Analyse de la clientèle d'un concessionnaire automobile dans l'objectif de pouvoir recommander des modèles de véhicules

MEMBRE GROUPE 6

DIEKE JONATHAN

DOUDOU BI TOUVOLY

SCHAEFFER PHILLIP

INTRODUCTION

I- ANALYSE EXPLORATOIRE DES DONNEES

1- Détection et gestion des anomalies

2- Compréhension et choix des variables

3- Transformation des variables choisies

II- CLUSTERING DES DONNEES

1- Etude des méthodes de clustering

2- Réduction de dimension et visualisation

3- Caractéristiques des clusters obtenus

4- Ajout de nouvelles variables

III- CLASSIFICATION ET PREDICTION

1- Les algorithmes de classifications

2- Api et déploiement

CONCLUSION

Contexte et mission

Contactés par un concessionnaire automobile, nous avons pour mission de l'aider à mieux cibler les véhicules susceptibles d'intéresser ses clients. Pour cela il met à notre disposition :

- son catalogue de véhicules (Catalogue.csv) ;
- son fichier clients contenant les achats de l'année en cours (Clients.csv - divisé en 2) ;
- un accès aux informations sur les immatriculations effectuées cette année (Immatriculations.csv) ;
- une brève documentation des données.

A la fin de ce projet nous devons proposer un outil permettant :

- d'évaluer en temps réel le type de véhicule le plus susceptible d'intéresser les clients qui se présenteront dans la concession ;
- d'envoyer une documentation précise sur le véhicule le plus adéquat pour des clients sélectionnés par son service marketing.

1- Détection et gestion des anomalies

Nous avons zéro valeur manquante dans ligne du dataset Catalogue et pas d'anomalie.

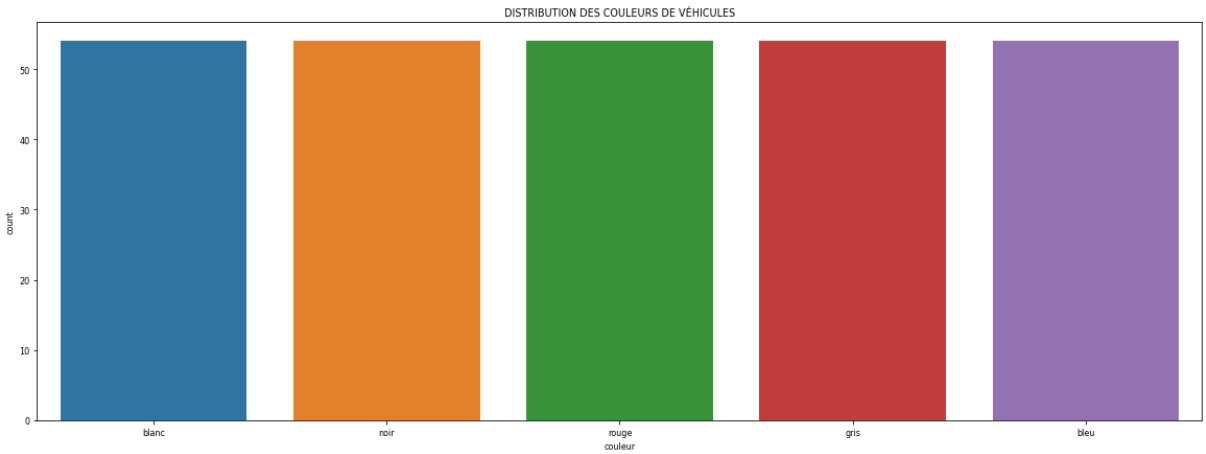
Dans le dataset immatriculation, nous n'avons pas de données manquantes mais plutôt des données dupliquées. Comme ces données ne sont pas nombreuses, nous décidons de les supprimer car une ligne dupliquée n'a pas de sens et ne donne pas assez d'information.

Les anomalies dans Client sont décrites dans le tableau suivant :

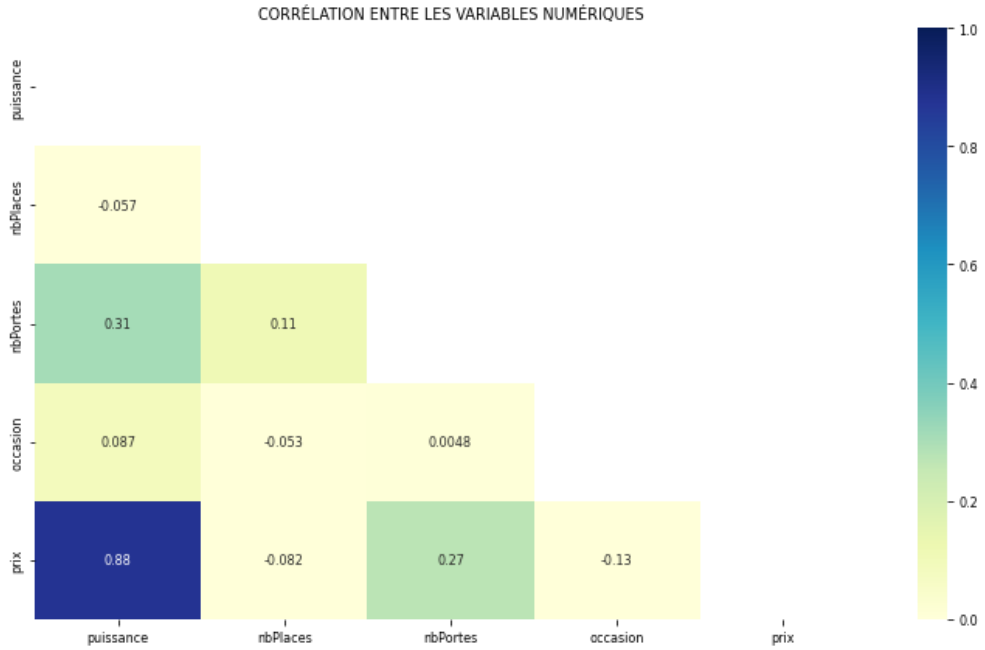
age	Sexe	Taux	situationFamiliale	nbEnfantsAcharge	2eme voiture
308	310	322	306	316	205

Nous allons imputer certaines variables et supprimer d'autres

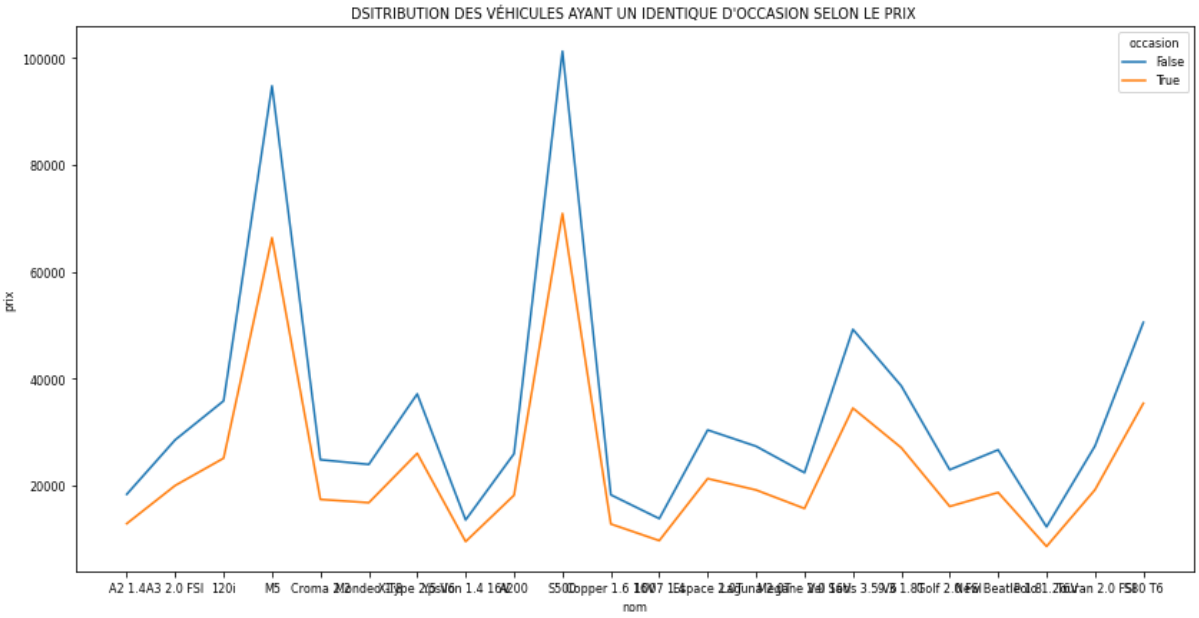
2- Compréhension et choix des variables



Nous une distribution parfaite du nombre de couleur dans le dataset, ce qui veut dire que tous les véhicules dans catalogue sont représentés dans 5 couleurs différentes. Ainsi la couleur n'est pas un facteur de distinction des véhicules car, pour un véhicule donné, nous aurons forcément 5 cinq couleurs. Nous décidons de se faire, de supprimer la couleur comme variable pour notre analyse.



