SUP'COM
Higher School of Communication of Tunis *

Atelier d'optimisation 2 :

Régression linéaire

Réalisé par : Abdelkader MILADI

INDP2 F

I. Objectifs :

- Comprendre la méthode de régression linéaire.
- Utiliser la méthode de régression linéaire pour résoudre un cas linéaire.
- Implémenter la solution à travers un code en Python.

II. Introduction à la régression :

1) Définition :

En mathématiques, la régression recouvre plusieurs méthodes d'analyse statistique permettant d'approcher une variable à partir d'autres qui lui sont corrélées.

Le but de la régression est d'estimer une valeur numérique de sortie à partir des valeurs d'un ensemble de caractéristiques en entrée (par exemple : estimer le prix d'une maison en se basant sur sa surface, son emplacement, le nombre des étages, etc.)

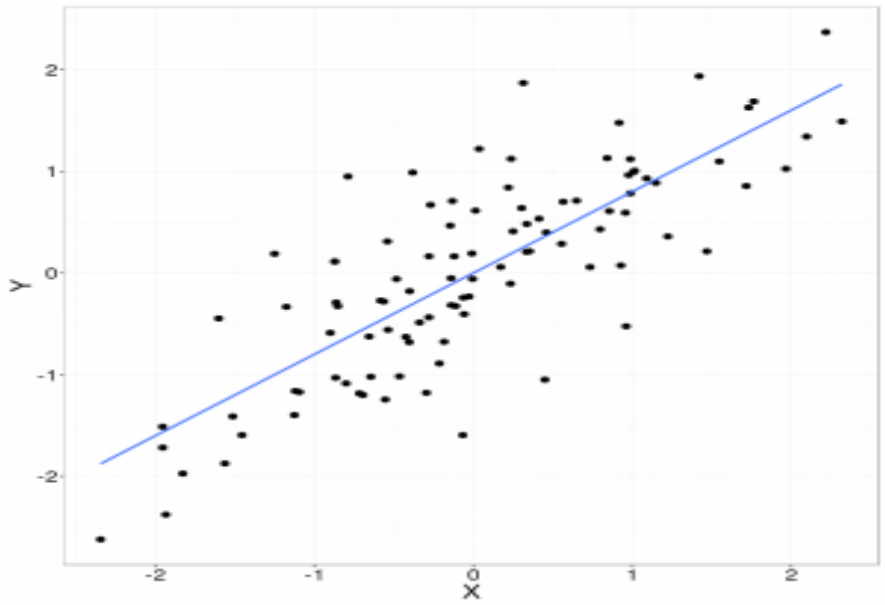
Il est possible de pratiquer l'apprentissage automatique avec le langage Python, en utilisant la bibliothèque « Scikit-Learn » qui propose un module « sklearn.linear_model » contenant une variété de modèles linéaires

2) Cas linéaire :

un modèle de régression linéaire est un modèle de régression qui cherche à établir une relation linéaire entre une variable, dite expliquée, et une ou plusieurs variables, dites explicatives.

