Université Abdelmalek Essaadi Faculté des Sciences et techniques de Tanger Département Génie Informatique

Cycle Ingénieur : LSI s4

Machine Learning

Pr. EL AACHAK LOTFI

2019/2020

Compte Rendue

Atelier 1 « **Régression** »

Réalisé par :

CHAOUKI Mouad

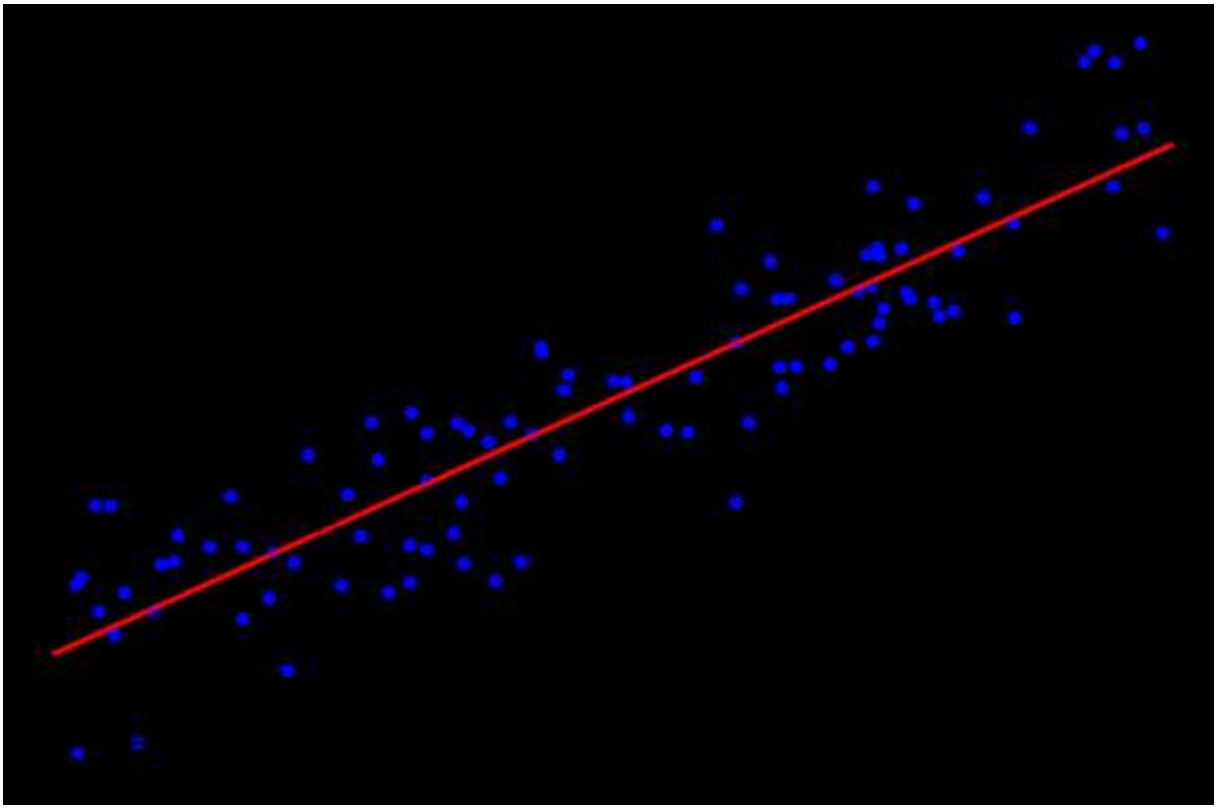
EL-KTIBI Elhassan

Encadré par :

Pr.ELAACHAK Lotfi

Objectif :

L'objective principal de cet atelier est de pratiquer les deux concepts de la régression : la régression linière simple et la régression linière multiple, en traitant des données de plusieurs Data Sets.



Outlies:

Python, Pandas, Sklearn, matplotlib.

