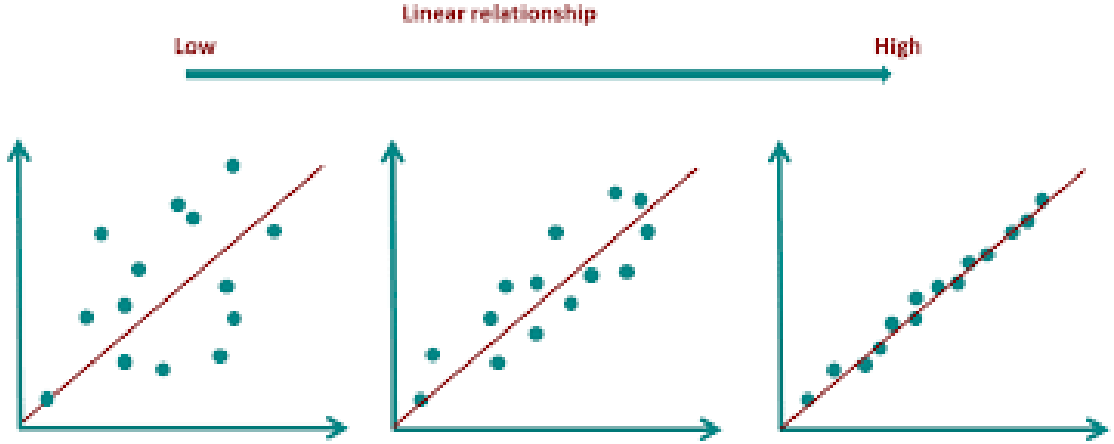
# TP4

# Régression linéaire




Département : Informatique
Institut : Mathématique et Informatique
1 Master I2A + STIC

2022/2023

# Plan

**TP 4:** Data mining (RL) avec R

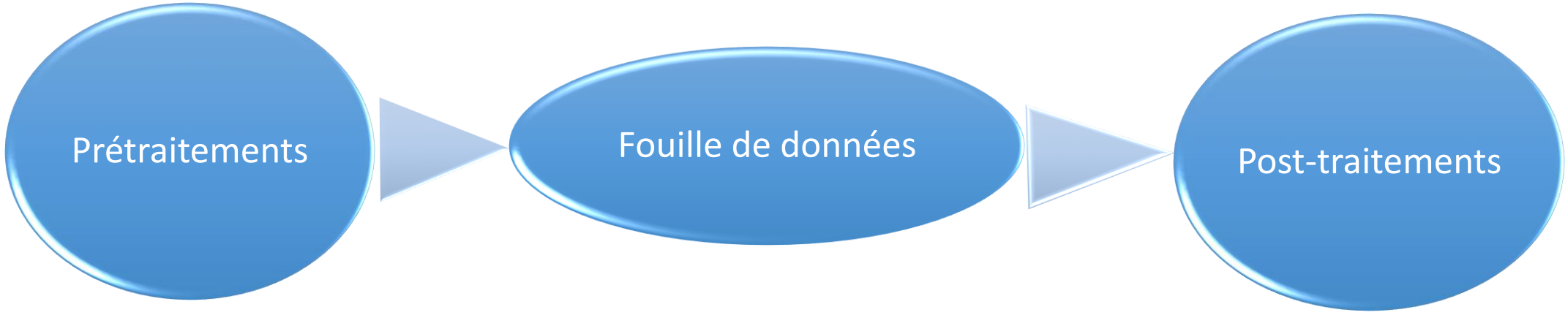
1. **Objectif**
2. **Processus du data mining**
3. **Prétraitement de données**
4. **Régression linaire ( data mining )**

# 1. Objectif

L’objectif de ce TP est d’initier les étudiants aux étapes essentielles d’un projet de data mining. Il vise notamment à montrer comment préparer les données et à appliquer la régression linéaire à travers une étude de cas

# 2. Processus d'Extraction des connaissances

- Les étapes précédentes d'un processus du data mining peuvent se résumer en trois grandes phases:




Le prétraitement vise à transformer les données brutes et souvent "sales" en un format propre, cohérent et adapté à l'analyse par les algorithmes.

C'est l'étape où les algorithmes du data mining sont appliqués aux données prétraitées pour identifier des modèles cachés, des tendances et des relations

Cette étape consiste à donner un sens aux modèles découverts et à les présenter sous une forme compréhensible pour la prise de décision.