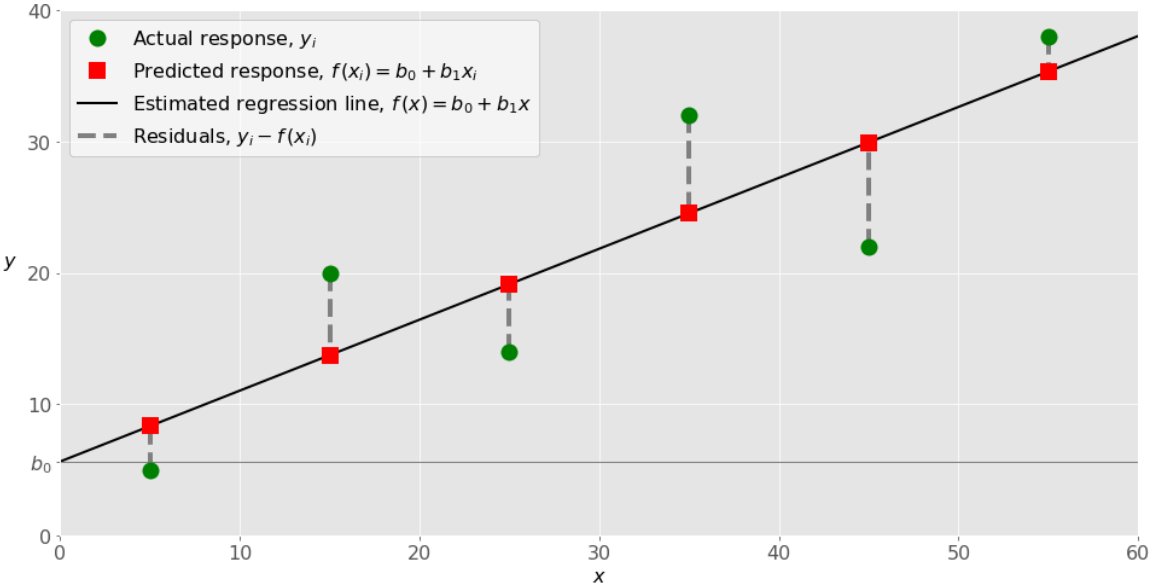
Dans le cadre d'un modèle linéaire simple, on peut représenter graphiquement la relation entre x et y à travers un **nuage de points**.
L'estimation du modèle linéaire permet de tracer la droite de régression d'équation **y =** $b_0 + b_1 x$

Exemple :



La régression linéaire simple ou à une variable est le cas le plus simple de régression linéaire, car elle a une seule variable indépendante, $\mathbf{x} = x$.

La figure suivante illustre une régression linéaire simple :



Lors de la mise en œuvre d'une régression linéaire simple ,on commence généralement avec un ensemble donné de paires entrée-sortie (x-y).
Ces paires sont vos observations, représentées par des cercles verts sur la figure.
La fonction de régression estimée, représentée par la ligne noire, a l'équation $f(x) = b_0 + b_1 x$. Votre objectif est de calculer les valeurs optimales des poids

prédits b_0 et b_1 qui minimisent le SSR et déterminent la fonction de régression estimée.
La valeur de b_0, également appelée interception, indique le point où la ligne de régression estimée croise l'axe y.
Les réponses prédites, représentées par des carrés rouges, sont les points sur la ligne de régression qui correspondent aux valeurs d'entrée.
Les lignes grises verticales en pointillés représentent les résidus, qui peuvent être calculés comme $y_i - f(\mathbf{x}_i) = y_i - b_0 - b_1 x_i$ pour $i = 1, \ldots, n$.
Ce sont les distances entre les cercles verts et les carrés rouges.
Lorsque vous implémentez la régression linéaire, vous essayez en fait de minimiser ces distances et de rendre les carrés rouges aussi proches que possible des cercles verts prédéfinis.

III. Programmation :

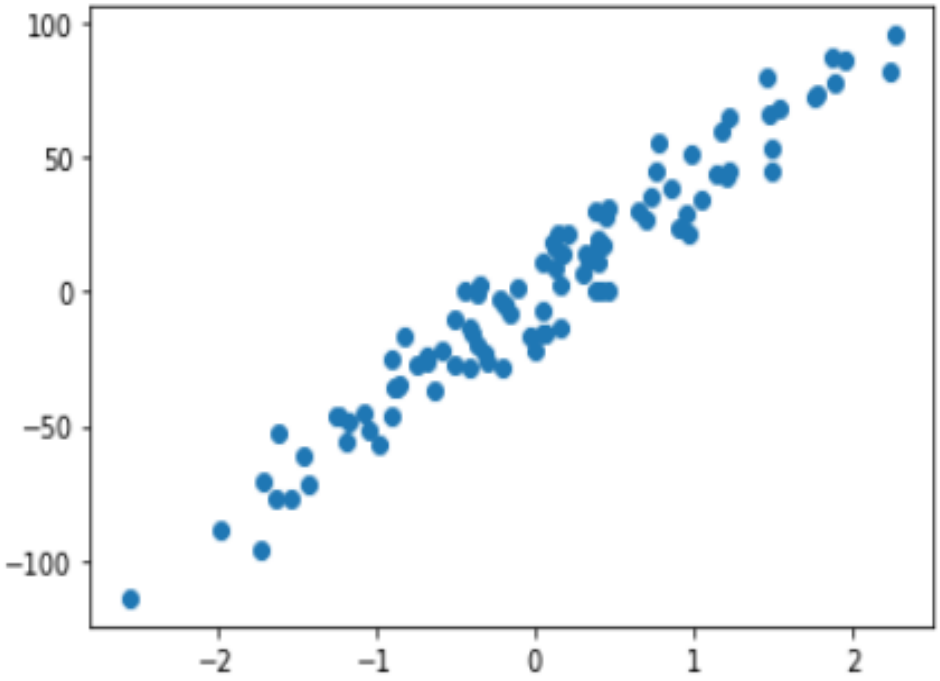
Importing the needed libraries :

```
import matplotlib.pyplot as plt
import random
import numpy as np
from sklearn import linear_model
from sklearn.datasets import make_regression
from sklearn import metrics
```