Nous avons une forte corrélation entre les variables **puissance** et **prix**. Cette forte corrélation peut causer un problème pour notre modèle. De ce fait, nous décidons de supprimer l'une des variables. Ici, nous choisissons de supprimer la variable prix.



Nous remarquons que la plupart des véhicules sont en doublons avec une neuve et l'autre d'occasion. En effet, 40% des véhicules sont d'occasion, le prix des véhicules d'occasion évolue de la même façon que ceux que les voitures neuves mais avec un prix moindre.

3- Transformation des variables choisies

Pour la suite de nos travaux, nous choisissons les variables suivantes :

la marque la puissance la longueur le nombre de places le nombre de portes.

Dans le domaine du machine Learning, le prétraitement ou encore preprocessing en AI désigne les différentes étapes effectuées sur les données d'entrée avant qu'elles ne soient introduites dans un modèle. Il peut s'agir de nettoyer les données, de les mettre à l'échelle et de les transformer de diverses manières pour les rendre plus adaptées au modèle. Le prétraitement peut contribuer à améliorer les performances d'un modèle d'apprentissage automatique en rendant les données plus prévisibles et plus faciles à exploiter par le modèle. Voici quelques techniques de prétraitement courantes :

- ***La normalisation*** : Il s'agit du processus de mise à l'échelle des données afin qu'elles aient une moyenne de 0 et un écart type de 1. Cela peut aider le modèle à converger plus rapidement et peut également améliorer ses performances.
- ***Standardisation*** : Il s'agit d'un processus similaire à la normalisation, mais au lieu de mettre à l'échelle les données pour qu'elles aient un écart-type de 1, on les met à

l'échelle pour qu'elles aient un écart-type de 0. Cela peut aider le modèle à être plus robuste face aux valeurs aberrantes des données.

- ***One-hot Encoder*** : Il s'agit d'une technique utilisée pour encoder des variables catégorielles en tant que données numériques. Elle crée un vecteur binaire pour chaque catégorie, avec un 1 dans la position correspondant à la catégorie et des 0 dans toutes les autres positions. Cela peut aider le modèle à mieux comprendre les données.

Nous effectuons ainsi, l'encodage des variables **nbPlace, nbPortes, occasion et longueur** et nous faisons également la standardisation de la variable **puissance**.

I- CLUSTERING DES DONNEES

1- Visualisation des données et interprétation

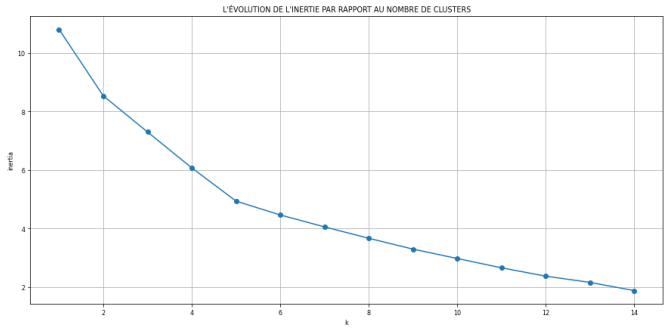
Le clustering est une technique d'apprentissage non supervisé en machine Learning. Il permet de catégoriser des éléments en fonction de leur similarité ou ressemblance. Il existe plusieurs algorithmes de clustering. Le but est de trouver les caractéristiques similaires entre données et les regrouper. Parmi ces algorithmes, nous avons le K-means, l'agglomération.

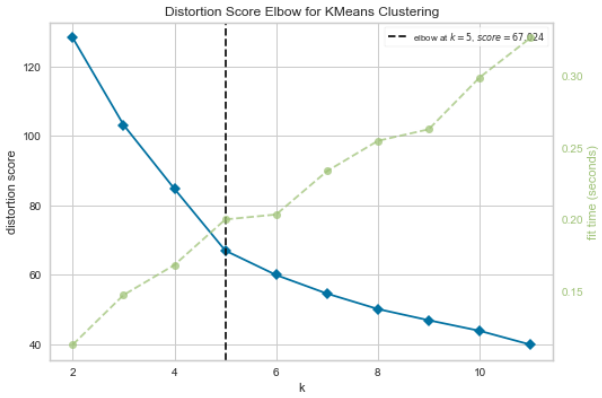
1-1- K-means

K-means est un algorithme basé sur les centroïdes, l'on choisit des centroïdes au hasard et on calcul la distance des points aux centroïdes. Chaque point du dataset est assigné au centroïde auquel il est le plus proche. Cette technique veut que l'on détermine le nombre de centroïdes à l'avance. Ce nombre représente le nombre de clusters. Cependant, comment choisie le nombre optimal de cluster ?

Différentes approches sont possibles notamment la courbe **elbow** et la courbe de **silhouette**.

- **Elbow** permet de déterminer le nombre optimal de cluster sur un intervalle donné. On précise un intervalle de valeur de cluster souhaité et on calcule la distance des points vis-à-vis de la variance. On trace ensuite une courbe des valeurs de cluster possible et de la variance pour chaque valeur. Le nombre de clusters est la partie de la courbe qui forme un coude, c'est-à-dire la partie où la valeur de l'inertie est la plus petite possible. Notons que la valeur optimale dépend de nous. En générales elle ne doit pas être trop grande ni trop petite.

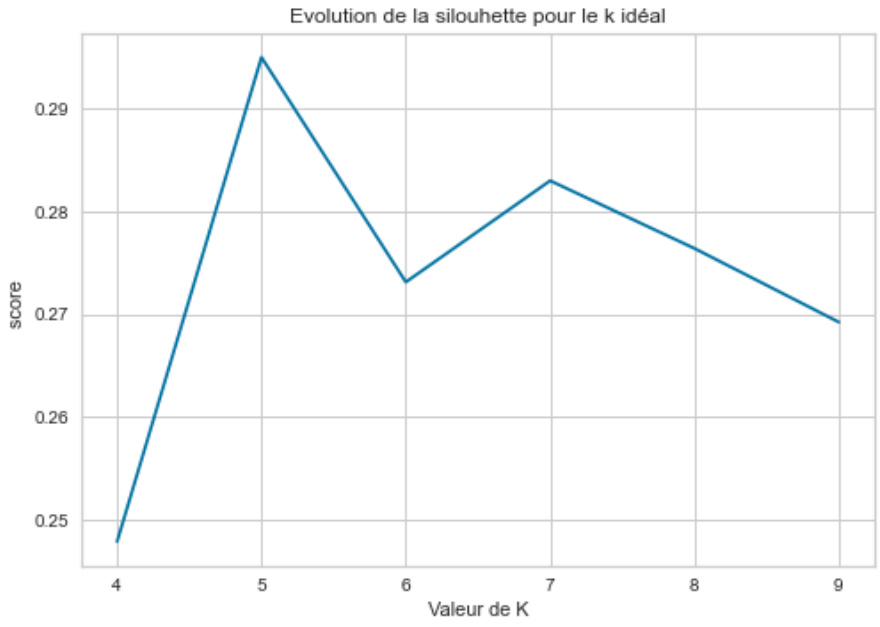




Nous remarquons que la partie de la courbe qui forme un coude se trouve en **k=4 et k=6.**

Essayons donc une autre méthode pour nous rapprocher de la valeur de **k** qu'il faut.

