ÉVALUATION UA1 : Prétraitement des Données avec Pipelines

Nom(s) et prénom(s) des membres du groupe :

Tatchuenwa Nziguem Kendric Guerrin

Aissata Sanoh
Roslin Ivan Jouanang Komguep

Wilson Dongmo Nentedemo

Objectif

L'objectif de cette évaluation est d'appliquer les techniques de prétraitement des données vues en cours sur un jeu de données réels provenant de Kaggle. Les étudiants doivent justifier le choix de chaque méthode utilisée.

Veuillez choisir une parmi les trois bases de données proposées ci-dessous :

Datasets à utiliser

1) Heart Disease UCI (Kaggle): UCI Heart Disease Data
2) Ames House Prices: House Prices - Advanced Regression Techniques | Kaggle
3) Credit Card Fraud Detection **:** Credit Card Fraud Detection

Partie 1 : Exploration des données

- Charger les datasets et afficher les 5 premières lignes.
- Afficher dimensions, types de colonnes et valeurs manquantes.
- Justifier pourquoi un prétraitement est nécessaire.

Partie 2 : Sélection des colonnes

- Identifier colonnes numériques, catégorielles (nominales et ordinales)
- Justifier ce choix.

Partie 3 : Imputation

- Appliquer SimpleImputer (en appliquant différentes stratégies).
- Appliquer KNNImputer.
- Comparer les résultats et justifier le choix.

Partie 4 : Encodage

- Appliquer LabelEncoder, OrdinalEncoder et OneHotEncoder.
- Justifier chaque méthode.

Partie 5 : Discrétisation

Appliquer la méthode KBinsDiscretizer sur une autre variable numérique du jeu de données choisi avec les paramètres suivants :

- n_bins = 3 ou n_bins = 4,
- une stratégie au choix : uniform, quantile ou kmeans,
- encode = 'ordinal' ou encode = 'onehot'.

Présenter :

- la répartition des données dans chaque bin,
- un tableau ou graphique comparatif avant/après transformation.

Partie 6 : Normalisation et Standardisation

- Appliquer MinMaxScaler, StandardScaler et RobustScaler.
- Comparer avant/après et justifier le choix.

Partie 7 : PowerTransformer

Appliquer la méthode PowerTransformer sur une variable (colonne) numérique afin de rendre sa distribution plus proche d'une distribution normale.

Présenter :

- la distribution avant et après transformation (histogramme)

Partie 8 : PolynomialFeatures

Appliquer la transformation PolynomialFeatures (degré 2 ou 3) sur au moins une variable numérique.

Indiquer :

- le degré choisi,
- les nouvelles variables générées,
- l'impact sur la dimension du jeu de données.

Justifier le choix du degré.

Partie 10 : Pipeline final

Construire un pipeline complet avec make_pipeline et ColumnTransformer intégrant :
- Imputation
- Encodage
- Normalisation / Standardisation
- Modèle de classification (au choix)

Livrables

1- Notebook Python (.ipynb)
2- Rapport (1 à 2 pages) avec justifications

Sujet : Prétraitement des données cardiaques (UCI Heart Disease)

1. Introduction et Exploration

L'analyse exploratoire du dataset *Heart Disease UCI* a révélé la nécessité impérative d'un prétraitement. Le jeu de données contient des valeurs manquantes significatives (notamment 611 manquants pour ca et 486 pour thal) et mélange des variables de types hétérogènes. Sans traitement, la plupart des algorithmes de Machine Learning (comme la régression logistique ou SVM) échoueraient à s'exécuter.

2. Stratégies de Nettoyage et d'Encodage

Sélection des colonnes

Nous avons séparé les variables en deux groupes distincts conformément aux principes du cours :

- **Variables Numériques** : Traitées mathématiquement.
- **Variables Catégorielles** : Divisées en **Nominales** et **Ordinales**. Cette distinction est cruciale car traiter une variable nominale comme ordinale introduirait un biais de hiérarchie inexistant.

Imputation

Pour gérer les valeurs manquantes, nous avons comparé deux approches :

1. **SimpleImputer :** Utilisation de la médiane pour les numériques (plus robuste aux valeurs extrêmes que la moyenne) et du mode pour les catégorielles.
2. **KNNImputer :** Utilise la distance entre les observations pour estimer la valeur manquante.

Choix final : Pour le pipeline final, nous privilégions SimpleImputer avec la médiane pour sa rapidité et sa robustesse, bien que KNN puisse offrir une précision supérieure sur de petits datasets fortement corrélés.

Encodage

- **OneHotEncoder :** Appliqué aux variables nominales. Cela évite que le modèle n'interprète le type de douleur thoracique 4 comme "supérieur" au type 1.
- **OrdinalEncoder :** Appliqué à slope car la pente du segment ST possède une progression clinique logique.

- **LabelEncoder :** Strictement réservé à la variable cible num pour ne pas polluer les variables explicatives.

3. Transformation des Données

Discrétisation

Nous avons appliqué KBinsDiscretizer sur l'âge (4 bins, stratégie 'quantile'). Cela permet de lisser le bruit et de gérer les non-linéarités le risque n'augmente pas forcément linéairement année par année, mais par tranche d'âge.

Normalisation

Trois méthodes ont été testées : MinMaxScaler, StandardScaler et RobustScaler. **Choix final : RobustScaler**. Les variables physiologiques comme le cholestérol (chol) contiennent des *outliers* naturels. StandardScaler (basé sur moyenne/écart-type) est trop sensible à ces extrêmes. RobustScaler, basé sur la médiane et l'intervalle interquartile, est la méthode recommandée dans ce contexte.

PolynomialFeatures

Nous avons généré des features de **degré 2** sur oldpeak.

- **Justification :** Un degré 2 permet de capturer des effets quadratiques (courbure) sans l'explosion combinatoire et le risque de sur-apprentissage (overfitting) qu'apporterait un degré 3 ou plus sur un dataset de cette taille (< 1000 échantillons).

4. Pipeline Final

Nous avons encapsulé toutes les étapes (imputation, encodage, scaling, modélisation) dans un Pipeline utilisant ColumnTransformer. Cela garantit que toutes les transformations appliquées au jeu d'entraînement sont appliquées identiquement au jeu de test, évitant ainsi toute fuite d'information (*data leakage*) et assurant la reproductibilité du modèle.