Université Sultan Moulay Slimane Faculté Polydisciplinaire Béni Mellal
Département INFORMATIQUE (MIP)

Filière : Science de données et sécurité des systèmes d'information
A.U : 2023-2024

Sujet

Compte Rendu de TP01 – Apprentissage Automatique

Présenté Par :
MAFTOUH Omar

Encadré Par :
A. MAARIR

Introduction :

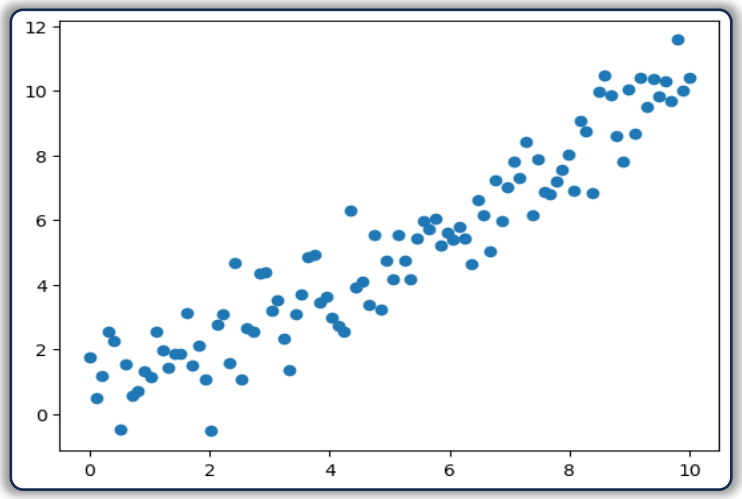
Le TP01 sur la régression linéaire explore trois méthodes fondamentales pour modéliser les relations entre variables : la méthode des moindres carrés, la méthode des moindres carrés ordinaires (OLS) et la descente de gradient. L'objectif principal de ce TP est de comprendre ces méthodes et de les appliquer à des données réelles pour prédire des valeurs cibles.

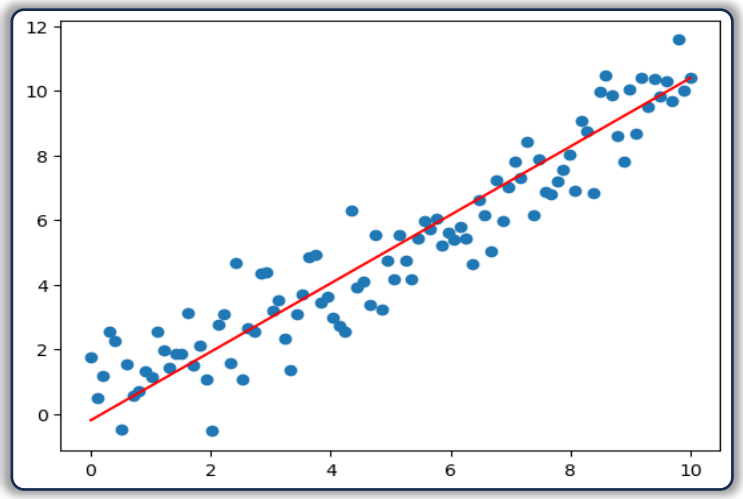
1. Méthode des moindres carrés :

La méthode des moindres carrés est l'une des approches les plus couramment utilisées pour estimer les paramètres d'un modèle linéaire. Elle consiste à minimiser la somme des carrés des écarts entre les valeurs observées et les valeurs prédites par le modèle.

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
n_observation = 100
X = np.linspace(0 , 10 , n_observation).reshape((n_observation , 1))
Y = X + np.random.randn(n_observation , 1)
plt.scatter(X , Y)
plt.show()
n = len(X)
x_moy = 0
y_moy = 0
for i in range(n) :
    x_moy += X[i]
    y_moy = x_moy + Y[i]
x_moy = x_moy / n
y_moy = y_moy / n
num = 0
den = 0
for i in range(n) :
    num = (X[i] - x_moy ) * (Y[i] - y_moy)
    den = (X[i] - x_moy )**2
thata1 = num / den
thata0 = y_moy - thata1 * x_moy
print(thata1 , thata0) # [1.05959389] [-0.19394958]