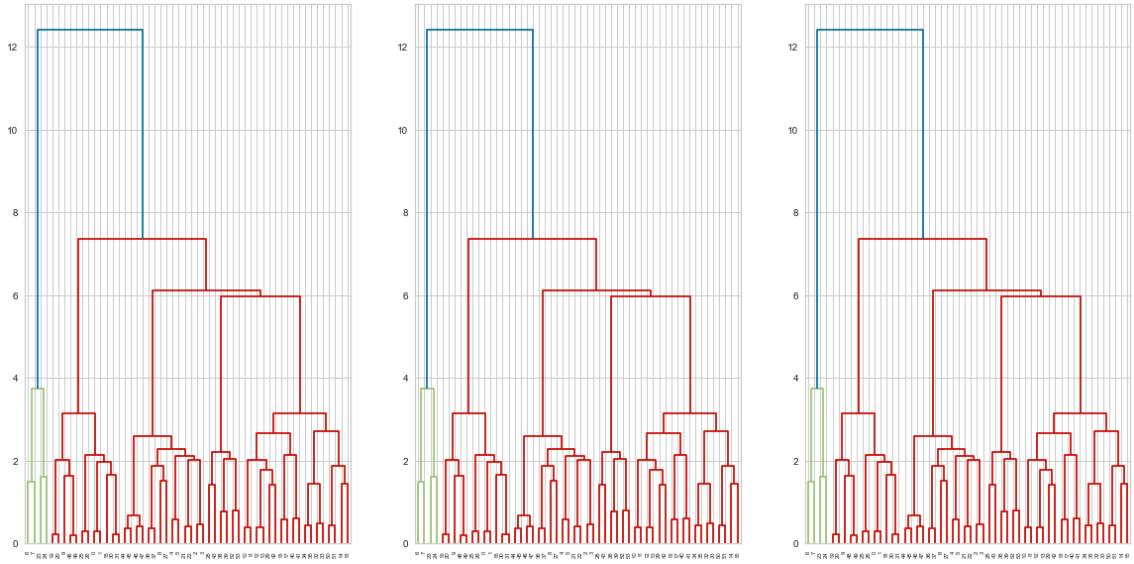
- Silhouette



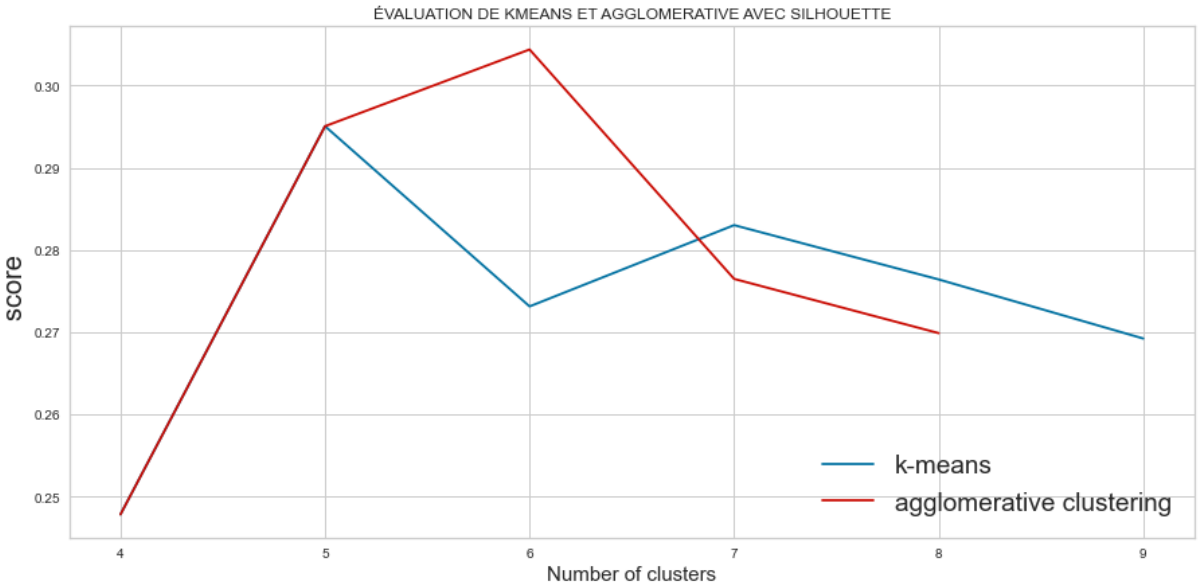
Plus le score est **grand,** mieux c'est. Avec la méthode silhouette, notre nombre de cluster est k = 5

1-2 Agglomérative cluster

C’est une technique de clustérisation, on départ chaque point est un cluster en lui-même. Ensuite, on calcul la similarité entre tous les points et les points avec la plus forte similarité sont regroupé entre eux pour former un cluster. Répète le processus plusieurs fois jusqu’à ce que tous les points soient dans un seul cluster. On parle alors de classification hiérarchique ascendante.



1-3 Aglomérative cluster et Kmeans sur le même graphique



Avec la méthode Elbow et le score de la silhouette, Kmeans nous fournit toujours k=5. De plus, l'Agglomerative Clustering nous fait supposer 5 clusters à partir du dendrogramme mais indique 6 clusters avec le score de la silhouette.

2- Réduction de dimension et visualisation

La réduction de dimension est une technique qui permet de visualiser des données de plusieurs dimensions dans une plus réduite. Notons que cette dimension réduite est une représentation des données originales à un certain pourcentage. Il existe plusieurs techniques de réduction dont la plus célèbre est le **PCA (*Principal Component Analysis*).** Nous avons utilisé **PCA avec K-means et Agglomérative** cluster avec des nombres de cluster différents et voir quel nombre de cluster est plus réaliste en fonction de leur repartion sur les graphique. Nous effectuons des représentations en deux dimensions. Ces deux dimensions expliquent **63.21%** des données d'origines.

