**IUT Informatique Clermont Auvergne – Aubière**


# Rapport projet Data Mining

Par
**Rochelle Hugo, Carreau Alexis, Sabatier Audric**

*Projet réalise durant le cours de data mining dispensé par Mr.Falih Issam durant le 2ème semestre de 3ème année de BUT Informatique.*

Clermont-Ferrand
2024

# Introduction

Durant notre 3ème et dernière année de BUT Informatique, nous avons eu un cours de "Data Mining". Nous avons dû réaliser ce projet qui a pour but de créer une application en Python qui permet à un utilisateur d'importer, analyser, nettoyer et visualiser des données. Nous devions également permettre à l'utilisateur de regrouper les données grâce à des algorithmes de clustering et de les afficher une fois qu'ils sont regroupés.

## 0.1 Architecture

Pour notre projet, travaillant à trois, nous avons décidé de séparer le visuel des fonctionnalités de traitement et de calcul des données dans des dossiers distincts. En plus des fonctionnalités principales, nous avons inclus un dossier pour la documentation générée via Sphinx, utilisant DocString. Notre configuration produit un fichier HTML qui décrit module par module les fonctions et leurs rôles. Biens sûr, chaque fichier correspond à une fonctionnalité et nous avons découpé les fonctionnalités par fonction au maximum.

Nous avons adapté l'architecture de base du projet Streamlit comme suit :

```
App
├── docs
│   └── build
│       └── html
│           └── index.html
└── src
    ├── back
    │   ├── clustering.py
    │   ├── prediction.py
    │   └── dataCleaning.py
    ├── pages
    │   ├── clusteringPrediction.py
    │   ├── dataCleaning.py
    │   ├── dataVisualisation.py
    │   └── loadData.py
    └── app.py
```

Cette architecture montre que chaque page a un fichier backend associé, auquel elle fait appel pour traiter les données en fonction des paramètres entrés par l'utilisateur dans l'interface. Les fonctions backend retournent ensuite des valeurs affichées dans l'interface utilisateur. Tout est initié et géré par app.py, notre point d'entrée dans l'application, qui connaît tous les dossiers et orchestre leur interaction.

## 0.2 Nos fonctionnalités

### 0.2.1 Importation et pré-visualisation des données

Notre application regroupe plusieurs fonctionnalités pour pouvoir traiter et analyser des jeux de données.

La première que nous avons dû faire a été de permettre l'importation de datasets dans notre application. Nous avons aussi permis à l'utilisateur de pouvoir faire une prévisualisation des données qu'il vient d'importer. Nous affichons les 3 premières lignes du dataset et les 3 dernières, des informations sur le fichier, comme le nombre de lignes et de colones, le nom des colonnes, mais aussi le nombre de valeurs manquantes pour chaque colonnes.

### 0.2.2 Nettoyage et normalisation des données

Le premier traitement que nous avons mis en place est la gestion des données de type string. Nous avons permi à l'utilisateur de choisir parmi 3 choix pour traiter ces données :

- Les supprimer
- Les encoder avec un label encoder, qui permet de les transformer en valeurs numériques pour chaque colonne sans créer de nouvelles colonnes
- Les encoder avec le "one-hot", qui permet de créer une nouvelle colonne pour chaque choix disponible dans la colonne d'origine et qui met à true ou false la valeur dans la nouvelle colonne

Nous avons ensuite laissé la possibilité à l'utilisateur de choisir comment gérer les valeurs manquantes dans le dataset de 3 façons :

- En supprimant les valeurs manquantes
  - En supprimant les colonnes avec un seuil de valeurs manquantes (choisi par l'utilisateur)
  - En supprimant les lignes avec un seuil de valeurs manquantes (choisi par l'utilisateur)
  - En supprimant les lignes et les colonnes avec le même seuil
- En remplaçant les valeurs manquantes
  - En remplaçant avec la moyenne
  - En remplaçant avec la médiane
  - En remplaçant avec le mode (valeur la plus présente)
- En faisant une imputation plus sophistiquée
  - Avec KNN, en choisissant le nombre de voisins à prendre en compte
  - Avec une régression en utilisant un estimateur au choix de l'utilisateur parmi :
    * BayesianRidge

* LinearRegression
* Ridge
* RandomForestRegressor
* SVR

Nous avons permis à l'utilisateur de normaliser ses données avec 2 méthodes différentes :

- Le Z-score qui permet de centrer les données autour de 0 avec écart type de 1. La formule du z-score :
$$z = \frac{x - \mu}{\sigma}$$
- Le min-max qui normalise les données entre 0 et 1. La formule de la normalisation min-max :
$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

### 0.2.3 Visualisation des données

Une fois les données nettoyé et préparé grace aux différents processus vu précédement, nous avons pu réaliser une première visualisation de celle-ci. Nous avons fait le choix de rendre la visualisation flexible en laissant le choix à l'utilisateur de l'outil de choisir le type d'affichage qu'il souhaite puis en fonction du type d'affichage la colonne qu'il souhaite voir représenté sur le graphique généré.

