[1]Projet Data Science - Prédiction du risque d'abandon scolaire

Réalisé par :
Yasmine Elidrissi et Soumaya Hachicha

Objectifs pédagogiques :

- Suivre toutes les étapes d'un projet de classification supervisée
- Appliquer les métriques de performance : accuracy, precision, recall, f1-score - Tester plusieurs algorithmes : régression logistique, arbre de décision, KNN
- Utiliser GridSearchCV et la validation croisée
- Fournir une analyse critique des résultats obtenus

Explication détaillée du code et des étapes :

1. Chargement des bibliothèques et des données :
On importe les bibliothèques nécessaires (pandas, matplotlib, seaborn, sklearn) puis on charge le fichier CSV contenant les données des étudiants.

2. Aperçu et préparation des données :
On affiche les premières lignes avec `head()` et les types de données avec `info()`. On vérifie s'il y a des valeurs manquantes.

3. Encodage des variables catégorielles :

`Sexe` et `Situation_familiale` sont des chaînes de caractères. On utilise `LabelEncoder()` pour les convertir en valeurs numériques, ce qui est indispensable pour entraîner les modèles.

4. Analyse des corrélations :
On génère une `heatmap` qui montre les relations entre les variables. Puis, on extrait spécifiquement les corrélations avec la variable cible `Abandon`.

5. Préparation des features :
On sépare la cible (`Abandon`) des variables explicatives (`X`) et on standardise les données avec `StandardScaler` pour éviter que certaines variables dominent les autres.

6. Séparation des données :
On divise le jeu de données en deux parties : entraînement (80%) et test (20%) à l'aide de `train_test_split`.

---

[1]

7. Définition des modèles :
On définit trois modèles : régression logistique, arbre de décision, et KNN, qui sont testés dans une boucle.

8. Entraînement et évaluation : Pour chaque modèle :

- On entraîne avec `fit()`
- On prédit les résultats avec `predict()`
- On affiche la `matrice de confusion` et le `rapport de classification` (precision, recall, f1-score)

9. Visualisation spécifique :

- Pour l'arbre de décision : on affiche une visualisation graphique avec `plot_tree`.
- Pour la régression logistique : on affiche les coefficients pour interpréter les poids de chaque variable.

10. Courbe ROC :
On affiche pour chaque modèle la courbe ROC (Receiver Operating Characteristic) qui permet de visualiser le taux de vrais positifs en fonction des faux positifs. On calcule également l'AUC (area under curve) pour évaluer la performance globale du modèle.

Choix techniques et justifications du code utilisé :

1. Librairies utilisées :
Nous avons utilisé `pandas` pour manipuler les données, `seaborn` et `matplotlib` pour les visualisations, ainsi que `scikit-learn` pour les modèles de classification, la standardisation et l'évaluation des performances. Ce sont des outils standards en data science.

2. Encodage des variables catégorielles :
Nous avons choisi `LabelEncoder` car nos colonnes catégorielles ne comportent que quelques modalités. Cela permet d'utiliser ces variables avec des modèles comme la régression logistique.

3. Standardisation des données :
Nous avons appliqué un `StandardScaler` sur les données numériques. Ceci est essentiel notamment pour le modèle KNN et la régression logistique, qui sont sensibles à l'échelle des variables.

4. Modèles choisis :
- Régression logistique : simple, rapide et très interprétable. Permet de comprendre l'influence de chaque variable sur la probabilité d'abandon.[2]

---

[2]

- Arbre de décision : très lisible, permet de générer un modèle explicable pour les décideurs. Il donne de bonnes performances sans nécessiter de standardisation.
- KNN : basé sur la similarité, utile pour comparer des profils similaires. Il permet de tester une approche distance-based.

5. Séparation entraînement/test :
Nous avons séparé les données avec un `train_test_split` en conservant la stratification, pour garder la proportion des classes (abandons/non abandons) dans les deux ensembles.

6. Métriques utilisées :
- Accuracy : évalue la performance globale du modèle.
- Precision : taux de vrais positifs parmi les positifs prédits.
- Recall : taux de détection des vrais abandons (prioritaire ici).
- F1-score : moyenne harmonique entre précision et recall.
- Matrice de confusion : visualisation des erreurs de classification.

7. Courbe ROC et AUC :
Nous avons affiché une courbe ROC pour chaque modèle, avec l'AUC (aire sous la courbe) qui mesure la performance globale. Cela permet de comparer les modèles même avec des classes déséquilibrées.

8. Visualisation spécifique par modèle :
- Arbre de décision : visualisation via `plot_tree` pour comprendre les règles de décision.
- Régression logistique : affichage des coefficients pour interpréter le rôle de chaque variable. - ROC : outil standard pour comparer tous les modèles.

9. Interprétation des résultats :
Nous avons mis l'accent sur le recall pour ne pas rater les étudiants en difficulté. L'arbre de décision a été préféré pour son compromis performance/interprétabilité. La régression est précieuse pour comprendre les variables importantes.

Ce rapport montre que chaque étape du code et chaque choix sont justifiés par des objectifs pédagogiques et des besoins métier réels.

10. Parmi les trois modèles testés (Régression Logistique, Arbre de Décision et KNN), c’est l’arbre de décision qui a donné les meilleurs résultats dans le cadre de ce projet.[3]

- Il obtient le rappel (recall) le plus élevé pour la classe 1 (étudiants ayant abandonné), ce qui est l'objectif principal de ce projet : ne pas rater un étudiant à risque.

3

- Il est aussi facile à visualiser et à interpréter, ce qui est un avantage dans un cadre éducatif où les utilisateurs finaux (professeurs, conseillers) doivent comprendre les critères de décision.
- La régression logistique est très claire pour l'interprétation des variables mais obtient un rappel plus faible.
- Le KNN a montré une précision correcte, mais son recall est insuffisant, ce qui rend le modèle moins adapté ici.

Explication de ce que fait la PCA :

- Il réduit le jeu de données à 2 dimensions principales tout en conservant un maximum de variance (information).
- Il trace un nuage de points coloré selon l'abandon, ce qui permet de visualiser si les abandons forment un groupe distinct ou s'ils sont mélangés dans la population étudiante.
- Il affiche la variance expliquée par chaque composante pour justifier si la réduction de dimension est représentative.

Conclusion :

L'arbre de décision est retenu comme modèle principal pour sa simplicité et son efficacité en recall. Ce projet montre l'importance de l'analyse de données et de l'interprétation critique dans un contexte éducatif.

Améliorations possibles :
- Utiliser SMOTE pour équilibrer les classes
- Tester d'autres modèles comme RandomForest ou XGBoost - Ajouter des variables comportementales ou sociales[4]

4