# 2. Processus d'Extraction des connaissances

## Prétraitement des Données (Data Preprocessing)

- *Nettoyage (Cleaning) :* Gestion des *valeurs manquantes* (imputation par la moyenne, la médiane, etc., ou suppression), détection et correction des erreurs ou des *valeurs aberrantes* (outliers) qui pourraient biaiser l'analyse.
- Intégration et Transformation **:** Consolidation des données provenant de sources hétérogènes. La transformation inclut des opérations comme la *normalisation* (mise à l'échelle des variables numériques) ou *l'agrégation* (résumer des données à un niveau supérieur).
- *Réduction (Reduction) et Sélection :* Réduction du nombre de variables (sélection d'attributs pertinents ou utilisation de techniques de réduction de dimensionnalité comme l'ACP) pour améliorer la performance des algorithmes et faciliter l'interprétation.

# 2. Processus d'Extraction des connaissances

## Data Mining (Exploration/Modélisation)

- Régression **:** Prédire une *valeur numérique* (ex: prédire le montant des ventes futures ou le prix d'une action).
  *Techniques :* régression linaire
- Classement : Prédire l'appartenance d'un élément à une catégorie prédéfinie (ex: prédire si un client va *quitter* (oui/non) ou *acheter* (oui/non)).
  *Techniques :* Arbres de décision, Réseaux de neurones, Machines à Vecteurs de Support.
- Clustering (Segmentation) **:** Regrouper des données similaires en clusters ou segments. C'est une tâche descriptive qui ne nécessite pas de catégories prédéfinies (apprentissage non supervisé).
  *Techniques :* K-Means, Clustering hiérarchique.
- …

# 2. Processus d'Extraction des connaissances

## Poste traitement (Interprétation et Visualisation)

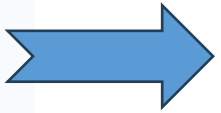
- *Évaluation du Modèle :* Mesurer la validité et la performance du modèle généré (ex: est-ce que le modèle de classification prédit correctement l'appartenance à la catégorie ?).
- *Interprétation des Résultats :* Examiner les modèles pour s'assurer qu'ils sont non seulement statistiquement corrects, mais aussi pertinents et exploitables du point de vue métier.
- *Visualisation (Data Visualization) :* Utilisation de graphiques, tableaux de bord et autres représentations visuelles pour simplifier et communiquer clairement les informations complexes extraites. Une bonne visualisation transforme les résultats bruts en insights (informations exploitables) pour les décideurs.

# 3. Prétraitement de Données

## Nettoyage de données

**Données Brutes (Avant Nettoyage)**

| ID Client | Âge | Ville | Montant d'Achat (DZD) |
|---|---|---|---|
| 101 | 28 | Alger | 1500 |
| 102 | 450 | Oran | 800 |
| 103 | *NULL* | Sétif | 2200 |
| 104 | 35 | alger | -500 |
| 105 | 51 | Alger | 1200 |



**Données Nettoyées (Après Nettoyage)**

| ID Client | Âge | Ville | Montant d'Achat (DZD) |
|---|---|---|---|
| 101 | 28 | Alger | 1500 |
| 102 | **35** | Oran | 800 |
| 103 | **35** | Sétif | 2200 |
| 104 | 35 | **Alger** | **0** |
| 105 | 51 | Alger | 1200 |

**Imputation**
**Médiane**

**Standardisation**

# 3. Prétraitement de Données

## Normalisation de données

### 1. Normalisation Min-Max (Mise à l'échelle linéaire)

- Cette méthode met les données à l'échelle dans une plage spécifique, le plus souvent entre 0 et 1 (données dans une plage bornée et fixe (comme [0, 1] ou [-1, 1]).

$$valeur_{norm} = \frac{valeur - Min_{intv}}{Max_{intv} - Min_{intv}}$$

Où :

- $valeur$ : la valeur à normaliser.
- $valeur_{norm}$ : la valeur après la normalisation.
- $Min_{intv}$ : la valeur la plus petite de l'intervalle à qui $valeur$ appartient.
- $Max_{intv}$ : la valeur la plus grande de l'intervalle à qui $valeur$ appartient.

# 3. Prétraitement de Données

## Normalisation de données

1. Normalisation Min-Max (Mise à l'échelle linéaire)

- Par exemple: Imaginez que vous avez les notes d'un groupe d'étudiants sur 20. Les notes sont : **[8, 12, 15, 20, 10]**.
- Donc : Min = 8, Max = 20, Max-Min= 12

| Valeur d'origine (x) | Calcul (x−8) / 12 | Valeur Normalisée (x') |
|---|---|---|
| 8 | 0/12 | 0.00 |
| 10 | 2/12 | 0.17 |
| 12 | 4/12 | 0.33 |
| 15 | 7/12 | 0.58 |
| 20 | 12/12 | 1.00 |

# 3. Prétraitement de Données

## Normalisation de données

2. Normalisation par Score Z (Standardisation)

- Cette méthode transforme les données pour qu'elles aient une moyenne $\mu$ de 0 et un écart-type $\sigma$ de 1. On parle de Standardisation plutôt que de normalisation.

$$X_{standardisé} = \frac{X - \mu}{\sigma}$$

où :

- $X$ est la valeur originale.
- $\mu$ est la moyenne des valeurs de la colonne.
- $\sigma$ est l'écart-type des valeurs de la colonne.

