2022

PROJET MACHINE LEARNING

Prédiction de prix de véhicules d'occasion

EL KADIRI MOHAMED
EL JARUDI FAYCAL
EL MESSOUAL EL MEHDI

Sommaire

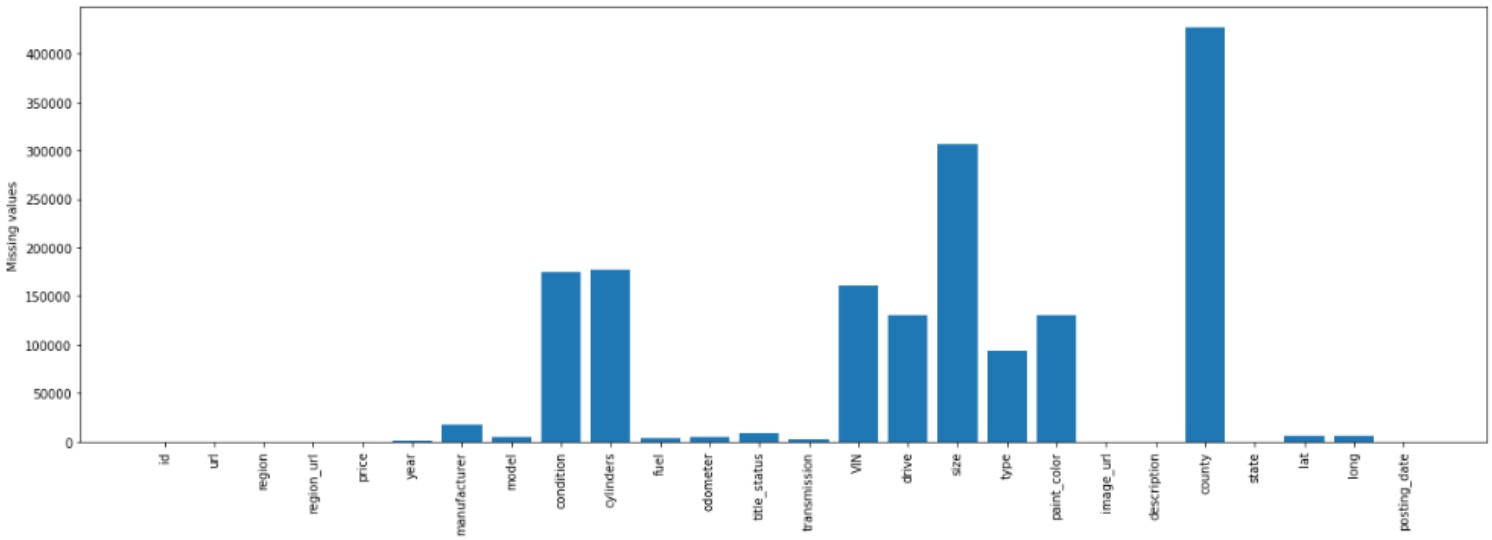
Data Preprocessing

- Objectif:

L'objectif de cette étape est de nettoyer les données du DataSet qu'on va utiliser pour entrainer notre modèle.

Elle consiste à éliminer ou remplacer les valeurs nulles ou inexistantes ainsi que se débarrasser des colonnes qui n'affectent pas les prix des voitures d'occasion.

Après avoir visualiser quelques lignes de notre DataSet, on constate qu'il y'a 26 colonnes au totales dont leurs noms et les valeurs nulles pour chacun d'eux sont présentés dans le diagramme ci-dessous :



On constate que les colonnes : **id, url, region_url, description, image_url, lat, long, posting_date, county,** VIN ne peuvent pas affecter le prix des voitures d'occasion car ce sont juste des colonnes descriptifs contenant des informations supplémentaires comme: le lien vers l'image de la voiture, sa description etc…

En plus de ça, on remarque que plus de 50% des valeurs de la colonne size sont des valeurs nulles, donc il doit être abandonné

```
In [102]: columnsToDrop = ['id', 'url', 'region_url', 'description', 'image_url', 'lat', 'long', 'posting_date', 'county', 'VIN', 'size']
          dataFrame.drop(columnsToDrop, axis = 1, inplace = True)
```

Pour le prix, on constate qu'il y'a des anomalies dans les prix de quelques voitures, donc on a pris la décision de prendre juste les lignes dont le prix est entre 1 000 et 200 000

```
In [104]: dataFrame = dataFrame[(dataFrame.price >= 1000) & (dataFrame.price <= 200000)]
          dataFrame.shape
Out[104]: (380441, 15)
```

