Apprentissage Automatique 1

TP N°2 – Régression Multiple

Objectifs du TP :

- Appliquer la Multiple
- Calculer la prédiction
- Calculer l'erreur
- Visualiser les résultats

Outils et version :

- Anaconda, Jupyter
- Python

Exercice N°1 : Régression Multiple " Exemple prix des maisons"

Créer un classifieur de la régression linéaire Multiple pour prédire les prix des maisons en fonction de la surface et le nombre de chambres.

Appliquer les étapes nécessaires pour créer le classificateur de la Régression linéaire Multiple afin d'analyser les prix des maisons.

1. Importer les bibliothèques nécessaires : pandas, numpy, matplotlib, et sklearn…etc
2. Importer la base de données par la fonction : read_excel()
3. Afficher les premières lignes chargées de votre fichier Excel
4. Récupérer le prix : les valeurs observées pour la variable Cible
5. Récupérer les variables prédictives : La superficie en pieds² et le nombre des chambres
6. Afficher les données utilisées sous forme d'un graphe 3D
7. Créer le modèle de la régression linéaire Multiple.
8. Appliquer l'apprentissage automatique par la régression Multiple
9. Calculer la prédiction
10. Afficher les résultats en fonction de la surface, nombre de chambre et la prédiction.
11. Commenter les résultats
12. Tester la prédiction sur l'exemple : (surface : 450, nombre de chambres : 5)

Exercice N°2 : Régression Multiple Exemple Prédiction des prix des voitures

Une société automobile aspire à pénétrer le marché international en y installant son unité de fabrication et en produisant des voitures localement pour faire concurrence à ses homologues.

Ils ont engagé une société de conseil automobile pour comprendre les facteurs dont dépend le prix des voitures. Plus précisément, ils veulent comprendre les facteurs affectant le prix des voitures sur le marché américain, car ceux-ci peuvent être très différents du marché chinois. L'entreprise veut savoir :

Quelles variables sont importantes pour prédire le prix d'une voiture ?

Dans quelle mesure ces variables décrivent le prix d'une voiture ?

Sur la base de diverses études de marché, la société a rassemblé un vaste ensemble de données sur différents types de voitures sur le marché.

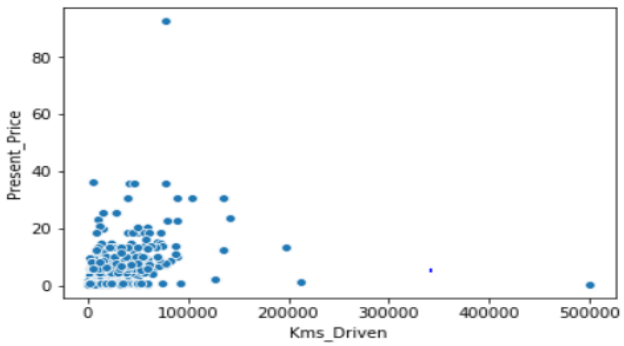
Pour cela, Créer un classifieur de la régression linéaire Multiple pour prédire les prix des voitures en fonction de plusieurs paramètres.

Appliquer les étapes nécessaires pour créer le classificateur de la Régression linéaire Multiple afin d'analyser les prix des voitures.

1. Importer les bibliothèques nécessaires : **numpy, pandas, matplotlib.pyplot et seaborn**
2. Importer la base de données par la fonction : read_csv()
3. Afficher les premières lignes chargées de votre fichier csv.
4. Afficher le nombre de lignes et de colonnes
5. Afficher d'autre information sur les données utilisés notamment le type de chaque variable int , float non nulle etc par la fonction suivante: **info()**
6. Afficher la somme des cases nulle pour chaque variable par la fonction suivante : **isnull().sum()**
7. Afficher une description totale de la base de données par la fonction suivante : **describe()**
8. Extraire le nombre des variables catégorielle notamment par la fonction suivante : **value_counts().**
9. Transformer les variables catégorielles en des variables numériques par la fonction suivante :

Exemple: :
car_dataset.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True)

10. Afficher la base de données avec les nouvelles modifications
11. Afficher l'évaluation des prix pour les voitures en utilisant la bibliothèque seaborn : **sns.distplot(car_dataset['Present_Price'],kde=True)**
12. Commenter les résultats
13. Afficher le type de la voiture en fonction du prix sous forme d'un graphe **boxplot** par la fonction suivante :
sns.boxplot(data=car_dataset, x="Fuel_Type", y="Present_Price")
14. Commenter les résultats
15. Afficher le type de transmission en fonction du prix sous forme d'un graphe **boxplot,** commenter les résultats.
16. Afficher le kilométrage en fonction du prix sous forme d'un graphe comme la figure montre :



17. Afficher la corrélation en les données par le code suivant :

```
plt.figure(figsize=(15,7))
sns.heatmap(car_dataset.corr(),annot=True,cmap="Blues")
plt.title("Data Correlation",size=15)
plt.ylabel("Columns",size=15)
plt.xlabel("Columns",size=15)
plt.show()
```

Commenter les résultats et trouver les corrélations qui existent dans cette base de données.

18. Afficher la somme des données dupliqués par ces deux fonctions :
duplicated()

sum()

19. Localiser les variables indépendantes et dépendante X, Y de te sorte on trouve l'affichage suivant :

Variable de X :

	Year	Selling_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner

Variable y : present_price 0 5.59

20. Diviser le jeu de données utilisé en deux parties (apprentissage et test)
21. Construire le modèle de la régression Linéaire Multiple.
22. Appliquer la phase de l'apprentissage automatique.
23. Appliquer la prédiction.
24. Calculer squared Error: error_score par la fonction suivante : **metrics.r2_score(Y_train, training_data_prediction)**

Nb : N'oubliez pas d'importer la bibliothèque metrics :

```
from sklearn import metrics
```

25. Normaliser le X_train , Y_test par la fonction suivante :

```
from sklearn.preprocessing import StandardScaler
n_scaler = StandardScaler()
x_train = n_scaler.fit_transform(X_train)
x_test = n_scaler.fit_transform(X_test)
```

26. Refaire le calcul de l'apprentissage
27. Appliquer le calcul du score sur les variables de test par la fonction score (X_test,y_test).
28. Appliquer le calcul du score sur les variables de l'apprentissage par la fonction score (X_train,y_train).
29. Calculer l'erreur moyenne quadratique MSE entre le test et la prédiction
30. Calculer l'erreur moyenne quadratique MSE entre le training et la prédiction
31. Commenter les résultats