CS 4661: Introduction to Data Science
Dr. M. Pourhomayoun
Homework4
Due Date: Fri, Nov 22

Write and submit your python codes in "Jupyter Notebook". **Please upload 2 separate ipynb files (One for Question1, and One for Question2).**

Question1: Cancer Diagnosis Using Machine Learning

In this homework, we work with a real dataset from UCI Database.

a- Read the dataset file "Cancer.csv" (from github using the following command), and assign it to a Pandas DataFrame:
df = pd.read_csv("https://github.com/mpourhoma/CS4661/raw/master/Cancer.csv")
Check out the dataset. As you see, the dataset includes 9 numerical features. The last column is the binary label ("1" means it is a malignant cancer, "0" means it is a benign tumor). You will use all 9 features in this homework.

b- Use sklearn functions to split the dataset into testing and training sets with the following parameters: **test_size=0.35, random_state=3**.

c- Use "Decision Tree Classifier" to predict Cancer based on the training/testing datasets that you built in part (b). Then, calculate and report the accuracy of your classifier. Use this command to define your tree:
my_DecisionTree = DecisionTreeClassifier(random_state=3).

d- Now, we want to perform a new Ensemble Learning method called "**Bagging**" based on **Voting** on 19 decision tree classifiers.
Note: you should write your own code to perform Bagging (don't use scikit-learn functions for Bagging!)
To do so, you need to perform bootstrapping first. You can write a "for" loop with loop variable i = 0…18.
In each iteration of the loop, you have to:

1. make a bootstarp sample of the original "Training" Dataset (build in part(b)) with the size of **bootstarp_size = 0.8*(Size of the original dataset).** You can use the following command to generate a random bootstrap dataset ("*i*" is the variable of the loop, so the random_state changes in each iteration):
resample(X_train, n_samples = bootstarp_size , random_state=*i* , replace = True)
2. Define and train a new base decision tree classifier on this dataset in each iteration:
Base_DecisionTree = DecisionTreeClassifier(random_state=3).

3. Perform prediction using "this base classifier" on the original "Testing" Dataset **X_test** (build in part(b)), and save the prediction results for all testing samples.

After finishing the "for" loop, you should have 19 different predictions for EACH sample in your testing set. Then, Perform Voting to make the final decision on each data sample based on the votes of all 19 classifiers.
Finally, calculate and report the final accuracy of your Bagging (Voting) method.
Note: You do NOT need to calculate the accuracy of each one of the base classifiers in each round of the loop! You have to just perform Voting to make the final decision on each data sample, and then calculate the accuracy on the final results.

e- Use scikit-learn "Random Forest" classifier to predict Cancer based on the training/testing datasets that you built in part (b). Then, calculate and report the accuracy of your classifier. Use this command to import and define your classifier:
from sklearn.ensemble import RandomForestClassifier
my_RandomForest =
RandomForestClassifier(n_estimators = 19, bootstrap = True, random_state=3)

Similar to previous syntax, use **my_RandomForest.fit** for training your random forest classifier and **my_RandomForest.predict** for prediction.

Question2: predict the probability of Heart Disease

a- In this question, we work with a simplified version of Heart dataset. Read the dataset file "Heart_short.csv" from github, and assign it to a Pandas DataFrame:
df =
pd.read_csv("https://github.com/mpourhoma/CS4661/raw/master/Heart_short.csv")

b- Generate the feature matrix and label vector (AHD). Then, normalize (**scale**) the features.

c- Split the dataset into testing and training sets with the following parameters: test_size=0.25, random_state=3.

d- Use Logistic Regression Classifier to **predict** Heart Disease occurrence based on the training/testing datasets that you built in part(c). Then, compute and report the **Accuracy**.

Now, Use Logistic Regression Classifier to **predict the probability** of Heart Disease based on the training/testing datasets that you built in part (c) (you have to use "my_logreg.**predict_proba**" method rather than "my_logreg.**predict**"). Then, Plot the **Roc Curve** for this classifier, and also Compute the **AUC** (Area Under Curve for ROC).