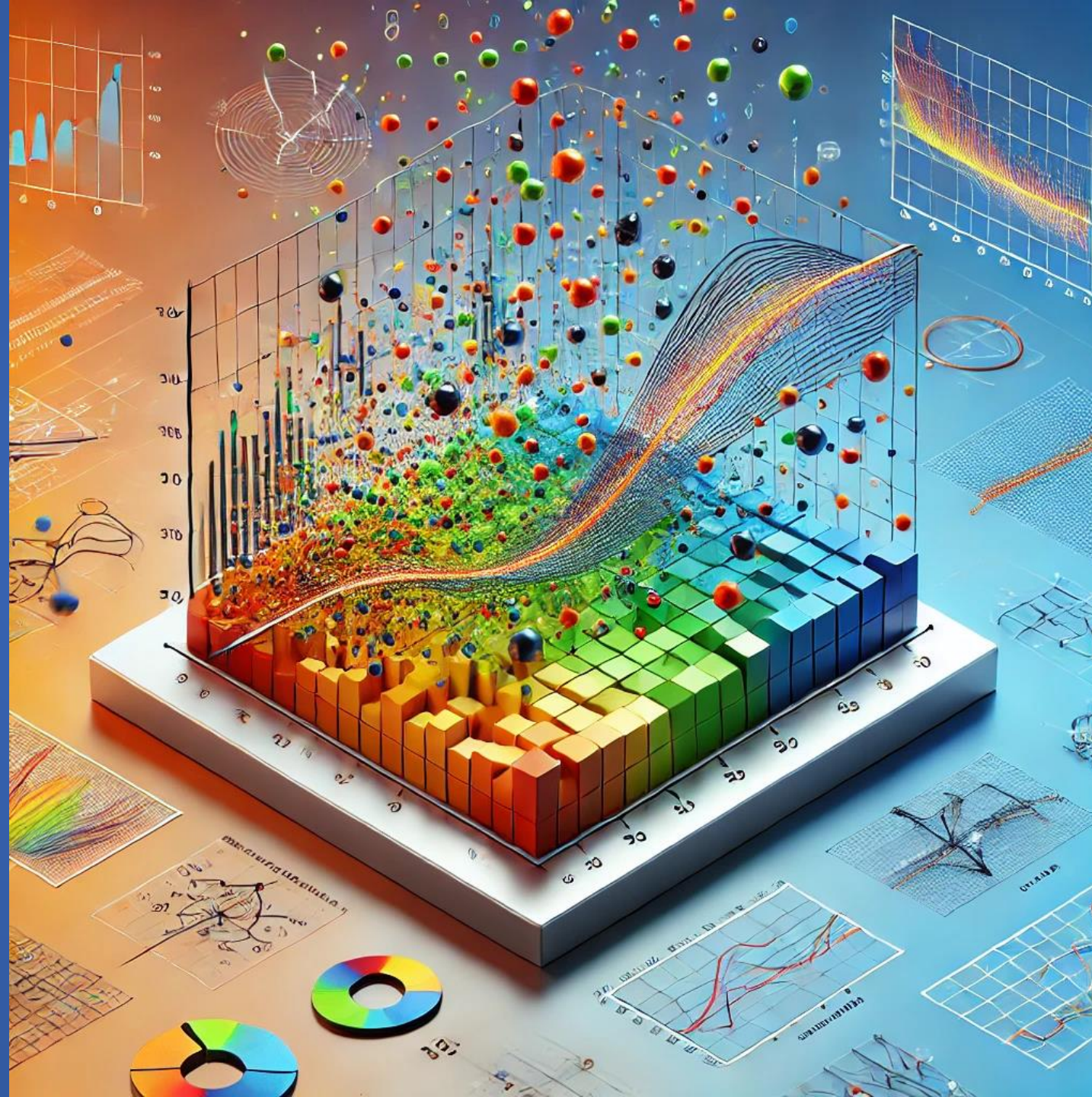
Chapitre 5

Regression linéaire

Pré[paré-senté] par :
Dr. Bilal Dendani



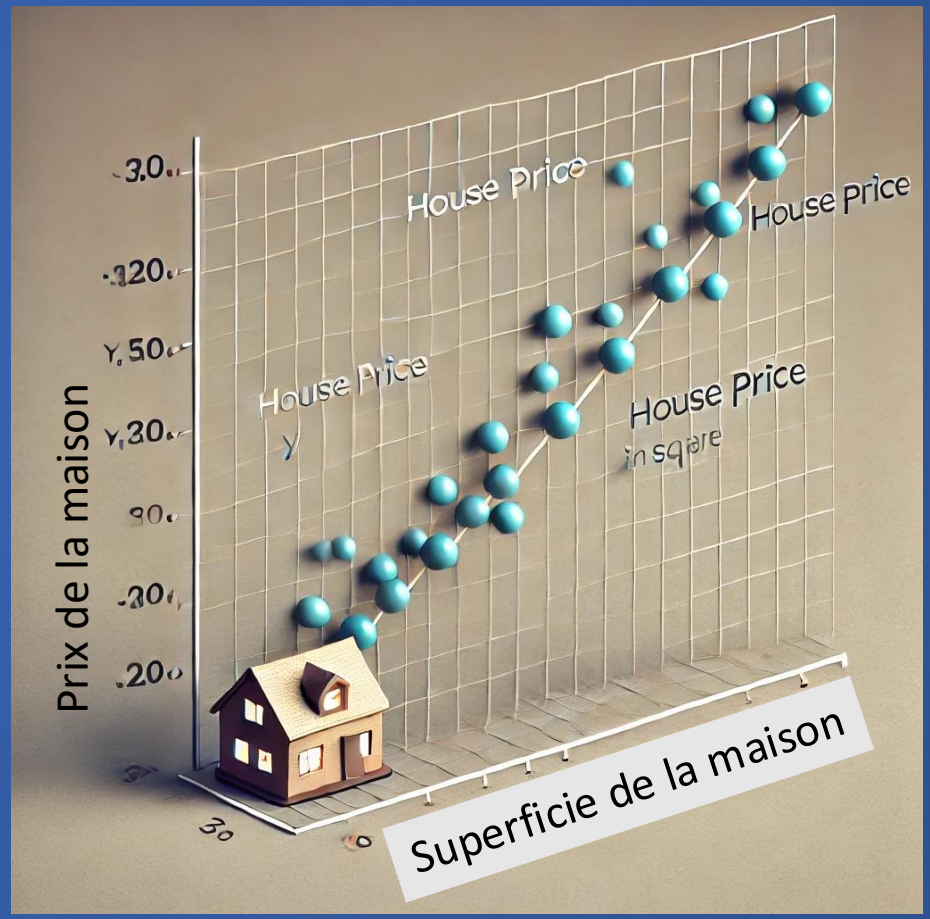
جامعة باجي مختار - عنابة
BADJI MOKHTAR - ANNABA UNIVERSITY
Dr. DENDANI Bilal

Chapitre 5 : Regression linéaire

- Régression Linéaire Simple :
  - Modèle de régression linéaire simple, estimation des paramètres.
  - Interprétation des résultats, erreurs et diagnostics.
- Régression Linéaire Multiple :
  - Extension aux modèles à plusieurs variables explicatives.
  - Sélection de modèles et régularisation (Ridge, Lasso).

Introduction à la Régression Linéaire Simple

