**Decision Tree: Report:**

**Describe the Decision Tree method:**

A decision tree is a decision support tool that uses a tree-like graph and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements i.e. pseudo code without any declaration statements and only the if- clause and for statements.

**The dataset and if you have done any pre-process:**

The dataset is the wine dataset which contains all the below characteristics as:

Samples 178

Features 13

Classes 3

Data Set Characteristics: Multivariate

Attribute Characteristics: Integer, Real

Associated Tasks: Classification

Missing Values None

column attribute

1) Class Label

2) Alcohol

3) Malic acid

4) Ash

5) Alcalinity of ash

6) Magnesium

7) Total phenols

8) Flavanoids

9) Nonflavanoid phenols

10) Proanthocyanins

11) Color intensity

12) Hue

13) OD280/OD315 of diluted wines

14) Proline

class samples

0 59

1 71

2 48

No, there is no pre processing for the dataset for constructing the decision tree or the gini/ entropy classification.

**Code for the Decision Tree on Gini classification:**

import numpy as np

import pandas as pd

from sklearn import datasets, linear\_model, svm

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

from sklearn import tree

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

import graphviz

#load datasets

wine = datasets.load\_wine()

df = pd.DataFrame(wine.data)

y = wine.target

z = wine.data

print ("The first 5 records of the Dataset are: \n ", df.head(5),"\n")

print ("The last 5 records of the Dataset are: \n", df.tail(5),"\n")

print ("The class names of the Wine Dataset are: \n",wine.feature\_names,"\n")

print("The class names of the Wine Dataset are: \n",wine.target\_names,"\n")

#split the training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df, y, test\_size=0.2)

print ("Dimensions of the Training Data Set are: \n",X\_train.shape, y\_train.shape,"\n")

print ("Dimensions of the Test Data Set are: \n",X\_test.shape, y\_test.shape,"\n")

#Using Gini to measure the quality of the split.

clf\_gini = DecisionTreeClassifier(criterion = "gini", random\_state = 100,

max\_depth=3, min\_samples\_leaf=5)

clf\_gini.fit(X\_train, y\_train)

#Prediction using Gini

y\_pred = clf\_gini.predict(X\_test)

y\_pred

#Confusion Matrix

confusion\_matrix(y\_test, y\_pred)

#Classification Report

print(classification\_report(y\_test, y\_pred, target\_names=wine.target\_names))

#Decision Tree Visualization

clf = tree.DecisionTreeClassifier()

clf = clf.fit(wine.data, wine.target)

dot\_data = tree.export\_graphviz(clf\_gini, out\_file=None,

feature\_names=wine.feature\_names,

class\_names=wine.target\_names,

filled=True, rounded=True,

special\_characters=True)

graph = graphviz.Source(dot\_data)

graph

**Code for the Decision Tree on Entropy Classification:**

import numpy as np

import pandas as pd

from sklearn import datasets, linear\_model, svm

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

from sklearn import tree

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

import graphviz

#load datasets

wine = datasets.load\_wine()

df = pd.DataFrame(wine.data)

y = wine.target

z = wine.data

print ("The first 5 records of the Dataset are: \n ", df.head(5),"\n")

print ("The last 5 records of the Dataset are: \n", df.tail(5),"\n")

print ("The class names of the Wine Dataset are: \n",wine.feature\_names,"\n")

print("The class names of the Wine Dataset are: \n",wine.target\_names,"\n")

#split the training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(df, y, test\_size=0.2)

print ("Dimensions of the Training Data Set are: \n",X\_train.shape, y\_train.shape,"\n")

print ("Dimensions of the Test Data Set are: \n",X\_test.shape, y\_test.shape,"\n")

#Using Entropy to measure the quality of the split.

clf\_entropy = DecisionTreeClassifier(criterion = "entropy", random\_state = 100,

max\_depth=3, min\_samples\_leaf=5)

clf\_entropy.fit(X\_train, y\_train)

