligne des données immatriculations effectuées cette année les catégories de véhicules adéquates dans le catalogue en faisant une prédiction avec le modèle de cluster qu’on a obtenu.

* On va d’abord charger les données d’immatriculations pour ne sélectionner que les colonnes pertinentes comme avec les données de catalogue:

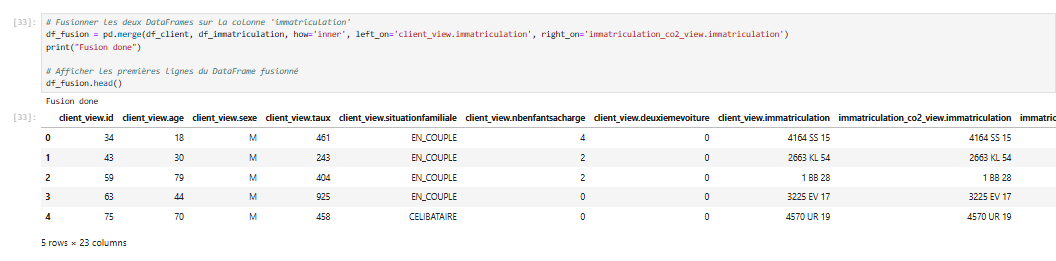


* Puis on va normaliser les données et faire la prédiction de catégorie de véhicules avec le modèle de cluster précédent:

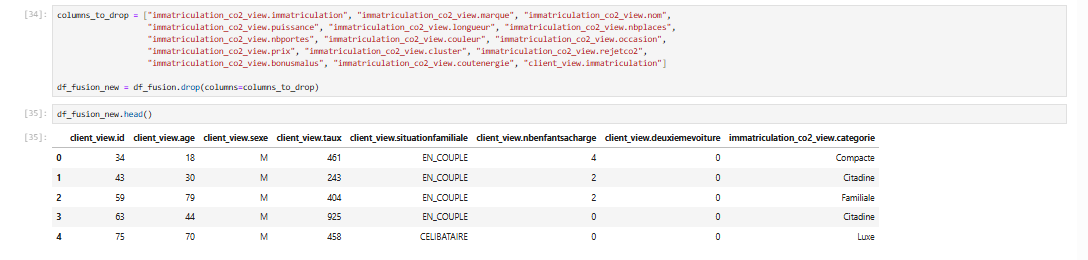


**8.4 Fusion des données Clients et Immatriculations :**

* On fusionne ensuite les données clients et immatriculations via jointures pour avoir les catégories de véhicules achetées par les clients:



* Puis on supprime les colonnes non nécessaires et on ne garde que les données des clients et les catégories de voitures qu’ils ont choisis pour procéder à l’analyse de donnée suivantes:



* On mets ces données obtenus dans hdfs pour la construction de modèle de prédiction utilisée dans la partie suivante:



# Conclusion générale

La conclusion est l’occasion pour vous de faire un bilan sur votre projet. Ce bilan doit

comporter les parties suivantes :

* Le bilan des résultats obtenus pour l’entreprise
* Le bilan des problèmes rencontrés et des solutions apportés
* Les perspectives du projet
* Le bilan personnel

# Références et Bibliographie

1. Auteur1, auteur 2, …, Titre\_du\_livre, Edition, date\_parution, lien s’il ya

…

1. Auteur1, auteur 2, …, Titre\_article, Revue de parution ou site Web, date\_parution, lien

s’il ya

…

Les liens web seuls ne peuvent être une bibliographie. Vous devez mettre dans la bibliographie : des articles de revues, des livres, des documents en respectant les exemples ci-dessus. Si vous avez un doute, merci d’aller voir les références et bibliographies de n’importe quel article WIKIPEDIA.

# Annexes

Les chemins indiqués dans cette partie sont tous expliqués en détails dans le projet. à la racine du projet, nous avons posé un fichier “markdown” pour les débuts d’explications.

## Vidéo de présentation de votre projet

Il s’agit là d’une vidéo de démonstration de votre projet étape par étape depuis la collecte des données jusqu’à l’analyse

Chaque membre du groupe doit présenter une partie.

## Dossier contenant les scripts et programmes de construction du lac de données

Les scripts et programmes qui nous ont permis de construire le lac de données se trouvent dans le dossier **“1\_data\_lake”.**

## Dossier contenant les scripts et programmes Hadoop Map Reduce

Les scripts et programmes qui nous ont permis de réaliser la partie Hadoop MapReduce sont dans le dossier **“2\_map\_reduce”.**

## Dossier contenant les scripts et programmes d’analyse de données

Les scripts et programmes qui nous ont permis de réaliser la partie analyse de données avec Python sont dans **“3\_data\_analysis”**.