Ensuite, selon le type de figure choisis, nous filtrons les variable proposé pour ne proposer que les variables qui peuvent être représenter pour un type de données. Par exemple, si l'utilisateur choisis un affichage en courbe, alors les variables de type string ne pourront pas être représenté car cene sont pas des variables quantitatives mais qualitatives.

Vous pouvez retrouver ci-dessous un exemple de visualisation de données sous forme d'histogramme. Ici, la variable représenté est le poids moyen des humains présent dans notre jeux de données :

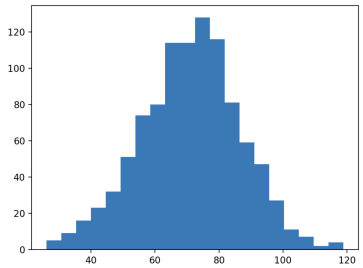


Figure 1: Exemple de visualisation de données

Ce qui peut être intéressant dans une visualisation de données est très bien illustré sur la figure ci-dessus. Par exemple, nous pouvons dors et déjà dire que la majorité des humains de notre dataset pèse entre 60kg et 90kg. On peut également savoir assez rapidement l'étendu de la variable.

### 0.2.4 Visualisation des clusters

Nous avons vu précédement une forme de visualisation des données après leur nettoyage ainsi que le faite que nous pouvions en tirer certaines informations.

Une autre visualisation que nous avons mis en place est celle de la visualisation des clusters après avoir appliqué un algorithme de clustering sur le jeux de données comme par exemple, K-Means ou bien DBSCAN.

Cette visualisation est assez intéressante car elle permet de représenter graphiquement certaines informations comme par exemple les centroïdes de chaque cluster. Afin de visualiser au mieux nos clusters, nous avons fait le choix de les représenter dans une figure à deux dimensions mais également dans une figure à trois dimensions. Vous pouvez retrouver ci-dessous la représentation d'un ensemble de clusters dans une visualisation en 3D :

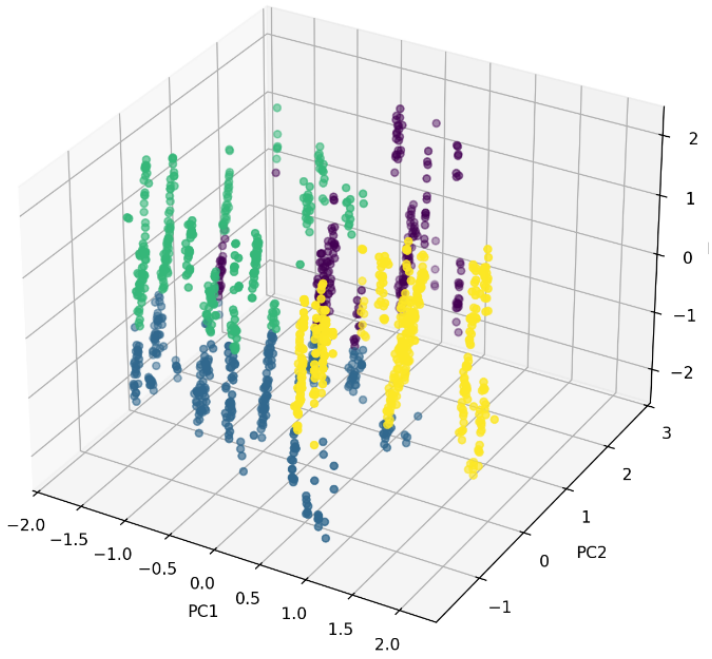


Figure 2: Exemple de visualisation de cluster (3d)

## 0.3 Analyses des résultats

Notre application nous permet de montrer à l'utilisateur certaines données utiles pour son analyse.

### 0.3.1 Prédiction

Régression :

- Erreur quadratique moyenne :moyenne des carrés des écarts entre les valeurs prévues par le modèle et les valeurs réelles (MSE faible indique un modèle précis)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Coefficient de détermination : mesure la proportion de la variance des variables dépendantes expliquée par les variables indépendantes (un $R^2$ proche de 1 indique un bon ajustement)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

- Erreur absolue moyenne : moyenne des valeurs absolues des erreurs entre les prédictions et les valeurs réelles (une MAE plus faible indique une meilleure performance)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Erreur quadratique moyenne racine : racine carrée de la MSE, donnant une mesure de l'erreur quadratique moyenne dans les mêmes unités que la variable à prédire (une RMSE plus faible indique un meilleur ajustement)

$$\text{RMSE} = \sqrt{\text{MSE}}$$

Classification :

Rapport de classification : inclut des métriques de performance pour évaluer un modèle de classification.