- La régression linéaire simple est un modèle statistique permettant de **prédire une variable dépendante (Y)** à partir d'une **variable indépendante (X)**.
- C'est un algorithme de machine learning fondamentale de type supervised. Elle permet de trouver la relation entre la variable dépendante et les variables indépendantes.



Objectif géneral de la régression linéaire

- **Forme générale du modèle :**

$$Y = \beta 0 + \beta 1X + \varepsilon$$ Où :

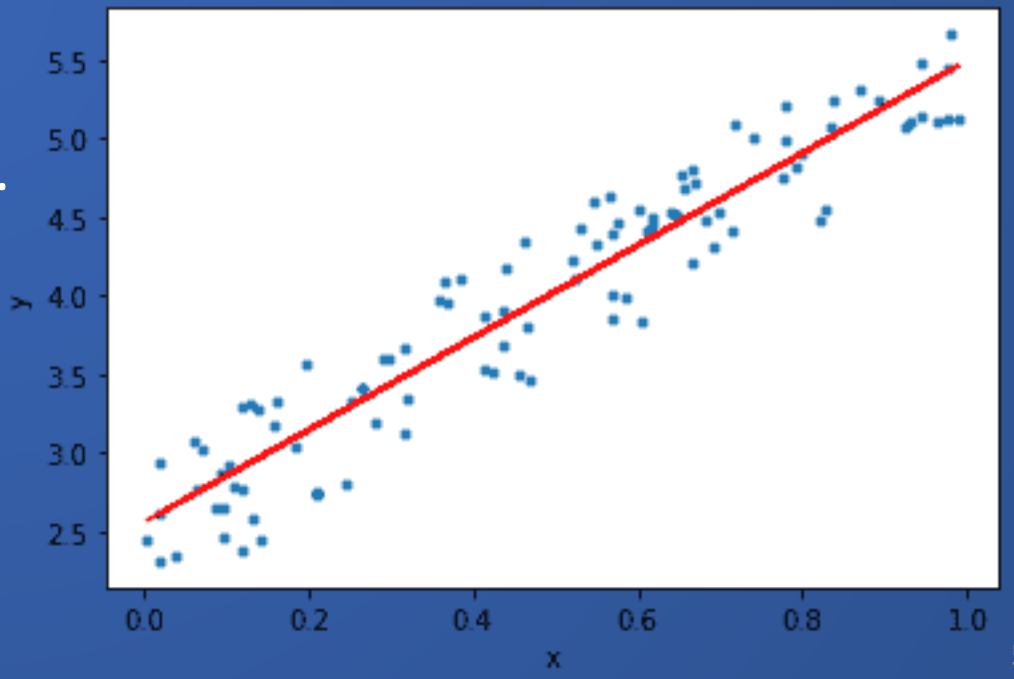
  - Y : Variable cible (dépendante).
  - X : Variable explicative (indépendante).
  - β0 : Intercept (constante).
  - β1: Pente (coefficient de X).
  - ε var epsilon ε : Erreur aléatoire (non expliquée par le modèle).

