AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering
Spring 2019

Course No: CSE 4108

Course Title: Artificial Intelligence Lab

Term Project No: 03

Topic: Decision Tree Regression and Adaptive Boosting Regression

Date of Submission: 6 October, 2019

Submitted To: Dr. S.M. Abdullah Al-Mamun
Professor, Department of CSE, AUST.

Md. Siam Ansary
Part time Faculty, Department of CSE, AUST.

Submitted By,

Name: Iffatun Nessa
Student ID: 160104031
Group: A2

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Input:

For the input dataset I used WhiteWineQuality.csv file.

Associated Tasks: Classification, Regression

Characteristics: Multivariate

Number of Instances: 4898

Number of attributes: 12

Attribute Information: Input variables- 1. fixed acidity, 2. volatile acidity, 3. citric acid, 4. residual sugar, 5. Chlorides, 6. free sulfur dioxide, 7. total sulfur dioxide, 8. Density, 9. pH, 10. Sulphates, 11. alcohol

Output variable 12. quality (score between 0 and 10)

Major steps:

1. import the dataset and split it into 11:1 ration using dataframe iloc.

2. insert the values in x and y.

3. split the dataset in 2/3 for test and training set.

4. fit the decision tree regression and AdaBoost regression and find the prediction using built in librarires.

5. find out the mean absolute error, r2 error, mean squared error and max error.

6. plot the error values in subplot feature of matlab.

Implementation in python:

```
# Import the necessary libraries

import numpy

import matplotlib.pyplot as plot

import pandas

from sklearn import metrics
from sklearn.ensemble import AdaBoostRegressor
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.tree import DecisionTreeRegressor
# Import the dataset
dataset = pandas.read_csv('WhiteWineQuality.csv')
x = dataset.iloc[:,0:10]
y = dataset.iloc[:,11]
print(x)
print(y)
# Split the dataset into the training set and test set
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size = 2/3)
# Creating a DecisionTreeRegressor on our trainging set.
regressor = DecisionTreeRegressor()
regressor.fit(xTrain, yTrain)
# Predicting the test set results
yPrediction = regressor.predict(xTest)
# Creating a AdaptiveBoostingRegressor on our training set
abr = AdaBoostRegressor()
abr.fit(xTrain, yTrain.values.ravel())
#Prediction
yPrediction2 = abr.predict(xTest)
#calculation errors
mabs1=metrics.mean_absolute_error(yTest, yPrediction)
r21=metrics.r2_score(yTest, yPrediction)
s1=metrics.mean_squared_error(yTest, yPrediction)
max1=metrics.max_error(yTest, yPrediction)

mabs2=metrics.mean_absolute_error(yTest, yPrediction2)
r22=metrics.r2_score(yTest, yPrediction2)
s2=metrics.mean_squared_error(yTest, yPrediction2)
max2=metrics.max_error(yTest, yPrediction2)
```

```
# Displaying errors for DecisionTree
print('Mean Absolute Error for DecisionTree:', mabs1)
print('r2:', r21)
print('mean squared error for DecisionTree:', s1)
print('mean squared error for DecisionTree:', max1)
# Displaying errors Adabooster
print('Mean Absolute Error for Adabooster:', mabs2)
print('r2 for Adabooster:', r22)
print('mean squared error for Adabooster:', s2)
print('mean Max error for Adabooster:', max2)

#Cross Validation
print("Cross Validation score for Decision Tree:",cross_val_score(regressor, x,y.values.ravel(),
                                             cv = 5, scoring = 'neg_median_absolute_error'))
print("Cross Validation score for Adaptive Boosting:",cross_val_score(abr, x, y.values.ravel(),
                                                cv = 5, scoring = 'neg_median_absolute_error'))
# Figure and comparison show
plot.figure(figsize= (10, 2))
#mean absolute
names = ['Decision Tree', 'Adaptive Boosting']
values = [mabs1,mabs2]
plot.subplot(121).set_ylabel('Mean Absolute Error')
plot.bar(names, values)
plot.show()
#r2
values = [r21,r22]
plot.subplot(121).set_ylabel('R2 score')
plot.bar(names, values)
plot.show()
#Mean Squared
values = [s1,s2]
```

plot.subplot(121).set_ylabel('Mean squared error')

plot.bar(names, values)

plot.show()

