TP2 : Collecte et préparation des données

Objectifs du TP

Dans ce TP, vous allez manipuler les concepts du **Chapitre 2: Collecte et préparation des données** :

- Méthodes d'échantillonnage
- Gestion des données manquantes
- Détection et traitement des valeurs aberrantes
- Normalisation et standardisation
- Vérification de la cohérence et des unités
- Transformation des données

---

1. Acquisition des données

- Ensembles de données existantes dans R (ex : `iris`, `mtcars`, etc.)

```
iris

mtcars
```

- Lecture à partir d'un fichier CSV ou Excel…

Vous pouvez télécharger le fichier `pacman.csv` ou obtenir son URL depuis ce lien.

```r
# Exemple de lecture d'un fichier CSV

# install.packages("tidyverse") # Si le package n'est pas déjà installé

library(readr)

url <- "https://...../pacman.csv"

pacman <- read_csv(url)
head(pacman)

# Ou si le fichier est local

# Supposons que le fichier se trouve sous le répertoire `data`
# dans le répertoire courant. Remplacez par le chemin correct.

pacman <- read_csv("data/pacman.csv")
head(pacman)
```

- Génération de données simulées

```r
set.seed(123) # Pour la reproductibilité
population <- data.frame(
  id = 1:1000, # Identifiant unique
  age = sample(18:70, 1000, replace = TRUE), # Âge entre 18 et 70 ans
  sexe = sample(c("H", "F"), 1000, replace = TRUE, prob = c(0.6, 0.4)), # H/F
  revenu = round(rnorm(1000, mean = 2000, sd = 500), 0) # Revenu mensuel
)
# Afficher les six premières lignes
head(population)
```

2. Méthodes d'échantillonnage

Exercice 1 : Échantillonnage aléatoire simple

Tirez un échantillon aléatoire de 50 individus et calculez la moyenne du revenu.

```r
# Tirer un échantillon aléatoire de 50 individus
echantillon_simple <- population[sample(1:nrow(population), 50), ]
# Calculer la moyenne du revenu et afficher la répartition par sexe
```

```r
mean(echantillon_simple$revenu)
# Afficher le nombre d'individus par sexe
table(echantillon_simple$sexe)
```

Exercice 2 : Échantillonnage stratifié

Faites un échantillon stratifié sur le sexe (H/F).

```r
library(dplyr)
# Sélectionner 25 hommes et 25 femmes
echantillon_stratifie <- population %>% group_by(sexe) %>% sample_n(25)
# Afficher le nombre d'individus par sexe
table(echantillon_stratifie$sexe)
```

Exercice 3 : Échantillonnage en grappes

```r
 # Ajouter une variable ville: ville1 ... ville20
population$ville <- sample(paste0("Ville", 1:20), 1000, replace = TRUE)
# Sélectionner 3 villes aléatoires
grappes <- sample(unique(population$ville), 3)
# Sélectionner tous les individus de ces villes
echantillon_grappes <- population %>% filter(ville %in% grappes)
# Afficher le nombre d'individus par ville
table(echantillon_grappes$ville)
```

---

3. Gestion des données manquantes

Exemple : Création d'un dataset avec valeurs manquantes

```r
set.seed(42)
donnees <- population
# Introduire des valeurs manquantes aléatoirement pour la variable revenu
donnees$revenu[sample(1:1000, 50)] <- NA
# Afficher les six premières lignes
head(donnees)
```

Exercice 4 : Identifier les valeurs manquantes

```r
# Nombre de valeurs manquantes
sum(is.na(donnees$revenu))
# Moyenne en ignorant les NA
mean(donnees$revenu, na.rm = TRUE)
```

Exercice 5 : Imputation des valeurs manquantes

- Supprimer les lignes avec valeurs manquantes
- Remplacer par la moyenne

```r
# Supprimer les lignes avec valeurs manquantes
donnees_sans_na <- na.omit(donnees)

# Remplacer les valeurs manquantes par la moyenne
donnees$revenu[is.na(donnees$revenu)] <- mean(donnees$revenu, na.rm = TRUE)
```