Generating a random regression with 100 samples , 100 features and with noise=10 :

```
X, Y = make_regression(n_samples=100, n_features=1, noise=10)
plt.scatter(X, Y)
```

```
<matplotlib.collections.PathCollection at 0x7f2f293a6e10>
```



Spliting data into training and testing sets :

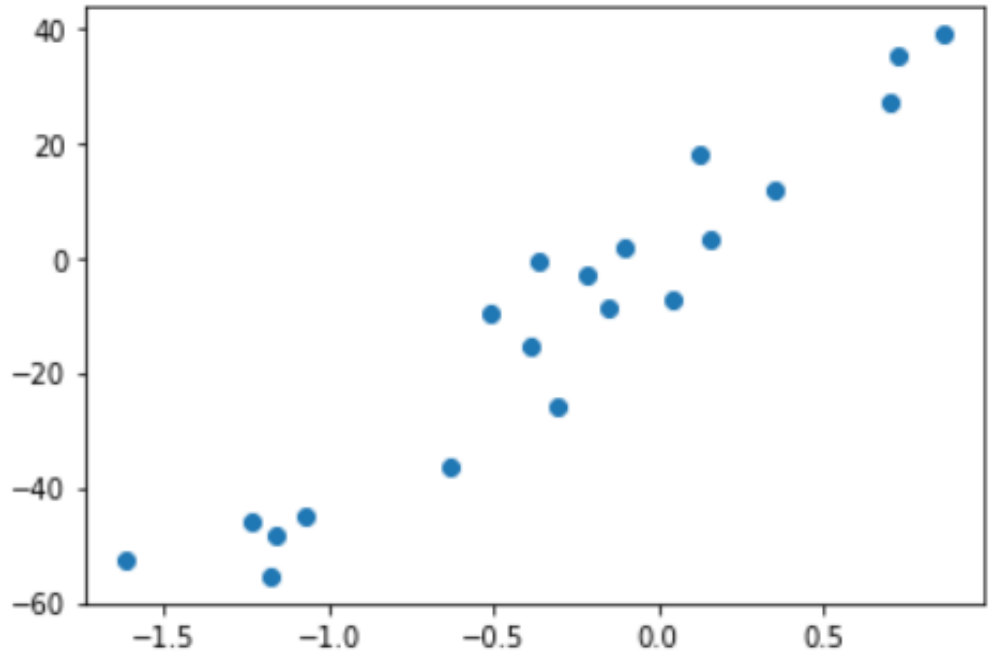
```
# Split the data into training/testing sets
X_train = X[:-20]
X_test = X[-20:]
```

```
# Split the targets into training/testing sets
Y_train = Y[:-20]
Y_test = Y[-20:]
```

Visualizing the testing sets :

```
#Visualizing the testing sets
plt.scatter(X_test, Y_test)
```