# 3. Prétraitement de Données

## Normalisation de données

- Normalisation par Score Z  avec R
- Exemple: Nous allons créer un petit jeu de données de taille (taille_cm) et appliquer la formule;
  taille_cm = (165, 178, 155, 185, 172)

| Taille (x) | Calcul (x−171)/10,37 | Score Z | Interprétation |
|---|---|---|---|
| 165 | (165−171)/10,37 | -0,58 | En dessous de la moyenne |
| 178 | (178−171)/10,37 | +0,67 | Au-dessus de la moyenne |
| 155 | (155−171)/10,37 | -1,54 | Très en dessous de la moyenne |
| 185 | (185−171)/10,37 | +1,35 | Très au-dessus de la moyenne |
| 172 | (172−171)/10,37 | +0,10 | Très proche de la moyenne |

# 3. Prétraitement de Données

## Normalisation de données

- Normalisation par Score Z  avec R
- Exemple: Nous allons créer un petit jeu de données de taille (taille_cm) et appliquer la formule

```
# Création du vecteur de données brutes
   taille_cm = c(165, 178, 155, 185, 172)
   print("Données originales :")
   print(taille_cm)
# Calcul de la moyenne) et de l'écart-type
    mu = mean(taille_cm)
   sigma = sd(taille_cm)
# Application de la formule du Score Z
  taille_standardisee <- (taille_cm - mu) / sigma
   print("Données standardisées (Score Z) :")
   print(taille_standardisee)
```

```
# Utilisation de la fonction R intégrée
pour la standardisation

taille_standard_auto = scale(taille_cm)
print("Résultat avec la fonction scale() :")
 print(taille_standard_auto)
```

# 3. Prétraitement de Données

## Normalisation de données

### 3. Mise à l'échelle Robuste (Robust Scaling)

Cette technique est une alternative à la standardisation qui utilise la médiane et l'intervalle interquartile (IQR), ce qui la rend très résistante aux valeurs aberrantes.

$$X_{robuste} = \frac{X - \text{Médiane}}{\text{IQR}}$$

où :

- $\text{Médiane}$ est la 50ème percentile des valeurs de la colonne.
- $\text{IQR}$ (Intervalle Interquartile) est la différence entre le 75ème percentile ($Q3$) et le 25ème percentile ($Q1$) (c'est-à-dire $Q3 - Q1$).

# 3. Prétraitement de Données

## Normalisation de données

### 4. Label Encoder

Le Label Encoder est principalement utilisé pour convertir des étiquettes de catégories en nombres, une étape indispensable dans le prétraitement des données pour le Data mining.

```
# Création du vecteur de données catégorielles
    couleur = c("Rouge", "Bleu", "Vert", "Bleu", "Rouge", "Vert", "Bleu")
    print("Données originales (Couleur) :")
    print(couleur)
# Convertir le vecteur en facteur
    couleur_facteur = factor(couleur)
    print("Vecteur converti en Facteur :")
    print(couleur_facteur)
    print("Niveaux du Facteur (l'encodage) :")
    print(levels(couleur_facteur))
```

# 3. Prétraitement de Données

## Normalisation de données

### 4. Label Encoder

La fonction *as.numeric()* appliquée au facteur retourne les niveaux numériques attribués :

```
# Appliquer as.numeric() au facteur pour obtenir les labels encodés
couleur_encodee = as.numeric(couleur_facteur)
print("Données après Label Encoding (numérique) :")
print(couleur_encodee)
```

# 4. Régression linéaire

Les données ont généralement la forme suivante :

| Nbr d'observations | X : Variable explicative | Y : Variable à expliquer |
|---|---|---|
| 1 | $x_1$ | $y_1$ |
| . | . | . |
| . | . | . |
| . | . | . |
| i | $x_i$ | $y_i$ |
| . | . | . |
| . | . | . |
| n | $x_n$ | $y_n$ |
| Moyenne | $\overline{x}$ | $\overline{y}$ |
| Écart-type | $\delta_x$ | $\delta_y$ |

# 4. Régression linéaire

## Le Modèle

- **Hypothèse** : Supposons que les données proviennent d'un phénomène suivants le modèle :

$$Y = f\ (X) = \ \alpha + \ \beta\ X$$

- Ou $X = (x_1, x_2\ , \dots , x_n)$
- Cette équation est appelée *mdèle de régression linéaire simple*
- $\alpha,\ \ \beta$ Son appelées des *paramètres* ou des *poids* de modéle.
- La fonction $Y\ =\ f(x)$ est une droite. On appelle cette droite *la droite de regression*.

