Visualisation de données
- L3 Math, ISD, TP 6 -

Ce TP vise à initier aux différentes phases de la classification supervisée : préparation des données, mise en oeuvre des méthodes de classification supervisée, entraînement des modèles, validation des modèles, évaluation des modèles et réalisation de graphiques.

Description des données

Suite à la collecte exhaustive de données sur le naufrage du RMS Titanic, notre équipe d'analystes de données se trouve confrontée à une tâche cruciale : déterminer les facteurs qui ont influencé les chances de survie des passagers à bord. Avec un ensemble de données comprenant des informations telles que le sexe, l'âge, la classe sociale, le port d'embarquement et d'autres variables pertinentes, nous sommes chargés d'identifier les tendances et les corrélations qui peuvent aider à comprendre pourquoi certains passagers ont eu plus de chances de survivre que d'autres.

Les données, disponibles en utilisant le package seaborn, contiennent pour un ensemble de 891 personnes les variables :

- PassengerId : Identifiant unique de chaque passager
- Survived : Indique si le passager a survécu (1) ou non (0)
- Pclass : Classe du billet (1ère, 2ème ou 3ème)
- Name : Nom du passager
- Sexe : Sexe du passager (Homme ou Femme)
- Age : Âge du passager (en années)
- SibSp : Nombre de frères et sœurs ou conjoints à bord
- Parch : Nombre de parents ou enfants à bord
- Ticket : Numéro du billet
- Fare : Prix du billet
- Cabin : Numéro de la cabine (si applicable)
- Embarked : Port d'embarquement (Southampton, Cherbourg ou Queenstown)

Partie 1 : Utilisation de l'algorithme des k plus proches voisins

1. Chargez les données du Titanic en utilisant la fonction *load_dataset* du package **seaborn**. Affichez les informations du jeu de données en utilisant les fonctions *head* et *info*.

2. Transformez les variables *sex* et *who* en catégories. Créez une nouvelle variable *family* comptant le nombre de liens familiaux d'un individu en utilisant les variables *sibsp* et *parch*. Supprimez les colonnes "embarked", "sibsp", "parch", "adult_male", "deck", "alive", "class".

3. Effectuez une visualisation des données deux à deux en utilisant la fonction *pairplot* de **seaborn**, en adaptant la couleur pour représenter la survie des individus. Remarque-t-on certaines combinaisons de variables permettant de séparer les 2 classes ?

4. Transformez les données *pclass*, *sex*, *who* et *embark_town* en utilisant la fonction *pd.get_dummies* pour faire une transformation "One-Hot-Encoding" des variables. Puis, supprimez les données manquantes et affichez le jeu de données avec *head*.

5. Séparez le jeu de données en une matrice X et un vecteur y puis normalisez les données en utilisant la fonction *StandardScaler* de **scikit learn**.

6. En vous aidant du code vu en cours, utilisez la fonction *train_test_split* de **scikit learn** afin de garder 15% du jeu de données pour le test.

7. En vous aidant du code vu en cours, utilisez la fonction *train_test_split* de **scikit learn** afin d'utiliser 20% du jeu de données pour la validation. Par la suite, nous cherchons à trouver le meilleur paramètre k (le nombre de voisins) pour la méthode des k plus proches voisins (*KNeighborsClassifier*). Pour cela, nous entraînons les différents modèles sur les données d'entraînement avec la fonction *fit* et nous calculons l'exactitude (accuracy) avec la fonction *score*. Enfin, nous souhaitons réaliser la même figure que la figure 1. Quel est le nombre de voisins optimal trouvé ?