- L’objectif de la régression linéaire est :
- Estimer les paramètres β0 et β1pour ajuster au mieux la droite de régression.
- Prédire de nouvelles valeurs pour Y en fonction des valeurs de X.
- Interpréter les résultats pour comprendre l'impact de X sur Y.
- Trouver une relation entre une ou plusieurs entités (variables indépendantes) et une variable cible continue (variable dépendante).

Représentation Graphique de la Régression Linéaire

- **Visualisation** de la relation linéaire entre X et Y sur un graphique.
- La droite de régression est la **meilleure approximation** linéaire des points observés.

Axe horizontal (X) : Variable indépendante.
Axe vertical (Y) : Variable dépendante.
Droite de régression : $Y = \beta 0 + \beta 1X$



Méthode des Moindres Carrés pour Estimer les Paramètres

Objectif : Minimiser les erreurs

- On cherche à minimiser la **somme des carrés des écarts (erreurs)** entre les valeurs observées Y_i et les valeurs prédites $\hat{Y}_i$.

$$\text{Erreur totale} = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

Où $\hat{Y}_i = \beta_0 + \beta_1 X_i$.

Formules pour β_0 et β_1 :

- La pente (β_1) :

$$\beta_1 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n} (X_i - \bar{X})^2}$$

- L'ordonnée à l'origine (β_0) :

$$\beta_0 = \bar{Y} - \beta_1 \bar{X}$$

Où $\bar{X}$ et $\bar{Y}$ sont les moyennes de X et Y.

Exemple : Prédire la taille d'une personne en fonction de son poids

Exemple : Prédire la taille d'une personne en fonction de son poids

Poids (X)	50	60	70	80
Taille (Y)	150	160	170	180

Étapes de calcul :

1. Calculer les moyennes $\bar{X}$ et $\bar{Y}$.
   - $\bar{X} = (50 + 60 + 70 + 80)/4 = 65.$
   - $\bar{Y} = (150 + 160 + 170 + 180)/4 = 165.$
2. Calculer β_1 :

$$\beta_1 = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sum(X_i - \bar{X})^2}$$

- Numérateur : $(50 - 65)(150 - 165) + (60 - 65)(160 - 165) + ...$
- Dénominateur : $(50 - 65)^2 + (60 - 65)^2 + ...$

3. Calculer β_0 :

$$\beta_0 = \bar{Y} - \beta_1 \bar{X}$$

4. Écrire l'équation de la droite de régression :
   $\hat{Y} = \beta_0 + \beta_1 X.$

Exemple pratique sur la régression linéaire : Prédire la taille en fonction du poids

- 1. Installation de la bibliothèque sckit-learn qui offre l'algorithm de la régression linéaire.

Installer la bibliothèque sckit-learn

```
!pip install -U scikit-learn
```

```
Requirement already satisfied: scikit-learn in /opt/miniconda3/lib/python3.12/site-packages (1.5.2)
Collecting scikit-learn
  Downloading scikit_learn-1.6.0-cp312-cp312-macosx_10_13_x86_64.whl.metadata (31 kB)
Requirement already satisfied: numpy>=1.19.5 in /opt/miniconda3/lib/python3.12/site-packages (from scikit-learn) (1.26.4)
Requirement already satisfied: scipy>=1.6.0 in /opt/miniconda3/lib/python3.12/site-packages (from scikit-learn) (1.14.1)
Requirement already satisfied: joblib>=1.2.0 in /opt/miniconda3/lib/python3.12/site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /opt/miniconda3/lib/python3.12/site-packages (from scikit-learn) (3.5.0)
Downloading scikit_learn-1.6.0-cp312-cp312-macosx_10_13_x86_64.whl (12.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.1/12.1 MB 1.6 MB/s eta 0:00:0000:0100:01m
Installing collected packages: scikit-learn
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.5.2
    Uninstalling scikit-learn-1.5.2:
      Successfully uninstalled scikit-learn-1.5.2
Successfully installed scikit-learn-1.6.0
```