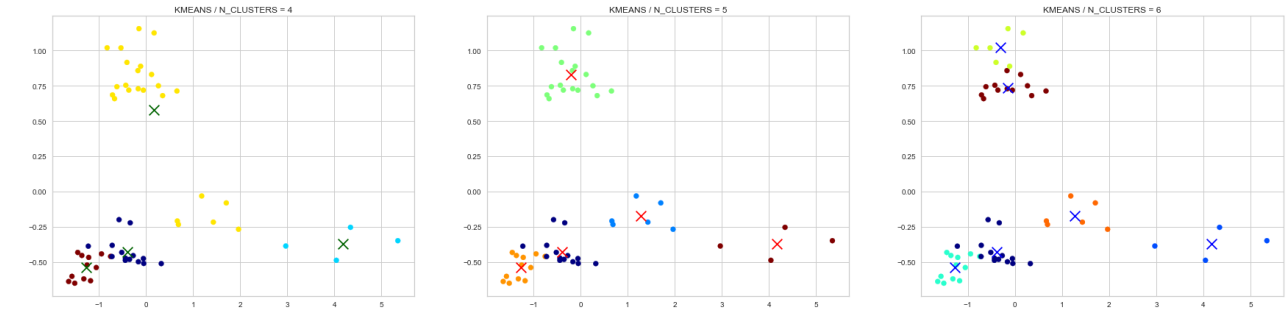


Figure 1 PCA avec K-means

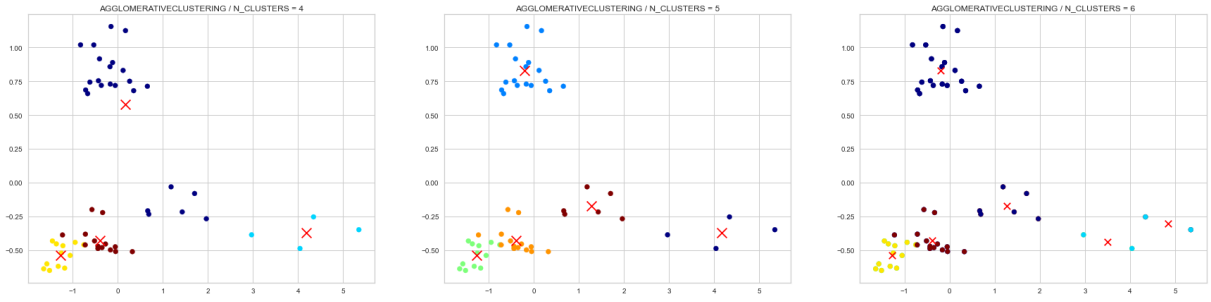


Figure 2 PCA avec agglomérative cluster

En visualisant en 2D après une réduction basée sur PCA, le K-Means et l'agglomérative clustering fournissent des résultats assez similaires. Après analyse, nous choisissons **k=5** clusters avec le modèle de k-Means car les données y sont mieux regroupées avec PCA.

3- Caractéristiques des clusters obtenus

En nous basant sur les techniques présente avec 5 comme nombre de clusters nous présentons les caractéristiques des différentes catégories de véhicules obtenus. Pour l'instant, nous avons labellisé les catégories **0,1,2,3 et 4.**

Pour la catégorie 0 :

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	label
2	Audi	A3 2.0 FSI	150	moyenne	5	5	0	28500	0
3	Audi	A3 2.0 FSI	150	moyenne	5	5	1	19950	0
4	BMW	120i	150	moyenne	5	5	0	35800	0
5	BMW	120i	150	moyenne	5	5	1	25060	0
8	Dacia	Logan 1.6 MPI	90	moyenne	5	5	0	7500	0
21	Mercedes	A200	136	moyenne	5	5	0	25900	0
22	Mercedes	A200	136	moyenne	5	5	1	18130	0
27	Nissan	Almera 1.8	115	moyenne	5	5	0	16450	0
36	Renault	Megane 2.0 16V	135	moyenne	5	5	0	22350	0
37	Renault	Megane 2.0 16V	135	moyenne	5	5	1	15644	0
44	Volkswagen	Golf 2.0 FSI	150	moyenne	5	5	0	22900	0
45	Volkswagen	Golf 2.0 FSI	150	moyenne	5	5	1	16029	0
46	Volkswagen	New Beatle 1.8	110	moyenne	5	5	0	26630	0
47	Volkswagen	New Beatle 1.8	110	moyenne	5	5	1	18641	0

Nous remarquons que tous les véhicules de la catégorie **0** ont une longueur **moyenne**, des nombres de portes et de places égale à **5.** La puissance est comprise entre **90 et 150.** De plus, nous avons des véhicules qui sont d'occasion et une représentation de plusieurs marques. Au vu de ces différentes caractéristiques, nous déduisons que ce sont des **véhicules normaux.**

Pour la catégorie 1 :

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	label
28	Nissan	Maxima 3.0 V6	200	très longue	5	5	0	30000	1
38	Renault	Vel Satis 3.5 V6	245	très longue	5	5	0	49200	1
39	Renault	Vel Satis 3.5 V6	245	très longue	5	5	1	34440	1
43	Skoda	Superb 2.8 V6	193	très longue	5	5	0	31790	1
52	Volvo	S80 T6	272	très longue	5	5	0	50500	1
53	Volvo	S80 T6	272	très longue	5	5	1	35350	1

Ce sont des voitures très longues dont la puissance est comprise entre **193 et 272.** Nous n'avons que deux voitures qui ne sont pas d'occasion. Nous pouvons dire que ce sont de **vieux véhicules.**

Pour la categorie 2 :

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	label
10	Fiat	Croma 2.2	147	longue	5	5	0	24780	2
11	Fiat	Croma 2.2	147	longue	5	5	1	17346	2
12	Ford	Mondeo 1.8	125	longue	5	5	0	23900	2
13	Ford	Mondeo 1.8	125	longue	5	5	1	16730	2
14	Honda	FR-V 1.7	125	longue	7	5	0	19550	2
15	Hyundaï	Matrix 1.6	103	longue	7	5	0	15960	2
16	Jaguar	X-Type 2.5 V6	197	longue	5	5	0	37100	2
17	Jaguar	X-Type 2.5 V6	197	longue	5	5	1	25970	2
29	Nissan	Primera 1.6	109	longue	5	5	0	18650	2
32	Renault	Espace 2.0T	165	longue	7	5	0	30350	2
33	Renault	Espace 2.0T	165	longue	7	5	1	21245	2
34	Renault	Laguna 2.0T	170	longue	5	5	0	27300	2
35	Renault	Laguna 2.0T	170	longue	5	5	1	19110	2
40	Saab	9.3 1.8T	150	longue	5	5	0	38600	2
41	Saab	9.3 1.8T	150	longue	5	5	1	27020	2
42	Seat	Toledo 1.6	102	longue	5	5	0	18880	2
50	Volkswagen	Touran 2.0 FSI	150	longue	7	5	0	27340	2
51	Volkswagen	Touran 2.0 FSI	150	longue	7	5	1	19138	2

Ce sont des véhicules longs avec des places allant jusqu'à **7** et une puissance comprise entre 102 et 197. Elles sont également disponibles en **occasion** avec des prix proportionnels à la puissance. Nous déduisons que ce sont des **véhicules de famille.**

Pour la catégorie 3 :

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	label
0	Audi	A2 1.4	75	courte	5	5	0	18310	3
1	Audi	A2 1.4	75	courte	5	5	1	12817	3
9	Daihatsu	Cuore 1.0	58	courte	5	3	0	8850	3
18	Kia	Picanto 1.1	65	courte	5	5	0	8990	3
19	Lancia	Ypsilon 1.4 16V	90	courte	5	3	0	13500	3
20	Lancia	Ypsilon 1.4 16V	90	courte	5	3	1	9450	3
25	Mini	Copper 1.6 16V	115	courte	5	5	0	18200	3
26	Mini	Copper 1.6 16V	115	courte	5	5	1	12740	3
30	Peugeot	1007 1.4	75	courte	5	5	0	13750	3
31	Peugeot	1007 1.4	75	courte	5	5	1	9625	3
48	Volkswagen	Polo 1.2 6V	55	courte	5	3	0	12200	3
49	Volkswagen	Polo 1.2 6V	55	courte	5	3	1	8540	3

Ce sont des véhicules de très petites tailles avec une puissance maximale égale à **115.** Disponible en neuf et en occasion avec 3 portes le plus souvent. Nous pouvons dire que ce sont des voitures **citadines,** généralement pour une personne.