prediction = thata1 * X + thata0
plt.scatter(X , Y)
plt.plot(X , prediction , color = 'red')
```

Résultat du programme :



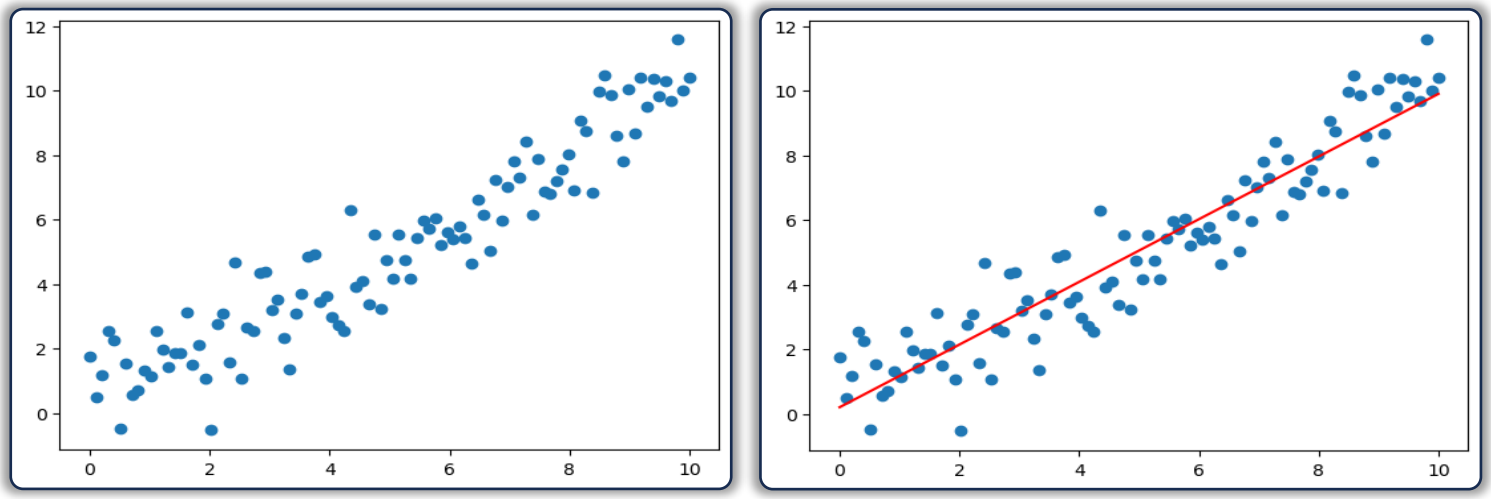


2. Méthode des moindres carrés ordinaires (OLS) :

L'OLS est une extension de la méthode des moindres carrés qui permet de gérer des situations plus complexes, notamment lorsque les données sont bruitées ou qu'il y a plusieurs variables explicatives.

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(0)
n_observation = 100
X = np.linspace(0, 10, n_observation).reshape((n_observation, 1))
Y = X + np.random.randn(n_observation, 1)
plt.scatter(X, Y)
plt.show()

x1 = np.ones(n_observation)
x = np.c_[x1, X]

def regression_lineare_ols(X, y):
    theta = np.linalg.inv(X.T @ X) @ X.T @ y
    return theta

theta = regression_lineare_ols(x, Y)

prediction = x @ theta

plt.scatter(x[:, 1], Y)
plt.plot(x[:, 1], prediction, color='red')
plt.show()
```

Résultat du programme :