# 4. Régression linéaire

## Le modèle

- Pour déterminer le modèle : $Y = f\,(X) = \alpha + \beta\, X$
- Nous pouvons écrire:

$$y_i = \alpha + \beta x_i + \xi_i, \quad i = 1, \ldots, n$$

Ou :

- ✓ les $x_i$ sont des valeurs non aléatoires *connus.*
- ✓ Les paramètres $\alpha, \beta$ du modèle sont *inconus.*
- ✓ Les $\varepsilon_i$ sont les réalisation *inconus* d'une variable aléatoire.
- ✓ Les $y_i$ sont les observations *connus*.

- Les hypothèses à ce modèle sont les suivantes :

$$E(\xi_i) = 0, \quad var(\xi_i) = \sigma^2. \quad cov(\xi_i, \xi_j) = 0, \quad \forall i \neq j.$$

# 4. Régression linéaire

## Le modèle

- Sous R on utilise la fonction ***lm(),*** pour faire la régression.
- La syntaxe générale est la suivante :

$$R = lm(variableexpliquer\ variable(s)explicative(s), data = ...)$$

- Dans le cas d'une *régression linéaire simple* on écrit :

$$R = lm\ (Y \quad X,\ data = ,,,)$$
$$Summary\ (\ R)$$

- Dans le cas d'une *régression linéaire multiple* :

$$R = lm\ (Y \quad X + Z, \qquad data = ,,,)$$
$$Summary\ (\ R)$$

# 4. Régression linéaire

## Étapes

1. Créer ou importer un jeu de données
2. préparation du jeu de données
3. diviser le jeu de données (donnée d'entrainement+ données de teste)
4. Effectuer une régression linéaire avec la fonction lm()
5. Analyser les résultats

# 4. Régression linéaire

## Exemple illustratif

- On veut prédire une variable **y** (par exemple le poids) en fonction d'une variable **x** (par exemple la taille):

**Création des données :**

Supposons que nous avons les données suivantes :

```
taille <- c(150, 160, 165, 170, 175, 180, 185)    # Variable explicative x
                                                   (en cm)
poids <- c(50, 55, 58, 63, 67, 72, 78)            # Variable à expliquer y
                                                     (en kg)
```

Affichage des données:

```
data <- data.frame(Taille = taille, Poids = poids)
print("Données :")
print(data)
```

# 4. Régression linéaire

## Exemple illustratif

**Régression linéaire simple:**

$Y = \beta0 + \beta1 * X + \varepsilon$ β0: L'intercept β1: La pente

Poids = β0 + β1 * Taille + ε

```
model <- lm(Poids ~ Taille, data = data)
```

- Pour résumer les résultats du modèle:
  ```
  summary(model)
  ```
- Lorsque vous lancez cette commande, R affiche:
  **1.(Intercept)** : C'est la constante (β0).
  **2.Taille** : C'est le coefficient (β1) associé à la variable X.
  **3.Pr(>|t|)** (p-value) : Si cette valeur est inférieure à **0,05**, cela signifie que la taille a une influence statistiquement significative sur le poids. Dans votre exemple, la corrélation est très forte, donc la p-value sera très petite.

# 4. Régression linéaire

## Exemple illustratif

**Régression linéaire simple:**

$Y = \beta0 + \beta1 * X + \varepsilon$ β0: L'intercept β1: La pente

Poids = β0 + β1 * Taille + ε

- Affichage de la droite de regression:

```
plot(data$Taille, data$Poids, main = "Régression Linéaire Simple", xlab = "Taille (cm)", ylab = "Poids (kg)", pch = 16, col = "blue")
```

- Ajout de la droite de regression:

```
abline(model, col = "red", lwd = 2)
```

# 4. Régression linéaire

## Exemple illustratif

**Analyser les résultats:**

- Prédiction d'une nouvelle valeur: prédire les poids des tailles 160, 170, et 190 :

```
nouvelle_taille <- data.frame(Taille = c(160, 170, 190))

predictions <- predict(model, nouvelle_taille)

print("Prédictions pour les nouvelles tailles :")
print(predictions)
```