



RÉSUMÉ THÉORIQUE – FILIÈRE INTELLIGENCE ARTIFICIELLE OPTION ASSISTANT DATA ANALYST M201 - Maîtriser les Systèmes Décisionnels



105 heures



SOMMAIRE



01 – Approfondir les statistiques

Rappeler les notions essentielles
Maitriser les mesures de dispersion et de position
Assimiler les probabilités et distributions avancées
Vulgariser le langage R

02 – Visualiser les données décisionnelles

Créer divers types de graphiques
Configurer les graphiques pour une analyse profonde

03 – Maitriser les bases de données décisionnelles

Introduire les bases de données décisionnelles
Explorer la modélisation dimensionnelle avancée
Appliquer le processus ETL dans le contexte décisionnel

MODALITÉS PÉDAGOGIQUES



1

LE GUIDE DE SOUTIEN

Il contient le résumé théorique et le manuel des travaux pratiques



2

LA VERSION PDF

Une version PDF est mise en ligne sur l'espace apprenant et formateur de la plateforme WebForce Life



3

DES CONTENUS TÉLÉCHARGEABLES

Les fiches de résumés ou des exercices sont téléchargeables sur WebForce Life



4

DU CONTENU INTERACTIF

Vous disposez de contenus interactifs sous forme d'exercices et de cours à utiliser sur WebForce Life



5

DES RESSOURCES EN LIGNES

Les ressources sont consultables en synchrone et en asynchrone pour s'adapter au rythme de l'apprentissage



PARTIE 1

APPROFONDIR LES STATISTIQUES

Dans ce module, vous allez :

- Rappeler les notions essentielles
- Maîtriser les mesures de dispersion et de position
- Assimiler les probabilités et distributions avancées
- Vulgariser le langage R



 45 heures

CHAPITRE 1

RAPPELER LES NOTIONS ESSENTIELLES



Ce que vous allez apprendre dans ce chapitre :

- Réviser rapidement les statistiques descriptives
- Identifier les types de variables statistiques
- Calculer les mesures centrales
- Appliquer pratiquement avec Python/Excel

 45 heures



CHAPITRE 1

RAPPELER LES NOTIONS ESSENTIELLES

1. Révision rapide des statistiques descriptives
2. Types de variables statistiques
3. Mesures centrales
4. Application pratique avec Python/Excel

Introduction aux statistiques

Les statistiques constituent un domaine essentiel des mathématiques appliquées qui se concentre sur la **collecte**, l'**analyse**, l'**interprétation**, la **présentation** et l'**organisation** des données. C'est un outil puissant utilisé dans une variété de disciplines pour prendre des décisions éclairées et tirer des **conclusions** basées sur des **informations numériques**.



01 – Rappeler les notions essentielles

Révision rapide des statistiques descriptives

Principales branches des statistiques

- **Statistique Descriptive** : Déterminer les **caractéristiques** d'une population. Elle utilise des mesures telles que la moyenne, la médiane, le mode, et d'autres outils pour synthétiser et visualiser les données de manière compréhensible.
- **Statistique Inférentielle** : Extrapoler et projeter les résultats numériques obtenus sur un **échantillon** à la population. Utilisant des échantillons représentatifs. Elle repose sur des concepts de **probabilité** pour extrapolier les résultats d'un échantillon à une population entière.



Décrire



Généraliser

Objectif de la statistique descriptive

L'objectif de la statistique descriptive consiste à **exposer et décrire** chaque Individu , c'est-à-dire à **résumer numériquement** et/ou à **représenter graphiquement**, les données disponibles lorsqu'elles sont nombreuses ou proviennent d'un recensement.

Définition de la statistique descriptive

Statistiques descriptives: méthode d'analyse de données visant à **résumer et décrire** des caractéristiques d'un ensemble de données.

Source: Dictionnaire Larousse.

A descriptive statistic is a **summary statistic** that quantitatively **describes** or summarizes features from a collection of information.



01 – Rappeler les notions essentielles

Révision rapide des statistiques descriptives

Remarques :

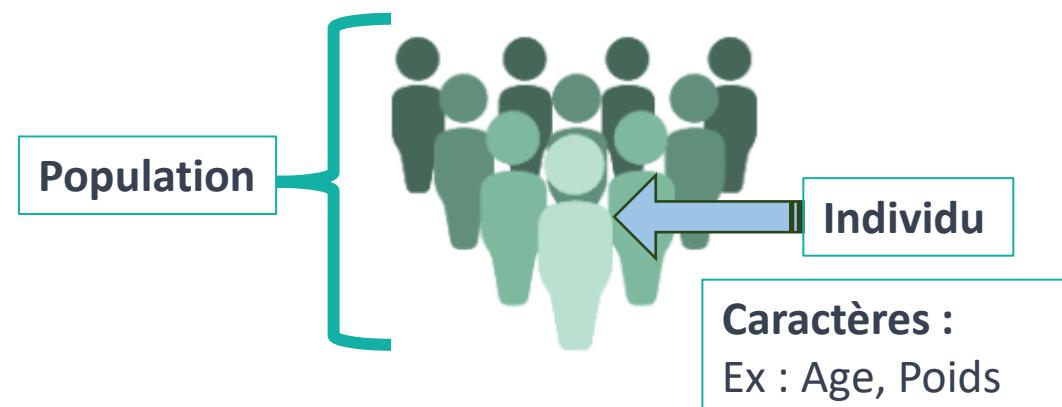


- Pour prédire : Utiliser les statistiques prédictives.
- Pour généraliser : Utiliser les statistiques inférentielles.
- Pour résumer : Utiliser l'analyse descriptive.



Définitions Fondamentales

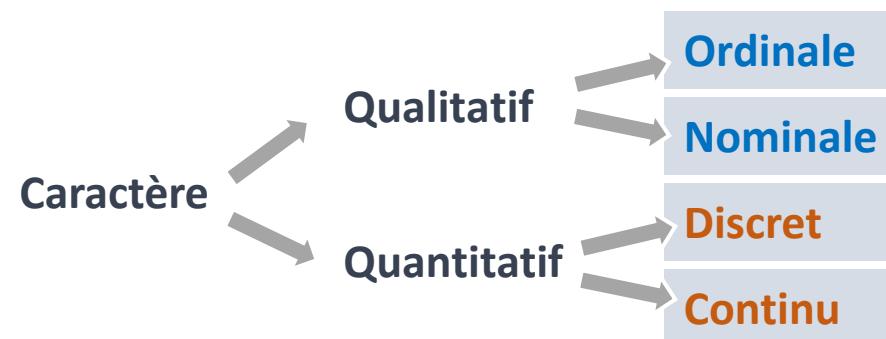
- L'ensemble étudié en statistique est appelé la **population**.
- Les éléments de cette population sont désignés comme **individus** ou **unités statistiques**.
- Les propriétés examinées sur les individus d'une population sont appelées les **caractères**.



Types de variables statistiques

Dans le domaine des statistiques, les caractères étudiés peuvent être classés en deux catégories distinctes :

- **Caractères Qualitatifs** : Ces caractères englobent des modalités non mesurables numériquement, exprimées par des mots, ou des phrases. Ces caractères qualitatifs peuvent, à leur tour, être de nature **ordinale** ou **nominale**
- **Caractères Quantitatifs** : Leur détermination conduit à un nombre ou une séquence de nombres. Ces caractères Quantitatifs peuvent, à leur tour, être de nature **discret** ou **continu**



Types de variables statistiques

- **Caractères Qualitatifs Ordinaux** : Les modalités présentent un ordre spécifique. Un exemple concret serait le niveau de satisfaction des clients dans un sondage, où les réponses sont "Insatisfait", "Neutre", "Satisfait", "Très satisfait".
- **Caractères Qualitatifs Nominaux** : Les modalités n'ont pas d'ordre intrinsèque. Par exemple, cela inclut les couleurs
- **Caractères Quantitatifs Discrets** : Ils se limitent à prendre des valeurs spécifiques. Un exemple concret serait le nombre de personnes dans une famille. Vous ne pouvez pas avoir un nombre fractionnaire de personnes.
- **Caractères Quantitatifs Continus** : Ils peuvent prendre n'importe quelle valeur réelle. Un exemple concret serait le poids des personnes en kilogrammes. Le poids peut prendre n'importe quelle valeur réelle entre zéro et l'infini, par exemple, 60,54 kg, 72,30 kg, 85,01 kg, etc. Il existe une infinité de valeurs possibles dans cette plage.

Types de variables statistiques



Parfois, pour classer les Caractères Quantitatifs Continus, nous pouvons créer des plages pour arrondir les données en Catégories Qualitatives Ordinales, par exemple, en regroupant les tailles des personnes en classes de "small", "medium", "large" et "Xlarge".



01 – Rappeler les notions essentielles

Mesures Centrales

Introduction

Les mesures centrales sont utilisées pour déterminer **le point central** ou typique d'un ensemble de données. Trois mesures clés sont la **moyenne**, le **mode** et la **médiane**.



01 – Rappeler les notions essentielles

Mesures Centrales



Moyenne (mean / Average)

- **Définition :** La moyenne est la somme de toutes les valeurs, divisée par le nombre total de valeurs.
- **Formula :** Moyenne = $\frac{\sum \text{Valeurs}}{\text{Nombre de Valeurs}}$
- **Interprétation :** Représente la valeur moyenne de l'ensemble de données.
- **Exemple :**

Valeurs : 5,6,8,9,12,15

$$\text{Moyenne} = 5+6+8+9+12+15$$

$$\text{Moyenne} = \frac{55}{6}$$

$$\text{Moyenne} \approx 9.16$$

Mode

- **Définition :** Le mode est **la valeur qui apparaît le plus fréquemment** dans un ensemble de données.
- **Relevance :** Indique la valeur la plus fréquente, **utile pour les données catégoriques**.
- **Exemple :** Dans une classe, les notes sont représentées par des mentions. La mention la plus fréquente est le mode.

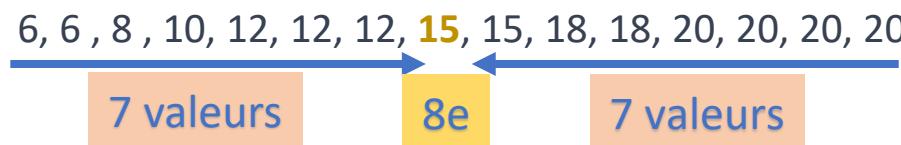
Valeurs : Assez Bien, Assez Bien, Bien, Bien, Très Bien, Très Bien, Très Bien, Excellent.

Effectif : Assez Bien : 2, Bien : 2, Très Bien : 3, Excellent : 1.

Mode : Très Bien

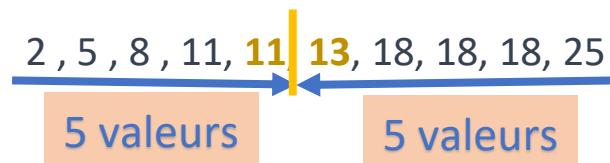
Médiane

- **Définition :** La médiane représente la valeur qui partage l'effectif total en **deux effectifs égaux**, après les avoir classées par ordre croissant.
- **Exemple 1:** La longueur de la liste est impaire
- Imaginons un enseignant évaluant des devoirs surveillés avec les notes suivantes :



Après le tri et avec un effectif total de 15, la médiane correspond à la 8e valeur, soit 15.

- **Exemple 2:** La longueur de la liste est paire
- Un autre enseignant recueille la série de notes suivante :



Après le tri et avec un effectif total de 10, la médiane se situe entre la 5e et la 6e valeur, la médiane est la moyenne de ces deux valeurs, soit 12.

01 – Rappeler les notions essentielles

Application pratique avec Excel

Exploration statistique des notes d'élèves en Excel : Mesures centrales

Cette application en Excel démontre l'utilisation des statistiques descriptives pour analyser les notes d'élèves.

| | A | B |
|----|--------------|------|
| 1 | ID Stagiaire | Note |
| 2 | S1 | 10 |
| 3 | S2 | 11 |
| 4 | S3 | 12 |
| 5 | S4 | 14 |
| 6 | S5 | 15 |
| 7 | S6 | 16 |
| 8 | S7 | 17 |
| 9 | S8 | 17 |
| 10 | S9 | 18 |
| 11 | S10 | 19 |
| 12 | Moyenne | 14,9 |
| 13 | Mode | 17 |
| 14 | Mediane | 15,5 |
| 15 | | |

La moyenne, calculée à l'aide de la fonction =MOYENNE(B2:B11), indique une répartition homogène des notes.

Utilisation de =MODE.SIMPLE(B2:B11) pour identifier le mode, mettant en lumière la note la plus fréquente.

La médiane, obtenue via =MEDIANE(B2:B11), offre une vue centrale résiliente aux valeurs extrêmes.

Comparaison des Fonctions Excel : Français vs. Anglais

Les fonctions principales d'Excel restent cohérentes, mais les noms des formules peuvent varier entre les versions linguistiques.



| Fonction | Français | Anglais |
|----------|----------------|--------------|
| Moyenne | =MOYENNE() | =AVERAGE() |
| Mode | =MODE.SIMPLE() | =MODE.SNGL() |
| Médiane | =MEDIANE() | =MEDIAN() |

Exploration Statistique des Notes d'Élèves en Python

Cette application Python démontre l'utilisation des statistiques descriptives pour analyser les notes d'élèves.

```
import statistics
# Données représentant les notes des élèves
notes = [17, 19, 15, 18, 14, 10, 12, 17, 11, 16]

# Calcul de la moyenne, du mode et de la médiane
moyenne = statistics.mean(notes)
mode_result = statistics.mode(notes)
median_result = statistics.median(notes)

# Affichage des résultats
print("Moyenne :", moyenne)
print("Mode :", mode_result)
print("Médiane :", median_result)

# Tri de la liste pour la comparaison
notes.sort()
print("Après le tri pour la comparaison :\n", notes)
```

Les méthodes incluent le calcul de la moyenne, du mode et de la médiane, ainsi que le tri de la liste pour une comparaison approfondie.



Exploration Statistique des Notes d'Élèves en Python

Résultats et Interprétations :

La moyenne des notes s'établit à un niveau équilibré, indiquant une tendance générale au sein de la classe.

Moyenne : 14.9
Mode : 17
Médiane : 15.5
Après le tri pour la comparaison :
[10, 11, 12, 14, 15, 16, 17, 17, 18, 19]

Mode identifie la note la plus fréquente, soulignant une tendance dominante.

Médiane, résiliente aux extrêmes, offre une vue centrale de la distribution des notes.

5 valeurs

5 valeurs

Tri de la liste facilitant la comparaison visuelle des résultats.

Consulter les Dernières Intégrations



Certaines fonctions de statistiques telles que la Moyenne, le Mode et la Médiane peuvent être utilisées à partir des packages `scipy` ou de `Numpy`, mais les packages Python sont soumis à des changements constants au fil du temps. Lorsque vous abordez ce cours, assurez-vous de vérifier les dernières intégrations disponibles.





CHAPITRE 2

MAITRISER LES MESURES DE DISPERSION ET DE POSITION

Ce que vous allez apprendre dans ce chapitre :

- Réviser les mesures de dispersion et de position
- Analyser les outliers
- Appliquer pratiquement avec Python/Excel

 10 heures



CHAPITRE 2

MAITRISER LES MESURES DE DISPERSION ET DE POSITION

1. Rappeler les notions essentielles
2. **Maitriser les mesures de dispersion et de position**
3. Assimiler les probabilités et distributions avancées
4. Vulgariser le langage R

01 – Maîtriser les mesures de dispersion et de position

Rappel des notions

Introduction

Les **mesures centrales** révèlent la tendance centrale ou la valeur centrale d'un ensemble de données.

Les **mesures de dispersion** décrivent la répartition ou la dispersion des valeurs dans un ensemble de données.

Les **mesures de position** indiquent la position relative d'une valeur par rapport aux autres dans un ensemble de données.

Mesures Centrales :
moyenne, médiane, mode

Mesures de Dispersion :
Étendue, Variance, Écart-type

Mesures de Position :
Quartiles, Déciles, Centiles

01 – Maitriser les mesures de dispersion et de position

Rappel des notions



Étendue (Range) : Mesure de dispersion

L'étendue ou range en anglais, est une mesure de dispersion qui représente **la différence** entre **la valeur maximale** et **la valeur minimale** d'un ensemble de données.

$$E = x(\max) - x(\min)$$

$$E(\text{Suite Ord.}) = x(n) - x(1)$$

- Dans une suite ordonnée croissante, $x(1)$ et $x(n)$ sont représentent respectivement l'observation extrême inférieure et supérieure

Interprétation :

Une étendue plus grande indique une dispersion plus importante des données, tandis qu'une étendue plus petite suggère une concentration autour de la moyenne

Limitations :

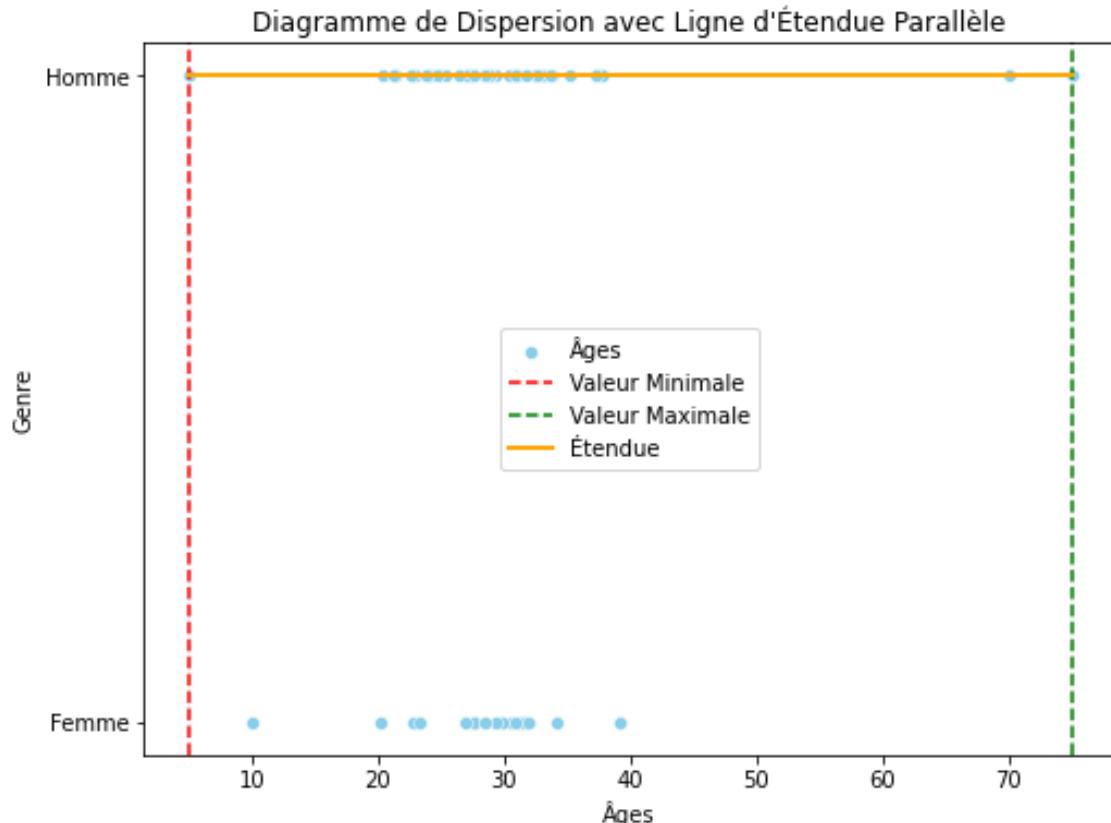
- Ne tient pas compte de la distribution interne des données.
- Peut être influencée par des valeurs aberrantes.

01 – Maîtriser les mesures de dispersion et de position

Rappel des notions

Étendue (Range) : Mesure de dispersion

Ce graphique de type scatter plot montre le genre en fonction de l'âge, avec une étendue de 70.00 unités, allant de 5.00 à 75.00.



La création et la configuration des graphiques seront expliquées dans les prochaines sections du cours.

01 – Maîtriser les mesures de dispersion et de position

Rappel des notions



Variance (Variation) : Mesure de dispersion

La variance σ^2 mesure la dispersion des valeurs au sein d'un ensemble de données. Elle est calculée en déterminant la moyenne des carrés des écarts **entre chaque valeur et la moyenne totale**. Pour calculer la variance de l'échantillon, divisez la somme des carrés des déviations par le nombre total de valeurs moins un

$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}$$

Interprétation :

Une variance élevée indique une dispersion importante, reflétant des valeurs plus éloignées de la moyenne, tandis qu'une faible variance suggère une concentration plus étroite autour de la moyenne.

Avantages :

Tient compte de la distribution interne des données.

Utile pour évaluer la variabilité.

Limitations :

Sensible aux valeurs aberrantes.

01 – Maitriser les mesures de dispersion et de position

Rappel des notions

Variance (Variation) : Mesure de dispersion

Méthode de Calcul :

Pour calculer la variance pour les données (6, 7, 8), nous utiliserons la formule de la variance pour un échantillon puisque les données représentent un échantillon :

✓ Calcul de la Moyenne (\bar{X}):

$$\bar{X} = \frac{\sum \text{Valeurs}}{\text{Nombre de Valeurs}}$$

$$\bar{X} = \frac{6 + 7 + 8}{3} = \frac{21}{3} = 7$$

✓ Calcul des Carrés des Déviations par rapport à la Moyenne de l'échantillon : $(X_i - \bar{X})^2$

Pour 6 : $(6 - 7)^2 = 1$

Pour 7 : $(7 - 7)^2 = 0$

Pour 8 : $(8 - 7)^2 = 1$

✓ Calcul de la Variance de l'échantillon :

$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \bar{X})^2}{n-1} \quad \rightarrow \quad \sigma^2 = \frac{1+0+1}{3-1} = \frac{2}{2} = 1$$

Par conséquent, la variance de l'échantillon pour les données (6, 7, 8) est 1.

01 – Maîtriser les mesures de dispersion et de position

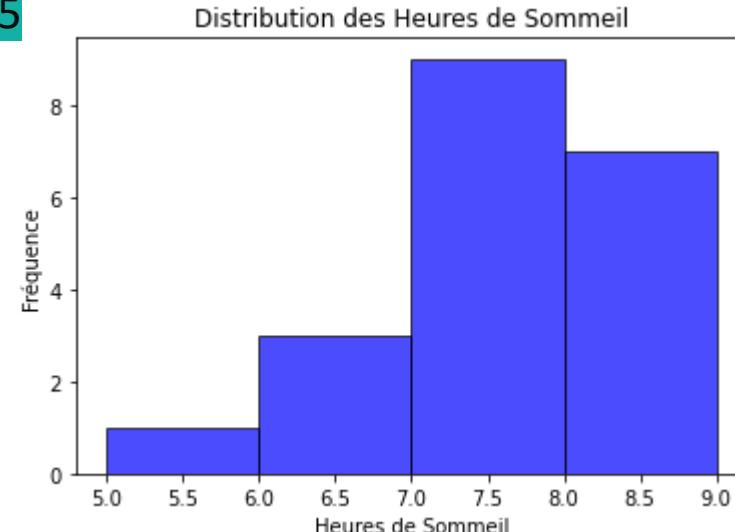
Rappel des notions

Variance (Variation) : Mesure de dispersion

Histogramme 1 - Distribution des Heures de Sommeil:

Moyenne: 7.25 Heures

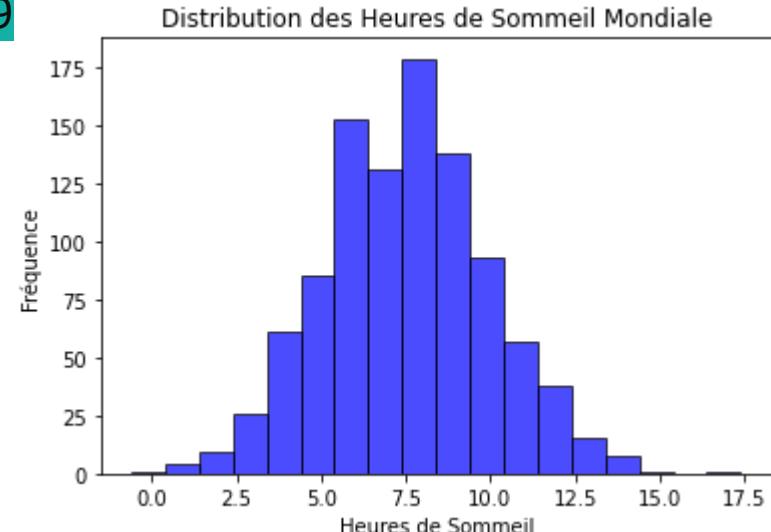
Variance: 1.0875



Histogramme 2 - Distribution des Heures de Sommeil Mondiale:

Moyenne: 7.55 Heures

Variance: 5.99



Interprétation :

Le premier histogramme illustre une **distribution localisée** avec une **faible variance** (1.0875), indiquant des habitudes de sommeil cohérentes autour de la moyenne. En revanche, le deuxième histogramme, représentant les heures de sommeil à l'échelle mondiale, présente une **distribution plus large** et plus diversifiée avec une **variance plus élevée** (5.99), soulignant une variabilité significative dans les habitudes de sommeil à l'échelle mondiale malgré une moyenne similaire.

La création de l'histogramme est expliquée au chapitre 1 : CRÉER DIVERS TYPES DE GRAPHIQUES du Deuxième partie du cours.

01 – Maitriser les mesures de dispersion et de position

Rappel des notions

Ecart-type(standard deviation) (σ) : Mesure de dispersion

L'Écart-type ou standard deviation en anglais mesure la dispersion des valeurs au sein d'un ensemble de données. Il est calculé en déterminant la racine carrée de la variance, qui elle-même est obtenue en moyennant les carrés des écarts entre chaque valeur et la moyenne totale.

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n-1}}$$

Interprétation :

Une valeur d'écart-type élevée indique une dispersion importante, signifiant que les valeurs sont plus éloignées de la moyenne. À l'inverse, un écart-type faible suggère une concentration plus étroite autour de la moyenne.

Avantages :

Mesure la dispersion des données dans les **mêmes unités que les valeurs d'origine**, ce qui facilite la compréhension de la variabilité des données.

Tient compte de la distribution interne des données.

Limitations :

Sensible aux valeurs aberrantes.

01 – Maîtriser les mesures de dispersion et de position

Rappel des notions

Variance (σ^2) Vs Ecart-type (" σ ")

| Caractéristique | Variance | Écart-type |
|--------------------|---|--|
| Définition | Mesure l'écart moyen au carré par rapport à la moyenne. | La racine carrée de la variance, représente la distance moyenne entre chaque point de données et la moyenne. |
| Nature | En unités carrées des données d'origine. | Dans les mêmes unités que les données d'origine. |
| Unités | Carrées | Originales |
| Magnitude | Mesure la dispersion dans des unités carrées. | Fournit une mesure interprétable dans les unités d'origine. |
| Sensibilité | Moins sensible aux valeurs aberrantes. | Plus sensible aux valeurs aberrantes en raison de la racine carrée. |

01 – Maitriser les mesures de dispersion et de position

Rappel des notions



Quartiles: Mesure de position

Les quartiles sont des valeurs qui divisent un ensemble de données **triées** en quatre parties égales, chacune représentant 25% des données.

Calcul : Q_1 (premier quartile) est à 25%, Q_2 (deuxième quartile ou médiane) est à 50%, Q_3 (troisième quartile) est à 75%.

Interprétation : Q_1 représente le point où 25% des données sont inférieures, Q_2 est la médiane, et Q_3 représente le point où 75% des données sont inférieures.

Q1 : 25%

Premier quartile

Q2 : 50%

Médiane

Q3 : 75%

Troisième quartile

01 – Maitriser les mesures de dispersion et de position

Rappel des notions



Déciles : Mesure de position

Les déciles sont des valeurs qui divisent un ensemble de données **triées** en dix parties égales, chaque décile représentant 10% des données.

Calcul : D_1 à D_9 représentent respectivement 10% à 90%, et D_{10} est équivalent au centile.

Interprétation : Les déciles permettent de visualiser la distribution des données en segments de 10%.

D1 : 10%

Premier décile

D2 : 20%

Deuxième décile

...

01 – Maitriser les mesures de dispersion et de position

Rappel des notions



Centiles : Mesure de position

Les centiles sont des valeurs qui divisent un ensemble de données **triées** en cent parties égales, chaque centile représentant 1% des données.

Calcul : C_1 à C_{99} représentent respectivement 1% à 99%, et C_{100} **est la valeur maximale.**

Interprétation : Les centiles permettent de quantifier la position d'une valeur spécifique par rapport à l'ensemble des données.

C1 : 1%
Premier centile

C2 : 2%
Deuxième centile

...

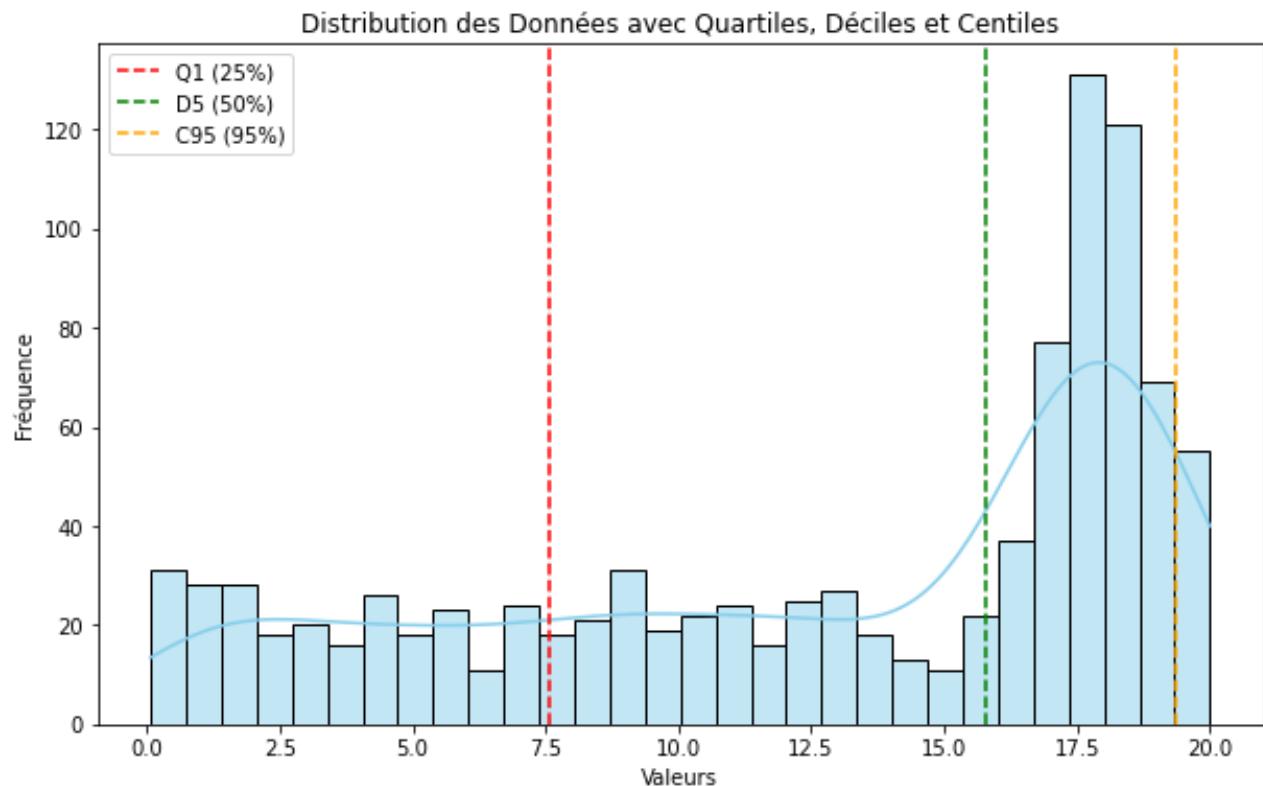
01 – Maîtriser les mesures de dispersion et de position

Rappel des notions

Exemple d'application aux résultats des étudiants : Interprétation et Analyse des Quartiles, Déciles et Centiles

Si Q_1 dans un ensemble de notes est égal à 7,5, cela signifie que 25% des étudiants ont obtenu 7,5 ou moins. Si D_5 est 16, alors 50% des données sont inférieures à 16. Enfin, si C_{95} est 19, cela indique que 95% des données sont inférieures à 19.

Ces termes permettent une compréhension détaillée de la distribution des données.



01 – Maîtriser les mesures de dispersion et de position

Rappel des notions

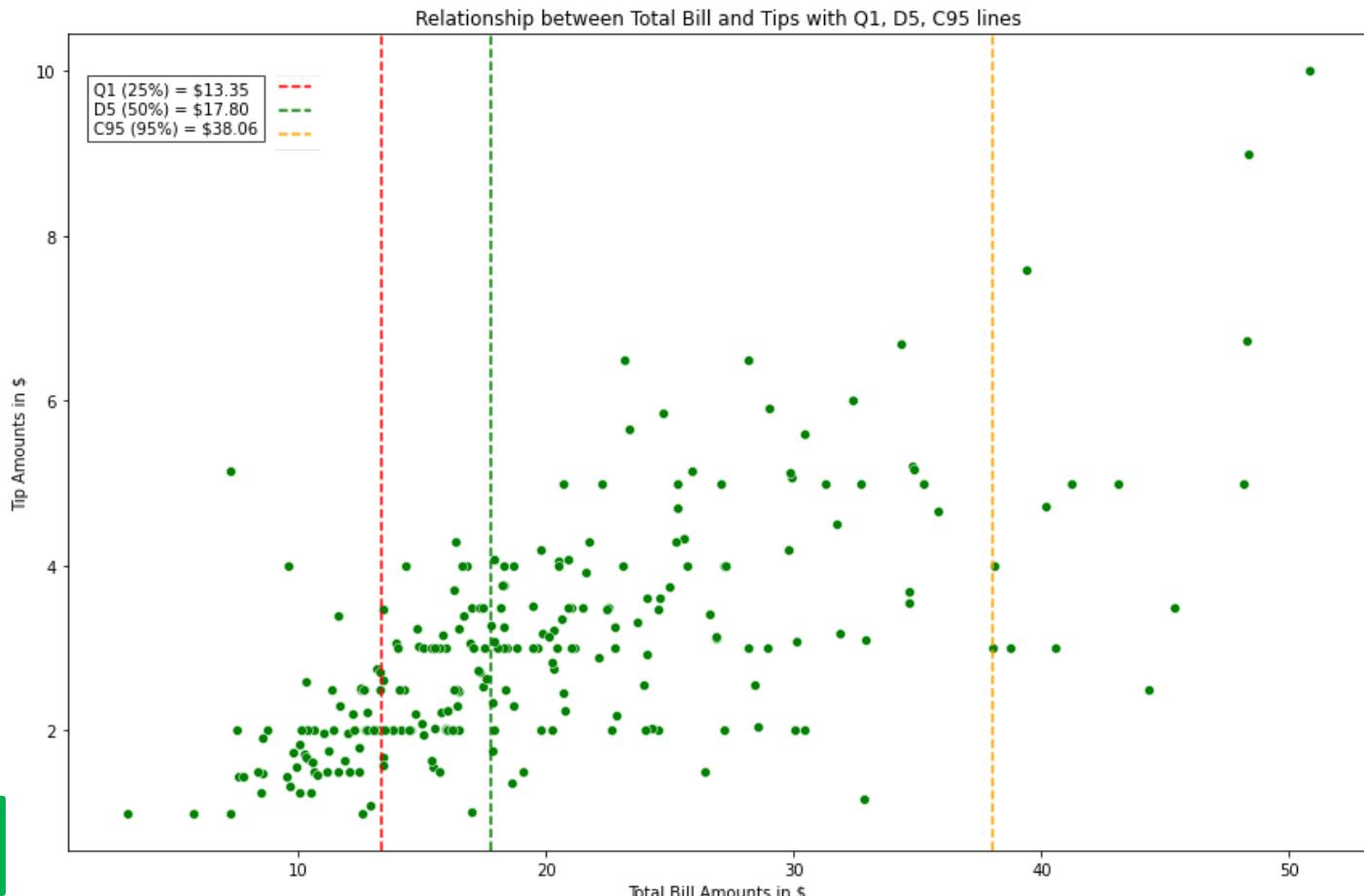
Exemple d'Application sur Tips Dataset : Visualisation des Quartiles, Déciles et Centiles

Pour aller plus loin dans notre exploration des statistiques, nous allons utiliser un ensemble de données réelles appelé 'Tips'. Ce jeu de données contient des informations sur les factures de restaurants, les pourboires et plus encore..

Cette approche, axée sur la compréhension des données, est appelée **l'analyse descriptive**.



La création de l'histogramme est expliquée au chapitre 1 : CRÉER DIVERS TYPES DE GRAPHIQUES du Deuxième partie du cours.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Introduction :

En statistique Les valeurs aberrantes ou **outliers** en anglais, sont des points de données qui diffèrent significativement de la majorité des données dans un ensemble. Ce sont des observations qui se situent à une distance anormale par rapport aux autres valeurs, indiquant potentiellement un événement rare ou une erreur de mesure. Identifier et comprendre ces outliers revêt une importance cruciale dans l'analyse statistique, car ils peuvent exercer une influence sur les résultats et l'interprétation des analyses. Particulièrement dans des ensembles de données plus petits, leur influence peut être majeure.



In statistics, an outlier is a data point that differs significantly from other observations.

01 – Maîtriser les mesures de dispersion et de position

Analyse des Outliers

Impact des valeurs aberrantes sur l'analyse et la modélisation des données :

Les valeurs aberrantes peuvent avoir un impact significatif sur les résultats de l'analyse et de la modélisation des données. Elles peuvent fausser les mesures statistiques, conduisant à des estimations incorrectes de la tendance centrale et de la dispersion.

Par exemple, si un ensemble de données sur les salaires des employés contient quelques salaires extrêmement élevés en raison de primes de direction, le salaire moyen peut être considérablement **s surestimé**, entraînant une moyenne salariale exagérée pour l'organisation.

De même, si un ensemble de données sur les prix immobiliers contient quelques prix très bas en raison d'**erreurs de saisie**, le prix médian peut ne pas représenter avec précision le prix typique des maisons dans cette région.

Les outliers peuvent également affecter les **performances des modèles de ML**. Certains modèles, tels que la régression linéaire, sont sensibles aux outliers.

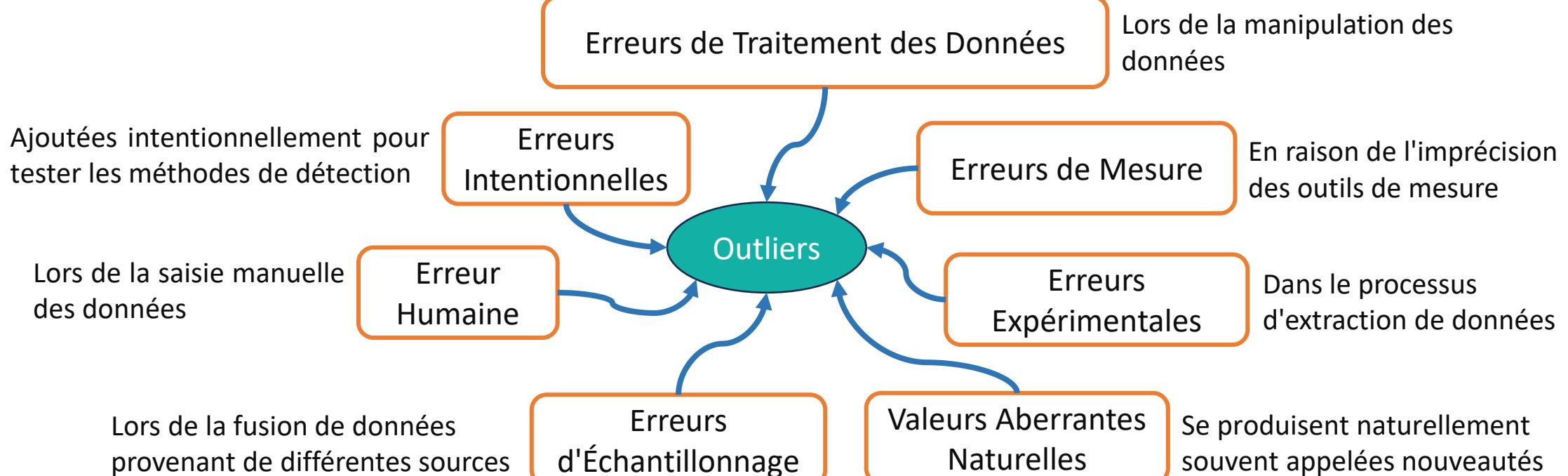


01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Causes des outliers ou valeurs aberrantes

Comprendre les causes des valeurs aberrantes aide grandement à les traiter, à prendre des décisions éclairées et à réduire leur occurrence lors des observations à venir.



01 – Maîtriser les mesures de dispersion et de position

Analyse des Outliers

Analyse et identification des Outliers

L'identification des outliers est un processus crucial dans l'analyse de données pour repérer les valeurs aberrantes. Dans ce cours, nous proposons trois méthodes principales :

- **Mesures Statistiques** : Utilisation des mesures telles que le minimum et le maximum.
- **Visualisation** : Emploi de graphiques tels que les box plots et même les tableaux d'Excel pour une identification visuelle.
- **Techniques Statistiques Avancées** : Mise en œuvre de méthodes comme DBSCAN pour une identification précise.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Gestion et traitement des outliers

Explorez le processus de gestion des valeurs aberrantes dans l'analyse de données. Trois éléments clés guideront cette gestion :

- **Stratégies de remplacement** : Explorez des stratégies telles que le remplacement par des valeurs moyennes ou médianes pour des analyses plus robustes.
- **Supprimer les outliers** : Déterminez judicieusement quand supprimer les valeurs aberrantes de l'ensemble de données.
- **Acceptez la présence d'outliers lorsque justifié** : Discernez les situations où la présence d'outliers est pertinente et peut apporter des insights significatifs.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers



Mesures statistiques : Méthode des deux écarts-types (Two standard deviations method)

La méthode des deux écarts-types est une technique statistique qui identifie les valeurs aberrantes en considérant celles qui se situent en dehors de l'intervalle défini par **la moyenne plus ou moins deux fois l'écart-type**, dans le cadre d'une **distribution normale**. Elle repose sur l'idée que la plupart des données dans une distribution normale se trouvent dans l'intervalle de deux écarts-types de la moyenne, et donc, les valeurs en dehors de cet intervalle peuvent être considérées comme des valeurs aberrantes.

Le calcul se fait à l'aide de la fonction :

$$\text{Seuil} = \bar{X} (+ ou -) 2\sigma$$

Application de la méthode : Example des revenus mensuels

Génération des données :

Revenus mensuels : [4900, 4800, 5200, 5100, 4900, 4800, 5000, 15000, 4900, 4800, 1000, ...] (100 valeurs au total)

Calcul des mesures statistiques :

Moyenne :

$$\bar{X} = \frac{\sum \text{Valeurs}}{\text{Nombre de Valeurs}}$$

$$\bar{X} = (4900+4800+5200+\dots+1000)/100 \Rightarrow \bar{X} = 4966$$

Écart-type :

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^n (Xi - \bar{X})^2}{n-1}}$$

$$\sigma = \sqrt{((4900-4966)^2 + (4800-4966)^2 + \dots + (1000-4966)^2) / 99} \Rightarrow \sigma = 1407$$

01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Mesures statistiques : Méthode des deux écarts-types

Identification des seuils supérieur et inférieur :

Seuil Supérieur : $S_s = \bar{X} + 2\sigma$

$$\text{Seuil Supérieur} = 4966 + 2 \times 1407 = 7781$$

Seuil Inférieur : $S_i = \bar{X} - 2\sigma$

$$\text{Seuil Inferieur} = 4966 - 2 \times 1407 = 2152$$

Identification des Outliers :

Les valeurs aberrantes « Outliers » sont celles qui sont au-dessus du seuil supérieur ou en dessous du seuil inférieur

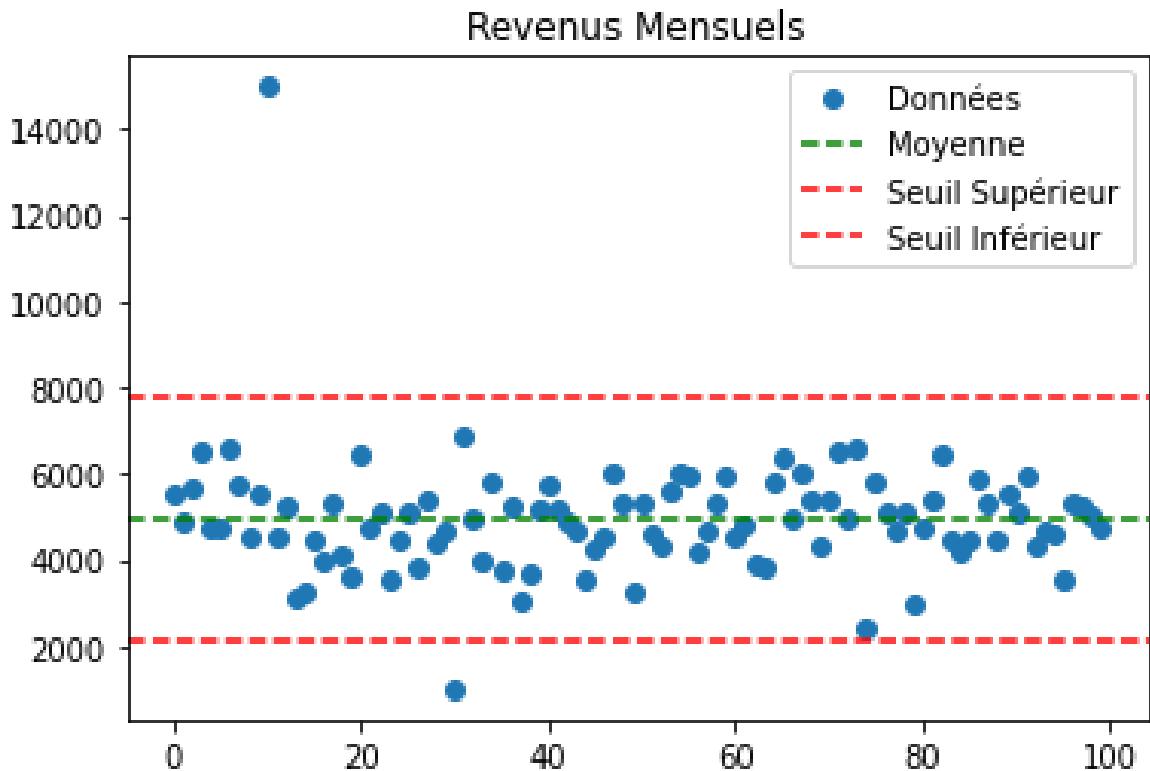
Affichage des résultats :

Revenus mensuels : [4900, 4800, 5200, 5100, 4900, 4800, 5000, 15000, 4900, 4800, 1000, ...]

Moyenne : 4966 | Écart-type : 1407

Seuil Supérieur : 4966 | Seuil Inférieur : 2152

Ex. Valeurs Aberrantes : [15000, 1000]



01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification des outliers : Example de calcul de la méthode des deux écarts-types avec Python

Génération des données :

Considérons un ensemble de données avec 10 valeurs :

```
import numpy as np  
  
data = np.array([15, 22, 18, 25, 20, 17, 23, 21, 19, 30])
```

Calcul des mesures statistiques :

Étape 1 : Calcul de la Moyenne et de l'Écart-type

```
mean = np.mean(data)  
  
std_dev = np.std(data)
```

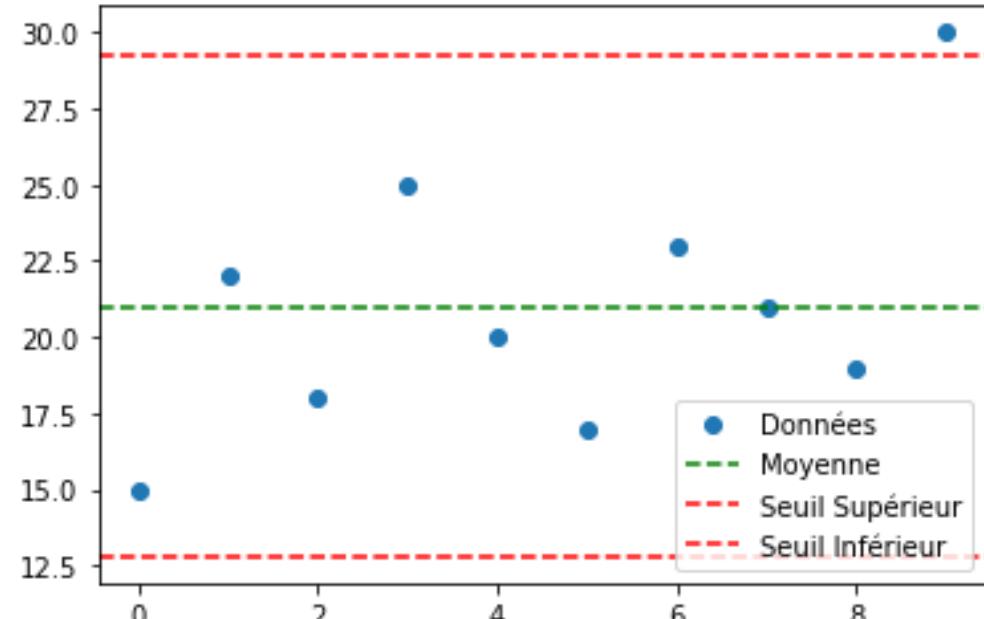
Étape 2 : Définition des Seuils

```
threshold_upper = mean + 2 * std_dev  
  
threshold_lower = mean - 2 * std_dev
```

Étape 3 : Identification des Valeurs Aberrantes

```
outliers = [value for value in data if value > threshold_upper or value < threshold_lower]
```

Affichage des résultats :



```
Données : [15 22 18 25 20 17 23 21 19 30]  
Moyenne : 21.0 Écart-type : 4.09878030638384  
Seuil supérieur : 29.19756061276768  
Seuil inférieur : 12.80243938723232  
Valeurs aberrantes : [30]
```

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification des outliers : Affichage de la méthode des deux écarts-types avec Python

Reference d'affichage numériques :

```
print("Données :", data)
print("Moyenne :", mean)
print("Écart-type :", std_dev)
print("Seuil supérieur :", threshold_upper)
print("Seuil inférieur :", threshold_lower)
print("Valeurs aberrantes :", outliers)
```

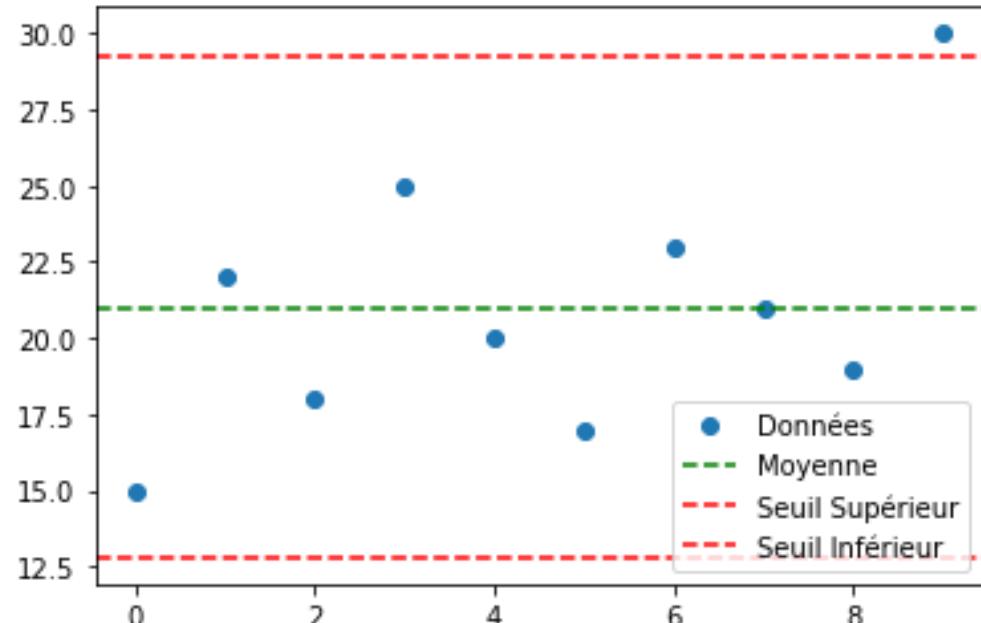


Sortie Attendue :

```
Données : [15 22 18 25 20 17 23 21 19 30]
Moyenne : 21.0 Écart-type : 4.09878030638384
Seuil supérieur : 29.19756061276768
Seuil inférieur : 12.80243938723232
Valeurs aberrantes : [30]
```

Reference d'affichage graphique :

```
import matplotlib.pyplot as plt
# Visualisation des données avec seuils
plt.plot(data, 'o', label='Données')
plt.axhline(y=mean, color='g', linestyle='--', label='Moyenne')
plt.axhline(y=threshold_upper, color='r', linestyle='--', label='Seuil Supérieur')
plt.axhline(y=threshold_lower, color='r', linestyle='--', label='Seuil Inférieur')
plt.legend()
plt.show()
```



Mesures statistiques :

Méthode IQR (Interquartile Range) : $IQR = Q3 - Q1$

L'intervalle interquartile (IQR) est une mesure statistique qui évalue la dispersion des données au sein d'un ensemble. Il est défini comme la différence entre le troisième quartile (Q3) et le premier quartile (Q1). Utilisé pour détecter les valeurs aberrantes, l'IQR identifie les points de données situés en dehors d'une plage définie par 1,5 (K) fois la longueur de l'IQR au-dessus du troisième quartile et en dessous du premier quartile. Cela permet de cibler les observations inhabituelles dans un ensemble de données.

$$\text{Seuil inférieure} = Q1 - 1,5 \times IQR \quad \text{Seuil supérieure} = Q3 + 1,5 \times IQR$$

Méthode Tukey :

L'utilisation de K = 3 au lieu de 1,5 signifie l'application de la méthode de Tukey, qui a tendance à éliminer moins de données.

Application de la méthode IQR :

Les Etapes à suivre :

1. Calculez le premier quartile (Q1) et le troisième quartile (Q3) de vos données.
2. Calculez l'écart interquartile (IQR) en soustrayant Q1 de Q3 : $IQR = Q3 - Q1$.
3. Définissez une limite inférieure comme $Q1 - 1,5 \times IQR$ et une limite supérieure comme $Q3 + 1,5 \times IQR$.
4. Toute valeur en dehors de ces limites est considérée comme une valeur aberrante.



Les valeurs couramment utilisées pour k sont 1.5 et 3, bien que d'autres valeurs puissent également utilisées en fonction des besoins.

Identification des outliers : Application de la méthode IQR et Tukey utilisent Python

Pour détecter les Outliers dans un ensemble de données. la méthode statistique IQR évalue la dispersion des données en identifiant les points situés en dehors d'une plage définie par le premier et le troisième quartile, offrant une approche efficace pour cibler les observations inhabituelles.

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

# Exemple de données
data = np.array([2, 3, 4, 5, 6, 7, 20])

# Calcul de Q1, Q3 et IQR
q1 = np.percentile(data, 25)
q3 = np.percentile(data, 75)
iqr = q3 - q1

# Calcul des limites pour la méthode IQR
iqr_limite_inf = q1 - 1.5 * iqr
iqr_limite_sup = q3 + 1.5 * iqr

# Calcul des limites pour la méthode de Tukey
tukey_limite_inf = q1 - 3 * iqr
tukey_limite_sup = q3 + 3 * iqr
```

```
# Identification des valeurs aberrantes
iqr_outliers = data[(data < iqr_limite_inf) | (data > iqr_limite_sup)]
tukey_outliers = data[(data < tukey_limite_inf) | (data > tukey_limite_sup)]

# Affichage des valeurs calculées
print(f"Q1 : {q1}, Q3 : {q3}, IQR : {iqr}")
print(f"Limite inférieure IQR : {iqr_limite_inf}, Limite supérieure IQR : {iqr_limite_sup}")
print(f"Limite inférieure Tukey : {tukey_limite_inf}, Limite supérieure Tukey : {tukey_limite_sup}")
print(f"Valeurs aberrantes IQR : {iqr_outliers}")
print(f"Valeurs aberrantes Tukey : {tukey_outliers}")
```

Sortie Attendue :

```
Q1 : 3.5, Q3 : 6.5, IQR : 3.0
Limite inférieure IQR : -1.0, Limite supérieure IQR : 11.0
Limite inférieure Tukey : -5.5, Limite supérieure Tukey : 15.5
Valeurs aberrantes IQR : [20]
Valeurs aberrantes Tukey : [20]
```

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification des outliers : Visualisation des mesures IQR et Tukey utilisent Python

Le graphique de visualisation présente les données initiales avec des points de données en bleu, les quartiles Q1 et Q3 en vert et rouge respectivement, ainsi que les limites définies par la méthode IQR et la méthode de Tukey. Les valeurs aberrantes détectées par chaque méthode sont marquées en rouge (IQR) ou orange (Tukey). Cette visualisation permet une compréhension rapide des points atypiques dans le jeu de données.



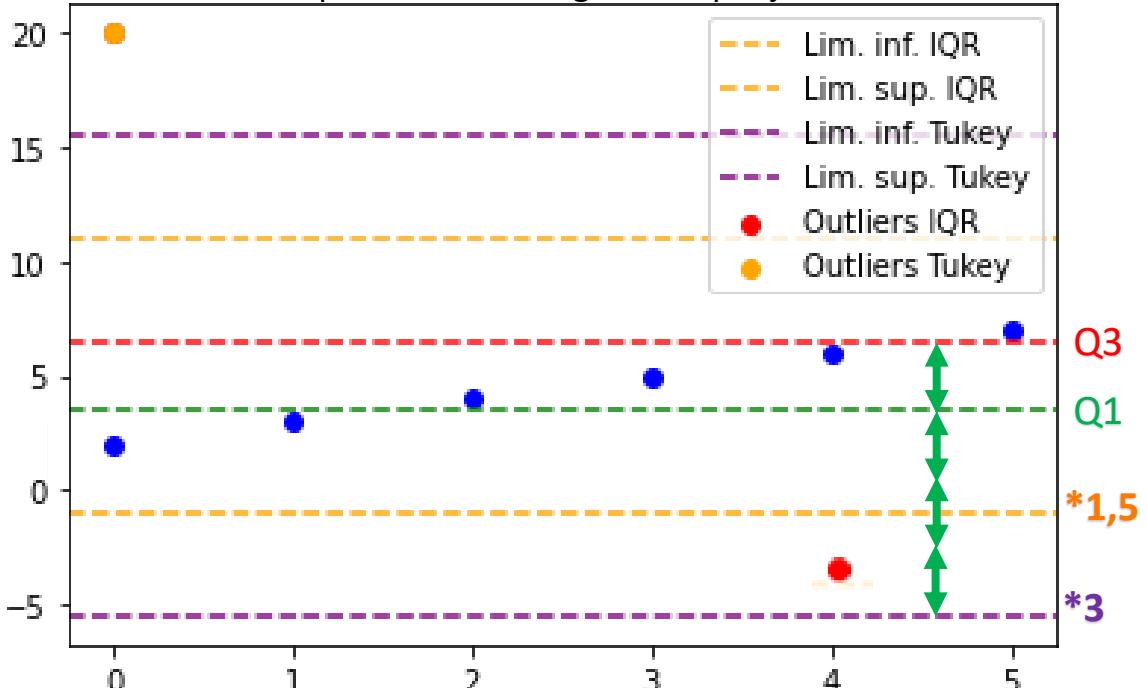
Des données bien nettoyées et propres sont essentielles pour renforcer la détection des intrusions et améliorer la fiabilité des systèmes.



La diapositive suivante présente les étapes à suivre pour générer ce graph de type scatter.

les types de graphiques et leur création sont abordés dans d'autres sections suivantes et intégrés en parallèle avec le cours.

Température du réfrigérateur par jour



01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python



Identification des outliers : Visualisation des mesures IQR et Tukey utilisent Python

Pour référence, nous présentons le code python nécessaire à la création de graphique précédente.

```
# Création d'un masque pour afficher uniquement les valeurs correctes
correct_data_mask = ~np.isin(data, iqr_outliers) & ~np.isin(data, tukey_outliers)

# Nuage de points avec uniquement les valeurs correctes
plt.scatter(range(len(data[correct_data_mask])), data[correct_data_mask], color='blue')#, label='Points de données'
plt.axhline(y=q1, color='green', linestyle='--') #, label='Q1'
plt.axhline(y=q3, color='red', linestyle='--') #, label='Q3'
plt.axhline(y=iqr_limite_inf, color='orange', linestyle='--', label='Lim. inf. IQR')
plt.axhline(y=iqr_limite_sup, color='orange', linestyle='--', label='Lim. sup. IQR')
plt.axhline(y=tukey_limite_inf, color='purple', linestyle='--', label='Lim. inf. Tukey')
plt.axhline(y=tukey_limite_sup, color='purple', linestyle='--', label='Lim. sup. Tukey')
plt.scatter(range(len(iqr_outliers)), iqr_outliers, color='red', label='Outliers IQR')
plt.scatter(range(len(tukey_outliers)), tukey_outliers, color='orange', label='Outliers Tukey')

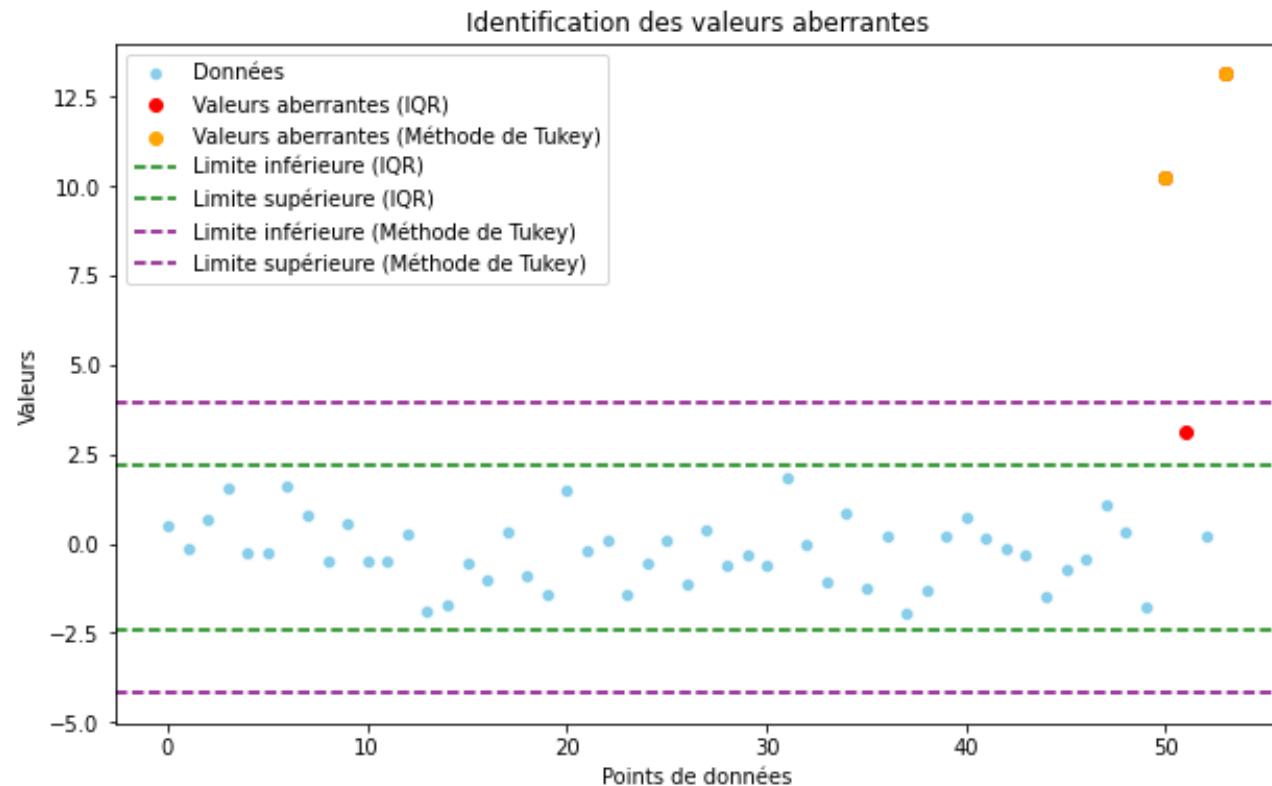
# Affichage du graphique
plt.legend()
plt.show()
```

01 – Maîtriser les mesures de dispersion et de position

Application pratique avec Python

Identification des outliers : 2^eme Visualisation des mesures IQR et Tukey

Le graph de visualisation présente les limites définies par la méthode IQR et la méthode de Tukey. Les valeurs aberrantes détectées par chaque méthode sont marquées en rouge (IQR) et orange (Tukey). Cette visualisation simple est suffisante pour la détection des points atypiques dans le jeu de données.



Identification par visualisation: Box plot un outil efficace pour identifier les valeurs aberrantes

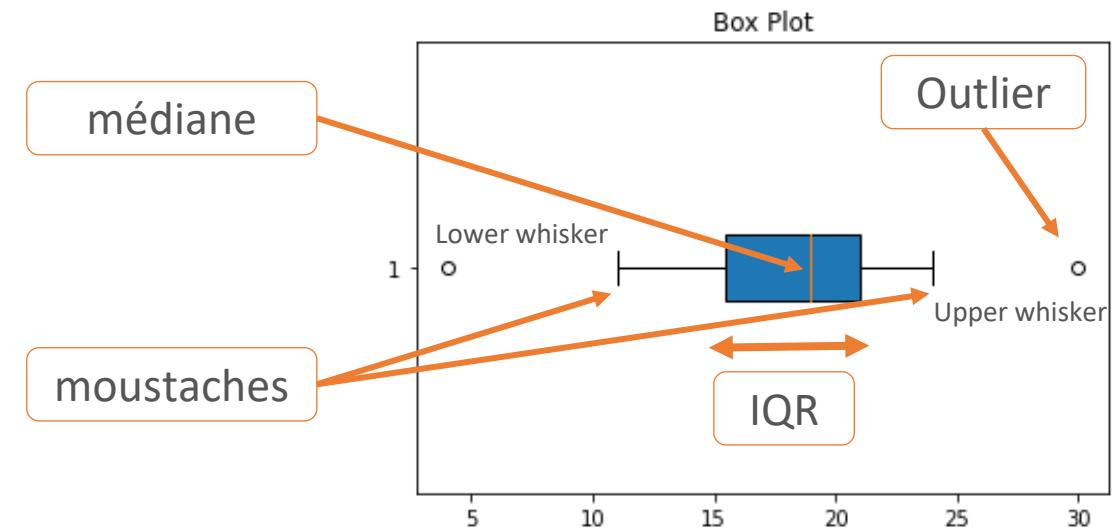
L'utilisation de graphiques, tels que les box plots et scatter plots, est une méthode puissante pour détecter visuellement les valeurs aberrantes. Les box plots présentent une représentation graphique des quartiles, de la médiane et des valeurs extrêmes, offrant une vue d'ensemble de la distribution des données.

Box Plots:

Les box plots permettent de repérer les valeurs aberrantes de manière intuitive. La boîte représente l'étendue interquartile IQR (**Interquartile Range**) entre le premier quartile (Q1) et le troisième quartile (Q3). Les lignes (moustaches) s'étendent jusqu'aux valeurs les plus éloignées qui ne sont pas considérées comme des valeurs aberrantes.

Interprétation des Box Plots :

Les box plots visualisent la distribution des données, montrant la médiane, les quartiles, et les valeurs minimale et maximale. Ils sont utiles pour comparer visuellement les ensembles de données. Les outliers sont représentés à l'extérieur des moustaches.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Identification par visualisation : Interprétation des Box Plots

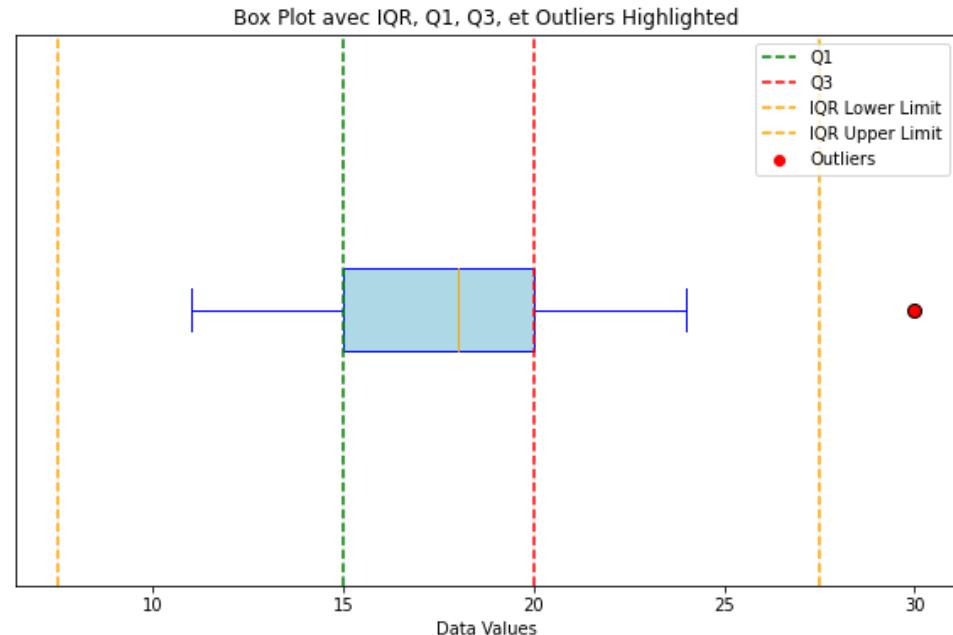
Lecture d'un Box Plot :

- Minimum et maximum alignés aux extrémités.
- Quartiles inférieur et supérieur alignés aux bords de la boîte.
- Médiane alignée avec la ligne intérieure de la boîte.

Analyse d'un Box Plot :

- **Médiane** plus grande indique une moyenne plus grande.
- **Étendue** = maximum – minimum. **Une plus grande étendue signifie une dispersion plus importante.**
- **IQR** = $Q3 - Q1$. Un IQR plus grand signifie une dispersion accrue pour la moitié centrale des données.

Outliers par box plot :
Valeurs aberrantes **à l'extérieur** des moustaches.



Rappel IQR method :
 $iqr_lower_limit = q1 - 1.5 * iqr$
 $iqr_upper_limit = q3 + 1.5 * iqr$

01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

Techniques statistiques avancées : Z-Score (Standardization)

Le Z-score, ou standard score, est une mesure statistique qui quantifie de **combien d'écart-types un point de données se trouve par rapport à la moyenne** d'un ensemble de données. Il est exprimé en tant que nombre d'écart-types par rapport à la moyenne, indiquant ainsi la position relative d'une observation ou d'un point de données particulier.

Formule Population : $Z = (x - \mu) / \sigma$

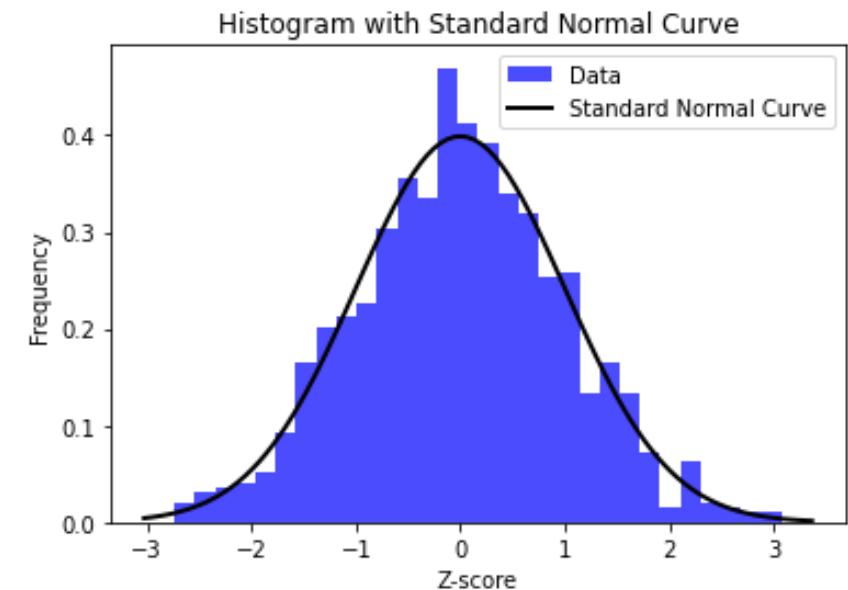
La formule pour calculer le Z-score d'un point de données 'x' dans un ensemble de données avec une moyenne ' μ ' et un écart-type ' σ '.

Formule Échantillon : $Z = (x - \bar{X}) / s$

utilisée lorsqu'on travaille avec un échantillon plutôt qu'une population. Où \bar{X} est la moyenne de l'échantillon et 's' l'écart-type de l'échantillon.

Interprétation :

Un Z-score de 0 signifie que le point de données est à la moyenne, les Z-scores **positifs** et **négatifs** indiquent une position **au-dessus** ou **en dessous** de la moyenne. Ils **identifient les valeurs aberrantes**, permettent la **comparaison entre distributions** et **définissent la position** relative des données.



Techniques statistiques avancées : Z-Score Examples de compréhension

Exemple de Calcul dans un Échantillon :

Supposons un échantillon de données $X=\{4,5,7,,\}$

Avec une moyenne d'échantillon $\bar{X} = 5$

Un écart-type d'échantillon $s=1$.

Pour calculer le Z-score du point de données 7, utilisez la formule :

$$Z=(7 - 5) / 1 = 2$$

Les étapes restent les mêmes, mais la formule est ajustée pour refléter l'utilisation des statistiques d'échantillon.



Rappel :

Un échantillon est un ensemble d'individus représentatifs d'une population.

01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers



Techniques statistiques avancées : Analyse des Z-scores

Explorer la Variabilité des Données :

Les Z-scores indiquent de combien d'écart-types un point de données s'éloigne de la moyenne.

En utilisant la bibliothèque NumPy de Python, on peut générer des données aléatoires suivant une distribution normale avec une moyenne de 50 et un écart-type de 10. Ensuite, les Z-scores pour chaque point de données sont calculés en utilisant la formule **(données - moyenne) / écart-type**.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

# Generate random data (replace this with your own dataset)
data = np.random.normal(loc=50, scale=10, size=1000)
# Calculate Z-scores
z_scores = (data - np.mean(data)) / np.std(data)
```

Techniques statistiques avancées : Analyse des Z-scores

Explorer la Variabilité des Données :

Deux histogrammes ont été créés pour visualiser les données.