Pour la catégorie 4 :

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	label
6	BMW	M5	507	très longue	5	5	0	94800	4
7	BMW	M5	507	très longue	5	5	1	66360	4
23	Mercedes	S500	306	très longue	5	5	0	101300	4
24	Mercedes	S500	306	très longue	5	5	1	70910	4

Ce sont des voitures avec une puissance minimale de 306 et allant jusqu'à 507. Nous n'avons que 4 véhicules dans cette catégorie. Leur rareté, leur prix et leur puissance nous font dire que ce sont des **véhicules de luxe.** Au final, nous avons obtenus 5 catégorie de véhicule qui sont :

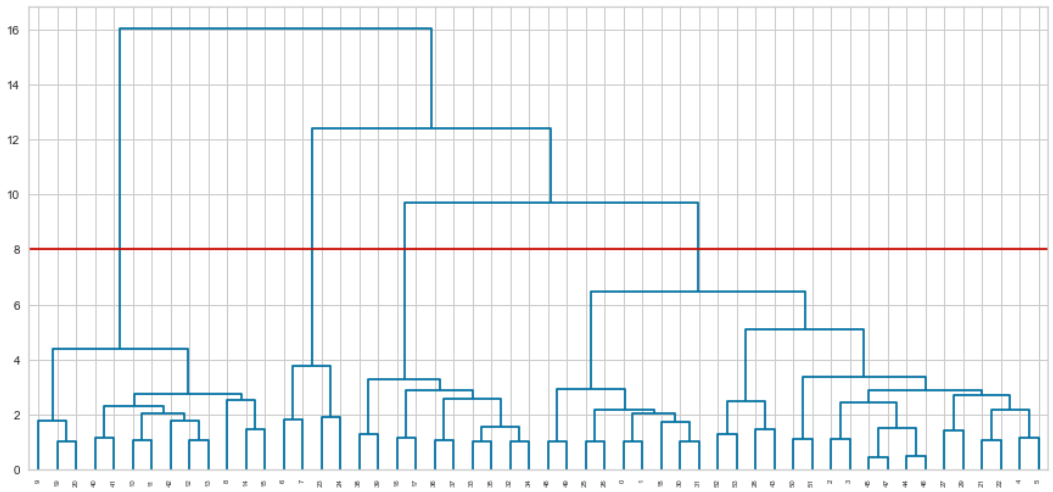
- normaux
- vieux
- citadines
- familles
- luxe

Tout le travail précédent s'est effectué sans tenir compte des informations dans le dataset CO2. La regroupement des véhicules s'est fait le plus sur la variable longueur.

4- Ajout de nouvelle variables

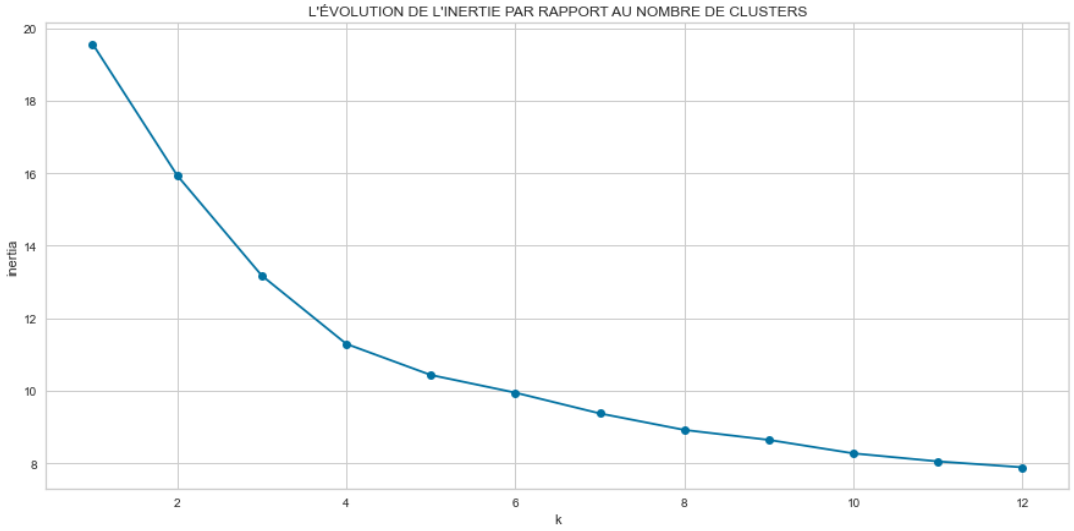
Dans la suite, nous avons ajouté des informations supplémentaires sur les données de catalogue et immatriculation (bonus-malus, coût et énergie, rejetco2). En procédant comme précent nous avons les graphismes suivants :

4-1- **Agglomérative cluster**



On obtient un nombre de cluster égale à 4

4-2- Evolution de l'inertie



Le point d'inflexion dans ce cas est 4, d'où le nombre de cluster

4-3- Présentation des classes

- Classe 0

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	bonus_malus	rejetsco2gkm	coutenergie	label
0	AUDI	A2 1.4	75	courte	5	5	False	18310	2851.800000	26.100000	191.600000	0
1	AUDI	A2 1.4	75	courte	5	5	True	12817	2851.800000	26.100000	191.600000	0
2	AUDI	A3 2.0 FSI	150	moyenne	5	5	False	28500	2851.800000	26.100000	191.600000	0
3	AUDI	A3 2.0 FSI	150	moyenne	5	5	True	19950	2851.800000	26.100000	191.600000	0
4	BMW	120i	150	moyenne	5	5	False	35800	7200.052632	39.263158	80.526316	0
5	BMW	120i	150	moyenne	5	5	True	25060	7200.052632	39.263158	80.526316	0
18	KIA	Picanto 1.1	65	courte	5	5	False	8990	1376.500000	15.500000	132.750000	0
21	MERCEDES	A200	136	moyenne	5	5	False	25900	8265.325424	187.627119	749.979661	0
22	MERCEDES	A200	136	moyenne	5	5	True	18130	8265.325424	187.627119	749.979661	0
25	MINI	Copper 1.6 16V	115	courte	5	5	False	18200	1376.500000	21.500000	126.000000	0
26	MINI	Copper 1.6 16V	115	courte	5	5	True	12740	1376.500000	21.500000	126.000000	0
27	NISSAN	Almera 1.8	115	moyenne	5	5	False	16450	5802.400000	160.000000	681.200000	0
28	NISSAN	Maxima 3.0 V6	200	très longue	5	5	False	30000	5802.400000	160.000000	681.200000	0
29	NISSAN	Primera 1.6	109	longue	5	5	False	18650	5802.400000	160.000000	681.200000	0
30	PEUGEOT	1007 1.4	75	courte	5	5	False	13750	1376.500000	15.833333	144.166667	0
31	PEUGEOT	1007 1.4	75	courte	5	5	True	9625	1376.500000	15.833333	144.166667	0
43	SKODA	Superb 2.8 V6	193	très longue	5	5	False	31790	7113.777778	27.555556	98.888889	0
44	VOLKSWAGEN	Golf 2.0 FSI	150	moyenne	5	5	False	22900	4537.857143	23.428571	96.000000	0
45	VOLKSWAGEN	Golf 2.0 FSI	150	moyenne	5	5	True	16029	4537.857143	23.428571	96.000000	0
46	VOLKSWAGEN	New Beatle 1.8	110	moyenne	5	5	False	26630	4537.857143	23.428571	96.000000	0
47	VOLKSWAGEN	New Beatle 1.8	110	moyenne	5	5	True	18641	4537.857143	23.428571	96.000000	0
48	VOLKSWAGEN	Polo 1.2 6V	55	courte	5	3	False	12200	4537.857143	23.428571	96.000000	0
49	VOLKSWAGEN	Polo 1.2 6V	55	courte	5	3	True	8540	4537.857143	23.428571	96.000000	0
50	VOLKSWAGEN	Touran 2.0 FSI	150	longue	7	5	False	27340	4537.857143	23.428571	96.000000	0
51	VOLKSWAGEN	Touran 2.0 FSI	150	longue	7	5	True	19138	4537.857143	23.428571	96.000000	0

➢ Classe 1

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	bonus_malus	rejetsco2gkm	coutenergie	label
6	BMW	M5	507	très longue	5	5	False	94800	7200.052632	39.263158	80.526316	1
7	BMW	M5	507	très longue	5	5	True	66360	7200.052632	39.263158	80.526316	1
23	MERCEDES	S500	306	très longue	5	5	False	101300	8265.325424	187.627119	749.979661	1
24	MERCEDES	S500	306	très longue	5	5	True	70910	8265.325424	187.627119	749.979661	1
52	VOLVO	S80 T6	272	très longue	5	5	False	50500	8753.000000	42.454545	72.727273	1
53	VOLVO	S80 T6	272	très longue	5	5	True	35350	8753.000000	42.454545	72.727273	1