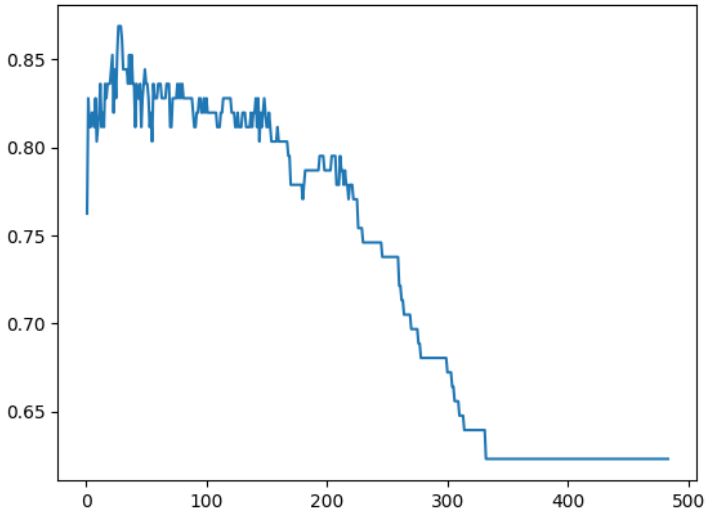

FIGURE 1 – Recherche du nombre de voisins optimal par Train-Val-Test Split.

8. Nous souhaitons maintenant calculer la matrice de confusion comme représenté sur la figure 2. Pour cela, créez une fonction prenant en entrée le modèle, ainsi que l'ensemble de données utilisé pour l'évaluation, c'est à dire X, y. Utilisez la fonction définie pour représenter la matrice de confusion du modèle des k plus proches voisins avec le nombre de k trouvé à la question précédente sur les données test. Comment peut-on l'interpréter ? Quelle est l'exactitude, la précision et le rappel sur les données test ?

9. En vous aidant du code vu en cours, réalisez les mêmes étapes que la question 7 en utilisant cette fois une validation croisée 5-Fold.

10. Utilisez la fonction permettant d'obtenir la matrice de confusion avec le nouveau k trouvé à la question précédente et les données test. Quelle est l'exactitude, la précision et le rappel sur les données test ?

11. Utilisez la fonction *GridSearchCV* de **scikit learn** avec 5-Fold en testant différentes valeurs de k , les distances L1, L2 ainsi que différentes manières de calculer les poids.

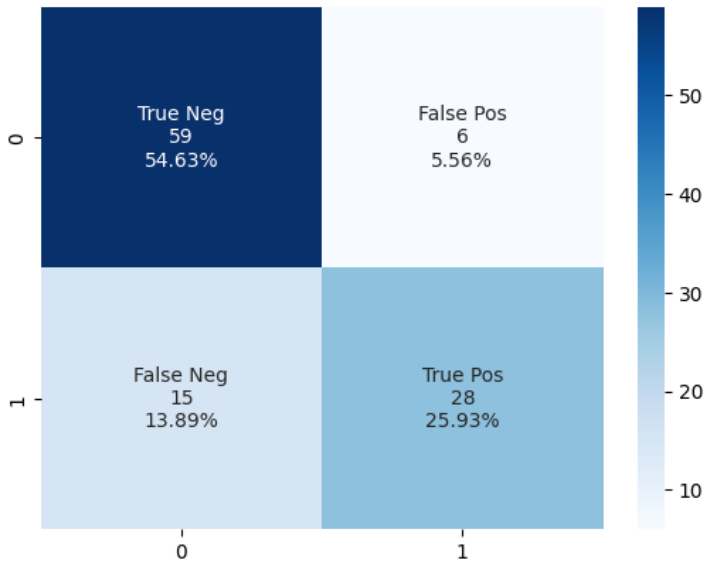

FIGURE 2 – Recherche du nombre de voisins optimal par Train-Val-Test Split.

12. Utilisez la fonction permettant d'obtenir la matrice de confusion avec le nouveau k trouvé à la question précédente et les données test. Quelle est l'exactitude, la précision et le rappel sur les données test ? Quel est le meilleur modèle sur les 3 utilisés vis à vis de l'exactitude, de la précision et du rappel ?