L'un montre la distribution des valeurs initiales sur l'axe des x,

L'autre présente la distribution des Z-scores sur l'axe des x..

Interprétation :

L'histogramme des Z-scores offre un aperçu de la variation des points de données par rapport à la moyenne.

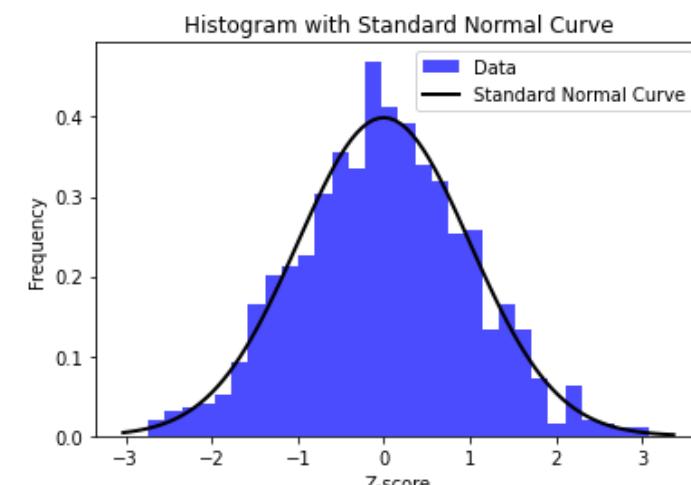
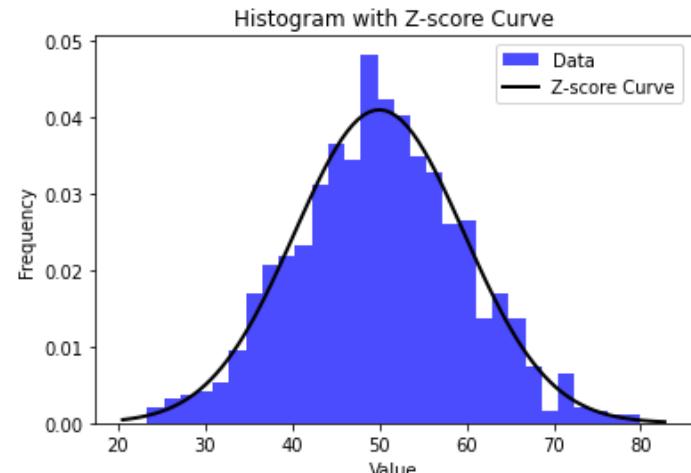
Par exemple, un Z-score de 2 indique une déviation de 20 par rapport à la moyenne.

L'application d'un seuil de 3 ou 4 pourrait être utilisée pour identifier et exclure les valeurs aberrantes qui s'éloignent significativement de la moyenne dans cette distribution normale.

Remarques :



Le Z-score diffère des valeurs brutes, avec une variation toujours à une échelle plus petite. On peut également utiliser le terme standardiser pour décrire ce processus, suggérant parfois le remplacement de certaines valeurs par leur version standardisée.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

La Standardisation et la Normalisation : Différences Théoriques et Utilisations Pratiques

Standardisation :

Théorie :

But : Transformer les données pour avoir une moyenne de 0 et un écart-type de 1.

Effet : Les données standardisées ont une variance unitaire et peuvent prendre n'importe quelle valeur. (positive et négatif)

Utilisations Pratiques :

1. Détection des Valeurs Aberrantes : Utilisation des z-scores pour identifier les outliers (valeurs au-delà de ± 3).

2. Algorithmes de Machine Learning :

Régression Linéaire et Logistique : Meilleure performance avec des données standardisées.

Analyse en Composantes Principales (ACP) : Requiert des caractéristiques sur la même échelle.

Clustering K-Means : Fonctionne mieux avec des caractéristiques standardisées pour éviter la domination de certaines caractéristiques.

- Après l'application de la **Standardisation** sur vos données normales, votre distribution sera comme ça :



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

La Standardisation et la Normalisation : Différences Théoriques et Utilisations Pratiques

Normalisation :

Théorie :

But : Échelonner les données pour qu'elles s'inscrivent dans une plage spécifique, généralement [0, 1].

Effet : Les données normalisées sont limitées à une plage fixe sans changer la forme de la distribution.

Utilisations Pratiques :

Réseaux de Neurones :Convergence plus rapide avec des entrées normalisées.

K-Nearest Neighbors (KNN) :Distance calculée de manière équitable entre les caractéristiques.

Visualisation des Données :Améliore l'interprétabilité des graphiques en maintenant une plage cohérente.

- Après l'application de la **Normalisation** vos données ressembleront à ceci :

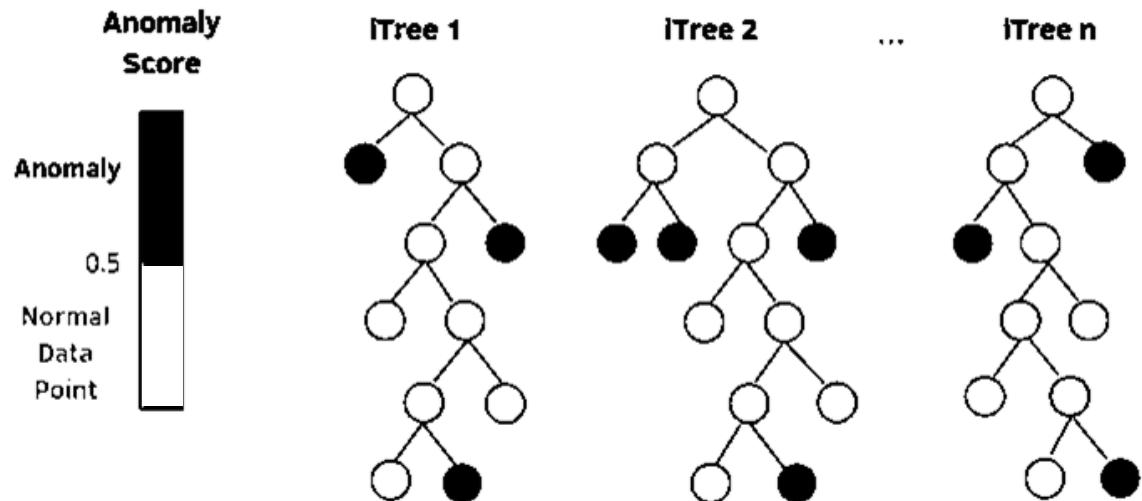


Identification par les techniques statistiques avancées : Isolation Forest (Intrusion detection algorithm)

Isolation Forest est un algorithme de ML détecte les anomalies à l'aide d'arbres binaires. L'algorithme a une complexité temporelle linéaire (À mesure que les données augmentent, le temps de détection des anomalies avec Isolation Forest augmente linéairement) et une faible exigence en mémoire, adaptée aux données de grand volume. Il détecte les outliers en mesurant la facilité avec laquelle chaque point de données peut être isolé dans un arbre de décision.

Fonctionnement de l'Isolation Forest :

L'algorithme fonctionne en isolant rapidement les anomalies dans des arbres de décision. Les anomalies nécessitent moins de divisions pour être isolées, car elles ont des caractéristiques distinctes qui les rendent plus faciles à séparer des valeurs normales.



01 – Maitriser les mesures de dispersion et de position

Analyse des Outliers

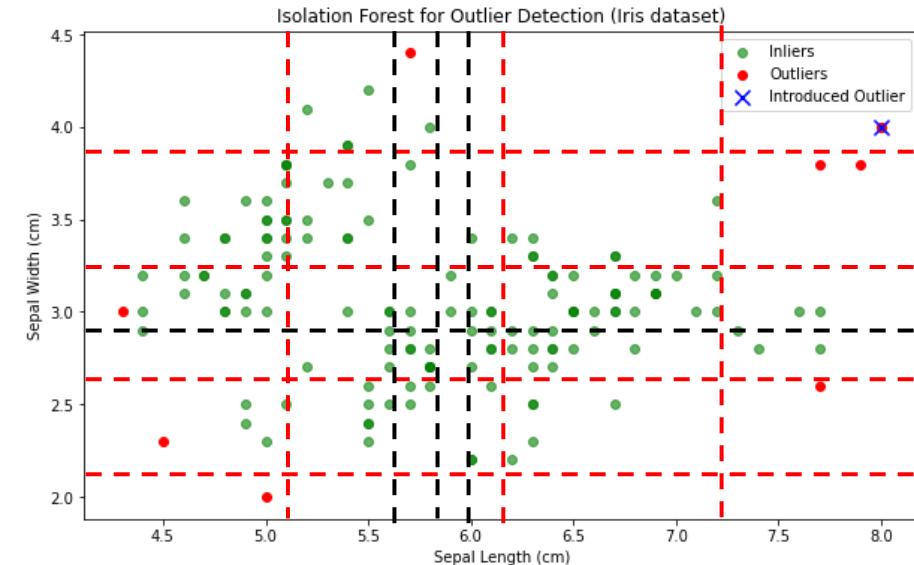
Identification par les techniques statistiques avancées : Isolation Forest (Intrusion detection algorithm)

Isolation Forest est un algorithme de ML basé sur des arbres de décision qui isole les anomalies plus efficacement que les méthodes traditionnelles. Il détecte les outliers en mesurant la facilité avec laquelle chaque point de données peut être isolé dans un arbre de décision.

Un Outlier sera détectée en nécessitant moins d'isolations que les valeurs typiques.

Comparé à d'autres méthodes, l'Isolation Forest est souvent plus rapide et nécessite moins de données pour détecter les anomalies.

Il est particulièrement adapté aux ensembles de données de grande dimension.



Introduced Outlier: ajouté intentionnellement pour tester la détection d'anomalies

Remarques :



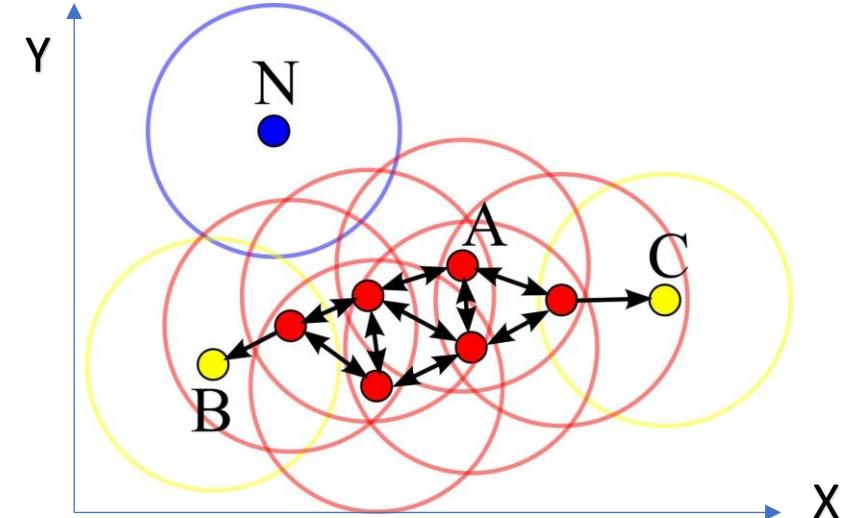
Il est essentiel de comprendre les caractéristiques spécifiques de votre ensemble de données et de comparer différentes méthodes en fonction de vos besoins avant de choisir l'approche de détection des anomalies la plus

Techniques statistiques avancées : DBSCAN Algorithm

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

est un algorithme de clustering basé sur la densité. Il identifie les zones denses de points de données en se basant sur le nombre de points voisins « Neighbors » dans une région donnée. Les points qui ne sont pas atteints par suffisamment de voisins sont considérés comme du bruit. DBSCAN est efficace pour détecter des groupes de formes arbitraires et s'adapte bien à différentes densités locales dans l'ensemble de données.

Remarques :



A : Core points (points centraux)
B, C : Density-connected (connectés par densité)
N : Noise (bruit) => **Outliere**

DBSCAN est utilisé dans des situations complexes où plusieurs colonnes ou dimensions sont prises en compte simultanément pour construire de nombreux clusters. Cette approche diffère des méthodes précédentes qui détectent principalement les valeurs aberrantes en se basant sur les données d'une seule dimension.

DBSCAN est un algorithme de regroupement dont le but principal est d'identifier les régions denses dans les données. Bien qu'il puisse être utilisé pour détecter les valeurs aberrantes (Outliers).

Gestion et traitement des Outliers : Consielles

- Gérer les outliers est essentiel pour garantir la précision des analyses de données. Pour ce faire, diverses méthodes peuvent être utilisées, telles que la règle de IQR, les algorithmes DBSCAN et Isolation Forest.
- Les outliers ont une signification statistique, pouvant signaler des erreurs ou des événements spéciaux dans les données. Leur présence peut influer considérablement sur les résultats et les conclusions tirées des analyses.
- Les Z-scores sont souvent utilisés pour identifier ces valeurs aberrantes.
- Enfin, la gestion des outliers est un processus continu qui nécessite une surveillance constante pour maintenir la qualité des analyses et des résultats.

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Pour cette illustration de DBSCAN, voici quelques points importants :

1. Paramètres Importants :

eps (epsilon) : Rayon maximal pour qu'un point soit considéré comme voisin.

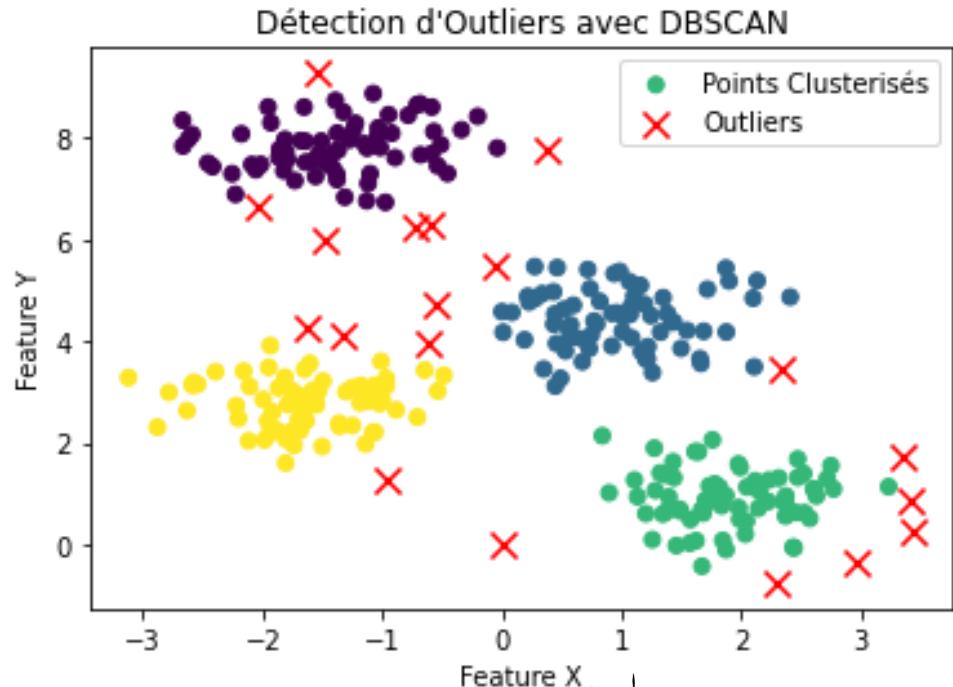
min_samples : Nombre minimal de points dans un cluster.

2. Commandes Essentielles en Python :

- `DBSCAN(eps=0.5, min_samples=5)` # Initialisation de l'algorithme DBSCAN
- `fit_predict(data)` # Prédire les clusters
- `clustered_data = data [labels != -1]` # Séparation des données clusterisées
- `outlier_data = data [labels == -1]` # Séparation des outliers fonction des labels

3. Visualisation avec Matplotlib :

- Scatter plot avec des points clusterisés colorés selon les clusters identifiés.
- Outliers marqués en rouge avec le symbole 'x'.
- Affichage du titre et des axes.



Remarques :

Des implémentations de DBSCAN peuvent être trouvées dans scikit-learn (R et Python).

Importation : `from sklearn.cluster import DBSCAN`
`epsilon ou eps` : le rayon autour de chaque point

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Solution :

Importer les bibliothèques et générer une distribution de données :



Remarques :



En cas de difficulté de compréhension de certains éléments. Il est recommandé de revenir à cette application après avoir avancé dans le cours, notamment sur la génération de distributions et la visualisation par scatter plot.
Une autre application de DBSCAN sur des données réelles (tips dataset) est fournie sur la partie TP

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import DBSCAN
from sklearn.datasets import make_blobs

# Générer des données de test avec quelques clusters et outliers
data, _ = make_blobs(n_samples=300, centers=4, cluster_std=0.60, random_state=0)
outlier = np.array([[0, 0]])
data_with_outlier = np.concatenate([data, outlier])
```

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python



Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

[Plus sur la génération de distribution de données avec make_blobs :](#)

make_blobs est une fonction de la bibliothèque sklearn.datasets en Python, utilisée pour générer des ensembles de données synthétiques pour des exemples et des tests. Plus précisément, elle crée des blobs de points de données, ce qui est utile pour les démonstrations de techniques de clustering ou d'autres algorithmes de machine Learning.

Voici une brève description de ses principaux paramètres :

n_samples : Nombre total de points générés. Vous pouvez également passer une liste pour spécifier le nombre de points pour chaque

cluster.centers : Nombre de centres de clusters à générer, ou les coordonnées des centres de clusters.

n_features : Nombre de caractéristiques pour chaque point.

cluster_std : L'écart type des clusters.

center_box : Les limites des valeurs pour les centres des clusters.

shuffle : Indique si les échantillons doivent être mélangés.

random_state : Contrôle le générateur de nombres aléatoires pour la reproductibilité des résultats.

Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Solution : suit

C'est notre section dédiée de l'application pour utiliser DBSCAN dans la détection des valeurs aberrantes. Il suffit d'appliquer cette méthode pour identifier les outliers :

Application de DBSCAN : Le modèle DBSCAN est configuré avec des paramètres spécifiques (`eps=0.5` et `min_samples=5`) et appliqué aux données contenant des outliers.

Appliquer DBSCAN avec des paramètres spécifiques

```
dbscan = DBSCAN(eps=0.5, min_samples=5)  
labels = dbscan.fit_predict(data_with_outlier)
```

Séparer les données clusterisées et les outliers

```
clustered_data = data_with_outlier[labels != -1]  
outlier_data = data_with_outlier[labels == -1]
```

Séparation des données : Les données sont ensuite divisées en deux groupes : les données clusterisées (celles qui ne sont pas des outliers) et les outliers (données avec le label -1).

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python

Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Solution de la Visualisation avec Matplotlib :

Signifie que la colormap "viridis" est utilisée pour colorer les points clusterisés.

Spécifie que les couleurs des points sont déterminées par les labels des clusters, en excluant les outliers (qui ont un label de -1)

spécifient les coordonnées des points sur les axes X et Y respectivement.

```
# Afficher les résultats avec un scatter plot
plt.scatter(clustered_data[:, 0], clustered_data[:, 1], c=labels[labels != -1], cmap='viridis',
label='Points Clusterisés')
plt.scatter(outlier_data[:, 0], outlier_data[:, 1], c='red', marker='x', s=100, label='Outliers')

plt.title('Détection d\'Outliers avec DBSCAN')
plt.xlabel('Feature X')
plt.ylabel('Feature Y')
plt.legend()
plt.show()
```



La fonction plt.scatter() attend les coordonnées X et Y comme ses premiers deux arguments positionnels, et non comme des arguments nommés X et Y.

Ce script visualise la séparation des points en clusters et les outliers détectés par l'algorithme DBSCAN, en utilisant deux caractéristiques des données pour les axes X et Y.

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Python



Identification par les techniques statistiques avancées : DBSCAN multi-clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Description des éléments de visualisation :

Voici une explication des différents éléments du graphique :

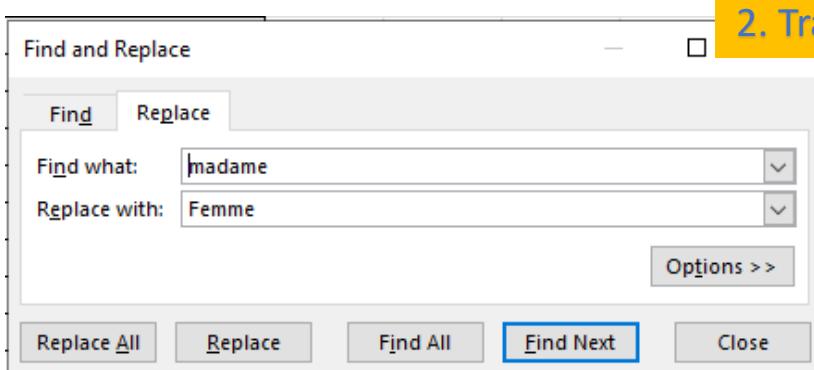
- **Points Clusterisés (Scatter Plot Vert)** : Les points sont affichés à l'aide d'un scatter plot, où chaque point représente une observation dans les données. Les couleurs des points (obtenues à l'aide de la palette de couleurs 'viridis') indiquent les différents clusters auxquels ces points appartiennent. Seuls les points appartenant à des clusters sont colorés.
- **Outliers (Scatter Plot Rouge)** : Les outliers, ou points aberrants, sont affichés en rouge avec un marqueur 'x'. Ces points ont été identifiés par l'algorithme DBSCAN comme ne faisant partie d'aucun cluster (label -1).
- **Feature X** : Représente la première caractéristique ou dimension des données.
- **Feature Y** : Représente la deuxième caractéristique ou dimension des données.
- **Titre du Graphique** : "Détection d'Outliers avec DBSCAN", indique que le graphique illustre les résultats de la détection d'outliers utilisant l'algorithme DBSCAN.
- **Légende** : La légende identifie les différents groupes de points :"Points Clusterisés" : Points appartenant à des clusters."Outliers" : Points considérés comme des outliers.

01 – Maitriser les mesures de dispersion et de position

Application pratique avec Excel

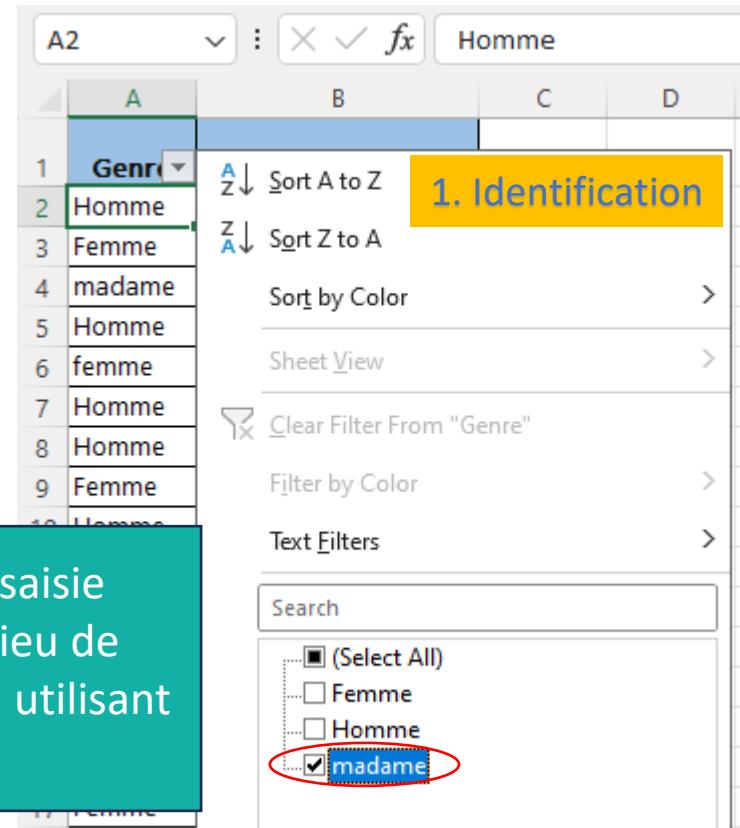
Identification et traitement des Outliers : Visualisation et traitement utilisent Excel

Malgré l'accès important aux outils statistiques et analytiques, dans de nombreuses situations professionnelles, utiliser des solutions simples et directes comme le Filtre d'Excel pour repérer les valeurs aberrantes et les traiter peut être une méthode valide et efficace, surtout avec l'existence d'un petit ensemble de données collectées directement auprès des utilisateurs, où une formule ou une feuille Excel.



2. Traitement

Identifier une faute de saisie (comme "madame" au lieu de "Femme") et la corriger en utilisant **Replace**



A2 : X ✓ f Homme

1. Identification

Sort A to Z

Sort Z to A

Sort by Color

Sheet View

Clear Filter From "Genre"

Filter by Color

Text Filters

Search

(Select All)

Femme

Homme

madame

Ce type d'erreurs souvent lors du traitement de données provenant de différentes sources ou de Big Data. Normalement, de telles erreurs doivent être corrigées au cours du processus ETL.



CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

Ce que vous allez apprendre dans ce chapitre :

- Explorer les notions avancées de probabilité
- Rappeler les distributions discrètes et continues
- Comprendre les fonctions de densité et de distribution
- Appliquer pratiquement avec Python/Excel

 10 heures



CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

1. Notions avancées de probabilité
2. Rappel des distributions discrètes et continues
3. Fonctions de densité et de distribution
4. Simulation de distributions
5. Application pratique avec Python/Excel

03 – CHAPITRE 3

Notions avancées de probabilité

Introduction

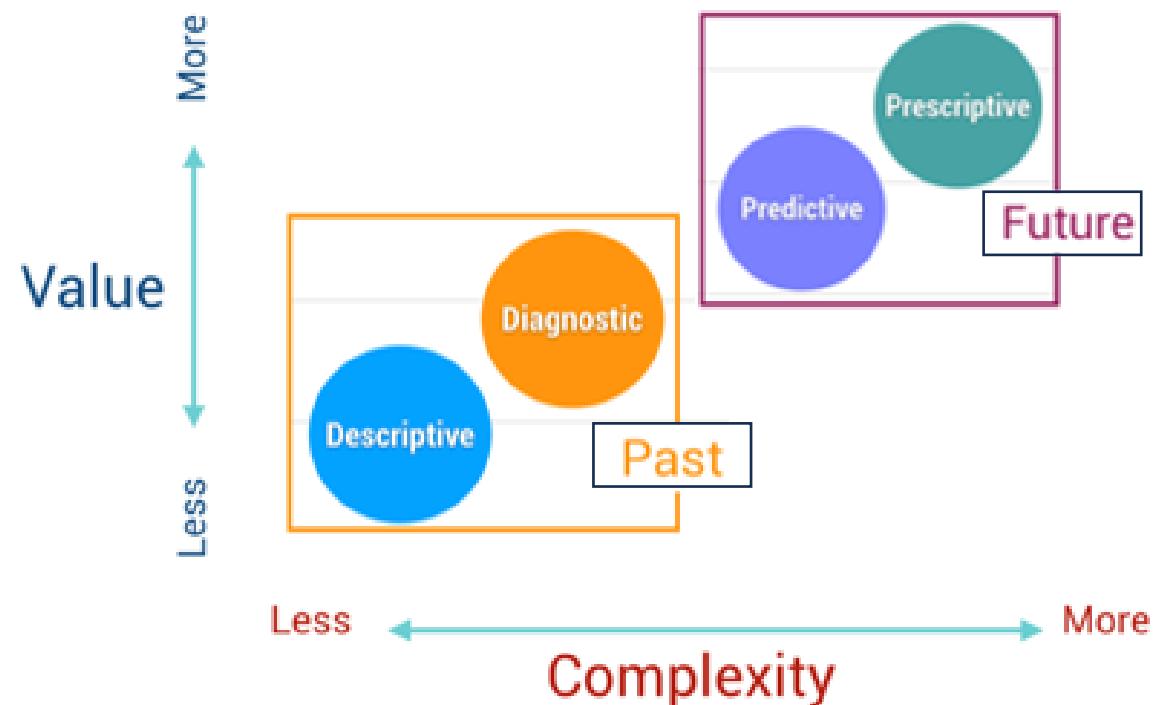
L'analyse de données fournit des informations sur les performances passées, présentes et futures de l'entreprise, aidant ainsi les entreprises à prendre des décisions éclairées et à améliorer les opérations globales. Nous présentons les quatre types d'analyse :

- **Descriptive Analytics:** Révèle "**What happened**" en interprétant les données historiques, fournissant un aperçu des performances de l'entreprise.
- **Diagnostic Analytics:** Adresse "**why things happened**" en identifiant des schémas et des corrélations dans les données, permettant des insights plus approfondis sur le comportement.
- **Predictive Analytics:** Prédit "**what is likely to happen**" à l'avenir en utilisant des données existantes et des techniques de modélisation statistique et **utilise la probabilité pour anticiper les résultats futurs**, aidant à la prise de décision et à la planification.
- **Prescriptive Analytics:** Pilotée par des systèmes d'IA, l'analyse prescriptive conseille "**what should be done next**" pour optimiser les résultats, bien que cela nécessite une expertise et des ressources qualifiées

Bayesian predictive analysis involves estimating probabilities of future events or outcomes based on prior knowledge and observed data.

Introduction

Chaque type d'analyse utilisé dans différentes entreprises a un coût spécifique basé sur sa complexité et ajoute de la valeur à l'entreprise. Les entreprises devraient commencer par analyser le passé pour progresser vers l'avenir.



Probabilité conditionnelle

La probabilité avancée est essentielle pour la compréhension approfondie des données et la prise de décision dans divers domaines. Nous explorerons également la manière dont ces concepts peuvent être appliqués en pratique en utilisant Python

Définition et formule de La probabilité conditionnelle :

- **La probabilité conditionnelle** est la probabilité qu'un événement A se produise, sachant que l'événement B s'est déjà produit. Elle est représentée par $P(A|B)$ et peut être interprétée comme la probabilité de A sous la condition que B se soit produit.
- La formule mathématique de la probabilité conditionnelle est donnée par:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Exemples et cas d'utilisation dans l'analyse de données :

- Probabilité de pluie  sachant que le ciel est couvert .
- Probabilité de défaut d'un produit  sachant qu'il a été fabriqué sur une machine spécifique .

Théorème de Bayes

Le théorème de Bayes est un outil fondamental dans la théorie des probabilités, utilisé pour mettre à jour nos croyances sur un événement en tenant compte de nouvelles preuves ou informations.

Définition et formule du théorème de Bayes :

- Le **théorème de Bayes** décrit la probabilité d'un événement, en se basant sur la connaissance préalable des conditions **qui pourraient être liées à cet événement**.
- La formule du théorème de Bayes est donnée par:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \text{ avec } P(B) > 0$$

Cette formule exprime la probabilité conditionnelle de A sachant B en termes de la probabilité conditionnelle de B sachant A, multipliée par la probabilité marginale de A, divisée par la probabilité marginale de B.

Application dans les problèmes de classification et de diagnostic:

- Utilisation dans les problèmes de classification, tels que la classification des e-mails  en spam ou non spam.
- Application dans les problèmes de diagnostic médical pour estimer la probabilité d'une maladie  sachant certains symptômes .

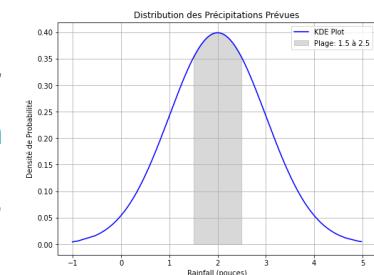
03 – CHAPITRE 3

Notions avancées de probabilité

Théorème de Bayes : Modèles probabilistes

Dans notre étude générale des modèles probabilistes « Bayésiens », nous avons examiné trois variantes principales du classificateur Naive Bayes et leurs domaines d'application spécifiques :

- **MultinomialNB** : Ce modèle est largement utilisé dans les tâches de classification de texte, telles que la détection de spam, la classification de documents et l'analyse des sentiments. **Il est adapté pour traiter des fonctionnalités qui représentent des nombres ou des fréquences, ce qui est commun dans les données textuelles où les mots sont comptés ou leur fréquence est mesurée.**
- **BernoulliNB** : Contrairement à MultinomialNB, BernoulliNB est plus adapté lorsque les caractéristiques sont binaires, c'est-à-dire qu'elles prennent seulement deux valeurs possibles, généralement 0 ou 1. **Il est couramment utilisé dans la classification de documents où chaque document est représenté par la présence ou l'absence de certains mots.**
- **GaussianNB** : Ce modèle est utilisé lorsque les caractéristiques sont continuées et suivent une distribution normale, également appelée distribution gaussienne. **Il est souvent utilisé dans des domaines tels que la classification biomédicale ou la reconnaissance de caractères, où les caractéristiques sont des mesures continues telles que la taille, le poids, la température, etc.**



Une implémentation Python de la détection du spam à l'aide d'un modèle Bayésien est présentée dans l'activité 3 du TP.



CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

1. Notions avancées de probabilité
2. **Rappel des distributions discrètes et continues**
3. Fonctions de densité et de distribution
4. Simulation de distributions
5. Application pratique avec Python/Excel

02 – CHAPITRE 3

Rappel des distributions discrètes et continues

Distributions discrètes

Les distributions discrètes en probabilité décrivent des variables aléatoires qui prennent des valeurs distinctes et finies. Cela signifie que les valeurs possibles sont des nombres entiers ou une liste de valeurs distinctes.

Un exemple courant de distribution discrète uniforme est la distribution de probabilité de jeter un dé. Lorsque vous lancez un dé à six faces, les valeurs possibles sont les nombres entiers de 1 à 6. Chaque valeur a une probabilité associée, où chaque face à une probabilité égale de $1/6$ si le dé est équitable.

Un autre exemple est la distribution de **probabilité binomiale**, qui modélise le nombre de succès dans une séquence de tentatives indépendantes, chaque tentative ayant seulement deux issues possibles (succès ou échec). Par exemple, le nombre de têtes obtenues en lançant une pièce de monnaie équitable plusieurs fois suit une distribution binomiale.



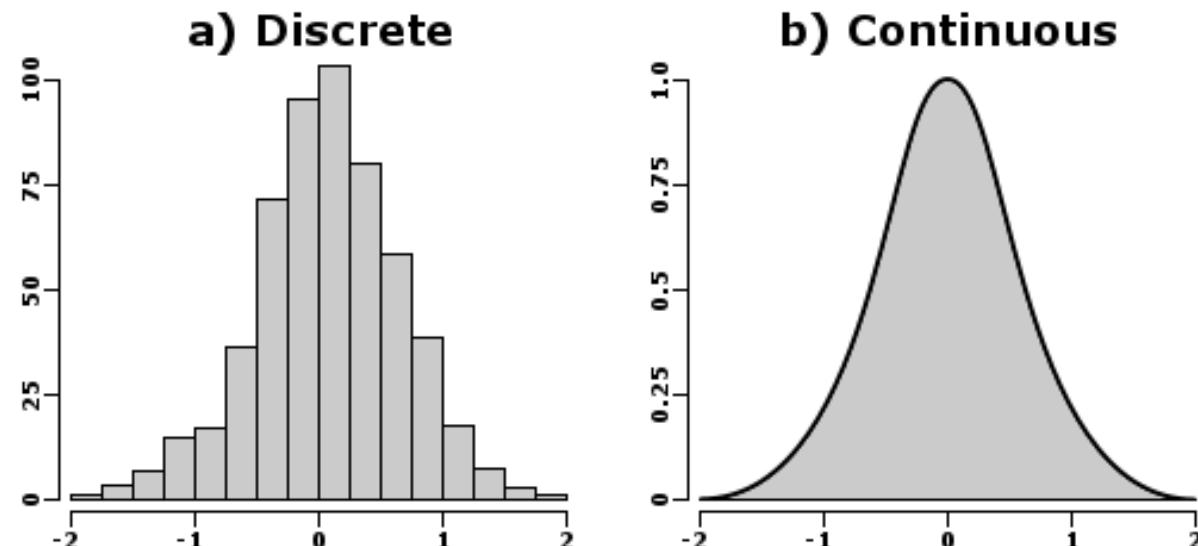
Astuce : La somme de toutes les issues possibles d'un événement doit être égale à 1.

Distributions continues

Les distributions continues en probabilité décrivent des variables aléatoires qui peuvent prendre n'importe quelle valeur dans un intervalle donné.

Contrairement aux distributions discrètes, les valeurs possibles sont infiniment nombreuses et souvent représentées par des nombres réels.

Un exemple courant de distribution continue est la distribution normale, également appelée distribution gaussienne. Elle est caractérisée par sa courbe en forme de cloche symétrique et est utilisée pour modéliser de nombreux phénomènes naturels tels que les tailles humaines, les scores de tests standardisés, la température, etc.





CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

1. Notions avancées de probabilité
2. Rappel des distributions discrètes et continues
- 3. Fonctions de densité et de distribution**
4. Simulation de distributions
5. Application pratique avec Python/Excel

Fonction de densité de probabilité : (PDF - Probability Density Function)

Les distributions de probabilités continues sont également appelées les fonctions de densité de probabilité.

Les densités de probabilité les plus courantes sont :

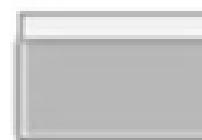
- Distribution Normale
- Distribution Binomial
- Distribution Uniform



Normal



Triangular



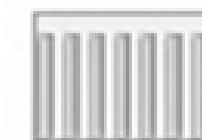
Uniform



Exponential



Binomial



Discrete Uniform



Poisson

Dans une distribution normale, gardez à l'esprit la règle des 68, 95, 99 : environ 68% des données se situent dans un écart type σ , 95 % dans $\pm 2\sigma$, et 99,7 % dans $\pm 3\sigma$.

La fonction de densité de probabilité décrit la probabilité qu'une variable aléatoire prenne une valeur spécifique.

- Pour une variable continue, elle représente la densité de probabilité à chaque valeur possible.
- Pour une variable discrète, elle donne la probabilité de chaque valeur discrète.

03 – CHAPITRE 3

Fonctions de densité et de distribution

Fonction de distribution cumulative : (CDF - Cumulative Distribution Function)

Problématique et motivations pour l'utilisation de CDF :

Gérer les distributions continues en probabilité présente des défis.

Par exemple, lors de la prévision de la quantité exacte de pluie demain, une prévision précise peut rencontrer des limites en raison de la nature continue des mesures de précipitations.

Dans de tels cas, prédire une valeur précise, telle que exactement 2 pouces de pluie, peut entraîner des inexactitudes, car même une légère déviation de cette valeur pourrait rendre la prévision incorrecte. (2,00001)

Par conséquent, il est plus judicieux d'évaluer la probabilité d'une plage de valeurs, telle que entre 1,5 et 2,5 pouces, en utilisant des techniques telles que les fonctions de distribution cumulée en analyse de données pour des prédictions plus robustes et précises.

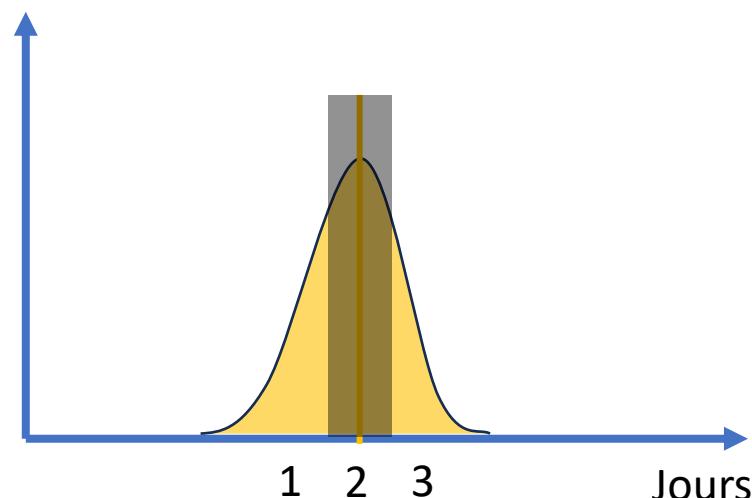


Fonction de distribution cumulative : (CDF - Cumulative Distribution Function)

La fonction de distribution cumulative donne la probabilité qu'une variable aléatoire soit inférieure ou égale à une valeur spécifique.

Elle représente la probabilité cumulée jusqu'à cette valeur. En d'autres termes, la CDF donne la probabilité que la variable aléatoire prenne une valeur inférieure ou égale à un certain seuil.

Ces deux fonctions sont fondamentales en statistiques et en probabilité, et sont utilisées pour analyser et modéliser différents types de variables aléatoires.



When dealing with continuous data, it's often more practical to determine the probability of outcomes falling within certain bounds, rather than predicting a specific value.

Range : less than 2,5

Range : greater than 1,5

Range : Between 1,5 and 2,5



CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

1. Notions avancées de probabilité
2. Rappel des distributions discrètes et continues
3. Fonctions de densité et de distribution
- 4. Simulation de distributions**
5. Application pratique avec Python/Excel

Monte Carlo Simulation :

Motivation :

"Biased data selection can significantly damage the analysis of a data distribution, leading to conclusions that diverge greatly from reality."

Comprendre les Simulations Monte Carlo :

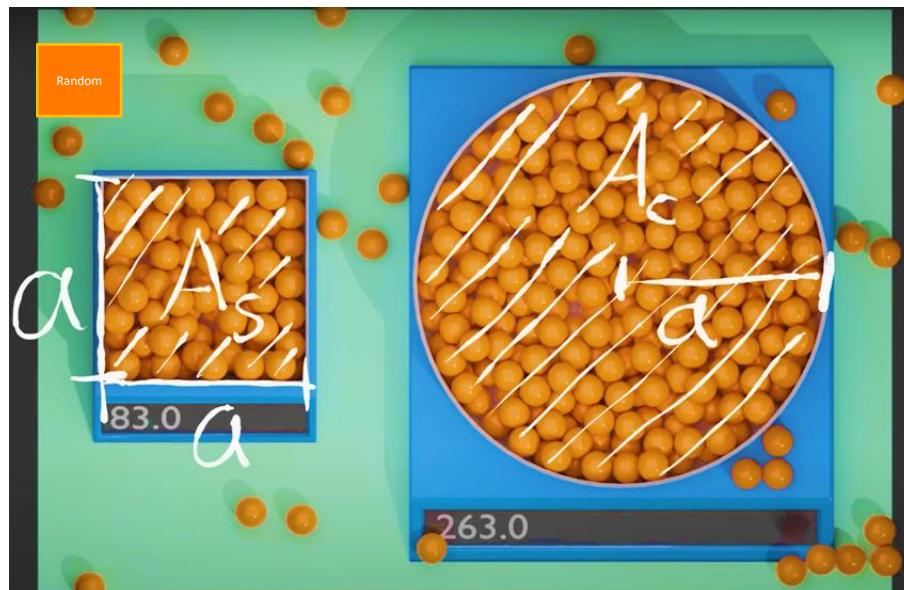
- Les simulations Monte Carlo tirent leur nom de la ville de Monte Carlo, synonyme de hasard en raison de son association avec les casinos et les jeux de hasard.
- Dans les simulations, "Monte Carlo" signifie aléatoire, semblable à l'incertitude dans les jeux de hasard.
- Les simulations Monte Carlo reposent sur l'obtention d'échantillons aléatoires pour inférer les propriétés d'un système.
- Les échantillons aléatoires aident à approximer des calculs complexes en explorant un sous-ensemble représentatif de possibilités.

Monte Carlo proof of concept : Estimation de Pi

Un exemple simple consiste à lâcher aléatoirement des billes sur une table avec deux bols : un carré et un circulaire.

En observant le rapport des billes dans le bol circulaire par rapport à celles dans le bol carré, nous pouvons estimer la valeur de pi.

La probabilité qu'une bille tombe dans chaque bol est proportionnelle à la section transversale du bol.



$$\frac{263}{83} = 3,168 \simeq \pi$$

wow mais pourquoi !!!

Car :

$$\frac{Ac}{As} = \frac{\pi * a^2}{a^2} = 3,168 \simeq \pi$$

surface du cercle

surface du carré

Monte Carlo et la loi des grands nombres :

- Ce principe est similaire aux études du monde réel, où l'échantillonnage permet d'estimer les paramètres de la population.
- La Loi des Grands Nombres stipule que lorsque la taille de l'échantillon augmente, la moyenne tend à converger vers la valeur attendue.
- De même, dans les simulations Monte Carlo, l'augmentation du nombre d'échantillons aléatoires améliore la précision de l'estimation.
- Ce principe sous-tend la fiabilité des simulations Monte Carlo dans l'approximation de systèmes complexes.

Conclusion :

- Les simulations Monte Carlo sont des simulations évoluant de manière aléatoire qui résolvent des problèmes complexes en explorant un sous-ensemble aléatoire de possibilités.
- Elles s'appuient sur la loi des grands nombres pour fournir des résultats précis avec suffisamment d'échantillons aléatoires.
- Les exemples incluent l'estimation de pi en lâchant des billes dans des bols... Ces simulations sont précieuses pour aborder les problèmes avec un nombre de possibilités trop grand pour être gérés autrement.



CHAPITRE 3

ASSIMILER LES PROBABILITÉS ET DISTRIBUTIONS AVANCÉES

1. Notions avancées de probabilité
2. Rappel des distributions discrètes et continues
3. Fonctions de densité et de distribution
4. Simulation de distributions
5. **Application pratique avec Python/Excel**

Application pratique avec Python : Simulation Monte Carlo

Imaginons que nous voulions connaître la taille moyenne mondiale, mais mesurer chaque personne est impossible. Nous pouvons donc mesurer un échantillon aléatoire, évitant les biais de sélection.

Plus l'échantillon est grand, plus la moyenne sera proche de la réalité, comme le dit la "loi des grands nombres".

Les simulations Monte Carlo fonctionnent sur le même principe : en prenant des échantillons aléatoires, elles représentent efficacement des situations complexes.



Application pratique avec Python : Simulation Monte Carlo

Pour cette application, nous partirons du principe que nous disposons les tailles de 1000 personnes comme population initiale.

Nous utiliserons la méthode de Monte Carlo pour prédire les caractéristiques de cette population en utilisant moins de données, puis nous comparerons les résultats obtenus.

Étapes de l'expérience :

1. Générer les hauteurs de 1000 personnes distribuées normalement autour de 170 cm.
2. Calculer et Afficher les statistiques de la population initiale.
3. Visualiser la distribution de la population initiale.
4. Effectuer une simulation Monte Carlo en échantillonnant aléatoirement 100 personnes (10% de population) à partir de la population initiale.
5. Calculer les statistiques de l'échantillon Monte Carlo.
6. Afficher les statistiques de l'échantillon Monte Carlo.
7. Visualiser la distribution de l'échantillon Monte Carlo.
8. Répéter les calculs, l'affichage et la visualisation pour différentes simulations Monte Carlo en échantillonnant aléatoirement (5% et 0,5%) de la population à partir de la population initiale.
9. Comparaison et synthèse des résultats.



Application pratique avec Python : Simulation Monte Carlo

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

# 1. Générer les hauteurs de 1000 personnes distribuées
# normalement autour de 170 cm
np.random.seed(42)
population_data = np.random.normal(loc=170, scale=10,
size=1000) # Moyenne de la taille : 170 cm, Écart type : 10
cm

# 2.A Calculer les statistiques de la population initiale
initial_mean = np.mean(population_data)
initial_std = np.std(population_data)

# 2.B Afficher les statistiques de la population initiale et
# de l'échantillon Monte Carlo
print("Statistiques de la population initiale:")
print("Moyenne :", initial_mean)
print("Écart type :", initial_std)
```

Sortie Attendue :

Statistiques de la population initiale:
Moyenne : 170.19332055822326
Écart type : 9.787262077473542

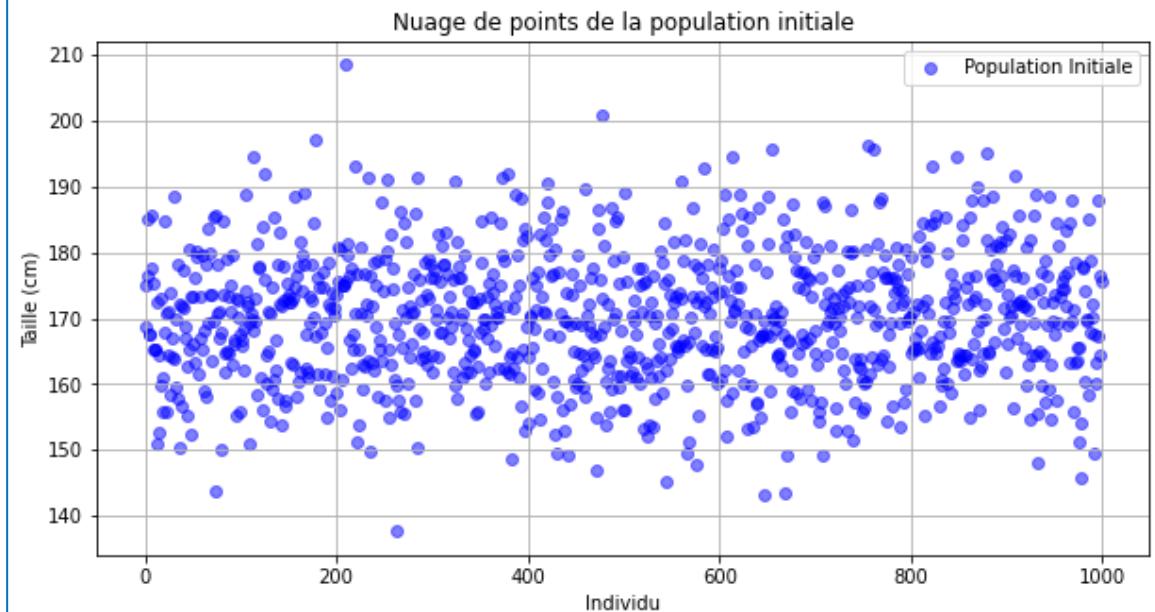


Application pratique avec Python : Simulation Monte Carlo

```
# 3. Visualiser la distribution de la population initiale avec un nuage de points
plt.figure(figsize=(10, 5))
plt.scatter(np.arange(len(population_data)),
population_data, color='blue', alpha=0.5,
label='Population Initiale')
plt.xlabel('Individu')
plt.ylabel('Taille (cm)')
plt.title('Nuage de points de la population initiale')
plt.legend()
plt.grid(True)
plt.show()
```

une explication détaillée sur la création et les types de graphiques est fournie dans la partie suivante

Sortie Attendue :



Application pratique avec Python : Simulation Monte Carlo

Création et Statistiques de la simulation : Ce code effectue une simulation Monte Carlo en sélectionnant aléatoirement 100 personnes à partir de la population initiale. Ensuite, il calcule la moyenne et l'écart type de cet échantillon Monte Carlo et les affiche.

```
# 4. Effectuer une simulation Monte Carlo en échantillonnant  
aléatoirement 100 personnes à partir de La population  
initiale
```

```
sample_size = 100  
monte_carlo_indices = np.random.choice(len(population_data),  
size=sample_size, replace=False)  
monte_carlo_sample = population_data[monte_carlo_indices]
```

```
# 5. Calculer les statistiques de L'échantillon Monte Carlo
```

```
monte_carlo_mean = np.mean(monte_carlo_sample)  
monte_carlo_std = np.std(monte_carlo_sample)
```

```
# 6. Afficher les statistiques de L'échantillon Monte Carlo
```

```
print("Statistiques de l'échantillon Monte Carlo:")  
print("Moyenne :", monte_carlo_mean)  
print("Écart type :", monte_carlo_std)
```

Sortie Attendue :

Statistiques de l'échantillon Monte Carlo:

Moyenne : 171.38470622599397

Écart type : 10.384381004252985

100 comme taille d'échantillon représente
10 % de la population

03 – CHAPITRE 3

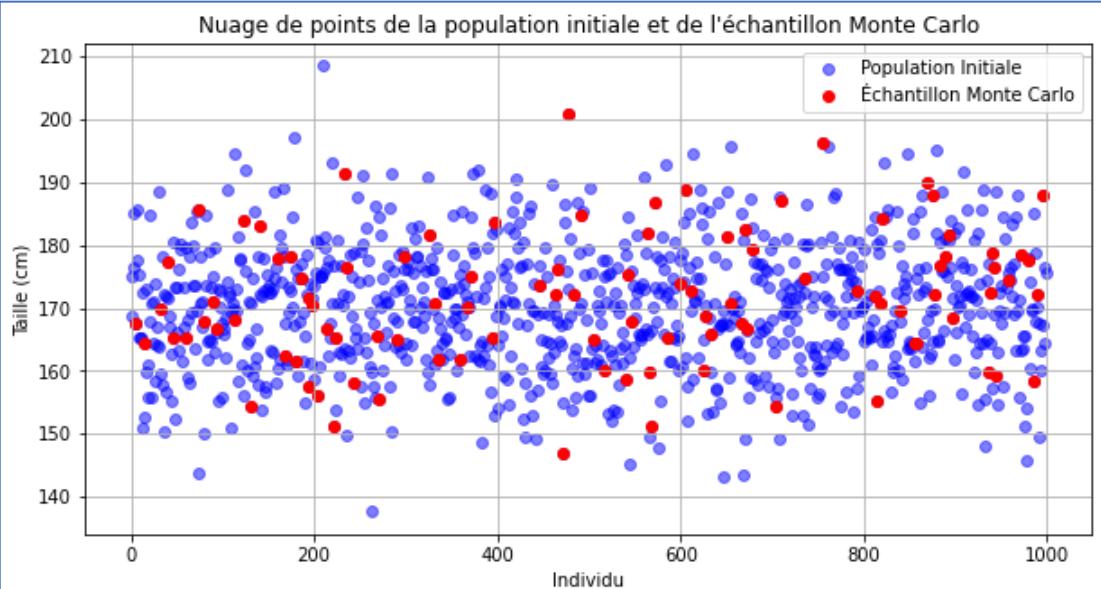
Application pratique avec Python/Excel

Application pratique avec Python : Simulation Monte Carlo

Affichage :

```
# 7. Visualiser la distribution de la population
initiale et de l'échantillon Monte Carlo ensemble
avec un nuage de points
plt.figure(figsize=(10, 5))
plt.scatter(np.arange(len(population_data)),
population_data, color='blue', alpha=0.5,
label='Population Initiale')
plt.scatter(monte_carlo_indices, monte_carlo_sample,
color='red', label='Échantillon Monte Carlo')
plt.xlabel('Individu')
plt.ylabel('Taille (cm)')
plt.title('Nuage de points de la population initiale
et de l\'échantillon Monte Carlo')
plt.legend()
plt.grid(True)
plt.show()
```

Sortie Attendue :



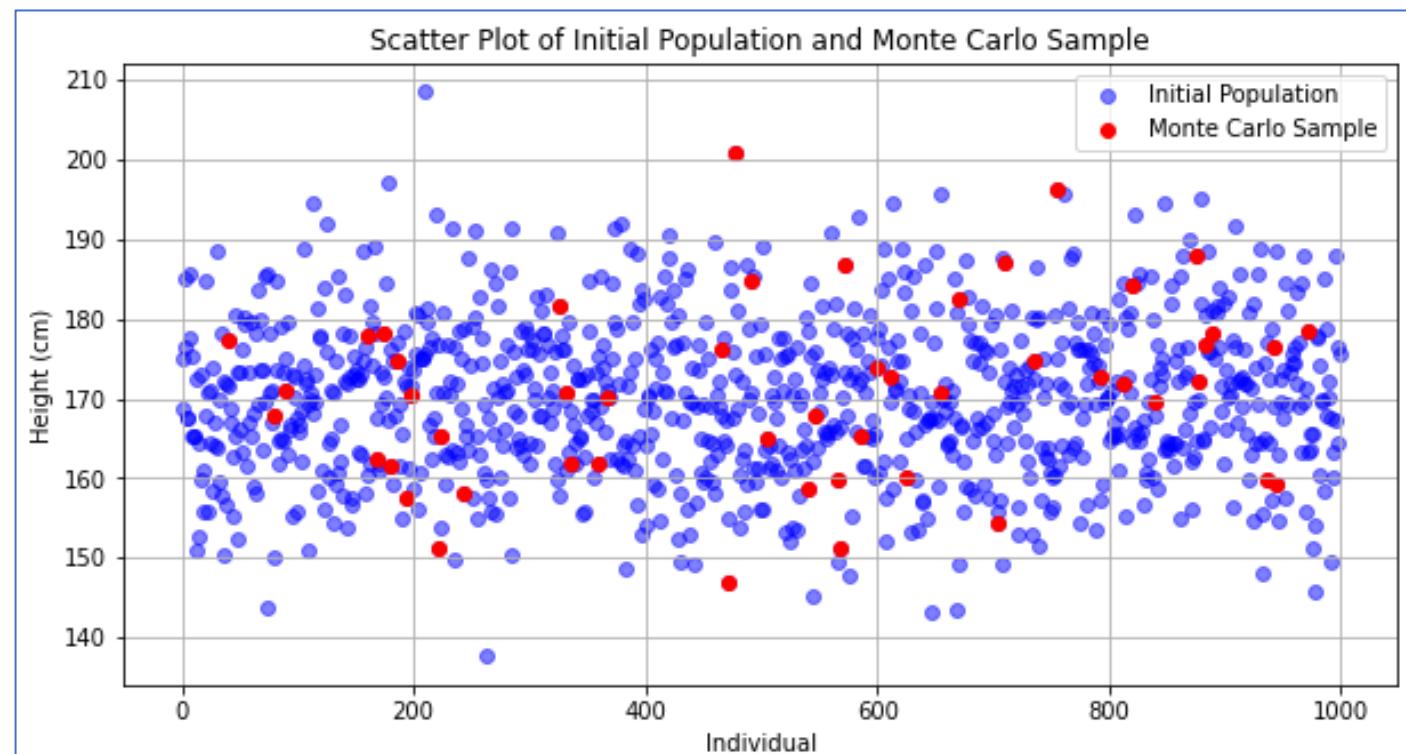
10 % de la population

Application pratique avec Python : Simulation Monte Carlo

Affichage pour 5% :

Remarquez comment même en sélectionnant seulement 5 % des données, nous avons obtenu un échantillon représentatif. Pour obtenir ces résultats, nous avons simplement changé la taille de l'échantillon à 50, ce qui correspond à 5 % des données.

Résultats Attendue :



Monte Carlo 5% Moyenne : 170.86293396896747

03 – CHAPITRE 3

Application pratique avec Python/Excel

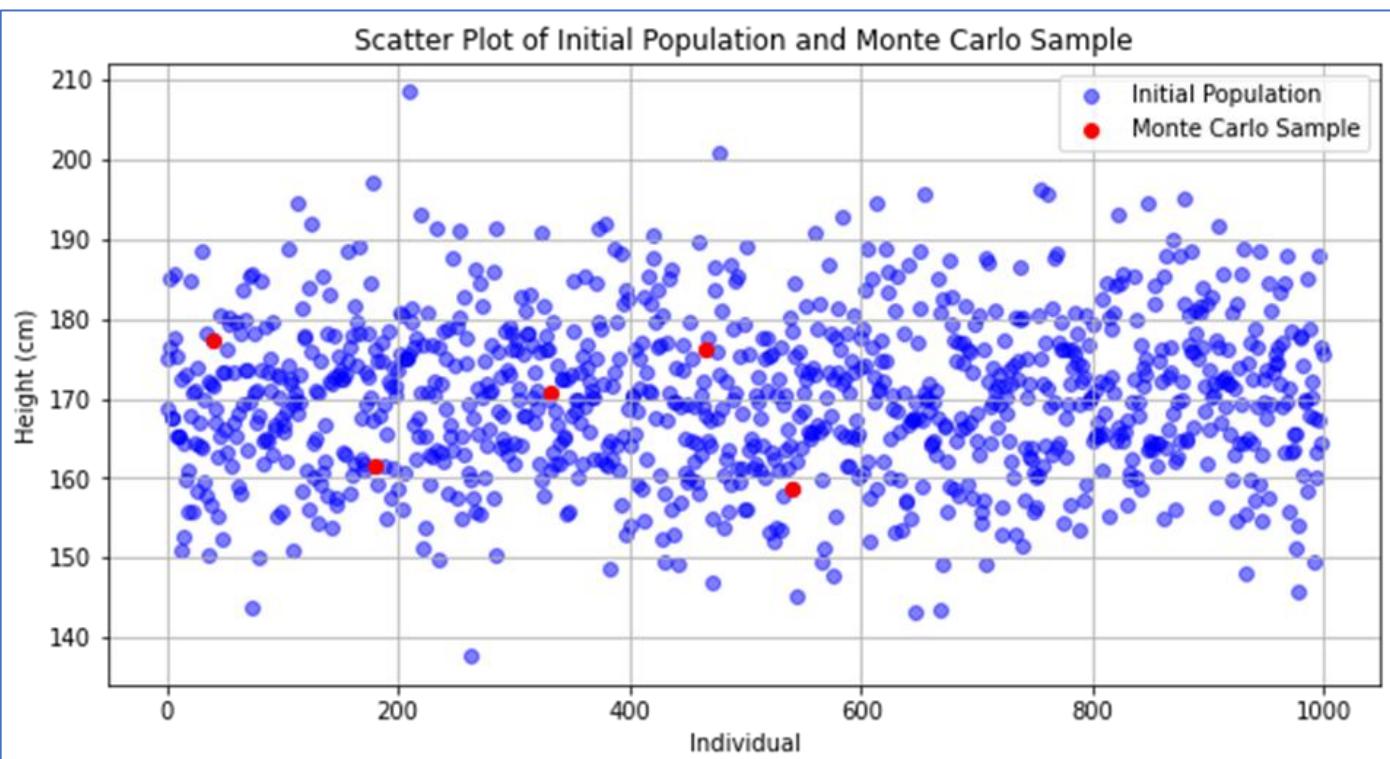
Application pratique avec Python : Simulation Monte Carlo

Affichage pour 0,5% :

Nous avons remarqué qu'avec seulement 0,5 % des données, nous avons également réussi à obtenir des données représentatives, mais avec une précision moindre dans les mesures statistiques.

Pour obtenir ces résultats, nous avons simplement changé la taille de l'échantillon à 5, ce qui correspond à 0,5 % des données.

Résultats Attendue :



Application pratique avec Python : Simulation Monte Carlo

Synthèse des résultats :

- L'expérience a démontré l'efficacité de la méthode Monte Carlo dans la prédiction des caractéristiques de la population avec différents échantillons de données.
- La comparaison des résultats montre que même avec des échantillons de seulement 5 % et 0,5 % des données, nous obtenons des données représentatives.
- La précision des mesures statistiques diminue avec la diminution de la taille de l'échantillon, soulignant l'importance d'une taille d'échantillon plus grande pour des estimations plus précises.
- La méthode Monte Carlo fonctionne particulièrement bien lorsque les données suivent une distribution normale et sont centrées, ce qui améliore sa précision.
- La plupart des données dans de nombreux domaines suivent cette loi de distribution normale, renforçant l'applicabilité et l'efficacité de la méthode de Monte Carlo dans divers scénarios d'analyse de données.

Application pratique avec Excel : Construction d'une estimation de risque probabiliste du coût des vacances au Maroc

Estimer le coût d'un voyage peut être un défi, mais notre outil simplifie ce processus en utilisant des techniques avancées pour prédire les dépenses.



Étapes de l'Application :

Catégories de Dépenses : Notre application prend en compte différentes catégories de dépenses telles que les vols, l'hébergement, la nourriture, le transport et les dépenses supplémentaires.

Simulation Monte Carlo : Nous utilisons la simulation Monte Carlo pour générer des valeurs aléatoires basées sur des distributions statistiques pour chaque catégorie de dépenses. Cela nous permet d'obtenir une gamme de résultats possibles plutôt qu'une seule estimation fixe.

Calcul du Coût Total : En combinant les valeurs aléatoires générées pour chaque catégorie de dépenses, notre application calcule le coût total estimé de votre voyage au Maroc.

Analyse des Résultats : Une fois le calcul terminé, vous obtenez une estimation du coût total de votre voyage. Vous pouvez également voir plusieurs résultats simulés pour évaluer la variabilité des coûts en fonction des différentes valeurs aléatoires générées.

03 – CHAPITRE 3

Application pratique avec Python/Excel

Application pratique avec Excel : Estimer le coût d'un voyage au Maroc

Basé sur l'expérience précédente des visiteurs et les sites de tourisme au Maroc, nous avons estimé les coûts pour les catégories suivantes : Vols, Transport, Billets d'événement, Dépenses supplémentaires, Hébergement, Repas, et l'achat d'un objet spécifique, par exemple un tapis de 300 \$.

