Ex No. 8	Decision Tree Pre-pruning and Post-pruning
Date:	

Aim

To construct a decision tree and apply pre-pruning and post-pruning operations on it.

Definition

Pruning

Decision trees are a machine learning algorithm that is susceptible to overfitting. One of the techniques you can use to reduce overfitting in decision trees is pruning.

Pre-pruning

The pre-pruning technique of Decision Trees is tuning the hyperparameters prior to the training pipeline. It involves the heuristic known as 'early stopping' which stops the growth of the decision tree - preventing it from reaching its full depth.

Post-pruning

Post-pruning does the opposite of pre-pruning and allows the Decision Tree model to grow to its full depth. Once the model grows to its full depth, tree branches are removed to prevent the model from overfitting.

Procedure

Open PyCharm Community Edition.

Go to File menu → New Project → Specify the project name → Press "Create" button.

Right Click on Project name → New → Python File → Specify the file name → Press Enter.

Type the following codes. Right click on file name or coding window → Select "Run" to view the result.

Decisiontree.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier

# Decision Tree Construction and Visualization
X,y=load_breast_cancer(return_X_y=True)
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=0)
clf=DecisionTreeClassifier(random_state=0)
clf.fit(X_train,y_train)
y_train_predicted=clf.predict(X_train)
y_test_predicted=clf.predict(X_test)
train_acc=accuracy_score(y_train,y_train_predicted)
test_acc=accuracy_score(y_test,y_test_predicted)

print("accuracy of training dataset:",train_acc)
print("accuracy of test dataset:",test_acc)

plt.figure(figsize=(16,8))
tree.plot_tree(clf)
plt.show()

# Post-Pruning
path=clf.cost_complexity_pruning_path(X_train,y_train)
#path variable gives two things ccp_alphas and impurities
ccp_alphas,impurities=path.ccp_alphas,path.impurities
print("ccp alpha wil give list of values :",ccp_alphas)
print("**************************************************************")
print("Impurities in Decision Tree :",impurities)

clfs=[]   #will store all the models here
for ccp_alpha in ccp_alphas:
    clf=DecisionTreeClassifier(random_state=0,ccp_alpha=ccp_alpha)
    clf.fit(X_train,y_train)
```

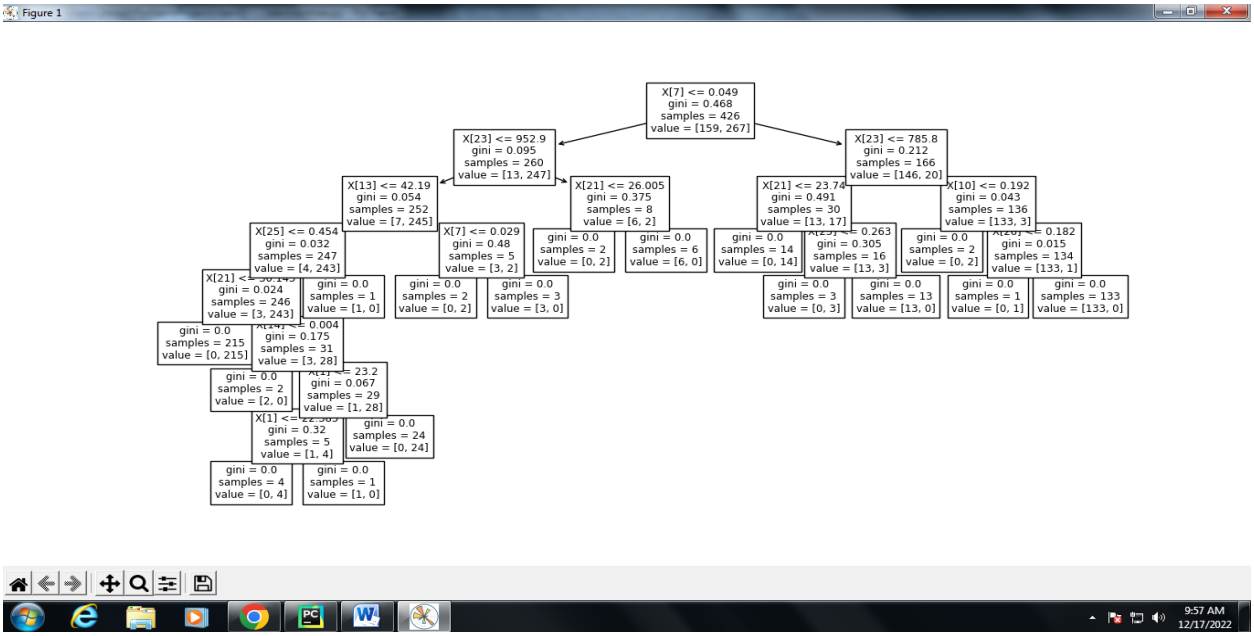
```
    clfs.append(clf)
print("Last node in Decision tree is {} and ccp_alpha for last node is {}".format(clfs[-
1].tree_.node_count,ccp_alphas[-1]))