Pour le reste des colonnes, on abandonne juste les lignes qui contiennent des valeurs nulles car elles peuvent affecter directement le prix des voitures, ce qu'on va voir dans la partie du 'Data Visualisation'

```
In [105]: dataFrame.dropna(inplace = True)
```

Data Visualization

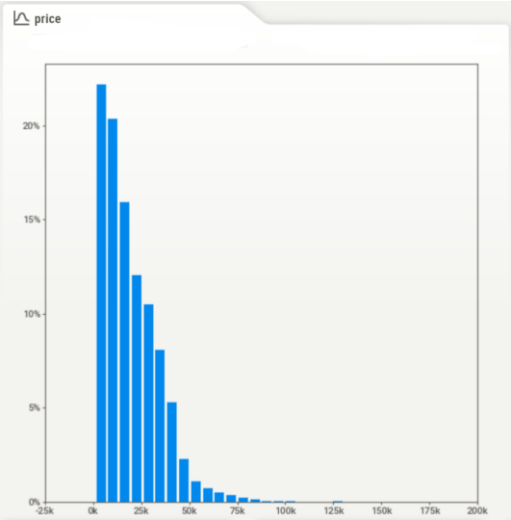
- Vu que la DataSet avait des colonnes qui n'ont absolument rien avoir avec l'objectif du projet, nous avons pris la décision d'écarter les données inutiles dans la phase de « Data Visualization », et ne prendre que les données pertinentes.

- Pour ceci, on a pris la décision d'utiliser une « Third-Party Library » nommée **SweetViz** qui donne un bilan détailler sur l'ensemble des **Features** ainsi que la relation de la **target** avec eux.

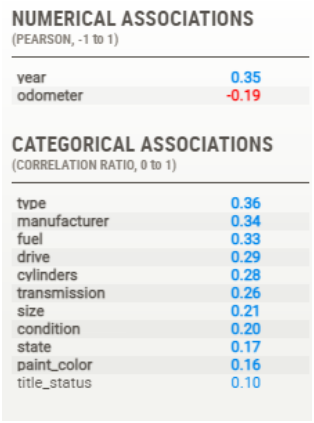
- Le bilan va être joint avec le rapport, or on se focalise dans cette partie à l'analyse de ce bilan.

1- Analyse des **Features** :

a. Price :

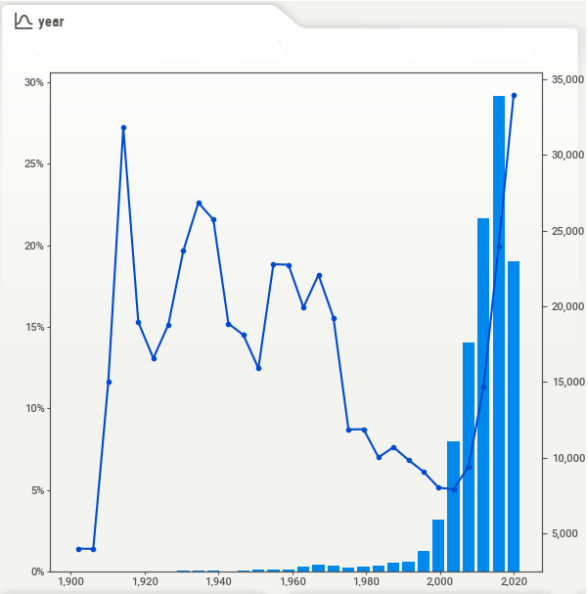


-On a un range de prix allant de 1K à 200K, avec un pourcentage de 70% pour les voiture de maximum 25K.
-Pour le taux de corrélation :



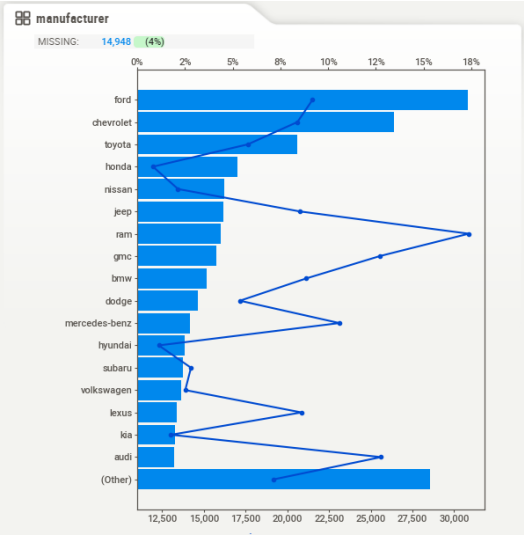
- Le type de voiture (Truck, mini-van, …) est la valeur qui affecte le plus le Target.
- L'année de production, La marque, et le type de carburant ont un très grand impact sur le prix de la voiture.
- Notant par exemple que l'odomètre n'a aucun impact sur le prix de la voiture

b. Year :