Types de distributions : Utilisation de fonctions telles que **NORM.INV()** pour les **distributions normales** et **RANDBETWEEN()** pour les **distributions uniformes**

Tableau des estimations initiales :

| Catégorie | Distribution | Paramètres | Coût \$ | Valeur Aléatoire Calcule |
|---------------------------|-------------------|--|---------|--|
| Vols | Normale | Moyenne = \$200, Écart Type = \$50 | 211,24 | NORM.INV(RAND(); mean; stdev) |
| Transport | Uniforme | Min = \$50, Max = \$150 | 135,00 | RANDBETWEEN(min; max) |
| Billets d'événement | Normale | Moyenne = \$50, Écart Type = \$10 | 66,76 | NORM.INV(RAND(); mean; stdev) |
| Dépenses Supplémentaires | Normale | Moyenne = \$100, Écart Type = \$30 | 75,57 | NORM.INV(RAND(); mean; stdev) |
| Hébergement | Normale | Moyenne = \$75, Écart Type = \$20 | 78,70 | NORM.INV(RAND(); mean; stdev) |
| Repas | Uniforme | Min = \$10, Max = \$30 | 27,00 | RANDBETWEEN(min; max) |
| Achat d'un tapis de 300\$ | Descret Binomiale | Probability = 80% Vrais = 1 , Faux = 0 | 1 | IF(RAND() < Probability ; 1; 0) |
| Coût Total Pour 7 Jours | Multi | Somme | 1528,47 | SUM(D9:D10) * 7 + SUM(D5:D8) + D11*300 |

Utilisation d'une distribution discrète binomiale pour modéliser la probabilité d'achat d'un tapis de 300\$

La fonction RAND() génère un nombre aléatoire entre 0 et 1, tandis que NORM.INV(RAND(); mean; stdev) génère un nombre aléatoire suivant une distribution normale avec une moyenne et un écart type donnés.



Application pratique avec Excel : Estimer le coût d'un voyage au Maroc

Cette formule utilise des techniques de simulation probabiliste sur différentes distributions pour estimer le coût total des vacances au Maroc :

```
=NORM.INV(RAND(); 200; 50)  
+ RANDBETWEEN(50;150)  
+ NORM.INV(RAND();50;10)  
+ NORM.INV(RAND();100;30)  
+ ( NORM.INV(RAND();75;20)  
    + RANDBETWEEN(10;30) ) * 7  
+ IF(RAND() < 0,8; 1; 0) * 300
```

- Génère une valeur aléatoire à partir d'une distribution normale, représentant le coût des vols.
- Génère une valeur aléatoire entre 50 \$ et 150 \$, représentant le coût du transport.
- Génère une valeur aléatoire à partir d'une distribution normale, représentant le coût des billets d'événement.
- Génère une valeur aléatoire à partir d'une distribution normale, représentant les dépenses supplémentaires.
- Il génère une valeur aléatoire à partir d'une distribution normale avec une moyenne de 75 \$ et un écart type de 20 \$ pour l'hébergement, ajoute une valeur aléatoire entre 10 \$ et 30 \$ pour les repas, et multiplie la somme par 7 pour couvrir 7 jours.
- Utilise une instruction conditionnelle pour déterminer si l'acheteur achètera un tapis de 300\$. Si la valeur aléatoire générée par RAND() est inférieure à 0,8 (probabilité de 80 %), elle renvoie 1, indiquant que le tapis est acheté. Sinon, elle renvoie 0. Le résultat est multiplié par 300 pour représenter le coût du tapis s'il est acheté.

Application pratique avec Excel : Estimer le coût d'un voyage au Maroc

En effectuant de multiples simulations, nous obtenons une distribution des coûts possibles, ce qui nous permet de comprendre la plage de valeurs que le coût total pourrait prendre et d'évaluer la probabilité d'atteindre certains seuils de coût. Cela nous donne une meilleure compréhension des risques financiers potentiels associés au voyage et nous aide à prendre des décisions éclairées.

| Estimation de Risque | Coût (\$) |
|----------------------|-----------|
| Attemption 1 | 1317,39 |
| Attemption 2 | 1469,40 |
| Attemption 3 | 1045,66 |
| Attemption 4 | 1202,68 |
| Attemption 5 | 1368,24 |
| Attemption 6 | 890,27 |
| Attemption 7 | 1221,37 |
| Attemption 8 | 1059,04 |
| Attemption 9 | 1513,39 |
| Attemption 10 | 1285,04 |
| Attemption N | |

- Présentation de l'estimation de risque avec plusieurs tentatives et leurs coûts correspondants
- Chaque tentative représente une simulation de l'estimation de coût basée sur des valeurs aléatoires générées pour chaque catégorie de dépenses

03 – CHAPITRE 3

Application pratique avec Python/Excel

Application pratique avec Excel : Estimer le coût d'un voyage au Maroc

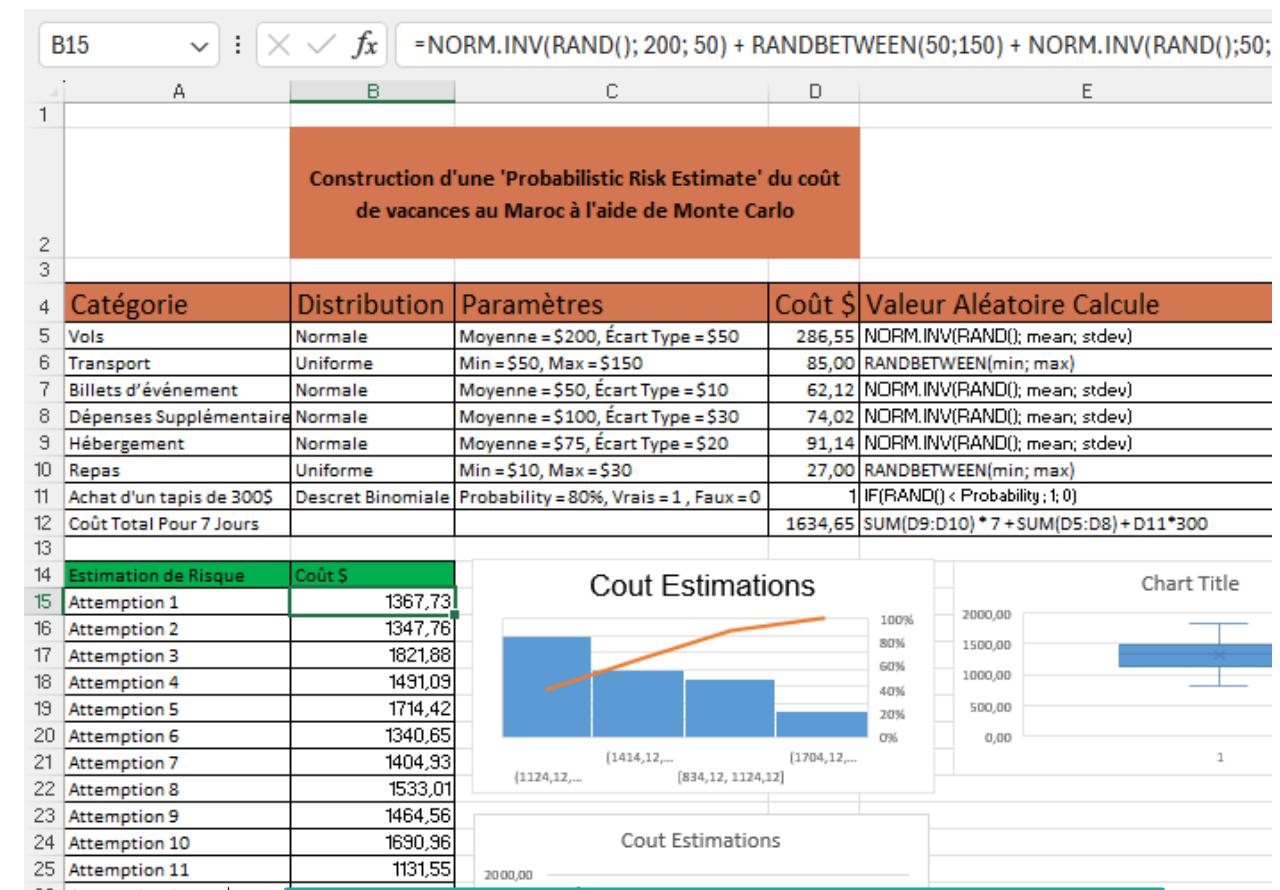
L'écran Excel affiche une synthèse complète de l'estimation des coûts de vacances au Maroc.

Les tables montrent les données pour chaque catégorie de dépenses, tandis que les graphiques illustrent la distribution des coûts et les résultats des simulations.

Cette présentation facilite la visualisation et l'analyse des résultats de manière claire et concise.

Les utilisateurs peuvent rapidement évaluer les scénarios potentiels et comprendre les risques financiers associés au voyage.

[[Lien vers le fichier Excel](#)]



Sur Excel Cliquez sur F9 pour réexécuter les simulations

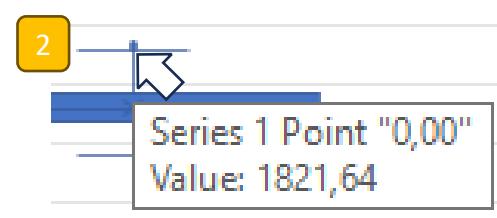
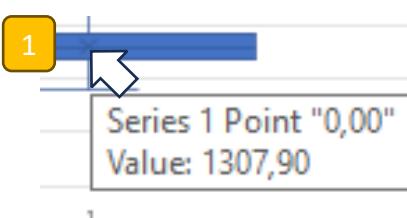
Application pratique avec Excel : Estimer le coût d'un voyage au Maroc

Analyse des Résultats :

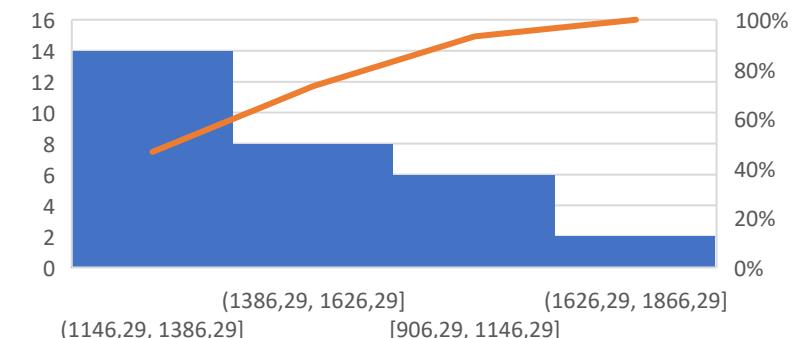
- On peut constater que la plupart des simulations se situent dans une plage moyenne de 1146 à 1626, ce qui représente environ 80 % des données.
- Étant donné que la plupart des estimations reposent sur des distributions normales, il est moins probable d'obtenir des coûts extrêmement élevés ou bas.
- Le coût moyen du voyage, basé sur les paramètres précis, devrait avoisiner 1300\$, avec un écart type de 240\$. Il est important de noter que ces résultats peuvent varier.
- En survolant le graphique en box plot :

1 Nous pouvons voir que la moyenne est de 1300 \$

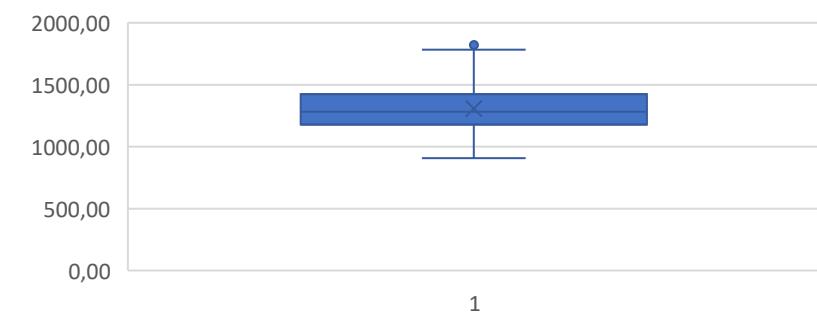
2 et, à 1821 \$. Qu'est-ce que c'est ? Oui, c'est une valeur aberrante «Outlier»



Coût Estimations



Coût Estimations en Box Plot



CHAPITRE 4

VULGARISER LE LANGAGE R



Ce que vous allez apprendre dans ce chapitre :

- Introduire R et son rôle en statistique
- Préparer l'environnement pour l'utilisation de R
- Appliquer la syntaxe de base et les principaux opérateurs en R
- Explorer les structures de données fondamentales en R
- Effectuer les premières manipulations de données en R
- Comparer les fonctions de base avec Python en R
- Adapter les dernières applications (Python/Excel) en R



15 heures



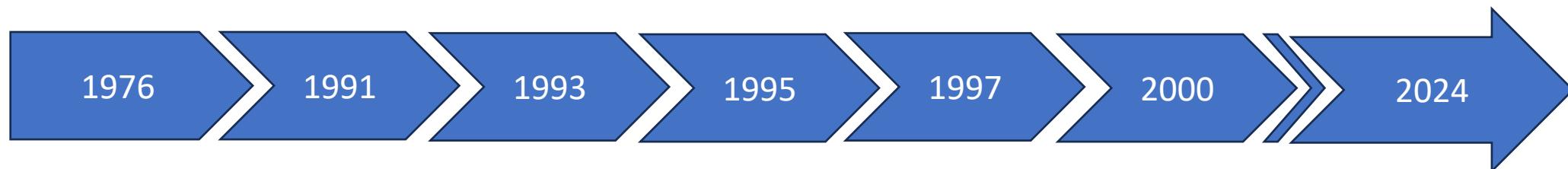
CHAPITRE 4

VULGARISER LE LANGAGE R

1. Introduction à R et son rôle en statistique
2. Préparation de l'environnement
3. Syntaxe de base et principaux opérateurs en R
4. Structures de données fondamentales en R
5. Premières manipulations de données en R
6. Comparaison des fonctions de base avec Python
7. Reprise des dernières applications (Python/Excel) en R

Introduction au langage R

- R est un langage et un environnement pour le calcul statistique et les graphiques.
- R est utilisé pour l'exploration de données, la bioinformatique et l'analyse de données.
- R a été lancé par les professeurs Ross Ihaka et Robert Gentleman en tant que langage de programmation pour enseigner les statistiques d'introduction à l'Université d'Auckland et a été lancé en 1993.



Développement du langage S à Bell Laboratoires pour la programmation statistique

Premières versions non publiques de R

Premières versions publiques de R

R est devenu gratuit et open-source

L'équipe de R Core a été composée.

Première version officielle, R 1.0.0

R reste largement utilisé en statistiques et analyse de données, R 4.3.2.

Caractéristiques de R

- **Open Source:** R est accessible gratuitement, et son code source est ouvert à la modification et à la distribution
- **Calcul Statistique :** Conçu spécifiquement pour l'analyse statistique, R offre un ensemble riche d'outils pour l'exploration et la modélisation des données
- **Bibliothèques Étendues :** R dispose d'une vaste collection de packages contribués par la communauté, couvrant les méthodes statistiques, l'apprentissage automatique et la visualisation des données
- **Manipulation des Données :** R offre des fonctions robustes pour la manipulation efficace des données
- **Graphiques et Visualisation :** R excelle dans la création de graphiques, de diagrammes et de graphiques de haute qualité pour une visualisation efficace des données

Caractéristiques de R

- **Compatibilité Multiplateforme** : R assurant une accessibilité sur différents systèmes d'exploitation, Windows, macOS et Linux
- **Support de la Communauté** : R bénéficie d'une grande et active communauté qui contribue au développement de packages et fournit un support
- **Intégration avec d'Autres Langages** : R s'intègre parfaitement avec des langages comme C, C++ et Java, améliorant la flexibilité et l'interopérabilité
- **Reproductibilité** : R favorise la recherche reproductible, permettant aux utilisateurs de documenter et de partager des analyses pour plus de transparence et de collaboration

Rôle en statistique

R et ses bibliothèques implémentent diverses techniques statistiques, notamment la modélisation linéaire, linéaire généralisée et non linéaire, les tests statistiques classiques, l'analyse des séries spatiales et temporelles, la classification, le regroupement, etc.

Ces caractéristiques font de **R un outil puissant pour l'analyse de données.**

Les graphiques statiques constituent un autre point fort de R. Il peut produire des graphiques de qualité professionnelle incluant des symboles mathématiques.



CHAPITRE 4

VULGARISER LE LANGAGE R

1. Introduction à R et son rôle en statistique
2. **Préparation de l'environnement**
3. Syntaxe de base et principaux opérateurs en R
4. Structures de données fondamentales en R
5. Premières manipulations de données en R
6. Comparaison des fonctions de base avec Python
7. Reprise des dernières applications (Python/Excel) en R

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de R

Aller sur : <https://cran.r-project.org/>



[CRAN](#)
[Mirrors](#)
[What's new?](#)
[Search](#)
[CRAN Team](#)

[About R](#)
[R Homepage](#)
[The R Journal](#)

[Software](#)
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Task Views](#)
[Other](#)

[Documentation](#)
[Manuals](#)
[FAQs](#)
[Contributed](#)

[Donations](#)
[Report a bug](#)

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux \(Debian, Fedora/Redhat, Ubuntu\)](#)
- [Download R for macOS](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2023-10-31, Eye Holes) [R-4.3.2.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Installation de R sur Windows

Cliquer sur [Download R for Windows](#), puis sur [base](#) s'il s'agit de l'installation de R pour la première fois :



[CRAN](#)
[Mirrors](#)
[What's new?](#)
[Search](#)
[CRAN Team](#)

[About R](#)
[R Homepage](#)
[The R Journal](#)

[Software](#)
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Task Views](#)
[Other](#)

Subdirectories:

[base](#)
[contrib](#)
[old contrib](#)
[Rtools](#)

Binaries for base distribution. This is what you want to [install R for the first time](#).
Binaries of contributed CRAN packages (for R >= 3.4.x).
Binaries of contributed CRAN packages for outdated versions of R (for R < 3.4.x).
Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement



Installation de R sur Windows

Cliquer sur le lien de download :



[CRAN](#)
[Mirrors](#)
[What's new?](#)
[Search](#)
[CRAN Team](#)
[About R](#)

R-4.3.2 for Windows

[Download R-4.3.2 for Windows](#) (79 megabytes, 64 bit)
[README on the Windows binary distribution](#)
[New features in this version](#)

This build requires UCRT, which is part of Windows since Windows 10 and Windows Server 2016. On older systems, UCRT has to be installed manually from [here](#).

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5sum](#) of the .exe to the [fingerprint](#) on the master server.

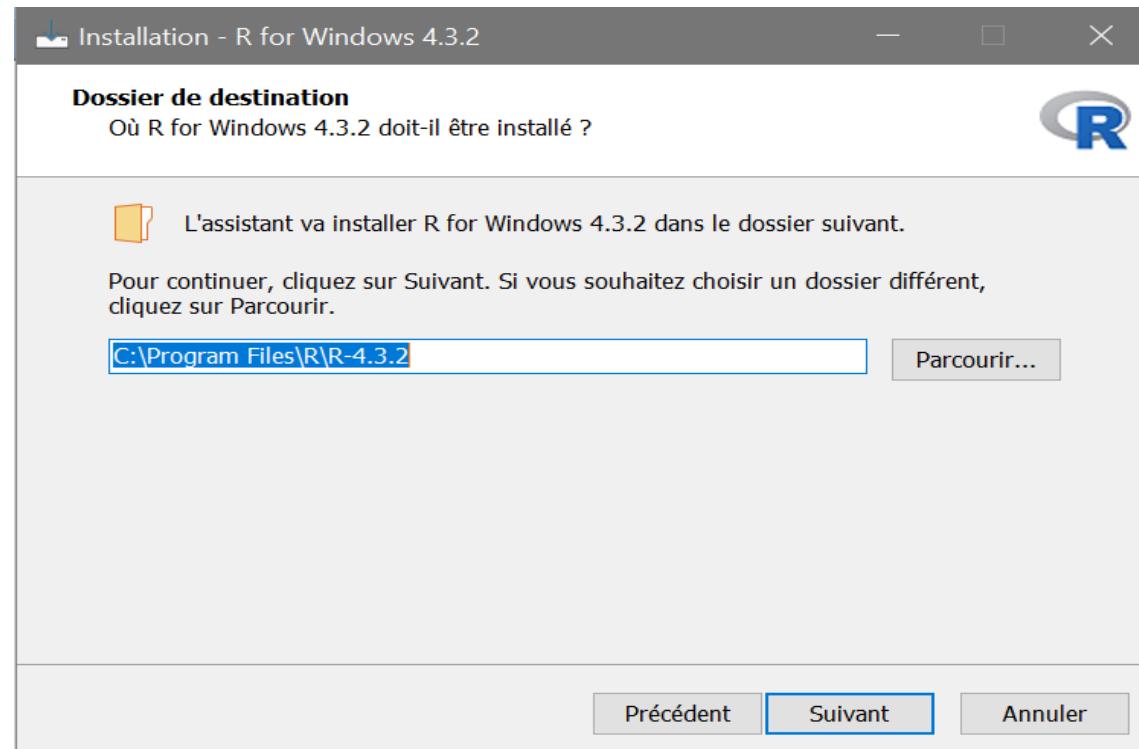
La version la plus récente de R jusqu'au 08 Janvier 2024 était la R-4.3.2

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de R sur Windows

Choisir le répertoire d'installation (le choix par défaut est recommandé) :

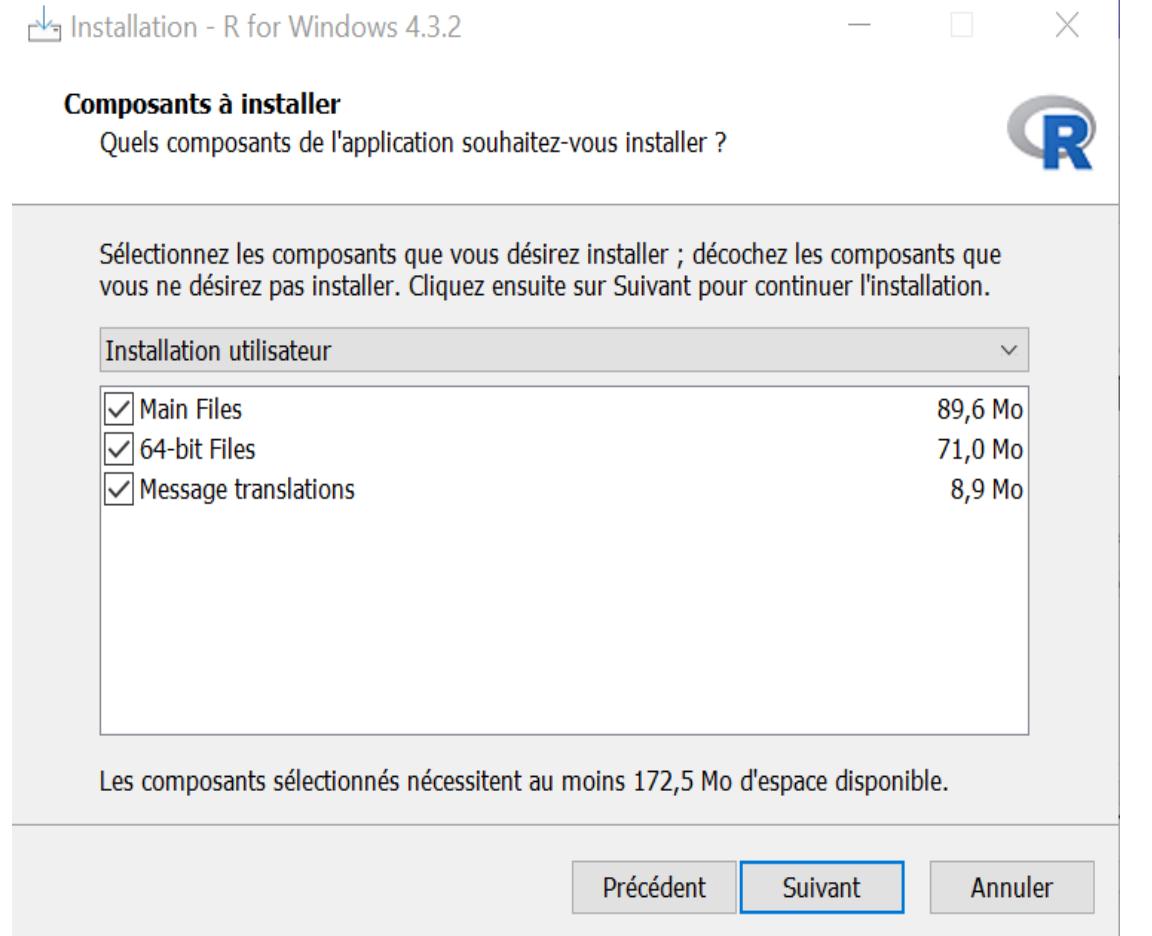


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de R sur Windows

Lors de l'installation de R, il est possible de choisir entre une installation utilisateur rapide avec des paramètres par défaut ou une installation personnalisée offrant davantage de flexibilité. L'installation personnalisée permet de définir le répertoire d'installation, d'ajuster des paramètres spécifiques et d'inclure des composants additionnels selon les préférences de l'utilisateur.

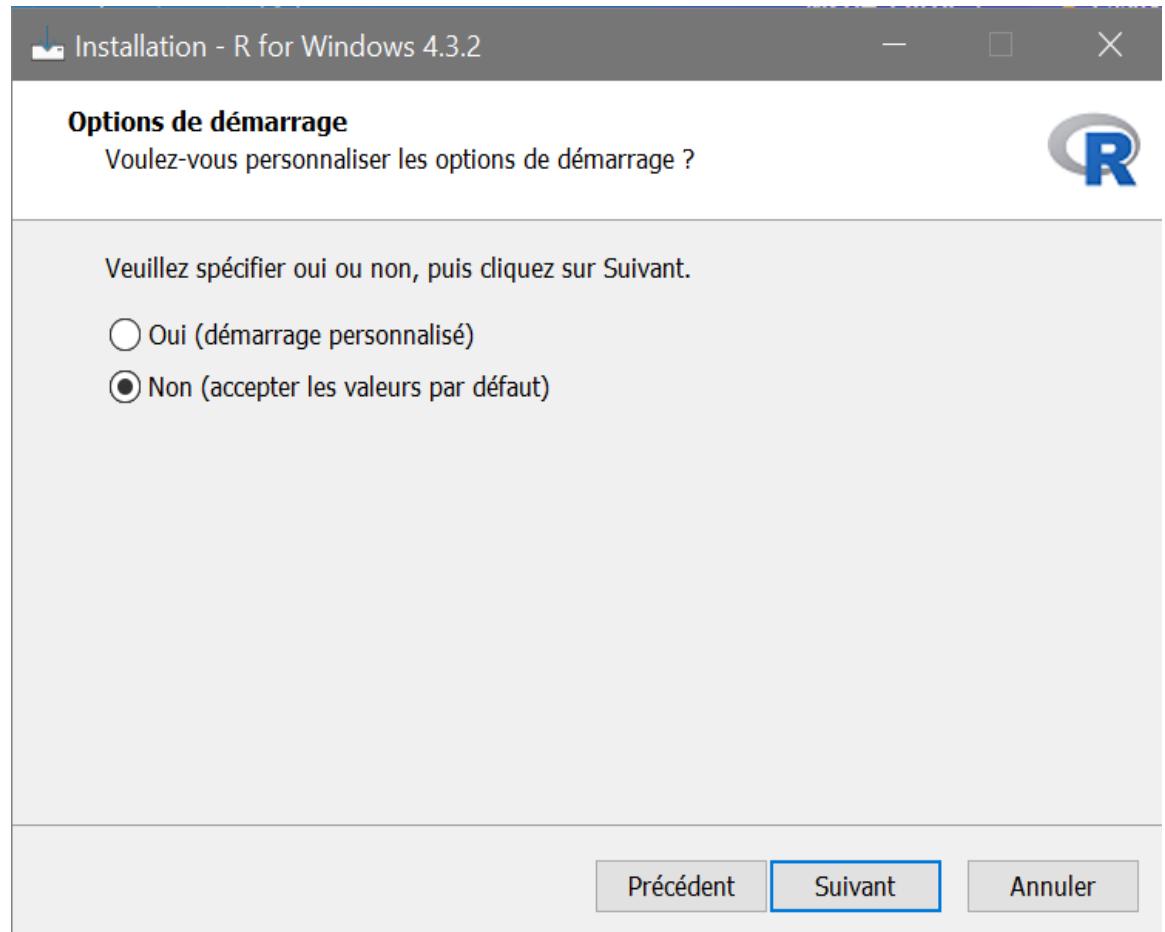


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de R sur Windows

R propose la flexibilité de personnaliser le mode de démarrage en choisissant entre MDI (une seule grande fenêtre) ou SDI (fenêtres séparées), ainsi que le style d'aide, que ce soit en texte ou en HTML



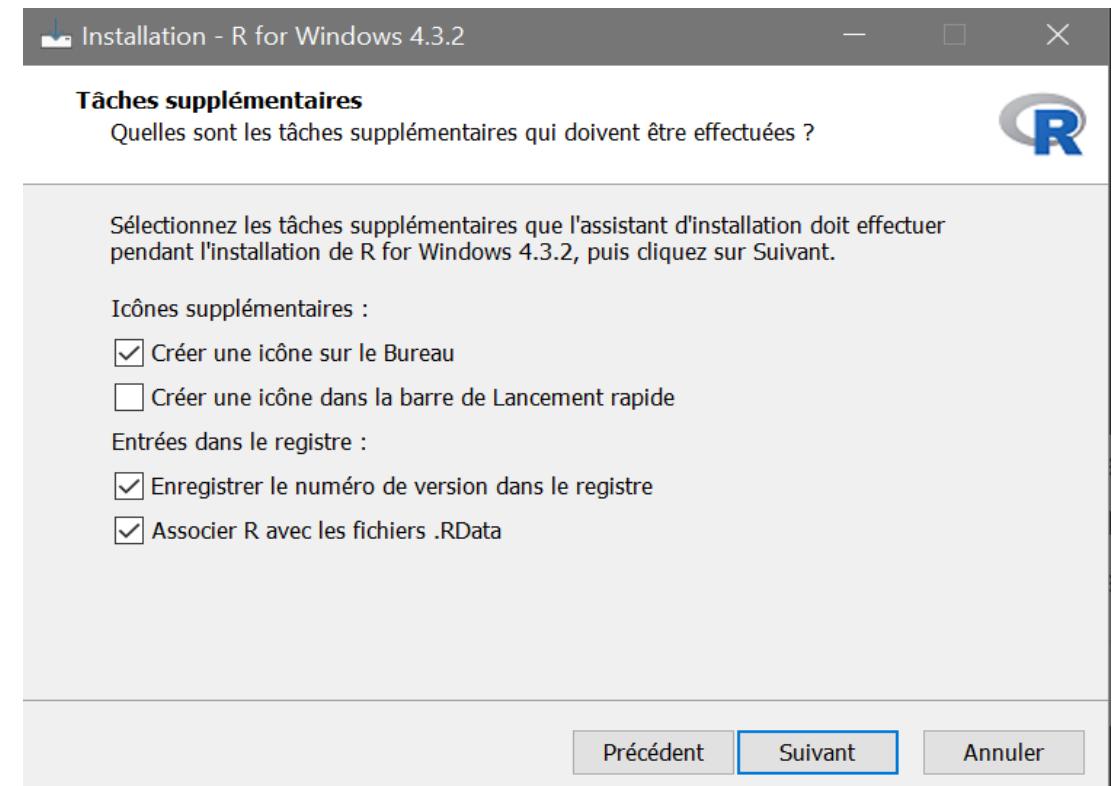
04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de R sur Windows

Cocher la case **Enregistrer le numéro de version dans le registre** lors de l'installation de R permet de stocker des informations sur la version du logiciel dans la base de registre de Windows. Cela facilite le suivi des installations, garantit la compatibilité avec d'autres logiciels et simplifie le dépannage en fournissant des détails précis sur la version installée

Cocher la case **Associer R avec des fichiers .RData** permet de définir R comme le programme par défaut pour ouvrir ces fichiers, simplifiant ainsi le processus d'ouverture et d'utilisation

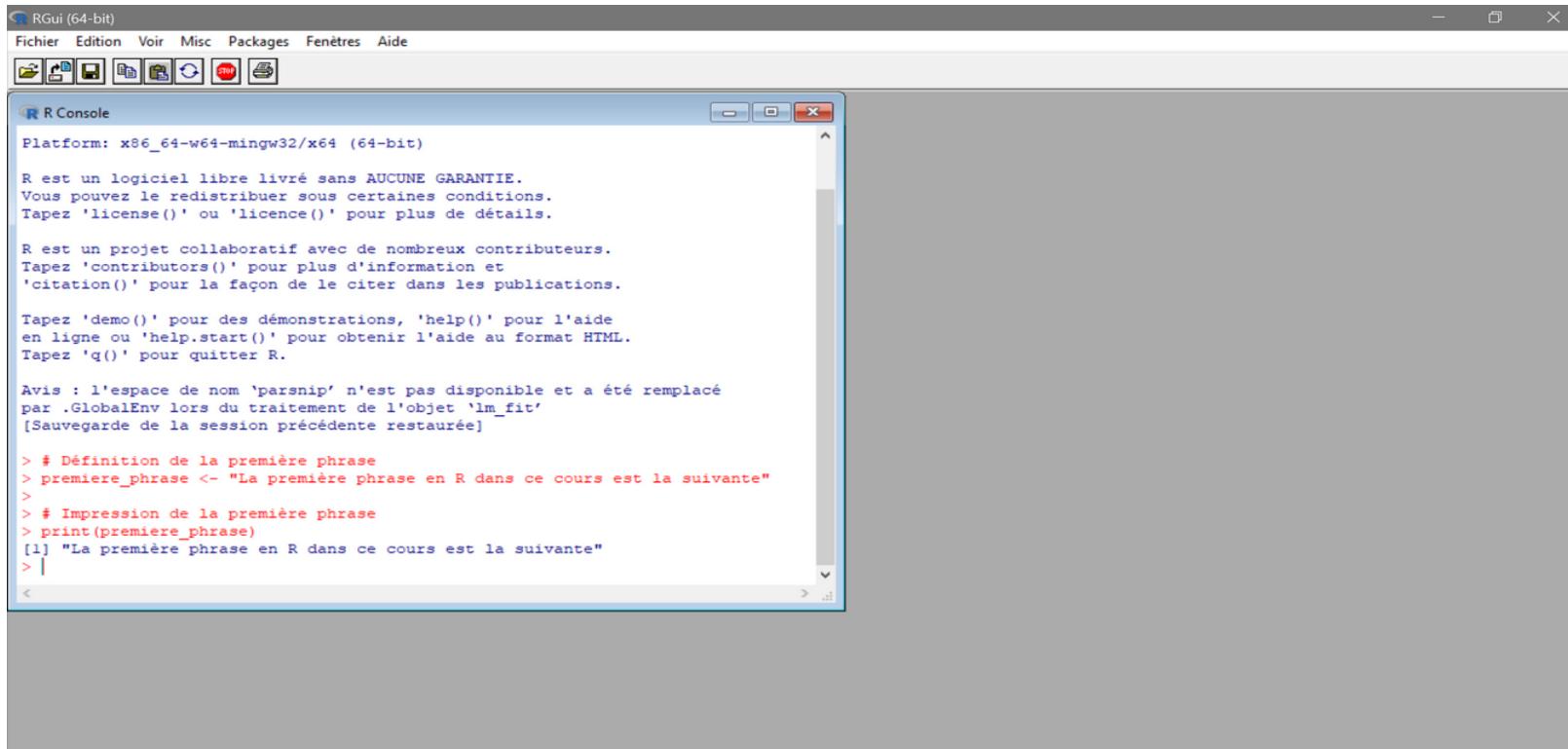


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Interface R

Voici l'apparence de l'interface graphique de R après l'installation, comprenant également un exemple de code affichant une chaîne de caractères



04 – VULGARISER LE LANGAGE R

Préparation de l'environnement



Installation de RStudio sur Windows

Aller sur : <https://posit.co/download/rstudio-desktop/#download> ensuite cliquer sur DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS

2: Install RStudio

[DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS](#)

Size: 215.66 MB | [SHA-256: 93C7F307](#) | Version:
2023.12.0+369 | Released: 2023-12-20

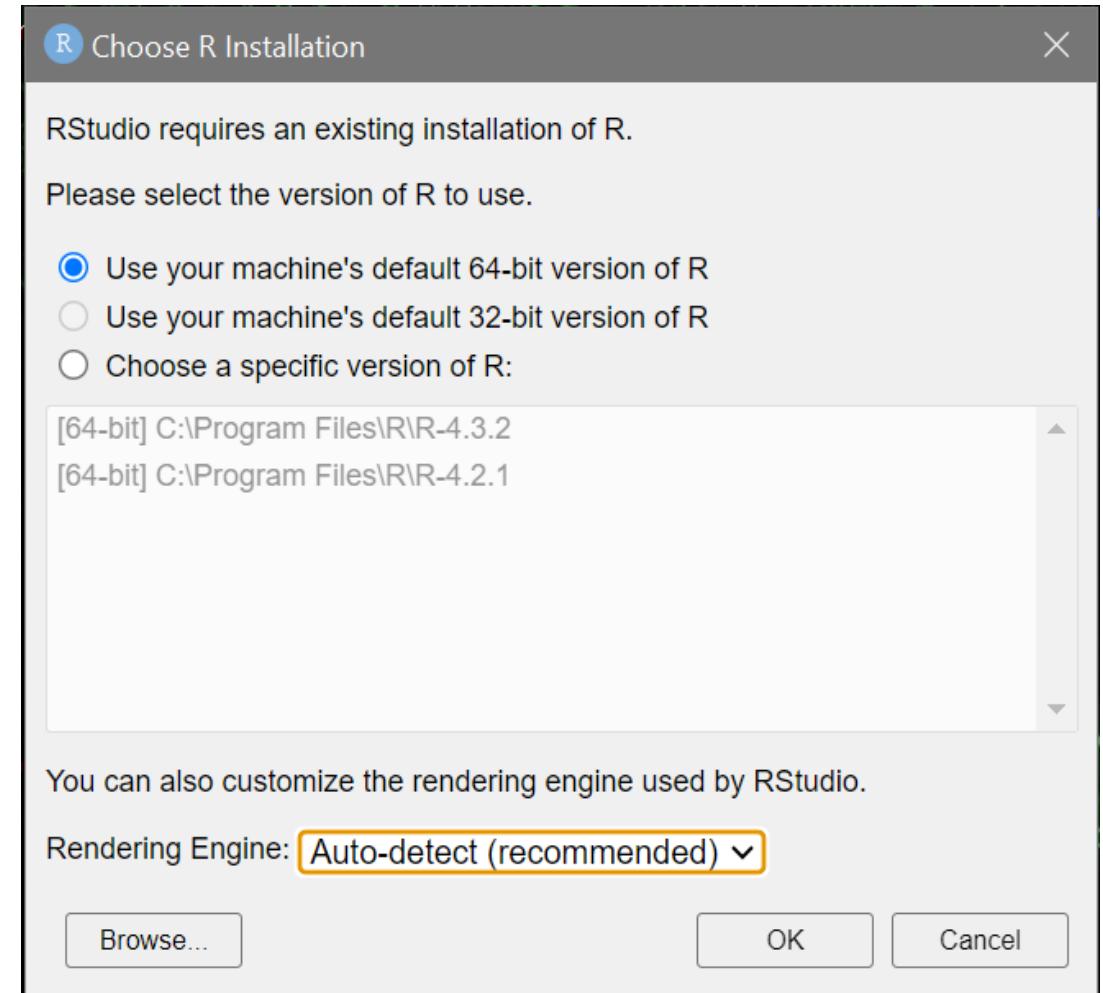
RStudio exige que R soit déjà installé sur le système

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de RStudio sur Windows

En suivant les étapes d'installation de base, l'interface qui suit apparaît. Cette configuration permet de déterminer comment RStudio interagit avec R, en précisant la version et l'architecture de R à utiliser. Elle permet également de choisir la manière dont les éléments graphiques sont générés et affichés dans l'interface utilisateur de RStudio, en fonction de préférences ou de besoins spécifiques

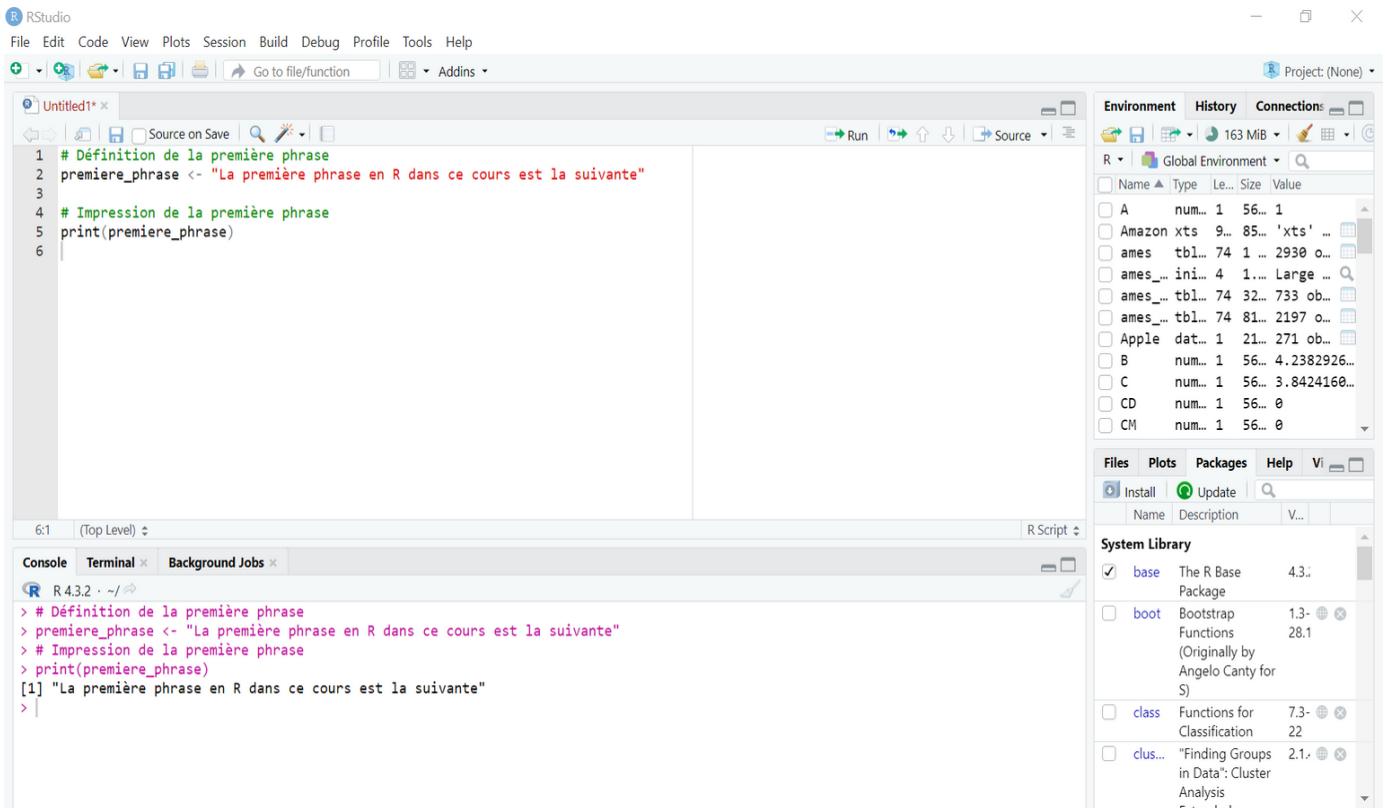


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Installation de RStudio sur Windows

Voici l'apparence de RStudio, avec le même script initial affichant une chaîne de caractères écrit en R



The screenshot shows the RStudio desktop application. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and a Project dropdown set to (None). The toolbar below has icons for file operations like Open, Save, and Run, along with Go to file/function and Addins. The main workspace contains two panes: a Script Editor on the left and a Console on the right. The Script Editor shows the following R code:

```
1 # Définition de la première phrase
2 premiere_phrase <- "La première phrase en R dans ce cours est la suivante"
3
4 # Impression de la première phrase
5 print(premiere_phrase)
6
```

The Console pane shows the output of running this code in R 4.3.2:

```
R 4.3.2 : ~/ ...
> # Définition de la première phrase
> premiere_phrase <- "La première phrase en R dans ce cours est la suivante"
> # Impression de la première phrase
> print(premiere_phrase)
[1] "La première phrase en R dans ce cours est la suivante"
>
```

On the right side of the interface are several toolbars and panels: Environment, History, Connections, Global Environment (listing objects like A, Amazon xts, ames, ames... ini, ames... tbl, ames... dat, B, C, CD, CM), Files, Plots, Packages, Help, and a System Library panel showing packages like base, boot, class, and clus... with their details.

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Navigation et utilisation efficace de RStudio

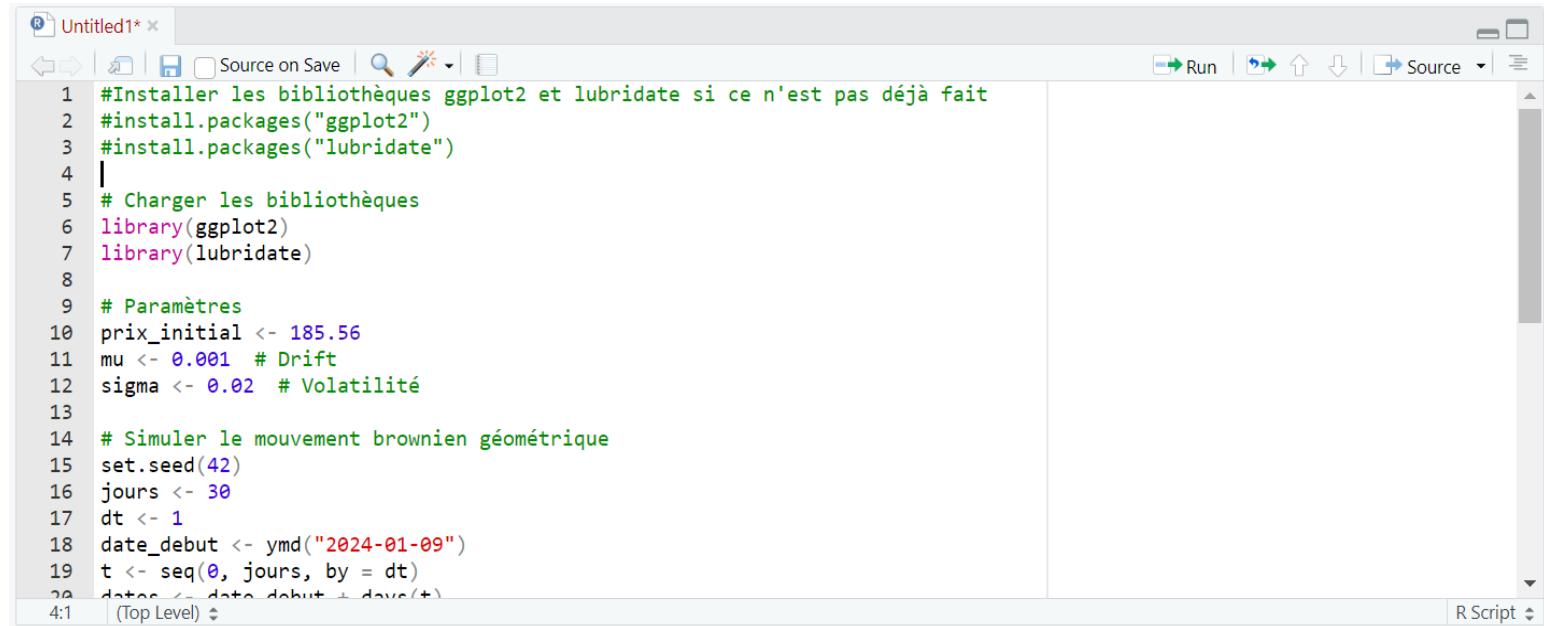
Comprendre comment naviguer et utiliser efficacement les différentes zones de RStudio est essentiel pour maximiser votre productivité



Navigation et utilisation efficace de RStudio

Zone Édition

La zone d'édition de RStudio est l'endroit central où vous saisissez et modifiez votre code R, bénéficiant de fonctionnalités comme la coloration syntaxique et l'auto-complétion



```
R Untitled1* 
Source on Save | Run | Source | 
1 #Installer les bibliothèques ggplot2 et lubridate si ce n'est pas déjà fait
2 #install.packages("ggplot2")
3 #install.packages("lubridate")
4 |
5 # Charger les bibliothèques
6 library(ggplot2)
7 library(lubridate)
8
9 # Paramètres
10 prix_initial <- 185.56
11 mu <- 0.001 # Drift
12 sigma <- 0.02 # Volatilité
13
14 # Simuler le mouvement brownien géométrique
15 set.seed(42)
16 jours <- 30
17 dt <- 1
18 date_debut <- ymd("2024-01-09")
19 t <- seq(0, jours, by = dt)
20 dates <- date_debut + days(t)
4:1 (Top Level) ♦
```

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Navigation et utilisation efficace de RStudio

Zone Console

La console de RStudio est l'espace interactif où vous pouvez exécuter des commandes R en temps réel et obtenir les résultats instantanément, facilitant le processus de développement, de test et d'exploration de code R



The screenshot shows the RStudio interface with the 'Console' tab selected. The title bar indicates 'Console' is active, along with 'Terminal' and 'Background Jobs'. The main area displays an R session:

```
R 4.3.2 · ~/ 
> 1+1
[1] 2
> |
```

Navigation et utilisation efficace de RStudio

Zone Fichiers

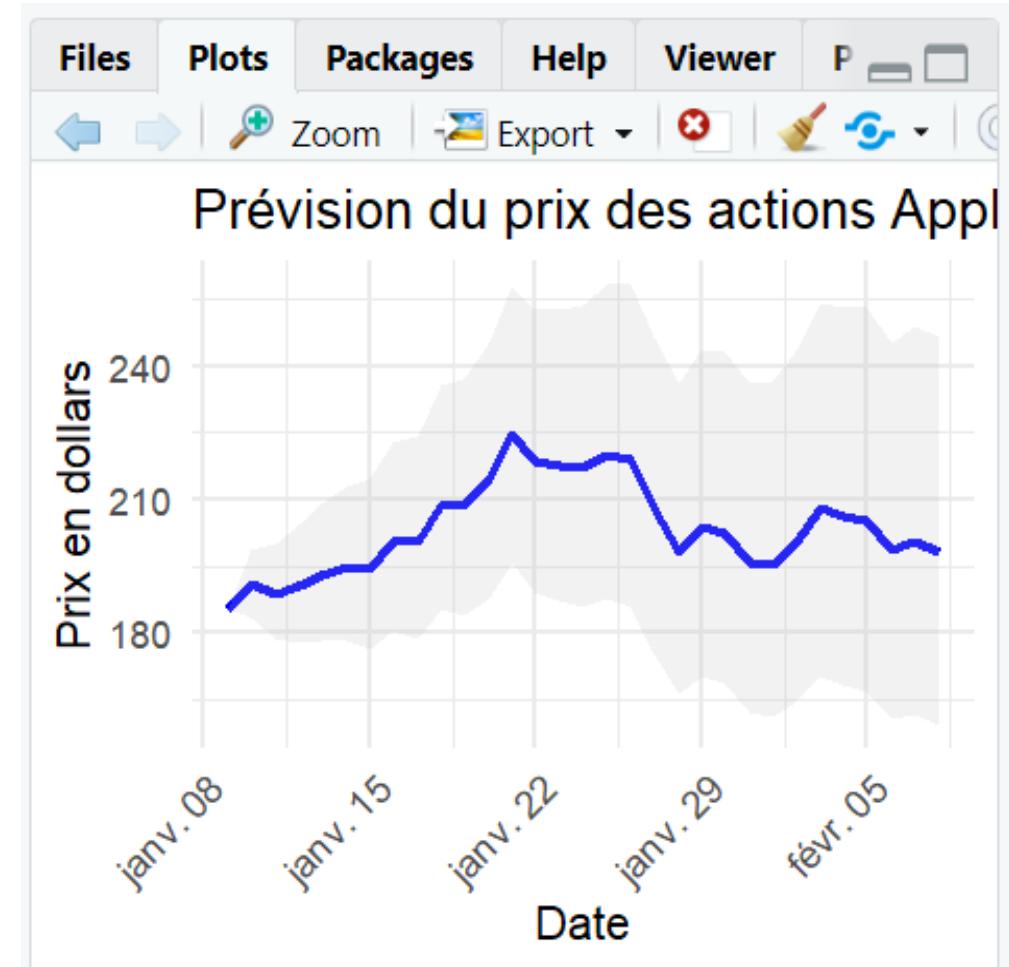
"**Files**", facilite la navigation dans les fichiers

"**Plots**", donne accès à la fenêtre d'affichage et d'exportation des graphiques

"**Packages**", permet l'installation et la mise à jour des packages

"**Help**", offre un accès à l'aide en ligne pour toutes les fonctions des packages chargés dans R

"**Viewer**" simplifie la visualisation de sorties complexes, comme des fichiers HTML, des tableaux, et des graphiques interactifs.

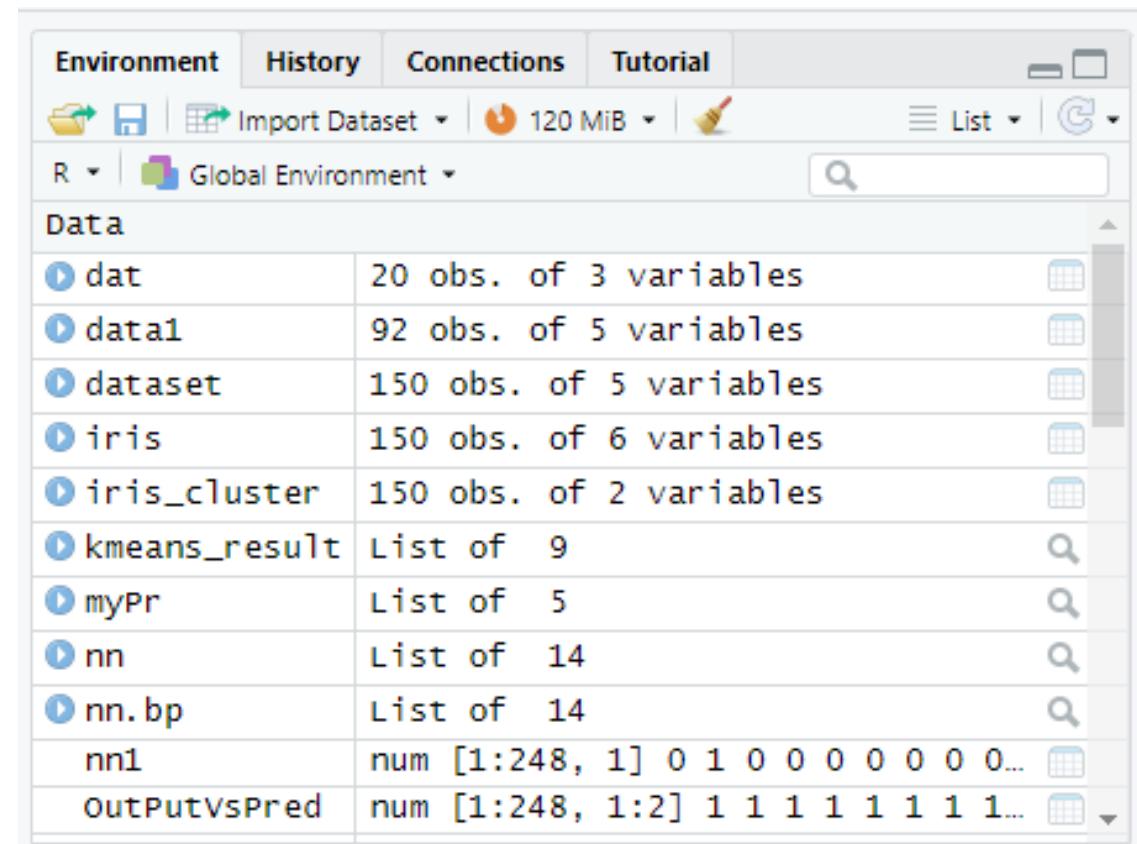


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Zone Environnement

La zone "Environnement" de RStudio affiche une liste interactive des objets (variables, données, etc.) actuellement chargés en mémoire, permettant un suivi en temps réel et une gestion facile des éléments utilisés dans votre session R



The screenshot shows the RStudio interface with the "Environment" tab selected. The top bar includes tabs for Environment, History, Connections, and Tutorial, along with various icons for file operations like Import Dataset, Save, and Undo. A search bar is also present. The main area is titled "Global Environment" and lists the following objects:

| Object | Description | View |
|---------------|--|-----------|
| dat | 20 obs. of 3 variables | grid icon |
| data1 | 92 obs. of 5 variables | grid icon |
| dataset | 150 obs. of 5 variables | grid icon |
| iris | 150 obs. of 6 variables | grid icon |
| iris_cluster | 150 obs. of 2 variables | grid icon |
| kmeans_result | List of 9 | grid icon |
| myPr | List of 5 | grid icon |
| nn | List of 14 | grid icon |
| nn.bp | List of 14 | grid icon |
| nn1 | num [1:248, 1] 0 1 0 0 0 0 0 0 0 ... | grid icon |
| OutPutVsPred | num [1:248, 1:2] 1 1 1 1 1 1 1 1 1 ... | grid icon |

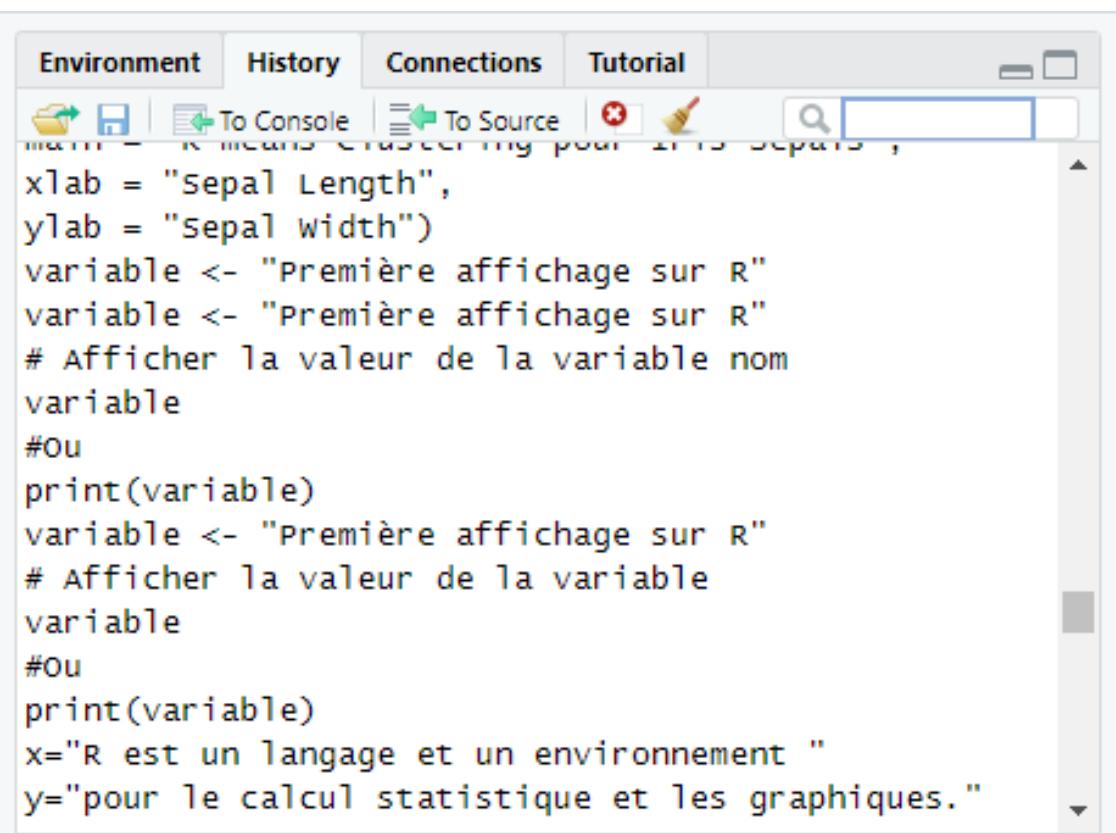
04 – VULGARISER LE LANGAGE R

Préparation de l'environnement



Zone History

La Zone "History" offre un enregistrement chronologique des commandes exécutées pendant la session actuelle dans RStudio



The screenshot shows the RStudio interface with the 'History' tab selected in the top navigation bar. Below the tabs, there are several icons: a folder, a file, a green arrow pointing to 'Console', a blue arrow pointing to 'Source', a red 'X', a pencil, and a magnifying glass. The main area displays a scrollable list of R code. The code consists of several lines of R syntax, including variable assignments and print statements. Some parts of the text are highlighted in blue, likely indicating they are comments or specific code blocks.

```
xlab = "Sepal Length",
ylab = "Sepal width")
variable <- "Première affichage sur R"
variable <- "Première affichage sur R"
# Afficher la valeur de la variable nom
variable
#Ou
print(variable)
variable <- "Première affichage sur R"
# Afficher la valeur de la variable
variable
#Ou
print(variable)
x="R est un langage et un environnement "
y="pour le calcul statistique et les graphiques."
```

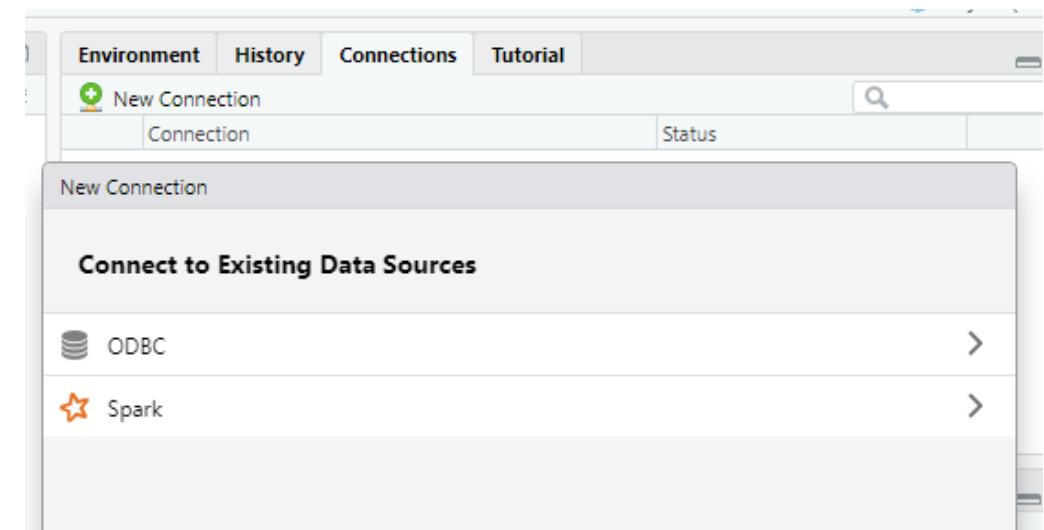
04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Zone Connections

La zone "Connections" dans RStudio permet d'établir des connexions à diverses sources de données pour l'analyse

En cliquant sur "New Connection", vous pouvez choisir entre différentes options telles que **ODBC** ou **Spark** pour établir des connexions avec des bases de données. ODBC (Open Database Connectivity) est une interface standard pour accéder à des bases de données relationnelles, tandis que Spark offre une connectivité aux bases de données distribuées, facilitant l'exploration et la manipulation des données dans divers contextes d'analyse décisionnelle

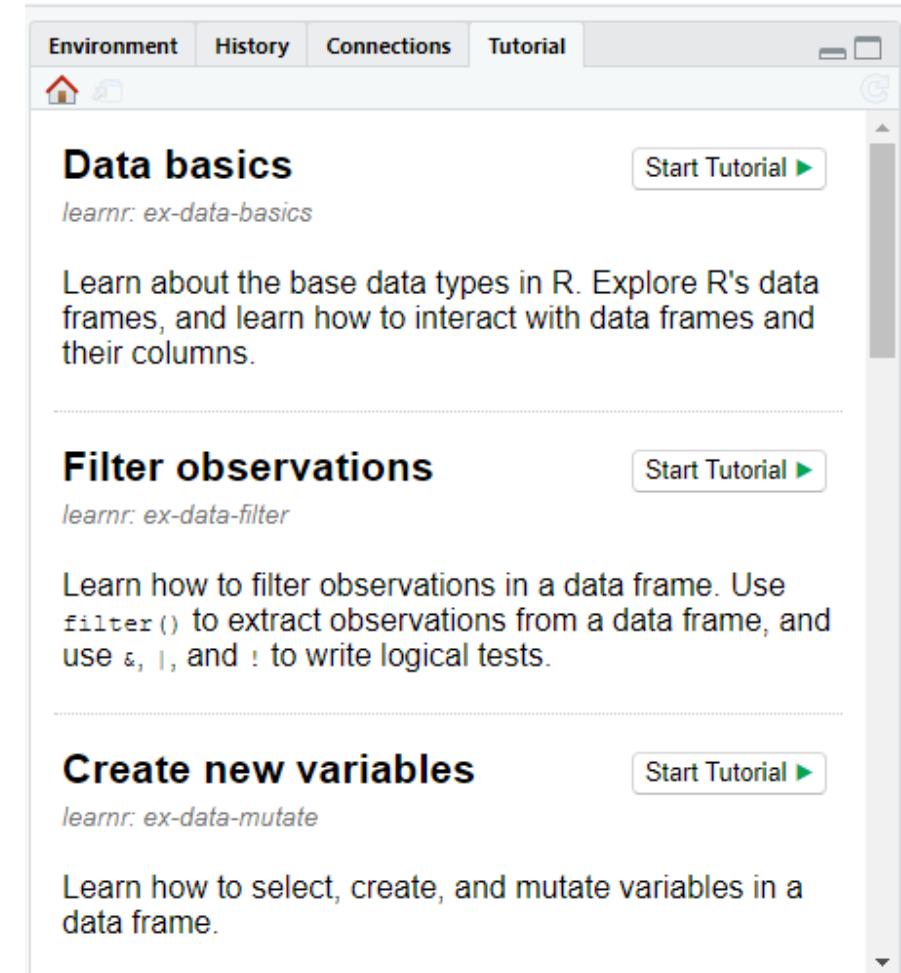


04 – VULGARISER LE LANGAGE R

Préparation de l'environnement

Zone Tutorial

La section "Tutoriels" de RStudio propose des ressources éducatives pour améliorer les compétences en programmation R, avec des instructions détaillées et des exemples pratiques. Adaptée à tous les niveaux d'expérience, cette section facilite l'apprentissage et le perfectionnement des utilisateurs



The screenshot shows the RStudio interface with the "Tutorial" tab selected in the top navigation bar. Below the tabs, there are three tutorial cards:

- Data basics** ([Start Tutorial](#))
learnr: ex-data-basics
Learn about the base data types in R. Explore R's data frames, and learn how to interact with data frames and their columns.
- Filter observations** ([Start Tutorial](#))
learnr: ex-data-filter
Learn how to filter observations in a data frame. Use `filter()` to extract observations from a data frame, and use `&`, `|`, and `!` to write logical tests.
- Create new variables** ([Start Tutorial](#))
learnr: ex-data-mutate
Learn how to select, create, and mutate variables in a data frame.

Navigation et utilisation efficace de RStudio

R Markdown

R Markdown est un langage de balisage léger qui permet d'intégrer du code R dans des documents pour créer des rapports dynamiques, des présentations ou même des pages web interactives

- Vous pouvez créer un nouveau fichier R Markdown en utilisant l'extension **.Rmd**
- Les documents générés peuvent être au format HTML, PDF, Word, et bien d'autres
- R Markdown est un outil extrêmement utile pour exporter, communiquer et diffuser les résultats d'une analyse



04 – VULGARISER LE LANGAGE R

Préparation de l'environnement



Navigation et utilisation efficace de RStudio

En R : Où trouver de l'aide en cas de blocage ?



Il existe plusieurs ressources pour obtenir de l'aide en cas de blocage. Voici quelques options que vous pouvez explorer :

- **Forums en ligne**
 - **Stack Overflow** (<https://stackoverflow.com/>) : Une communauté active où vous pouvez poser des questions sur des problèmes spécifiques que vous rencontrez en R. Assurez-vous de fournir des détails pertinents et un exemple de code pour obtenir des réponses utiles
 - **RStudio Community** (<https://community.rstudio.com/>) : Un forum dédié à RStudio. Vous pouvez poser des questions sur l'utilisation de RStudio et sur des problèmes liés à R

04 – VULGARISER LE LANGAGE R

Préparation de l'environnement



Navigation et utilisation efficace de RStudio

En R : Où trouver de l'aide en cas de blocage ?



