Green University Of Bangladesh

Department Of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Fall, Year: 2023), B.Sc. in CSE (DAY)

LAB REPORT NO - 04

Course Title: Data Mining Lab

Course Code: CSE-436 **Section:** D2

Lab Experiment Name: Decision trees using Python

Student Details

	Name	ID
1	Sk. Nahid	201902073

Lab Date : 27/10/2023

Submission Date : 03/11/2023

Course Teacher Name : **Rezwanul Haque**

Lab Report Status

Mark:..... **Signature:.....**

Comments:..... **Date:.....**

1 INTRODUCTION

A decision tree is a hierarchical decision support model that employs a tree-like representation of decisions and their potential outcomes, such as resource costs, utility, and chance event outcomes.

2 OBJECTIVE

This lab report explores Decision Tree pruning, presenting the theory, dataset selection, Python implementation, 15-fold cross-validation, and results. Screenshots and full tree figures illustrate the impact of pruning on classification accuracy.

3 THEORY

3.1 Decision Tree

Decision Trees are a versatile machine learning tool for classification and regression tasks. They partition data based on feature values, forming a tree-like structure to make decisions. Pruning is the process of reducing tree complexity, avoiding overfitting and improving generalization.

3.2 Pruning Reasons and Types

Pruning is essential to optimize Decision Trees. Without pruning, trees tend to overfit, capturing noise in the data. Pruning reduces tree size by removing branches with little predictive power, improving model efficiency and generalization, ensuring it performs better on unseen data. There are two types of pruning:

1. **Pre-Pruning:** A type of pruning, stops the tree from growing beyond a certain point. It sets conditions on the tree structure during its growth to prevent overfitting and make it more generalizable.

2. **Post-Pruning:** Post-pruning involves growing a complete tree and then selectively removing branches to improve its accuracy. This technique optimizes an initially overgrown tree by trimming off nodes that do not contribute significantly to classification accuracy, resulting in a more efficient and generalized decision tree.

4 IMPLEMENTATION

```
1 import numpy as np
2 import pandas as pd
3 import sklearn
```

```
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import confusion_matrix
from sklearn.tree import DecisionTreeClassifier, export_graphviz,
    export_text
from sklearn.metrics import accuracy_score
from IPython . display import Image, display
from sklearn.datasets import load_wine
from sklearn.metrics import accuracy_score
from matplotlib import pyplot as plt
from PIL import Image as PILImage
from sklearn import tree
import graphviz
```

Listing 1: Import Library & Dataset

```
#Loading datasets
data = load_wine()
iris=pd.DataFrame(data.data)
iris_targets=pd.DataFrame(data.target)
feature_names = data.feature_names
print("Feature Names:")
for feature in feature_names:
    print(feature)
print ("Targets Name : ", data.target_names)
print ("Dataset Shape: ", iris.shape)
print ("Dataset: ",iris.head())
print ("Dataset: ",iris_targets.head())
# features and targets
X = data.data
y = data.target
```

Listing 2: Tuples, Attributes & Class

```
# Splitting the dataset into train and test
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2,
    random_state = 42)
```

Listing 3: Splitting Dataset

```
    dt_classifier = DecisionTreeClassifier(random_state=42)

cv_scores_before = cross_val_score(dt_classifier, X, y, cv=15)
mean_cv_score_before = cv_scores_before.mean()

dt_classifier.fit(X_train, y_train)

print("Unpruned Decision Tree:")
text_representation_before = export_text(dt_classifier)
print(text_representation_before)

dot_data_before = export_graphviz(dt_classifier, out_file=None,
```

```
                                    feature_names=data.feature_names,
    class_names=data.target_names, filled=True, rounded=True,
    special_characters=True)
graph_before = graphviz.Source(dot_data_before)
display(Image(graph_before.pipe(format='png')))

path = dt_classifier.cost_complexity_pruning_path(X_train, y_train)
ccp_alphas, impurities = path.ccp_alphas, path.impurities

clfs = []
for ccp_alpha in ccp_alphas:
    clf = DecisionTreeClassifier(random_state=0, ccp_alpha=ccp_alpha)
    clf.fit(X_train, y_train)
    clfs.append(clf)

clfs = clfs[:-1]
ccp_alphas = ccp_alphas[:-1]

cv_scores_after = [cross_val_score(clf, X, y, cv=15).mean() for clf in
    clfs]

max_accuracy_index = cv_scores_after.index(max(cv_scores_after))
best_clf = clfs[max_accuracy_index]

print("\nUnpruned Decision Tree - Mean Cross-validation score before
    pruning:", mean_cv_score_before)
```

Listing 4: Unpruned Decision Tree

```

print("\nBest Pruned Decision Tree:")
text_representation_after = export_text(best_clf)
print(text_representation_after)
dot_data_after = export_graphviz(best_clf, out_file=None, feature_names=
    data.feature_names,
                                 class_names=data.target_names, filled=True
    , rounded=True, special_characters=True)
graph_after = graphviz.Source(dot_data_after)
display(Image(graph_after.pipe(format='png')))
print("Best Pruned Decision Tree - Mean Cross-validation score after
    pruning:", cv_scores_after[max_accuracy_index])
```