Ateliers code sources :


Partie 1 -- Data Visualisation.py


Partie 2 -- Regression Simple.p


Partie 3 -- Regression Multiple

Data Sets :

Expérience et Salaire : https://www.kaggle.com/rohankayan/years-of-experience-and-salarydataset
Assurance : https://www.kaggle.com/sinaasappel/tutorial-multiple-regression/data

Partie 1 – Visualisation des données:

Pour la visualisation des données. Il faut tout d'abord, importer les Datasets en utilisant le paquetage « ***Pandas » :***

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  dsSalaire = pd.read_csv('E:/MDoc/Cycle Courses/Semestres/2nd Year/s4/Machine Learning -- Aachak/Ateliers/Atelier 1/Salary_Data.csv')
5  dsAssurance = pd.read_csv('E:/MDoc/Cycle Courses/Semestres/2nd Year/s4/Machine Learning -- Aachak/Ateliers/Atelier 1/insurance.csv')
```

Puis, on peut afficher notre Datasets en détails, par la fonction ***describe(),*** qui va nous permettre de faire une description à notre Datasets ***:***

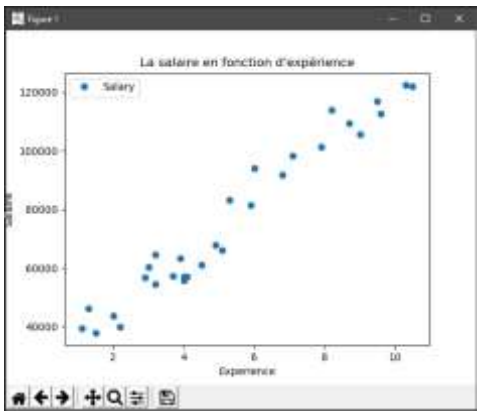
```
6
7  print(dsSalaire.describe())
8  print(dsAssurance.describe())
9
```

	YearsExperience	Salary
count	30.000000	30.000000
mean	5.313333	76003.000000
std	2.837888	27414.429785
min	1.100000	37731.000000
25%	3.200000	56720.750000
50%	4.700000	65237.000000
75%	7.700000	100544.750000
max	10.500000	122391.000000

Visualisation des données simples :

On projette toutes les valeurs du Datasets en deux variables X et Y. Puis on utilise le paquetage « matplotlib.pyplot » pour faire le nuage du points correspondant au X et Y.

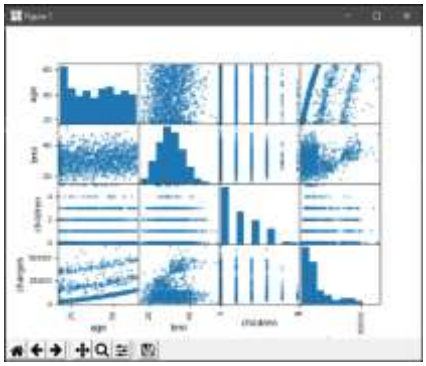
```
10 X = dsSalaire['YearsExperience'].values
11 y = dsSalaire['Salary'].values
12
13 #Data Plot
14 dsSalaire.plot(x='YearsExperience', y='Salary', style='o')
15 plt.title(' La salaire en fonction d\'expérience ')
16 plt.xlabel('Experience')
17 plt.ylabel('Salaire')
18 plt.show()
```



Visualisation des données multiples :

De même, on projette toutes les valeurs du Datasets en deux variables X et Y mais cette on spécifie l'ensemble des colonnes au variable X. Puis on utilise l'outil 'scatter_matrix' du paquetage « pandas.plotting» pour afficher l'histogramme des différents colonnes.

