Rapport TDM1 : Introduction aux sciences des données et à l'intelligence artificielle

Fanani Selma et Amsaf Rim (Groupe 2)

September 11, 2024

1 Introduction

Dans le cadre de ce TP, nous allons tout d'abord tester différentes méthodes de normalisation sur un jeu de données. Ensuite, ces données normalisées seront représentées sous forme de nuages de points afin de mieux visualiser les distributions et les éventuels patterns sous-jacents.

Dans une seconde étape, nous appliquerons l'Analyse en Composantes Principales (ACP) sur un autre jeu de données pour évaluer son impact sur la visualisation. Cette technique nous permettra de réduire la dimensionnalité des données tout en essayant de conserver un maximum d'information afin de mieux interpréter les structures des données.

Enfin, nous testerons trois méthodes différentes pour estimer les valeurs manquantes dans nos données. Ces méthodes seront comparées en termes d'efficacité afin de déterminer laquelle permet la meilleure estimation dans le contexte des jeux de données étudiés.

2 Normalisation :

2.1 Les différents types de normalisation :

Pour cette partie, nous importons un jeu de données à partir d'un fichier CSV, puis stockons les données dans une liste. Après l'importation, la liste créée contient des lignes vides que nous allons simplement ignorer. Pour pouvoir normaliser nos données vu que nous utilisons des méthodes de la bibliothèque pandas nous allons à partir de la liste créer une dataframe, et étant donnée que les valeurs sont initialement de type String, les transformer en int pour pouvoir effectuer les calculs.

Nous allons implémenter trois fonctions de normalisation, en plus de la fonction déjà implémentée qui normalise nos données en fonction des valeurs minimales et maximales que peuvent prendre nos variables.

Le but de la normalisation des données est de mettre toutes les variables sur une même échelle afin de rendre les comparaisons plus significatives, surtout lorsque les variables ont des unités ou des plages de valeurs très différentes.

2.1.1 Normalisation par la moyenne :

$$x_{i,j} = \frac{x_{i,j} - \mu_j}{\max_k(x_{i,k}) - \min_k(x_{i,k})}$$

Cette formule a été utilisée pour normaliser les données en suivant le modèle de la normalisation Min-Max. Nous avons parcouru toutes les valeurs des variables et appliqué cette formule à chacune d'entre elles, ce qui permet de transformer les données dans une plage généralement comprise entre 0 et 1.

Remarque : Certaines valeurs se sont retrouvées négatives après la normalisation. Ces valeurs correspondaient aux observations qui étaient inférieures à la moyenne μ_j de la variable, d'où leur position en dessous de zéro après la transformation.

Nous allons procéder de la même manière pour les deux derniers types de normalisation ; seule la formule qui calcule les nouvelles valeurs change.

2.1.2 Normalisation par standaridation :

Formule de normalisation par standardidation :

$$x_{i,j} = \frac{x_{i,j} - \mu_j}{\sigma_j}$$

avec σ_j l'écart-type.

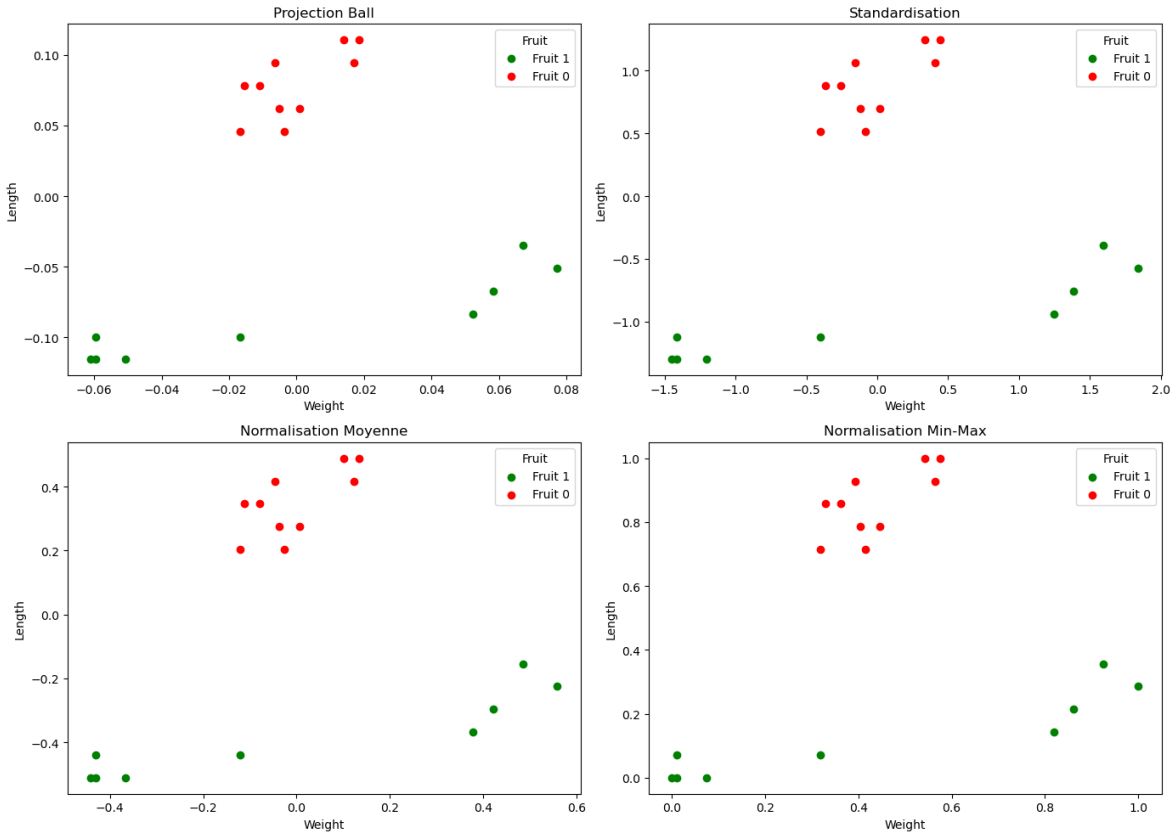
2.1.3 Normalisation par projection sur la surface d'une sphère :

