Analyse de données - TP 5

ISEP – 24 Novembre 2020

Instructions : Déposez votre rapport au format pdf sur Moodle

Bibliothèques

Ce TP nécessite l'installation des librairies suivantes : pandas, matplotlib, numpy, scipy, sklearn et pydotplus.

A Données du titanic

A.1 Analyse préliminaire

1. Ouvrez les fichiers "titanic_train.csv" et "titanic_test.csv"
   - Pourquoi y a-t-il 2 fichiers ? Comparez-les. Sont-ils identiques ?
   - A l'aide de la fonction *head()* et de la propriété *dtype*, décrivez les différents attributs du jeu de données titanic_train.
   - En utilisant les fonctions *isnull()* et *sum()* que pouvez-vous dire des données manquantes des différents attributs ?
2. A partir du fichier titanic_train, affichez les informations suivantes :
   - Affichez un histogramme pour l'âge des passagers. Justifiez votre gestion des valeurs manquantes.
   - Affichez le pourcentage de personnes décédées.
   - Affichez le pourcentage de personnes en 1ère, 2nde et 3ème classe. Affichez un diagramme en camembert.
   - Affichez les pourcentages d'hommes et de femmes à bord et le taux de survie dans les 2 catégories.
   - En sachant que l'âge adulte était à 18 ans, estimez les pourcentages de survivants chez les enfants et les adultes ? Justifiez votre gestion des données manquantes.
   - Estimez les taux de survie pour toutes les combinaisons adulte/enfant, homme/femme.
   - Affichez les taux de survie en 1ère, 2nd et 3ème classe. Expliquez à nouveau votre gestion des données manquantes.
3. A partir des réponses précédentes, que pouvez-vous dire sur le respect de la politiques "les femmes et les enfants d'abord" lors du naufrage du Titanic ?
4. Affichez une matrice de correlation des différents attributs. Commentez sur d'éventuelles correlations remarquables.

5. Dans les 2 jeux des données (training et test), créez un attribut "Fare2" dont la valeur est 1 si le prix du ticket était inférieur à 10, 2 en dessous de 20, 3 en dessous de 30, et 4 sinon. En utilisant les bons outils mathématiques, évaluez la force du lien entre votre nouvel attribut "Fare2" et la classe dans laquelle les passagers voyageaient. Affichez ensuite le taux de survivants pour chaque valeur de "Fare2". Commentez et concluez sur l'influence du prix des tickets et les chances de survie.

A.2 Prédiction avec Naive Bayes

1. Si vous ne l'aviez pas déjà fait, rajoutez un attribut "Child" dans les deux jeux de données avec un valeur de "1" pour les passagers de moins de 18 ans, et 0 sinon. N'oubliez pas de préciser votre politique pour les données manquantes en indiquant les avantages et les inconvénients de votre choix.
2. Utilisez l'algorithme Naive Bayes pour prédire les taux de survie des passagers en fonction de leur sexe et de s'ils étaient adultes ou non. Confirmez vos résultats sur les jeux de données d'apprentissage et de test. Expliquez et commentez les différentes métriques de la fonction *classification_report*. Un exemple de code est disponible ci-dessous.

```
# needed imports
from sklearn import metrics
from sklearn.naive_bayes import GaussianNB

#classifier choice
gnbModel=GaussianNB()
#choice of the training set, considered attributes and variable to predict
gnbModel.fit(df_train[['Child','Sex_cat']], df_train['Survived'])
            #Sex_cat stands for transformed categorical array

#expected results are stored in a separate vector
expected =df_train['Survived']

#predictions on the training set
predicted = gnbModel.predict(df_train[['Child','Sex_cat']])
#displaying relevant metrics
print(metrics.classification_report(expected, predicted))

#same when applying the model to the test set
expected =df_test['Survived']
predicted = gnbModel.predict(df_test[['Child','Sex_cat']])
print(metrics.classification_report(expected, predicted))
```

3. Appliquez à nouveau l'algorithme Naive Bayes avec le sexe, adulte ou non, et l'attribut Fare2. Commentez les résultats.

A.3 Prédiction à partir d'un arbre de décision

1. Entrainez un arbre de décision pour prédire si les passagers vont survivre ou non. Pour celà, utilisez tous les attributs qui vous semblent pertinents, y compris ceux que vous avez créés. Vous pouvez essayer plusieurs combinaisons. Commentez les résultats.

Remarque 1 : Tous les classifiers de sklearn fonctionnent de la même façon et ont une syntaxe similaire. La fonction *fit()* permet d'entrainer l'algorithme sur le jeu de données d'entrainement. La fonction *predict()* permet de calculer la classe la plus probable à partir des attributs de nouvelles données. Le code ci-dessous est un exemple de comment utiliser un arbre de décision et afficher le résultat.

Remarque 2 : Si vous rencontrez des difficultés avec la librairie *graphviz*. Utilisez plutôt la fonction *plot_tree* de sklearn.

```
# needed imports
import pydotplus
import collections
from IPython.display import Image
from sklearn.tree import export_graphviz
from sklearn import tree

clf =tree.DecisionTreeClassifier()
clf=clf.fit(df_train[['Pclass','Fare2','Child','Sex_cat']],df_train['Survived'])        #Sex_cat stands for transformed categorical array

dot_data = tree.export_graphviz(clf, out_file=None, feature_names=data_feature_names,class_names=True, filled=True,rounded=True, precision=0)
graph = pydotplus.graph_from_dot_data(dot_data)
Image(graph.create_png())        # to plot a tree
Image(graph.write_png('./filename.png'))        # to save the plot
```