#max error

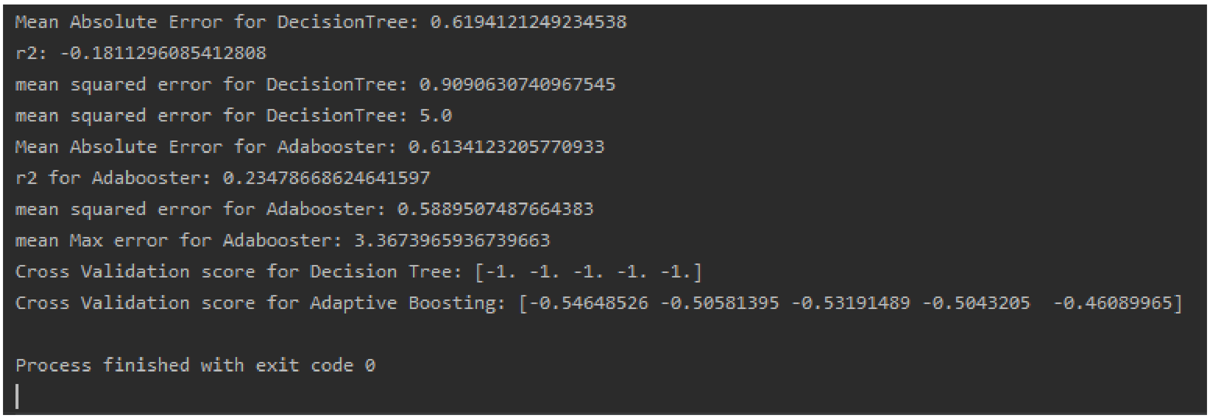
values = [max1,max2]

plot.subplot(121).set_ylabel('Max error')

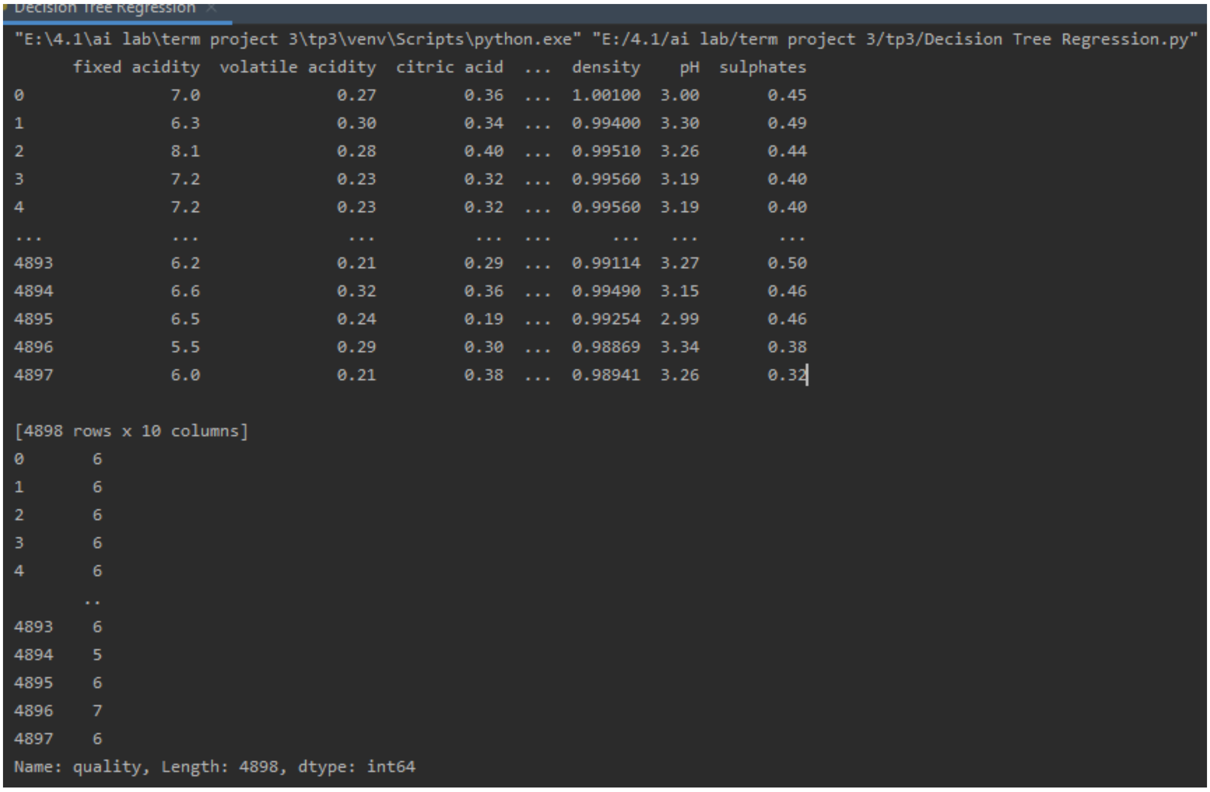
plot.bar(names, values)