---

4. Détection des valeurs aberrantes

Exemple : Règle des écarts-types

```r
# Détection des outliers avec la règle des 3 écarts-types

# Variable à analyser
revenus <- population$revenu
# Calcul de la moyenne
moy <- mean(revenus)
# Calcul de l'écart-type
sd <- sd(revenus)
# Identifier les outliers
outliers_sd <- which(revenus > moy + 3*sd | revenus < moy - 3*sd)
# Afficher les outliers
revenus[outliers_sd]
```

Exemple : IQR (Interquartile Range)

```
# Détection des outliers avec la méthode IQR
Q1 <- quantile(revenus, 0.25) # Premier quartile
Q3 <- quantile(revenus, 0.75) # Troisième quartile
IQR <- Q3 - Q1 # Intervalle interquartile
# Définir les bornes pour les outliers
borne_inf <- Q1 - 1.5*IQR # Borne inférieure
borne_sup <- Q3 + 1.5*IQR # Borne supérieure
# Identifier les outliers
outliers_iqr <- which(revenus < borne_inf | revenus > borne_sup)
# Afficher les outliers
revenus[outliers_iqr]
```

Exercice 6 : Visualisation avec boxplot

```
# Visualiser les outliers avec un boxplot
boxplot(revenus, main="Détection des outliers", col="lightblue")
```

---

5. Normalisation et standardisation

Exemple : Normalisation Min-Max

```
# Normalisation Min-Max

# Valeur minimale
min_val <- min(population$revenu)
# Valeur maximale
max_val <- max(population$revenu)
# Normalisation
revenu_norm <- (population$revenu - min_val) / (max_val - min_val)
# Afficher les premières valeurs
head(revenu_norm)
```

Exemple : Standardisation (Z-score)

```r
# Standardisation (Z-score)

# Centrer et réduire
revenu_std <- scale(population$revenu)
# Afficher les premières valeurs
head(revenu_std)
```

Exercice 7 : Comparer les distributions

```r
# Comparer les distributions avant et après transformation
par(mfrow=c(1,2))
hist(revenu_norm, main="Revenu normalisé", col="orange")
hist(revenu_std, main="Revenu standardisé", col="green")
```

---

6. Vérification de la cohérence et des unités

Exemple : Dataset incohérent

```r
# Exemple de dataset avec incohérences
data_incoherent <- data.frame(
  id = 1:5,
  age = c(25, -3, 200, 40, 30),
  poids = c("70kg", "80000g", "75kg", "150lb", "80kg"),
  montant = c("200€", "300$", "500€", "400€", "250$")
)
# Afficher le dataset
data_incoherent
```

Exercice 8 : Correction des incohérences

- Corriger les âges invalides
- Convertir les poids en kg (1lb = 0.453592kg)

- Convertir les montants en € (1$ = 0.93€ par ex.)

---

7. Transformation des données

Exemple : Label Encoding

```
# Label Encoding

# Variable catégorique
couleurs <- c("Rouge","Vert","Bleu","Rouge","Vert","Bleu")
# Encodage en numérique
as.numeric(factor(couleurs)) - 1
```

Exemple : One-Hot Encoding

```
# One-Hot Encoding avec le package fastDummies
# install.packages("fastDummies") # Si le package n'est pas déjà installé
library(fastDummies)
# Variable catégorique
couleurs <- c("Rouge", "Vert", "Bleu", "Rouge", "Vert", "Bleu")
# Créer un data frame
data <- data.frame(couleurs)
# Appliquer le One-Hot Encoding
dummy_fd <- dummy_cols(data, select_columns = "couleurs")
# Afficher le résultat
dummy_fd
```

Exercice 9 : Feature Engineering

- Créer une nouvelle variable : ratio revenu/âge
- Créer une variable binaire indiquant si revenu > 3000

```r
# Créer une nouvelle variable ratio revenu/âge
population$ratio <- population$revenu / population$age
# Créer une variable binaire indiquant si revenu > 3000
population$revenu_cat <- ifelse(population$revenu > 3000, 1, 0)
# Afficher les six premières lignes
head(population)
```