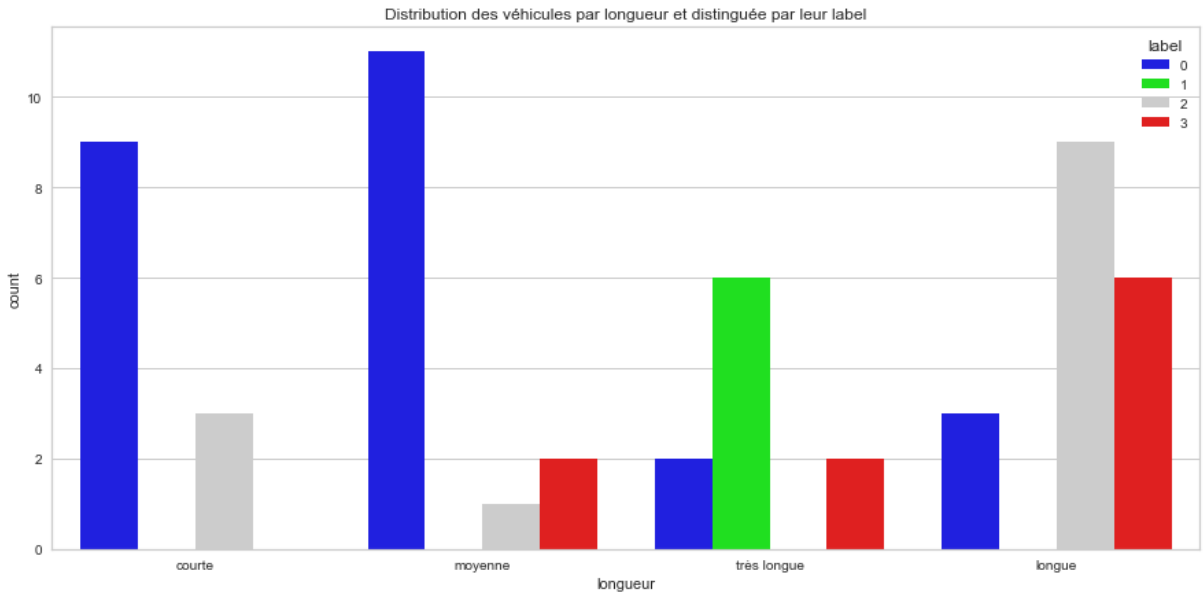
➢ Classe 2

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	bonus_malus	rejetsco2gkm	coutenergie	label
8	DACIA	Logan 1.6 MPI	90	moyenne	5	5	False	7500	6794.36092	6794.36092	6794.36092	2
9	DAIHATSU	Cuore 1.0	58	courte	5	3	False	8850	6794.36092	6794.36092	6794.36092	2
10	FIAT	Croma 2.2	147	longue	5	5	False	24780	6794.36092	6794.36092	6794.36092	2
11	FIAT	Croma 2.2	147	longue	5	5	True	17346	6794.36092	6794.36092	6794.36092	2
12	FORD	Mondeo 1.8	125	longue	5	5	False	23900	6794.36092	6794.36092	6794.36092	2
13	FORD	Mondeo 1.8	125	longue	5	5	True	16730	6794.36092	6794.36092	6794.36092	2
14	HONDA	FR-V 1.7	125	longue	7	5	False	19550	6794.36092	6794.36092	6794.36092	2
15	HYUNDAÏ	Matrix 1.6	103	longue	7	5	False	15960	6794.36092	6794.36092	6794.36092	2
19	LANCIA	Ypsilon 1.4 16V	90	courte	5	3	False	13500	6794.36092	6794.36092	6794.36092	2
20	LANCIA	Ypsilon 1.4 16V	90	courte	5	3	True	9450	6794.36092	6794.36092	6794.36092	2
40	SAAB	9.3 1.8T	150	longue	5	5	False	38600	6794.36092	6794.36092	6794.36092	2
41	SAAB	9.3 1.8T	150	longue	5	5	True	27020	6794.36092	6794.36092	6794.36092	2
42	SEAT	Toledo 1.6	102	longue	5	5	False	18880	6794.36092	6794.36092	6794.36092	2

➢ Classe 3

	marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix	bonus_malus	rejetsco2gkm	coutenergie	label
16	JAGUAR	X-Type 2.5 V6	197	longue	5	5	False	37100	-6000.0	0.0	271.0	3
17	JAGUAR	X-Type 2.5 V6	197	longue	5	5	True	25970	-6000.0	0.0	271.0	3
32	RENAULT	Espace 2.0T	165	longue	7	5	False	30350	-6000.0	0.0	206.0	3
33	RENAULT	Espace 2.0T	165	longue	7	5	True	21245	-6000.0	0.0	206.0	3
34	RENAULT	Laguna 2.0T	170	longue	5	5	False	27300	-6000.0	0.0	206.0	3
35	RENAULT	Laguna 2.0T	170	longue	5	5	True	19110	-6000.0	0.0	206.0	3
36	RENAULT	Megane 2.0 16V	135	moyenne	5	5	False	22350	-6000.0	0.0	206.0	3
37	RENAULT	Megane 2.0 16V	135	moyenne	5	5	True	15644	-6000.0	0.0	206.0	3
38	RENAULT	Vel Satis 3.5 V6	245	très longue	5	5	False	49200	-6000.0	0.0	206.0	3
39	RENAULT	Vel Satis 3.5 V6	245	très longue	5	5	True	34440	-6000.0	0.0	206.0	3

Nous remarquons que le regroupement par classe ne s'est pas fait selon une seule variable. Précédemment, nous avions une longueur unique par classe mais ce n'est plus le cas.



Toutes les longueurs sont présentes dans toutes les classes. Finalement, nous considérerons 4 classes de véhicules.

II- CLASSIFICATION ET PREDICTION

Après avoir déterminé les catégories de véhicule et leur attribuer des labels, nous allons effectuer une classification basée sur ces catégories.

1- Les algorithmes de classifications

Pour la classification, nous avons utilisé deux approches combinées, le **grid search** et le **voting classifier**. Le grid search pour la recherche de paramètres optimaux et le voting classifier pour un ensemble Learning (regroupement de plusieurs modèles).

1-1- Recherche de paramètres optimaux (grid search)

Pour ce faire, nous définissons une classe **Trainer** qui prend en entrées les jeux de données (train & test) et les modèles ensuite les entraîne, affiche les scores et logue les métriques et model grâce à **MLFLOW**.

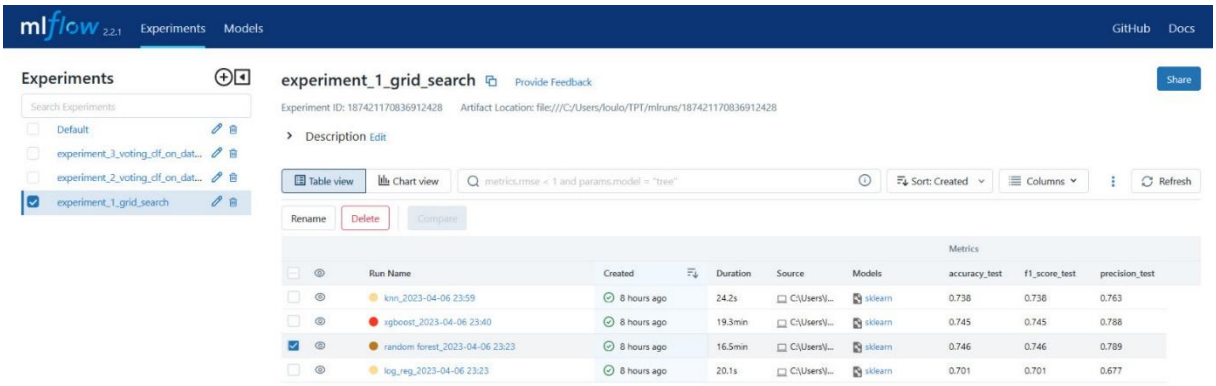
- **Expérience 1**

```
Entrée [9]:
models_test = {
    "log_reg": LogisticRegression(random_state=RANDOM_STATE, n_jobs=-1),
    "random forest" : RandomForestClassifier(bootstrap=False, n_jobs=-1, random_state=RANDOM_STATE),
#     "adaboost" : AdaBoostClassifier(random_state=RANDOM_STATE),
    "xgboost" : XGBClassifier(subsample=0.2, random_state=RANDOM_STATE),
    "knn" : KNeighborsClassifier(n_jobs=-1),
}
params_grid = [
    {'solver' : ["lbfgs", "elasticnet"], "penalty" : ["l2", "l1"], "C":[1, .1, .01] ,"max_iter": [1000]},
    {'n_estimators': [300], 'max_depth': [None, 5, 10, 15], 'min_samples_split': [2, 5, 10], 'min_samples_leaf':[2, 5, 10],
#     {'n_estimators': [250], 'learning_rate': [0.1, 0.05], 'base_estimator': [DecisionTreeClassifier(min_samples_split=5, m
    {'max_depth': [3, 5, 7], 'learning_rate': [0.1, 0.01, 0.05], 'n_estimators': [200, 300, 400]},
    {'n_neighbors': [15, 20, 25], 'weights': ['uniform']},
]
scorings = [['accuracy', 'precision']] * len(models_test)


trainer = Trainer(X_train, X_test, y_train, y_test, models_test, params_grid, scorings, experiment_id=1)
accs, precs, f1s = trainer.run()
```