train_scores = [clf.score(X_train, y_train) for clf in clfs]
test_scores = [clf.score(X_test, y_test) for clf in clfs]
fig, ax = plt.subplots()
ax.set_xlabel("alpha")
ax.set_ylabel("accuracy")
ax.set_title("Accuracy vs alpha for training and testing sets")
ax.plot(ccp_alphas, train_scores, marker='o', label="train",drawstyle="steps-post")
ax.plot(ccp_alphas, test_scores, marker='o', label="test",drawstyle="steps-post")
ax.legend()
plt.show()

clf=DecisionTreeClassifier(random_state=0,ccp_alpha=0.02)
clf.fit(X_train,y_train)
plt.figure(figsize=(12,8))
tree.plot_tree(clf,rounded=True,filled=True)
plt.show()

acc=accuracy_score(y_test,clf.predict(X_test))
print("accuracy of post-pruning operation:",acc)

# Pre-Pruning

clf=DecisionTreeClassifier(criterion= 'gini',max_depth= 17,min_samples_leaf= 3,min_samples_split=
12,splitter= 'random')
clf.fit(X_train,y_train)
plt.figure(figsize=(20,12))
tree.plot_tree(clf,rounded=True,filled=True)
plt.show()

y_predicted=clf.predict(X_test)
accuracy=accuracy_score(y_test,y_predicted)
print("accuracy of pre-pruning operation:",accuracy)
```

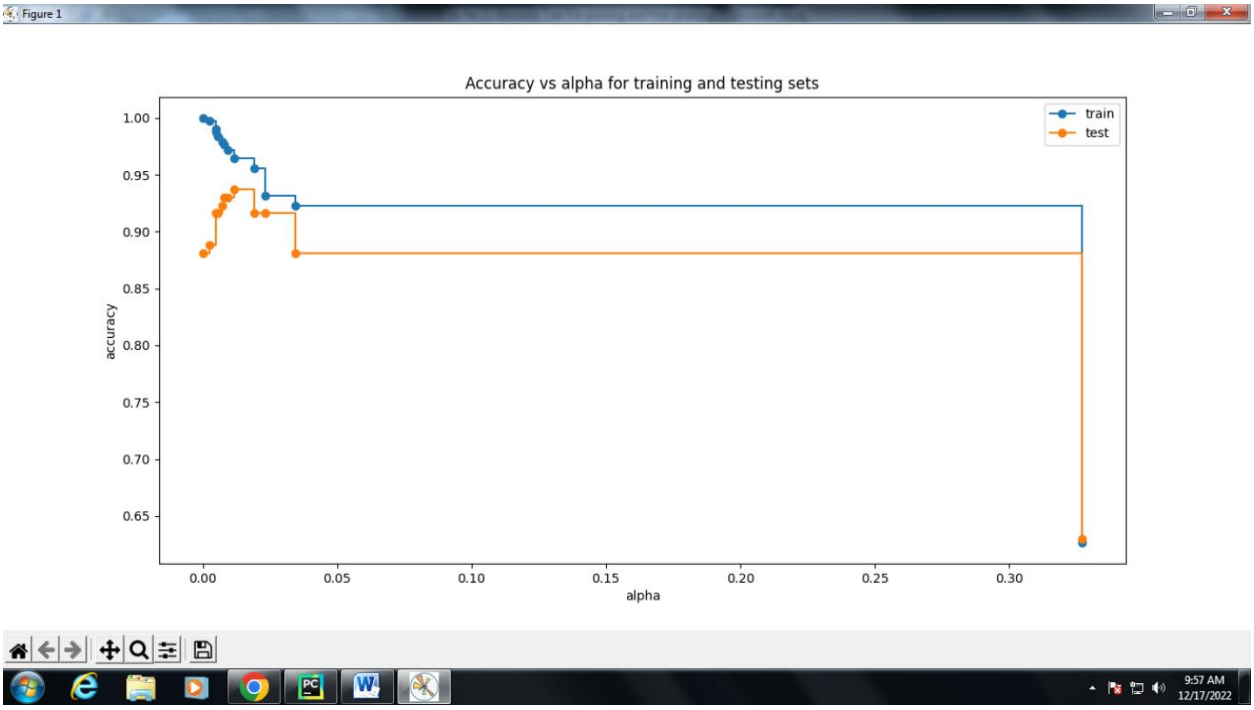
Output