Pour calculer la norme de toutes les valeurs d'une colonne nous avons utiliser la méthode linalg.norm() de la bibliothèque numpy. Voici la formule utilisée pour ce type de normalisation :

$$x_{i,j} = \frac{x_{i,j} - \mu_j}{\|x_j\|}$$

2.2 Représentation des données normalisées par les différentes méthodes en nuage de points :

Après avoir représenté nos données en nuages de points selon les différents types de normalisation nous obtenons des représentations similaires, nos données ont bien gardés leurs propriétés mais la marge dans laquel se trouve nos données est differente. Celle-ci nous donnes des informations suplémentaires sur nos données, prenons l'exemple de la normalisation par la moyennes toutes les valeurs de poid négatives signifie que notre Fruit est à un poid inférieur à la moyenne et vice-versa.



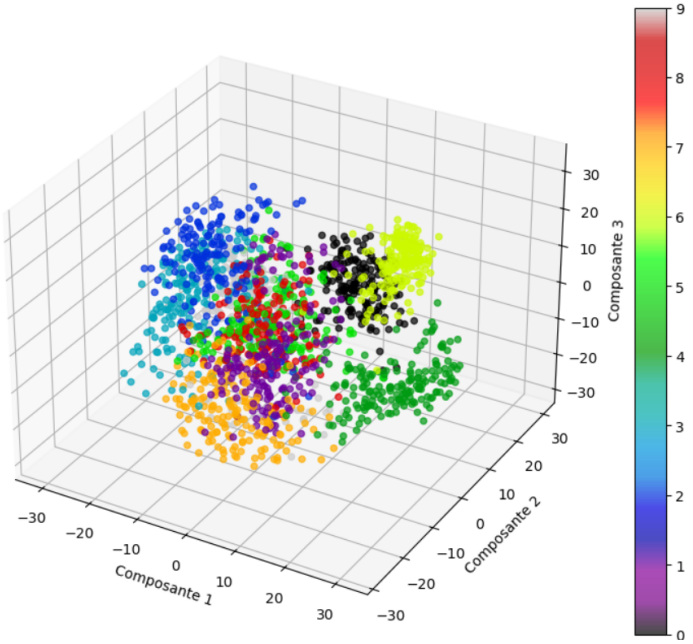
3 ACP

On remarque que la projection des données dans un espace à deux dimensions à l'aide de l'ACP ne permet pas de définir des frontières nettes entre les 10 classes de chiffres. Bien que certains regroupe-

ments soient visibles, notamment pour les chiffres "0", "1", et "9", d'autres classes, comme "3", "5", "8", et "2", montrent un fort chevauchement, ce qui entraîne des confusions importantes. De plus, les chiffres "7" et "5" partagent également des régions communes, compliquant davantage leur distinction. Ces résultats s'expliquent par la nature non supervisée de l'ACP, qui maximise la variance totale des données sans chercher à optimiser la séparation entre les classes. Ainsi, il est difficile de délimiter des frontières précises entre certaines classes, ce qui pourrait poser des problèmes si cette projection était utilisée pour une tâche de classification.

L'ACP est efficace pour réduire la dimension et visualiser les données, mais elle n'est pas optimale pour une tâche de classification car elle ne prend pas en compte les labels des données. Les chevauchements observés montrent les limites de l'ACP pour séparer des classes dans un espace réduit.