Nous avons utilisé les algorithmes de Logistic Regression, de Random Forest, Xgboost et KNN avec les métriques de accuracy et precision. On obtient ainsi les résultats suivants :

```
log_reg
----------
accuracy score on train : 69.83%
accuracy score on test: 70.05%

precision score on train : 67.57%
precision score on test : 67.74%

f1 score on train: 69.83%
f1 score on test: 70.05%

random forest
----------
accuracy score on train : 76.77%
accuracy score on test: 74.61%

precision score on train : 81.61%
precision score on test : 78.95%

f1 score on train: 76.77%
f1 score on test: 74.61%

xgboost
----------
accuracy score on train : 76.29%
accuracy score on test: 74.53%

precision score on train : 80.99%
precision score on test : 78.75%

f1 score on train: 76.29%
f1 score on test: 74.53%

knn
----------
accuracy score on train : 76.66%
accuracy score on test: 73.80%

precision score on train : 80.07%
precision score on test : 76.34%

f1 score on train: 76.66%
f1 score on test: 73.80%
```

Le random Foreste présente de meilleurs résultats, ensuite vient le Xgboost.



1-2- Ensemble Learning (voting classifier)

Nous utilisons ensuite les paramètres optimaux de chaque modèle obtenu grâce au grid search pour effectuer le voting classifier.

- **Expérience 2**

Entrée [6]:

```
# Define models
p = {'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'lbfgs'}
model1 = LogisticRegression(**p, random_state=RANDOM_STATE, n_jobs=-1)

p = {'criterion': 'gini', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 10, 'min_samples_split': 2, 'n_estima
model2 = RandomForestClassifier(**p, bootstrap=True, n_jobs=-1, random_state=RANDOM_STATE) #bootsrap=Fasle : use all dataset

p =  {'learning_rate': 0.01, 'max_depth': 7, 'n_estimators': 200}
model3 = XGBClassifier(**p, random_state=RANDOM_STATE)

p = {'n_neighbors': 7, 'weights': 'uniform'}
model5 = KNeighborsClassifier(n_jobs=-1)

```

Entrée [7]:

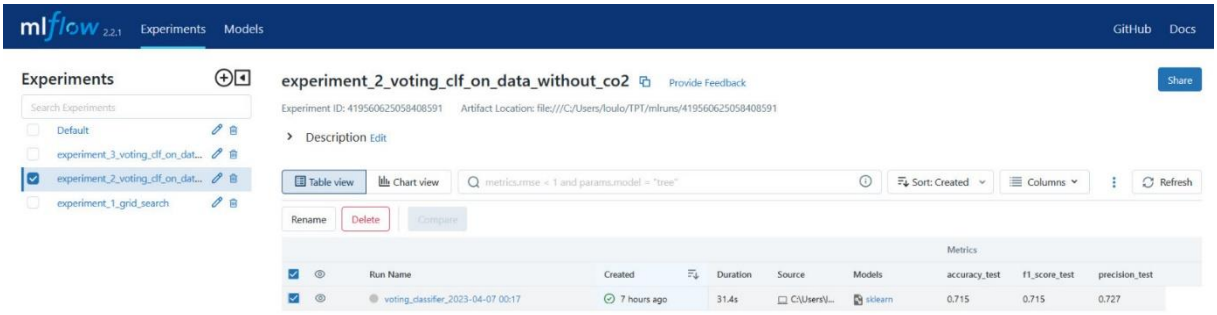
```
EXPERIMENT_ID = 2

# Load expermient or create new
experiment_name = 'experiment_'+ str(EXPERIMENT_ID)+ '_voting_clf_on_data_without_co2'
experiment_id = mlflow.get_experiment_by_name(experiment_name)
experiment_id = experiment_id.experiment_id if experiment_id else mlflow.create_experiment(experiment_name)

with mlflow.start_run(experiment_id=experiment_id, run_name="voting_classifier_" + str(dt.datetime.now())[0:-10] ) :
    # Define voting classifier and train
    voting_classifier = VotingClassifier(estimators=[('lr', model1), ('rf', model2), ('ada', model3), ('knn', model5)], voti
    voting_classifier.fit(X_train, y_train)

    pred_train = voting_classifier.predict(X_train)
    pred_test = voting_classifier.predict(X_test)

    # compute and logs metrics for each model
    compute_metrics(y_train, pred_train, y_test, pred_test, model="voting_clf")

    mlflow.sklearn.log_model(voting_classifier, "voting_clf")
    mlflow.log_param("model", "voting_clf (logistic_regression - random_forest - adaboost - gaussian_nb - knn)")
    mlflow.set_tag('estimator_class', type(voting_classifier))
```

train accuracy : 76.47%

On définit chaque modèle avec ses paramètres optimaux et on les passe en entrée dans le voting classifier. On obtient les résultats suivants :

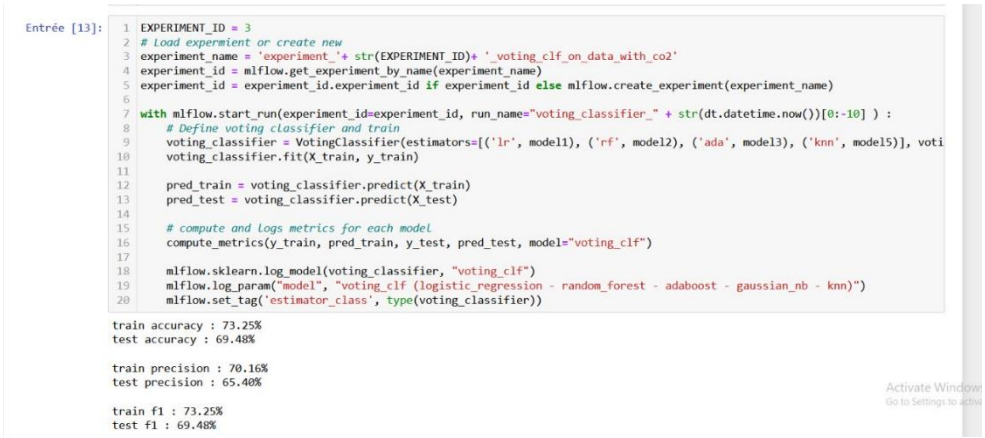
```
train accuracy : 76.47%
test accuracy : 71.51%

train precision : 79.42%
test precision : 72.67%

train f1 : 76.47%
test f1 : 71.51%
```

Nous avons de bons résultats mais pas autant que celui du random forest seul.