- **Listes de diffusion**
 - **R-help** (<https://stat.ethz.ch/mailman/listinfo/r-help>) : Une liste de diffusion R officielle où vous pouvez poser des questions et recevoir des réponses de la communauté R
- **Documentation en ligne**
 - **R Documentation** (<https://www.rdocumentation.org/>) : Une ressource utile pour rechercher des fonctions et des packages R, ainsi que pour accéder à la documentation officielle
 - **RStudio Cheat Sheets** (<https://www.rstudio.com/resources/cheatsheets/>) : Des fiches de référence pratiques pour divers aspects de R et RStudio



CHAPITRE 4

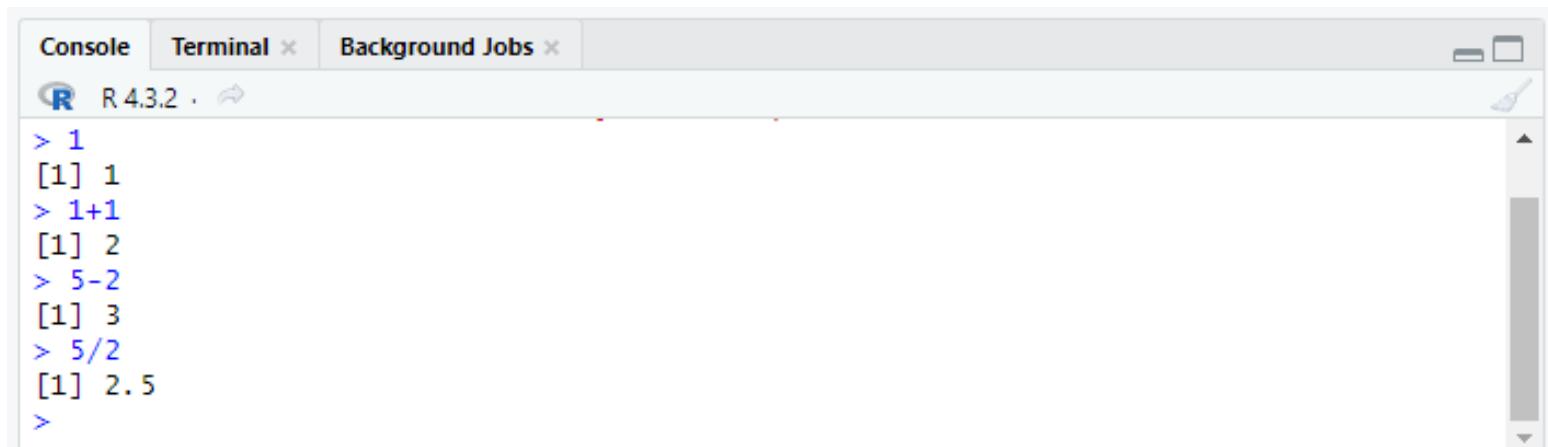
VULGARISER LE LANGAGE R

1. Introduction à R et son rôle en statistique
2. Préparation de l'environnement
- 3. Syntaxe de base et principaux opérateurs en R**
4. Structures de données fondamentales en R
5. Premières manipulations de données en R
6. Comparaison des fonctions de base avec Python
7. Reprise des dernières applications (Python/Excel) en R

Premières commandes

Comme Python, **R est un langage interprété**, ce qui implique la possibilité d'écrire, valider et observer les résultats d'une seule ligne de code sans nécessiter une étape préalable de compilation. Ainsi, pour rédiger du code en R, il suffit de démarrer la **console** et saisir les instructions directement.

Exemple d'application :



The screenshot shows an R console window with three tabs: 'Console', 'Terminal x', and 'Background Jobs x'. The 'Console' tab is active, displaying the following R session:

```
R 4.3.2 .
> 1
[1] 1
> 1+1
[1] 2
> 5-2
[1] 3
> 5/2
[1] 2.5
>
```

Déclaration des variables

En R, il n'existe pas de commande spécifique pour déclarer une variable ; plutôt, une variable est créée au moment où vous lui assignez une valeur

Vous pouvez attribuer une valeur à une variable en utilisant le signe <-

Nous pouvons également utiliser le signe =

Pour afficher ou imprimer la valeur d'une variable, il vous suffit de taper son nom

Exemple d'application :

```
variable <- "Première affichage sur R"  
# Afficher la valeur de la variable nom  
  
variable  
#Ou  
  
print(variable)
```



```
> variable <- "Première affichage sur R"  
> # Afficher la valeur de la variable  
> variable  
[1] "Première affichage sur R"  
> #Ou  
> print(variable)  
[1] "Première affichage sur R"  
> |
```

Spécification de l'affectation du type

Dans R, les variables ne nécessitent pas d'être déclarées avec un type particulier et peuvent même changer de type après avoir été définies

Nous pouvons utiliser la fonction `class()` ou `typeof()` pour déterminer le type d'une variable

Exemple d'application :

```
var <- 27                      # var est de type numérique
class(var)                     # => [1] "numeric"
var <- "Type caractère"       # var est maintenant de type caractère
typeof(var)                    # => [1] "character"
var # Afficher var             # => [1] "Type caractère"
```

Types de données les plus utiliser

| Type de données | Description | Déclaration |
|-----------------|-----------------------------------|------------------|
| numérique | Nombres réels (virgule flottante) | Pi <- 3,14 |
| entier | Nombres entiers | y <- 42L |
| caractère | Texte ou chaîne de caractères | nom <- "John" |
| logique | Valeurs booléennes (VRAI ou FAUX) | est_vrai <- TRUE |

Exemple d'application :

```
# numeric  
var <- 15.7  
class(var)
```

```
# integer  
var <- 20L  
class(var)
```

```
# character/string  
var <- "R"  
typeof(var)
```

```
# logical/boolean  
var <- TRUE  
typeof(var)
```

Opérateurs en R

- Les opérateurs sont des éléments essentiels en programmation, permettant d'effectuer des opérations entre des variables et des valeurs. Dans l'exemple ci-dessous, l'opérateur + est utilisé pour additionner deux valeurs :

```
# Exemple d'un opérateur arithmétique
```

```
resultat <- 10 + 5
```

- En R, les opérateurs sont divisés en groupes suivants :

- Opérateurs arithmétiques
- Opérateurs d'assignation
- Opérateurs de comparaison
- Opérateurs logiques
- Opérateurs divers

- Different de Python, en R, on ne peut pas utiliser les opérateurs arithmétiques sur les caractères :

```
> x="R est un langage et un environnement "
> y="pour le calcul statistique et les graphiques."
> z=x+y
```

Erreur dans x + y : argument non numérique pour un opérateur binaire



Opérateurs arithmétiques

- Les opérateurs arithmétiques sont utilisés avec des valeurs numériques pour effectuer des opérations mathématiques courantes :

| Opérateur | Nom | Exemple |
|-----------|-------------------------------|---------------|
| + | Addition | $x + y$ |
| - | Soustraction | $x - y$ |
| * | Multiplication | $x * y$ |
| / | Division | x / y |
| \wedge | Exposant | $x \wedge y$ |
| $\% \%$ | Modulo (Reste de la division) | $x \% \% y$ |
| $\% / \%$ | Division entière | $x \% / \% y$ |

Opérateurs d'assignation

- Les opérateurs d'assignation sont utilisés pour **attribuer des valeurs aux variables**
- <- est un opérateur d'assignation global utilisé pour attribuer une **variable globale**
- Il est également possible d'inverser la direction de l'opérateur d'assignation, var <- 5 est équivalent à 5 -> var

| Opérateur | Description | Exemple |
|-----------|---|------------|
| <- | Assignation standard (vers la droite) | var <- 5 |
| = | Assignation alternative (équivalent à <-) | var = 5 |
| -> | Assignation alternative (vers la gauche) | 5 -> var |
| <<- | Assignation globale | var <<- 10 |

```
# Exemple d'un opérateur d'assignation en R
option <- 'Assistant Data Analyst'
```

Opérateurs de comparaison

- Les opérateurs de comparaison sont utilisés pour comparer deux valeurs en R

| Opérateur | Nom | Exemple |
|--------------------|---------------------|------------------------|
| <code>==</code> | Égal | <code>x == y</code> |
| <code>!=</code> | Non égal | <code>x != y</code> |
| <code>></code> | Supérieur à | <code>x > y</code> |
| <code><</code> | Inférieur à | <code>x < y</code> |
| <code>>=</code> | Supérieur ou égal à | <code>x >= y</code> |
| <code><=</code> | Inférieur ou égal à | <code>x <= y</code> |

Opérateurs logiques

- Les opérateurs logiques sont utilisés pour **combiner** des déclarations **conditionnelles** en R :

| Opérateur | Description | Exemple d'Entrée | Exemple de Sortie |
|-----------|--|--|-------------------|
| & | Opérateur logique AND élément par élément. Renvoie TRUE si les deux éléments sont TRUE. | c(TRUE, TRUE, FALSE) & c(TRUE, FALSE, FALSE) | TRUE FALSE FALSE |
| && | Opérateur logique AND - Renvoie TRUE si les deux déclarations sont TRUE. | TRUE && c(TRUE, FALSE, FALSE) | FALSE |
| | Opérateur logique OR élément par élément. Renvoie TRUE si l'une des déclarations est TRUE. | c(TRUE, FALSE, FALSE) c(FALSE, TRUE, TRUE) | TRUE TRUE FALSE |
| | Opérateur logique OR - Renvoie TRUE si l'une des déclarations est TRUE. | TRUE FALSE | TRUE |
| ! | NOT logique - Renvoie FALSE si la déclaration est TRUE. | !TRUE | FALSE |

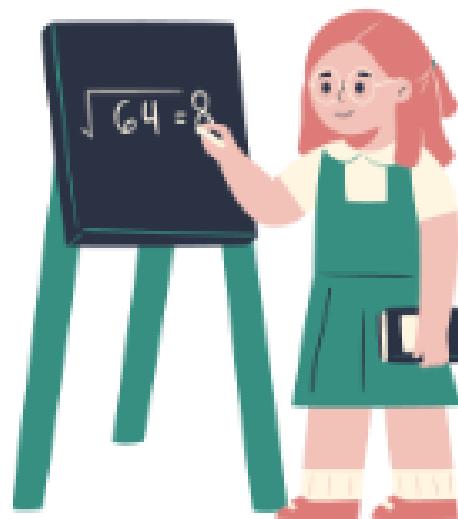
Opérateurs divers

Les opérateurs divers sont utilisés pour **manipuler les données** en R

| Opérateur | Description | Exemple |
|-----------|---|-----------------------------------|
| : | Crée une séquence de nombres | x <- 1:10 |
| %in% | Détermine si un élément appartient à un vecteur | 5 %in% c(1, 3, 5, 7, 9) |
| %*% | Multiplication de matrices | resultat <- Matrice1 %*% Matrice2 |

Fonctions mathématiques

Les fonctions mathématiques de base en R peuvent être utilisées dans divers contextes pour effectuer des calculs mathématiques :



| Fonction | Description | Exemple d'Utilisation | Sortie Attendue |
|-----------|---------------------|----------------------------|-----------------|
| sqrt() | Racine carrée | sqrt(25) | 5 |
| exp() | Exponentielle | exp(2) | 7.389056 |
| log() | Logarithme népérien | log(10) | 2.302585 |
| log10() | Logarithme base 10 | log10(100) | 2 |
| abs() | Valeur absolue | abs(-7) | 7 |
| round() | Arrondi | round(3.14159, digits = 2) | 3.14 |
| floor() | Arrondi inférieur | floor(5.9) | 5 |
| ceiling() | Arrondi supérieur | ceiling(5.1) | 6 |
| sin() | Sinus | sin(pi/2) | 1 |
| cos() | Cosinus | cos(pi) | -1 |
| tan() | Tangente | tan(pi/4) | 1 |

Lire depuis l'entrée utilisateur

- En R, la fonction **readline()** permet de lire des données depuis la console
- La fonction **cat()** en R est utilisée pour imprimer du texte dans la console. Elle concatène et affiche les arguments fournis, offrant une flexibilité pour la mise en forme et l'affichage de résultats, messages, ou variables
- Cette approche est utile pour personnaliser les interactions avec l'utilisateur, comme les salutations personnalisées

Afficher un message, stocker l'entrée dans une variable et attendre que l'utilisateur saisisse son nom.

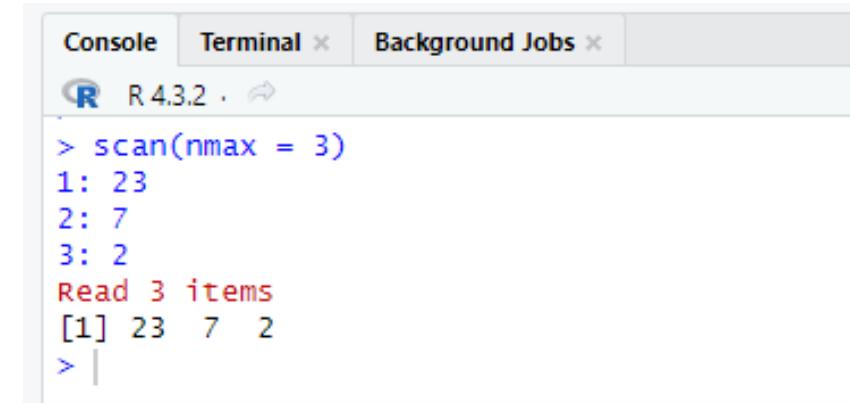
Affiche un message de salutation incluant le nom saisi

```
> # Lire l'entrée utilisateur
> prenom <- readline("Entrez votre nom : ")
Entrez votre nom : Mohamed
> # Afficher un message de salutation
> cat("Bonjour : ", prenom, "\n")
Bonjour : Mohamed
```

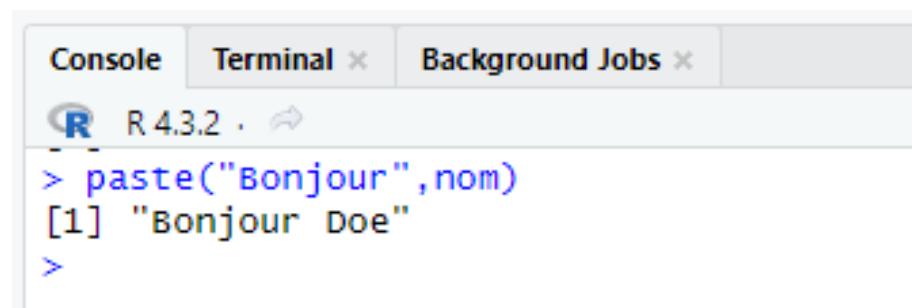


Lire depuis l'entrée utilisateur

- La commande `scan()` lit des valeurs que vous lui entrez dans la console R. C'est comme une petite boîte qui attend vos données !
- Pour le contrôler :
 - Tapez deux fois Entrer quand vous avez fini d'entrer des valeurs.
 - Donnez-lui un nombre d'entrées précis avec `scan(nmax = 5)`.
 - Cachez ses messages avec `scan(quiet = TRUE)` personnalisées
- commande `paste()` permet de coller bout à bout différents morceaux de texte



```
R 4.3.2 
> scan(nmax = 3)
1: 23
2: 7
3: 2
Read 3 items
[1] 23 7 2
>
```



```
R 4.3.2 
> paste("Bonjour", nom)
[1] "Bonjour Doe"
>
```



Conditions et instructions conditionnelles en R

- Les instructions conditionnelles en R permettent d'orienter l'exécution du code en fonction de certaines conditions
- L'utilisation de constructions **if-else** offre une flexibilité cruciale pour automatiser des décisions basées sur des critères spécifiques
- Les instructions **if-else** sont fondamentales pour la programmation conditionnelle en R. Elles permettent d'exécuter différentes parties du code en fonction de la véracité d'une condition

```
if (condition) {  
    # Bloc de code à exécuter si la condition est vraie  
} else {  
    # Bloc de code à exécuter si la condition est fausse  
}
```

- En R, les accolades { } sont utilisées pour définir la portée « scope » dans le code, ce qui **diffère de Python** mais ressemble à la plupart des langages de programmation tels que C, Java, C#, PHP et Javascript
- Dans ces langages, ainsi que pour R, l'espacement est simplement utilisé comme une bonne pratique et pour organiser le code de manière claire et lisible

Instructions conditionnelles en R

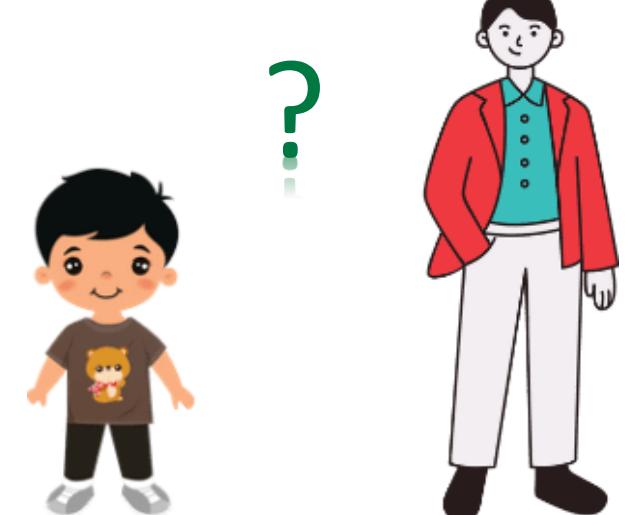
Exemple d'Application

- Supposons que nous voulons créer une instruction conditionnelle qui imprime un message différent **en fonction de l'âge**

```
age <- 25
if (age < 18) {
  print("Vous êtes mineur.")
} else {
  print("Vous êtes majeur.")}
```

Sortie Attendue : Vous êtes majeur.

- Dans cet exemple, le programme imprime "Vous êtes mineur" si l'âge est inférieur à 18, sinon il imprime "Vous êtes majeur"



Instructions conditionnelles en R

Conditions multiples :

Il est également possible d'utiliser des conditions multiples avec l'instruction **if-else**

```
# Variables
temperature <- -5
is_raining <- TRUE
# Prise de décision en fonction des conditions
if (is_raining) {
    print("Il pleut. Prenez un parapluie si vous sortez.")
} else if (temperature < 10) {
    print("Il fait froid. Portez une veste chaude.")
} else if (temperature < 0) {
    print("Il gèle. Il est recommandé de rester à l'intérieur.")
} else {
    print("Le temps est clément. Profitez de votre journée !")
}
```

Instructions imbriquées :

Évaluer des conditions à l'intérieur d'une condition principale pour des décisions plus complexes.

```
# Sample variables
temperature <- 15
is_raining <- TRUE
# Nested if statements
if (temperature > 10) {
    if (is_raining) {
        print("Il pleut. Prenez un parapluie si vous sortez.")
    } else {
        print(" Le temps est clément. Profitez de votre
journée!")
    }
} else {
    print("Il fait froid. Portez une veste chaude.")
}
```

Conditions et opérateurs logiques

En R, les opérateurs logiques **ET (&)** et **OU (|)** sont des outils essentiels pour la gestion des conditions. L'utilisation de ces opérateurs au sein d'une condition permet de combiner astucieusement plusieurs critères dans une seule déclaration, facilitant ainsi la mise en place de conditions complexes et la prise de décisions fines en fonction de multiples facteurs simultanément.

Rappel et Syntaxe en R :

Tableau de Vérité pour l'Opération ET (Logical AND):

| Input 1 | Input 2 | Output AND |
|---------|---------|------------|
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | False |

```
if (a > b & c > a) {  
  print("Les deux conditions sont vraies")  
}
```

Tableau de Vérité pour l'Opération OU (Logical OR) :

| Input 1 | Input 2 | Output OR |
|---------|---------|-----------|
| True | True | True |
| True | False | True |
| False | True | True |
| False | False | False |

```
if (a > b | a > c) {  
  print(" Au moins une des conditions est vraie ")  
}
```

Boucles en R et comparaison avec python :

Les boucles sont des structures de contrôle essentielles dans la programmation, permettant d'exécuter des blocs de code de manière répétée. En R, nous avons principalement deux types de boucles : la boucle **for**, la boucle **while**, et la boucle **repeat**.

Boucle for en R :

La boucle **for** est utilisée pour itérer sur une séquence d'éléments.

Voici un exemple :

```
# Exemple de boucle for en R
for (i in 1:5) {
  print(i)
}
```

Cela affichera les nombres de 1 à 5. En comparaison, voici la même boucle en Python :

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

```
# Exemple de boucle for en Python
for i in range(1, 6):
    print(i)
```



Boucles en R et comparaison avec Python :

Boucle While en R :

La boucle **while** continue à exécuter un bloc de code tant qu'une condition est vraie.

Voici un exemple :

```
# Exemple de boucle while en R
counter <- 1
while (counter <= 5) {
  print(counter)
  counter <- counter + 1
}
```



Cela affichera les nombres de 1 à 5. En comparaison, voici la **même boucle en Python** :

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

```
# Exemple de boucle while en Python
counter = 1
while counter <= 5:
  print(counter)
  counter += 1
```



Boucles en R et comparaison avec Python :

Boucle repeat en R :

La boucle **repeat** offre une manière d'itérer sur un bloc de code jusqu'à ce qu'une condition spécifiée soit rencontrée

Voici un exemple illustrant l'affichage des nombres de 1 à 5 à l'aide de la boucle "repeat" :

```
# Exemple de boucle repeat en R
counter <- 1
repeat {
  print(counter)
  counter <- counter + 1
  # Condition pour sortir de la boucle
  if (counter > 5) {
    break
  }
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

La différence Avec la Boucle while :

La boucle "repeat" exécute le code avant de vérifier la condition, tandis que la boucle "while" vérifie d'abord la condition.

Comparaison avec Python

Bien que flexibles, la boucle "repeat" n'est pas présente en Python. Toutefois, elle correspond à une structure de boucle similaire appelée "**do while**" que l'on trouve dans d'autres langages.

Fonctions en R

Une fonction est un bloc de code qui ne s'exécute que lorsqu'il est appelé.

Définition d'une Fonction :

Pour créer une fonction, utilisez le mot-clé **function**

```
my_function <- function() {  
  print("Hello World!")  
}
```

Appel d'une Fonction :

Pour appeler une fonction, utilisez son nom suivi de parenthèses, comme dans **my_function()**

```
my_function()
```

Gestion des paramètres et valeurs de retour d'une fonction

Paramètres et Arguments:

Un **paramètre** est la variable dans la définition de la fonction, tandis qu'un **argument** est la valeur passée lors de l'appel de la fonction.

Valeur par Défaut des Paramètres :

Vous pouvez définir une **valeur par défaut** pour les paramètres. Si la fonction est appelée sans argument, elle utilise la valeur par défaut.

Retourner des Valeurs :

Utilisez la **fonction return()** pour permettre à une fonction de renvoyer un résultat.

```
# Définition d'une fonction avec deux paramètres et une valeur par défaut
concatener_noms <- function(prenom, nom =
"Doe") {
  # Concaténation des prénom et nom
  nom_complet <- paste(prenom, nom)
  # Retourne le nom complet
  return(nom_complet)
}
# Appel de la fonction avec un seul argument
resultat1 <- concatener_noms("John")
print(resultat1)
# Appel de la fonction avec deux arguments
resultat2 <- concatener_noms("Mohamed", "EL
BAKKALI")
print(resultat2)
```

Fonctions en R

Install packages datasets

```
install.packages("skimr")
```

```
library(skimr)
```

```
skim(data)
```

→ Similar to describe in Python



CHAPITRE 4

VULGARISER LE LANGAGE R

1. Introduction à R et son rôle en statistique
2. Préparation de l'environnement
3. Syntaxe de base et principaux opérateurs en R
- 4. Structures de données fondamentales en R**
5. Premières manipulations de données en R
6. Comparaison des fonctions de base avec Python
7. Reprise des dernières applications (Python/Excel) en R

Introduction

- En R, les structures de données servent à organiser et stocker l'information
- Tout en R est un objet
- Les vecteurs, matrices, listes et data frames sont des exemples clés. Ils offrent une flexibilité essentielle pour manipuler efficacement les données dans le langage R
- Chaque structure est conçue en fonction de besoins particuliers
- Parmi les structures de données les plus utilisées, on trouve:

Vecteurs

Listes

Data Frames

Matrices

Arrays



Résumé des structures de données en R

| Structure de Données | Description | Exemple de Création |
|----------------------|--|--|
| Vecteurs | Liste ordonnée d'éléments du même type. | <code>vecteur <- c(1, 2, 3, 4, 5)</code> |
| Listes | Collection ordonnée d'éléments hétérogènes. | <code>liste <- list("John", 25, TRUE)</code> |
| Matrices | Tableau bidimensionnel d'éléments du même type. | <code>matrice <- matrix(1:9, nrow = 3, ncol = 3)</code> |
| Arrays | Tableau multidimensionnel d'éléments du même type. | <code>array <- array(1:8, dim = c(2, 2, 2))</code> |
| Data Frames | Tableau rectangulaire de données hétérogènes. | <code>data_frame <- data.frame(Nom = c("John", "Jane"), Age = c(25, 30), Marié = c(TRUE, FALSE))</code> |

Ces structures de données offrent différentes façons de stocker et organiser l'information en fonction des besoins spécifiques de l'analyse de données.

Introduire les vecteurs

- Les vecteurs sont des séquences ordonnées d'éléments du même type
- Ils peuvent être créés à l'aide de la fonction **c()** et sont utiles pour stocker des valeurs simples

Création de Vecteurs :

```
```{r }
villes_marocaines <- c('Agadir', 'Tanger', 'Tinghir', 'Casablanca', 'Rabat')
nouveaux_taux_tva <- c(0.20, 0.17, 0.16, 0.18)
print(villes_marocaines)
print(nouveaux_taux_tva)
```
```

```
[1] "Agadir"      "Tanger"       "Tinghir"       "Casablanca"   "Rabat"
[1] 0.20 0.17 0.16 0.18
```

Fonctions et opérations sur les vecteurs

Longueur d'un Vecteur :

```
longueur_villes <- length(villes)
```

Accès aux Vecteurs :

```
premiere_ville <- villes[1]
```

Accédez à plusieurs éléments par indice :

```
villes_selectionnes <- villes[c(1, 3)]
```

Accédez à tous les éléments sauf le premier :

```
villes_restants <- villes[c(-1)]
```

Modification d'un Élément :

```
villes[1] <- "Kenitra"
```

Génération de Vecteurs Séquencés :

```
nombres_seq_personnalise <- seq(from = 0, to = 100, by = 20)
```

Répétition des Vecteurs :

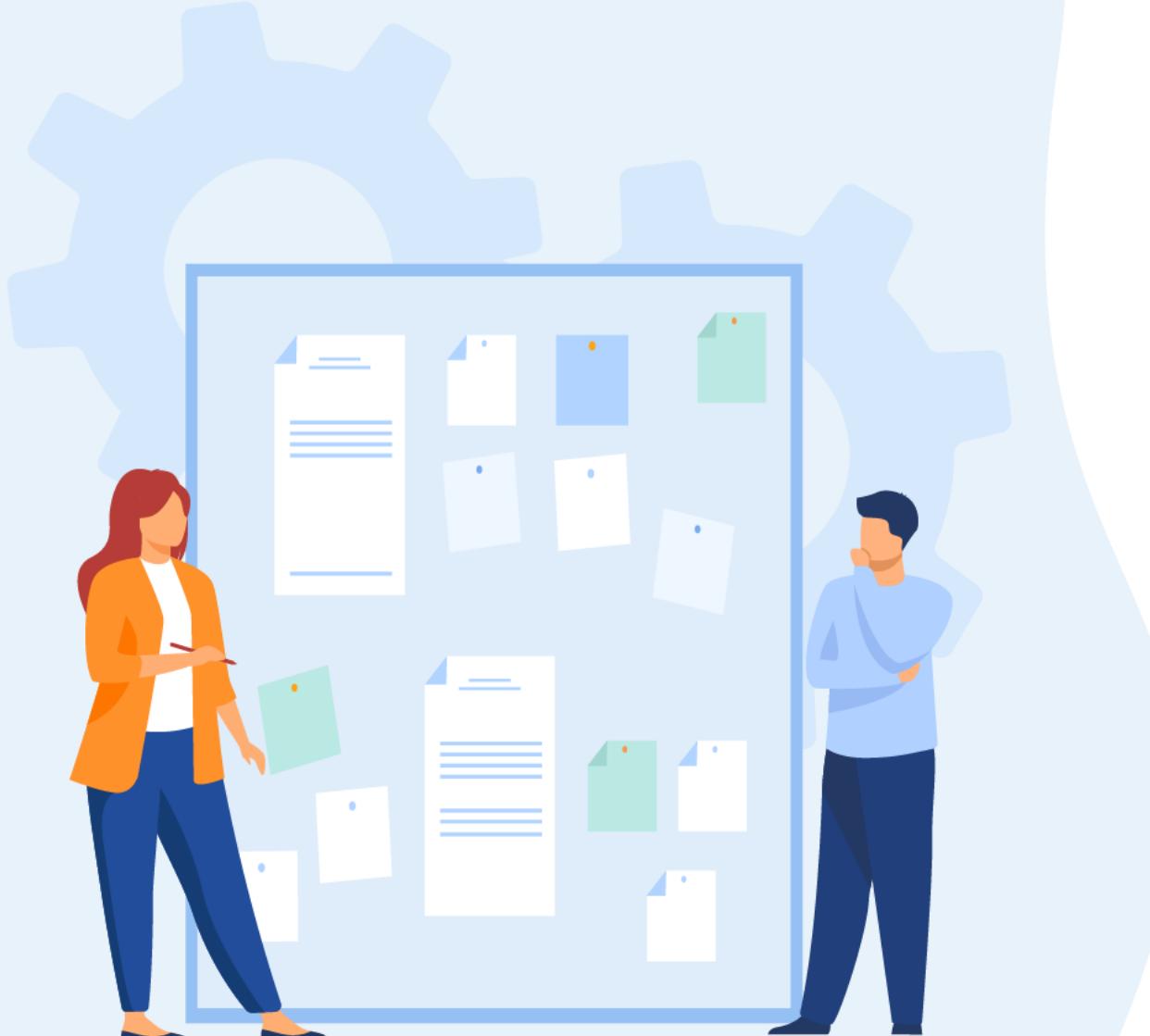
Répétez les valeurs de différentes manières :

```
repeter_chaque <- rep(c(1, 2, 3), each = 3)
```

```
repeter_n_fois <- rep(c(1, 2, 3), times = 3)
```

```
repeter_independant <- rep(c(1, 2, 3), times = c(5, 2, 1))
```

Autres fonctions et opérations sur les vecteurs peuvent être trouvées dans la section suivante, 'Premières manipulations de données en R'



CHAPITRE 4

VULGARISER LE LANGAGE R

1. Introduction à R et son rôle en statistique
2. Préparation de l'environnement
3. Syntaxe de base et principaux opérateurs en R
4. Structures de données fondamentales en R
- 5. Premières manipulations de données en R**
6. Comparaison des fonctions de base avec Python
7. Reprise des dernières applications (Python/Excel) en R

Manipulation de texte

- La manipulation de texte en R avec les fonctions de chaînes de caractères permet de traiter les données de caractères, incluant des opérations comme **le comptage, l'extraction, la concaténation, le changement de cas, la recherche de motifs**, etc.
- Ces fonctions offrent une flexibilité précieuse pour travailler avec les chaînes de caractères en R



Fonctions de chaînes de caractères

| Fonction | Description | Exemple d'Utilisation | Sortie Attendue |
|-------------------|--|--|------------------------------|
| nchar() | Nombre de caractères dans une chaîne | chaine <- "Bonjour"; nchar(chaine) | 7 |
| substr() | Extraire une sous-chaîne de caractères | texte <- "Exemple"; substr(texte, 1, 4) | "Exem" |
| paste() | Concaténer des chaînes de caractères | nom <- "Alice"; age <- 25; paste(nom, age) | "Alice 25" |
| toupper() | Convertir une chaîne en majuscules | texte <- "bonjour"; toupper(texte) | "BONJOUR" |
| tolower() | Convertir une chaîne en minuscules | texte <- "Bonjour"; tolower(texte) | "bonjour" |
| paste0() | Concaténer sans espace | nom <- "John"; age <- 30; paste0(nom, age) | "John30" |
| strsplit() | Diviser une chaîne en éléments d'un vecteur | phrase <- "Bonjour à tous"; strsplit(phrase, " ") | list("Bonjour", "à", "tous") |
| gsub() | Remplacer des occurrences dans une chaîne | chaine <- "excellent"; gsub("e", "E", chaine) | "ExcellEnt" |
| grep() | Rechercher des motifs dans une chaîne | chaine <- c("chat", "chien", "oiseau"); grep("ch", chaine) | c(1, 2) |
| sub() | Remplacer la première occurrence dans une chaîne | chaine <- "excellent"; sub("e", "E", chaine) | "Excellent" |
| sprintf() | Formatage de chaînes de caractères | nom <- "Alice"; age <- 25; sprintf("Nom: %s, Age: %d", nom, age) | "Nom: Alice, Age: 25" |

Applications en manipulation de chaînes

1. Extraction de Sous-chaînes :

Exemple : Extraire les trois premiers caractères d'une chaîne

```
chaine <- "Analyse de Données"  
sous_chaine <- substr(chaine, 1, 3)  
cat("Sous-chaîne :", sous_chaine, "\n")
```

Sortie Attendue : Sous-chaîne : Ana

2. Concaténation de Chaînes :

Exemple : Concaténer le prénom et le nom avec un espace entre eux

```
prenom <- "John"; nom <- "Doe"  
nom_complet <- paste(prenom, nom, sep = " ")  
cat("Nom Complet :", nom_complet, "\n")
```

Sortie Attendue : Nom Complet : John Doe

3. Modification de la Casse :

Exemple : Convertir une chaîne en majuscules

```
texte <- "analyse de données"  
texte_majuscule <- toupper(texte)  
cat("Majuscules :", texte_majuscule, "\n")
```

Sortie Attendue : Majuscules : ANALYSE DE DONNÉES

4. Recherche:

Exemple : Compter toutes les occurrences de "R" dans une chaîne

```
texte <- "R est un langage pour l'analyse de données"  
motifs <- grep("R", strsplit(texte, " "))  
cat("Occurrences de 'R' :", motifs, "\n")
```

Sortie Attendue : Occurrences de 'R' : 1

Applications en manipulation de vecteurs

- La manipulation des vecteurs en R offre un éventail d'opérations pour traiter efficacement les données vectorielles. Ces opérations incluent **le tri, la recherche d'éléments uniques, la répétition, la modification sélective**, et bien plus encore

1. Filtrer le Vecteur :

Exemple : Utilisation de techniques avancées pour filtrer les éléments d'un vecteur

```
filtre_conditionnel <- villes[villes == "Tanger" | villes == "Rabat"]
```

2. Reverser un Vecteur :

Exemple : Inverser l'ordre des éléments d'un vecteur.

```
villes_inverses <- rev(fruits)
```

3. Trier un Vecteur ::

Exemple : Trie les éléments d'un vecteur.

```
villes_tries <- sort(villes)
```

4. Éléments Uniques :

Exemple : Trouver les éléments uniques dans un vecteur

```
elements_uniques <- unique(vecteur)
```

Applications en manipulation de vecteurs

5. Opérations Arithmétiques :

Exemple : Opérations arithmétiques sur un vecteur.

```
somme <- sum(nombres)  
max_val <- max(nombres)  
min_val <- min(nombres)
```

6. Statistiques Descriptives :

Exemple : Calcul de statistiques descriptives (**le moyenne, la médiane, l'écart-type**).

```
moyenne <- mean(nombres)  
mediane <- median(nombres)  
ecart_type <- sd(nombres)
```

- En comprenant les différentes fonctionnalités et opérations de R, qu'il s'agisse d'opérations arithmétiques ou de manipulations de chaînes de caractères, nous pouvons les combiner pour exploiter la puissance de R dans la manipulation des vecteurs.
- L'efficacité de cette manipulation dépend étroitement du contexte spécifique de l'analyse et du type de données que nous traitons.

Fonctions de manipulation de Structures

| Fonction | Description | Listes | Vecteurs | Tableaux | Matrices | Data Frames |
|------------|---|--------|----------|----------|----------|-------------|
| length() | Obtenir la longueur de l'objet | ✓ | ✓ | ✓ | ✓ | ✓ |
| str() | Afficher la structure de l'objet | ✓ | ✓ | ✓ | ✓ | ✓ |
| c() | Combiner des objets en un vecteur/liste | ✓ | ✓ | ✓ | ✓ | |
| unlist() | Aplatir une liste imbriquée | ✓ | | | | |
| sapply() | Appliquer une fonction à chaque élément | ✓ | | | | |
| lapply() | Appliquer une fonction à chaque élément d'une liste | ✓ | | | | |
| matrix() | Créer une matrice | | | ✓ | ✓ | |
| array() | Créer un tableau | | | ✓ | | |
| dim() | Obtenir les dimensions de l'objet | | | ✓ | ✓ | ✓ |
| colnames() | Obtenir ou définir les noms de colonnes | | ✓ | | ✓ | ✓ |
| rownames() | Obtenir ou définir les noms de lignes | | ✓ | ✓ | ✓ | |
| names() | Obtenir ou définir les noms (d'éléments de liste) | ✓ | ✓ | ✓ | ✓ | ✓ |
| apply() | Appliquer une fonction sur les marges | ✓ | ✓ | ✓ | ✓ | |
| colMeans() | Calculer les moyennes des colonnes | | | ✓ | ✓ | ✓ |
| rowMeans() | Calculer les moyennes des lignes | | | ✓ | ✓ | ✓ |
| summary() | Afficher un résumé statistique d'un objet | ✓ | ✓ | ✓ | ✓ | ✓ |
| View() | Ouvrir un éditeur interactif pour visualiser un objet | ✓ | ✓ | ✓ | ✓ | ✓ |

Applications en manipulation de matrices

La **matrice "notes"** représente les notes de trois étudiants dans trois matières différentes (Mathématiques, Sciences, Français). Les fonctions R sont utilisées pour explorer les données, calculer des statistiques descriptives, et obtenir des moyennes par matière et par étudiant.

```
# Création d'une matrice de notes d'étudiants
notes <- matrix(c(85, 90, 78, 92, 88, 75, 95, 89, 80), nrow = 3,
byrow = TRUE)
colnames(notes) <- c("Mathématiques", "Sciences", "Anglais")
rownames(notes) <- c("Étudiant1", "Étudiant2", "Étudiant3")

# Affichage de la matrice des notes
cat("Matrice des notes des étudiants:\n")
print(notes)

# Dimensions de la matrice
dimensions <- dim(notes)
cat("\nDimensions de la matrice:\n")
print(dimensions)

# Noms des colonnes et des lignes
cat("\nNoms des matières:\n")
print(colnames(notes))
cat("\nNoms des étudiants:\n")
print(rownames(notes))
```

```
# Longueur de la matrice
longueur_matrice <- length(notes)
cat("\nNombre total d'éléments dans la matrice:\n")
print(longueur_matrice)

# Utilisation de sapply pour calculer la moyenne par matière
moyennes_matieres <- sapply(notes, mean)
cat("\nMoyennes par matière:\n")
print(moyennes_matieres)

# Utilisation de lapply pour obtenir le résumé statistique par étudiant
resume_etudiants <- lapply(1:nrow(notes), function(x)
summary(notes[x, ]))
cat("\nRésumé statistique par étudiant:\n")
print(resume_etudiants)

# Utilisation de apply pour calculer la moyenne par étudiant
moyennes_etudiants <- apply(notes, 1, mean)
cat("\nMoyennes par étudiant:\n")
print(moyennes_etudiants)
```

Fonctions de création, affichage et manipulation Spécifiques au Data Frames

| Fonction | Description | Exemple d'application | Sortie |
|-----------------|---|---|--|
| data.frame() | Créer un data frame | df <- data.frame(nom = c("Jean", "Pierre", "Marie"), age = c(20, 25, 30)) | data frame avec les colonnes "nom" et "age" |
| names() | Obtenir ou définir les noms de colonnes | names(df) names(df) <- c("Nom", "Âge") | Affiche les noms de colonnes Définit les noms de colonnes |
| head() | Afficher les premières lignes | head(df) | Affiche les 6 premières lignes (par défaut) |
| tail() | Afficher les dernières lignes | tail(df) | Affiche les 6 dernières lignes (par défaut) |
| rbind() | Ajouter des lignes | df_new <- rbind(df, data.frame(nom = "Alice", age = 35)) | data frame avec une nouvelle ligne ajoutée |
| cbind() | Ajouter des colonnes | df_new <- cbind(df, ville = c("Paris", "Lyon", "Marseille")) | data frame avec une nouvelle colonne ajoutée |
| [-c(1), -c(1)] | Supprimer des lignes/colonnes | df_new <- df[-c(1), -c(1)] | data frame sans la première ligne et colonne |
| ncol() | Obtenir le nombre de colonnes | ncol(df) | Affiche le nombre de colonnes |
| nrow() | Obtenir le nombre de lignes | nrow(df) | Affiche le nombre de lignes |

Applications en manipulation de Data Frames

Reprends le même exemple avec les "notes" représentant les notes de trois étudiants dans trois matières différentes (Mathématiques, Sciences, Français). mais cette fois en utilisant un **Data Frame**.

```
# Création d'un data frame de notes d'étudiants
etudiants <- data.frame(
  Nom = c("Étudiant1", "Étudiant2", "Étudiant3"),
  Math = c(85, 90, 78),
  Sciences = c(92, 88, 75),
  Francais = c(95, 89, 80)
)

# Affichage du data frame des notes
cat("Data frame des notes des étudiants:\n")
print(etudiants)
```

Sortie Attendue :

```
Console Terminal × Background Jobs ×
R 4.3.2 · ~/ 
> # Affichage du data frame des notes
> cat("Data frame des notes des étudiants:\n")
Data frame des notes des étudiants:
> print(etudiants)
  Nom Math Sciences Francais
1 Étudiant1    85      92      95
2 Étudiant2    90      88      89
3 Étudiant3    78      75      80
```

Applications en manipulation de Data Frames

Ensuite, nous affichons les moyennes par matière et par étudiant à partir du data frame des notes des étudiants, suivi par les statistiques descriptives pour chaque matière.

```
# Moyennes par matière
moy_matieres <- colMeans(etudiants[, 2:4])
cat("\nMoyennes par matière:\n")
print(moy_matieres)

# Moyennes par étudiant
moy_etudiants <- rowMeans(etudiants[, 2:4])
cat("\nMoyennes par étudiant:\n")
print(moy_etudiants)

# Obtention des statistiques descriptives
stat_descriptives <- summary(etudiants[, 2:4])
cat("\nStatistiques descriptives:\n")
print(stat_descriptives)
```

Excluant les formats non numériques

Sortie Attendue :

```
Moyennes par matière:
> print(moy_matieres)
  Math Sciences Francais
84.33333 85.00000 88.00000

Moyennes par étudiant:
> print(moy_etudiants)
[1] 90.66667 89.00000 77.66667

Statistiques descriptives:
> print(stat_descriptives)
    Math          Sciences          Francais
Min. :78.00  Min. :75.00  Min. :80.0
1st Qu.:81.50 1st Qu.:81.5  1st Qu.:84.5
Median :85.00  Median :88.0  Median :89.0
Mean   :84.33  Mean   :85.0  Mean   :88.0
3rd Qu.:87.50 3rd Qu.:90.0  3rd Qu.:92.0
Max.   :90.00  Max.   :92.0  Max.   :95.0
```

Applications en manipulation de Data Frames

La fonction `summary()` :

- La fonction `summary()` en R fournit un résumé statistique descriptif des données.
- Lorsqu'elle est appliquée à une trame de données (data frame) ou à une matrice, elle retourne des statistiques clés pour chaque colonne.
- Ces statistiques comprennent généralement la valeur minimum, le premier quartile (1er Qu.), la médiane, la moyenne, le troisième quartile (3ème Qu.) et la valeur maximum.

| Math | Sciences | Français |
|---------------|--------------|--------------|
| Min. :78.00 | Min. :75.0 | Min. :80.0 |
| 1st Qu.:81.50 | 1st Qu.:81.5 | 1st Qu.:84.5 |
| Median :85.00 | Median :88.0 | Median :89.0 |
| Mean :84.33 | Mean :85.0 | Mean :88.0 |
| 3rd Qu.:87.50 | 3rd Qu.:90.0 | 3rd Qu.:92.0 |
| Max. :90.00 | Max. :92.0 | Max. :95.0 |



PARTIE 2

Visualiser les données décisionnelles

Dans ce module, vous allez :

- Créer divers types de graphiques
- Configurer les graphiques pour une analyse profonde



40 heures



CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

Ce que vous allez apprendre dans ce chapitre :

- Utiliser efficacement les scatter plots
- Utiliser efficacement les bar charts
- Utiliser efficacement les histograms
- Utiliser efficacement les box plots
- Utiliser efficacement les bubble charts
- Pratiquer la création de graphiques avec Python/Excel/R



30 heures



CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. Bar charts
3. Histograms
4. Box plots
5. Bubble charts
6. Création de graphiques avec Python/Excel/R

Introduction

Introduction sur la visualisation

La science de la visualisation des données consiste à représenter graphiquement les données. Elle permet de rendre les données complexes plus accessibles et compréhensibles, facilitant ainsi l'interprétation des tendances, et la détection des anomalies.

Concepts fondamentaux : Axes X et Y

Variable Indépendante (X) : La variable que vous **manipulez** ou contrôlez pour voir son effet sur une autre variable. Elle est placée sur l'axe des abscisses horizontal (X)

Variable Dépendante (Y) : La variable que vous **mesurez**, qui est influencée par la variable indépendante. Elle est placée sur l'axe des ordonnées vertical (Y)

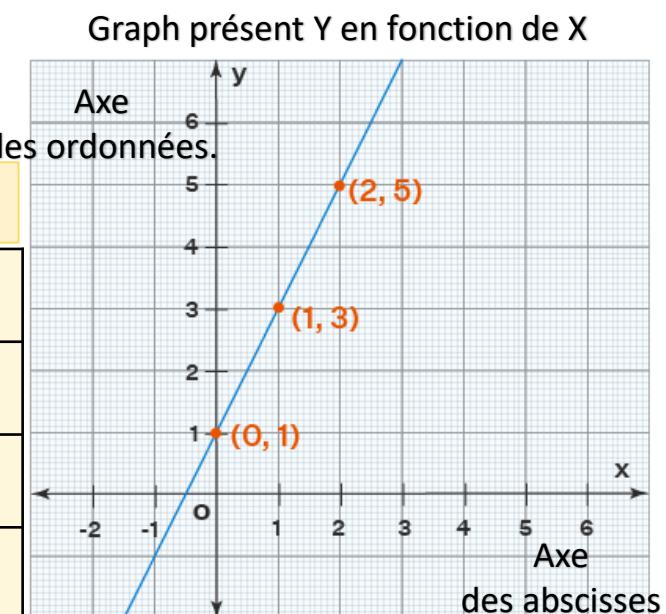
Tracer Y en fonction de X : En géométrie, une équation linéaire peut être représentée graphiquement à l'aide des axes X et Y sous forme de ligne droite.

Prenons un exemple pour mieux comprendre :

Considérons l'équation linéaire $y = 2x + 1$. Pour la tracer, construisons un tableau avec deux colonnes pour les valeurs de x et y.

Choisissons quelques valeurs pour la variable x et trouvons les valeurs correspondantes pour y.

| Données : | |
|-----------|---|
| x | y |
| 0 | 1 |
| 1 | 3 |
| 2 | 5 |



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Introduction

Exemple Pratique : Relation entre l'âge et le revenu

Formulation :

Nous étudions le **revenu (variable dépendante Y)** en fonction de l'**âge (variable indépendante X)**.

Cela signifie que nous observons comment les variations de l'âge (variable indépendante) affectent le revenu (variable dépendante).

Revenu en fonction de l'âge



Interprétation :

Dans ce diagramme de dispersion (Scatter Plot), chaque point représente un individu avec son âge sur l'axe X et son revenu sur l'axe Y. En observant ces points, on peut déterminer s'il existe une tendance ou une relation entre l'âge et le revenu.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Scatter plots

Scatter Plot (Diagramme de dispersion / Nuage de points) :

Un diagramme de dispersion est un type de graphique qui affiche des points de données individuels sur un axe bidimensionnel, montrant la relation entre deux variables.

Usage : diagramme de dispersion utilisé pour démontrer les **relations** entre deux variables (indépendante X, dépendante Y).

Caractéristiques :

- **Ordre important.**
- Les deux axes contiennent généralement des **valeurs numériques**.

Interprétation :

Cherchez des tendances (trends and patterns), des clusters et des valeurs aberrantes.

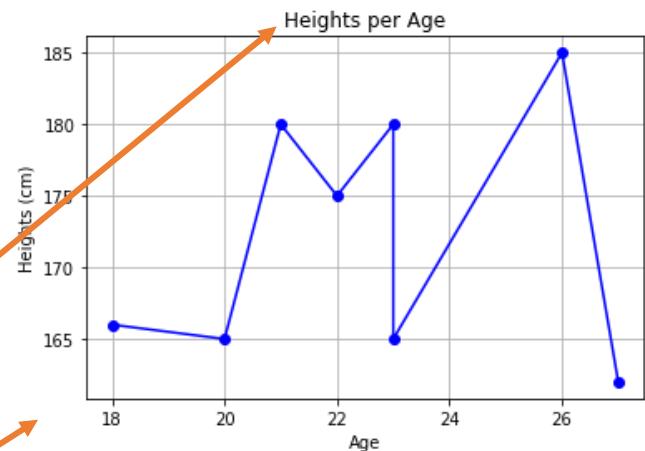
Par exemple, un motif ascendant pourrait indiquer une corrélation positive.

Exemple :

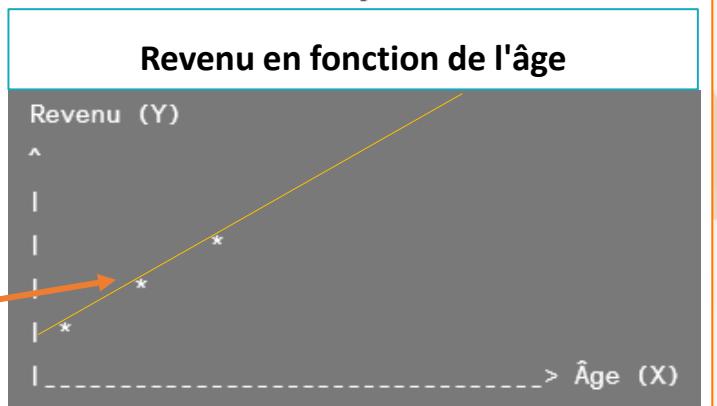
Tracer l'âge par rapport au revenu pour voir s'il y a une corrélation.

"heights per age" équivaut à dire "hauteurs en fonction d'âge". Cela signifie que les hauteurs sont représentées en fonction de l'âge.

Aucune corrélation



Corrélation positive



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Scatter plots

Scatter Plot (Diagramme de dispersion / Nuage de points) :

Exemple sur Tips Data :

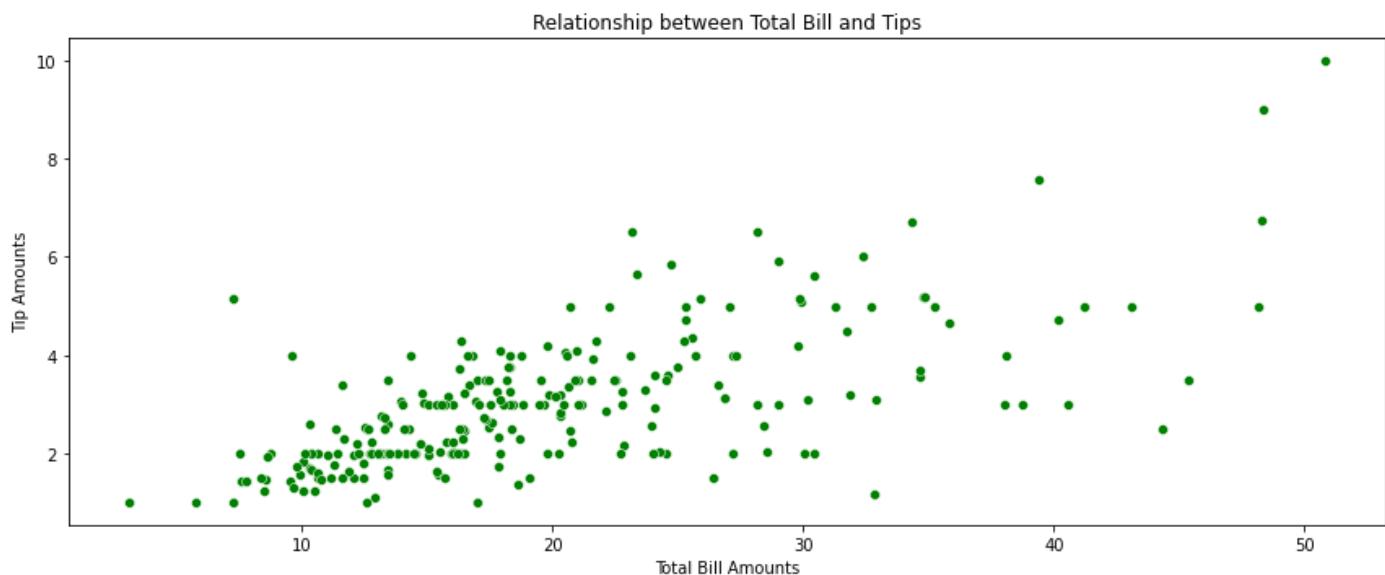
Tracer « Total bill » par rapport au « Tips » pour voir s'il y a une corrélation.

Indique que :

- Total bill sur l'axe X (axe horizontal).
- Tips sur l'axe Y (axe vertical).

Nous étudions les **Tips** en fonction de **Total bill**.

Il est courant de dire "y en fonction de x" pour indiquer que la variable y dépend de la variable x



Dans ce diagramme de **dispersion**, chaque point représente une facture totale et le pourboire correspondant. En observant la disposition des points, vous pouvez déterminer s'il existe une relation entre le montant total de la facture et le montant des pourboires. Si les points suivent une tendance ascendante, cela indique une **corrélation positive** où des factures plus élevées sont associées à des pourboires plus élevés.



CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. **Bar plots**
3. Histograms
4. Box plots
5. Bubble charts
6. Création de graphiques avec Python/Excel/R

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Bar Plot

Bar Plot (Graphique à barres)

Un diagramme à barres est un graphique qui représente les catégories de données avec des barres rectangulaires dont les longueurs et hauteurs sont proportionnelles aux valeurs qu'elles représentent.

Usage: Utilisé pour **comparer** différentes catégories ou groupes, souvent pour visualiser des données catégorielles.

Caractéristiques :

- **Ordre pas important.**
- Un des axes contiennent habituellement des **valeurs catégoriques**.

Types: Barres verticales, barres horizontales.

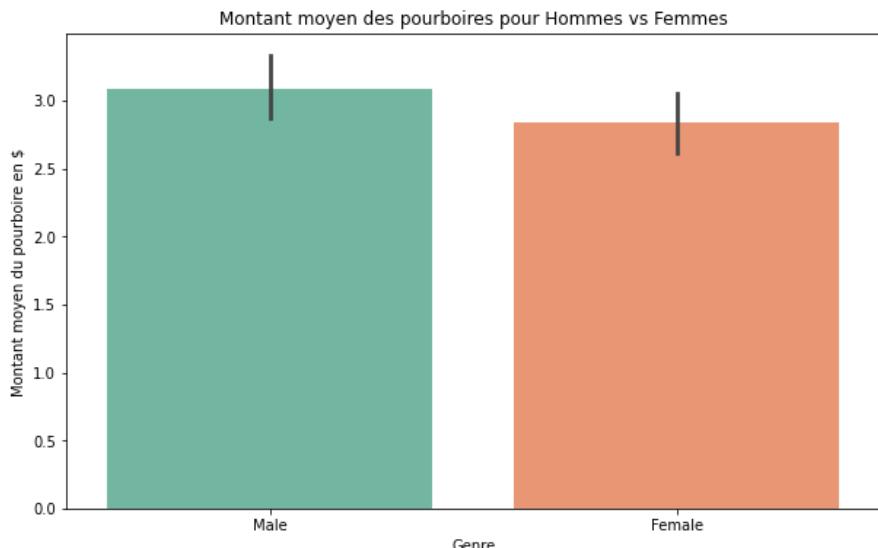
Interprétation:

Comparez les longueurs des barres pour comprendre les différences entre les catégories.

Exemples:

Comparer les chiffres de vente dans différentes régions.

Ou comparez les montants des pourboires entre hommes et femmes dans un restaurant.



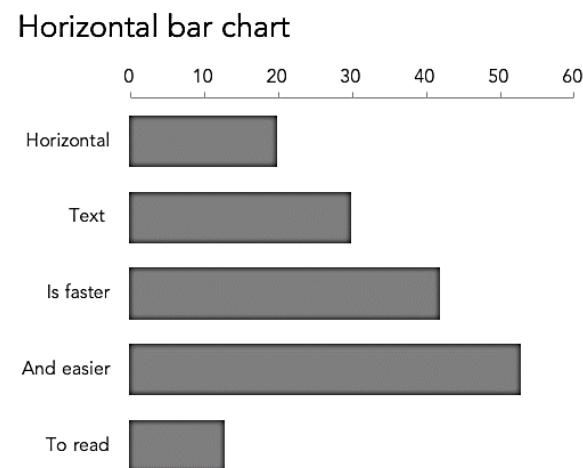
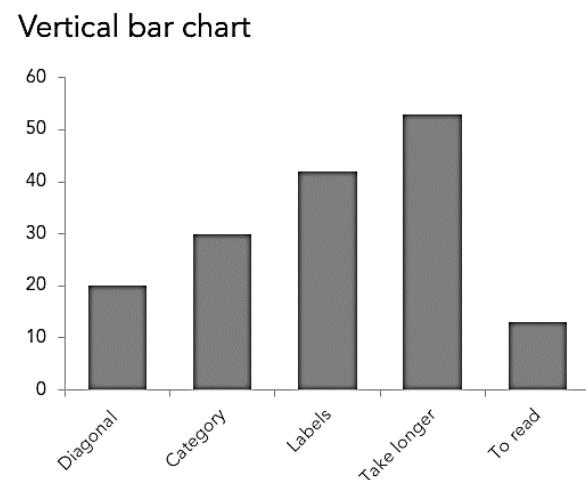
Source : Application au jeu de données Tips de la section précédente

Bar Chart

Bar Chart (Histogramme à barres)

Visualisation Catégorielle et Comparaison :

- Les diagrammes à barres sont couramment utilisés pour représenter graphiquement des données **catégorielles** et les **comparer** entre elles. Ils offrent une visualisation claire des relations et des distinctions entre différentes catégories, permettant une interprétation rapide et précise des données.
- Chaque barre représente une catégorie distincte et sa hauteur est proportionnelle à la valeur associée à cette catégorie.
- Les barres peuvent être agencées **horizontalement** ou **verticalement** en fonction de la préférence ou de la lisibilité du graphique.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

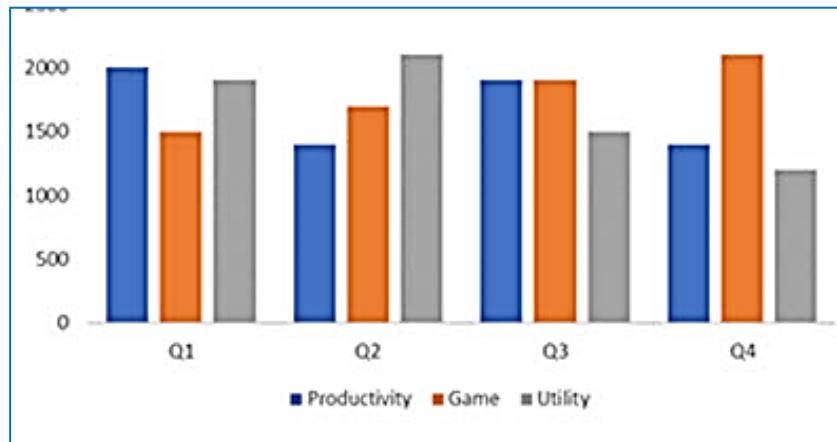
Bar Chart

Bar Chart (Histogramme à barres)

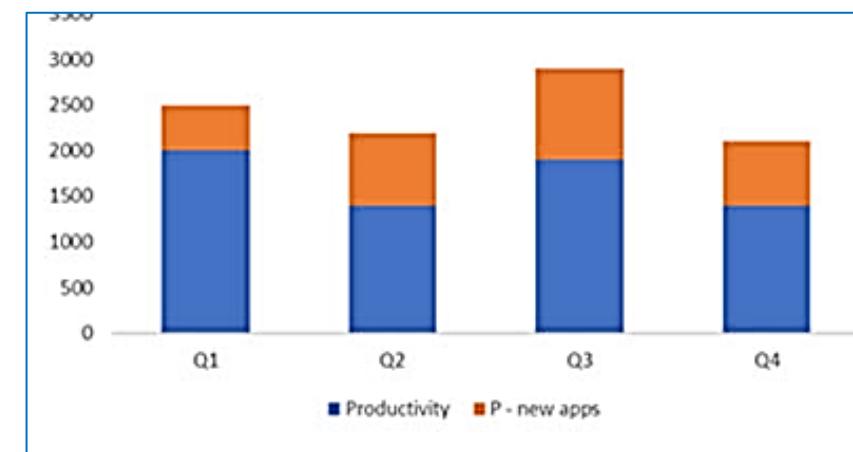
Visualisation Catégorielle et Comparaison :

- Les colonnes empilées ‘Stacked’ et groupées ‘Clustered’ dans un graphique à barres permettent de comparer des catégories et des sous-catégories de données.
- Les colonnes empilées ‘Stacked’ montrent la contribution de chaque sous-catégorie à l'ensemble, tandis que les colonnes groupées ‘Clustered’ permettent de comparer directement les valeurs des sous-catégories entre elles.

Clustered Column Chart



Stacked Column Chart



Les deux graphiques montrent la productivité d'une entreprise au cours des trimestres de l'année. Le diagramme à barres groupées « Clustered » compare les éléments au sein de chaque groupe, tandis que le diagramme à barres empilées « Stacked » facilite les comparaisons entre les groupes les uns par rapport aux autres pour chaque trimestre.



CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. Bar charts
- 3. Histograms**
4. Box plots
5. Bubble charts
6. Création de graphiques avec Python/Excel/R

Histograms

Histogram (Histogramme)

Un histogramme est un graphique qui représente la distribution d'un ensemble de données numériques à l'aide de barres continues dont la hauteur correspond à la fréquence des valeurs dans chaque intervalle.

Usage : L'Histogramme permet de visualiser la **distribution** d'un ensemble de données numériques. Il montre comment les valeurs sont réparties et permet d'identifier la forme de la distribution, la dispersion des données et leur relation par rapport à la moyenne.

Caractéristiques :

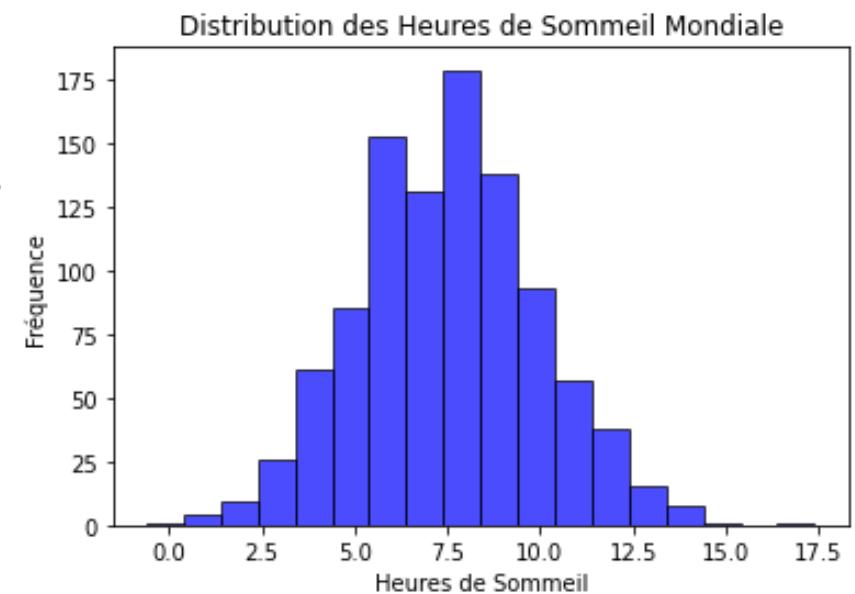
- Barre continue et **l'ordre est important**
- Les deux axes d'un histogramme contiennent habituellement des **valeurs numériques** .

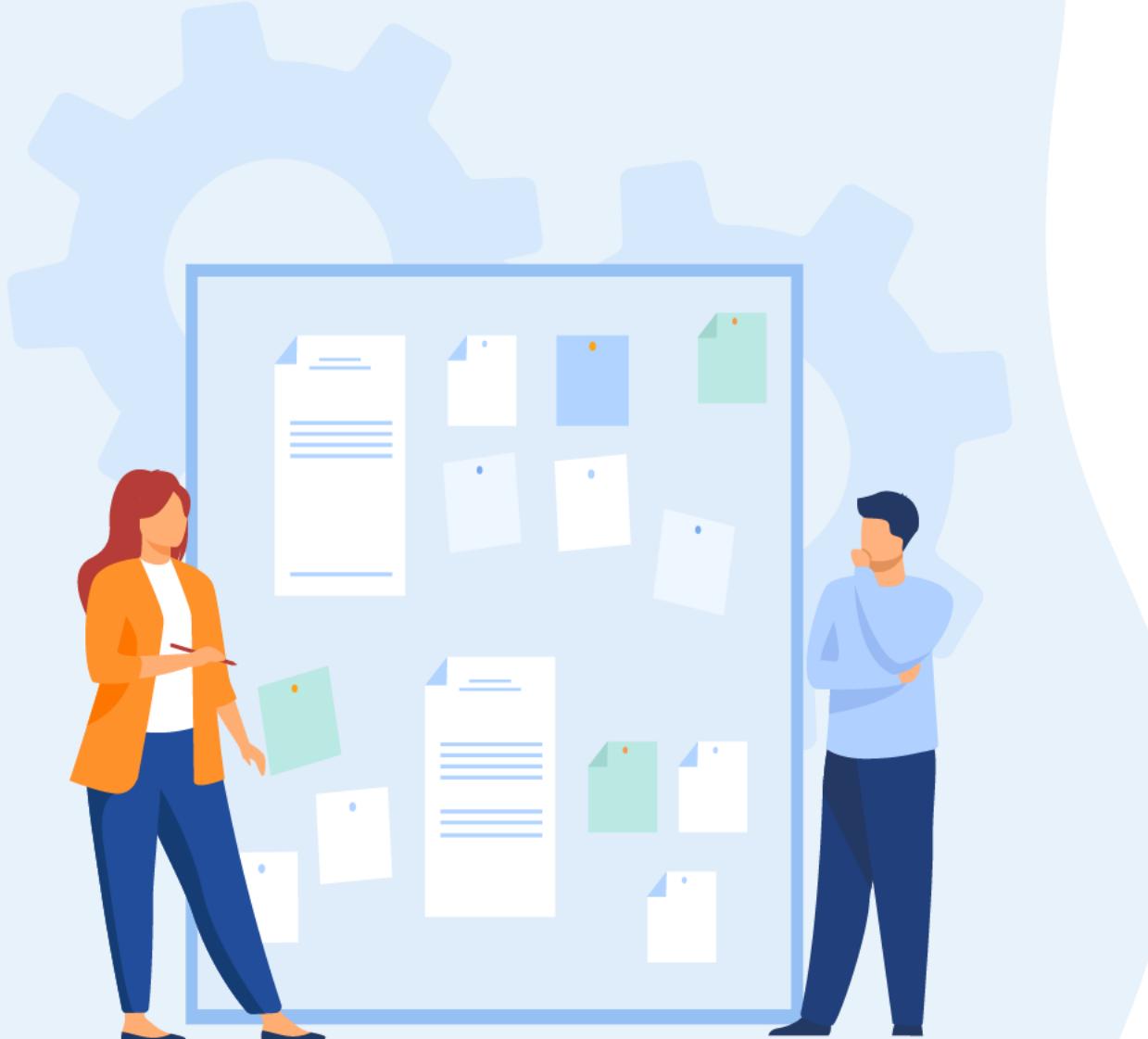
Interprétation : Observez la forme de l'histogramme pour comprendre la distribution (par exemple, distribution normale, uniform, asymétrie..).

Exemple : Montrer la fréquence des heures de sommeil d'un échantillon .

Cela reflète bien que :

l'axe X représente les **intervalles** de valeurs des données,
et l'axe Y représente la **fréquence** des données dans ces intervalles.





CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. Bar charts
3. Histograms
- 4. Box plots**
5. Bubble charts
6. Création de graphiques avec Python/Excel/R

Box plot

Box Plot (Boîte à moustaches)

Le Boxplot est un graphique résumant la distribution de données.

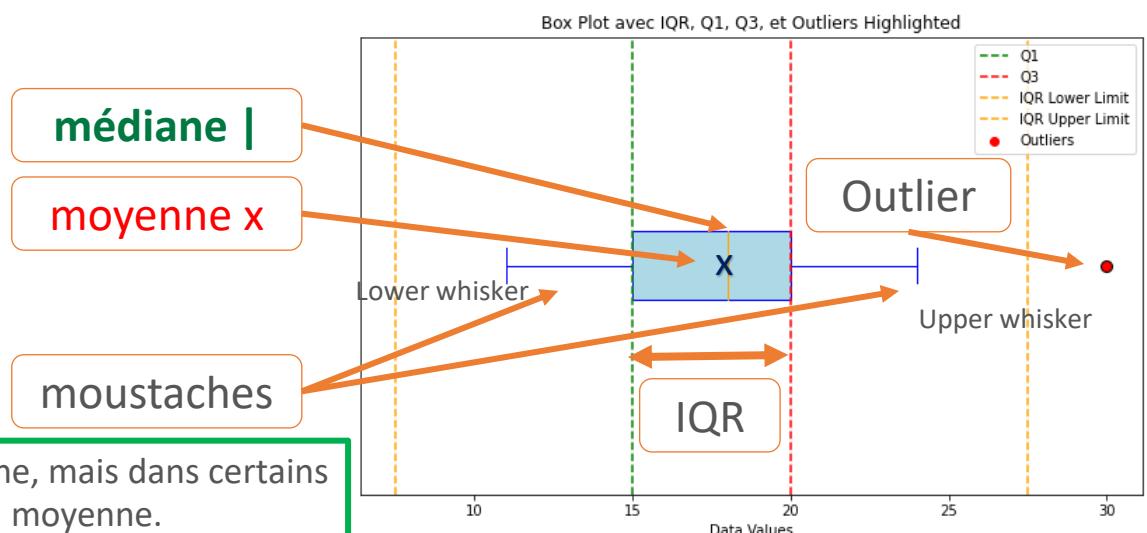
Usage : Visualisez les distributions en affichant la médiane, les quartiles supérieurs et inférieurs où se concentre la majorité des données (50 %), ainsi que les moustaches supérieures et inférieures indiquant les valeurs min et max, permettant aussi d'identifier les valeurs aberrantes, comme expliqué dans la partie précédente du cours qui étudie les valeurs aberrantes.

Caractéristiques :

- "Moustaches" s'étendant aux valeurs minimale et maximale, à l'exclusion des valeurs aberrantes.
- Boîte montrant l'intervalle interquartile (Q1 25% à Q3 75% quartiles).
- Ligne à l'intérieur de la boîte représentant la médiane.

Exemple : Résumer la distribution des revenus dans une population.

Interprétation : Analysez la dispersion et identifiez les valeurs aberrantes.



Normalement Les box plots affichent la médiane, mais dans certains cas (Excel), un petit 'x' peut faire référence à la moyenne.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Box plot

Box Plot (Boîte à moustaches) : Points clés

- La partie **box** d'un box-and-whisker plot couvre les 50% centraux des valeurs dans l'ensemble de données.
- Les whiskers couvrent chacun 25% des valeurs des données.
 - Le **Lower whisker (inférieur)** couvre toutes les valeurs des données depuis la valeur minimale jusqu'à Q1, c'est-à-dire les 25% les plus bas des valeurs des données.
 - L' **Upper whisker (supérieur)** couvre toutes les valeurs des données entre Q3 et la valeur maximale, c'est-à-dire les 25% les plus élevés des valeurs des données.
- La **médiane** se situe à l'intérieur de la box et représente le centre des données. 50% des valeurs des données se trouvent au-dessus de la médiane et 50% se trouvent en dessous de la médiane.
- Les **outliers**, ou valeurs extrêmes, dans un ensemble de données sont généralement indiqués sur un box-and-whisker plot par des points individuels en dehors des moustaches.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Box plot

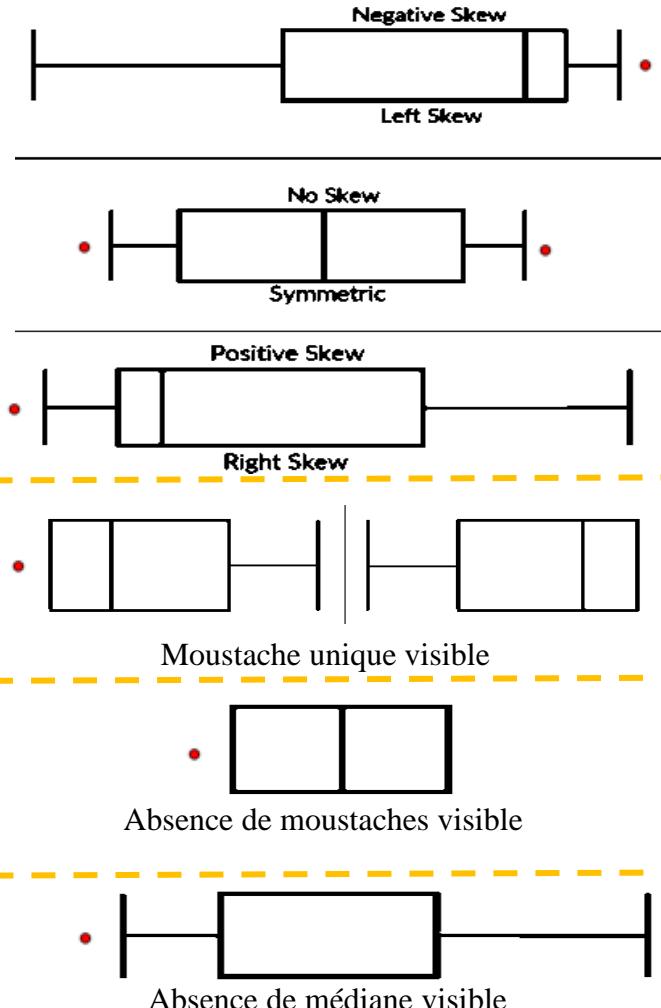
Box Plot (Boîte à moustaches) : Evaluation de la symétrie de la distribution

Skewness (Symétrie) :

- **Asymétrie négative** : si la médiane proche de la fin et dernière moustache plus courte.
- **Symétrique** : si la ligne médiane est au centre.
- **Asymétrie positive** si la médiane proche du début et première moustache plus courte.

L'absence d'un élément dans une Box plot peut indiquer différentes situations :

- **Moustache unique au bord de la boîte** : Cela se produit lorsque **la valeur minimale est égale au premier quartile (Q1) ou si la valeur maximale est égale au troisième quartile (Q3)**. Dans ce cas, une moustache semble se chevaucher avec le bord de la boîte et n'est donc pas visible.
- **Absence de moustaches** : Si **la valeur minimale est égale à Q1 et la valeur maximale est égale à Q3**, le box plot peut sembler ne pas avoir de moustaches. Cela donne l'apparence d'une simple boîte sans extensions, car les moustaches se chevauchent avec la boîte.
- **Absence de ligne médiane visible** : Si la **ligne médiane est égale à Q1 ou Q3**, elle peut se chevaucher avec les bords de la boîte, rendant la ligne médiane invisible.





CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. Bar charts
3. Histograms
4. Box plots
5. **Bubble charts**
6. Création de graphiques avec Python/Excel/R

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Bubble charts

Bubble Chart (Graphique à bulles)

Les graphiques à bulles, une sous-catégorie des Scatter plots , permettent de visualiser des données multivariées. Dans ce type de visualisation, les nuages de points peuvent représenter la relation entre plusieurs variables en colorant les points ou en modifiant leur forme ou leur taille.

Usage : Utilisé pour démontrer les **relations** entre trois variables (X, Y, taille de la bulle).

Caractéristiques:

Combinaison de nuages de points avec une dimension supplémentaire représentée par la taille de la bulle.

Permet de visualiser les interactions complexes entre trois variables.

Interprétation: Comparez les bulles pour comprendre les relations et les tailles relatives.

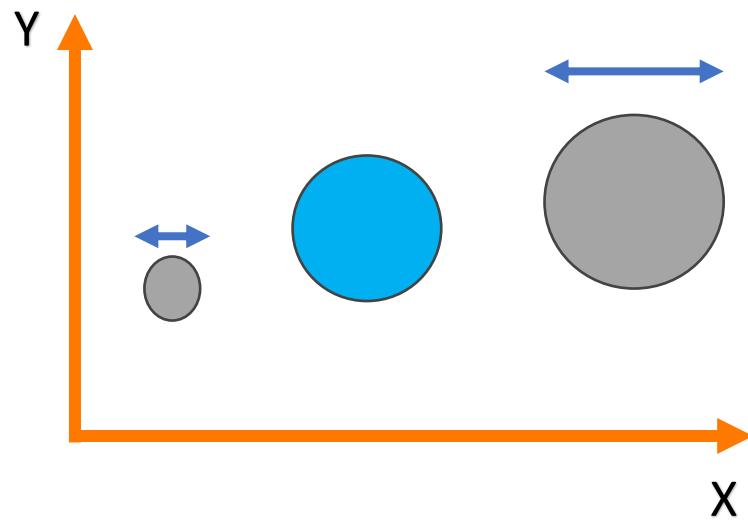
Exemples:

Tracer les ventes (X), les villes (Y) et la part de marché (taille de la bulle).

Tracer le GDP par habitant (X), l'espérance de vie (Y) et la population (taille de la bulle).

*GDP: Gross Domestic Product (En)

*PIB : Produit Intérieur Brut (Fr)



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Pie charts



Titre

- Pie charts : used to show compositions, often used to visualise categorical data for example 46% from the population voted for yes while 56% Voted No
- `df.query("year==2007")`, j=hadî khes t3awed fe tps le ôter graph show just ,,,filer by other col



CHAPITRE 1

CRÉER DIVERS TYPES DE GRAPHIQUES

1. Scatter plots
2. Bar charts
3. Histograms
4. Box plots
5. Bubble charts
6. **Création de graphiques avec Python/Excel/R**

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R



Création de graphiques avec Python : Scatter Plot (Diagramme de dispersion / Nuage de points)

Exemple sur Tips Data :

Pour visualiser la relation entre le montant total de la facture et les pourboires, nous allons utiliser les bibliothèques Seaborn et Matplotlib en Python. **Seaborn** simplifie la création de graphiques esthétiques, tandis que **Matplotlib** permet une personnalisation fine des visualisations.

Voici comment procéder :