En refaisant cette ACP mais en ne conservant que les 3 premières composantes principales, On obtient bien le graphe suivant :



La projection des données de chiffres manuscrits en 3D à l'aide de l'ACP montre que les chevauchements observés en 2D persistent. Bien que la visualisation en 3D offre une perspective différente, elle n'améliore pas significativement la séparation des classes. Certains chiffres, comme "0", "1", et "9", qui étaient relativement bien séparés en 2D, ne le sont plus en 3D, On remarque aussi que d'autres classes continuent de montrer des chevauchements importants. Cela montre que l'ajout d'une dimension supplémentaire ne résout pas les problèmes de séparation inhérents à la nature complexe des données. Les chiffres manuscrits sont des données visuellement très variées, et il est difficile de capturer cette diversité avec une méthode linéaire comme l'ACP.

En conclusion, bien que la projection en 3D fournisse une vue différente des données, elle ne permet pas d'améliorer de manière significative la séparation entre les classes. L'ACP, en tant que méthode non supervisée, maximise la variance globale sans tenir compte des labels, ce qui limite son efficacité pour des tâches de classification.

4 Données manquantes

4.1 Calcul des valeurs manquantes globalement

La fonction **count_total_missing_values(data)** calcule le nombre total de valeurs manquantes en utilisant **np.isnan(data)** pour identifier les **nan**, puis np.sum() pour les additionner. En l'exécutant

sur notre jeu de données, nous avons trouvé 1241 valeurs manquantes.

4.2 Calcul des valeurs manquantes par colonne

La fonction **count_missing_values_per_column()** identifie les valeurs manquantes dans chaque colonne à l'aide de **np.isnan(data)**. Elle crée un tableau booléen où chaque élément est **True** pour les valeurs **nan**, puis utilise **np.sum(np.isnan(data), axis=0)** pour calculer le nombre de valeurs manquantes dans chaque colonne. Cette approche permet d'avoir une vue détaillée de la répartition des valeurs manquantes.

Pour vérifier la cohérence des résultats, nous avons calculé le nombre total de valeurs manquantes en utilisant la somme des valeurs manquantes par colonne. Nous avons effectué cette opération avec **count_missing_values_per_column()**. Ce total global des valeurs manquantes doit correspondre au nombre total calculé directement pour l'ensemble du jeu de données, confirmant ainsi la précision et la cohérence des deux méthodes de calcul.

4.3 Méthodes de complétion des valeurs manquantes :

Nous allons tester trois méthodes de complétion des valeurs manquantes, abordées en cours. Pour cela, nous avons utilisé des fonctions disponibles dans les bibliothèques Pandas, Numpy et Scikit-learn, faute de temps pour les implémenter nous-mêmes. Afin d'évaluer leur efficacité, nous avons calculé le score d'écart quadratique moyen. Les résultats obtenus sont les suivants :

Score d'écart moyen au carré pour l'imputation par mode : 41530.0

Score d'écart moyen au carré pour l'imputation par la moyenne des valeurs de la classe : 12092.226904459467

Score d'écart moyen au carré pour l'imputation par KNN : 4692.5999999999985

La méthode la plus efficace est : KNN avec un score d'écart moyen au carré de 4692.5999999999985

Le score d'écart moyen au carré (ou erreur quadratique moyenne) mesure la différence moyenne entre les valeurs réelles et les valeurs prédites ou imputées, en élevant chaque écart au carré. Il quantifie ainsi l'erreur moyenne commise par une méthode, les grandes erreurs ayant un impact plus significatif en raison de l'élévation au carré. Plus ce score est faible, plus la méthode est précise.

Les trois méthodes de complétion testées offrent des approches différentes pour traiter les valeurs manquantes.

Tout d'abord, la complétion par le mode remplace les valeurs manquantes par la valeur la plus fréquente dans chaque colonne. Bien que cette méthode soit rapide et facile à mettre en œuvre, elle peut entraîner un biais lorsque la distribution des données est hétérogène.

Ensuite, la complétion par la moyenne des classes est plus adaptée lorsque les données sont segmentées en catégories. Elle remplace les valeurs manquantes par la moyenne des valeurs pour chaque classe, ce qui permet une imputation plus fine en tenant compte des spécificités de chaque classe, mais cela nécessite une bonne définition des classes.

Enfin, la méthode des k plus proches voisins (KNN) remplace les valeurs manquantes en fonction des observations similaires dans le jeu de données. Cette méthode est plus flexible et robuste, car elle capture les relations complexes entre les variables, bien qu'elle soit plus coûteuse en termes de calcul.

References

[1] Pandas Documentation. *Pandas User Guide*. https://pandas.pydata.org/docs/user_guide/index.html

[2] Scikit-learn Documentation. *Scikit-learn User Guide*. https://scikit-learn.org/stable/

[3] NumPy Documentation. *NumPy Linear Algebra Reference*. https://numpy.org/doc/stable/reference/generated/numpy.linalg.norm.html