#Prediction using Entropy

y\_pred = clf\_entropy.predict(X\_test)

y\_pred

#Confusion Matrix

confusion\_matrix(y\_test, y\_pred)

#Classification Report

print(classification\_report(y\_test, y\_pred, target\_names=wine.target\_names))

#Decision Tree Visualization

dot\_data = tree.export\_graphviz(clf\_entropy, out\_file=None,

feature\_names=wine.feature\_names,

class\_names=wine.target\_names,

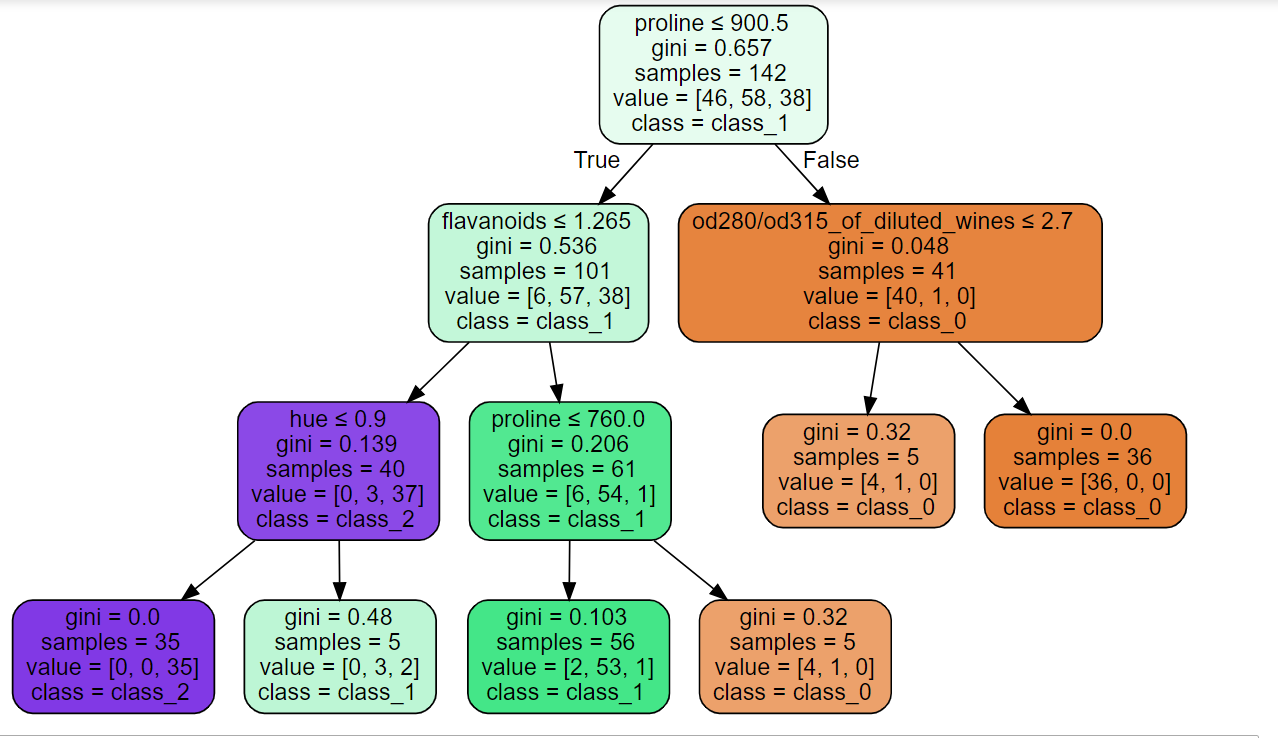
filled=True, rounded=True,

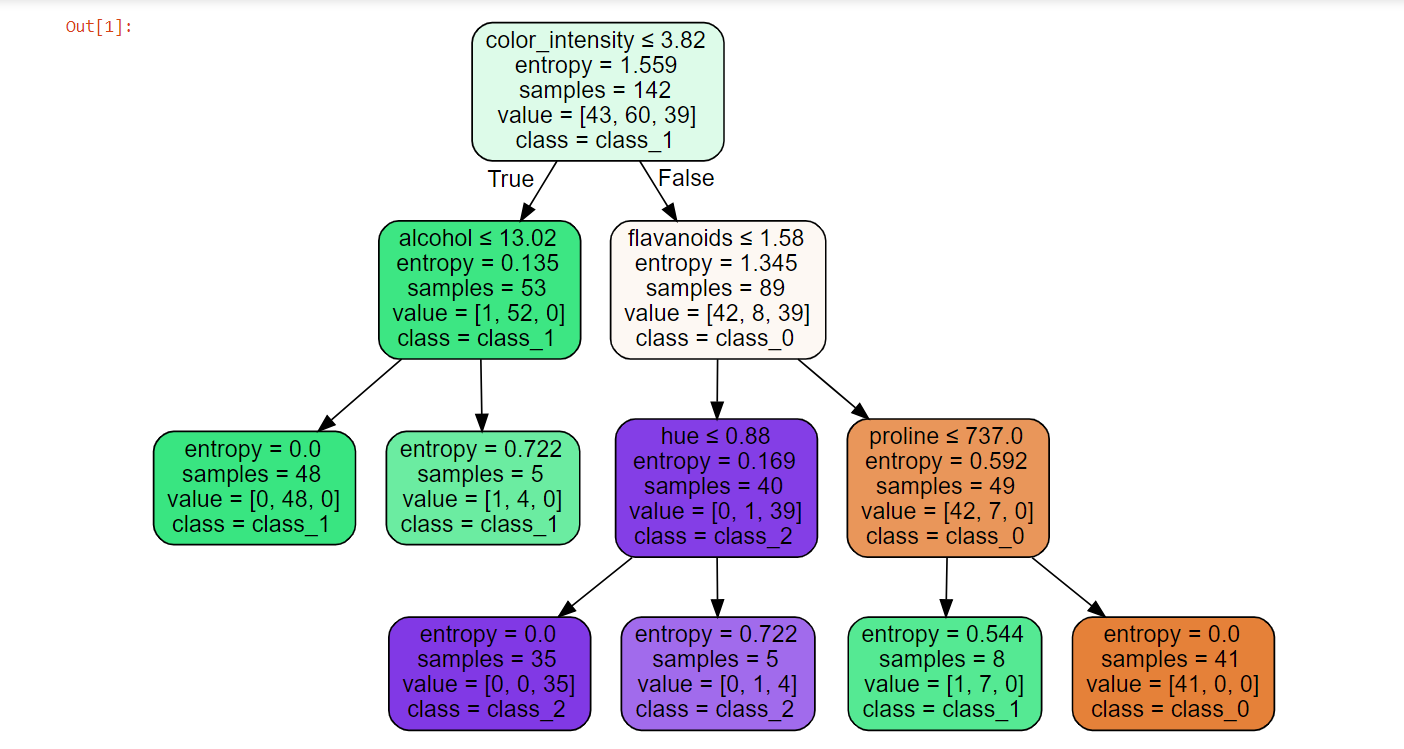
special\_characters=True)

graph = graphviz.Source(dot\_data)

graph

**Visualization for Gini and Entropy Decision Trees:**





**Interpret and compare results:**

**Entropy:**

precision recall f1-score support

class\_0 1.00 0.75 0.86 16

class\_1 0.73 1.00 0.85 11

class\_2 1.00 1.00 1.00 9

avg / total 0.92 0.89 0.89 36

**Gini:**

precision recall f1-score support

class\_0 1.00 1.00 1.00 9

class\_1 0.89 0.89 0.89 19

class\_2 0.75 0.75 0.75 8

avg / total 0.89 0.89 0.89 36

As we can see in the above results,

Precision was greater for Entropy.

Recall and F1 score are pretty much equal.