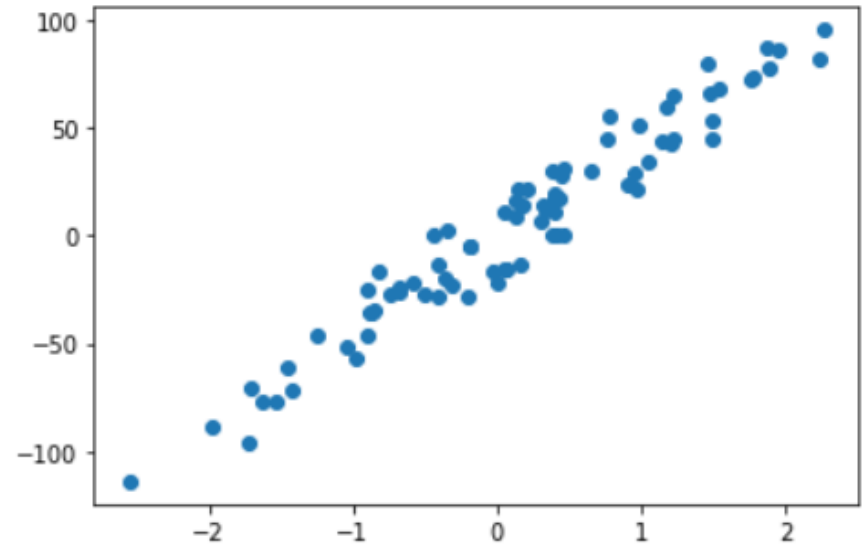
```
<matplotlib.collections.PathCollection at 0x7f2f2938b0d0>
```



Visualizing the training sets :

```
#Visualizing the training sets
plt.scatter(X_train, Y_train)
```

```
<matplotlib.collections.PathCollection at 0x7f2f292f4ed0>
```



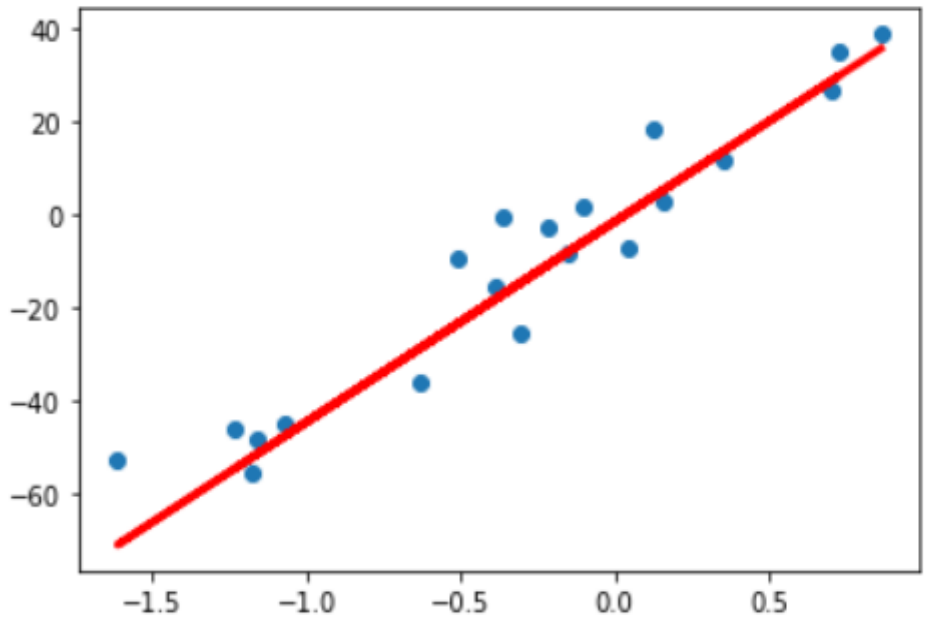
Training and predicting:

```
# Create linear regression object
regr = linear_model.LinearRegression()
# Train the model using the training sets
regr.fit(X_train, Y_train)
# Make predictions using the testing set
Y_pred = regr.predict(X_test)
```

Visualizing the linear regression on the testing set:

```
# Plot outputs
plt.scatter(X_test, Y_test)
plt.plot(X_test, Y_pred, color="red", linewidth=3)


plt.show()
```



Evaluating :

```
mse=mean_squared_error(Y_test, Y_pred)
print("MSE =",mse)
```

```
MSE = 76.9101389650662
```

```
acc=metrics.r2_score(Y_test,Y_pred)
print("Accuracy =",acc)
```

```
Accuracy = 0.9059143507912354
```

IV. Conclusion :

Grace à cet atelier, on s'est familiariser avec la méthode de la régression linéaire en l'utilisant dans un cas simple pratique qui nous a permet d'acquérir de nouvelles techniques.