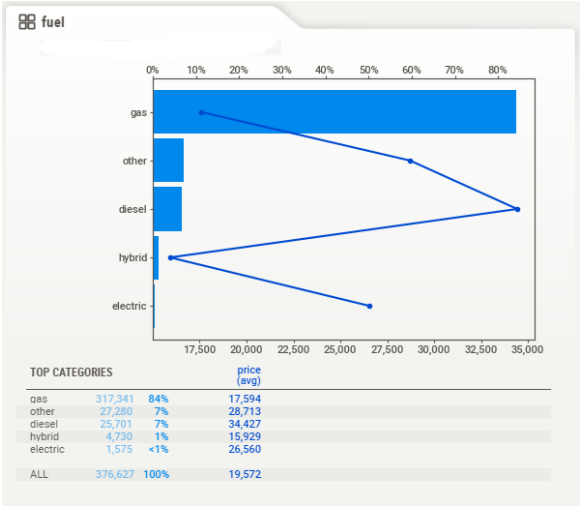
- Ce qu'on voit est parfaitement logique, a vrai dire le prix de la voiture diminue avec son âge, or les voitures qui sont très vielles, leur prix est vachement haut car il sent acheté en raison de collection.
- Les voitures qui sont produits avant 1960 sont chère aussi que les voitures récemment produits.

c. Manufacturer :

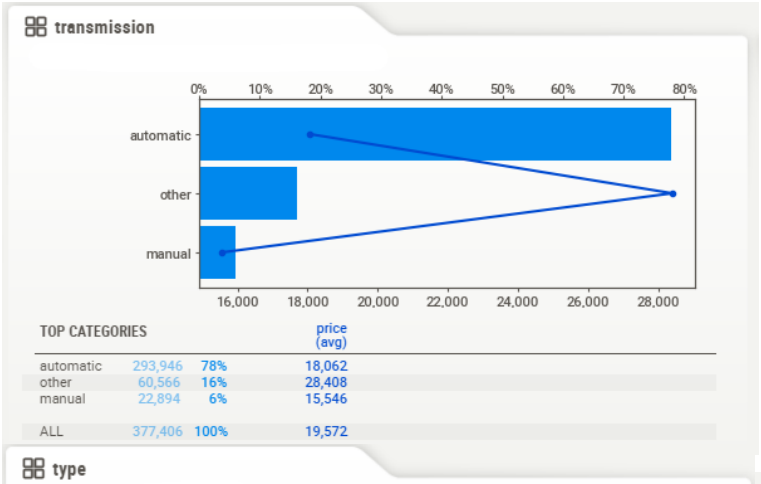


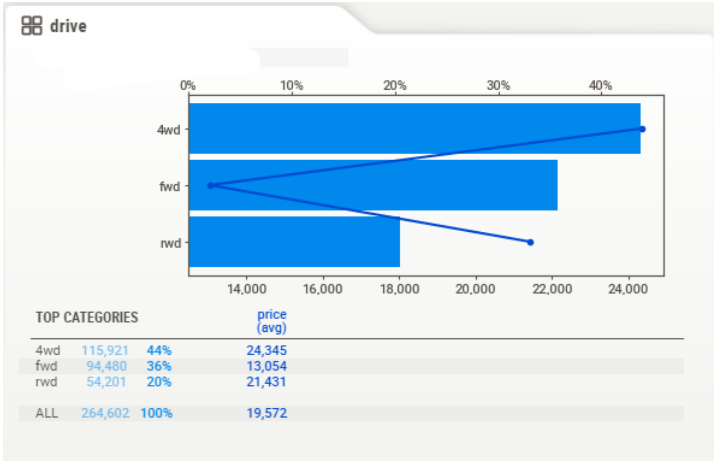
- Les marques RAM, GMC sont connues par ses « Trucks » qui ont généralement un prix élevé.
- Après vient les marques de luxe comme « Mercedes-Benz » et « Audi »