```
# Importer les bibliothèques pour la visualisation et le chargement des données :  
  
import seaborn as sns  
import matplotlib.pyplot as plt  
# Charger le jeu de données Tips depuis seaborn  
tips = sns.load_dataset("tips")
```

Seaborn est principalement utilisé pour la visualisation des données, même s'il contient quelques fonctions pour charger des jeux de données d'exemple.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Scatter Plot (Diagramme de dispersion / Nuage de points)

Exemple sur Tips Data :

```
# Créer une figure
plt.figure(figsize=(10, 4))

# Tracer le diagramme de dispersion
sns.scatterplot(data=tips, x='total_bill',
y='tip', color='green')

# Ajouter le titre
plt.title('Relation entre Total Bill et Tips')

# Étiqueter l'axe X
plt.xlabel('Montants Total Bill')
# Étiqueter l'axe Y
plt.ylabel('Montants Tips')

# Ajouter une grille
plt.grid(True)

# Afficher le graphique
plt.show()
```

- Initialisez une figure avec des dimensions adaptées, avec figsize=(width ,height). Utilisez Seaborn « sns.scatterplot() » pour tracer vos données : spécifiez la source des données, les axes X et Y, et les couleurs appropriées.
- Utilisez Matplotlib pour définir un titre descriptif et des étiquettes clairs pour le graphique.
 - Ajouter une grille pour améliorer la lisibilité du graphique en fournissant des repères visuels pour les valeurs des axes X et Y.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Scatter Plot (Diagramme de dispersion / Nuage de points)

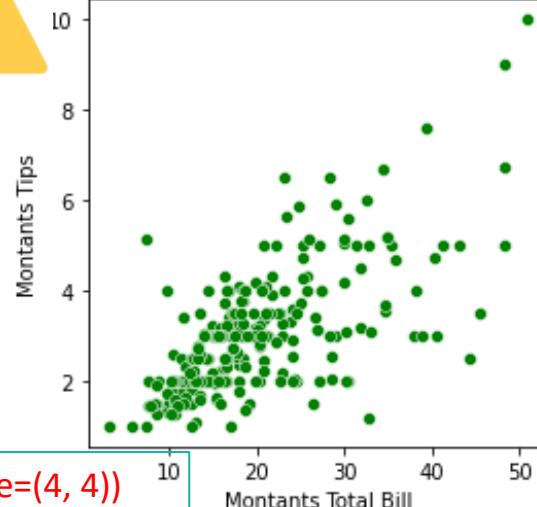
Recommandation :

La spécification de la taille du graphique est importante car elle peut intentionnellement affecter l'interprétation des données.

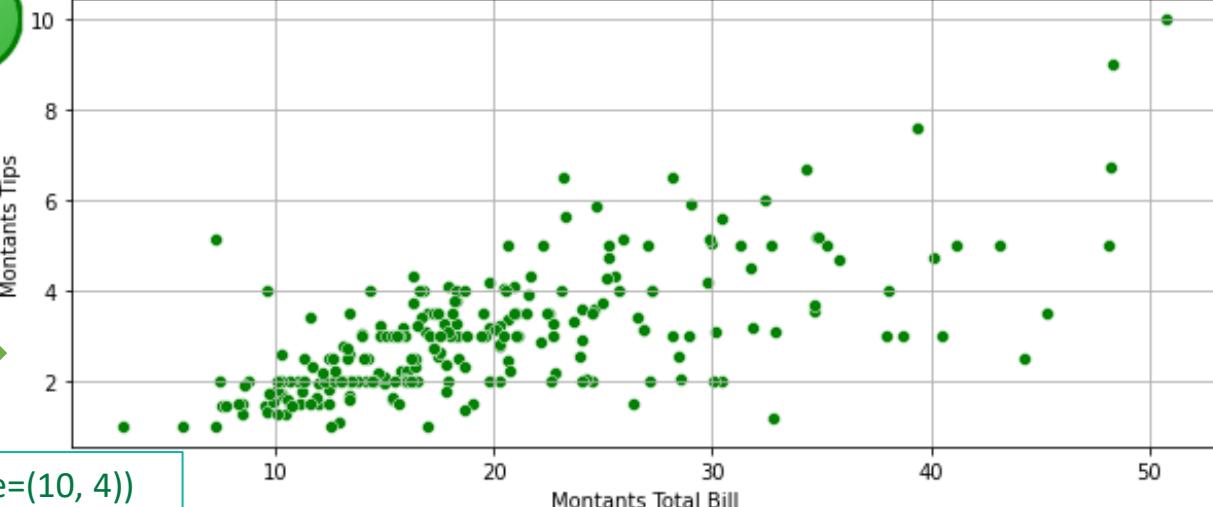
La taille du graphique doit être **proportionnelle** aux données pour une interprétation précise. Par exemple, si le pourboire va jusqu'à **10\$** et le total de la facture jusqu'à **50\$ (même unité '\$')**, nous avons choisi une hauteur de figure plus petite pour les pourboires afin de maintenir cette proportionnalité, assurant une représentation équilibrée des données et une analyse précise de leur relation.



Relation entre Total Bill et Tips



Relation entre Total Bill et Tips



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R



Création de graphiques avec Python : Bar Chart (Histogramme à barres)

À ce stade, vous devez réaliser que chaque graphique comporte un titre, une source de données et peut avoir des axes. L'initialisation de ces éléments en Python est similaire au graphique précédent. Maintenant, pour la création d'un diagramme à barres, la tâche devient très simple : Nous utiliserons **la fonction barplot() de seaborn** pour la création d'un Bar Chart.

Voici les étapes à suivre :

- Importer les bibliothèques de visualisation et de chargement des données et charger le jeu de données Tips depuis Seaborn.
- Créer un graphique en barres pour comparer les pourboires entre Hommes et Femmes et initialiser une figure avec une taille de 10x6 pouces.
- Utiliser Seaborn pour tracer un graphique en barres de tips avec le Genre sur l'axe X et pourboire sur l'axe Y, et choisir une palette de couleurs 'Set2'
- Définir le titre, les étiquettes et Afficher le graphique

```
import matplotlib.pyplot as plt
import seaborn as sns

# Charger de données
tips = sns.load_dataset("tips")

# Créer un graphique en barres
plt.figure(figsize=(10, 6))
sns.barplot(x='sex', y='tip', data=tips, palette='Set2')

# Définir le titre et les étiquettes
plt.title('Montant moyen des pourboires pour Hommes vs Femmes')
plt.xlabel('Genre')
plt.ylabel('Montant moyen du pourboire en $')

# Afficher le graphique
plt.show()
```

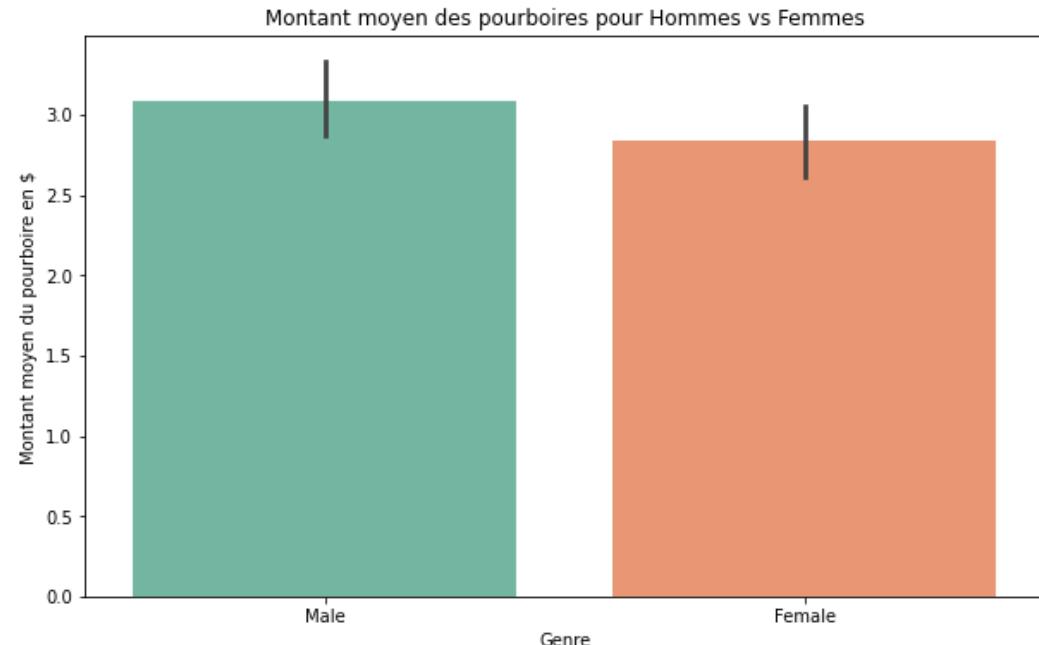
01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bar Chart (Diagramme à barres)

Pour lire le graphique de l'application précédente :

Ce graphique montre le montant moyen des pourboires (axe Y) **en fonction** du genre (axe X). Les barres montrent la moyenne des pourboires donnés par les hommes et les femmes, respectivement.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R



Création de graphiques avec Python : Histogram

La génération de la distribution des Heures de Sommeil Mondiale :

En tant qu'application de création d'histogramme, nous reviendrons pour créer l'histogramme précédent qui montre la distribution des heures de sommeil mondiale, qui a été créé pour mettre en valeur l'effet de différentes variances sur les données.

Voici les étapes à suivre :

1- Génération de données :

Générer un ensemble de données simulant les heures de sommeil dans le monde, en supposant une moyenne de 7,5 heures et un écart type de 2,5 heures.

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

# Générer une distribution plus réaliste des heures de sommeil dans le monde
np.random.seed(42) # Fixer la graine pour la reproductibilité
mean_sleep_hours = 7.5 # Supposons une moyenne de 7,5 heures
std_deviation = 2.5 # Supposons un écart type de 2,5 heures

# Générer un ensemble de données
sleep_hours = np.random.normal(mean_sleep_hours,
std_deviation, 1000)
```

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

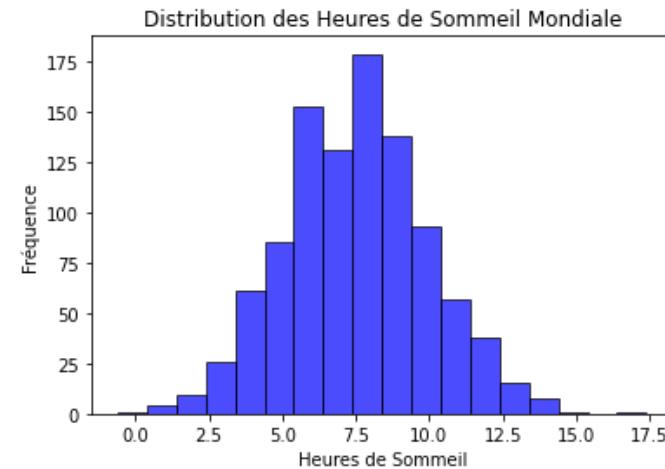
Création de graphiques avec Python : Histogram

La génération de la distribution des Heures de Sommeil Mondiale :

2- Création de l'histogramme :

```
# Visualisation graphique
plt.hist(x=sleep_hours,
          bins=np.arange(min(sleep_hours),
                         max(sleep_hours) + 1, 1),
          alpha=0.7, color='blue', edgecolor='black')

# Créer l'histogramme avec des barres de Largeur 1 heure
# Ajouter un titre au graphique
plt.title('Distribution des Heures de Sommeil Mondiale')
# Étiqueter l'axe des X
plt.xlabel('Heures de Sommeil')
# Étiqueter l'axe des Y
plt.ylabel('Fréquence')
# Afficher le graphique
plt.show()
```



bins : Définir les intervalles des barres de l'histogramme. Et crée des bacs d'une largeur de 1 heure, couvrant toutes les valeurs de sommeil observées.

Il est possible de créer un histogramme sans spécifier explicitement le paramètre bins. Si vous ne fournissez pas ce paramètre, Matplotlib choisira automatiquement un nombre approprié de bacs en fonction des données.

alpha : pour ajuster la transparence des barres.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R



Création de graphiques avec Python : Histogram

Paramètres principaux de plt.hist() :

x : Les données que vous souhaitez visualiser sous forme d'histogramme.

bins : Le nombre de bacs (ou barres) à utiliser dans l'histogramme.

Peut-être **un entier ou une séquence** définissant les bornes des bacs.

range : La plage de valeurs à inclure dans l'histogramme (tuple de deux valeurs).

density : Si True, l'aire sous l'histogramme sera normalisée à 1 (produit une estimation de la densité).

weights : Une séquence de poids, de la même longueur que x. Utilisé pour pondérer les valeurs dans x.

cumulative : Si True, produit un histogramme cumulatif.

Autrement dit, les hauteurs des barres de l'histogramme ne représentent plus le nombre de données dans chaque intervalle (ou bac), mais plutôt la densité des données. C'est utile pour comparer des distributions avec des échantillons de tailles différentes ou pour visualiser la fonction de densité de probabilité des données.

```
# Exemple :
plt.hist(x=sleep_hours,
          bins=my_bins,
          range=(0, 15),
          density=False,
          weights=None,
          cumulative=False,
          histtype='bar',
          align='mid',
          orientation='vertical',
          rwidth=0.9,
          log=False,
          color='blue',
          label='Heures de sommeil',
          stacked=False)
```

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Histogram

Paramètres principaux de plt.hist() : Suit

histtype : Le type d'histogramme à tracer ('bar', 'barstacked', 'step', 'stepfilled').

align : L'alignement des barres ('left', 'mid', 'right').

orientation : L'orientation des barres ('vertical', 'horizontal').

rwidth : La largeur relative des barres en pourcentage de la largeur des bacs.

log : Si True, l'axe Y sera en échelle logarithmique.

color : La couleur des barres de l'histogramme.

label : L'étiquette de la légende pour cet histogramme.

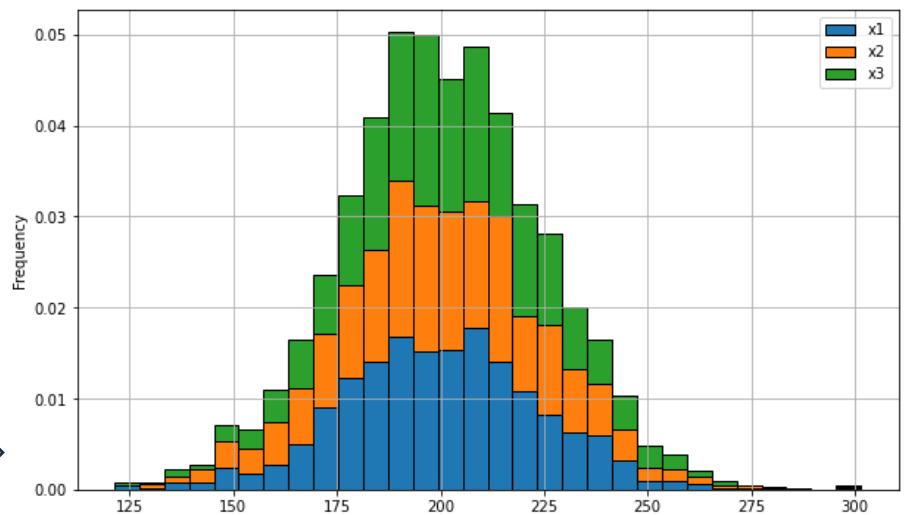
stacked : Si True et plusieurs données sont fournies, les histogrammes seront empilés.

Un histogramme empilé « **stacked Histogram** » est un graphique montrant la distribution de données en catégories, où les segments colorés empilés représentent différentes sous-catégories, permettant de visualiser à la fois la distribution totale et les contributions individuelles.



Pour créer un histogramme avec Matplotlib en utilisant la fonction plt.hist(), le seul paramètre vraiment nécessaire est le **x** « Les données », qui représente les données à visualiser. Les autres paramètres sont facultatifs.

Stacked Histogram



condition : les variables doivent avoir la même unité

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Box plot

Application sur CarsMPG data set :

Pour créer un box plot utilisant le dataset des voitures (mpg) et visualiser la consommation de carburant (miles per gallon) en fonction du nombre de cylindres, nous allons suivre les mêmes étapes que précédemment. Nous utiliserons la bibliothèque Seaborn pour créer le graphique et Matplotlib pour le personnaliser et l'afficher.

Description du Jeu de Données mpg: Le jeu de données mpg (miles per gallon) contient des informations sur les performances de consommation de carburant de diverses voitures. Le jeu de données mpg de Seaborn comprend les données suivantes :

| | mpg | cylinders | displacement | horsepower | weight | acceleration | model_year | origin | name | |
|-----|------|-----------|--------------|------------|--------|--------------|------------|--------|---------------------------|--|
| 0 | 18.0 | 8 | 307.0 | 130.0 | 3504 | 12.0 | 70 | usa | chevrolet chevelle malibu | |
| 1 | 15.0 | 8 | 350.0 | 165.0 | 3693 | 11.5 | 70 | usa | buick skylark 320 | |
| 2 | 18.0 | 8 | 318.0 | 150.0 | 3436 | 11.0 | 70 | usa | plymouth satellite | |
| 3 | 16.0 | 8 | 304.0 | 150.0 | 3433 | 12.0 | 70 | usa | amc rebel sst | |
| 4 | 17.0 | 8 | 302.0 | 140.0 | 3449 | 10.5 | 70 | usa | ford torino | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 393 | 27.0 | 4 | 140.0 | 86.0 | 2790 | 15.6 | 82 | usa | ford mustang gl | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |



Création de graphiques avec Python : Box plot

Application sur CarsMPG data set : suit

Signification de chaque colonne :

- **Miles per gallon (mpg)** : Mesure de la consommation de carburant d'une voiture, indiquant combien de miles une voiture peut parcourir par gallon de carburant.
- **Cylinders** : Le nombre de cylindres dans le moteur de la voiture.
- **Displacement** : La cylindrée totale du moteur, mesurée en pouces cubes.
- **Horsepower** : La puissance du moteur, mesurée en chevaux-vapeur.
- **Weight** : Le poids de la voiture, mesuré en livres.
- **Acceleration** : Le temps nécessaire pour passer de 0 à 60 mph, mesuré en secondes.
- **Model year** : L'année modèle de la voiture.
- **Origin** : Le pays d'origine de la voiture (États-Unis, Europe, Japon).
- **Name** : Le nom du modèle de la voiture.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Box plot

Voici les étapes à suivre :

1. Importer les bibliothèques pour la visualisation et le chargement des données :

```
import seaborn as sns # Pour la visualisation des données
import matplotlib.pyplot as plt # Pour personnalisation et affichage des graphiques

# Charger le jeu de données mpg de Seaborn
cars = sns.load_dataset("mpg")
```



2. Initialiser la figure et Créer un box plot pour visualiser la consommation de carburant par nombre de cylindres :

```
plt.figure(figsize=(10, 6)) # Initialiser une figure avec une taille définie
# Créer un box plot
sns.boxplot(x='cylinders', y='mpg', data=cars, palette='Set2')
```

Nous observons la consommation de carburant mpg (Y) en fonction du nombre de cylindres (X).



Comme vous pouvez le constater, la création de ce graphique est très simple et similaire à celle d'autres types de graphiques. Il suffit d'initialiser les axes X et Y, de spécifier la source des données et de choisir une palette de couleurs.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Box plot

3. Ajouter les étiquettes, le titre et Afficher le graphique

```
# Ajouter les étiquettes et le titre
plt.xlabel('Nombre de Cylindres')
plt.ylabel('Consommation de Carburant (mpg)')
plt.title('Consommation de Carburant par Nombre de Cylindres')
# Afficher le graphique
plt.show()
```



Maintenant, l'output attendu de ce graphique sera la création d'un box plot pour chaque nombre de cylindres avec la variation de la consommation de carburant mpg (miles per gallon).

Avec :

1 mile (US) = 1.6093472187 km

1 gal (US) = 3.785411784 L

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Box plot

Interprétation des résultats :

3 Cylindres : La médiane est relativement basse, indiquant que les voitures avec 3 cylindres ne sont pas très efficaces en termes de consommation de carburant par rapport aux autres.

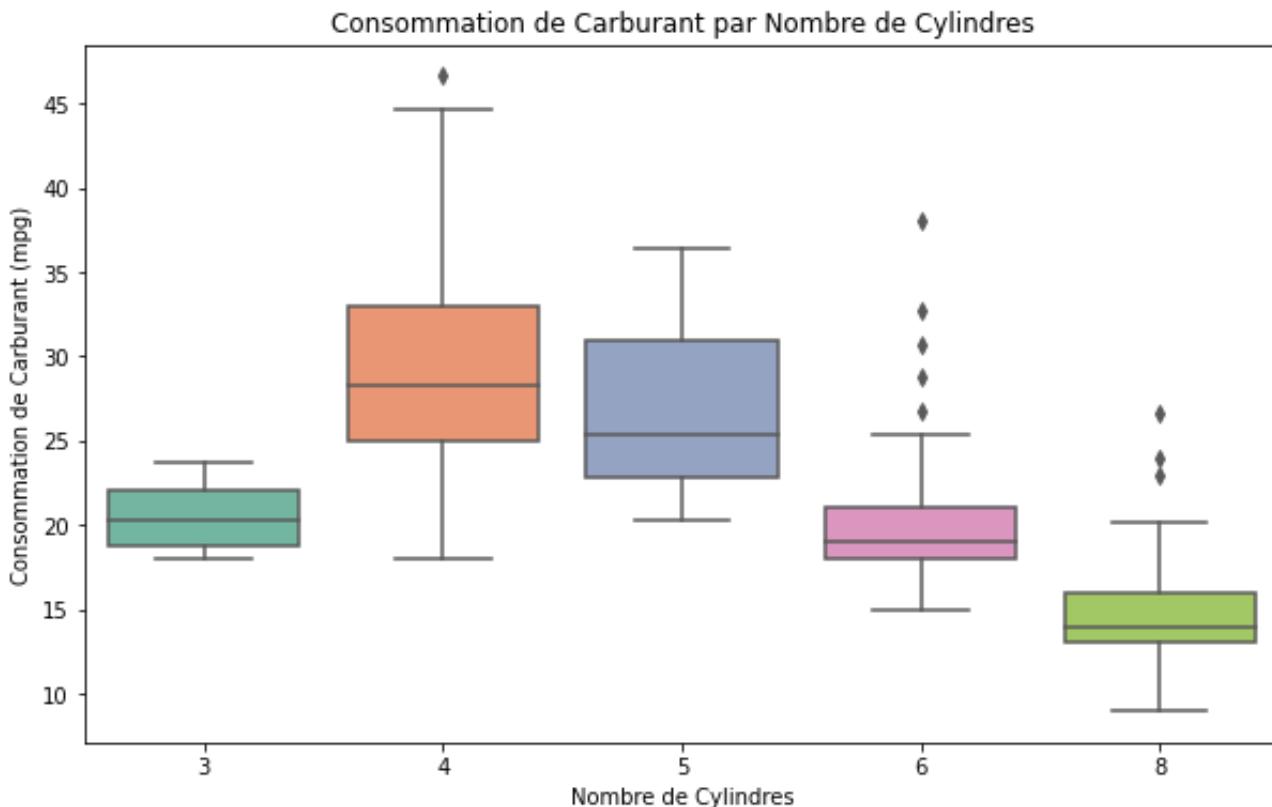
4 Cylindres : Les voitures avec 4 cylindres montrent la plus grande variabilité et la médiane la plus élevée (28 mpg), suggérant que ces voitures sont généralement plus efficaces et peuvent parcourir plus de miles par gallon de carburant.

5 Cylindres : La médiane est intermédiaire, Cela suggère une efficacité énergétique modérée.

6 Cylindres : La médiane est inférieure à celle des 4 cylindres, et il y a quelques valeurs aberrantes au-dessus de la boîte. indiquant une efficacité énergétique moindre.

8 Cylindres : Ces voitures ont l'efficacité de consommation la plus basse, avec une médiane basse (14 mpg) et plusieurs valeurs aberrantes. Cela signifie qu'elles sont les moins efficaces en termes de consommation de carburant.

Graphique produit :



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Box plot

Conclusion de l'analyse de l'application CarsMPG :

Le box plot montre que les voitures avec 4 cylindres sont les plus efficaces en termes de consommation de carburant, pouvant parcourir plus de miles par gallon. Les voitures avec 8 cylindres, en revanche, sont les moins efficaces, parcourant moins de miles par gallon de carburant.

Les valeurs plus élevées de miles per gallon (mpg) indiquent une meilleure performance énergétique, ce qui signifie que les voitures qui ont des valeurs élevées sur l'axe Y sont meilleures en termes de consommation de carburant.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

La Relation entre le GDP , l'espérance de vie et la population :

Pour analyser la relation entre le PIB par habitant, l'espérance de vie et la population, nous devons créer un graphique à bulles interactif.

Le graphique à bulles vous permet de visualiser facilement au moins trois dimensions essentielles :

- PIB par habitant (**X**) : Représentant la richesse économique par personne.
- Espérance de vie (**Y**) : Indiquant la santé et le développement des pays.
- Population (**taille de la bulle**) : Montrant la taille relative de la population de chaque pays.
- Note : **Les couleurs** des bulles peuvent être utilisées pour représenter les continents, facilitant ainsi la distinction géographique des données.

Voici les étapes et le script Python correspondant pour réaliser cette visualisation avec Plotly Express :

```
# Importer La bibliothèque Plotly Express :  
import plotly.express as px
```

```
# Charger Les données Gapminder  
df = px.data.gapminder()
```

Plotly Express est un package de visualisation de données qui vous permet de créer des plots interactifs avec moins de code.

Objectif : Utiliser les données Gapminder, qui contiennent des informations sur le développement des pays, pour notre analyse.

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

La Relation entre le GDP , l'espérance de vie et la population :

```
# Filtrer les données pour
# L'année 2007 et créer le
# graphique à bulles
fig = px.scatter(
    df.query("year==2007"),
    x="gdpPercap",
    y="lifeExp",
    size="pop",
    color="continent",
    hover_name="country",
    log_x=True,
    size_max=60
)

# Afficher le graphique
fig.show()
```

Paramètres du graphique à bulles :

- `df.query("year==2007")` : Sélectionner uniquement les données de l'année 2007 pour une analyse précise.
- `x="gdpPercap"` : L'axe des X représente le PIB par habitant.
- `y="lifeExp"` : L'axe des Y représente l'espérance de vie.
- `size="pop"` : La taille des bulles représente la population des pays.
- `color="continent"` : Les bulles sont colorées en fonction des continents pour une distinction géographique.
- `hover_name="country"` : Le nom du pays s'affiche lors du survol de la bulle pour des informations supplémentaires.
- `log_x=True` : L'échelle logarithmique de l'axe des X permet de mieux visualiser les différences de PIB par habitant.
- `size_max=60` : Limite la taille maximale des bulles pour une meilleure lisibilité.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

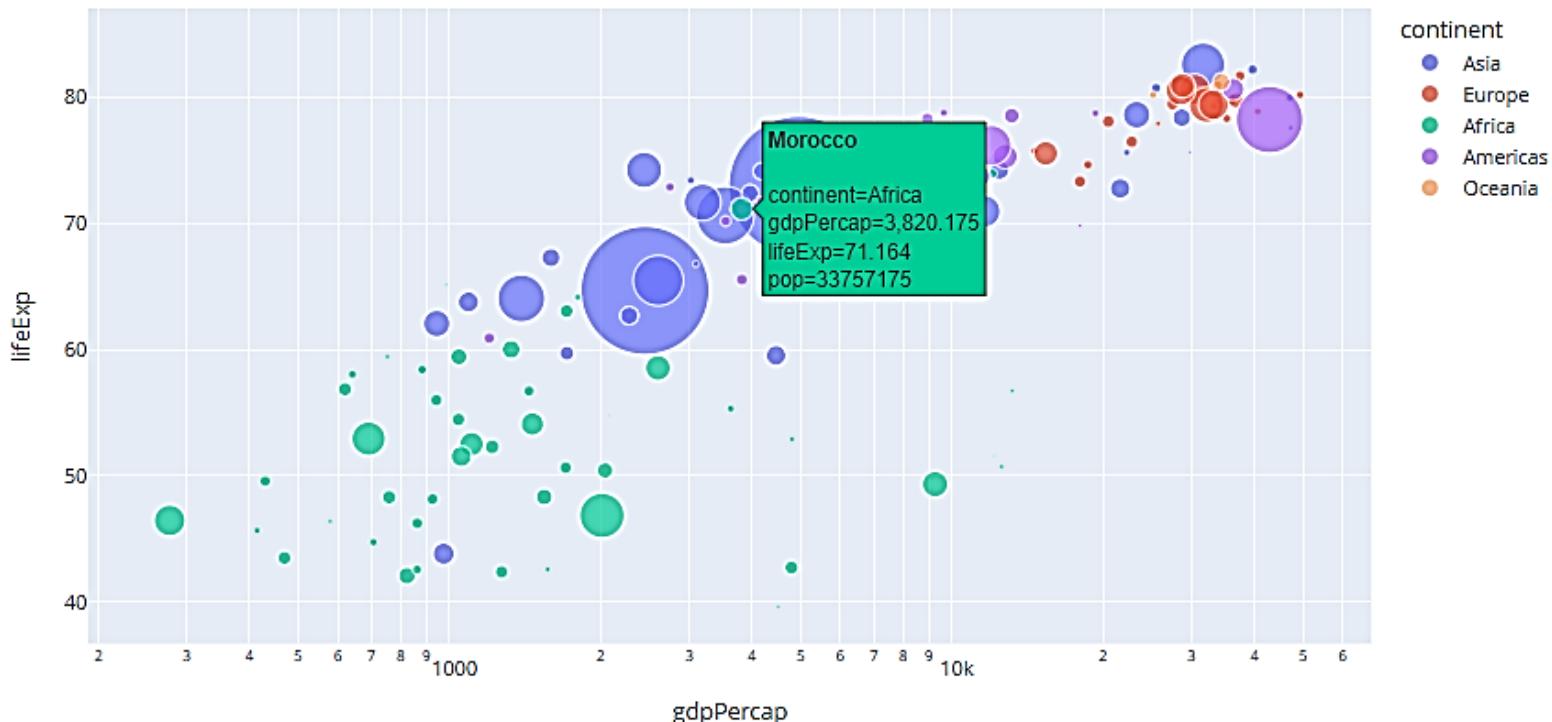
Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

La Relation entre le GDP , l'espérance de vie et la population :

Pour tracer le PIB par habitant (X), l'espérance de vie (Y) et la population (taille de la bulle) avec les couleurs représentant les continents, utilisez des données de Kaggle ou une bibliothèque. Pour visualiser les données récentes, téléchargez « World Banc Data » depuis Kaggle.

Lors du survol «hovering» de chaque point, vous pouvez maintenant voir le nom du pays, ce qui représente la cinquième dimension. Cela permet d'identifier facilement les pays en plus des quatre autres dimensions (PIB par habitant, espérance de vie, population, et continent). Cette fonctionnalité améliore la compréhension et l'interactivité du graphique.



Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

Recommandation : Résolution des Erreurs d'importation (Troubleshooting Import Errors)

Dans le cadre de l'utilisation de la bibliothèque Plotly pour des visualisations interactives en Python, il est possible de rencontrer des erreurs d'importation. La section actuelle décrit le problème rencontré, les tentatives de résolution, et les mesures correctives prises pour garantir le bon fonctionnement de la bibliothèque.

Problème : Lors de l'importation de Plotly, l'erreur suivante a été rencontrée :

```
In [5]: import plotly  
plotly.__version__  
  
-----  
ModuleNotFoundError Traceback (most recent call last)  
~\AppData\Local\Temp\ipykernel_9000\632329486.py in <module>  
----> 1 import plotly  
      2 plotly.__version__  
  
ModuleNotFoundError: No module named 'plotly'
```



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

Recommandation : Résolution des Erreurs d'importation (Troubleshooting Import Errors)

Tentatives de résolution :

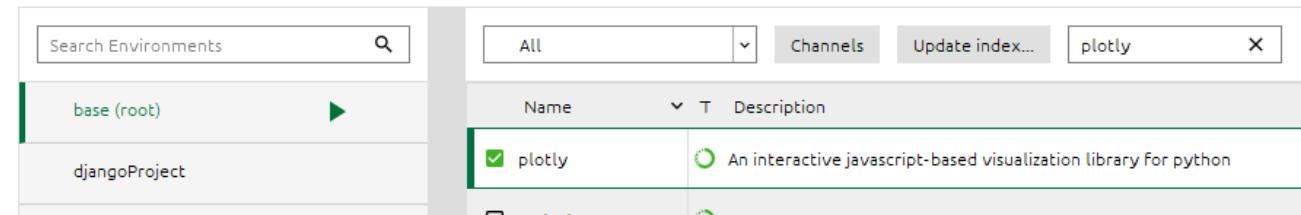
1. Installation via Conda :

Une tentative d'installation de Plotly en utilisant Conda a été effectuée :

```
In [3]: ! conda install plotly
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... done
# All requested packages already installed.
```



==> WARNING: A newer version of conda exists. <==



The screenshot shows the Anaconda Navigator interface. In the top search bar, 'plotly' is typed. Below the search bar, the 'base (root)' environment is selected. In the main pane, a table lists packages. The 'plotly' row is highlighted with a green border and has a checkmark in the 'Selected' column. The description for 'plotly' is: 'An interactive javascript-based visualization library for python'.

Par l'utilisation de la commande « conda install »
Ou par l'interface « Anaconda Navigator / environnement »

Si le package ne fonctionne toujours pas, nous pouvons mettre à jour l'ensemble de l'environnement conda, ou simplement utiliser la solution suivante.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

Recommandation : Résolution des Erreurs d'importation (Troubleshooting Import Errors)

2. Installation via Pip :

Une seconde tentative d'installation de Plotly a été effectuée en utilisant Pip :

```
In [2]: pip install plotly
Collecting plotly
  Note: you may need to restart the kernel to use updated packages.

    Downloading plotly-5.22.0-py3-none-any.whl (16.4 MB)
Collecting tenacity>=6.2.0
  Downloading tenacity-8.3.0-py3-none-any.whl (25 kB)
Requirement already satisfied: packaging in c:\programdata\anaconda3\envs\myenv\lib\
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\programdata\anaconda3\
g->plotly) (3.0.4)
Installing collected packages: tenacity, plotly
Successfully installed plotly-5.22.0 tenacity-8.3.0
```



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Python : Bubble Chart (Graphique à bulles)

Recommandation : Résolution des Erreurs d'importation (Troubleshooting Import Errors)

Vérification :

Après l'installation via Pip, il est recommandé de redémarrer le noyau (kernel) de l'environnement Jupyter pour appliquer les modifications.

Voici les étapes à suivre :

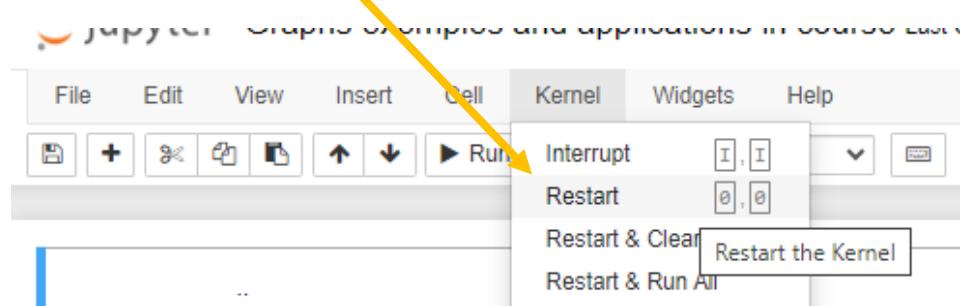
1. Redémarrage du kernel :

Après l'installation de Plotly, redémarrez le kernel en allant dans le menu Kernel et en sélectionnant Restart Kernel.

2. Vérification de l'importation :

Après le redémarrage du kernel, réessayez d'importer Plotly :

```
import plotly  
print(plotly.__version__)
```



Si l'installation et le redémarrage ont été effectués correctement, l'importation de Plotly devrait fonctionner sans erreur et la version de Plotly devrait s'afficher.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Bar Cahrt

1. Pour créer un graphique à barres simple :

```
# valeurs de l'axe x
x <- c("A", "B", "C", "D")
# valeurs de l'axe y
y <- c(2, 4, 6, 8)
# création du diagramme en barres
barplot(y, names.arg = x, xlab = "Catégories", ylab = "Valeurs", main = "Diagramme en Barres")
```

xlab : définit l'étiquette pour l'axe des x ("Catégories").
ylab : définit l'étiquette pour l'axe des y ("Valeurs").
main : définit le titre principal du graphique ("Diagramme en Barres")

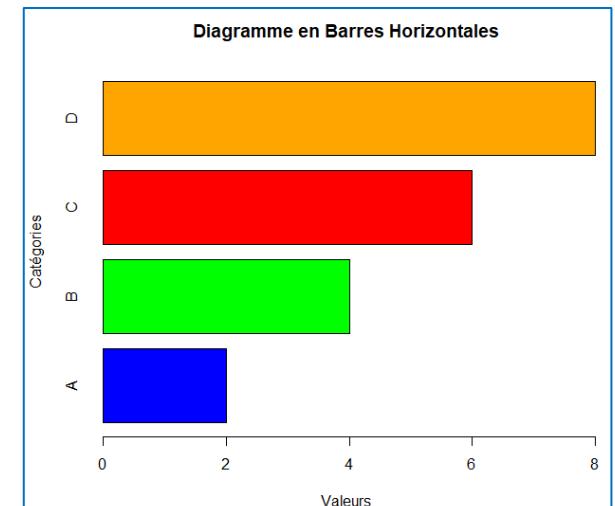
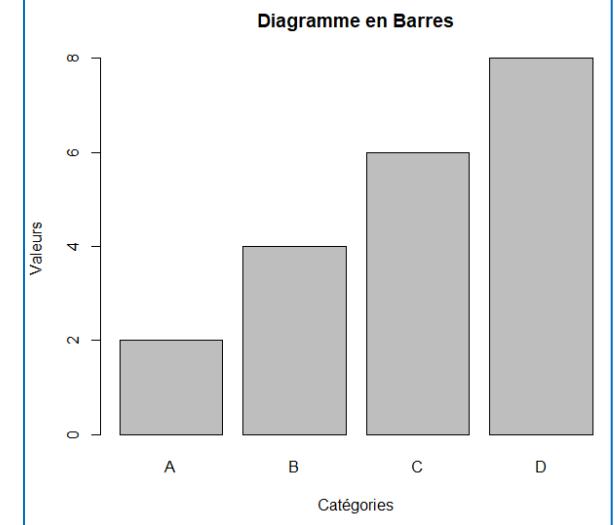
2. Pour personnaliser votre graphique à barres :

Sur RStudio :

```
# couleurs des barres
couleurs <- c("blue", "green", "red", "orange")
|
# création du diagramme en barres horizontales avec couleurs personnalisées
barplot(y, names.arg = x, horiz = TRUE, col = couleurs, xlab = "Valeurs", ylab = "Catégories", main = "Diagramme en Barres Horizontales")
```

horiz = TRUE : spécifie que le diagramme en barres sera horizontal.

col : utilise le vecteur couleurs pour définir les couleurs des barres.



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Scatter Plot

1. Pour créer un Nuage de points :

Utilisons jeu de données intégré à R pour créer un graphique de dispersion. Nous allons utiliser l'ensemble de données, **mtcars** qui contient des informations sur différentes voitures. Voici comment procéder :

```
# Chargement de l'ensemble de données mtcars
```

```
data(mtcars)
```

```
# Affichage des premières lignes de l'ensemble de données mtcars pour comprendre sa structure
```

```
head(mtcars)
```

```
> head(mtcars)
   mpg cyl disp hp drat wt qsec vs am gear carb
Mazda RX4     21.0   6 160 110 3.90 2.620 16.46 0  1    4    4
Mazda RX4 Wag 21.0   6 160 110 3.90 2.875 17.02 0  1    4    4
Datsun 710    22.8   4 108  93 3.85 2.320 18.61 1  1    4    1
Hornet 4 Drive 21.4   6 258 110 3.08 3.215 19.44 1  0    3    1
Hornet Sportabout 18.7   8 360 175 3.15 3.440 17.02 0  0    3    2
Valiant      18.1   6 225 105 2.76 3.460 20.22 1  0    3    1
```

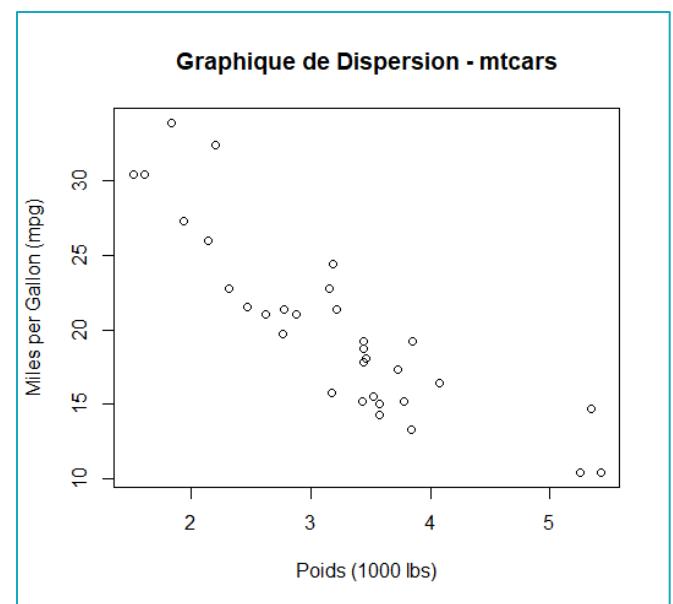
```
# Création du graphique de dispersion
```

```
plot(mtcars$wt, mtcars$mpg,
```

```
  xlab = "Poids (1000 lbs)", ylab = "Miles per Gallon (mpg)", main = "Graphique de Dispersion - mtcars")
```

Étiquettes les axes

Titre du graphique



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R



Création de graphiques avec R : Scatter Plot

2. Explication des étapes suivies dans l'application précédente mtcars :

- **Chargement des données** : Nous avons chargé l'ensemble de données mtcars en utilisant data(mtcars).
- **Exploration initiale** : Pour mieux comprendre la structure de l'ensemble de données et voir ses premières observations, nous avons utilisé head(mtcars).
- **Création du graphique de dispersion** : Nous avons créé un graphique de dispersion avec plot(mtcars\$wt, mtcars\$mpg, ...), où **wt** représente le poids des voitures en (thousands of pounds) et **mpg** indique le nombre de miles par gallon.
- **Personnalisation du graphique** : Les options **xlab**, **ylab** et **main** ont été utilisées pour définir respectivement les étiquettes des axes x et y, ainsi que le titre du graphique, tous en français.

Vous pouvez explorer d'autres variables de l'ensemble de données mtcars en remplaçant wt et mpg par d'autres colonnes telles que hp (puissance), disp (déplacement du moteur), etc., selon ce qui vous intéresse.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Histogram

Création d'un histogramme :

Voici comment créer un histogramme avec l'ensemble de données **mtcars** en R :

```
# Histogramme de la variable mpg (miles per gallon)
```

```
hist(mtcars$mpg,  
      xlab = "Miles per Gallon (mpg)", ylab = "Fréquence",  
      main = "Histogramme de la Consommation de Carburant")
```

Crée un histogramme des valeurs de la variable mpg.

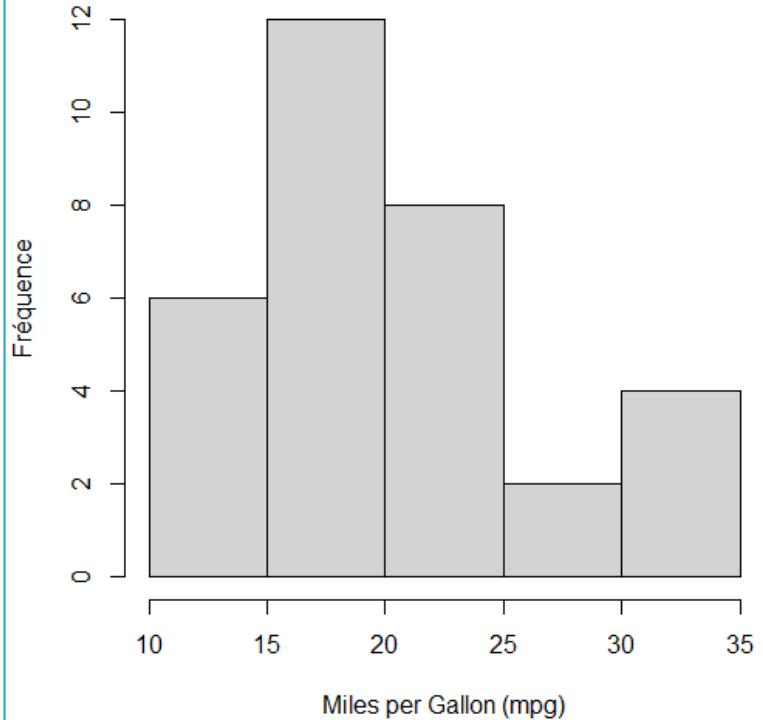
Étiquettes respectivement les étiquettes des axes x et y

Définit le titre de l'histogramme



L'histogramme montre clairement que la barre représentant l'intervalle de 15 à 20 mpg atteint une fréquence de 12. Alors on peut dire que 12 voitures ont une consommation comprise entre 15 et 20 miles par gallon.

Histogramme de la Consommation de Carburant



```
> dim(mtcars)  
[1] 32 11
```

L'histogramme nous permet de visualiser la répartition des consommations en carburant (mpg) des voitures dans l'ensemble de données mtcars.

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

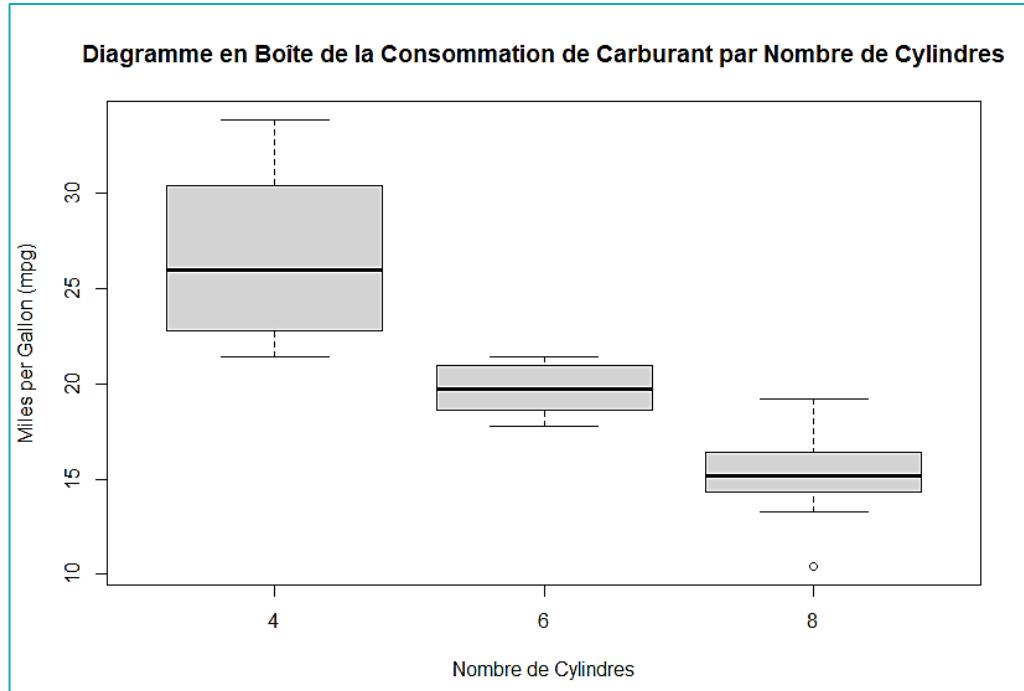
Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Box Plot

Le diagramme en boîte (box plot) est utilisé pour représenter la distribution des données et pour identifier les valeurs aberrantes (outliers). Nous allons créer un diagramme en boîte pour visualiser la consommation de carburant (mpg) par nombre de cylindres (cyl) des voitures.

Création d'un Box Plot :

```
# Diagramme en boîte de la variable mpg par nombre de cylindres  
  
boxplot(mpg ~ cyl, data = mtcars,  
  
xlab = "Nombre de Cylindres",  
  
ylab = "Miles per Gallon (mpg)", # Étiquettes des axes  
  
main = "Diagramme en Boîte de la Consommation de Carburant par Nombre de  
Cylindres") # Définit le titre de l'histogramme
```



01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Box Plot

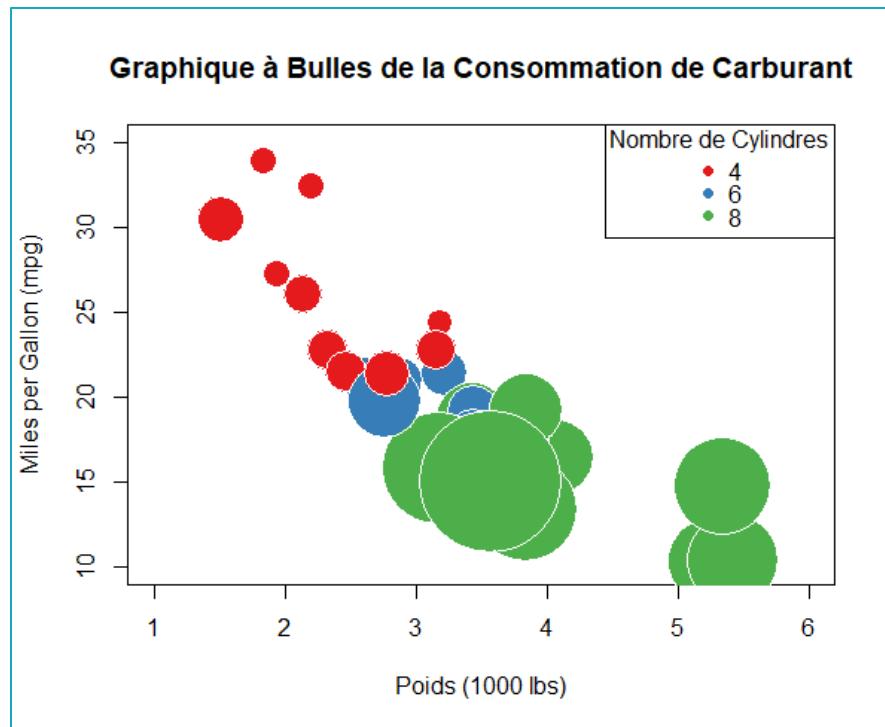
Pour créer un graphique à bulles, nous pouvons utiliser la fonction symbols pour représenter les bulles. Voici comment procéder avec l'ensemble de données mtcars :

```
# Chargement du package nécessaire pour des couleurs
library(RColorBrewer)

# Définition des couleurs pour les différentes catégories de cylindres
colors <- brewer.pal(3, "Set1")

# Création du graphique à bulles
symbols(mtcars$wt, mtcars$mpg, circles = mtcars$hp, inches = 0.5, fg = "white",
        bg = colors[as.factor(mtcars$cyl)], xlab = "Poids (1000 lbs)",
        ylab = "Miles per Gallon (mpg)", main = "Graphique à Bulles de la Consommation de Carburant",
        xlim = c(1, 6), ylim = c(10, 35))

# Ajout d'une légende
legend("topright", legend = levels(as.factor(mtcars$cyl)),
       col = colors, pch = 16, title = "Nombre de Cylindres")
```



wt: Poids en milliers de livres
mpg: Miles per gallon (consommation de carburant)
hp: Puissance en chevaux-vapeur (horsepower)
cyl: Nombre de cylindres

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

Création de graphiques avec Python/Excel/R

Création de graphiques avec Excel :

Remarque :



Le prochain chapitre de la formation "CONFIGURER DES GRAPHIQUES POUR UNE ANALYSE APPROFONDIE" abordera la création de graphiques dans Excel. Dans la section "Choisir les graphiques en fonction de l'objectif et des données", nous ne nous concentrerons pas uniquement sur la création des graphiques. De plus nous verrons pourquoi un type de graphique est préférable à un autre, en fonction de nos objectifs et des données disponibles. Ainsi, vous apprendrez à sélectionner le graphique le plus approprié pour représenter vos données de manière claire et efficace, afin de répondre aux besoins spécifiques de votre projet d'analyse.

Titre

- Many dimensions by spider plots it is used to measure similar columns such as evaluation sport capacity such as speed (like players one at fifa)

Lesson Video: Box and Whisker Plots

<https://www.nagwa.com/en/videos/575198521364/>

01 – CRÉER DIVERS TYPES DE GRAPHIQUES

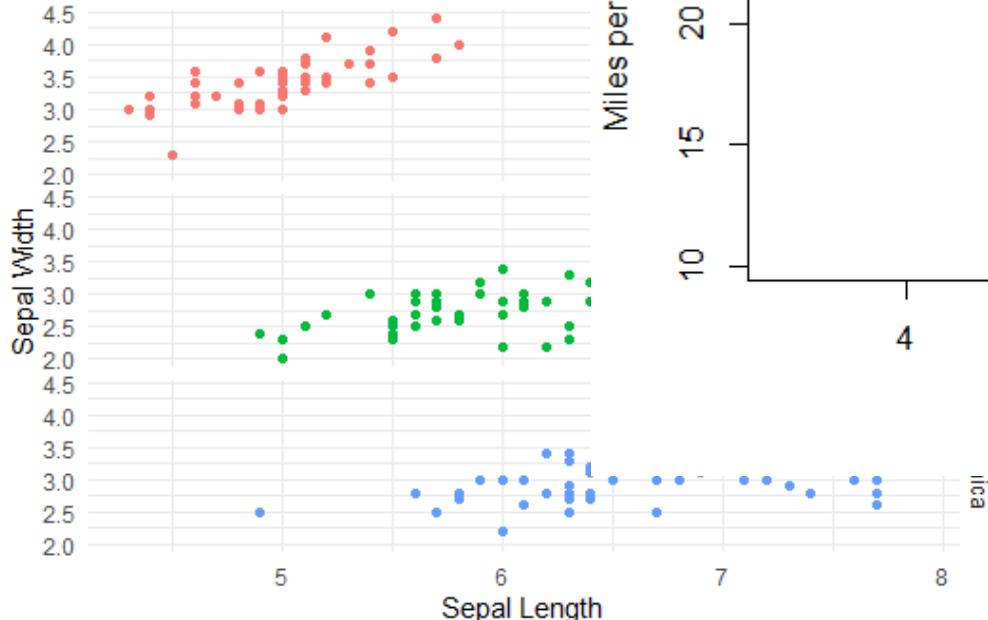
Création de graphiques avec Python/Excel/R

Création de graphiques avec R : Exemples

Les graphiques statiques constituent une forme de communication de qualité professionnelle incluant des symboles et des textes.

Exemples de graphiques : application

Pair Plot pour Iris Dataset



Boîte en Boîte de la Consommation de Carburant par Nombre de Cylindres

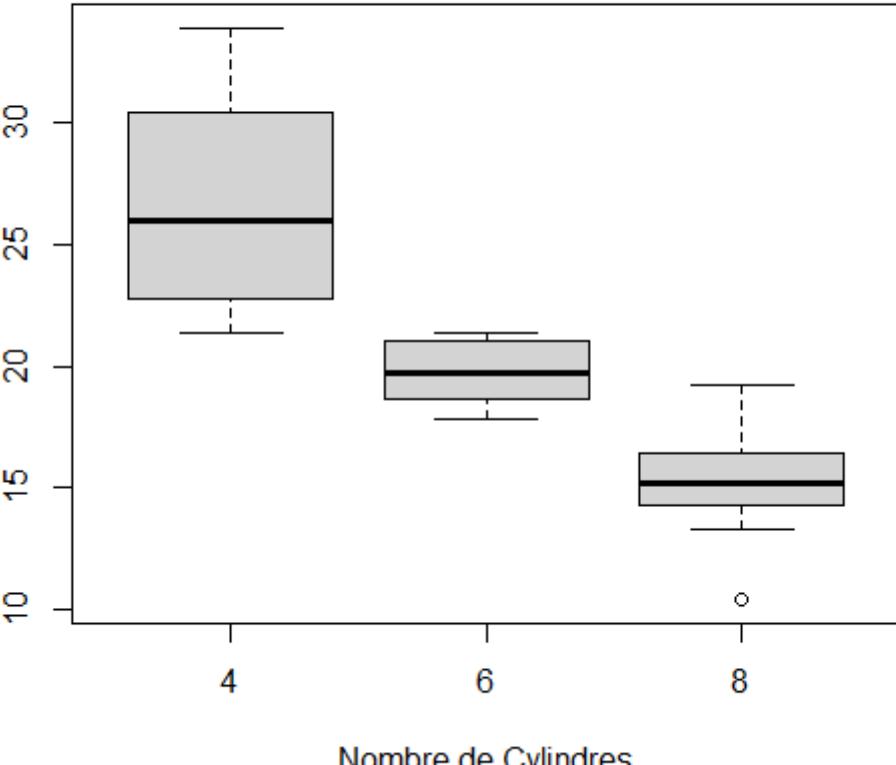
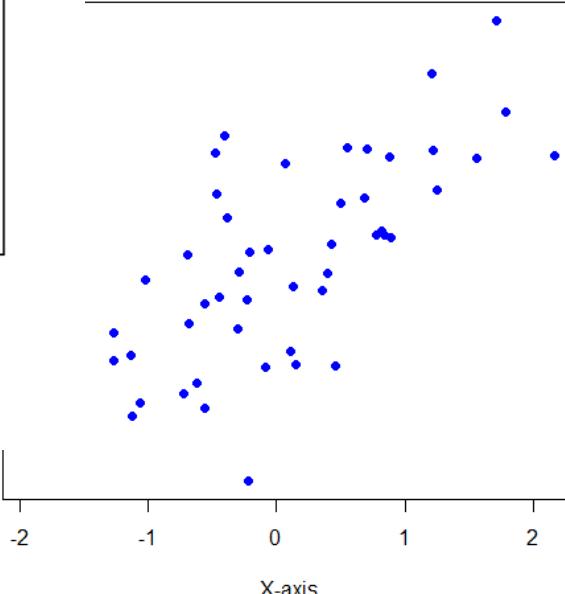


Diagramme de dispersion / Scatter Plot





CHAPITRE 2

CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Ce que vous allez apprendre dans ce chapitre :

- Choisir les graphiques en fonction d'objectif
- Choisir les graphiques en fonction des données
- Explorer la visualisation multivariée
- Utiliser stratégiquement les couleurs et les annotations
- Sélectionner les graphiques avancés avec Python/Excel

 30 heures



CHAPITRE 2

CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

1. Choix des graphiques en fonction d'objectif
2. Choix des graphiques en fonction des données
3. Exploration de la visualisation multivariée
4. Utilisation stratégique des couleurs et des annotations
5. Sélection des graphiques avancés avec Python/R

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Introduction :

- Dans l'analyse de données (**Data Analysis**), les besoins métier se présentent souvent sous forme de questions. En tant qu'analyste de données, il est essentiel d'extraire l'**objectif** sous-jacent de ces questions et, en s'appuyant sur votre connaissance des **types de données**, de choisir le graphique le plus approprié pour une visualisation efficace.

Les objectifs (Goals) d'analyse : « What are your business needs ?? »

Pour l'instant, concentrons-nous sur la partie "objectif" du cadre en ce qui concerne l'affichage des données. La plupart des visualisations remplissent l'un des quatre objectifs principaux :

- Montrer comment les valeurs se comparent entre elles.  **Comparaison**
- Montrer comment les données sont distribuées.  **Distribution**
- Montrer comment les données sont composées.  **Composition**
- Montrer comment les valeurs sont liées entre elles.  **Relations**



On commence par préciser **l'objectif**, puis on se concentre sur les **types de données**.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif



Exemples : De la question à l'objectif du graphique

Dans les scénarios à venir, nous extrairons l'objectif de cas réguliers. Nous verrons également comment cette approche contribue à la création de spécifications graphiques. Nous avons également pris en compte les types de données requis pour ces graphiques.

Analyse du 1er Problème :

Question : Comment les chiffres de vente se comparent-ils entre les différentes régions ?

