Jules MATHIEU

Projet Machine Learning :
Classification supervisée

1. Présentation des données

Nous nous intéressons à un jeu de donnée intitulé « Cirrhosis Patient Survival Prediction ». Il contient 17 variables et 418 instances. La classification s'effectue sur la variable 'Statut'. Cette variable prend 3 valeurs, D = Dead, C = Censored et CL = Censored due to liver transplantation.

La valeur Dead est attribuée aux patients qui sont décédés pendant la période de suivi de l'étude en raison de complications liées à la cirrhose du foie ou à d'autres causes.

La valeur Censored est utilisé dans les études de survie pour désigner les patients dont l'état final n'est pas connu à la fin de la période de suivi. Ils peuvent être perdus de vue, se retirer de l'étude ou être toujours en vie à la fin de la période de suivi.

Enfin, la valeur Censored due to liver transplantation (censuré en raison de la transplantation hépatique) : certains patients atteints de cirrhose du foie peuvent subir une transplantation hépatique pour traiter leur maladie durant la période d'étude. Lorsqu'un patient est transplanté, il est censuré dans l'analyse de survie à partir du moment de la transplantation.

L'objectif de ce modèle de classification supervisée est donc de prédire l'état futur d'un patient atteint d'une cirrhose du foie à partir de données personnelles. On retrouve : son âge, son genre et des données issues d'une prise de sang et d'une biopsie.

Le jeu de données contient des valeurs manquantes (marquées NA) et les classes ne sont pas équilibrées (C = 168, CL = 19 et D = 125). La classe CL contient très peu de données par rapport aux deux autres, donc dans notre cadre d'étude, nous faisons le choix de combiner les données de C et CL en une seule classe. L'objectif est alors de prévoir si le patient concerné va survire ou non.

2. Méthodes utilisées

Pour résoudre le problème des valeurs manquantes, nous stockons les variables manquantes dans un tableau. Les variables avec des valeurs manquantes ont été imputées par la moyenne. La méthode d'imputation a été choisi selon les préconisations de l'auteur.

On teste ensuite l'équilibre des données en sommant les itérations de chaque catégorie. Le déséquilibre sera traité plus tard.

Comme précisé dans la présentation du jeu de données par l'auteur, on utilise la méthode one-hot-encoding pour les variables catégorielles. On utilise pour cela la librairie *pandas* avec la fonction *get_dumnies().*

Pour trouver la meilleure méthode à suivre, on test la pertinence du Knn (K Nearest Neighbors) et la régression logistique.

On découpe les données grâce à la fonction *train_test_split()* de *sklearn*. Cela permet de créer un jeu de données Test (30% des données) et un jeu de données d'entrainement (70%).

On peut maintenant traiter les données déséquilibrées à l'aide d'un sur-échantillonnage (car le nombre de données est restreint).

Nous utilisons la fonction *RandomOverSampler()* de *imblearn*. Nous avons alors un nouveau jeu de données d'entrainement équilibré.

L'apprentissage est effectué grâce à la fonction *log_reg.fit()* et la prédiction grâce à la fonction *log_reg.predict() de sklearn*

On peut ensuite comparer la prédiction avec le test avec la fonction *metrics.accuracy_score() de sklearn.*

Enfin on vérifie le modèle par 3 méthodes, issues de sklearn:

- Cross-Validation grâce à *cross_val_score()*
- Matrice de confusion grâce à *confusion_matrix()*
- Rapport de classification grâce à *classification_report()*

3. Résultats

Après comparaison, la régression logistique a une précision pour les données de 0.89 contre 0.82 pour le Knn avec K = 5. Nous avons effectué le test avec d'autre k mais cela se révèle toujours inférieur à la régression logistique.

Après le sur-échantillonage, nous avons alors 125 C et 125 D.

La matrice de confusion donne par exemple :

- Vrai positif : Il y a 57 observations qui étaient C et qui ont été correctement prédites comme C.
- Faux positif : Il y a 5 observations qui étaient D, mais qui ont été incorrectement prédites comme C.
- Faux négatif : Il y a 5 observations qui étaient C mais qui ont été incorrectement prédites comme D.
- Vrai négatif : Il y a 27 observations qui étaient D et qui ont été correctement prédites comme D.

D'après la matrice de confusion le modèle semble bien prédire les données tests.

Le rapport de classification présente des valeurs supérieures à 0.85 pour la précision, le recall et le f1-score ce qui valide à nouveau la qualité du modèle.

Sur le graph de la cross validation on remarque une variabilité importante entre les 8 plis ce qui peut indiquer que la performance du modèle n'est pas très stable.

4. Manuel d'utilisation

Les packages nécessaires : *Matplotlib, pandas, imblearn* et *sklearn*

Le programme peut être exécuté tel quel (à condition de changer l'emplacement du jeu de données au début du programme principal.)