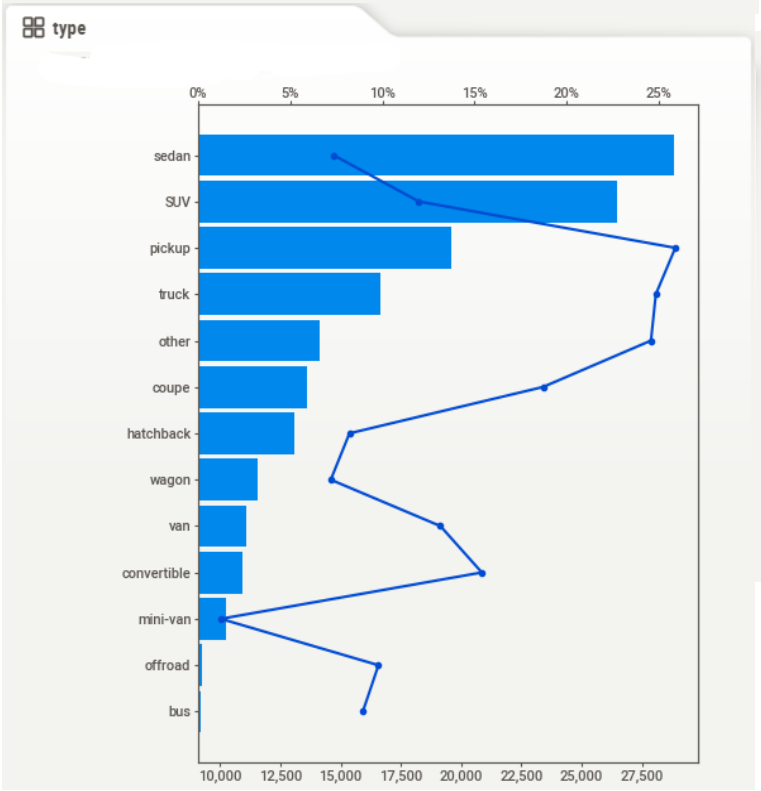
d. Fuel :

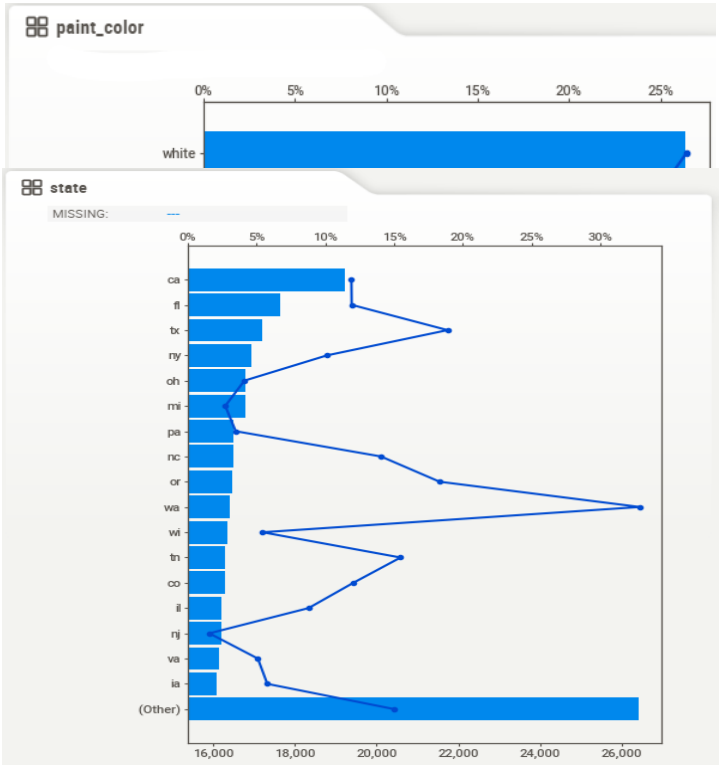


e. Le reste des **Features** :









Training

- On a un problème de régression, donc on a eu la décision de choisir 3 modèles exemplaires :
  - La Régression Linéaire avec plusieurs variables.
  - Les Arbres Aléatoires.
  - Cas des plus proches voisins.

- Après la phase de l'entrainement, on a obtenu les résultats suivants :

```
ModelLinearRegression = train(X_train, y_train, LinearRegression())
print("Linear Regression : ",score(X_test, y_test, ModelLinearRegression))
```

```
Linear Regression :  0.40423268065738005
```

```
ModelRandomForest = train(X_train, y_train, RandomForestRegressor())
print("Random Forest : ",score(X_test, y_test, ModelRandomForest))
```

```
Random Forest :  0.8972148205746237
```

```
ModelKNN = train(X_train, y_train, KNeighborsRegressor())
print("Random Forest : ",score(X_test, y_test, ModelKNN))
```

```
Random Forest :  0.5807781774119936
```

- Donc on va choisir le model RandomForest, car il a la plus grande précision.
- On entame la validation croisée :

```
scores = cross_val_score(RandomForestRegressor(), X, Y, cv=3)
scores
```

	0
0	0.805068
1	0.882061
2	0.862469

Qui résulte un score de : **0.849 ~ 85%.**