1 Objectif : Montrer comment les valeurs se comparent entre chaque catégorie (régions).

2 Données :

| Régions | Ventes en DH |
|-------------|--------------|
| Casa-settat | 45626 |
| Rabat-Sale | 21449 |

Graphique Proposé ?



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

Solution du 1er problème :

3

Graphique Proposé : Utilisez un diagramme à barres (Bar Chart).

Chaque région aura une barre représentant ses ventes.

Guide de création :

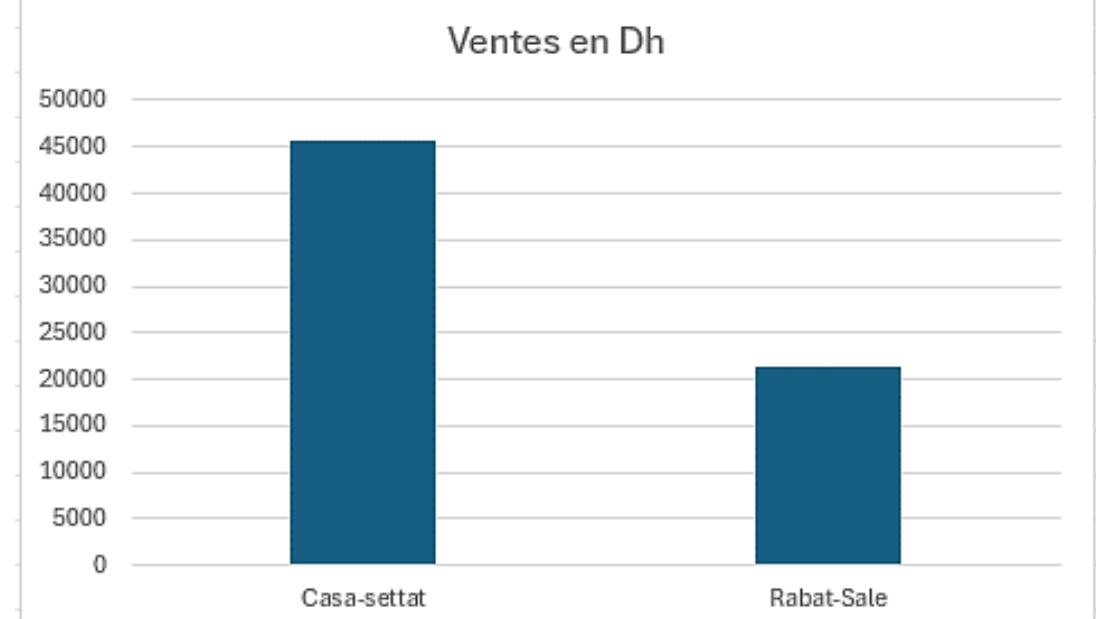


Sur Excel, vous pouvez coller le tableau, sélectionner les deux colonnes et insérer un graphique de type Clustered Column.

| Regions | Ventes Dh |
|-------------|-----------|
| Casa-settat | 45626 |
| Rabat-Sale | 21449 |

4

Résultat du graphique :



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

Analyse du 2ème problème :

Question : Quelle est la distribution d'âge de notre base de clients ?

1 Objectif : Montrer comment les données sont distribuées en termes d'âge.

2

Données :

| Age Clients | nembre of individuals |
|-------------|-----------------------|
| 18 | 1 |
| 19 | 2 |
| 20 | 3 |
| 21 | 4 |
| 22 | 5 |
| 23 | 6 |
| 24 | 10 |
| 25 | 12 |
| 26 | 14 |
| 27 | 15 |
| 28 | 15 |
| 29 | 14 |
| 30 | 13 |
| 31 | 10 |
| 32 | 2 |
| 33 | 1 |

Graphique Proposé ?



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

Solution du 2ème problème :

3 Graphique Proposé : Utilisez un histogramme (Histogram).

Les groupes d'âge seront sur l'axe horizontal, et la fréquence (nombre de clients) sur l'axe vertical.

Guide de création :

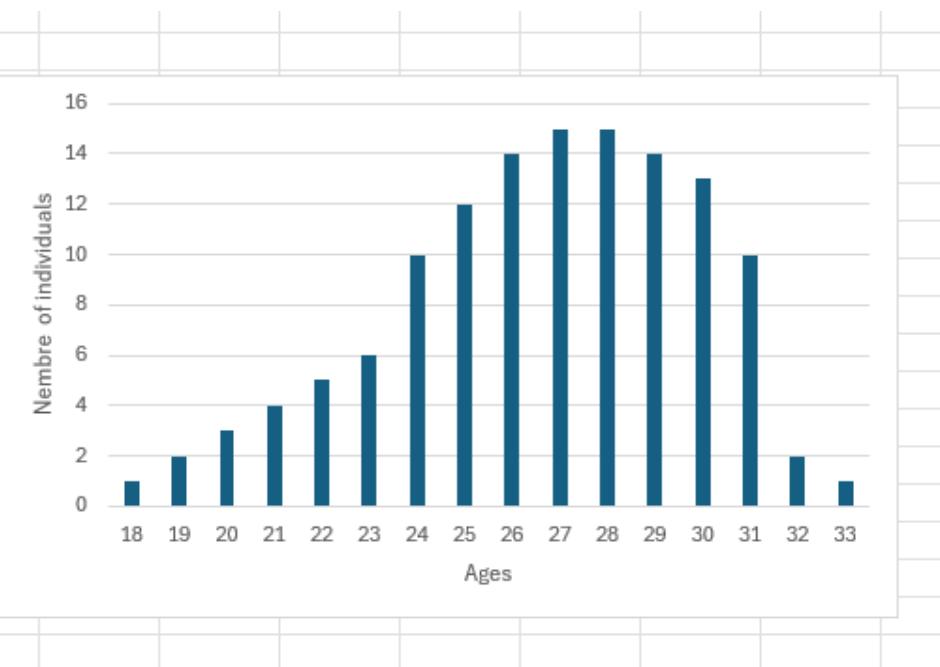


Sur Excel, sélectionnez la colonne à visualiser, puis allez à insérer un graphique et insérez un graphique à Colonnes groupées.

| Age Clients | nembre of individuals |
|-------------|-----------------------|
| 18 | 1 |
| 19 | 2 |
| 20 | 3 |
| 21 | 4 |
| 22 | 5 |
| 23 | 6 |
| 24 | 10 |
| 25 | 12 |
| 26 | 14 |
| 27 | 15 |
| 28 | 15 |
| 29 | 14 |
| 30 | 13 |
| 31 | 10 |
| 32 | 2 |
| 33 | 1 |

4

Résultat du graphique :



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

Analyse du 3ème problème :

Question : Y a-t-il une corrélation entre les dépenses de marketing et les revenus ?

1 **Objectif :** Montrer comment les valeurs sont liées entre elles (relation entre dépenses de marketing et revenus).

2 **Données :**

| Period | Marketing Expenses (USD) | Revenue (USD) |
|---------|--------------------------|---------------|
| Q1 2023 | 10 | 50 |
| Q2 2023 | 15 | 60 |
| Q3 2023 | 20 | 70 |
| Q4 2023 | 25 | 80 |
| Q1 2024 | 30 | 90 |

Graphique Proposé ?



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

Résolution du 3ème problème :

3

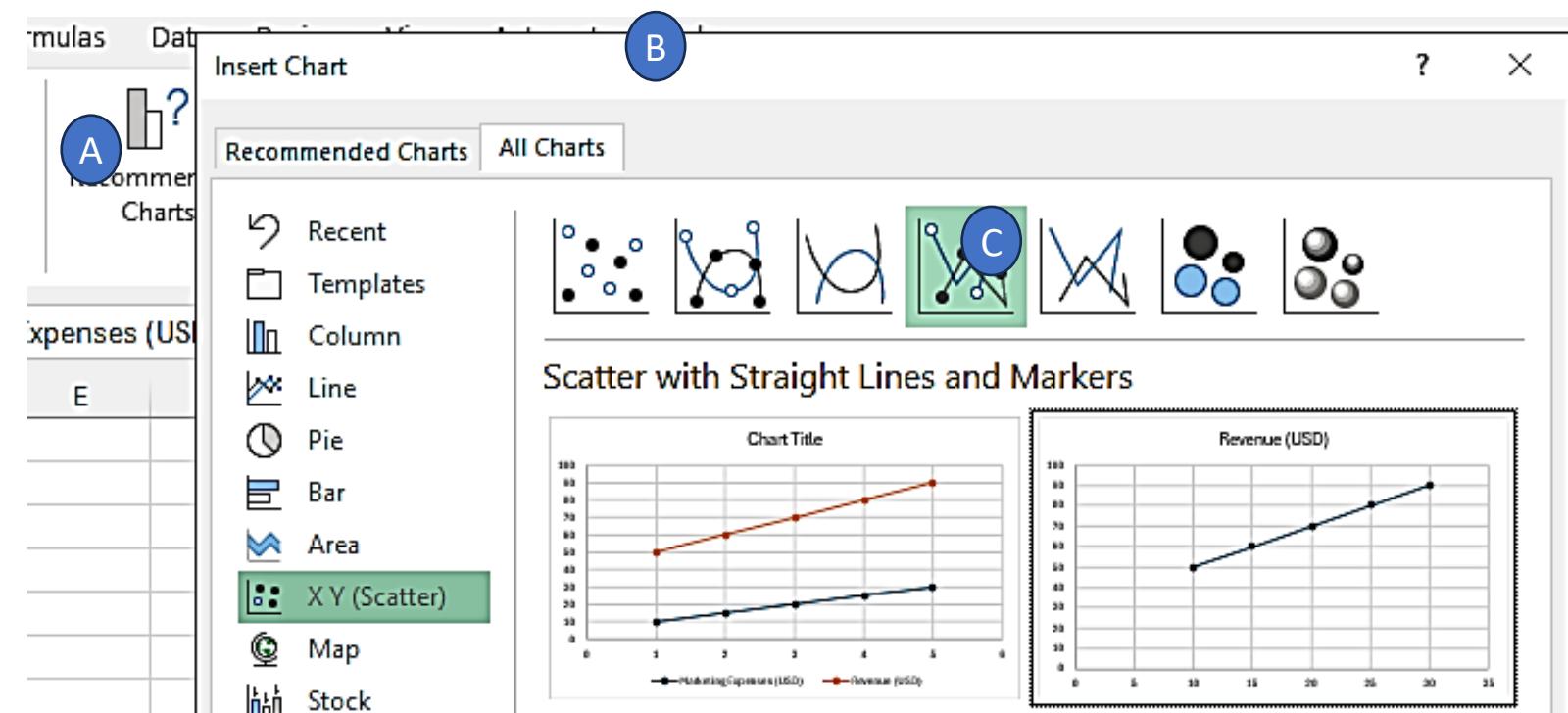
Graphique Proposé : Utilisez un nuage de points (Scatter plots).

Placez les dépenses de marketing sur l'axe horizontal et les revenus sur l'axe vertical.

Guide de création :



Sur Excel, sélectionnez les deux colonnes, puis allez à insérer un graphique et insérez un graphique à nuages de points X Y.



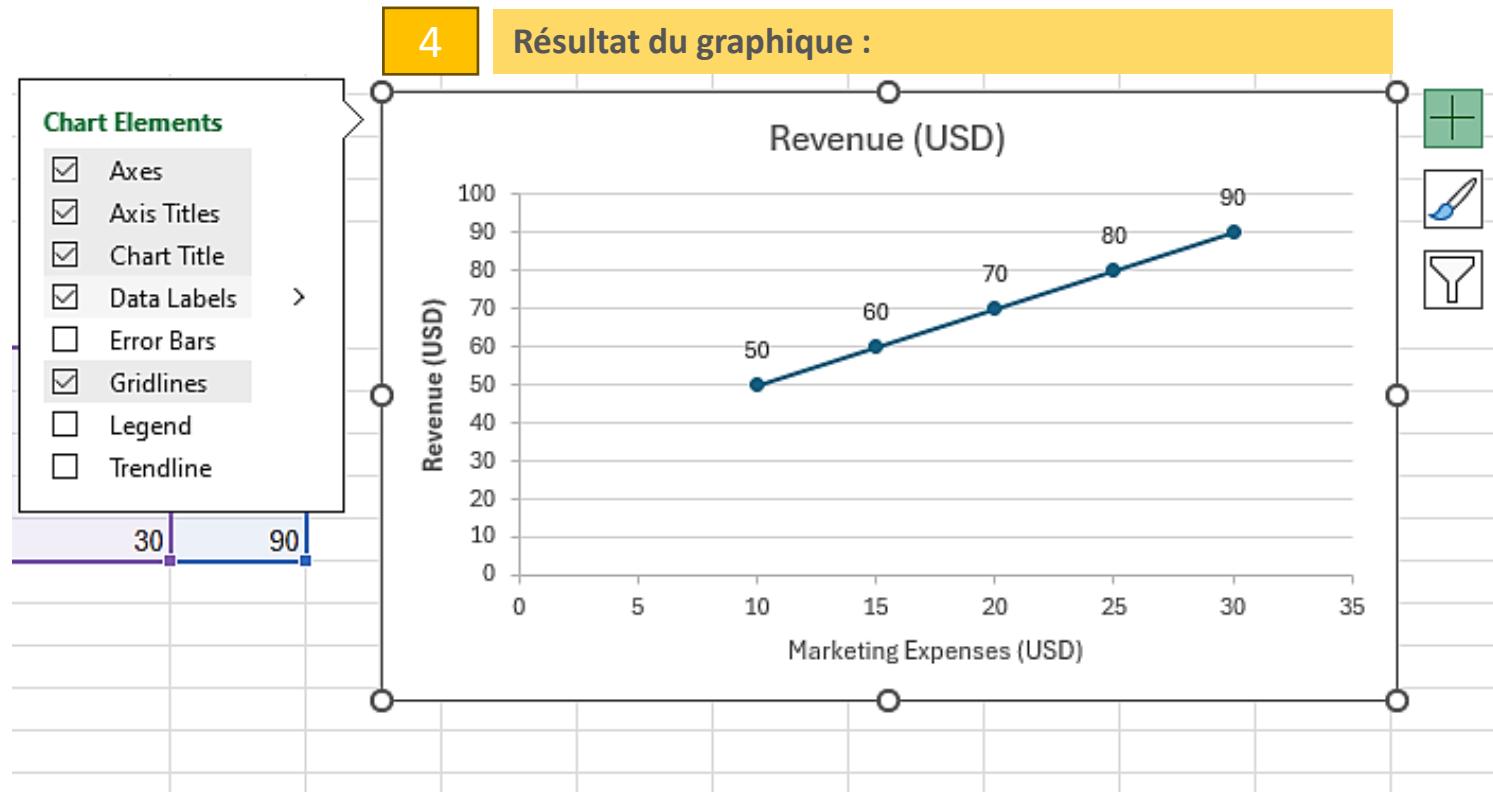
The screenshot shows the 'Insert Chart' dialog box in Excel. The 'Recommended Charts' tab is selected. On the left, under 'Recent', there are icons for a scatter plot, a line chart, and a pie chart. Below these are icons for 'Templates', 'Column', 'Line', 'Pie', 'Bar', 'Area', 'Map', and 'Stock'. The 'XY (Scatter)' icon is highlighted with a green box and a blue circle labeled 'A'. At the top right of the dialog box is a blue circle labeled 'B' pointing to the 'Insert' button in the ribbon. To the right of the dialog box are two preview charts. The first preview, labeled 'Scatter with Straight Lines and Markers', shows a scatter plot with a red line and a blue line. The second preview, labeled 'Revenue (USD)', shows a scatter plot with a single black line. Both charts have axes ranging from 0 to 100.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction d'objectif

Exemples : De la question à l'objectif du graphique

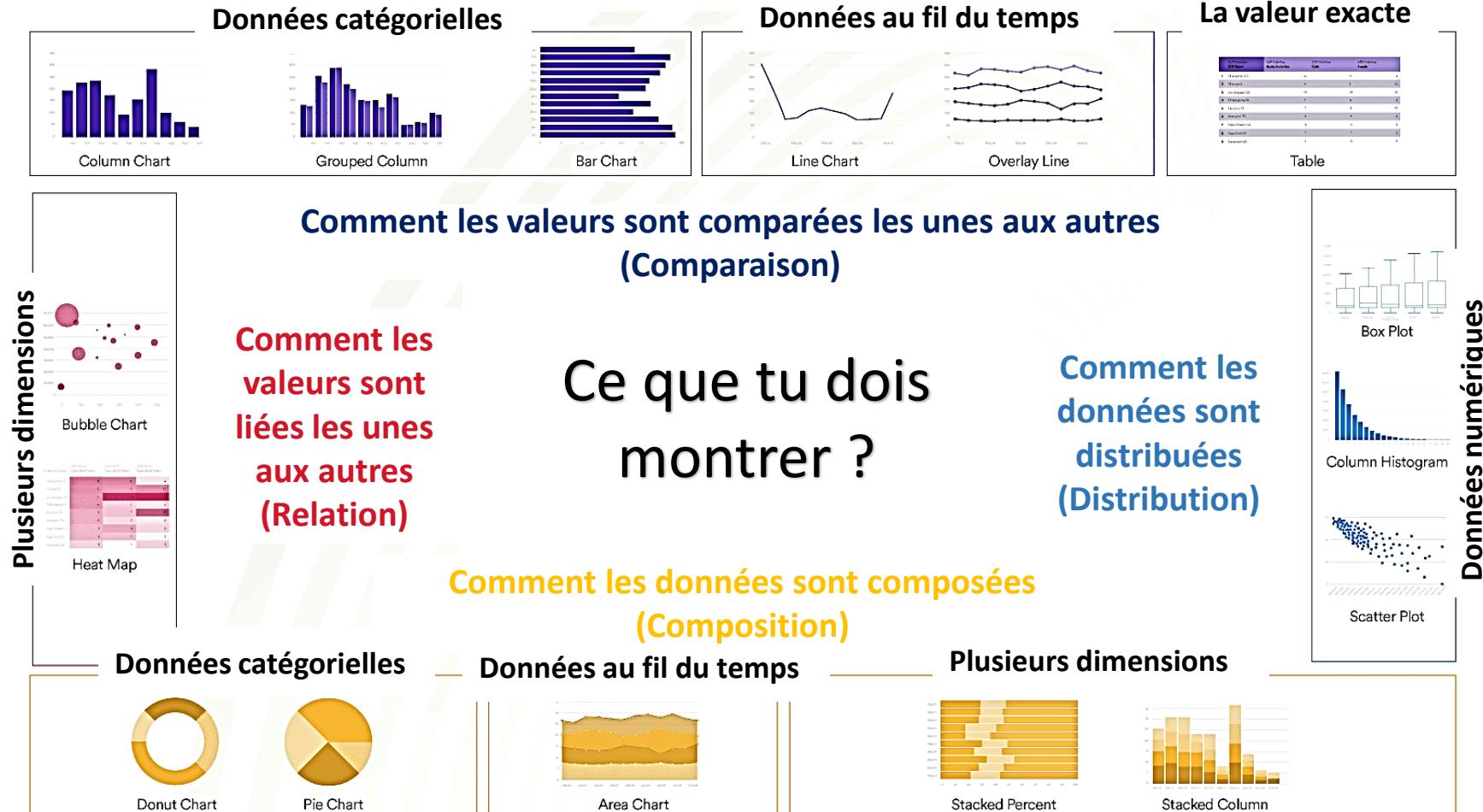
Optimisation et paramètres de graphiques : Après la sélection du graphique, nous pouvons ajouter des étiquettes d'axes (Axis Titles) sur les axes de graphique et des étiquettes de données (Data Labels) sur chaque point.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Choix des graphiques en fonction des données

Mind map du choix d'un graphique : Synthèse





CHAPITRE 2

CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

1. Choix des graphiques en fonction d'objectif
2. Choix des graphiques en fonction des données
- 3. Exploration de la visualisation multivariée**
4. Utilisation stratégique des couleurs et des annotations
5. Sélection des graphiques avancés avec Python/R

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Exploration de la visualisation multivariée

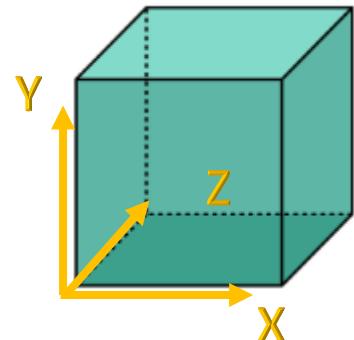
Problématique : Curse of dimensionality

- La visualisation multivariée se réfère à la représentation de jeux de données contenant plus de trois variables.
- Le "mal de la dimensionnalité" complique cette tâche, car les graphiques courants supportent généralement jusqu'à trois dimensions.
- L'efficacité des éléments visuels (**couleur, forme, taille**) diminue avec l'augmentation du nombre de variables.

Le fléau de la dimension, ou mal de la dimension « **Curse of dimensionality** », est un terme inventé par Richard Bellman en 1961. Il décrit divers phénomènes qui surviennent lorsqu'on tente d'analyser ou d'organiser des données dans des espaces de grande dimension, alors que ces phénomènes n'apparaissent pas dans des espaces de dimension inférieure.



Les techniques de visualisation sont essentielles pour résoudre le problème du "mal de la dimensionnalité". La réduction de dimension, qui sera étudiée dans d'autres cours de cette formation, est l'une des solutions proposées. Pour l'instant, nous nous concentrerons sur les techniques de visualisation, une compétence essentielle que tout analyste de données doit maîtriser.



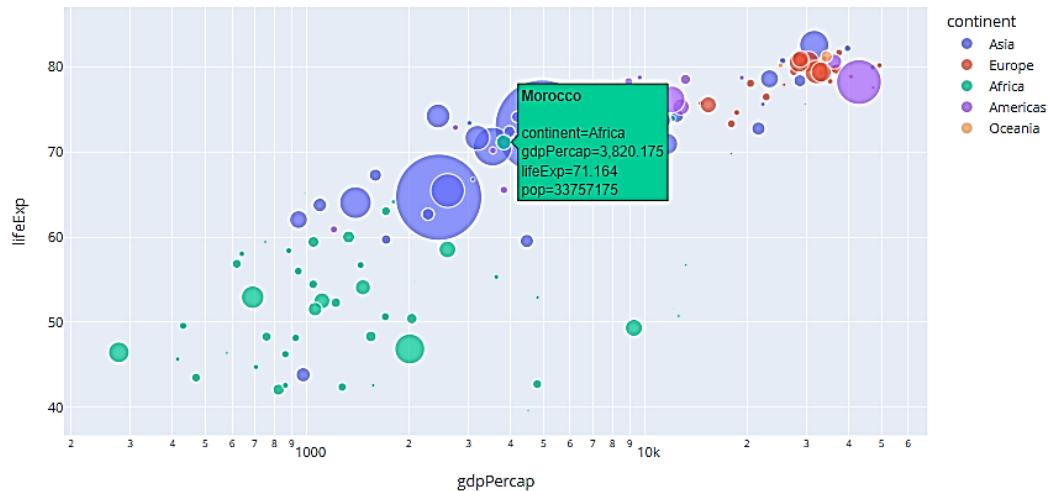
02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Exploration de la visualisation multivariée

Conseils

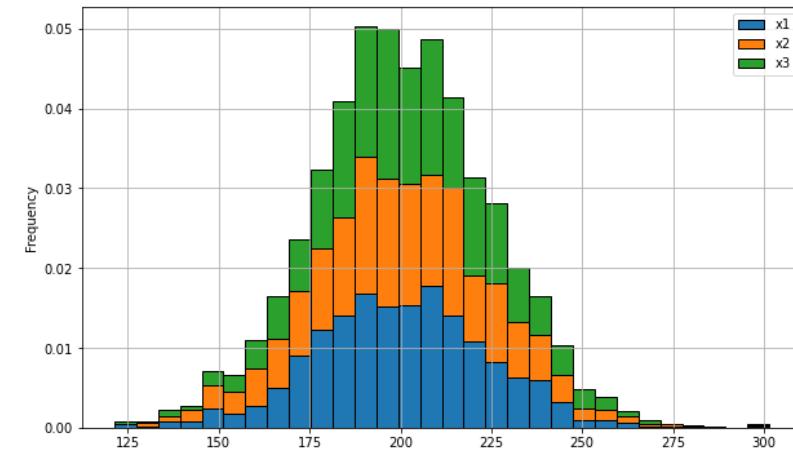
Tout d'abord, il est important de penser aux solutions courantes pour la visualisation multivariée, telles que les graphiques à bulles (Bubble Chart) ou colonne empilée (**stacked column**), etc. Non seulement pour la simplicité de leur création ou la disponibilité de ces graphiques, mais surtout pour votre audience. Dans un contexte professionnel, il est essentiel de toujours livrer les analyses de manière directe et facile à comprendre.

Bubble chart GDP Countries



Bubble Chart : Utilisé pour représenter des variables de types mixtes (catégorielles ou numériques) et différentes unités de mesure (en \$, kg, etc.).

Histogram 3 stacked columns



Stacked Histogram : Préféré lorsque les données sont du même type et de la même unité de mesure.

Stacked Bar chart : L'approche d'empilage (stacking) peut être appliquée aux diagrammes en barres (**bar charts**).

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

1. Technique de Projection Géométrique :

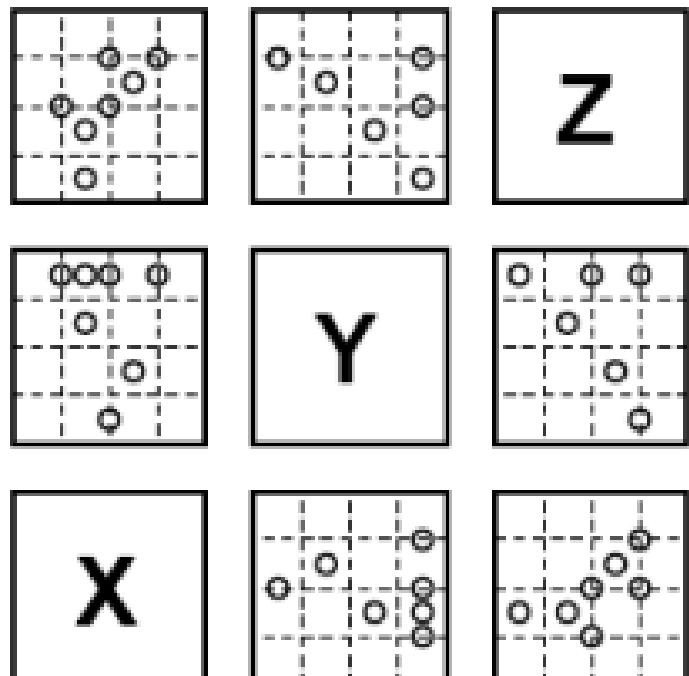
Utilisent des transformations mathématiques pour représenter des données multidimensionnelles dans des espaces de plus faibles dimensions.

Exemple : Scatter Plot Matrix

Scatter Plot Matrix : Représente les relations entre paires de variables.

- Chaque panneau dans la matrice est identifié par ses coordonnées de ligne et de colonne.
- Le panneau à la ième ligne et jème colonne est un nuage de points de X_j contre X_i .

Exemple de Scatter Plot Matrix



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

1. Techniques de Projection Géométrique : Suit

Lecture de la matrice de nuages de points :

Scatter Plot Matrix : Matrice de nuages de points avec trois variables X, Y et Z

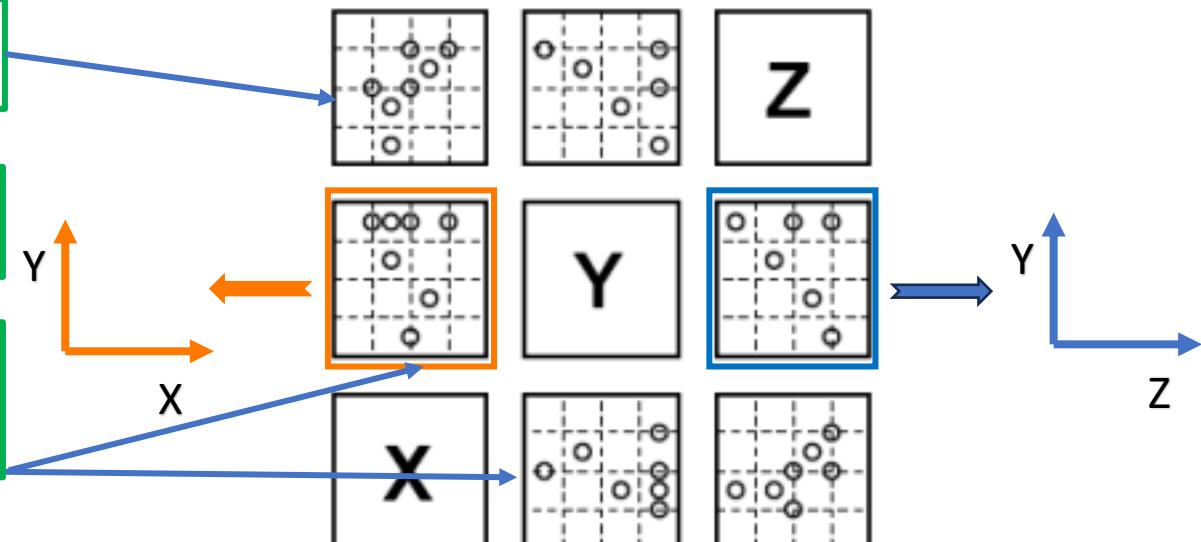
- X_j contre X_i est équivalent à dire ‘j’ en fonction de ‘i’ ou line en fonction de colonne. Tous ces expressions sont couramment utilisées pour décrire des relations dans des graphiques de dispersion.

Le panneau à la 1ère ligne et 1ère colonne est un nuage de points de Z contre X.

Les motifs peuvent être détectés à la fois horizontalement et verticalement.

Les panneaux symétriques par rapport à la diagonale XYZ ont les mêmes variables comme coordonnées, mais tournées de 90°. Cette redondance améliore le lien visuel.

Scatter Plot Matrix avec trois variables X, Y et Z



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée



2. Techniques Orientées Pixels :

Affectent chaque pixel de l'écran à une valeur de donnée, optimisant ainsi l'utilisation de l'espace disponible. **Exemple : Heatmaps**

Heatmaps (la carte thermique) : Montre des valeurs de données par des variations de couleur.

- Visualisation de Données : Utilise des variations de couleur pour montrer les valeurs des données.
- Analyse de Corrélation : Idéal pour visualiser des matrices de corrélation, des densités de données ou des fréquences.
- Comparaison Facile : Permet de comparer facilement les valeurs à travers différentes catégories ou périodes.
- Détection de Motifs : Révèle des motifs et des tendances qui peuvent être invisibles dans les données brutes.
- Code Couleur : Les couleurs plus chaudes (rouge, orange) représentent souvent des valeurs plus élevées, tandis que les couleurs plus froides (bleu, vert) représentent des valeurs plus faibles.
- Grandes Quantités de Données : Idéal pour analyser de grandes quantités de données ou des ensembles de données complexes.
- Analyse Multidimensionnelle : Pratique pour l'analyse de données multidimensionnelles.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

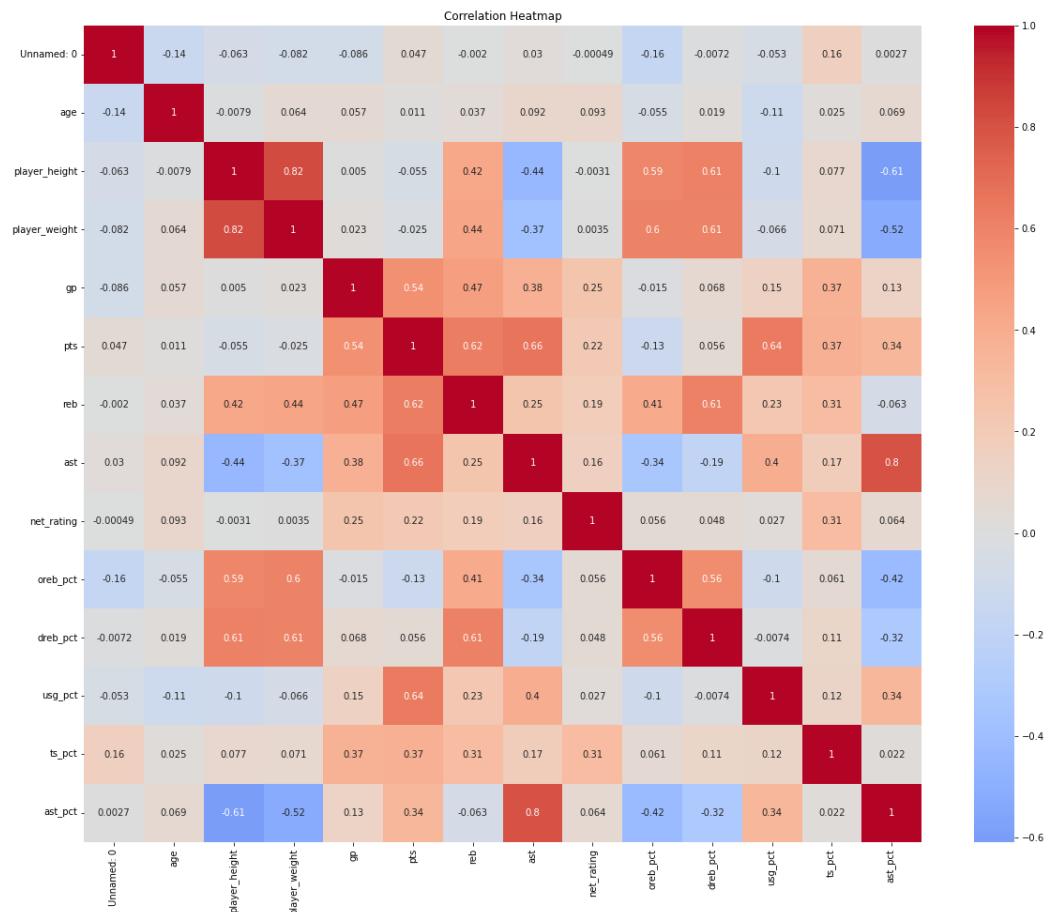
2. Techniques Orientées Pixels : Suit

"heatmap" de la matrice de corrélation pour l'analyse probabiliste des joueurs NBA :

Nous avons créé une heatmap dans une application précédente pour visualiser la matrice de corrélation dans l'analyse probabiliste des joueurs NBA.

C'est une excellente méthode pour l'analyse exploratoire des données afin de comprendre les corrélations entre différentes colonnes.

NBA Heatmap



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée



2. Techniques Orientées Pixels : Suit

Conseils pour Lire une Heatmap :

Comprendre l'Échelle de Couleur : Identifiez les couleurs qui représentent les valeurs hautes et basses. Généralement, les couleurs plus chaudes indiquent des valeurs élevées et les couleurs plus froides indiquent des valeurs faibles.

Examiner les Motifs : Recherchez des motifs ou des regroupements de couleurs similaires. Cela peut indiquer des corrélations fortes ou des anomalies dans les données.

Légende et Étiquettes : Utilisez la légende et les étiquettes des axes pour identifier quelles variables sont corrélées.

Analyse des Corrélations : Notez les paires de variables avec des corrélations élevées (positives ou négatives). Cela peut fournir des insights importants pour l'analyse.

Comparaison Horizontale et Verticale : Les heatmaps permettent de détecter des motifs à la fois horizontalement et verticalement, facilitant l'analyse multidimensionnelle.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

3. Techniques Basées sur les Icônes :

Représentent les données à l'aide de symboles visuels, où chaque dimension est mappée à une caractéristique de l'icône. Exemple : Radar Chart / Star Plots .

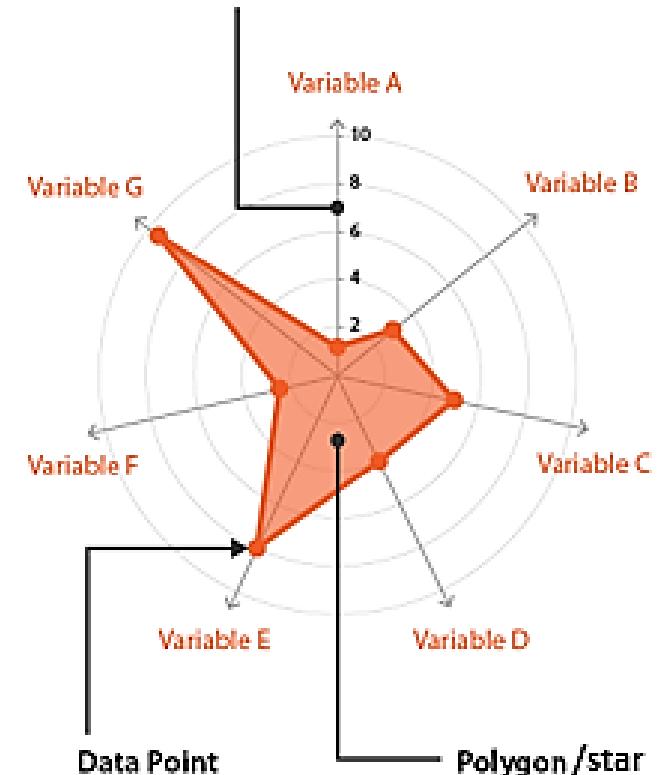
Star Plots (Graphiques en étoile) : Connu également sous le nom de « Radar chart »

Les graphiques radar utilisent des symboles visuels pour représenter plusieurs variables, chaque "rayon" ou "branche" de l'icône représentant une variable distincte. Cela permet de visualiser les valeurs de plusieurs variables en une seule **figure iconique**.

- **Représentation** : Chaque enregistrement de données est représenté par une figure en forme d'étoile avec un rayon pour chaque variable.
- **Longueur des Rayons** : La longueur de chaque rayon est proportionnelle à la valeur de sa variable correspondante.
- **Normalisation** : Chaque variable est généralement normalisée entre un nombre très petit (proche de 0) et 1.
- **Connexions** : Les extrémités ouvertes des rayons sont généralement connectées par des lignes.

Composants d'un Radar Chart

Axis and Scale



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

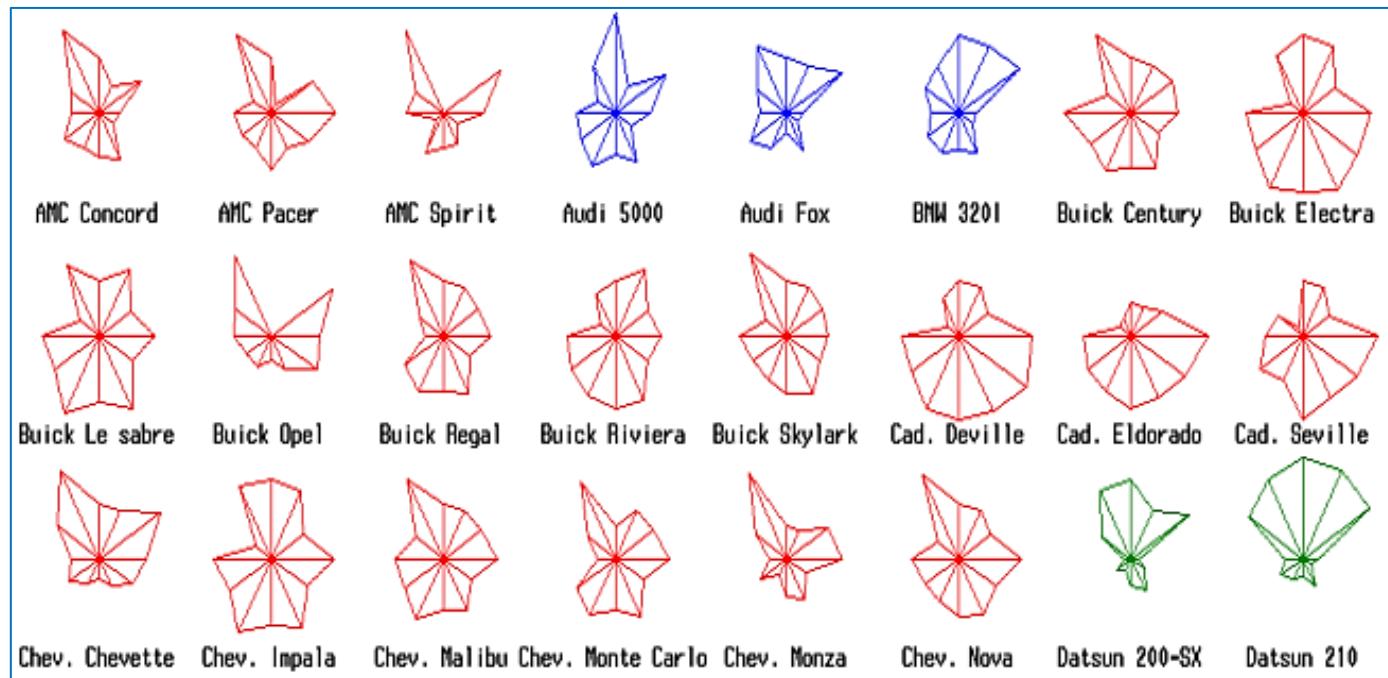
Techniques de Visualisation Multivariée

3. Techniques Basées sur les Icônes : Suit

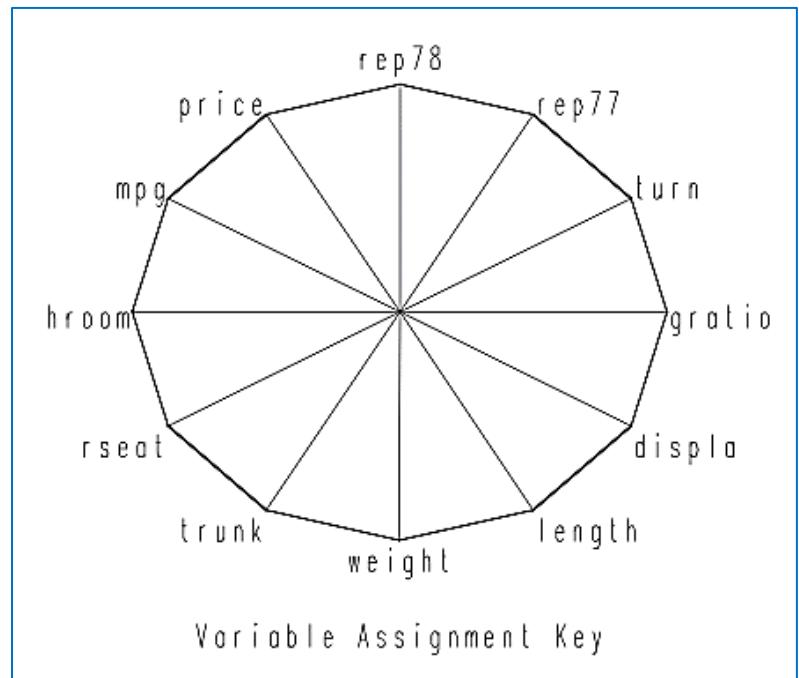
Exemple d'un Star Plot : Données automobiles

Représentation d'un ensemble de données automobiles avec 12 variables.

Représentation en étoiles d'un ensemble de données automatique avec 12 variables



Clé d'affectation des variables



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

3. Techniques Basées sur les Icônes : Suit

Exemple d'un Star Plot : Données des joueurs de football

Un autre exemple d'utilisation des graphiques en étoile (radar) peut être trouvé dans le domaine du football pour illustrer les performances des joueurs les uns par rapport aux autres.

Dans le graphique suivant, nous pouvons comparer les performances des joueurs en montrant les mesures de vitesse, tirs, passes, dribbles, défense et physique sur une échelle de 100 points.

Les graphiques radar, ou en étoile, trouvent également des applications courantes dans la comparaison des compétences des candidats par rapport à une offre spécifique. Ils permettent d'évaluer plusieurs compétences. Cette méthode offre une visualisation claire des points forts et des faiblesses des candidats par rapport aux exigences du poste.

Représentation en Radar d'un ensemble de données de football



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

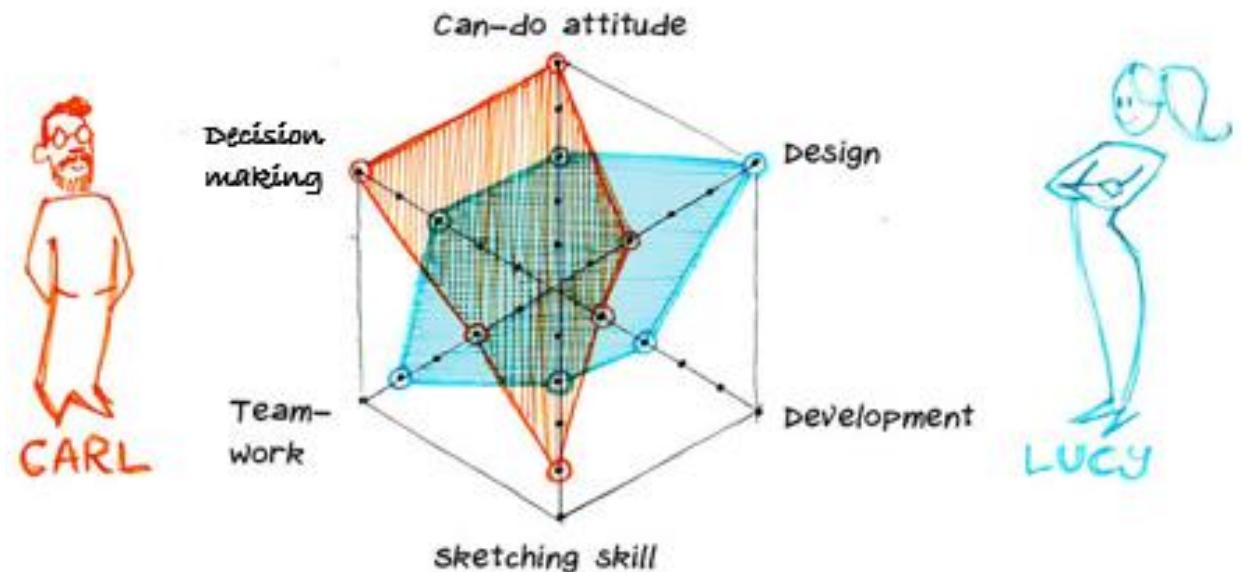
3. Techniques Basées sur les Icônes : Suit

Exemple d'un Star Plot :

Données d'évaluation psychotechnique et professionnelle.

Les graphiques radar, ou en étoile, trouvent également des applications courantes dans la comparaison des compétences des candidats par rapport à une offre spécifique. Ils permettent d'évaluer plusieurs compétences. Cette méthode offre une visualisation claire des points forts et des faiblesses des candidats par rapport aux exigences du poste.

Représentation en Radar d'un ensemble de données d'évaluation



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée



3. Techniques Basées sur les Icônes : Suit

Contraints et Amélioration possible sur un Star Plot (Radar Chart) :

Contraints :

Densité des Rayons : À mesure que le nombre de rayons augmente, il devient plus difficile de les distinguer.

Séparation : Les rayons doivent être séparés d'au moins 30° pour être discernables (max 12 variables).

Comparabilité des Données : Il est préférable que les données aient les mêmes unités de mesure pour permettre une comparaison visuelle entre les variables de la même ligne de données.

Améliorations possibles :

Normalisation des Données : Pour résoudre le problème de comparabilité, on peut normaliser les données afin qu'elles varient toutes de 0 à 1, facilitant ainsi la comparaison visuelle.

Propriétés Visuelles : Le nombre de rayons distinguables peut être augmenté en ajoutant des propriétés visuelles comme la couleur, la luminance, la largeur, etc.

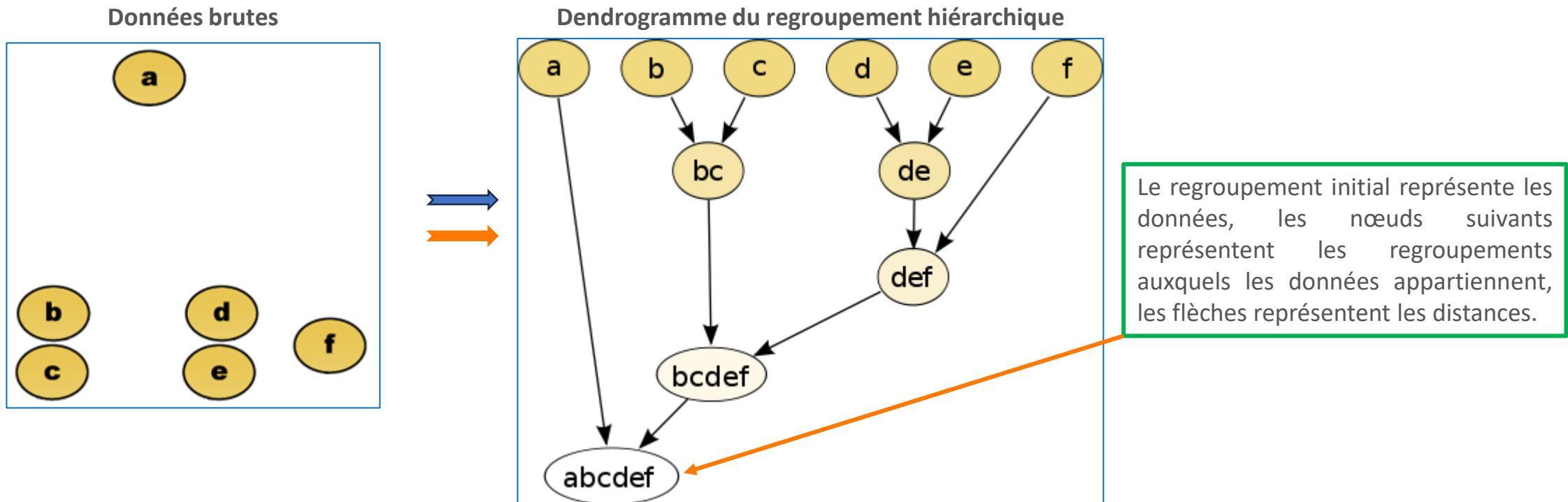
02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

4. Techniques Hiérarchiques :

Organisent les données en structures hiérarchiques pour révéler des relations à différents niveaux de granularité. Exemple : Dendrogrammes.

Dendrogrammes : Visualise des structures de données hiérarchiques comme des arbres.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Techniques de Visualisation Multivariée

5. Techniques Hybrides :

Combinent plusieurs approches pour tirer parti des avantages de chacune.

Exemple : « Radar Heat Map » Radial Plots combinés avec Heatmaps.

Radar HeatMap : Combine des diagrammes radiaux avec des cartes de chaleur pour une visualisation détaillée.

Axes Radiaux : Les axes partent du centre du cercle et s'étendent vers l'extérieur, représentant différentes variables.

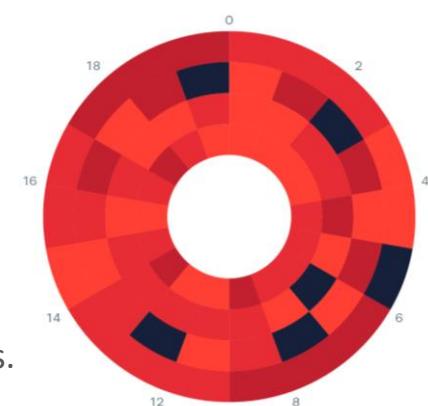
Variations de Couleur : Les heatmaps ajoutent une dimension de couleur pour indiquer l'intensité ou la fréquence des valeurs.

Analyse Multivariée : Permet de visualiser plusieurs variables simultanément et de voir leurs interactions et corrélations.

Patterns et Tendances : Les motifs circulaires et les gradients de couleur facilitent la détection de tendances et de relations complexes entre les variables.

Comparaison de Données : Idéal pour comparer des ensembles de données ou des performances sur différentes dimensions.

Radial Heatmap / Radar Heatmap



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

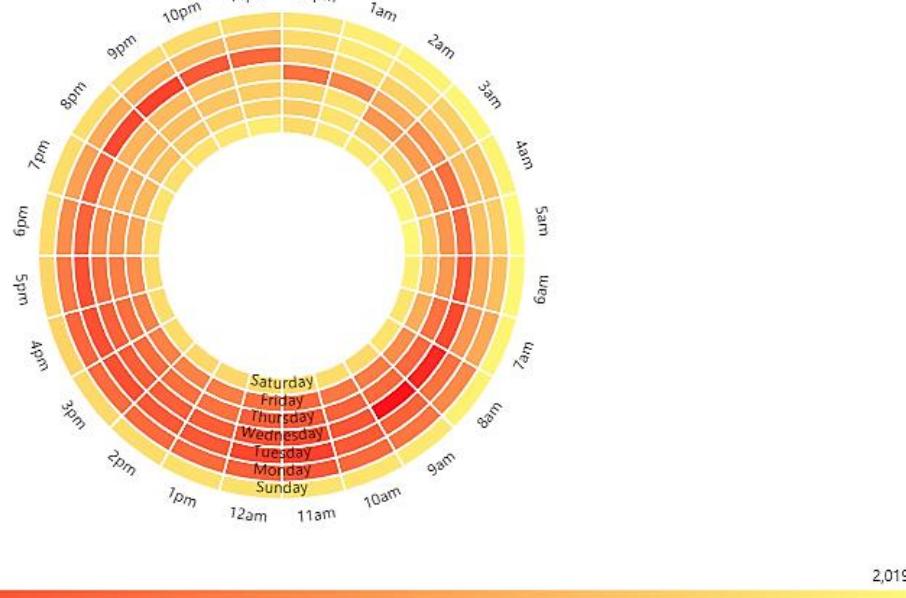
Techniques de Visualisation Multivariée

5. Techniques Hybrides : suivit

Exemple : Radial Heatmap de vents par heures pendant la semaine

Le Heatmap radial suivant illustre les données de vente de l'entreprise pour chaque heure de chaque jour de la semaine, mettant en évidence des ventes plus élevées pendant les week-ends et les heures de nuit.

Radial Heatmap Ventes par heures pendant la semaine



Pour mieux comprendre la gradation des couleurs de chaleur, nous consultons la légende des chaleurs en bas.



CHAPITRE 2

CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

1. Choix des graphiques en fonction d'objectif
2. Choix des graphiques en fonction des données
3. Exploration de la visualisation multivariée
- 4. Utilisation stratégique des couleurs et des annotations**
5. Sélection des graphiques avancés avec Python/R

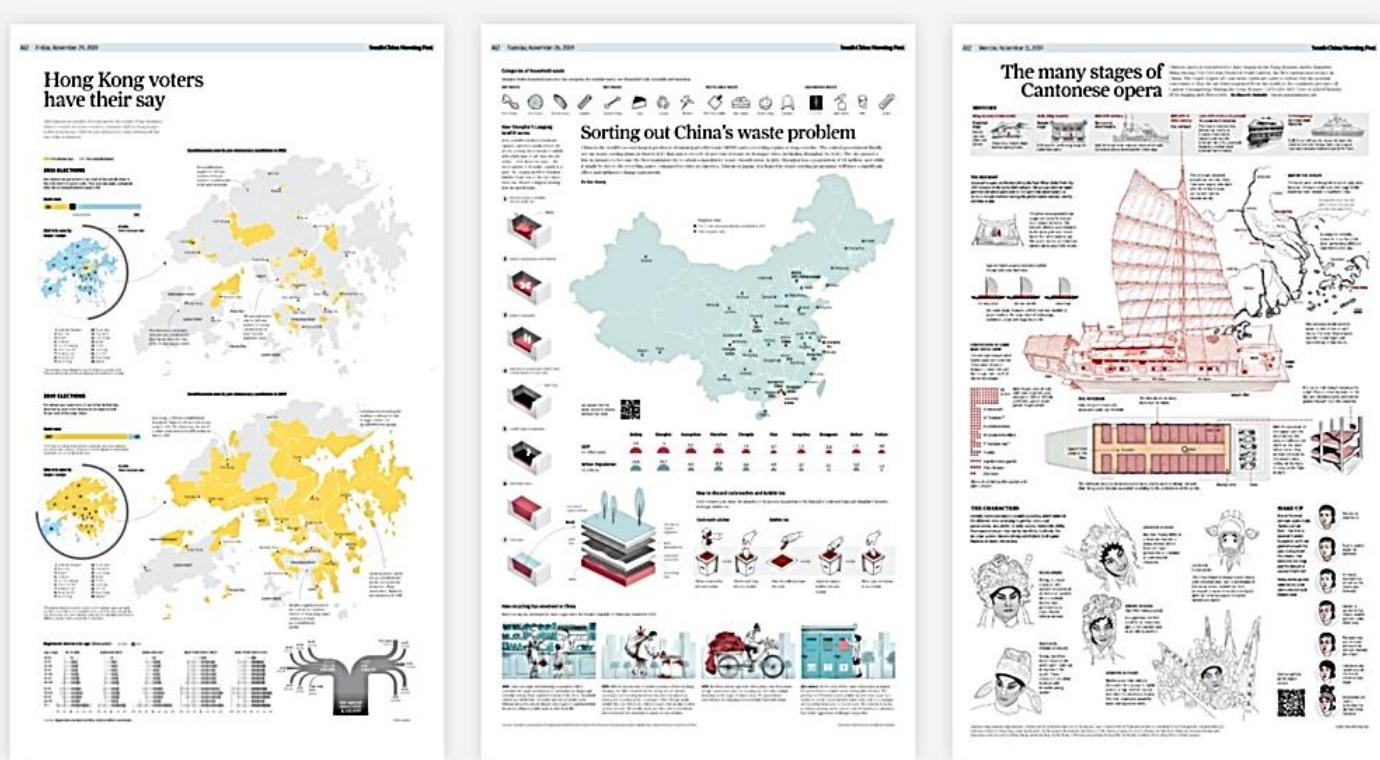
02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Introduction :

- L'utilisation stratégique des couleurs et des annotations est essentielle pour améliorer la clarté et l'impact des visualisations de données.

Trois pages du South China Morning Post



Pensez d'abord aux grandes images, que vos graphiques peuvent placer dans des tableaux de bord numériques, des publications sur des médias sociaux, des journaux ou des rapports. Pour cela, vous devez utiliser des annotations claires et une palette de couleurs qui peut s'adapter partout.

Il existe une combinaison de couleurs complémentaires qui est particulièrement appréciée par les concepteurs de visualisation de données : **jaune/orange/ rouge et bleu**.

Faites défiler les portfolios graphiques comme celui du South China Morning Post ou celui de The Economist, et vous remarquerez qu'ils utilisent ces couleurs beaucoup plus souvent que des couleurs comme le **violet** ou le **vert**.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Stratégie de couleurs :

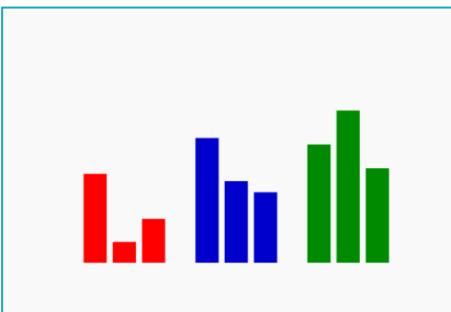
Astuce 0 : Élargissez votre compréhension des couleurs

Voici quelques points clés à prendre en compte :

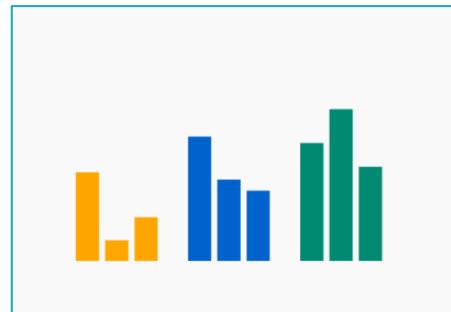
- Évitez d'utiliser les couleurs pures ; privilégiez plutôt les nuances, la transparence, etc.
- Les couleurs chaudes et le bleu sont super polyvalents pour les catégories.
- Le jaune, l'orange et le rouge sont très agréables ensemble  et également significatif, mais les gens les percevront toujours comme différents, ce qui est exactement ce que nous voulons pour les couleurs catégorielles.
- Et le bleu est plus flexible que toute autre teinte. Beaucoup de bleus, peu importe s'ils sont sombres ● ou léger ● ou saturés ● ou non saturé ●, Ayez l'air agréable, apaisant et professionnel.
- Et ils sont accessibles : les daltoniens peuvent facilement distinguer le bleu et l'orange/rouge l'un de l'autre.
- Donc, en cas de doute, utilisez un orange/rouge avec du bleu.

.....jaune / orange / rouge et bleu.
... et pas
.....jaune / orange / rouge et bleu.

Pas Idéal



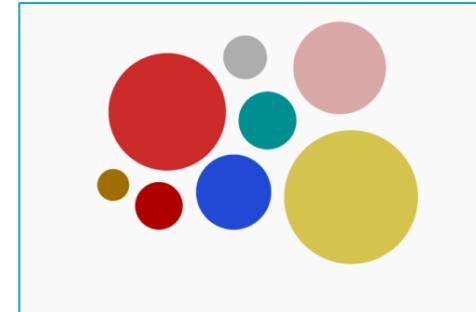
Mieux



Pas Idéal



Mieux



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

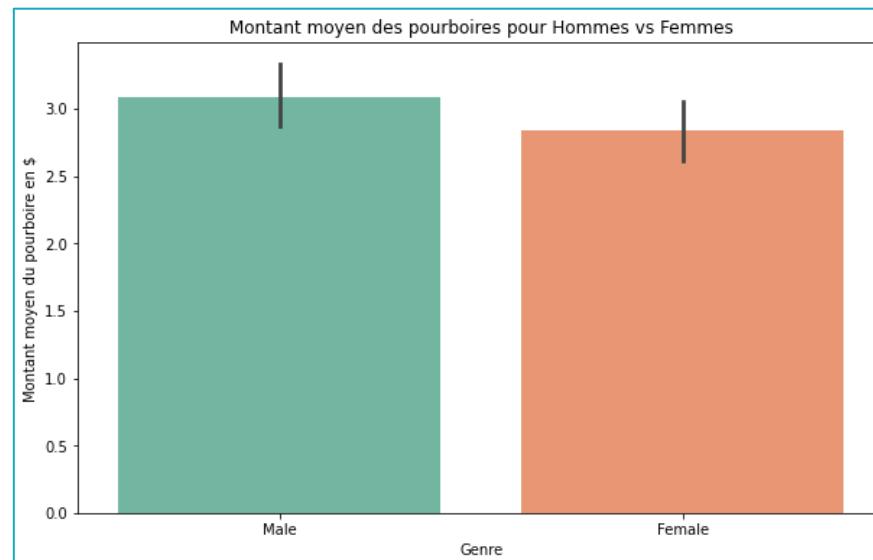
Utilisation stratégique des couleurs et des annotations

Stratégie de couleurs : Suit

1.Signification des Couleurs :

Consistance : Utilisez des couleurs de manière cohérente pour représenter des catégories ou des valeurs similaires.

Intuition : Choisissez des couleurs qui ont une signification intuitive (par exemple, rouge pour indiquer une alerte ou un danger, vert pour indiquer la sécurité ou la croissance).



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Stratégie de couleurs :

Couleurs : Suit

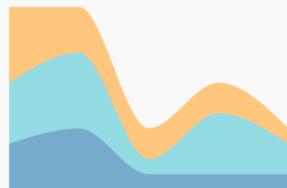
2. Contraste et Lisibilité :

Contraste : Assurez-vous qu'il y a suffisamment de contraste entre les couleurs pour rendre les données faciles à lire.

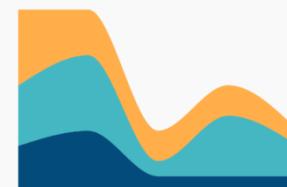
Lisibilité : Utilisez des couleurs qui sont facilement distinguables, même pour les personnes ayant des déficiences visuelles.

Eviter trop peu de contraste avec l'arrière-plan

Pas Idéal

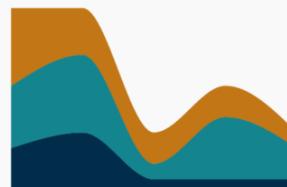


Mieux



Évitez trop de contraste avec l'arrière-plan

Pas Idéal



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Stratégie de couleurs :

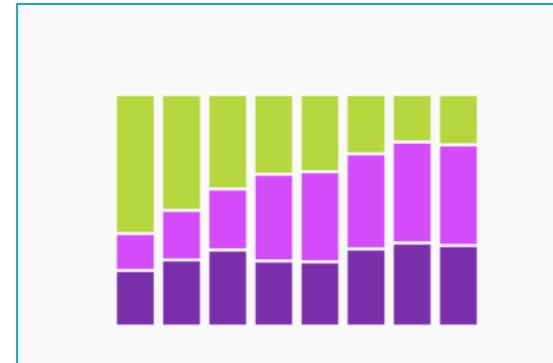
Couleurs : Suit

3. Palette de Couleurs :

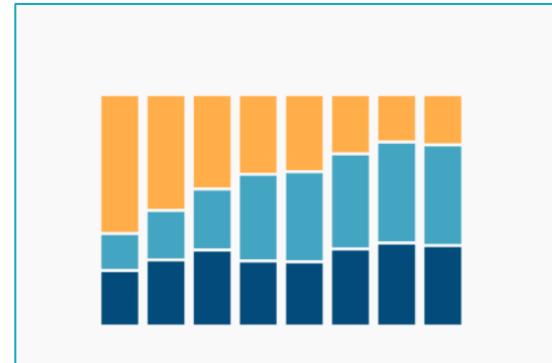
Palette Restreinte : Limitez le nombre de couleurs pour éviter la surcharge visuelle.

Palette Adaptée : Utilisez des palettes de couleurs adaptées à la nature des données (par exemple, une palette séquentielle pour des données continues, une palette catégorielle pour des données discrètes).

Pas Idéal



Mieux



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Application des consignes dans la création d'un plot :

- Pour appliquer les consignes d'utilisation stratégique des couleurs et des annotations dans la création de graphiques, nous allons aborder les étapes en utilisant les bibliothèques Python (Seaborn) et R (ggplot2).

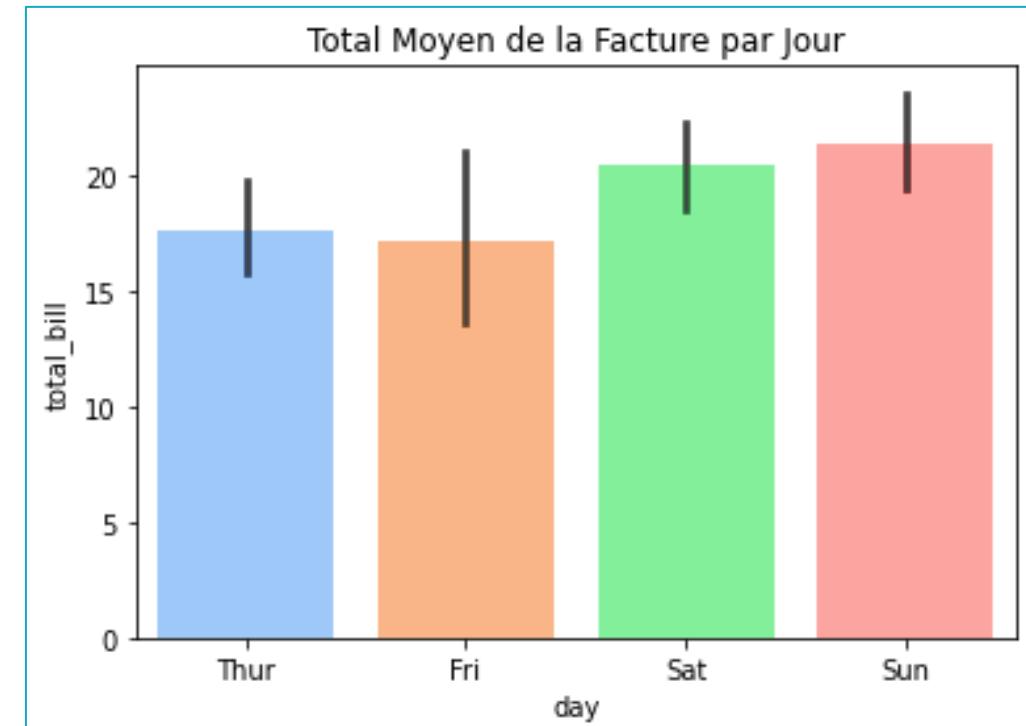
Python (Seaborn) : Application sur tips data set

Barplot :

```
import seaborn as sns  
import matplotlib.pyplot as plt  
# Charger le dataset  
tips = sns.load_dataset("tips")  
# Obtenir les jours uniques  
unique_days = tips['day'].unique()  
# Définir une palette de couleurs spécifique pour chaque jour  
palette = sns.color_palette("pastel", len(unique_days))  
# Tracé avec la palette définie  
sns.barplot(x="day", y="total_bill", data=tips, palette=palette)  
plt.title('Total Moyen de la Facture par Jour')  
plt.show()
```

Assurez-vous que la palette contient autant de couleurs que le nombre de jours différents dans les données.

