université angers

Licence 3 Informatique
Python

TP3 - Pandas et les K plus proches voisins

Pour l'ensemble des TP vous pourrez utiliser au choix l'EDI PyCharm ou le docker Jupyter notebook. Pour lancer les éditeurs :

- Ouvrir un terminal,
- `pycharm-community` ou `_jupyter_notebook`
- Créer un nouveau projet pour chaque TP et un fichier pour chaque exercice.

ATTENTION : Chaque TP fera l'objet d'un dépôt sur Moodle qui sera noté à chaque séance.
Date limite de dépôt pour le TP3 **jeudi 20 Février 2020, 23h59.**

Une documentation Python est disponible à cette adresse : https://docs.python.org/3/tutorial/
Une documentation NumPy est disponible à cette adresse : https://docs.scipy.org/doc/numpy/reference/
Une documentation MatPlotLib : https://matplotlib.org/3.1.1/api/index.html
Une documentation Pandas : https://pandas.pydata.org/pandas-docs/stable/reference/index.html

Exercice 1 -Application des K plus proches voisins sur les Iris (la fleur)

1. Charger le fichier "iris.csv" dans un dataframe
2. Afficher les 10 premières valeurs du dataframe.
3. Les données représentées sont la longueur des pétales, la largeur des pétales et l'espèce d'iris. On identifie l'espèce par 0 = "iris setosa", 1 = "iris virginica", 2 = "iris versicolor". Afficher dans un graphique, un nuage de points représentant la longueur en abscisse et la largeur en ordonnée ainsi qu'une couleur différent en fonction de l'espèce.
4. Écrire une fonction euclidian_distance(v1,v2) qui renvoie la distance euclidienne entre deux vecteurs.
5. Afficher la distance euclidienne entre le premier iris du dataframe et un iris choisi au hasard (par exemple longueur = 3 et largeur = 1 ou un autre iris du dataframe)
6. Écrire une fonction neighbors qui prend en paramètres le dataframe, un iris à tester et une valeur k et renvoie les k plus proches voisins de l'iris de test.
7. Tester le bon fonctionnement avec les valeurs suivantes :

```
k = 3
x_test1 = np.array([3,0.6])
x_test2 = np.array([5,1.6])
x_test3 = np.array([1,0.3])
```

Le résultat obtenu devrait être :

```
    label  distance
57      1       0.5
93      1       0.5
98      1       0.5
    label  distance
83      1       0.1
77      1       0.1
119     2       0.1
    label  distance
22      0  0.100000
```

```
14      0  0.223607
35      0  0.223607
```

8. Écrire une fonction prediction qui à partir d'un voisinage renvoie l'espèce à laquelle doit appartenir l'iris testé. Nous choisirons l'espèce qui apparaît le plus dans le voisinage (si toutes les espèces apparaissent autant de fois on choisira une des trois espèce "au hasard").
9. Á partir des données sur les iris, prenez un échantillon aléatoire de 60% qui servira d'entraînement et utiliser les 40% restant pour calculer un taux de réussite pour la prédiction. La prédiction est réussie si un iris est classée dans la bonne espèce. (Pensez à utiliser la méthode `sample` des dataframe pour créer les deux échantillons).
10. Afficher le taux de réussite du classement.
11. Afficher le taux de réussite pour chaque espèce.
12. Afficher graphiquement le classement des espèces de l'échantillon de test en rapport avec l'échantillon d'entraînement.
13. Tester avec différentes valeurs de k.

Exercice 2 -Application des K plus proches voisins sur le diabète

1. Charger le fichier "diabetes.csv" dans un dataframe
2. Afficher les 10 premières valeurs du dataframe.
3. Les données représentées sont le nombre de grossesses, le taux de Glucose, la pression artérielle, l'épaisseur de la peau au niveau du triceps, le taux d'insuline, l'BMI (indice de masse corporelle (IMC)), le diabetes pedigree function, et l'âge. On identifie un individu comme diabtéique si Outcome = 1 et non diabétique si Outcome = 0. Certaines de ces données sont incomplètes et représentées par des NaN, c'est le cas pour les colonnes Glucose, BMI, BloodPressure, SkinThickness, et Insulin.
4. Écrire une fonction euclidian_distance(v1,v2) qui renvoie la distance euclidienne entre deux vecteurs.
5. Afficher la distance euclidienne entre le premier individu du dataframe et un second individu choisi au hasard dans le dataframe.
6. Écrire une fonction neighbors qui prend en paramètres le dataframe, un individu à tester et une valeur k et renvoie les k plus proches voisins de l'individu de test.
7. Écrire une fonction prediction qui à partir d'un voisinage renvoie si l'indivu est diabétique ou non.
8. Á partir des données sur les individus diabétiques, prenez un échantillon aléatoire de 60% qui servira d'entraînement et utiliser les 40% restant pour calculer un taux de réussite pour la prédiction. La prédiction est réussie si le diabète d'un individu est détecté correctement.
9. Afficher le taux de réussite du classement.
10. Afficher le nombre de faux positif.
11. Afficher le nombre de cas non détectés.
12. Tester différentes valeurs de k.
13. Tester avec moins de critères (par exemple juste en prenant en compte le glucose et la pression artérielle). Essayez de trouver des critères qui optimisent le taux de réussite. (Supprimer les critères qui ne sont pas pertinents).