```
C:\Users\2mca2\PycharmProjects\tam1\venv\Scripts\python.exe
C:/Users/2mca2/PycharmProjects/tam1/decisiontree.py
accuracy of training dataset: 1.0
accuracy of test dataset: 0.8811188811188811
ccp alpha wil give list of values : [0.         0.00226647 0.00464743 0.0046598  0.0056338  0.00704225
 0.00784194 0.00911402 0.01144366 0.018988   0.02314163 0.03422475
 0.32729844]
************************************************************
Impurities in Decision Tree : [0.         0.00453294 0.01847522 0.02313502 0.02876883 0.03581108
 0.04365302 0.05276704 0.0642107  0.0831987  0.10634033 0.14056508
 0.46786352]
Last node in Decision tree is 1 and ccp_alpha for last node is 0.3272984419327777
accuracy of post-pruning operation: 0.916083916083916
```

accuracy of pre-pruning operation: 0.9300699300699301

Process finished with exit code 0





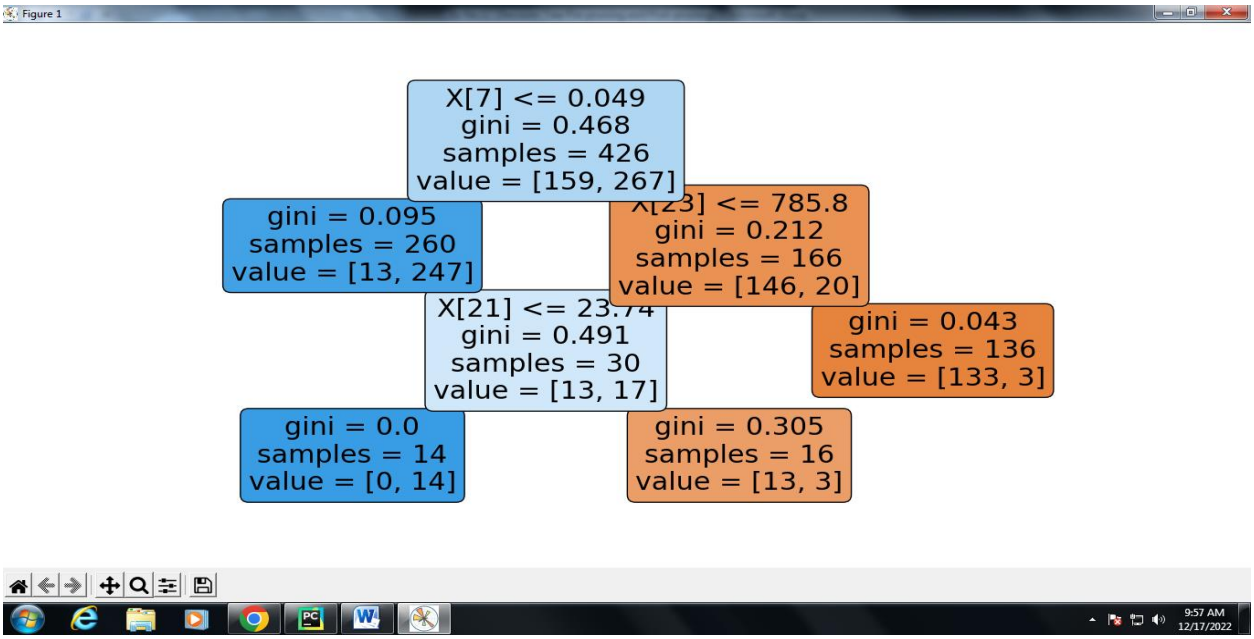


Figure 1
X[7] <= 0.049
gini = 0.468
samples = 426
value = [159, 267]
gini = 0.095