1-3- Ajout des information du dataset CO2

Nous ajoutons les information de CO2 dans le dataset de catalogue et d'immatriculation :

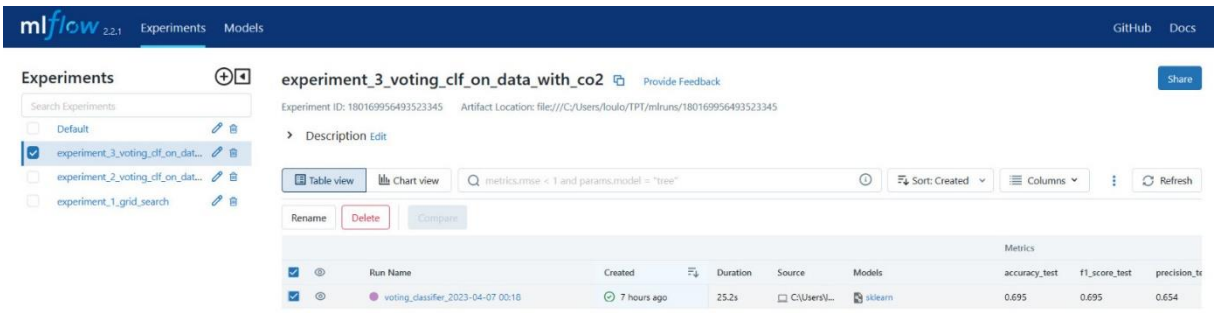


Entrée [13]:

```
EXPERIMENT_ID = 3
# Load experiment or create new
experiment_name = 'experiment_'+ str(EXPERIMENT_ID)+ '_voting_clf_on_data_with_co2'
experiment_id = mlflow.get_experiment_by_name(experiment_name)
experiment_id = experiment_id.experiment_id if experiment_id else mlflow.create_experiment(experiment_name)

with mlflow.start_run(experiment_id=experiment_id, run_name="voting_classifier_" + str(dt.datetime.now())[0:-10] ) :
    # Define voting classifier and train
    voting_classifier = VotingClassifier(estimators=[('lr', model1), ('rf', model2), ('ada', model3), ('knn', model5)], voti
    voting_classifier.fit(X_train, y_train)

    pred_train = voting_classifier.predict(X_train)
    pred_test = voting_classifier.predict(X_test)

    # compute and logs metrics for each model
    compute_metrics(y_train, pred_train, y_test, pred_test, model="voting_clf")

    mlflow.sklearn.log_model(voting_classifier, "voting_clf")
    mlflow.log_param("model", "voting_clf (logistic_regression - random_forest - adaboost - gaussian_nb - knn)")
    mlflow.set_tag('estimator_class', type(voting_classifier))
```

```
train accuracy : 73.25%
test accuracy : 69.48%

train precision : 70.16%
test precision : 65.40%

train f1 : 73.25%
test f1 : 69.48%
```

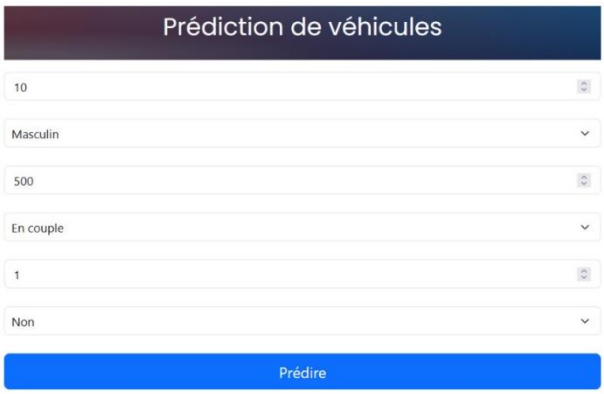
L'ajout de nouvelles variable n'a pas amélioré notre score.



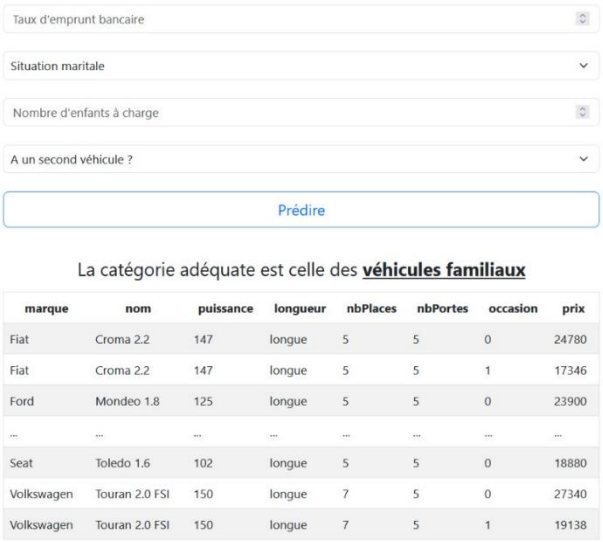
2- Api et page client

Pour l'utilisation de notre modèle, nous avons mis en place une API et une page web. Cette page est utilisée pour recueillir les informations des clients et l'api les envoie dans notre modèle. Le résultat (la prédiction) du modèle est ensuite retourné à notre API qui permet à notre page de l'afficher

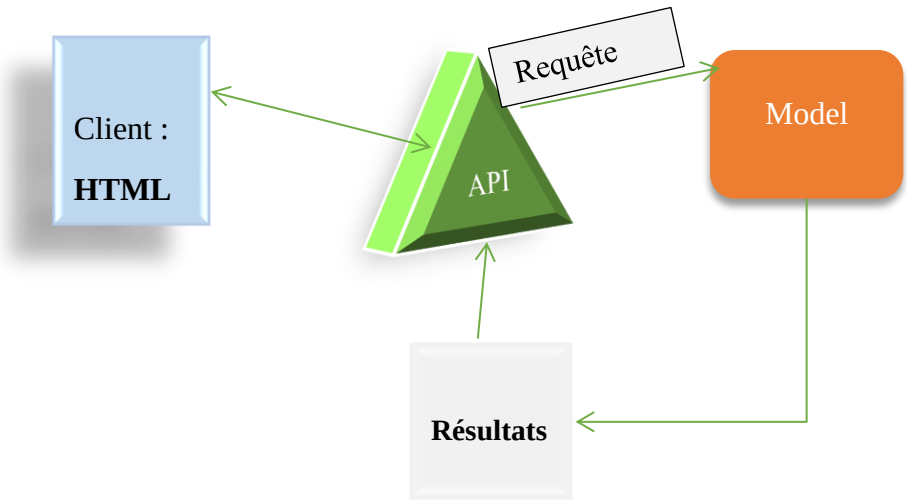
.



Ce formulaire permet de rentrer les informations d'un véhicule. Ces informations sont ensuite envoyées à notre API.



marque	nom	puissance	longueur	nbPlaces	nbPortes	occasion	prix
Fiat	Croma 2.2	147	longue	5	5	0	24780
Fiat	Croma 2.2	147	longue	5	5	1	17346
Ford	Mondeo 1.8	125	longue	5	5	0	23900
...	...	...	...	...	...	...	...
Seat	Toledo 1.6	102	longue	5	5	0	18880
Volkswagen	Touran 2.0 FSI	150	longue	7	5	0	27340
Volkswagen	Touran 2.0 FSI	150	longue	7	5	1	19138



Conclusion

Dans ce projet, nous avons dans un premier temps chercher le nombre de clusters nécessaire pour regrouper les véhicules. Nous avons obtenu 5 clusters dans un premier temps mais le groupement était fait selon une seule variable, la longueur. Nous avons ajouté des informations supplémentaires dans nos données pour obtenir 4 clusters avec un regroupement basé sur plusieurs variables. Dans un second temps, nous avons effectué une classification qui reposait sur l'utilisation de plusieurs algorithmes de machine Learning (ensemble Learning) pour effectuer nos prédictions. Enfin, nous avons mis en place une api et une page web pour entrer des caractéristiques de véhicules et effectuer nos prédictions en fonction des données de l'utilisateur. Pour notre part, ce fut une très belle expérience de découvrir le machine Learning dans plusieurs aspects