Homework1, Due Date: Fri, Mon 24

Instructor: Dr. Mohammad Pourhomayoun

Cancer Diagnosis Using Machine Learning

Write and submit your python codes in "Jupyter Notebook" to perform the following tasks. Make sure to provide proper descriptions as MarkDown for each section of your code.

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

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/CS4662/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=0.3, random_state=2**.

c- Use "Decision Tree Classifier" to predict Cancer based on the training/testing datasets that you built in part (h). Then, calculate and report the accuracy and AUC of your classifier. Later in part (g), you will plot the ROC curve as well. Use this command to define your tree: **my_DecisionTree = DecisionTreeClassifier(random_state=2).**

d- Now, we want to perform "Bagging" based on 29 "base 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:

- make a bootstarp sample of the original "Training" Dataset (build in part(b)) with 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)
- Define and train a new base decision tree classifier on this dataset in each iteration:
Base_DecisionTree = DecisionTreeClassifier(random_state=2).
- Test "this base classifier" on the original "Testing" Dataset build in part(b), and save the prediction results for all testing samples.

Then, Perform Voting to make the final decision on each data sample based on the votes of all 29 classifiers. Finally, calculate and report the accuracy and AUC of your Bagging method.

NOTE: You need to calculate the probability of “malignant cancer” to be able to find AUC and plot the ROC curve. As mentioned in the class, you can consider the average (mean) of the votes as the probability for each sample.

e- Use scikit-learn “Adaboost” classifier to predict Cancer based on the training/testing datasets that you built in part (b). Then, calculate and report the accuracy and AUC of your classifier. Use this command to import and define your classifier:

```
from sklearn.ensemble import AdaBoostClassifier
my_AdaBoost = AdaBoostClassifier(n_estimators = 29,random_state=2)
```

f- In this section, we use an extremely popular Boosting algorithm called “XGBoost”. This algorithm is not included in sklearn, so you need to install the XGBoost library. Please see this for more infor: https://xgboost.readthedocs.io/en/latest/build.html
Mac users can easily install it with “*pip install xgboost*”.
Repeat part (e) with XGBoost. Use this command to import and define your classifier:

```
from xgboost import XGBClassifier
my_XGBoost = XGBClassifier(n_estimators = 29,random_state=2)
```

g- 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 and AUC of your classifier. Use this command to import and define your classifier:

```
from sklearn.ensemble import RandomForestClassifier
my_RandomForest =
RandomForestClassifier(n_estimators = 29, bootstrap = True, random_state=2)
```

h- Now, plot the ROC curves of your algorithms in parts (c), (d), (e), (f), (g) in a single plane with different colors along with the name of each method. Show the AUCs on the graph as well.

- Which algorithm is the best w.r.t the AUC value?
- Which algorithm is the best w.r.t the Accuracy value?
- Which algorithm is the best when we want a False Positive Rate of %7?