plot.show()

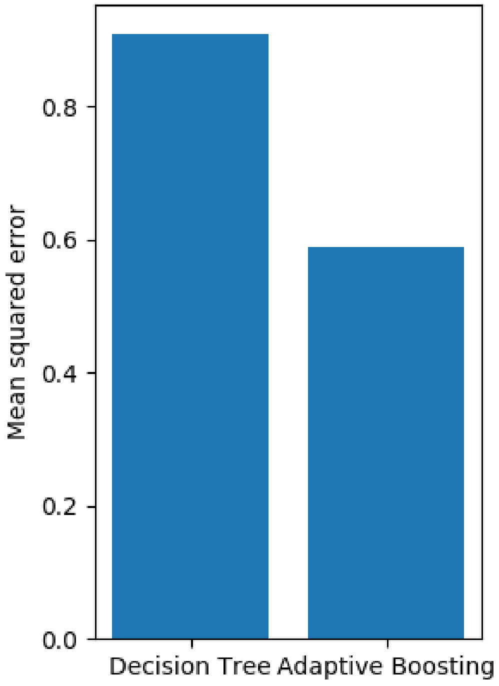
Output:

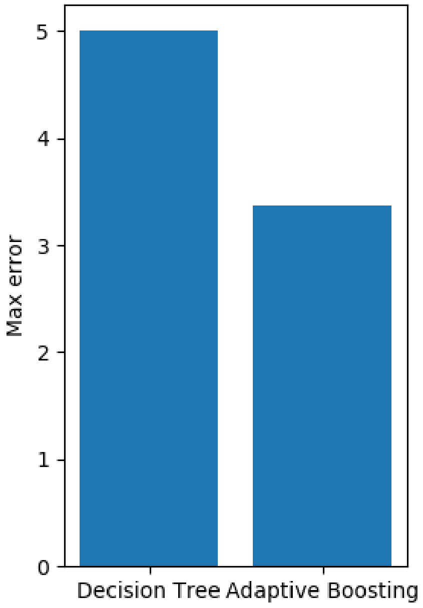


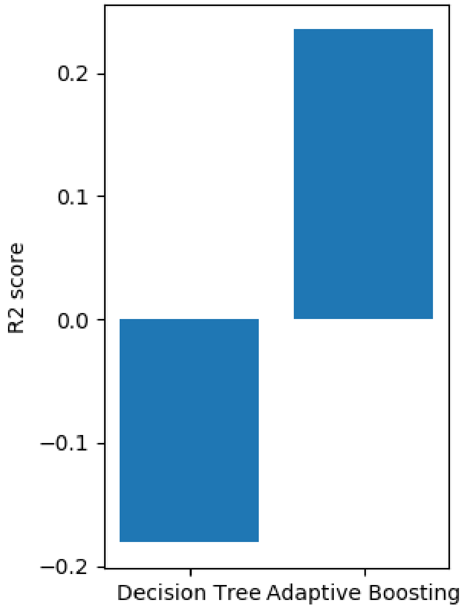
```
Mean Absolute Error for DecisionTree: 0.6194121249234538
r2: -0.1811296085412808
mean squared error for DecisionTree: 0.9090630740967545
mean squared error for DecisionTree: 5.0
Mean Absolute Error for Adabooster: 0.6134123205770933
r2 for Adabooster: 0.23478668624641597
mean squared error for Adabooster: 0.5889507487664383
mean Max error for Adabooster: 3.3673965936739663
Cross Validation score for Decision Tree: [-1. -1. -1. -1. -1.]
Cross Validation score for Adaptive Boosting: [-0.54648526 -0.50581395 -0.53191489 -0.5043205  -0.46089965]

Process finished with exit code 0
```

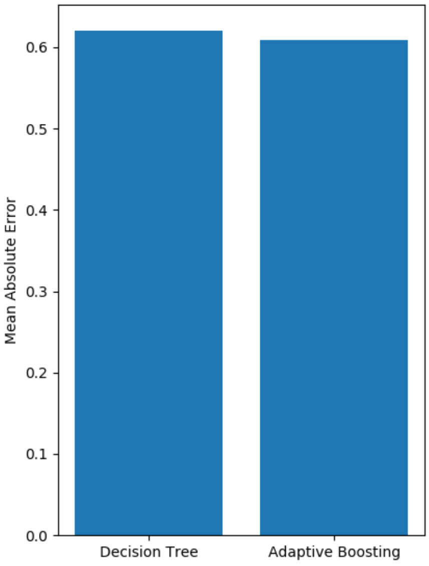


```
"E:\4.1\ai lab\term project 3\tp3\venv\Scripts\python.exe" "E:/4.1/ai lab/term project 3/tp3/Decision Tree Regression.py"
      fixed acidity  volatile acidity  citric acid  ...  density    pH  sulphates
0               7.0              0.27         0.36  ...  1.00100  3.00       0.45
1               6.3              0.30         0.34  ...  0.99400  3.30       0.49
2               8.1              0.28         0.40  ...  0.99510  3.26       0.44
3               7.2              0.23         0.32  ...  0.99560  3.19       0.40
4               7.2              0.23         0.32  ...  0.99560  3.19       0.40
...             ...               ...          ...  ...      ...   ...        ...
4893            6.2              0.21         0.29  ...  0.99114  3.27       0.50
4894            6.6              0.32         0.36  ...  0.99490  3.15       0.46
4895            6.5              0.24         0.19  ...  0.99254  2.99       0.46
4896            5.5              0.29         0.30  ...  0.98869  3.34       0.38
4897            6.0              0.21         0.38  ...  0.98941  3.26       0.32

[4898 rows x 10 columns]
0       6
1       6
2       6
3       6
4       6
       ..
4893    6
4894    5
4895    6
4896    7
4897    6
Name: quality, Length: 4898, dtype: int64
```


Mean squared error
0.0
0.2
0.4
0.6
0.8
Decision Tree
Adaptive Boosting


Max error
0
1
2
3
4
5
Decision Tree
Adaptive Boosting


R2 score
−0.2
−0.1
0.0
0.1
0.2
Decision Tree
Adaptive Boosting


Mean Absolute Error
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Decision Tree
Adaptive Boosting