Partie 2 : Utilisation d'un arbre de décision

1. Nous allons utiliser des données différentes en entrée. En effet, un arbre de décision peut prendre en entrée des données catégorielles directement, il n'est donc pas nécessaire de faire autant de prétraitement. Repartez du dataframe obtenu à la fin de la question 2 de la partie 1 et utilisez la fonction *LabelEncoder* de **scikit learn** pour encoder les variables *sex*, *who* et *embark_town*. Enfin, supprimez les données manquantes.

2. Séparez le nouveau jeu de données en une matrice X et un vecteur y. Pour les arbres de décision, il n'y pas besoin de standardiser les données.

3. En vous aidant des exemples vus en cours, cherchez le paramètre **complexity parameter** optimal par validation train-test-split en utilisant l'exactitude. Quelle est la profondeur de l'arbre obtenu, son nombre de feuilles ?

4. Utilisez la fonction permettant d'obtenir la matrice de confusion avec l'arbre de décision ainsi que le α trouvé à la question précédente et les données test. Quelle est l'exactitude, la précision et le rappel sur les données test ? Est-ce que le modèle est meilleur que la meilleure méthode des k plus proches voisins ?

5. Affichez la répartition des éléments de la variable cible *survived* avec la fonction *distplot* de **seaborn**. Est-ce que la classe est équilibrée ? Que proposez-vous ?

6. Cherchez le paramètre **complexity parameter** optimal par validation train-test-split en utilisant le *rappel* comme métrique. Quelle est la profondeur de l'arbre obtenu, son nombre

de feuilles ? Affichez la matrice de confusion et les différentes métriques. Est-ce que le rappel est plus important que le modèle précédent ?

7. Faîtes exactement la même chose que la question 6 mais en utilisant cette fois-ci la précision comme métrique. Est-ce que la précision est plus importante que celle obtenue pour le modèle précédent ?

8. Il est compliqué d'utiliser le rappel ou la précision pour trouver les meilleurs hyperparamètres car la maximisation de l'un va diminuer l'autre. Il existe différentes manières de résoudre le problème. Une première est l'utilisation de la métrique *f1-score* qui est une formule combinant les 2 métriques. Une deuxième est l'utilisation de l'aire sous la courbe ROC (AUC). Réalisez la courbe ROC de l'arbre obtenu à la question précédente en utilisant les fonctions *RocCurveDisplay*, *roc_curve* afin d'obtenir la figure 3. Quelle serait la courbe ROC d'un modèle parfait et d'un modèle aléatoire ?

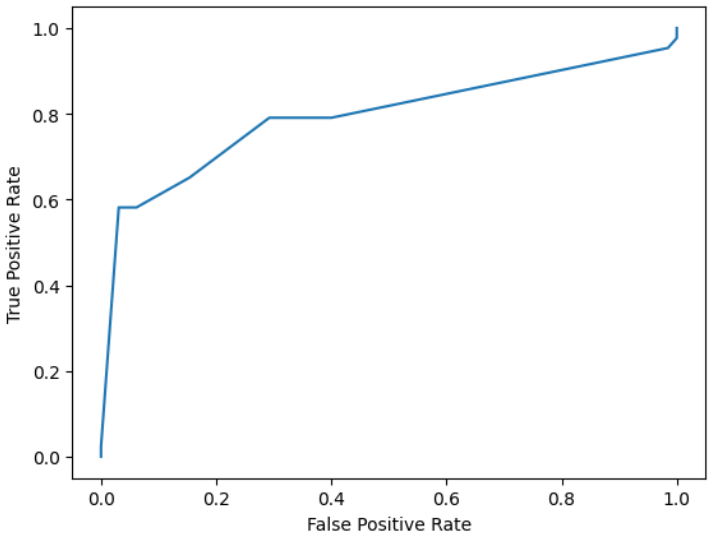

FIGURE 3 – Courbe ROC d'un arbre de décision.