Utilisation de `sns.color_palette("pastel")` pour une palette de couleurs pastel.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

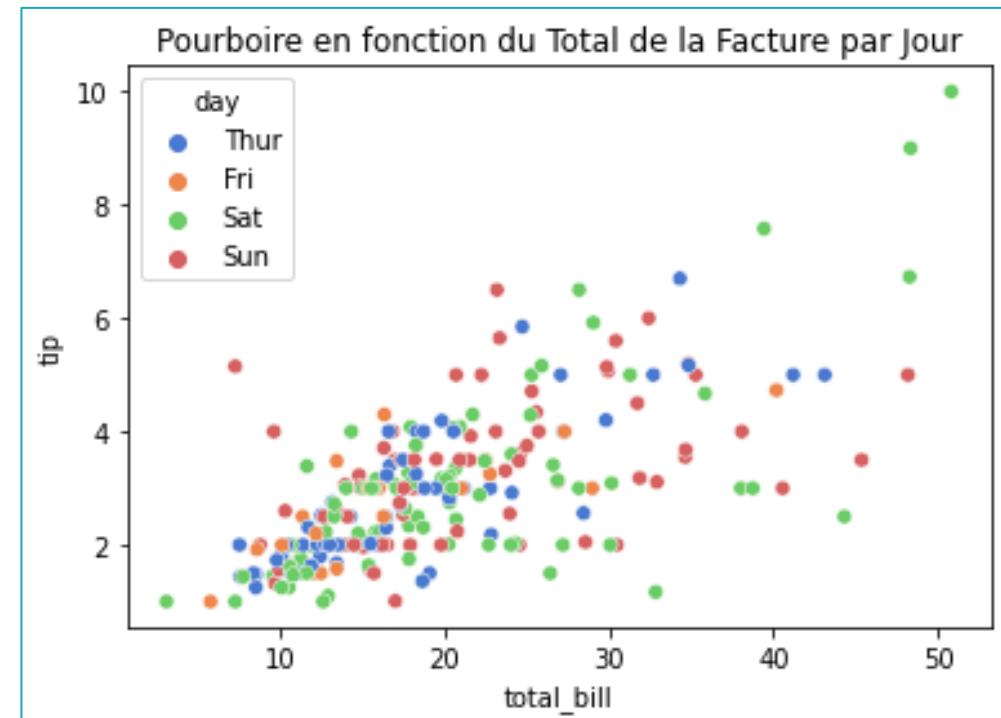
Application des consignes dans la création d'un plot :

Python (Seaborn) : Application sur tips data set, suit

Scatterplot :

```
import seaborn as sns
import matplotlib.pyplot as plt
# Charger le dataset
tips = sns.load_dataset("tips")
# Obtenir les jours uniques
unique_days = tips['day'].unique()
# Définir une palette de couleurs spécifique pour chaque jour
palette = sns.color_palette("muted", len(unique_days))
# Tracé avec la palette définie
sns.scatterplot(x="total_bill", y="tip", hue="day", data=tips, palette=palette)
plt.title('Pourboire en fonction du Total de la Facture par Jour')
plt.show()
```

Utilisation de sns.color_palette("muted") pour une palette de couleurs atténuees.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

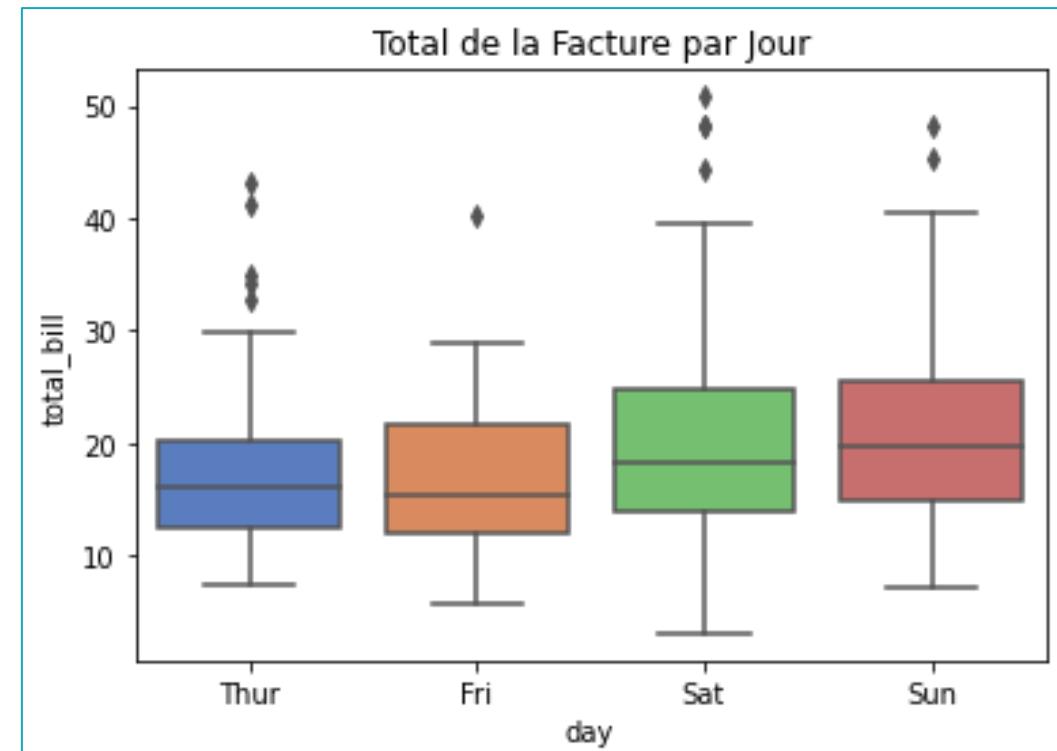
Utilisation stratégique des couleurs et des annotations

Application des consignes dans la création d'un plot :

Python (Seaborn) : Application sur tips data set, suit

Boxplot :

```
import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Charger le dataset  
tips = sns.load_dataset("tips")  
  
# Obtenir les jours uniques  
unique_days = tips['day'].unique()  
  
# Définir une palette de couleurs spécifique pour chaque jour  
palette = sns.color_palette("muted", len(unique_days))  
  
# Tracé avec la palette définie  
sns.boxplot(x="day", y="total_bill", data=tips, palette=palette)  
plt.title('Total de la Facture par Jour')  
plt.show()
```



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

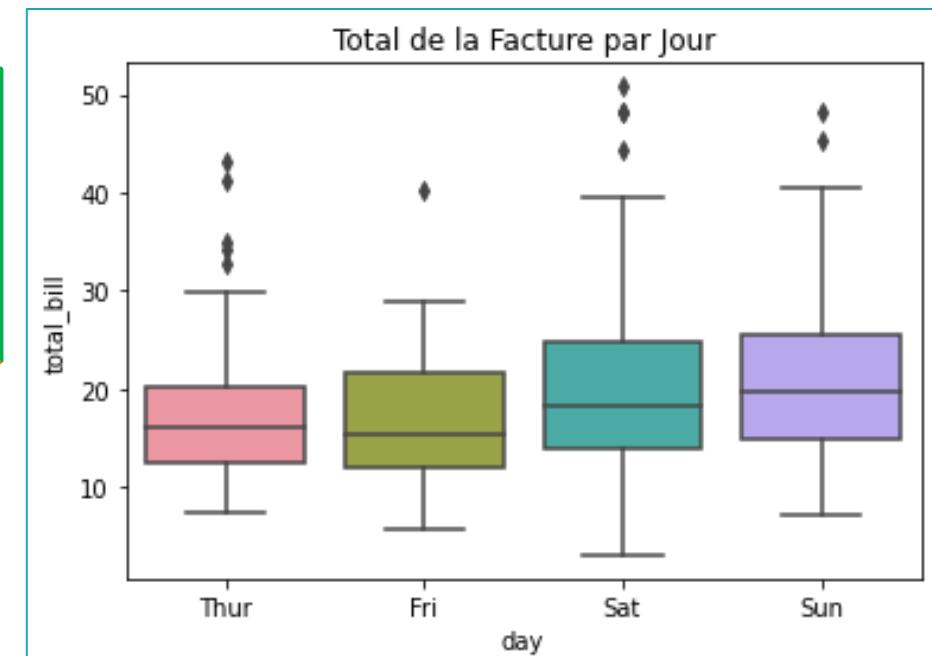
Création de votre propre couleur Pallet :

Python (Seaborn) : Application sur tips data set, suit

Boxplot :

```
import seaborn as sns
import matplotlib.pyplot as plt
# Charger le dataset
tips = sns.load_dataset("tips")
# Définir la palette de couleurs
sns.set_palette(sns.color_palette(['#4E79A7', '#F28E2B'])) # Bleu et Orange
# Tracé
sns.boxplot(x="day", y="total_bill", data=tips)
plt.title('Total de la Facture par Jour')
plt.show()
```

Nous avons défini une palette de seulement deux couleurs. Seaborn étend automatiquement cette palette aux catégories supplémentaires en utilisant des variations des couleurs de base si nécessaire, ou en recyclant les couleurs définies, comme montré dans l'exemple ci-dessus.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

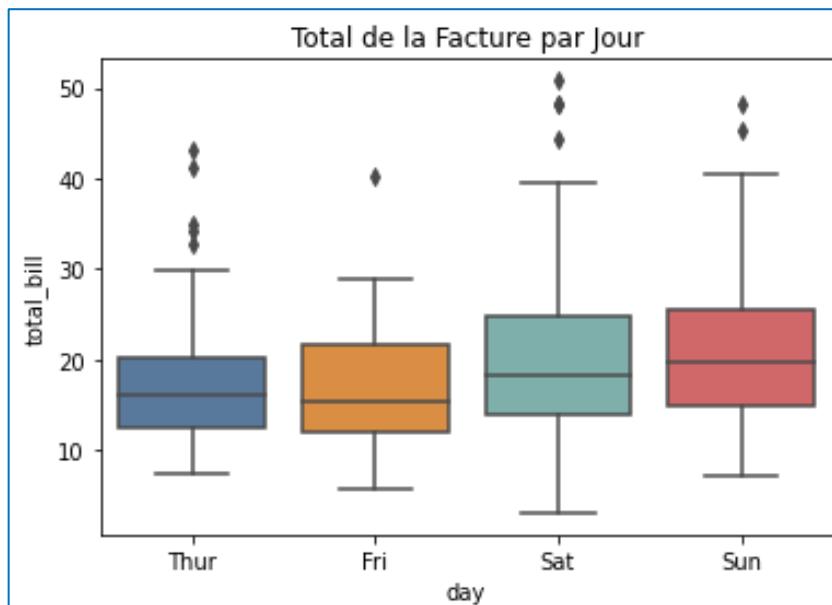
Utilisation stratégique des couleurs et des annotations

Création de votre propre couleur Pallet :

Python (Seaborn) : Application sur tips data set, suit

Si vous voulez un contrôle précis sur chaque catégorie, vous devez définir explicitement les couleurs pour chaque catégorie présente dans vos données :

```
# Définir une palette de couleurs spécifique pour chaque catégorie de jours  
palette = {'Thur': '#4E79A7', 'Fri': '#F28E2B', 'Sat': '#76B7B2', 'Sun': '#E15759'}
```



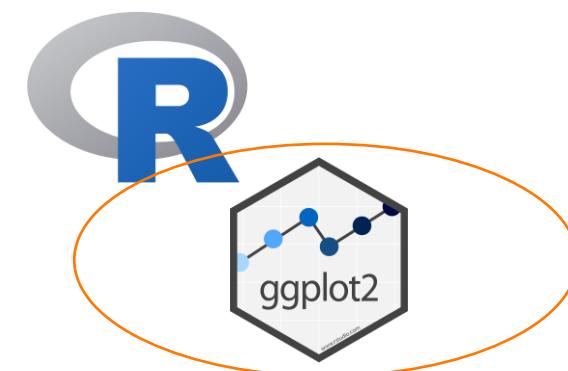
02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Bibliothèques des palettes de couleurs sur Python et R

Matplotlib et Seaborn : Ces bibliothèques Python permettent de définir des palettes de couleurs personnalisées et de les utiliser dans vos graphiques. Dans les exemples, nous utilisons une palette bleue et orange pour respecter les principes des couleurs complémentaires et de l'accessibilité visuelle.

ggplot2 : En R, la fonction `scale_color_manual` permet de définir manuellement les couleurs pour différents groupes dans vos données. Le package `viridis` propose des palettes de couleurs uniformes et adaptées aux daltoniens, garantissant une lisibilité et une accessibilité élevées.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations



Annotations :

Les **annotations** enrichissent les visualisations de données en ajoutant des explications claires, contextuelles et précises.

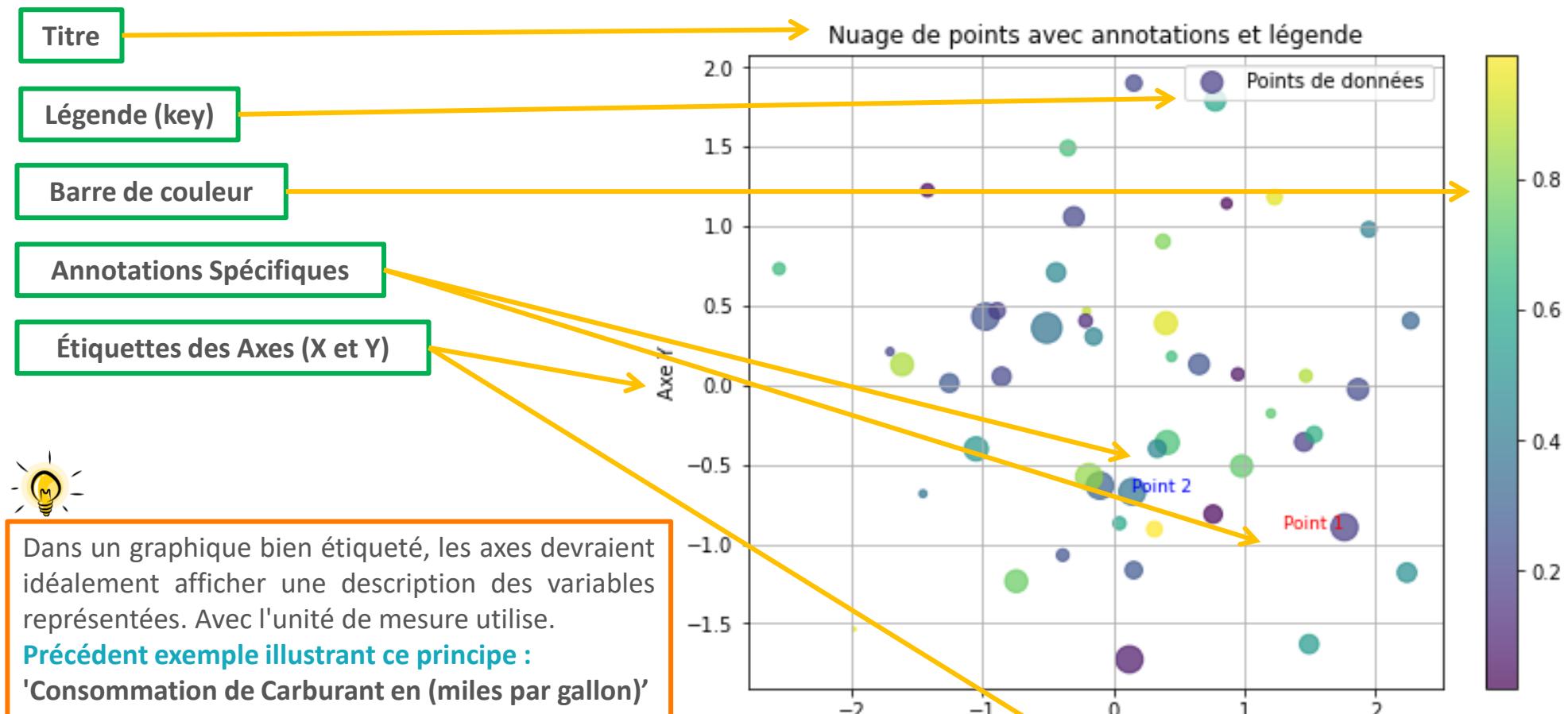
Les Annotations incluent à la fois les étiquettes des axes X et Y, le titre du graphique et les explications spécifiques aux points de données. Elles peuvent également comprendre des légendes pour clarifier les couleurs et les symboles utilisés dans le graphique. Voici une description plus détaillée :

- **Étiquettes des Axes (X et Y)** : Indiquent ce que représentent les axes pour une meilleure compréhension des données.
- **Titre** : Fournit un résumé concis du sujet ou de l'objectif du graphique.
- **Légendes (key)** : Clarifient les représentations des couleurs et des symboles.
- **Annotations Spécifiques** : Explique les points clés ou les anomalies dans les données pour ajouter du contexte et de la clarté
- **Barres de couleur** : Représenter graphiquement la correspondance entre des valeurs numériques et des couleurs. Elles servent souvent de légende interactive, permettant aux spectateurs de comprendre rapidement quelle couleur correspond à quelle gamme de valeurs ou à quel niveau de mesure.
- **Autres** : Peuvent inclure des flèches, des lignes de repère, et d'autres éléments visuels qui aident à interpréter et à contextualiser les données représentées.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations

Annotations :



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Utilisation stratégique des couleurs et des annotations



Annotations :

Les **annotations** doivent respecter les critères suivants :

- **Clarté et Compréhension** : Les annotations doivent être descriptives et directes, offrant des explications claires des points clés des données.
- **Positionnement Précis et Non-Obstruction** : Placer les annotations près des points de données pour éviter toute confusion, sans masquer les informations cruciales.
- **Contexte et Pertinence** : Fournir un contexte adéquat à travers les annotations aide à situer les données dans une perspective plus large, permettant ainsi une meilleure compréhension globale.

Pendant ce cours, nous avons adopté des annotations claires et descriptives. Essayez également, dans vos futurs graphiques, d'adopter des annotations claires et basées sur le contexte pour clarifier et enrichir vos visualisations.



CHAPITRE 2

CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

1. Choix des graphiques en fonction d'objectif
2. Choix des graphiques en fonction des données
3. Exploration de la visualisation multivariée
4. Utilisation stratégique des couleurs et des annotations
5. Sélection des graphiques avancés avec Python/R

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R



Sélection des graphiques avancés avec Python



1. Création de graphique avec Annotations Spécifiques

Dans le cadre de l'analyse de données, il est souvent essentiel de créer des graphiques qui non seulement affichent les données de manière claire, mais qui incluent également des annotations spécifiques pour mettre en évidence des points critiques ou fournir des explications contextuelles. Les annotations peuvent inclure des étiquettes, des titres, des légendes ou d'autres éléments visuels qui aident à interpréter les informations présentées.

Script Python :

```
import matplotlib.pyplot as plt  
import numpy as np  
  
# Génération de données aléatoires  
np.random.seed(0)  
x = np.random.randn(50)  
y = np.random.randn(50)  
sizes = np.abs(np.random.randn(50)) * 100  
colors = np.random.rand(50)
```

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec Python

1. Création de graphique bubble chart avec Annotations Spécifiques

Script Python : suit

```
# Création d'un graphique de dispersion avec annotations, ou plus précisément d'un bubble chart.
```

```
plt.figure(figsize=(8, 6))  
plt.scatter(x, y, s=sizes, c=colors, alpha=0.7, cmap='viridis', label='Points de données')
```

```
# Ajout des labels des axes
```

```
plt.xlabel('Axe X')
```

```
plt.ylabel('Axe Y')
```

```
# Ajout du titre
```

```
plt.title('Graphique de dispersion avec Annotations')
```

```
# Annotations pour des points spécifiques
```

```
plt.text(x[0], y[0], 'Point 1', fontsize=9, ha='right', color='red')
```

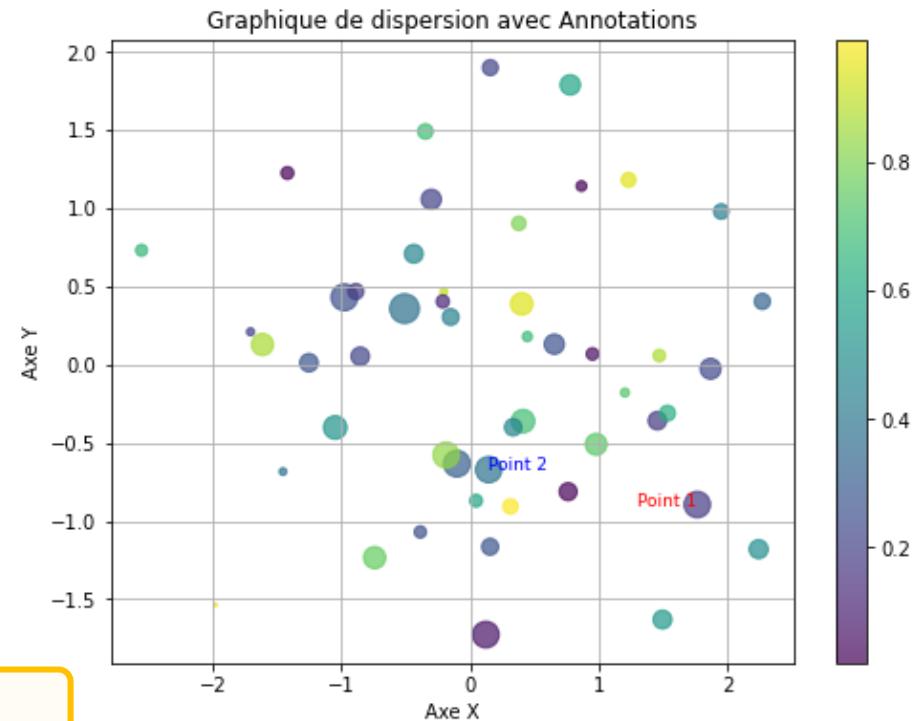
```
plt.text(x[10], y[10], 'Point 2', fontsize=9, ha='left', color='blue')
```

```
# Affichage du graphique
```

```
plt.grid(True)
```

```
plt.colorbar()
```

```
plt.show()
```



Rappel :

Un graphique à bulles (bubble chart) est un sous-type spécifique de nuage de points (scatter plot). Contrairement à un scatter plot simple où toutes les observations sont représentées par des points de taille uniforme, dans un bubble chart, chaque point peut avoir une taille différente, représentant une troisième dimension de données.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec Python

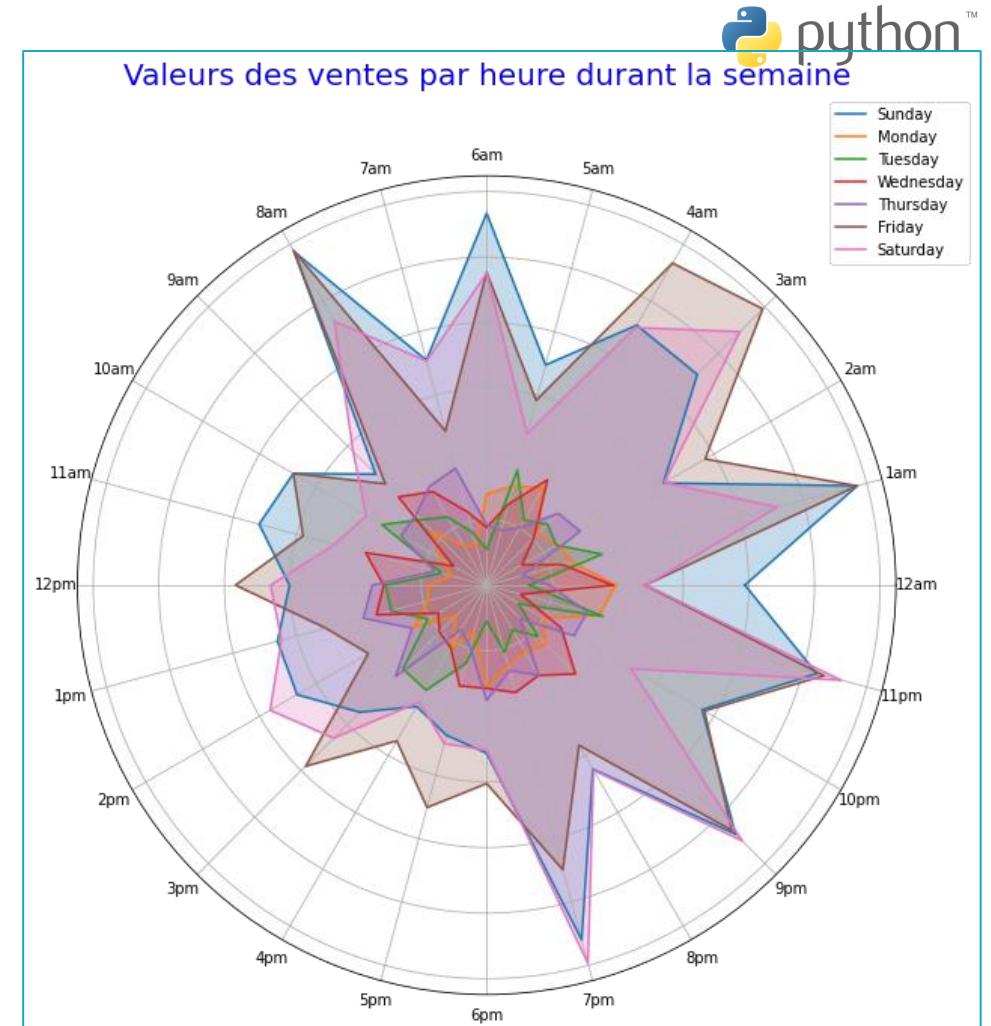
2. Création de graphique Radar Heatmap des ventes

Dans cette application, nous allons créer un graphique radar qui montre les ventes pour chaque heure de chaque jour de la semaine, avec des ventes plus élevées les week-ends et les nuits.

Étapes de création du graphique :

1. Chargement des bibliothèques

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



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec Python



Étapes de création du graphique : suit

2. Génération des données :

```
jours = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
heures = ["12am", "1am", "2am", "3am", "4am", "5am", "6am", "7am", "8am", "9am", "10am", "11am",
          "12pm", "1pm", "2pm", "3pm", "4pm", "5pm", "6pm", "7pm", "8pm", "9pm", "10pm", "11pm"]
data = []
for jour in jours:
    for heure in heures:
        if jour in ["Friday", "Saturday", "Sunday"] and heure in ["7pm", "8pm", "9pm", "10pm", "11pm", "12am", "1am",
          "2am"]:
            valeur = np.random.randint(8000, 12000)
        elif jour in ["Friday", "Saturday", "Sunday"]:
            valeur = np.random.randint(4000, 8000)
        else:
            valeur = np.random.randint(1000, 4000)
        data.append([jour, heure, valeur])
df = pd.DataFrame(data, columns=["jour", "heure", "vents en Dh"])
```

Définir les jours de la semaine et les heures de la journée.

Générer les valeurs de ventes aléatoires en fonction des jours et des heures spécifiques.

Data Fram généré :

| | jour | heure | vents en Dh |
|---|--------|-------|-------------|
| 0 | Sunday | 12am | 8895 |
| 1 | Sunday | 1am | 8644 |
| 2 | Sunday | 2am | 8789 |
| 3 | Sunday | 3am | 6921 |

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R



Sélection des graphiques avancés avec Python



Étapes de création du graphique : suit

3. Pivotement des données:

```
df_pivot = df.pivot(index='heure', columns='jour', values='valeur')
```

4. Initialisation du graphique radar:

```
fig, ax = plt.subplots(figsize=(10, 10), subplot_kw=dict(polar=True))
```

5. Création du graphique radar:

```
for jour in jours:
```

```
    valeurs = df_pivot[jour].values
```

```
    angles = np.linspace(0, 2 * np.pi, len(heures), endpoint=False).tolist()
```

```
    valeurs = np.concatenate((valeurs, [valeurs[0]]))
```

```
    angles += angles[:1]
```

```
    ax.plot(angles, valeurs, label=jour)
```

```
    ax.fill(angles, valeurs, alpha=0.25)
```

Réorganiser les données pour faciliter leur utilisation dans le tracé radar.

Créer une figure avec des axes polaires.

Pour chaque jour, tracer les valeurs des ventes sur le graphique radar.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec Python



Étapes de création du graphique : suit

6. Ajout des étiquettes et du titre:

```
ax.set_yticklabels([])  
  
ax.set_xticks(np.linspace(0, 2 * np.pi, len(heures), endpoint=False))  
  
ax.set_xticklabels(heures)  
  
plt.title("Ventes par heure durant la semaine", size=20, color='blue', y=1.1)  
  
plt.legend(loc='upper right', bbox_to_anchor=(1.1, 1.1))
```

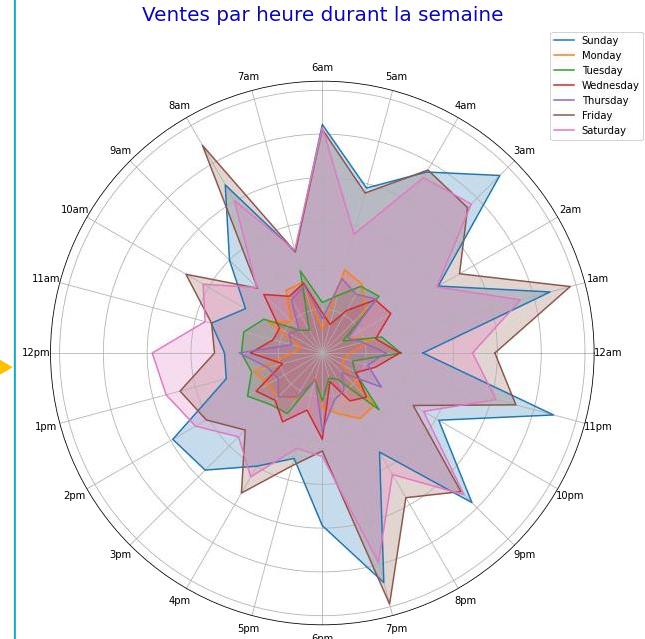
7. Affichage du graphique:

```
plt.show()
```

Note : Chaque exécution du processus de génération de données produira des valeurs distinctes, entraînant ainsi des trajectoires variées pour chaque jour de la semaine.

Conclusion : Avec ces étapes, vous serez en mesure de créer un graphique radar complet qui visualise les ventes par heure et par jour de la semaine, en mettant en évidence les périodes de ventes plus élevées.

Ajouter des étiquettes aux axes et un titre au graphique.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec Python

3. Création Heatmap des joueurs NBA de basketball :

Pour créer une heatmap de corrélation en Python, vous pouvez suivre les étapes suivantes. Le jeu de données provient d'une activité précédente d'analyse des joueurs de la NBA, où nous avons calculé des probabilités, généré et visualisé une heatmap à partir d'une matrice de corrélation comme étape de l'exploration des données (EDA).

Étapes de création du graphique :

1. Création de la matrice de corrélation :

```
correlation_matrix = df.corr()
```

2. Création de Heatmap :

```
# Définir la taille du graphique
```

```
plt.figure(figsize=(20, 16))
```

```
# Créer la heatmap
```

```
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0)
```

```
# Ajouter un titre
```

```
plt.title('Correlation Heatmap')
```

```
# Afficher le graphique
```

```
plt.show()
```

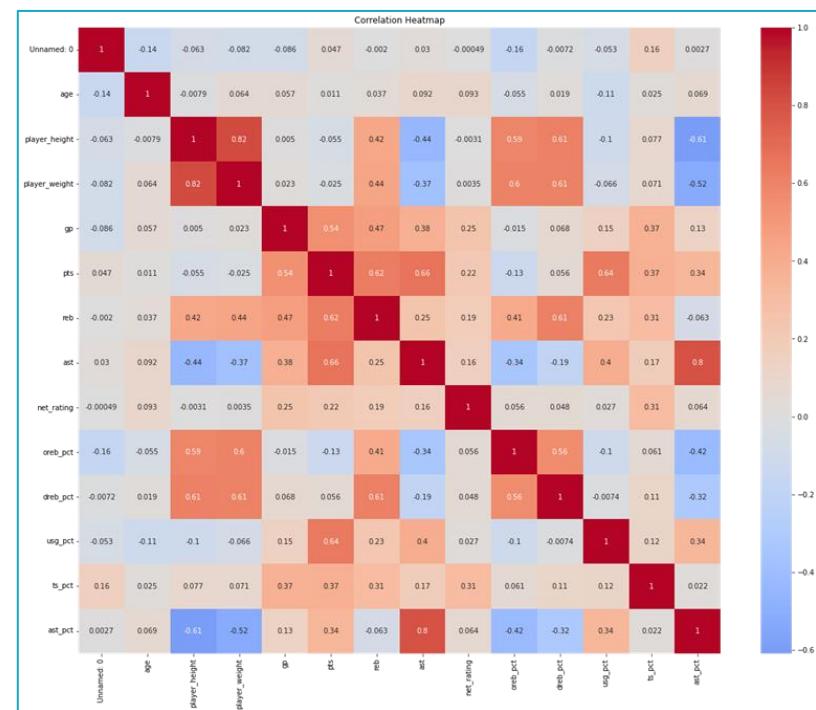
```
# 0.1 Chargement des bibliothèques :
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# 0.2 Chargement des données :
df = pd.read_csv('E:/../Application sur
Python/Probability/NBA_all_seasons.csv')
```



Lien de Téléchargement du Dataset sur kaggle :

<https://www.kaggle.com/datasets/justinas/nba-players-data>

| | Unnamed: 0 | age | player_height | player_weight |
|---------------|------------|-----------|---------------|---------------|
| Unnamed: 0 | 1.000000 | -0.136497 | -0.062835 | -0.081557 |
| age | -0.136497 | 1.000000 | -0.007904 | 0.063561 |
| player_height | -0.062835 | -0.007904 | 1.000000 | 0.822141 |
| player_weight | -0.081557 | 0.063561 | 0.822141 | 1.000000 |



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R



Sélection des graphiques avancés avec Python



3. Création Heatmap des joueurs NBA de basketball : suit

Paramètres utilisés dans sns.heatmap()

correlation_matrix : Il s'agit de la matrice de corrélation que nous voulons visualiser. Elle est créée à partir des données du Data Frame en utilisant la méthode `.corr()`.

- Utilisation : `correlation_matrix = df.corr()`

annot=True : Ce paramètre permet d'afficher les valeurs de la matrice directement sur la carte thermique.

• Utilisation : En réglant **annot** sur **True**, chaque cellule de la carte thermique affichera la valeur de corrélation correspondante.

cmap='coolwarm' : Ce paramètre spécifie la palette de couleurs à utiliser pour la carte thermique.

• Utilisation : '**coolwarm**' est une **palette de couleurs** qui va du **bleu** (valeurs basses) au **rouge** (valeurs hautes), ce qui permet de visualiser facilement les corrélations négatives et positives.

center=0 : Ce paramètre définit la valeur autour de laquelle la carte thermique doit être centrée.

• Utilisation : En réglant center à **0**, les couleurs de la carte thermique seront équilibrées autour de zéro, ce qui permet de distinguer plus clairement les corrélations **positives** et **négatives**.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

0. Contexte de l'application : Dataset attitude

Dans cette application, nous allons utiliser le jeu de données « **attitude** » disponible dans le package **datasets** de R. Ce jeu de données contient les résultats d'une enquête sur les performances des employés dans une entreprise. Chaque employé a rempli le questionnaire en donnant ses propres évaluations sur plusieurs aspects de son travail et de son environnement de travail. Les évaluations portent sur les points suivants :

rating : Évaluation globale de la satisfaction de l'employé.

raises : Opinion de l'employé sur les augmentations de salaire.

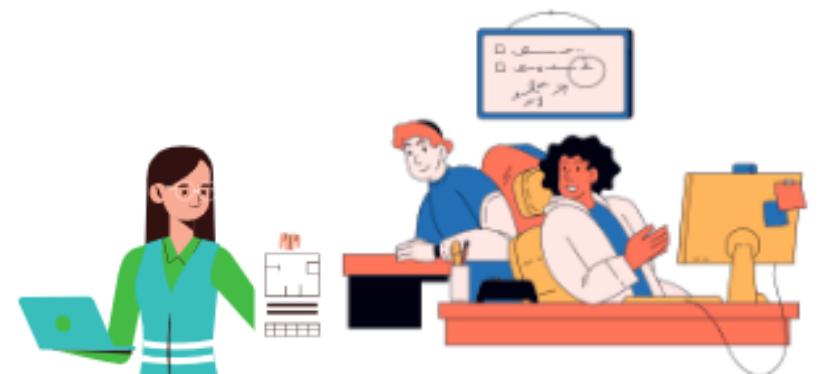
privileges : Sentiment de l'employé quant aux priviléges accordés.

advance : Perception des opportunités d'avancement au sein de l'entreprise.

learning : Évaluation de l'apprentissage et des formations disponibles.

complaints : Nombre de plaintes formulées par l'employé.

critical : Niveau de critique de l'employé concernant son environnement de travail.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R



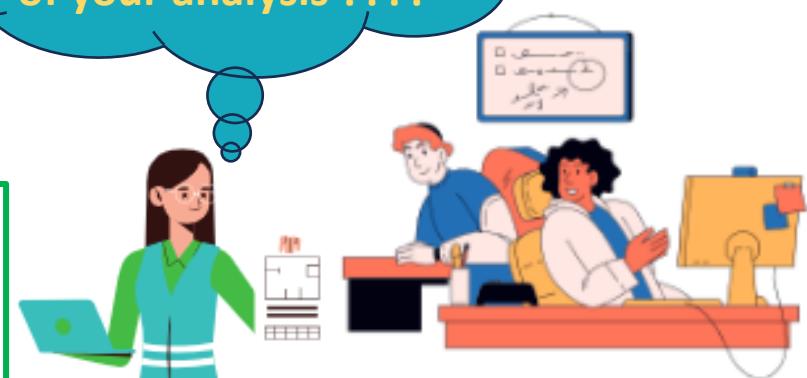
0.1. Votre mission : En tant que Assistance Data Analyste de l'entreprise

En tant qu'analyste de données de l'entreprise, vous êtes censé d'analyser les résultats de cette enquête et **proposer le graphique qui expliquent et communiquent mieux l'évaluation.**

Pour vous guider à travers cette mission complète en R, en utilisant le jeu de données attitude disponible dans le package datasets. Nous allons couvrir les étapes suivantes :

1. Chargement des données.
2. Analyse exploratoire des données (EDA)
3. Création de graphiques avancés :
 - A- Matrice de Corrélation,
 - B- Matrice de Nuages de Points,
 - C- Diagramme Radar,
 - D- Carte de Chaleur Radar,

What type of chart would you choose to present a summary of your analysis ????



Tip : To determine the best chart for presenting a summary of the analysis, we need to understand the data, go through the analysis process, and compare the results.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R



1. Chargement des données :

```
# Charger le package datasets (généralement chargé par défaut)
```

```
library(datasets)
```

```
# Charger le jeu de données attitude
```

```
data("attitude")
```

```
# Afficher les premières lignes du jeu de données
```

```
head(attitude)
```

2. Analyse exploratoire des données (EDA)

```
# Fournit des statistiques descriptives pour chaque variable.
```

```
summary(attitude)
```

```
# Afficher la structure du jeu de données
```

```
str(attitude)
```

```
> str(attitude)
'data.frame': 30 obs. of 7 variables:
 $ rating   : num  43 63 71 61 81 43 58 71 72 67 ...
 $ complaints: num  51 64 70 63 78 55 67 75 82 61 ...
 $ privileges: num  30 51 68 45 56 49 42 50 72 45 ...
 $ learning  : num  39 54 69 47 66 44 56 55 67 47 ...
 $ raises    : num  61 63 76 54 71 54 66 70 71 62 ...
 $ critical  : num  92 73 86 84 83 49 68 66 83 80 ...
 $ advance   : num  45 47 48 35 47 34 35 41 31 41 ...
```

```
> head(attitude)
   rating complaints privileges learning raises critical advance
1     43         51        30       39      61      92      45
2     63         64        51       54      63      73      47
3     71         70        68       69      76      86      48
4     61         63        45       47      54      84      35
5     81         78        56       66      71      83      47
6     43         55        49       44      54      49      34
```

```
> summary(attitude)
      rating      complaints      privileges      learning
 Min.   :40.00   Min.   :37.0   Min.   :30.00   Min.   :34.00
 1st Qu.:58.75   1st Qu.:58.5   1st Qu.:45.00   1st Qu.:47.00
 Median :65.50   Median :65.0   Median :51.50   Median :56.50
 Mean   :64.63   Mean   :66.6   Mean   :53.13   Mean   :56.37
 3rd Qu.:71.75   3rd Qu.:77.0   3rd Qu.:62.50   3rd Qu.:66.75
 Max.   :85.00   Max.   :90.0   Max.   :83.00   Max.   :75.00
      raises      critical      advance
 Min.   :43.00   Min.   :49.00   Min.   :25.00
 1st Qu.:58.25   1st Qu.:69.25   1st Qu.:35.00
 Median :63.50   Median :77.50   Median :41.00
 Mean   :64.63   Mean   :74.77   Mean   :42.93
 3rd Qu.:71.00   3rd Qu.:80.00   3rd Qu.:47.75
 Max.   :88.00   Max.   :92.00   Max.   :72.00
```

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

2. Analyse exploratoire des données (EDA) : suit

Histogrammes des variables

```
library(ggplot2)
```



Charge ggplot2 (package pour la visualisation).

Créer une fonction pour tracer les histogrammes

```
plot_histogram <- function(data) {
```

```
  for (col in names(data)) {
```

```
    p <- ggplot(data, aes_string(col)) +
```

```
      geom_histogram(binwidth = 5, fill = "blue", color = "black", alpha = 0.7) +
```

```
      ggtitle(paste("Histogramme de la variable ", col)) +
```

```
      theme_minimal()
```

```
    print(p)
```

```
}
```

```
}
```

Tracer les histogrammes

```
plot_histogram(attitude)
```



Fonction personnalisée pour tracer les histogrammes de chaque variable dans le jeu de données attitude.

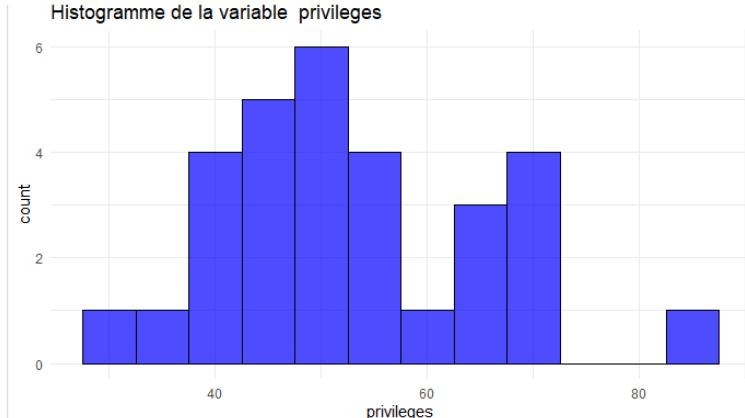
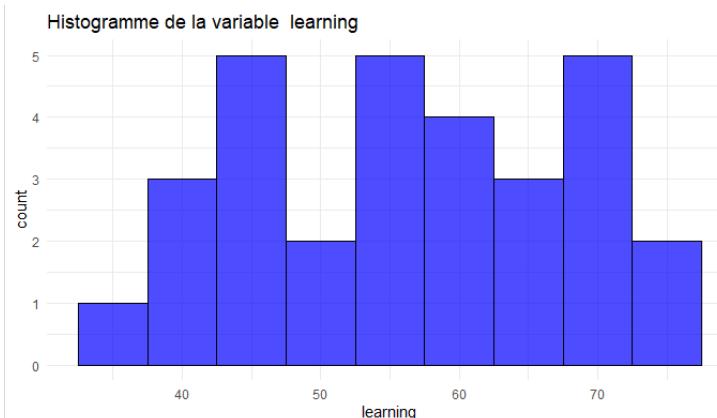
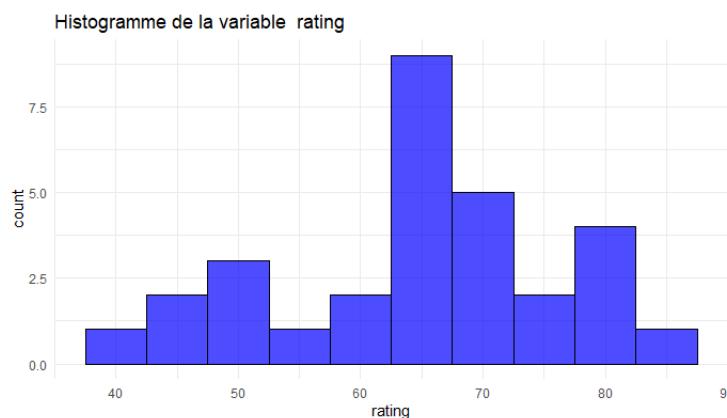
02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

2. Analyse exploratoire des données (EDA) : suiv

Histogrammes des variables : **Output**



Exemple d'interprétation de l'histogramme de la variable rating :

Distribution : L'histogramme de la variable rating montre une distribution variée des notes attribuées par les employés. On observe un pic notable autour de la note 65-70, ce qui suggère que la majorité des évaluations se situent dans cette plage.

Fréquence : Le plus grand nombre d'employés a attribué des notes comprises entre 65 et 70. Il y a également des fréquences relativement élevées pour les notes entre 75 et 80.

Variation : Il existe une certaine variation dans les évaluations, avec des notes allant de 40 à 90. Cela indique des opinions diverses parmi les employés concernant la performance de l'entreprise.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

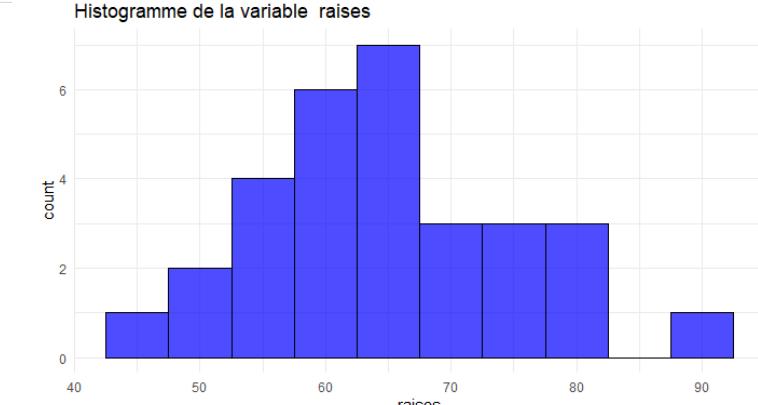
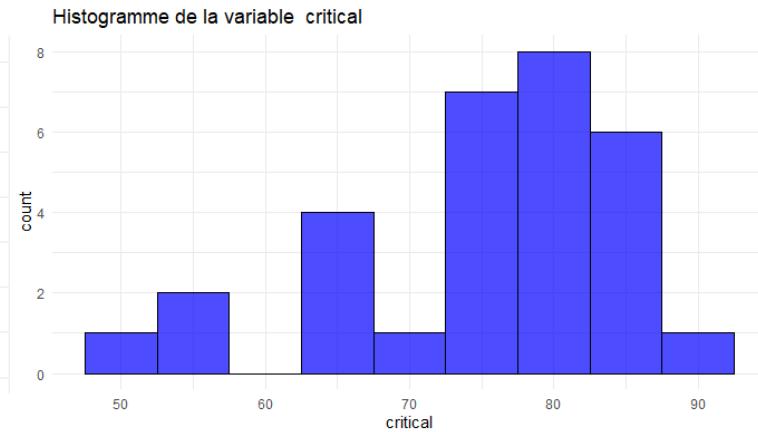
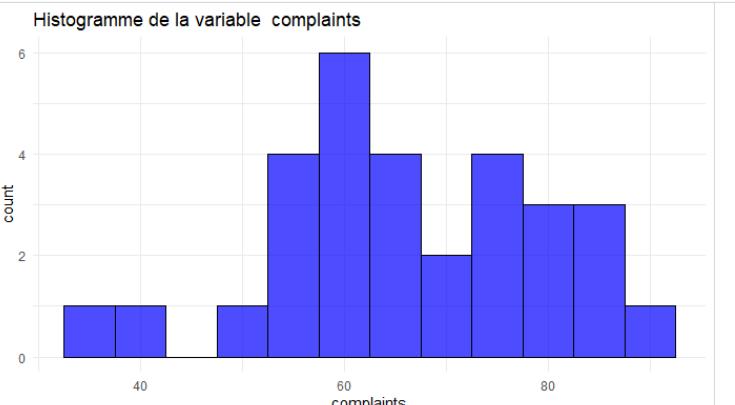
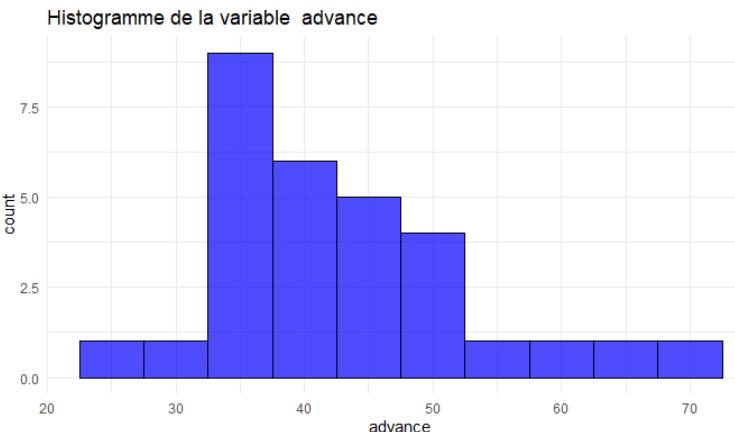
Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

2. Analyse exploratoire des données (EDA) : suivre

Histogrammes des variables : Output

Ces graphiques offrent une vue d'ensemble des perceptions des employés sur différentes dimensions de la performance de l'entreprise, permettant une analyse approfondie pour informer la prise de décision et les stratégies d'amélioration.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

A- Matrice de Corrélation : Création

```
# Installer le package nécessaire
```

```
install.packages("corrplot")
```

```
# Charger le package nécessaire
```

```
library(corrplot)
```

```
# Calculer la matrice de corrélation
```

```
cor_matrix <- cor(attitude)
```

```
# Tracer la matrice de corrélation
```

```
corrplot(cor_matrix, method = "color", addCoef.col = "black", tl.col = "black", tl.srt = 45)
```

B- Matrice de Nuages de Points : Création

```
# Tracer la matrice de nuages de points
```

```
pairs(attitude, main = "Matrice de Nuages de Points des Variables de l'Enquête")
```



Note : Avant de pouvoir utiliser un package, il doit être installé sur votre système.
« l'équivalent de pip install en Python »



Note : La matrice de corrélation montre les coefficients de corrélation entre chaque paire de variables, ce qui aide à comprendre les relations entre les variables.

Cette ligne trace la matrice de corrélation en utilisant le package corrplot.

cor_matrix : La matrice de corrélation calculée précédemment.

method = "color" : Utilise des couleurs pour représenter les coefficients de corrélation.

addCoef.col = "black" : Ajoute les coefficients de corrélation en noir sur les cellules.

tl.col = "black" : Définit la couleur des étiquettes de texte en noir.

tl.srt = 45 : Fait pivoter les étiquettes de texte à 45 degrés.

B- Matrice de Nuages de Points : Création

```
# Tracer la matrice de nuages de points
```

```
pairs(attitude, main = "Matrice de Nuages de Points des Variables de l'Enquête")
```

Crée une matrice de nuages de points pour les variables du jeu de données attitude.

attitude : Le jeu de données utilisé.

main = .. : Ajoute un titre au graphique.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

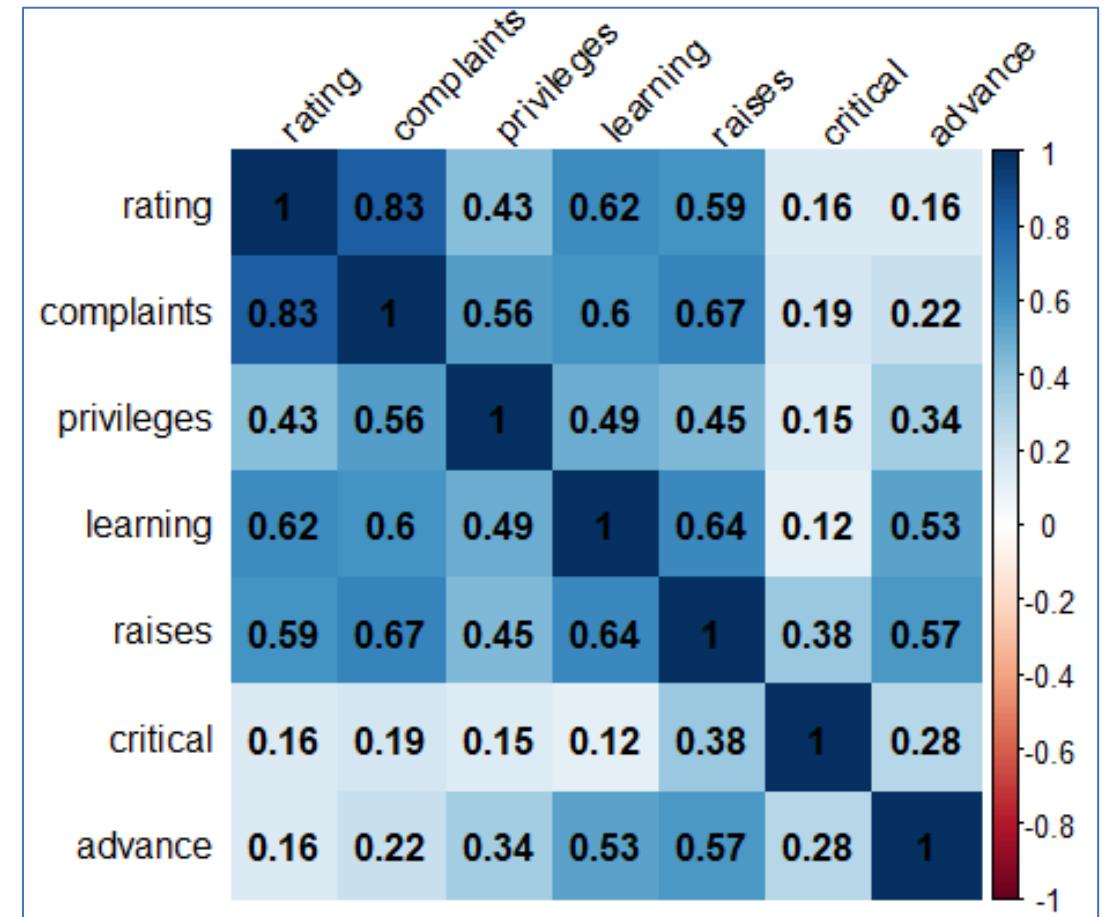
Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

- A- Matrice de Corrélation : **Output**

Visualiser la matrice de corrélation aide à identifier rapidement les relations positives ou négatives fortes entre les variables.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

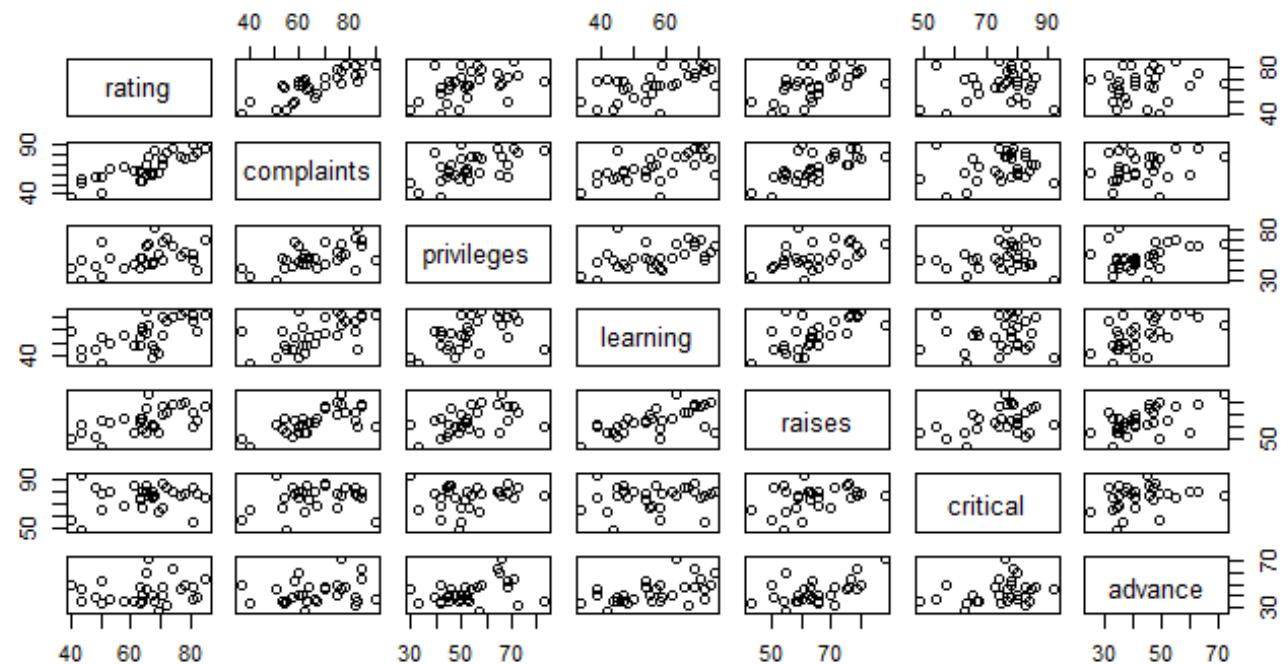
Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

B- Matrice de Nuages de Points : Output

Matrice de Nuages de Points des Variables de l'Enquête



La matrice de nuages de points crée des graphiques de dispersion pour chaque **paire de variables** dans le jeu de données, permettant de visualiser les **relations bivariées** entre les variables. Elle aide à identifier les **corrélations** et les **tendances** entre les différentes mesures de l'enquête.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

C- Diagramme Radar, (Radar chart) : [Création](#)

```
# Installer et charger le package nécessaire
```

```
install.packages("fmsb")
```

```
library(fmsb)
```

```
# Normaliser les données dans la plage [0, 100]
```

```
normalized_attitude <- as.data.frame(lapply(attitude, function(x) (x - min(x)) / (max(x) - min(x)) * 100))
```

```
# Ajouter des valeurs maximum et minimum pour le diagramme radar
```

```
normalized_attitude <- rbind(rep(100, 7), rep(0, 7), normalized_attitude)
```

```
# Tracer le diagramme radar
```

```
radarchart(normalized_attitude, axistype = 1,  
           pcol = rainbow(3),  
           pfcol = alpha(rainbow(3), 0.5),  
           plwd = 2,  
           plty = 1)
```

Un diagramme radar aide à visualiser les données multivariées dans une disposition radiale. Nous utiliserons le package fmsb.

Note : Vous devrez redémarrer votre environnement si vos packages de données nécessitent une mise à jour.

L'argument `axistype` détermine le style des axes et des étiquettes du graphique radar. La fonction `rainbow(30)` génère une palette de 30 couleurs, et `alpha(rainbow(30), 0.5)` ajoute de la transparence à ces couleurs pour les remplir (`pfcol`)

Découvrez-en plus sur les autres paramètres dans la diapositive suivante

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

C- Diagramme Radar, (Radar chart) : Paramètres de la fonction radarchart

normalized_attitude : Les données normalisées entre 0 et 100.
axistype = 1 : Affiche des lignes d'axes avec des étiquettes de valeurs aux points terminaux.
pcol = rainbow(30) : Utilise une palette de 30 couleurs pour les lignes.
pfcol = alpha(rainbow(30), 0.5) : Utilise une palette de 30 couleurs avec transparence pour le remplissage.
plwd = 2 : Définit la largeur des lignes à 2 unités.
plty = 1 : Utilise des lignes continues.

Voici les différentes valeurs possibles pour **axistype** et leur signification :

- axistype** = 0 : Pas de lignes d'axes ni d'étiquettes de valeurs.
- axistype** = 1 : Lignes d'axes avec des étiquettes de valeurs aux points terminaux.
- axistype** = 2 : Lignes d'axes avec des étiquettes de valeurs sur les cercles concentriques.
- axistype** = 3 : Seulement des lignes d'axes sans étiquettes de valeurs.
- axistype** = 4 : Seulement des étiquettes de valeurs sans lignes d'axes.

Les types de ligne sont codés par des entiers. Par exemple:

- 1 signifie une ligne continue,
- 2 signifie une ligne en tirets,
- 3 signifie une ligne en points,
- 4 signifie une ligne en tirets et points, etc.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

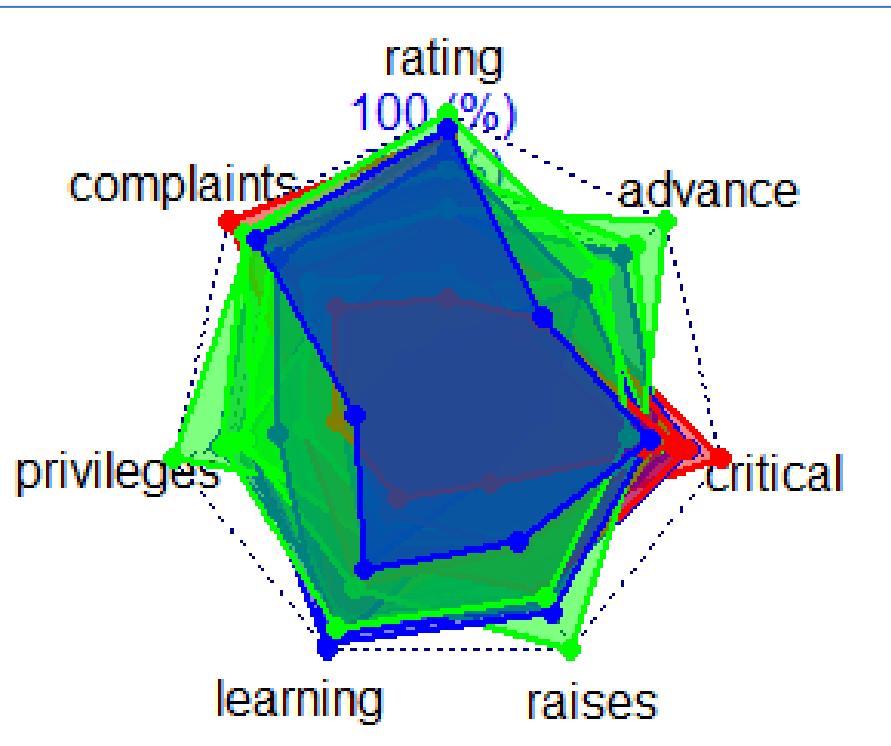
3. Création de graphiques avancés :

C- Diagramme Radar, (Radar chart) : Output



On peut clairement observer que la carte radar n'est **pas adaptée** à la présentation d'une analyse globale de tous les employés. Elle peut cependant être utile si l'on souhaite comparer certaines lignes de données, par exemple, comparer deux employés.

Toutefois, pour 30 éléments, le graphique devient trop **visuellement chargé**, ce qui en fait un choix peu optimal pour résumer les avis des employés.



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

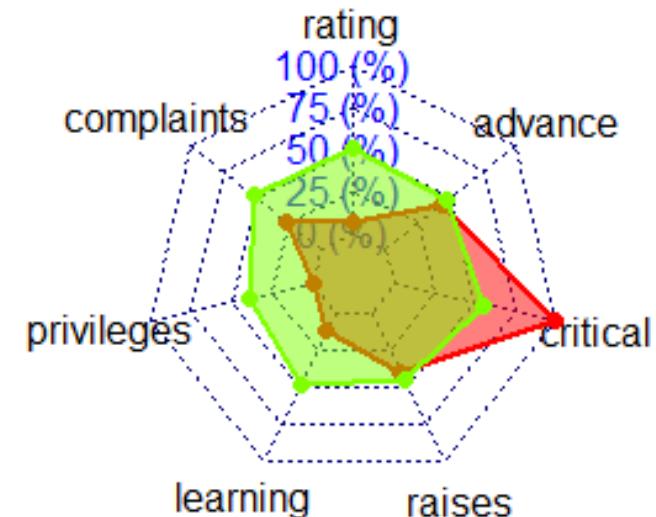
3. Création de graphiques avancés :

C- Diagramme Radar, (Radar chart) : Amélioration

Pour éditer le code précédent afin de montrer uniquement la comparaison de deux employés, nous avons fait les ajustements nécessaires.

Le graphique montre maintenant mieux les avis de deux employés. Toutefois, il est important de noter que, dans notre analyse, nous sommes chargés de satisfaire le besoin d'afficher un graphique qui résume la totalité des avis des employés. Ce type de graphique radar n'est donc pas approprié pour cela. En effet, lorsqu'on essaie de représenter les données de 30 employés, le graphique devient trop encombré et difficile à lire.

Comparaison des Avis de Deux Employés



Nous demandons aux étudiants de modifier le code pour comparer seulement deux employés et constatons que même si le graphique est désormais plus lisible, il n'est toujours pas adéquat pour résumer les opinions de tous les employés.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

C- Diagramme Radar, (Radar chart) : Solution Proposé

Sélectionner deux employés pour la comparaison (par exemple, les 1er et 2èmes lignes du jeu de données normalisées)

```
employees_to_compare <- normalized_attitude[c(1, 2, 3, 4), ]
```

3

Tracer le diagramme radar

```
radarchart(employees_to_compare, axistype = 1,  
          pcol = rainbow(4),  
          pfcol = alpha(rainbow(4), 0.5),  
          plwd = 2,  
          plty = 1)
```

Ajouter un titre au graphique

```
title("Comparaison des Avis de Deux Employés")
```

Ajouter des valeurs maximum et minimum pour le diagramme radar

```
normalized_attitude <- rbind(rep(100, 7), rep(0, 7) , normalized_attitude)
```

1

> normalized_attitude

| | rating | complaints | privileges | learning | raises | critical | advance |
|---|------------|------------|------------|------------|------------|------------|------------|
| 1 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 |
| 2 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 3 | 6.666667 | 26.415094 | 0.000000 | 12.19512 | 40.00000 | 100.00000 | 42.55319 |
| 4 | 51.111111 | 50.943396 | 39.622642 | 48.78049 | 44.44444 | 55.81395 | 46.80851 |
| 5 | 68.888889 | 62.264151 | 71.698113 | 85.36585 | 73.33333 | 86.04651 | 48.93617 |
| 6 | 46.666667 | 49.056604 | 28.301887 | 31.70732 | 24.44444 | 81.39535 | 21.27660 |

2

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

D- Carte de Chaleur Radar (Radar HeatMap) : Préparation

```
# Charger les packages nécessaires
```

```
install.packages("ggplot2")
```

```
install.packages("reshape2")
```

```
library(ggplot2)
```

```
library(reshape2)
```

```
# Charger les données
```

```
data("attitude")
```

```
# Normaliser les données dans la plage [0, 100]
```

```
normalized_attitude <- as.data.frame(lapply(attitude, function(x) (x - min(x)) / (max(x) - min(x)) * 100))
```

```
normalized_attitude$Category <- rownames(normalized_attitude)
```

Installe et charge les packages ggplot2 et reshape2 nécessaires pour créer la carte de chaleur.

Normalise les données pour qu'elles soient dans la plage [0, 100] et ajoute une colonne **Category** pour les identifiants des lignes.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

D- Carte de Chaleur Radar (Radar HeatMap) : **Création**

```
# Fondre les données
```

```
melted_data <- melt(normalized_attitude, id.vars = "Category")
```

```
# Créer la carte de chaleur radar avec ggplot2
```

```
ggplot(melted_data, aes(x = variable, y = Category, fill = value)) +  
  geom_tile() + scale_fill_gradient(low = "white", high = "blue") +  
  coord_polar(theta = "x") +  
  theme_minimal() +  
  theme(axis.text.x = element_text(size = 6, angle = 45, hjust = 1)) +  
  labs(title = "Radar Heat Map des Variables de l'Enquête", x = "", y = "")
```

Convertit le Data Frame en un format long, ce qui est nécessaire pour la fonction **ggplot**

Utilise **ggplot** pour créer une carte de chaleur radar. La fonction **geom_tile** crée les tuiles de la carte de chaleur, **scale_fill_gradient** définit les couleurs, **coord_polar** applique une transformation polaire pour créer l'effet radar, et **theme_minimal** simplifie le thème du graphique.

L'ajustement des étiquettes (**axis.text.x**) réduit leur taille et ajuste leur angle pour éviter qu'elles se chevauchent.

02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

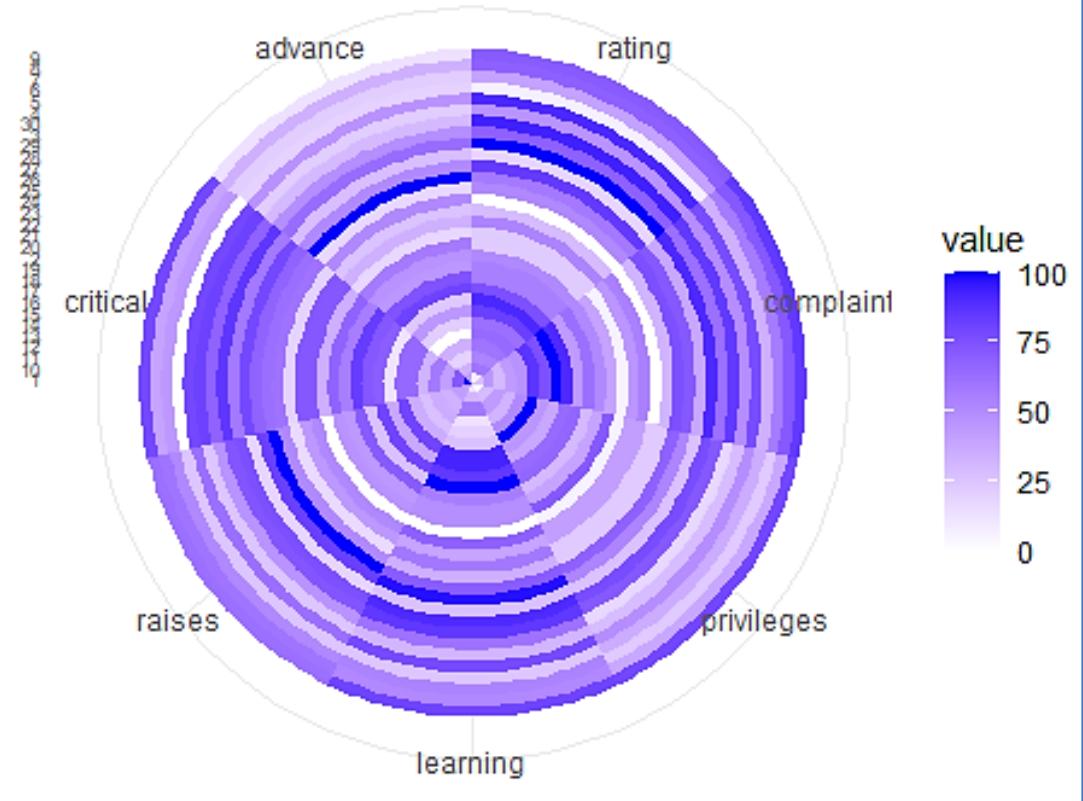
D- Carte de Chaleur Radar (Radar HeatMap) : Output

En suivant les étapes précédentes, vous devriez obtenir une carte de chaleur radar claire et lisible pour analyser les variables de l'enquête.