Exemple pratique sur la régression linéaire : Prédire la taille en fonction du poids

- 2. Importation de bibliothèques nécessaires et fournir les données pour l'ntrainement

Importation de bibliothèqies nécessaires

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

Données

- X : Un tableau contenant les poids.
- Y : Un tableau contenant les tailles.
- reshape(-1, 1) : Reformate X en une matrice colonne pour qu'il soit compatible avec LinearRegression.

```
X = np.array([50, 60, 70, 80]).reshape(-1, 1)
Y = np.array([150, 160, 170, 180])
print(X)
print(Y)
```

```
[[50]
 [60]
 [70]
 [80]]
[150 160 170 180]
```

Exemple pratique sur la régression linéaire : Prédire la taille en fonction du poids

- 3. Création et entrainement de modèle, extraction de coefficients

Création et Entraînement du Modèle

- model : Crée une instance de LinearRegression.
- fit(X, Y) : Entraîne le modèle sur les données X et Y.

```
model = LinearRegression()
model.fit(X, Y)
```

```
▾ LinearRegression
LinearRegression()
```

Extraction des Coefficients

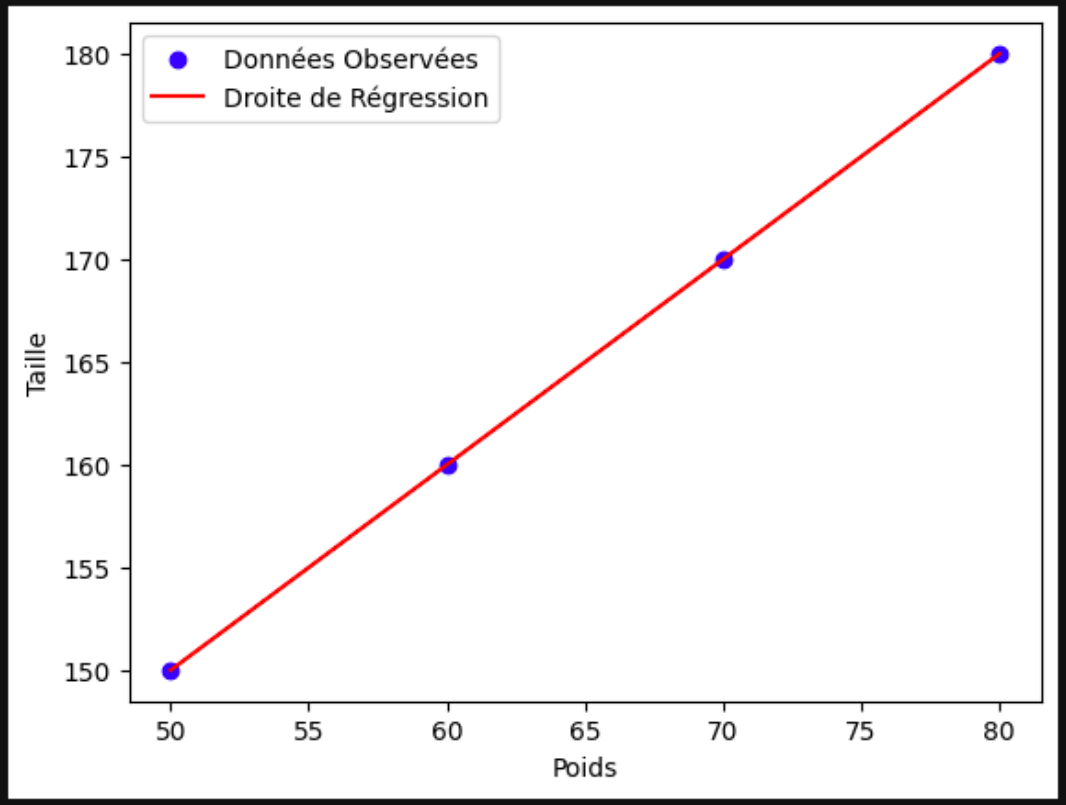
- intercept_ : L'ordonnée à l'origine (intercept) du modèle.
- coef_[0] : La pente (coefficient) du modèle.

```
beta_0 = model.intercept_
beta_1 = model.coef_[0]
print(f"Intercept (beta_0): {beta_0}")
print(f"Pente (beta_1): {beta_1}")
```

```
Intercept (beta_0): 100.0
Pente (beta_1): 1.0
```

4. Exemple pratique sur la régression linéaire : Prédiction et affichage de modèle

```
Y_pred = model.predict(X)
plt.scatter(X, Y, color="blue", label="Données Observées")
plt.plot(X, Y_pred, color="red", label="Droite de Régression")
plt.legend()
plt.xlabel("Poids")
plt.ylabel("Taille")
plt.show()
```




```
predicted_model = model.predict(X)
predicted_model
```

```
array([150., 160., 170., 180.])
```