9. Réalisez une validation croisée avec 5-Fold en utilisant la métrique AUC ("roc_auc" dans scikit). Affichez la matrice de confusion ainsi que sa courbe ROC et calculez les métriques. Est-ce que le modèle a une meilleure précision, rappel ou exactitude que les modèles précédents ?

Partie 3 : Comparaison entre la méthode des k-plus proches voisins, l'arbre de décision et la forêt aléatoire

1. A l'aide de la fonction *GridSearchCV* de **scikit learn**, essayez de trouver les meilleurs hyperparamètres (distance, nombre de voisins, ...) avec la métrique AUC pour la méthode des k plus proches voisins.

2. A l'aide de la fonction *GridSearchCV* de **scikit learn**, essayez de trouver les meilleurs hyperparamètres (complexity parameter, ...) avec la métrique AUC pour un arbre de décision.

3. A l'aide de la fonction *GridSearchCV* de **scikit learn**, essayez de trouver les meilleurs hyperparamètres (nombre d'estimateurs, nombre minimum d'invidu par feuille) avec la métrique

AUC pour une forêt aléatoire. Vous pouvez vous aider des exemples vu dans le cours.

4. En utilisant la fonction *permutation_importance* de **scikit learn**, calculez l'importance de chaque variable en utilisant la forêt aléatoire avec les paramètres optimaux de la question précédente. Refaites le graphique de la figure 4. Quelles sont les variables qui ont le plus d'importance et au contraire les variables qui ont très peu d'importance ?

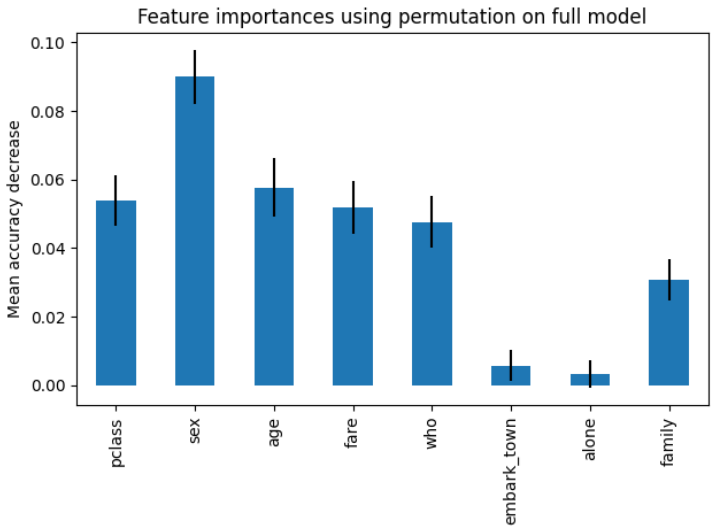

FIGURE 4 – Importance des variables.

5. Affichez les 3 courbes ROC des modèles obtenus ainsi que la courbe ROC d'un modèle aléatoire sur le même graphique. Que peut-on conclure ? Quel est le meilleur modèle ?

Partie 4 : Prédire la gravité d'un accident en fonction de différentes variables

Pour chaque accident corporel (soit un accident survenu sur une voie ouverte à la circulation publique, impliquant au moins un véhicule et ayant fait au moins une victime ayant nécessité des soins), des saisies d'information décrivant l'accident sont effectuées par l'unité des forces de l'ordre (police, gendarmerie, etc.) qui est intervenue sur le lieu de l'accident. Ces saisies sont rassemblées dans une fiche intitulée bulletin d'analyse des accidents corporels. L'ensemble de ces fiches constitue le fichier national des accidents corporels de la circulation dit « Fichier BAAC » administré par l'Observatoire national interministériel de la sécurité routière "ONISR.

Il est possible de télécharger un ensemble globale des accidents avec le lien suivant. L'idée est de tester différents modèles permettant d'expliquer la gravité des accidents. Vous pouvez visiter le lien suivant suivant afin d'obtenir plus d'informations sur la base de données.