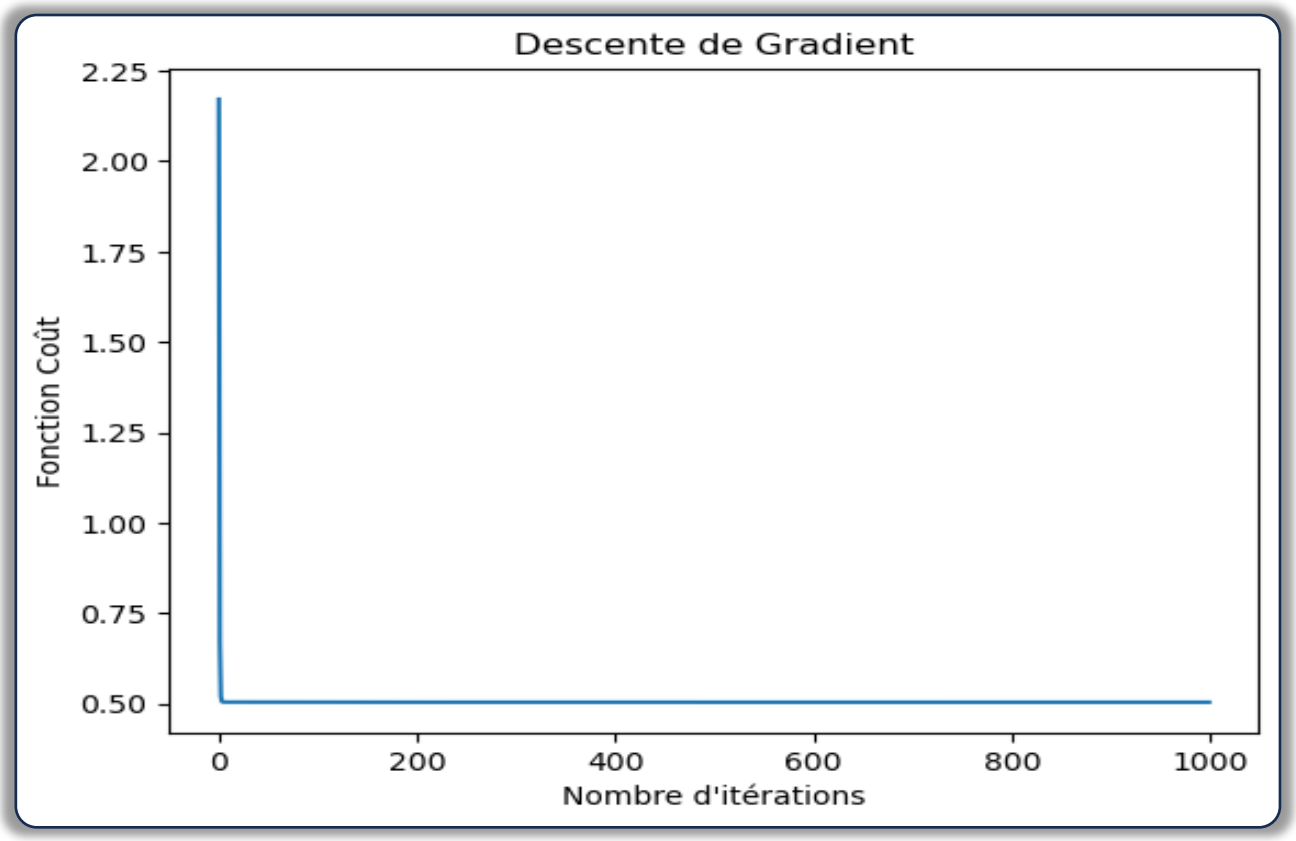


3. La descente de gradient :

La descente de gradient est une méthode d'optimisation couramment utilisée pour trouver les paramètres d'un modèle qui minimisent une fonction de coût. Dans le contexte de la régression linéaire, la descente de gradient peut être utilisée pour trouver les coefficients optimaux en ajustant itérativement les paramètres du modèle.

```
def fonction_cout(X, y, theta):
    n = len(y)
    predictions = X.dot(theta)
    cout = (1 / (2 * n)) * np.sum(np.square(predictions - y))
    return cout


def gradient(X, y, theta):
    m = len(y)
    return (1 / m) * X.T.dot(X.dot(theta) - y)


def descente_gradient(X, y, theta, alpha, n_iterations):
    j_historique = np.zeros((n_iterations, 1))
    for i in range(n_iterations):
        gradient_theta = gradient(X, y, theta)
        theta = theta - alpha * gradient_theta
        j_historique[i] = fonction_cout(X, y, theta)
    return theta, j_historique


np.random.seed(0)
n_observation = 100
X = np.linspace(0, 10, n_observation).reshape((n_observation, 1))
y = X + np.random.randn(n_observation, 1)

X_biais = np.c_[np.ones((n_observation, 1)), X]

alpha = 0.02
n_iterations = 1000

theta_init = np.array([[0], [0]])

theta_final, j_historique = descente_gradient(X_biais, y, theta_init, alpha,
n_iterations)

print("Paramètres finaux (theta):", theta_final) # [[0.20804923][0.9703308 ]]
plt.plot(range(n_iterations), j_historique,)
plt.xlabel('Nombre d\'itérations')
plt.ylabel('Fonction Coût')
plt.title('Descente de Gradient')
plt.show()
```

Résultat du programme :



Conclusion :

Ce TP01 nous a permis de nous familiariser avec trois méthodes de régression linéaire : la méthode des moindres carrés, la méthode OLS et la descente de gradient. Chacune de ces méthodes a ses avantages et ses inconvénients, et leur choix dépend des données et des objectifs spécifiques de modélisation. En comprenant ces méthodes et en les appliquant à des données réelles, nous avons acquis une base solide pour explorer des modèles plus complexes et des techniques d'apprentissage automatique avancées.