Listing 5: Pruned Decision Tree

5 OUTPUT

5.1 Tuples, Attributes & Classes

```
Feature Names:
alcohol
malic_acid
ash
alcalinity_of_ash
magnesium
total_phenols
flavanoids
nonflavanoid_phenols
proanthocyanins
color_intensity
hue
od280/od315_of_diluted_wines
proline
Targets Name :  ['class_0' 'class_1' 'class_2']
Dataset Shape:  (178, 13)
Dataset:        0     1     2     3      4     5     6     7     8     9     10    11  \
0  14.23  1.71  2.43  15.6  127.0  2.80  3.06  0.28  2.29  5.64  1.04  3.92
1  13.20  1.78  2.14  11.2  100.0  2.65  2.76  0.26  1.28  4.38  1.05  3.40
2  13.16  2.36  2.67  18.6  101.0  2.80  3.24  0.30  2.81  5.68  1.03  3.17
3  14.37  1.95  2.50  16.8  113.0  3.85  3.49  0.24  2.18  7.80  0.86  3.45
4  13.24  2.59  2.87  21.0  118.0  2.80  2.69  0.39  1.82  4.32  1.04  2.93
```

Figure 1: Tuples, Attributes & Classes

5.2 Before Pruning

Unpruned Decision Tree - Mean Cross-validation score before pruning: 0.866161616161616

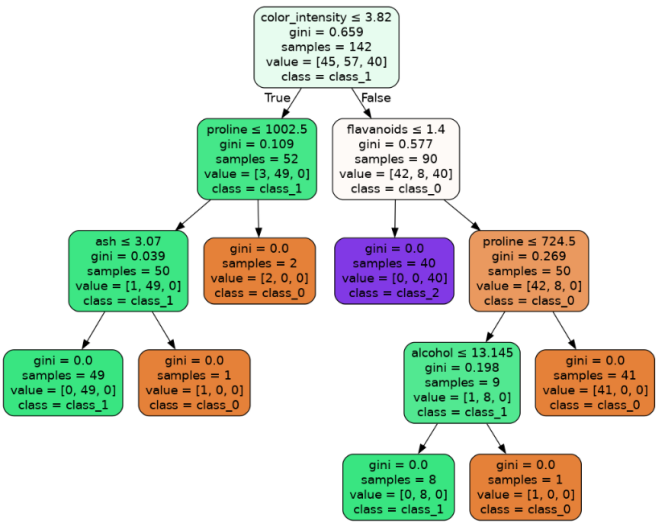

Figure 2: Unpruned Decision Tree

5.3 After Pruning

Best Pruned Decision Tree - Mean Cross-validation score after pruning: 0.8828282828282829

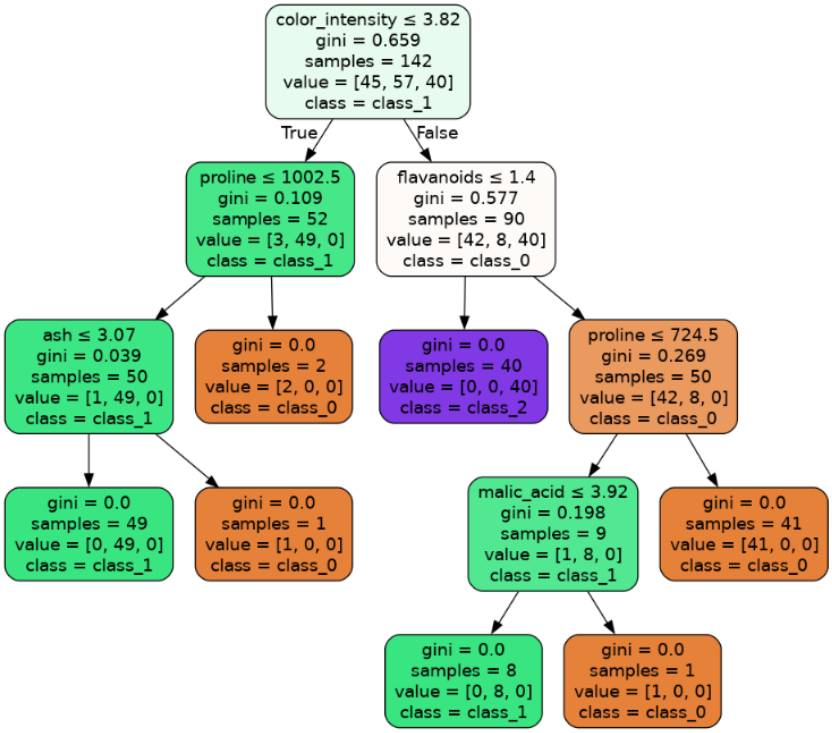

Figure 3: Pruned Decision Tree

6 DISCUSSION & ANALYSIS

The unpruned decision tree achieved a mean cross-validation score of 0.866, while the best-pruned decision tree improved to a score of 0.883. Pruning enhanced the tree's performance, making it a more reliable classifier for the dataset, indicating the importance of pruning in decision tree optimization.