En observant la carte de chaleur radar des variables de l'enquête, pensez-vous que ce graphique résume efficacement la totalité des valeurs de l'enquête ? **Pourquoi ou pourquoi pas ?**



Radar Heat Map des Variables de l'Enquête



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés :

D- Carte de Chaleur Radar (Radar HeatMap) : Interprétation

Micro analysis : Pas recommander (element par element)

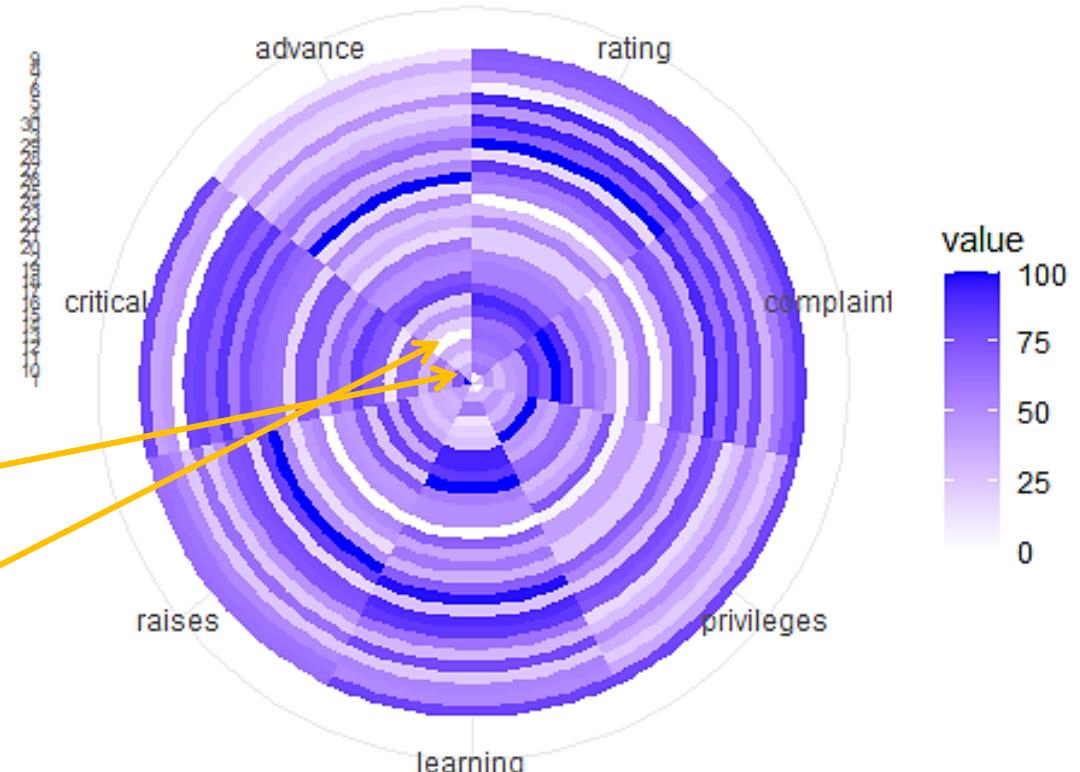
On peut identifier les évaluations de chaque employé, les premières lignes de données disposées au centre du radar, tandis que les dernières lignes de données représentent les cercles extérieurs. Les couleurs plus claire ou blanche présentent une faible valeur, tandis que les nuances plus foncées de bleu indiquent une valeur élevée pour toutes les variables de l'enquête.

```
> normalized_attitude
      rating complaints privileges learning   raises critical advance
1 100.000000 100.000000 100.000000 100.000000 100.000000 100.000000 100.000000
2 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
3 6.666667 26.415094 0.000000 12.19512 40.00000 100.00000 42.55319
4 51.111111 50.943396 39.622642 48.78049 44.44444 55.81395 46.80851
5 68.888889 62.264151 71.698113 85.36585 73.33333 86.04651 48.93617
6 46.666667 49.056604 28.301887 31.70732 24.44444 81.39535 21.27660
```

Macro analysis : Recommender (Vue globale)

Critique : Les nuances plus foncées de bleu indiquent que la majorité des employés ont des critiques élevées envers l'entreprise.

Radar Heat Map des Variables de l'Enquête



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

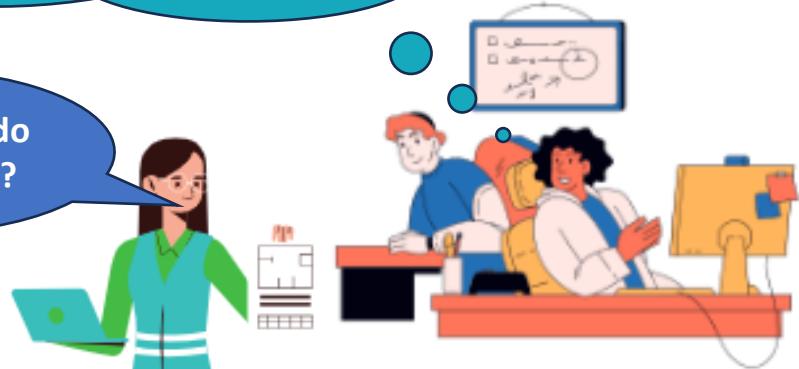
3. Création de graphiques avancés : Choisir le graphique approprié

D- Carte de Chaleur Radar (Radar HeatMap) : Décision

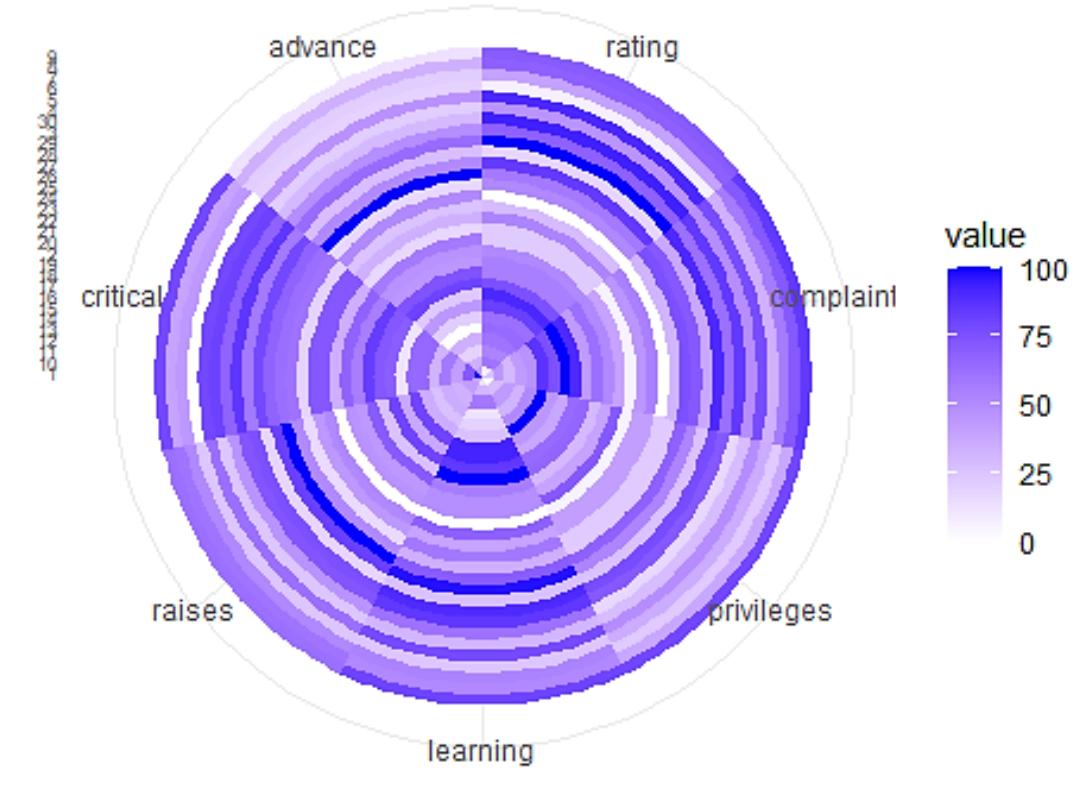
Ce graphique de la carte de chaleur radar permet d'identifier facilement les tendances générales des évaluations des employés.

Ce graphique de la carte de chaleur est approprié pour présenter les besoins d'analyse de l'entreprise

What trends do you observe ?



Radar Heat Map des Variables de l'Enquête



02 – CONFIGURER LES GRAPHIQUES POUR UNE ANALYSE PROFONDE

Sélection des graphiques avancés avec Python/R

Sélection des graphiques avancés avec R

3. Création de graphiques avancés : la prise de décision

D- Carte de Chaleur Radar (Radar HeatMap) : Décision

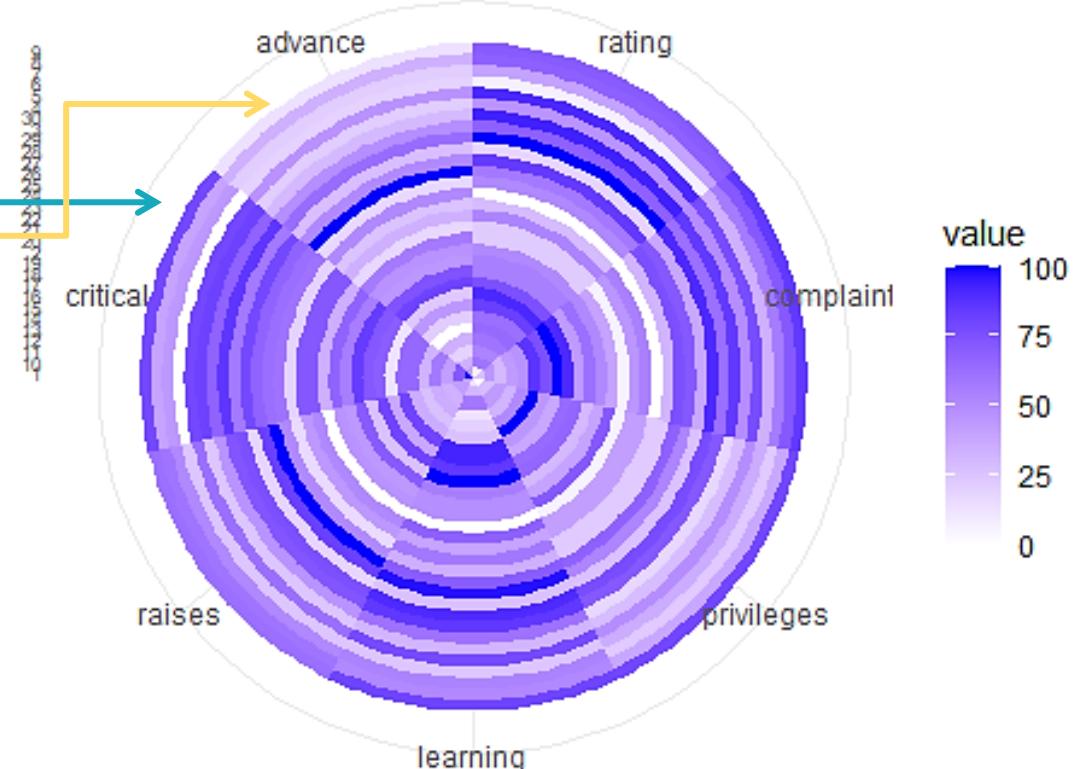
ce graphique montre :

Des critiques élevées (zones plus foncées) de la part des employés, ce qui peut indiquer des mécontentements ou des insatisfactions spécifiques. Un faible taux de satisfaction concernant les priviléges et les opportunités d'avancement (zones plus claires), ce qui suggère que les employés ne se sentent pas suffisamment soutenus dans leur développement professionnel et de carier.

Ces observations peuvent aider à orienter les efforts de l'entreprise pour améliorer la satisfaction des employés dans ces domaines spécifiques.

L'analyse décisionnelle permet des ajustements en temps réel et des changements **stratégiques** à long terme qui suppriment les inefficacités, s'adaptent aux changements du marché, corrigent les problèmes d'approvisionnement et résolvent les problèmes client.

Radar Heat Map des Variables de l'Enquête





PARTIE 3

MAITRISER LES BASES DE DONNÉES DÉCISIONNELLES

Dans ce module, vous allez :

- Introduire les bases de données décisionnelles
- Explorer la modélisation dimensionnelle avancée
- Appliquer le processus ETL dans le contexte décisionnel



 20 heures



CHAPITRE 1

INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Ce que vous allez apprendre dans ce chapitre :

- Définir les bases de données décisionnelles.
- Comparer avec les bases de données opérationnelles.
- Analyser l'architecture d'une base de données décisionnelle.



X heures



CHAPITRE 1

INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

1. **Définitions et objectifs**
2. Différences avec les bases de données opérationnelles
3. Architecture d'une base de données décisionnelle

Introduction :

Problématique :

- La gestion et l'analyse des données massives (big data) pour extraire des informations utiles à la prise de décision est complexe. C'est pourquoi de nombreuses entreprises en quête de croissance utilisent de plus en plus les outils de l'informatique décisionnelle. La mise en place d'un Système d'Information Décisionnel (SID) représente un défi majeur pour ces entreprises.

Rappel du Cours de Première Année : Modélisation de données et ETL

En première année, nous avons exploré les fondamentaux de l'informatique décisionnelle, y compris ses objectifs principaux :

- **Objectif opérationnel** : Traitement des données opérationnelles Le sauvegarde/la conservation opérationnel se réfère à l'**OLTP** (On-line Transactional Processing) qui correspond au traitement transactionnel en ligne.
- **Objectif décisionnel** : Prise de décision analytique Le traitement analytique se réfère à l'**OLAP** (On-line Analytical Processing) qui correspond à l'analyse analytique en ligne.

Définitions : Data Warehouse

- Les bases de données décisionnelles, également appelées entrepôts de données ou data Warehouse, sont des systèmes spécialisés conçus pour soutenir les processus de prise de décision.
- Les bases de données décisionnelles se concentrent sur l'analyse des données historiques et actuelles pour fournir des informations exploitables aux décideurs.



Définition : Un entrepôt de données (DW) est une base de données relationnelle destinée à l'analyse plutôt qu'au traitement des transactions. Il comprend des données historiques dérivées de données de transaction provenant de sources uniques ou variées, aidant les entreprises à prendre des décisions stratégiques basées sur des analyses solides.



Data Warehouse

Objectifs : Data Warehouse

Les bases de données décisionnelles ou les data Warehouse dans un environnement d'entreprise vise à atteindre plusieurs objectifs clés :

- **Consolidation des Données** : Intégration et unification des données de diverses sources.
- **Historisation** : Stockage des données historiques pour analyser les tendances.
- **Optimisation des Performances** : Amélioration des requêtes analytiques.
- **Support à la Prise de Décision** : Fourniture de rapports et visualisations pour des décisions informées.
- **Qualité des Données** : Assurer l'exactitude et la fiabilité des données.





CHAPITRE 1

INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

1. Définitions et objectifs
2. **Différences avec les bases de données opérationnelles**
3. Architecture d'une base de données décisionnelle

01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Différences avec les bases de données opérationnelles

Les bases de données opérationnelles

- La base de données opérationnelle est la **source d'information** de l'entrepôt de données.
- Elle contient des informations détaillées utilisées pour gérer les opérations quotidiennes de l'entreprise. Les données changent fréquemment au fur et à mesure des mises à jour et reflètent la valeur actuelle des dernières transactions.
- Les bases de données décisionnelles (data Warehouse) sont distinctes des bases de données opérationnelles (transactionnelles), qui sont principalement utilisées pour les opérations quotidiennes de traitement des transactions.
- Operational Database Management Systems also called as OLTP (Online Transactions Processing Databases), are used to manage dynamic data in real-time.



01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Différences avec les bases de données opérationnelles



Décisionnelles Vs Opérationnelles Data Bases

- Data Warehouse et la base de données OLTP sont toutes deux des bases de données relationnelles. Cependant, les objectifs de ces deux bases de données sont différents.

| Base de Données Opérationnelle | Entrepôt de Données | Résumé |
|---|--|---|
| Conçue pour le traitement des transactions à volume élevé.(OLTP) | Conçue pour le traitement analytique à volume élevé (OLAP). | Transactions vs Analyse |
| Se concentre généralement sur les données actuelles. | Se concentre généralement sur les données historiques. | Données actuelles vs historiques |
| Les données sont mises à jour régulièrement selon les besoins. | Données non volatiles, nouvelles données ajoutées régulièrement et rarement modifiées. | Mises à jour régulières vs modification rares |
| Conçue pour les opérations et processus en temps réel. | Conçue pour l'analyse par sujet, catégorie et attribut. | Temps réel vs Analyse |
| Optimisée pour des transactions simples, ajout ou récupération d'une ligne à la fois par table. | Optimisée pour des chargements étendus et des requêtes complexes et imprévisibles accédant à de nombreuses lignes par table. | Transactions simples vs Requêtes complexes |
| Optimisée pour la validation des informations entrantes pendant les transactions, utilise des tables de validation. | Chargée d'informations cohérentes et valides, ne nécessite pas de validation en temps réel. | Validation vs Consistance |

01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Différences avec les bases de données opérationnelles



Décisionnelles Vs Opérationnelles Data Bases : suit

- Les bases de données opérationnelles sont conçues pour le traitement rapide et en temps réel des transactions courantes, tandis que les entrepôts de données sont optimisés pour l'analyse complexe et l'exploitation des données historiques.
- Les bases de données opérationnelles mettent l'accent sur les mises à jour fréquentes et la gestion des processus en temps réel, alors que les entrepôts de données se concentrent sur l'intégration de grandes quantités de données pour des analyses approfondies et la prise de décision stratégique.

| Base de Données Opérationnelle | Entrepôt de Données | Résumé |
|---|---|--|
| Supporte des milliers de clients simultanés. | Supporte peu de clients simultanés par rapport à OLTP. | Nombre de clients : élevé vs faible |
| Orientée processus. | Orientée sujet. | Processus vs Sujet |
| Optimisée pour des insertions et mises à jour rapides de petits volumes de données. | Optimisée pour des récupérations rapides de volumes de données relativement élevés. | Insertions/mises à jour vs Récupérations |
| Données entrantes. | Données sortantes. | Entrée vs Sortie |
| Moins de données accédées. | Grand nombre de données accédées. | Accès faible vs élevé |
| Bases de données relationnelles créées pour le traitement transactionnel en ligne (OLTP). | Entrepôt de données conçu pour le traitement analytique en ligne (OLAP). | OLTP vs OLAP |



CHAPITRE 1

INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

1. Définitions et objectifs
2. Différences avec les bases de données opérationnelles
3. **Architecture d'une base de données décisionnelle**

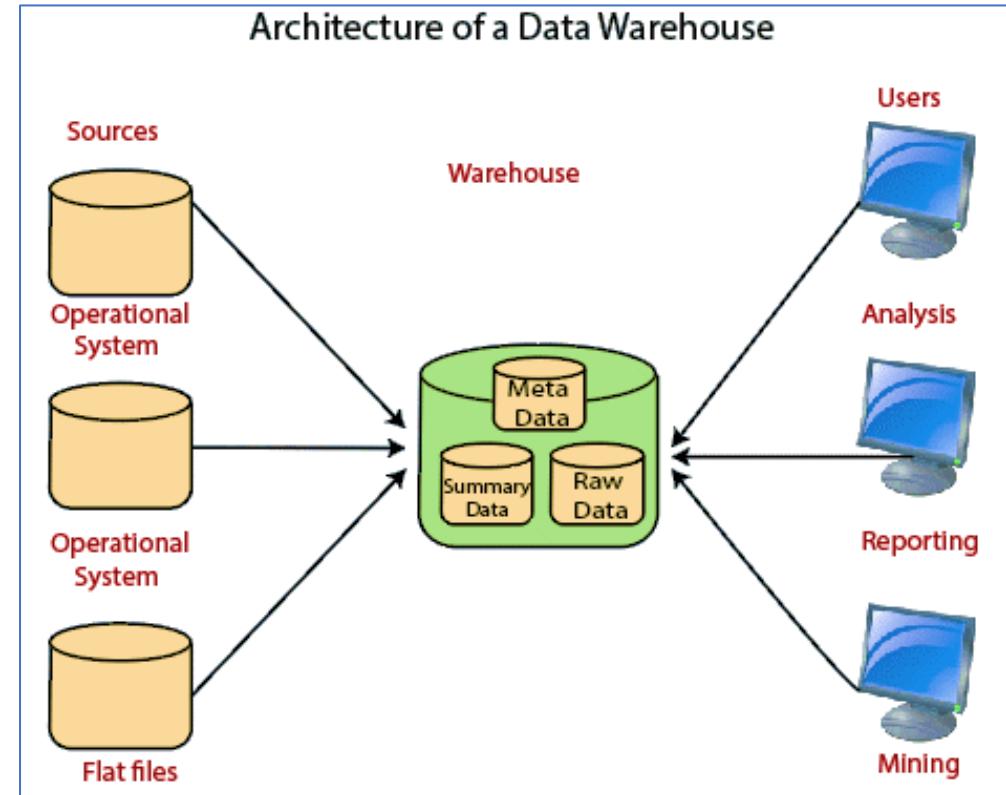
01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

Rappel sur Architecture de Data Warehouse : Basique

1. Composants de l'architecture :

- **Système Opérationnel** : Utilisé pour traiter les transactions quotidiennes.
- **Fichiers plats** : Système où les données transactionnelles sont stockées, chaque fichier ayant un nom unique.
- **Meta Données** : Ensemble de données fournissant des informations sur d'autres données (ex. auteur, date de création, taille du fichier).
- **Données Résumées** : Données agrégées prédéfinies pour accélérer les performances des requêtes.
- **Outils d'Accès Utilisateur Final** : Utilisés par les gestionnaires pour la prise de décision stratégique (ex. outils de reporting, développement d'applications, systèmes d'information exécutifs, OLAP, data mining).



01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

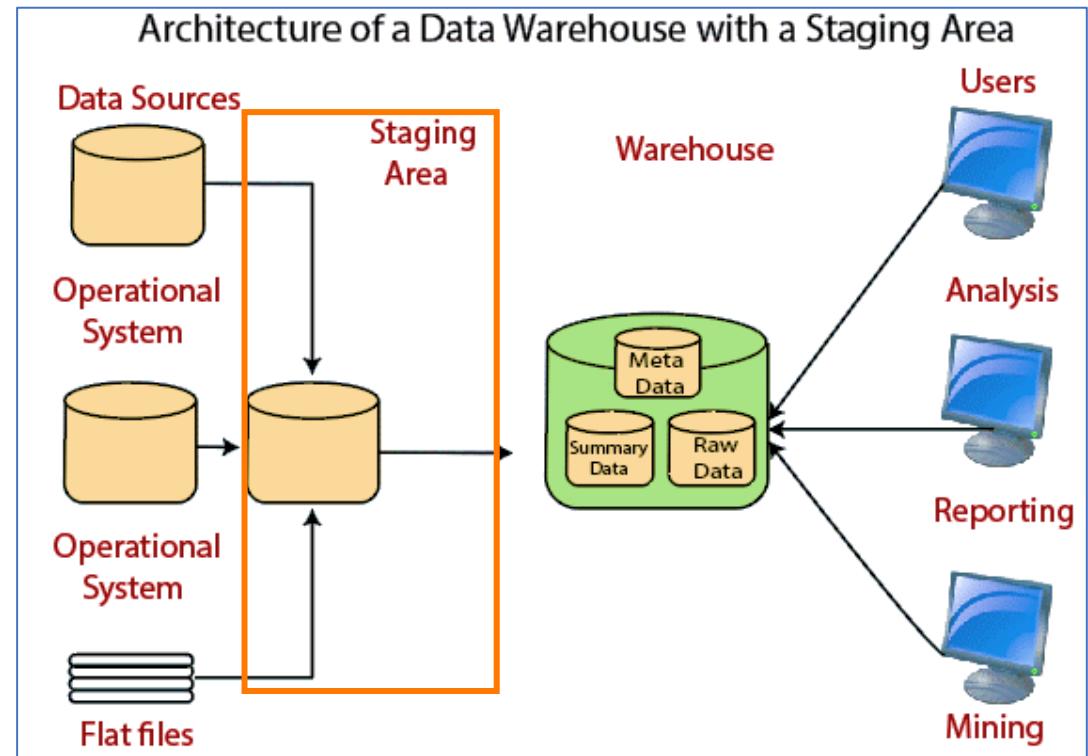
Rappel sur Architecture de Data Warehouse : Avec Zone de Transit (Staging Area)

1. Composants de l'architecture :

- **Zone de Transit** : Espace temporaire où les données opérationnelles sont nettoyées et consolidées avant d'entrer dans l'entrepôt de données.

Utilité :

- **Simplifie le Nettoyage et la Consolidation** : Particulièrement utile pour les entrepôts de données d'entreprise qui consolident des données provenant de **multiples systèmes sources**.



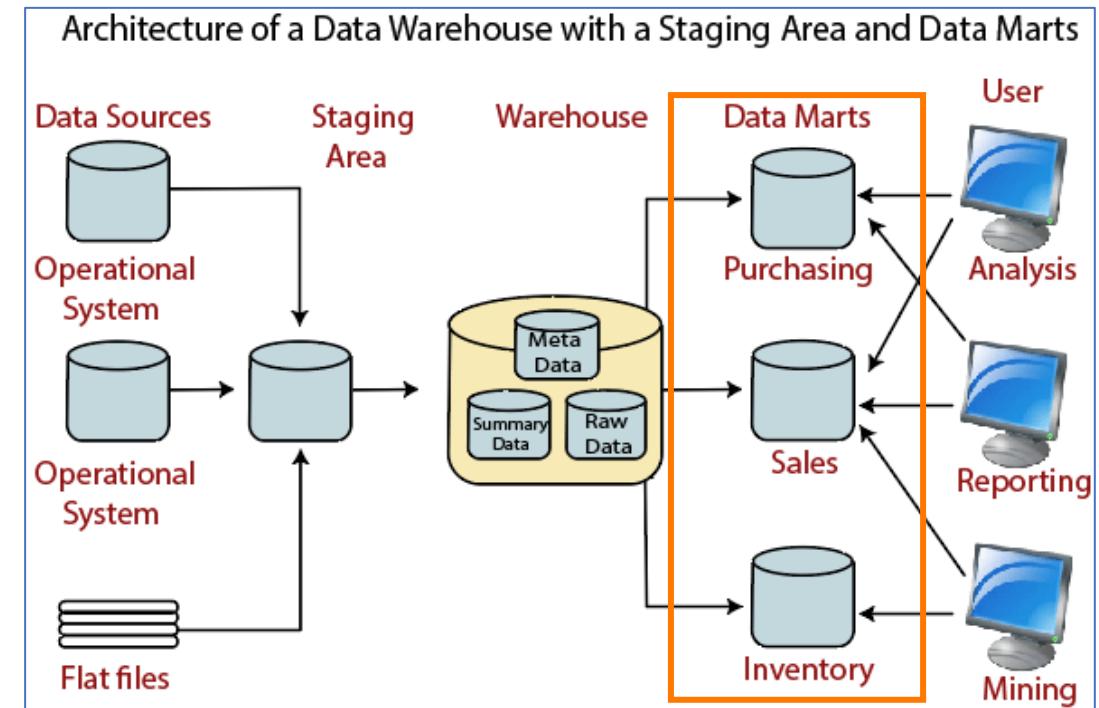
01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

Rappel sur Architecture de Data Warehouse : Avec Zone de Transit et Data Marts

1. Composants de l'architecture :

- **Data Marts** : Segments de l'entrepôt de données fournissant des informations pour le reporting et l'analyse d'une section spécifique (ex. ventes, paie, production).
- **Utilité** :
- **Personnalisation pour Divers Groupes** : Permet de **personnaliser l'architecture** pour répondre aux besoins de différents départements de l'organisation.



01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

Rappel sur Architecture de Data Warehouse : Avec Zone de Transit et Data Marts

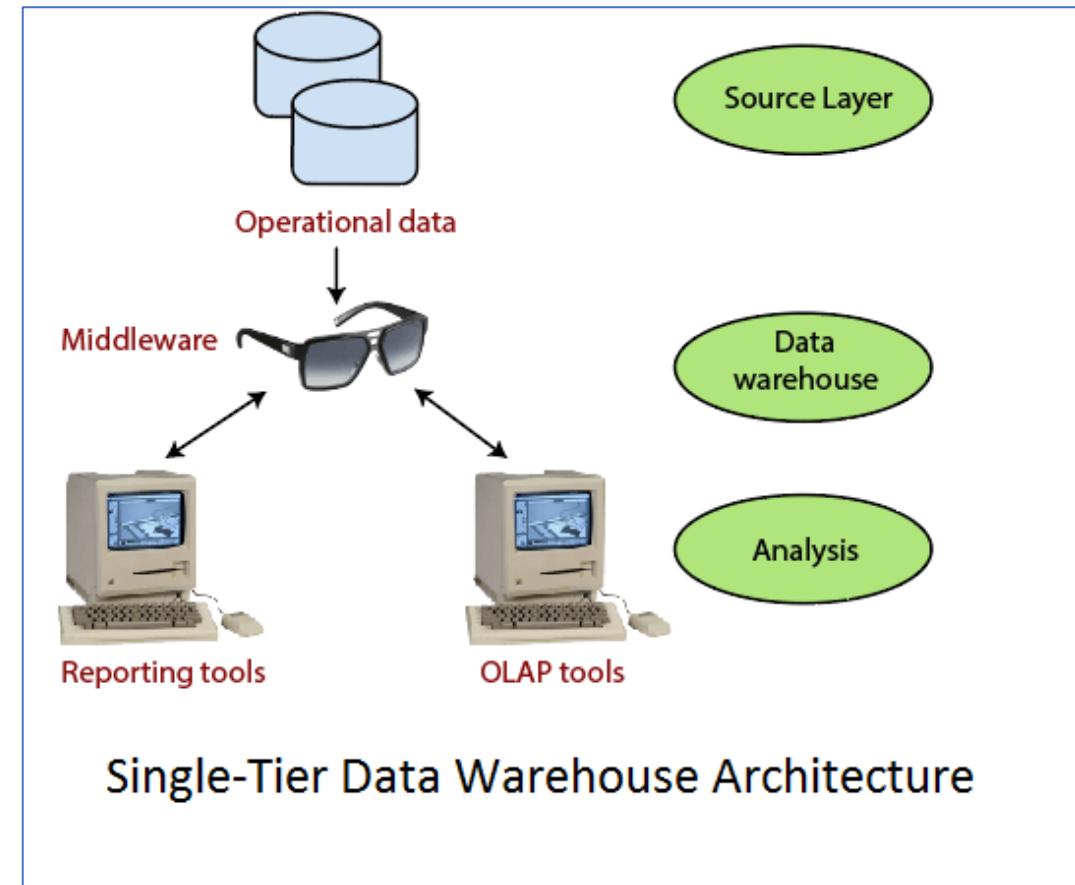
2. Types des architectures : Flux de Données

A. Architecture Monolithique (Single-Tier)

- Description :** Minimise les données stockées en supprimant les redondances. La seule couche disponible est la couche source. Les entrepôts de données **sont virtuels**, implémentés comme **une vue multidimensionnelle** des données opérationnelles.
- Inconvénients :** Ne répond pas aux exigences de séparation entre analytique et transactionnel, impactant les charges transactionnelles.



Déconseillé



01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

Rappel sur Architecture de Data Warehouse : Avec Zone de Transit et Data Marts

2. Types des architectures : Flux de Données

B. Architecture à Deux Niveaux (Two-Tier)

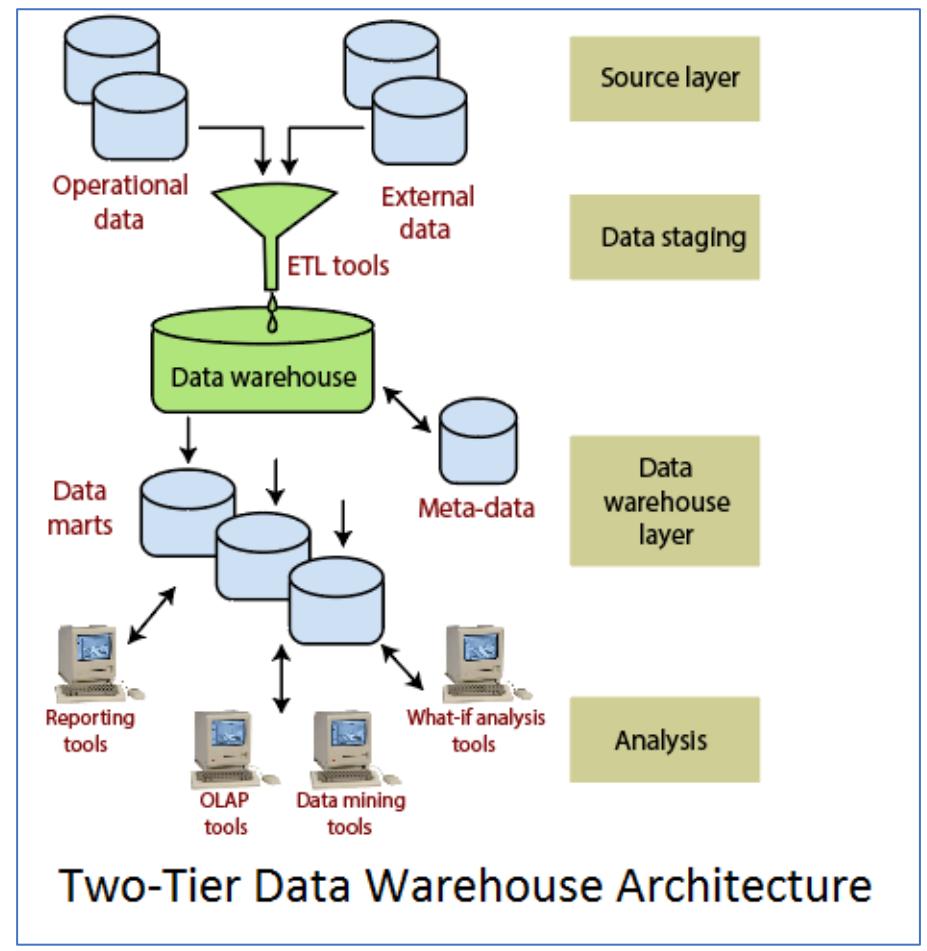
- Description :** Sépare les sources de données physiques et l'entrepôt de données en quatre étapes de flux de données.
- Avantages :** Séparation claire des responsabilités, optimisation des performances analytiques
- Étapes :** Source, mise en transit (ETL), entrepôt de données, et analyse.

Couche Source : Utilise des sources de données hétérogènes (bases de données relationnelles, systèmes d'information externes).

Mise en Transit (ETL) : Extraction, transformation, nettoyage, validation, filtrage, et chargement des données source dans l'entrepôt.

Couche Entrepôt de Données : Stocke les informations dans un dépôt centralisé, utilisé aussi pour créer des data marts pour différents départements.

Analyse : Accès aux données intégrées pour générer des rapports, analyser dynamiquement, et simuler des scénarios commerciaux.



01 – INTRODUIRE LES BASES DE DONNÉES DÉCISIONNELLES

Architecture d'une base de données décisionnelle

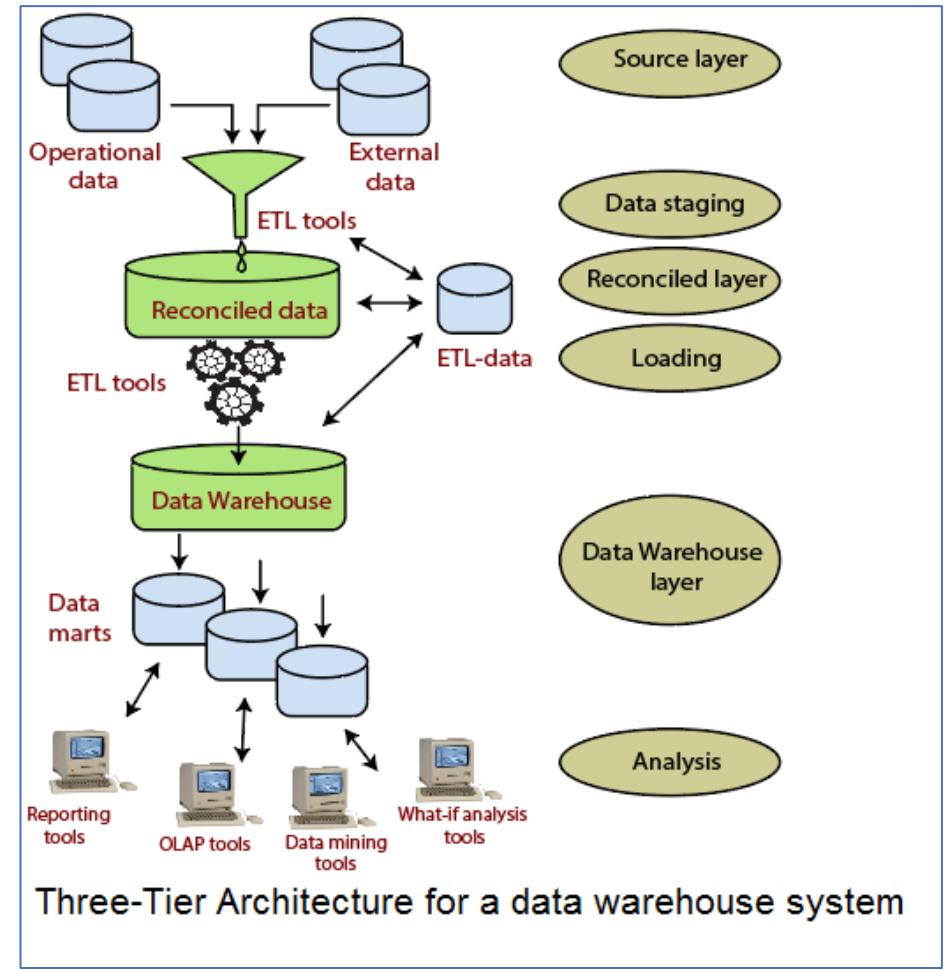
Rappel sur Architecture de Data Warehouse : Avec Zone de Transit et Data Marts

2. Types des architectures : Flux de Données

C. Architecture à Trois Niveaux (Three-Tier)

- **Description :** Comprend le niveau source, le niveau réconcilié, et le niveau entrepôt de données (incluant les data marts).
- **Avantages :** Modèle de référence standardisé pour l'entreprise, séparation des problèmes d'extraction et d'intégration des données source.
- **Inconvénients :** Utilisation supplémentaire de l'espace de stockage, outils analytiques légèrement éloignés du temps réel.

Ces architectures permettent aux entreprises de choisir la structure qui répond le mieux à leurs besoins en matière de traitement et d'analyse de données, tout en assurant une gestion efficace et sécurisée des informations.



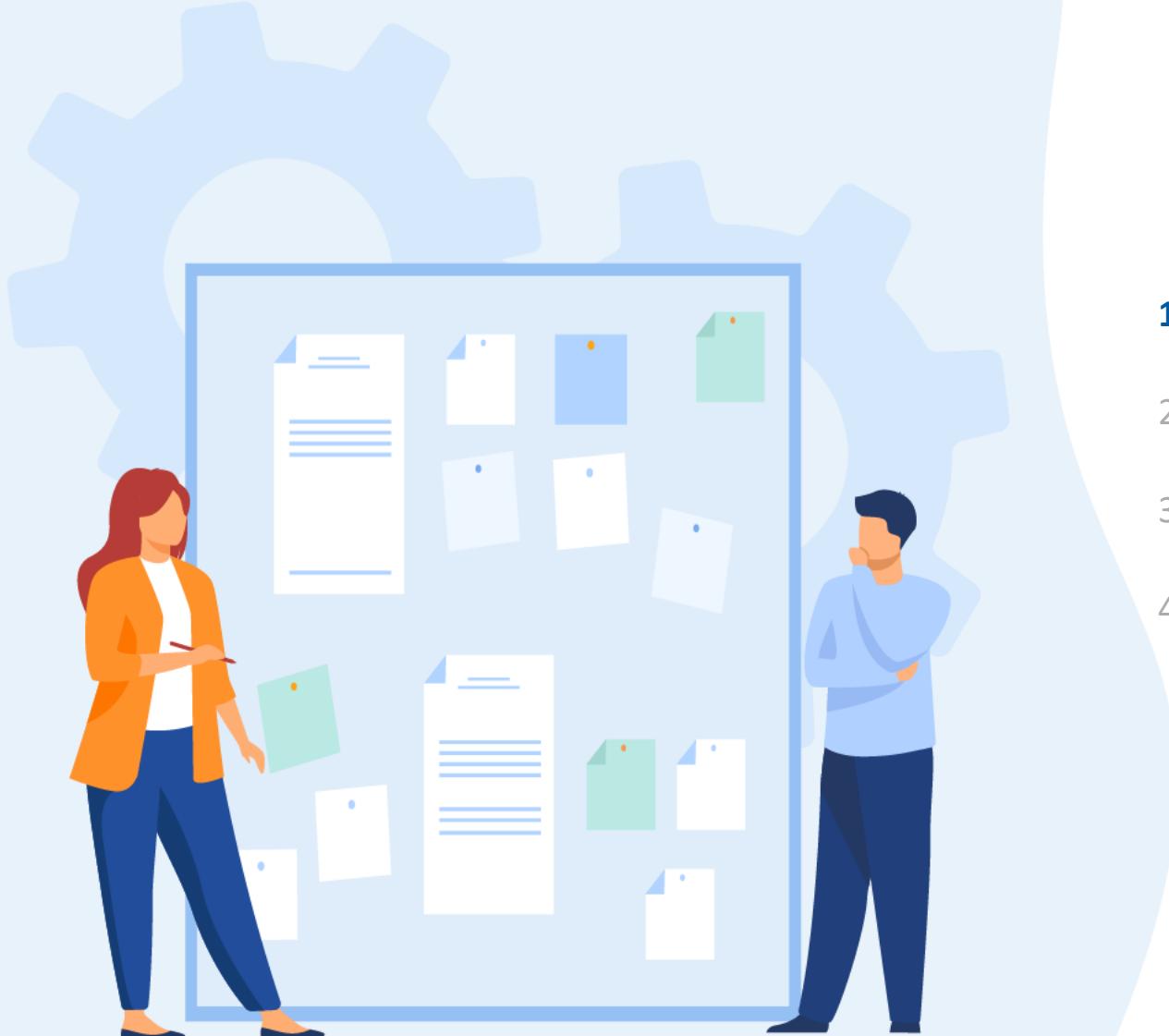


CHAPITRE 2

EXPLORER LA MODÉLISATION DIMENSIONNELLE AVANCÉE

Ce que vous allez apprendre dans ce chapitre :

- Rappeler les concepts avancés de modélisation dimensionnelle.
- Explorer les hiérarchies dans la modélisation dimensionnelle.
- Analyser les niveaux de granularité.
- Appliquer avec Python/Excel/R.



CHAPITRE 2

EXPLORER LA MODÉLISATION DIMENSIONNELLE AVANCÉE

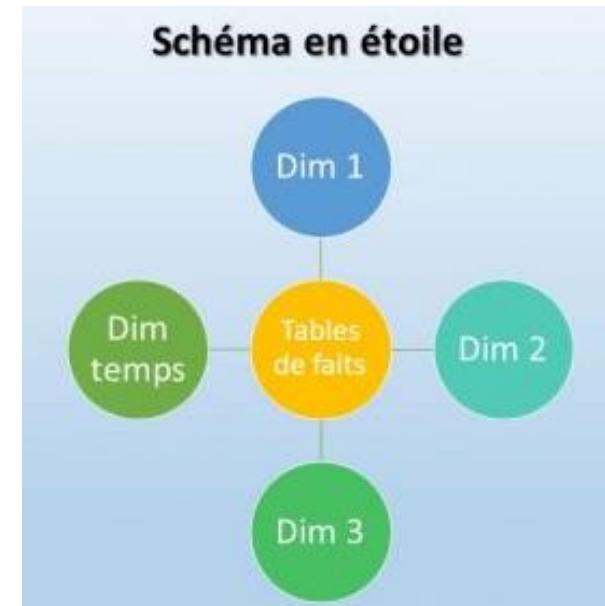
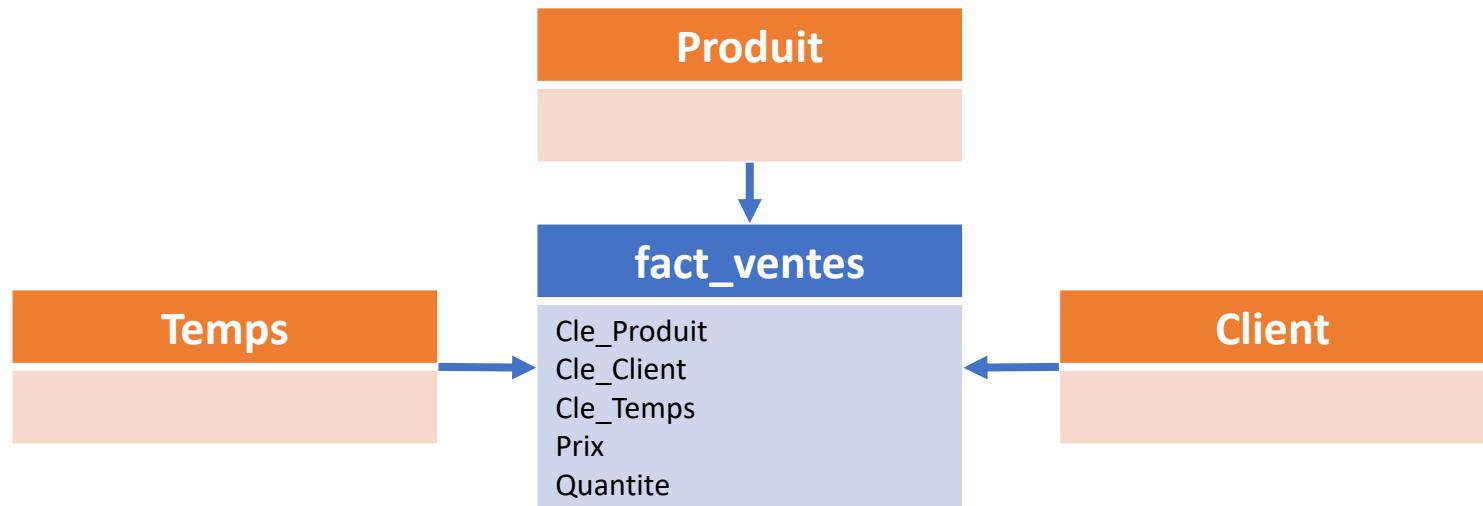
1. Rappel des concepts avancés de modélisation dimensionnelle
2. Hiérarchies dans la modélisation dimensionnelle
3. Niveaux de granularité
4. Consignes pour les applications futures

02 – Explorer la modélisation dimensionnelle avancée

Rappel des concepts avancés de modélisation dimensionnelle

Modèle en étoile (Star):

- **Définition :** Structure simplifiée avec une table de faits centrale connectée à plusieurs tables de dimensions.
- **Avantages :** Facile à comprendre et à utiliser pour les analyses simples et les requêtes rapides.
- **Exemple :** Table de faits des ventes liée aux tables de dimensions de produit, de client et de temps.



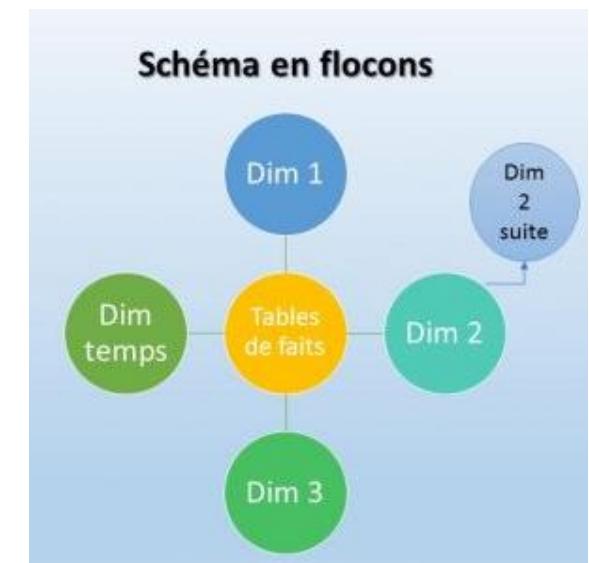
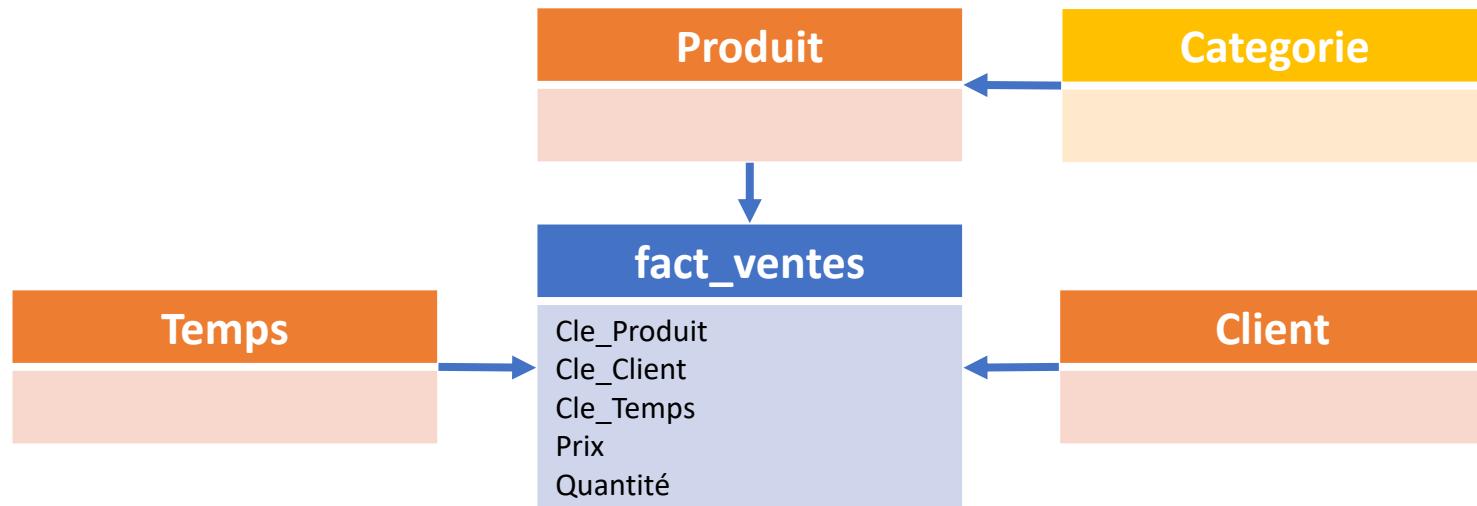
fact_ventes : cette table contient les références des tables de dimensions ainsi que deux faits Prix et quantité vendue.
Les 5 clés étrangères qui composent cette table **forment la clé primaire** de la table des faits.

02 – Explorer la modélisation dimensionnelle avancée

Rappel des concepts avancés de modélisation dimensionnelle

Modèle en flocon (Snowflake):

- Définition :** Extension du modèle en étoile où les tables de dimensions sont normalisées en plusieurs niveaux de tables.
- Avantages :** Réduction de la redondance et amélioration de la maintenance des données.
- Exemple :** Table de faits des ventes liée à une table de dimension produit, qui est à son tour liée à une table de catégorie de produit.



02 – Explorer la modélisation dimensionnelle

avancée

Hiérarchies dans la modélisation dimensionnelle

Modélisation dimensionnelle : Example de hiérarchie temporelle

- L'Organisation des données en niveaux, permettant des analyses à différents degrés de détail.
- **Avantages :** Facilite l'agrégation des données et les analyses multi-niveaux.
- **Utilisation :** Naviguer à travers les niveaux pour des insights plus précis et détaillés.
- **Exemple :** Hiérarchie temporelle (année > trimestre > mois > jour).



Niveaux de granularité : Quotidiennement, Mensuellement, ou Annuellement ?

Définition : Degré de détail ou de précision des données dans une table de faits.

- **Le niveau de granularité** dans une table de faits réfère au degré de détail ou de précision des données. Par exemple, les données peuvent être agrégées quotidiennement, mensuellement, ou annuellement. Le choix de la granularité est crucial car il détermine la spécificité des analyses possibles. Une granularité fine, comme les données quotidiennes, permet des analyses très détaillées mais peut entraîner une augmentation significative du volume de données, ce qui peut affecter les performances du système.
- À l'inverse, une granularité grossière, comme les données mensuelles ou annuelles, réduit le volume de données, facilitant ainsi le stockage et les performances des requêtes, mais au prix d'une perte de détails.

Résumé :

Granularité élevée : Données agrégées offrant une vue d'ensemble des tendances globales.

Granularité moyenne : Données plus spécifiques permettant d'explorer des modèles et des relations entre variables.

Granularité fine : Détails très précis pour identifier les anomalies et les transactions individuelles.

02 – Explorer la modélisation dimensionnelle avancée

Hiérarchies dans la modélisation dimensionnelle



Choix de niveaux de granularité : Quotidiennement, Mensuellement, ou Annuellement ?

- Les **niveaux de granularité** se rapportent principalement au **niveau de détail des données**, mais ils peuvent être appliqués à différents aspects, y compris le **temps** et les **types de données**.
- Le choix de la granularité dépend des besoins d'analyse spécifiques de l'entreprise et des contraintes techniques du système de gestion des bases de données décisionnelles.
- Combiner différents niveaux de granularité permet d'obtenir une compréhension complète des données, en équilibrant le détail et la complexité.

Granularité temporelle :

La granularité temporelle peut être exprimée par des périodes comme des années, des mois, des jours, voire des minutes. Une granularité temporelle élevée (années) donne une vue d'ensemble, tandis qu'une granularité faible (minutes) fournit des détails plus précis.

Granularité des données:

Elle s'applique également à la structure des données. Par exemple, des données agrégées par produit (**granularité élevée**) peuvent être décomposées jusqu'à chaque transaction individuelle (**granularité faible**)

Quand on parle de **granularité élevée**, cela signifie que les données sont agrégées, offrant ainsi une vue d'ensemble ou une image globale. À ce niveau, les détails sont réduits et les données sont présentées de manière plus synthétique, comme les ventes totales par région plutôt que par magasin individuel.

Utilisation combinée de Python, Excel et R, ou d'autres outils performants

Déconseillé en milieu professionnel :

- **Importer et nettoyer les données avec Python:** Utiliser pandas pour transformer les données brutes en tables de faits et de dimensions.
- **Analyser et visualiser les données avec Excel:** Utiliser les tableaux croisés dynamiques et les graphiques pour explorer les données de manière interactive.
- **Effectuer des analyses statistiques avancées avec R:** Utiliser dplyr et ggplot2 pour des analyses détaillées et des visualisations personnalisées.

Recommandé en milieu professionnel :

- Préparez-vous à utiliser des outils comme **PowerBI** pour la création de dashboards interactifs et le reporting dynamique.
- **Talend** est recommandé pour automatiser les processus ETL complexes.
- Familiarisez-vous avec les solutions cloud comme **Azure** pour le traitement de données à grande échelle et **BOOMI** pour l'intégration de données entre divers systèmes.

Ces technologies sont essentielles pour tout Data Analyst travaillant dans des environnements professionnels modernes où l'intégration et la transformation des données sont cruciales.



CHAPITRE 3

APPLIQUER LE PROCESSUS ETL DANS LE CONTEXTE DÉCISIONNEL

Ce que vous allez apprendre dans ce chapitre :

- Rappeler le processus ETL.
- Évaluer la qualité des données.
- Comparer les outils ETL et le processus de codage manuel
- Identifier les points clés durables



CHAPITRE 3

APPLIQUER LE PROCESSUS ETL DANS LE CONTEXTE DÉCISIONNEL

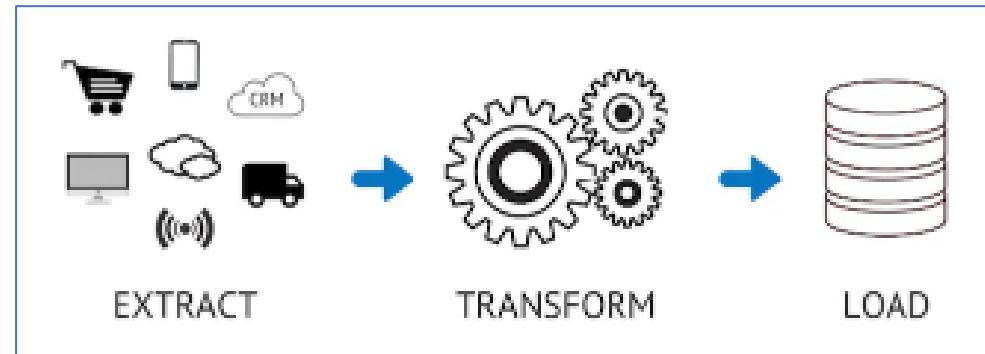
1. Rappel de l'ETL
2. Qualité des données
3. Outils ETL et processus de codage manuel
4. Points clés durables

03 – Appliquer le processus ETL dans le contexte décisionnel

Rappel de l'ETL

Introduction vers le processus ETL

- **Définition:** L'ETL (Extraction, Transformation, Chargement) désigne un ensemble de processus visant à collecter, structurer et centraliser des données provenant de diverses sources.
- **Problématique:** Une étude révèle que deux tiers des entreprises n'exploitent pas pleinement le potentiel de leurs données, souvent confinées dans des silos ou des systèmes hérités.



03 – Appliquer le processus ETL dans le contexte décisionnel

Rappel de l'ETL



Cas d'Usage

Migration de données:

Transfert d'informations entre applications.

RéPLICATION:

Création de sauvegardes et analyse des redondances.

Optimisation:

Déplacement des données d'un CRM vers un ODS pour enrichissement.

Stockage:

Préparation des données dans un entrepôt pour la Business Intelligence.

Migration vers le Cloud:

Déplacement d'applications vers des infrastructures cloud.

CRM Customer Relationship Management: Un CRM est utilisé pour gérer les interactions d'une entreprise avec ses clients, en centralisant des informations sur les ventes, le marketing et le service client.

ODS Operational Data Store: Un ODS est une base de données qui stocke des données opérationnelles à jour et consolidées. Il est souvent utilisé pour des requêtes et des analyses en temps réel.

03 – Appliquer le processus ETL dans le contexte décisionnel

Rappel de l'ETL

Étape 1 – Extraction :

- **Objectif:** Produire des données propres et accessibles.
- **Sources:** Les données brutes peuvent être extraites de différentes sources, en particulier :
 - Bases de données existantes
 - Data warehouse
 - Fichiers (XML, JSON, Excel, CSV, text, etc.)
 - Logs d'activité (trafic réseau, rapports d'erreurs, etc.)
 - Comportement, performances et anomalies des applications
 - Événements de sécurité
 - Autres opérations qui doivent être décrites aux fins de conformité



03 – Appliquer le processus ETL dans le contexte décisionnel

Rappel de l'ETL

Étape 2 – Transformation :

- **Objectif** : Nettoyage et conversion des données pour le reporting.
- **Normes à rappeler** :

Standardisation : Définir formats et modes de stockage.

Déduplication : Identifier et supprimer doublons.

Vérification : Comparaison pour détection d'anomalies.

Tri : Organisation par catégories pour optimiser l'efficacité.

- **Rappel** :

Normalization

Adjusting values to a specific range, typically between 0 and 1.

Standardization

Rescaling data to have a mean of 0 and a standard deviation of 1.



TRANSFORM

03 – Appliquer le processus ETL dans le contexte décisionnel

Rappel de l'ETL



Étape 3 – Load / Chargement :

- **Objectif:** transférer des données extraites et transformées vers un emplacement centralisé, garantissant leur intégrité et facilitant l'analyse pour soutenir la prise de décision.
- **Modes:** Chargement complet ou incrémentiel.
- **Gestion des erreurs:** Importance de la surveillance et du traitement des exceptions.



Sources d'erreurs

Problèmes d'intégration : Mauvaise communication entre différentes sources de données (ERP, CRM, etc.), entraînant des erreurs dans la transmission ou l'unification des informations.

Erreurs humaines : Mauvaise **saisie** de données par les utilisateurs finaux ou lors de la migration de données.

Doublons : Données redondantes ou dupliquées, souvent présentes lorsqu'il y a plusieurs sources ou bases de données **non synchronisées**.

Données manquantes : Absence de certaines valeurs dans les ensembles de données, souvent due à une **mauvaise collecte** ou à une **migration de systèmes**.

Systèmes legacy : Les anciens systèmes peuvent générer des données obsolètes ou **incompatibles** avec les nouveaux environnements, créant des erreurs lors de l'intégration ou de la **migration**.

Données incohérentes : Incohérences dans les **formats** ou les **valeurs** (par ex. dates au format US vs format EU), causées par des entrées de données provenant de **multiples systèmes ou applications**.

Vous devez anticiper ces erreurs et être prêt à les gérer.

Techniques d'amélioration de la qualité des données

- Nettoyage des données** : Supprime ou corrige les erreurs telles que les valeurs aberrantes, les doublons, ou les données manquantes pour assurer une qualité optimale.
- Déduplication** : Identifie et élimine les doublons dans les bases de données en utilisant des techniques de correspondance pour améliorer l'exactitude.
- Profilage des données** : Analyse les données pour identifier des incohérences et erreurs potentielles, permettant une correction proactive et une meilleure compréhension.
- Gouvernance des données** : Implémente des politiques et des procédures pour garantir une gestion cohérente et contrôlée des données à long terme.
- Surveillance continue** : Utilise des systèmes de monitoring pour détecter et alerter sur les anomalies en temps réel, permettant des corrections rapides.

Rôle « Data Quality Analysts » : Qu'est-ce qu'un Analyste de la Qualité des Données ?

Un **analyste de la qualité des données** garantit que les ensembles de données sont **précis, fiables, et utilisables** en les nettoyant, les validant et en corrigeant les incohérences pour répondre aux objectifs métier. Responsabilités clés :

1. Développement de normes de qualité
2. Nettoyage et enrichissement des données
3. Suivi et reporting de la qualité des données
4. Analyse des causes racines
5. Amélioration des processus

Rôle « Data Quality Analysts » : Exemple de cas d'utilisation Gestion des données clients dans un CRM

- 1. Développement de normes :** Une entreprise de e-commerce définit des **normes de qualité** pour ses données clients afin de garantir la **précision et la cohérence** des informations, telles que l'exactitude des adresses email et des numéros de téléphone. Les critères incluent la validation de l'unicité des adresses et la vérification des champs obligatoires.
- 2. Nettoyage et enrichissement :** L'entreprise détecte des **erreurs** (doublons, adresses incorrectes) et utilise des outils de nettoyage pour **rectifier ces erreurs**. Ensuite, elle enrichit les profils clients en ajoutant des données manquantes, comme les préférences d'achat, à partir de sources externes fiables.
- 3. Suivi et reporting :** L'entreprise implémente des **indicateurs de qualité** pour surveiller la validité des informations, comme le taux de rebond des emails marketing, et produit des **rapports mensuels** qui permettent aux décideurs d'identifier et d'intervenir rapidement.
- 4. Analyse des causes :** En cas de forte dégradation de la qualité des données (ex. : taux élevé de retours de colis dus à des adresses erronées), une **analyse des causes** est menée pour comprendre l'origine de ces erreurs (ex. : mauvaise saisie par les utilisateurs ou défaut dans le processus de mise à jour des données).
- 5. Amélioration des processus :** L'entreprise recommande des **améliorations du processus de saisie** dans le CRM, telles que la mise en place de formulaires avec des contrôles de validation automatisés pour **minimiser les erreurs futures** lors de la saisie des données par les utilisateurs.



CHAPITRE 3

APPLIQUER LE PROCESSUS ETL DANS LE CONTEXTE DÉCISIONNEL

1. Rappel de l'ETL
2. Qualité des données
3. **Outils ETL et processus de codage manuel**
4. Points clés durables

03 – Appliquer le processus ETL dans le contexte décisionnel

Outils ETL et processus de codage manuel



Outils ETL vs. processus de codage manuel ETL

De nombreux professionnels de l'IT se demandent si **le codage manuel** n'est pas une meilleure solution que d'investir dans de nouveaux outils. La plupart des fonctions ETL peuvent être définies par codage manuel, mais **les outils ETL** sont généralement plus évolutifs et moins coûteux à long terme.

Le **codage manuel** présente de nombreux défis : **administration, support technique, et réutilisation** sont **complexes**. Il est souvent difficile pour un développeur de comprendre ou réutiliser le code d'un autre.

Les **outils ETL** offrent une représentation visuelle des flux de données, bien plus **facile à comprendre**. Avec le codage manuel, la réécriture de code est courante, augmentant les coûts de maintenance qui peuvent être jusqu'à deux fois plus élevés.



CHAPITRE 3

APPLIQUER LE PROCESSUS ETL DANS LE CONTEXTE DÉCISIONNEL

1. Rappel de l'ETL
2. Qualité des données
3. Outils ETL et processus de codage manuel
- 4. Points clés durables**

03 – Appliquer le processus ETL dans le contexte décisionnel

Points clés durables



Points clés durables

Dans ce module, plusieurs concepts fondamentaux ont été abordés, des connaissances qui vous accompagneront tout au long de votre carrière en analyse de données. Voici quelques conseils pratiques à retenir pour maximiser votre impact en tant que professionnel de la donnée.

1. Comprendre vos objectifs et vos données : Avant toute analyse, il est crucial de bien comprendre l'objectif que vous poursuivez. Demandez-vous : **Quelle est la question à laquelle je veux répondre ?** Cette réflexion orientera vos choix de mesures statistiques et de méthodes d'analyse. En parallèle, identifiez toujours le type de données dont vous disposez (**discrètes, continues, ordinaires, nominales**). Cela vous évitera des erreurs d'interprétation et vous permettra de choisir **les méthodes appropriées**.

03 – Appliquer le processus ETL dans le contexte décisionnel

Points clés durables



Points clés durables : suit

2. Maîtrisez les mesures statistiques pour mieux interpréter vos résultats : Les **mesures centrales** comme la moyenne, la médiane et le mode sont des bases solides, mais ne vous limitez pas à elles. Les **mesures de dispersion** (écart-type, variance) vous donnent une idée de la variabilité des données et sont essentielles pour évaluer la qualité de vos modèles. Gardez en tête que chaque mesure a sa propre signification et qu'elle doit être choisie en fonction du contexte de votre analyse.

3. Choisissez vos graphiques stratégiquement : Ne sous-estimez jamais le pouvoir de la visualisation des données. Le choix du graphique doit toujours être aligné avec votre objectif. Si vous cherchez à **montrer la relation** entre deux variables, un **scatter plot** est souvent le plus pertinent. Si votre objectif est de **comparer des catégories**, tournez-vous vers les **bar charts**. Plus vos données sont complexes, plus la sélection de vos graphiques doit être réfléchie. Pour des **analyses multivariées**, par exemple, explorez les techniques comme les **projections géométriques** ou les **techniques basées sur les icônes..**

03 – Appliquer le processus ETL dans le contexte décisionnel

Points clés durables



Points clés durables : suit

4. Utilisez la couleur et les annotations de manière judicieuse : La couleur et les annotations sont des outils puissants pour renforcer la clarté de vos visualisations. Un bon choix de couleurs peut rendre vos visualisations **claires et accessibles**, tandis qu'un **mauvais choix** peut prêter à confusion. Optez pour des palettes contrastées qui permettent une bonne lisibilité et soyez attentif à la signification des couleurs que vous employez (**rouge pour le danger ou les baisses**, **vert pour les augmentations**). De plus, utilisez **les annotations** pour **clarifier** les informations clés, ajouter des explications pertinentes, ou guider l'interprétation des données **sans surcharger le graphique**.

5. Pensez toujours à la qualité des données : La qualité des données est essentielle pour toute analyse fiable. Apprenez à identifier les outliers et à décider s'ils doivent être conservés ou exclus de votre analyse. Adoptez une approche rigoureuse dans le **processus ETL** (extraction, transformation, chargement) pour garantir que vos données sont non seulement **propres**, mais aussi **pertinentes pour l'analyse**.

Ces principes, lorsqu'ils sont bien intégrés, ne serviront pas seulement dans les prochains modules, mais constitueront une ressource précieuse tout au long de votre carrière.