```
21 #Data Plot Assurance
22 scatter_matrix(dsAssurance)
23 plt.show()
24
```



Partie 2 -- Regression Simple :

Pour la partie du regression et avant faire l'apprentissage, il faut qu'on devisé notre Datasets en deux catégories, selon l'indice ***' test_size '***:

- Une pour faire l'apprentissage du notre modèle.
- L'autre faire pour faire le test et l'valuation de notre modèle, c-à-d comparer les données réelles avec les données que notre modèle a prédit.

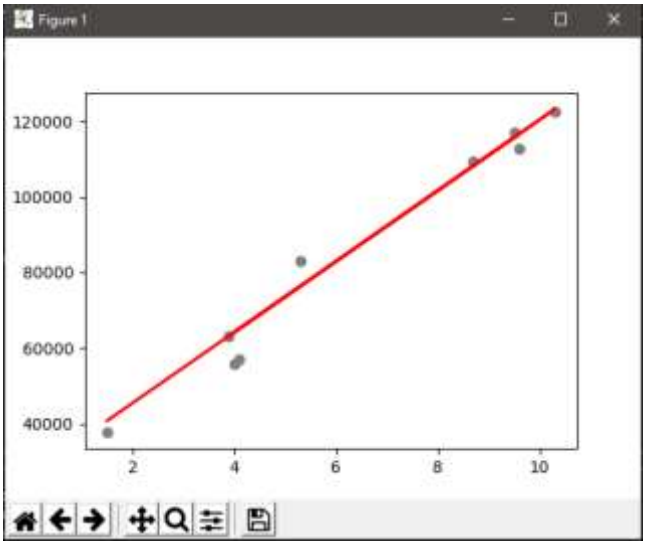
Ensuite, on lance l'apprentissage des données de training **(X_train et Y_train)** avec la fonction ***fit()*** qui se trouve dans l'api « ***sklearn.linear_model*** »

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3  import numpy as np
4  from sklearn.model_selection import train_test_split
5  from sklearn.linear_model import LinearRegression
6  from sklearn import metrics
7
8  DS_FOLDER_PATH = "E:/MDoc/Cycle Courses/Semestres/2nd Year/s4/Machine Learning -- Aachak/Ateliers/Atelier 1/"
9
10 #importer les datasets
11 dsSalaire = pd.read_csv( DS_FOLDER_PATH+'Salary_Data.csv')
12 dsAssurance = pd.read_csv( DS_FOLDER_PATH+'insurance.csv')
13
14 #training the algorithm
15 X = dsSalaire['YearsExperience'].values.reshape(-1,1)
16 y = dsSalaire['Salary'].values.reshape(-1,1)
17
18 #spliter les données à données de test et de training
19 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.05,random_state=0)
20 regressor = LinearRegression()
21 regressor.fit(X_train, y_train)
```

Puis on fait la prédiction de la partie de test, et on affiche le graphe qui visualise la droite de régression de notre modèle.

```
22
23 #prédire les données
24 y_pred = regressor.predict(X_test)
25
```

```
26 #plot the line
27 plt.scatter(X_test, y_test, color='gray')
28 plt.plot(X_test, y_pred, color='red', linewidth=2)
29 plt.show()
30
```



Et finalement, on fait l'valuation à travers l'écart d'erreur :

```
31  #Evaluer les résultats
32  print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
33  print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
34  print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
35
```

On trouve que l'écart est assez grand à cause le manque du data pour faire un bon apprentissage :

```
Mean Absolute Error: 3737.417861878896
Mean Squared Error: 23370078.800832972
Root Mean Squared Error: 4834.260936361728
```

Partie 3 -- Régression Multiple :

Pour cette partie, on aura la même démarche de :

- Importer les données dans deux variables X et Y, en éliminant les colonnes ayant des valeurs se forme des chaines de caractères.
- La partie de l'apprentissage sera la même de la régression linéaire.

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3   import numpy as np
4   from sklearn.model_selection import train_test_split
5   from sklearn.linear_model import LinearRegression
6   from sklearn import metrics
7
8   DS_FOLDER_PATH = "E:/MDoc/Cycle Courses/Semestres/2nd Year/s4/Machine Learning -- Aachak/Ateliers/Atelier 1/"
9
10  #importer les datasets
11  dsSalaire = pd.read_csv( DS_FOLDER_PATH+'Salary_Data.csv')
12  dsAssurance = pd.read_csv( DS_FOLDER_PATH+'insurance.csv')
13
14  print(dsAssurance)
15
16  #training the algorithm
17  X = dsAssurance[['age', 'bmi', 'children']].values
18  y = dsAssurance['charges'].values
19
20   #spliter les données à données de test et de training
21   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,random_state=0)
22   regressor = LinearRegression()
23   regressor.fit(X_train, y_train)
24
25   #prédire les données
26   y_pred = regressor.predict(X_test)
```

On ne peut pas faire la visualisation de la régression à cause des colonnes nombreuses

```
28   #Evaluation des resultas prédit avec les données réels
29   df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
30   dfHead = df.head(5)
31   print(dfHead)
32
33   #Evaluer les résultats
34   print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
35   print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
36   print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
37
```

mais on fait l'affichage des valeurs prédits et les valeurs réelles ainsi que l'écart d'erreurs :

```
[1338 rows x 7 columns]
        Actual     Predicted
0   9724.53000  15773.108836
1   8547.69130  14442.016417
2  45702.02235  18422.477731
3  12950.07120  19490.269480
4   9644.25250  11521.191555
Mean Absolute Error: 9147.160154423707
Mean Squared Error: 135590303.17963448
Root Mean Squared Error: 11644.324934474926
```