- Précision : proportion des prédictions correctes parmi les prédictions positives faites.

$$\text{Précision} = \frac{TP}{TP + FP}$$

- Rappel : proportion des vrais positifs détectés parmi tous les échantillons positifs.

$$\text{Rappel} = \frac{TP}{TP + FN}$$

- F1-Score : moyenne harmonique de la précision et du rappel, équilibrant les deux.

$$\text{F1-Score} = 2 \cdot \frac{\text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

### 0.3.2 Clustering

Loadings : analyse de l'amplitude et direction. L'amplitude indique la force de la relation entre l'entité et le composant principal. La direction (signe positif ou négatif) indique si la caractéristique est en corrélation positive ou négative avec le composant. Nous visualisons les résultats sous forme de matrice.

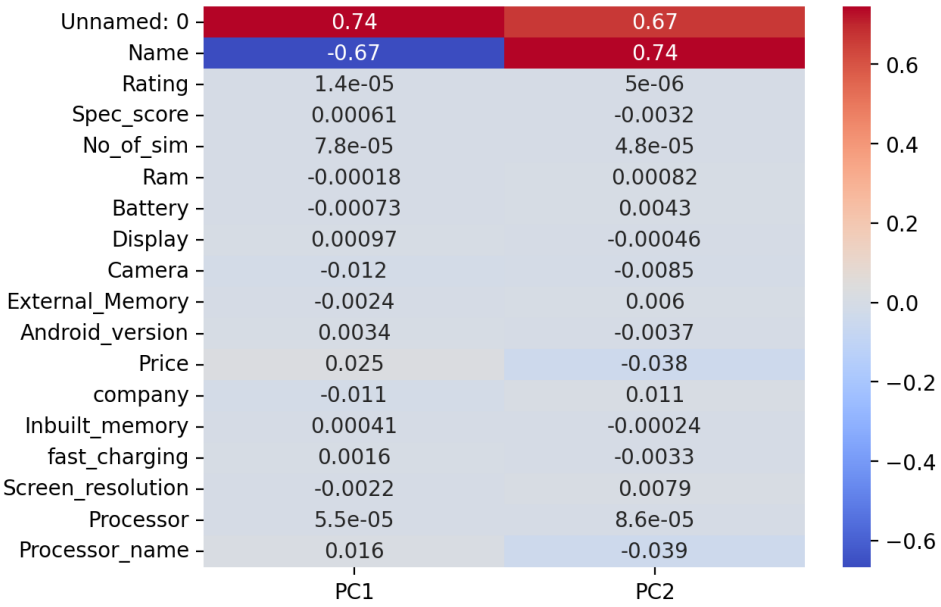


Figure 3: Matrice des loadings

Silhouette : analyse de l'amplitude et direction. L'amplitude indique la force de la relation entre l'entité et le composant principal. La direction (signe positif ou négatif) indique si la caractéristique est en corrélation positive ou négative avec le composant. Nous visualisons les résultats sous formes de matrice.

## 0.4 Problèmes rencontrés

Lors de ce projet, nous avons rencontré plusieurs problèmes, notamment lorsque nous appelions nos fonctions de traitement telles que le clustering ou même la visualisation, sans avoir au préalable nettoyé nos données. Ce manque de nettoyage préalable des données a conduit à des erreurs et à des résultats incohérents, car les données brutes contenaient des valeurs manquantes, des valeurs aberrantes et d'autres imperfections qui affectaient la qualité de l'analyse. Pour résoudre ce problème, nous avons décidé de rendre le nettoyage des données obligatoire avant de pouvoir accéder à ces parties critiques du projet. Ainsi, nous avons pu garantir que les données utilisées pour le clustering et la prédiction étaient toujours optimal, ce qui a grandement amélioré la fiabilité et la pertinence de nos résultats.

Nous avons également rencontré des difficultés lorsqu'il s'agissait d'interpréter et de sélectionner les différents indicateurs que nous souhaitions proposer dans ce projet. La multitude d'indicateurs disponibles et les différentes méthodologies d'évaluation pouvaient prêter à confusion et rendre la tâche de sélection des indicateurs complexe. Pour surmonter ce défi, nous nous sommes informer sur les indicateurs les plus pertinents.

## 0.5 Utilisation Outils génératifs

Utilisation d'outils génératifs au sein du projet à hauteur de 80%. Nous les avons principalement utilisés pour des tâches de correction de bug et de documentation sur les méthodes à utiliser.

# Remerciements

Nous tenons à remercier M. Falih Issam pour son enseignement durant ce cours de data mining. Ses précieux conseils et sa disponibilité nous ont permis de mener à bien ce projet et d'approfondir nos connaissances en data mining. Nous remercions également nos camarades de classe pour leur soutien et leur collaboration tout au long de ce projet.

# Références

1. Falih Issam, "Cours de data mining", IUT Informatique Clermont Auvergne – Aubière, 2024.

2. "Streamlit documentation", consulté en ligne le 25 juin 2024, disponible à l'adresse : `https://docs.streamlit.io/`

3. "Scikit-learn documentation", consulté en ligne le 25 juin 2024, disponible à l'adresse : `https://scikit-learn.org/`