samples = 260
value = [13, 247]
X[23] <= 785.8
gini = 0.212
samples = 166
value = [146, 20]
X[21] <= 23.74
gini = 0.491
samples = 30
value = [13, 17]
gini = 0.043
samples = 136
value = [133, 3]
gini = 0.0
samples = 14
value = [0, 14]
gini = 0.305
samples = 16
value = [13, 3]
9:57 AM
12/17/2022

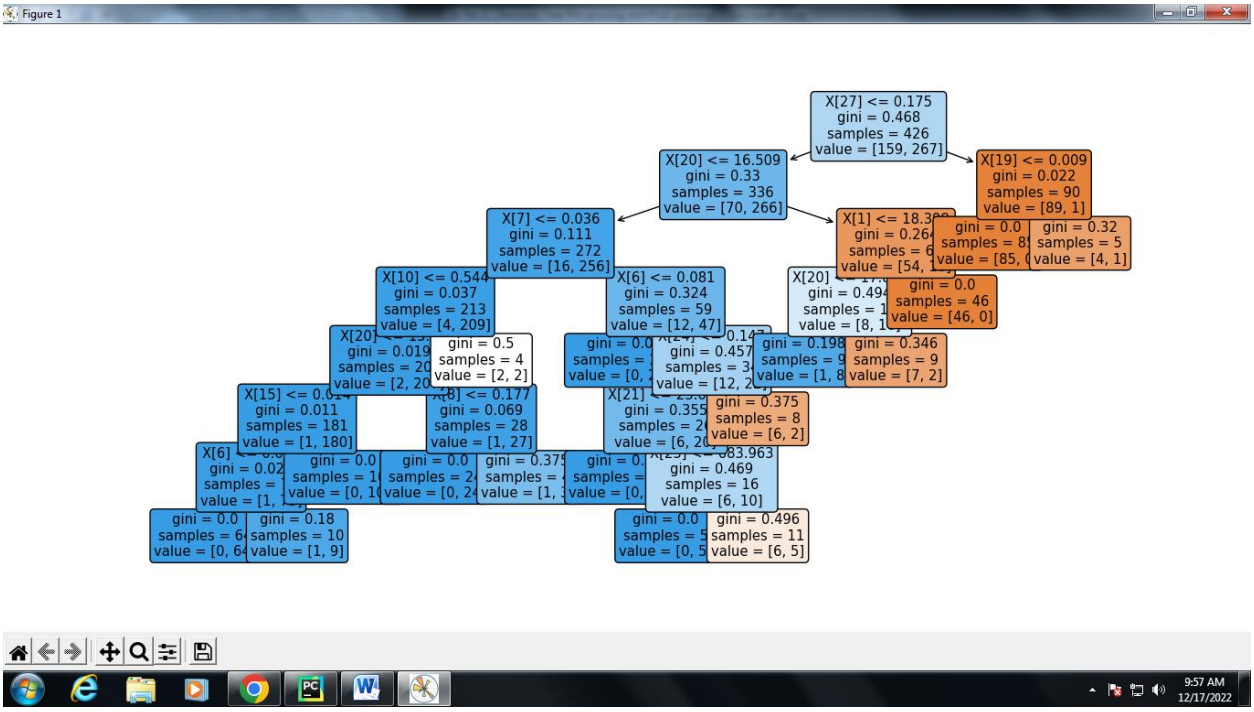


Figure 1
X[27] <= 0.175
gini = 0.468
samples = 426
value = [159, 267]
X[20] <= 16.509
gini = 0.33
samples = 336
value = [70, 266]
X[19] <= 0.009
gini = 0.022
samples = 90
value = [89, 1]
X[7] <= 0.036
gini = 0.111
samples = 272
value = [16, 256]
gini = 0.0
samples = 46
value = [46, 0]
gini = 0.5
samples = 4
value = [2, 2]
gini = 0.375
samples = 8
value = [6, 2]
gini = 0.496
samples = 11
value = [6, 5]
gini = 0.18
samples = 10
value = [1, 9]
9:57 AM
12/17/2022

Result

Thus, a decision tree has been successfully constructed. Pre-pruning and post-pruning operations have been performed and accuracy of both operations has compared.