**AIM** :

*Program to implement decision trees using any standard dataset available in the public Domain and find the accuracy of the algorithm.*

**Program**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

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

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import classification\_report, confusion\_matrix

from sklearn.tree import plot\_tree

df = sns.load\_dataset('iris')

print(df.head())

print(df.info())

df.isnull().any()

print(df.shape)

# Let's plot pair plot to visualise the attributes all at once

sns.pairplot(data=df, hue="species")

plt.savefig('pne.png')

# Correction matrix

sns.heatmap(df.corr())

plt.savefig('one.png')

target = df['species']

df1 = df.copy()

df1 = df1.drop('species', axis=1)

print(df1.shape)

print(df1.head())

# Defining the attributes

x = df1

print(target)

# label encoding

le = LabelEncoder()

target = le.fit\_transform(target)

print(target)

y = target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state=42)

print("Training split input- ", X\_train.shape)

print("Testing split input- ", X\_test.shape)

# Defining the decision tree algorithm

dtree = DecisionTreeClassifier()

dtree.fit(X\_train, y\_train)

print('Decision Tree Classifier Created')

y\_pred = dtree.predict(X\_test)

print('Classification report - \n', classification\_report(y\_test, y\_pred))

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

plt.figure(figsize=(5, 5))

sns.heatmap(data=cm, linewidth=.5, annot=True, square=True, cmap='Blues')

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

all\_sample\_title = 'Accuracy score: {0}'.format(X\_test, y\_test)

plt.title(all\_sample\_title, size=15)

plt.savefig('two.png')

plt.figure(figsize=(20, 20))

dec\_tree = plot\_tree(decision\_tree=dtree, feature\_names=df1.columns,

class\_names=['setosa', 'vercicolor', 'verginica'], filled=True, precision=4, rounded=True)

plt.savefig('tree.png')

**Output**

"C:\Users\aksa\PycharmProjects\add number\venv\Scripts\python.exe" "C:/Users/aksa/PycharmProjects/add number/dt2.py"

sepal\_length sepal\_width petal\_length petal\_width species

0 5.1 3.5 1.4 0.2 setosa

1 4.9 3.0 1.4 0.2 setosa

2 4.7 3.2 1.3 0.2 setosa

3 4.6 3.1 1.5 0.2 setosa

4 5.0 3.6 1.4 0.2 setosa

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 150 entries, 0 to 149

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 sepal\_length 150 non-null float64

1 sepal\_width 150 non-null float64

2 petal\_length 150 non-null float64

3 petal\_width 150 non-null float64

4 species 150 non-null object

dtypes: float64(4), object(1)

memory usage: 6.0+ KB

None

(150, 5)

(150, 4)

sepal\_length sepal\_width petal\_length petal\_width

0 5.1 3.5 1.4 0.2

1 4.9 3.0 1.4 0.2

2 4.7 3.2 1.3 0.2

3 4.6 3.1 1.5 0.2

4 5.0 3.6 1.4 0.2

0 setosa

1 setosa

2 setosa

3 setosa

4 setosa

...

145 virginica

146 virginica

147 virginica

148 virginica

149 virginica

Name: species, Length: 150, dtype: object

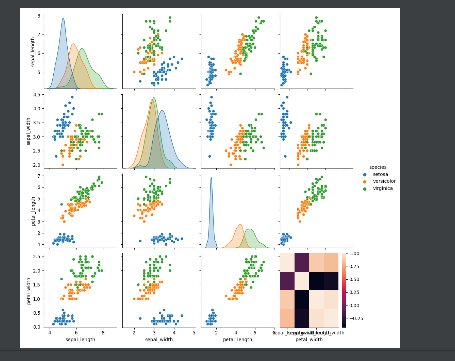
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

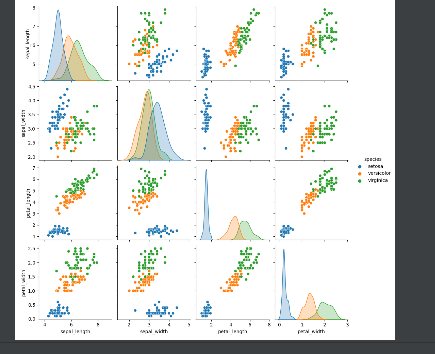
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2

2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

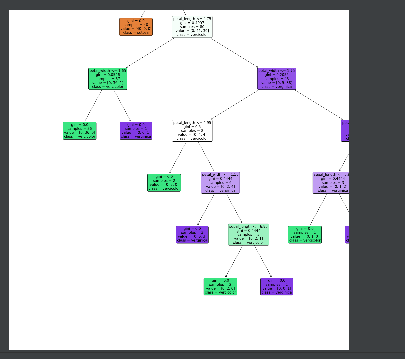
One.png



Pne.png



Tree.png



Two.png

