Experiment No.: 8

Aim

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

CO3

Use different packages and frameworks to implement text classification using SVM and clustering using k-means

Procedure

```
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)
sns.pairplot(data=df, hue='species')
plt.savefig("pne.png")
# correlation matrix
sns.heatmap(df.corr())
plt.savefig("one.png")
target = df['species']
```

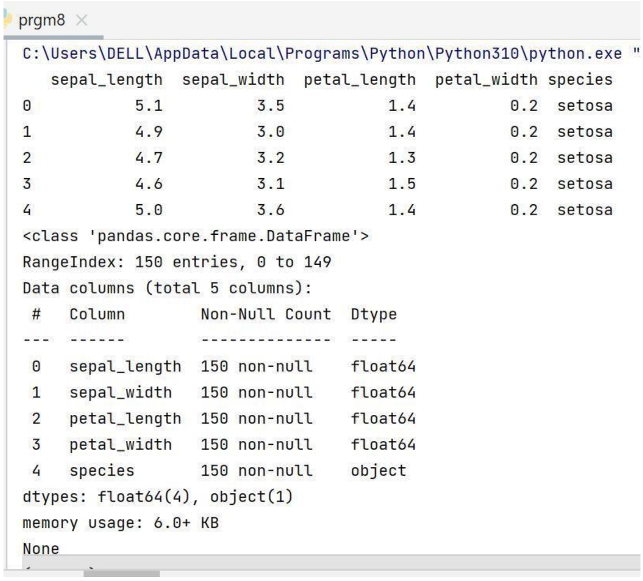
```
df1 = df.copy()
df1 = df1.drop('species', axis=1)
print(df1.shape)
print(df1.head())
# defining attributes
x = df1
print(target)
# label encoding
le = LabelEncoder()
target = le.fit_transform(target)  # learn scaling parameters(species)
print(target)
y = target
x_train, x_test, y_train, y_test = 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')
# predicting the value of test data
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, linewidths=.5, annot=True, square=True, cmap='Blues')
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
all_sample_title = 'Accuracy score: {0}'.format(dtree.score(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"],

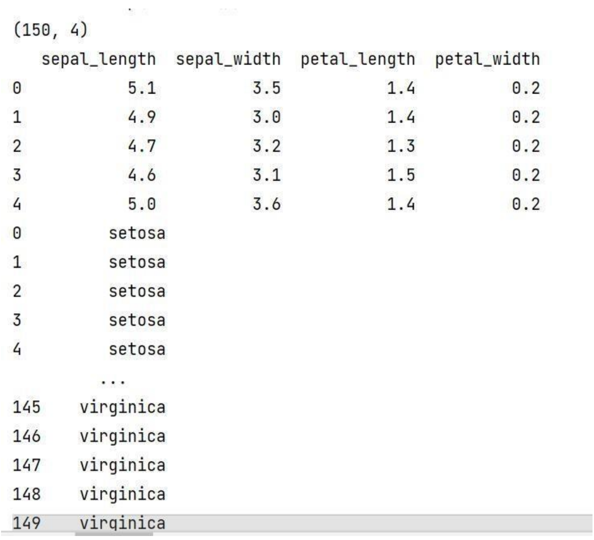
filled=True, precision=4, rounded=True)

plt.savefig("three.png")

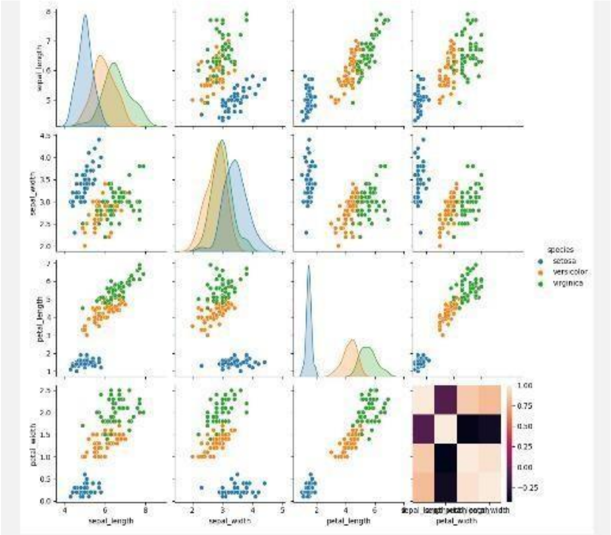
Output Screenshot

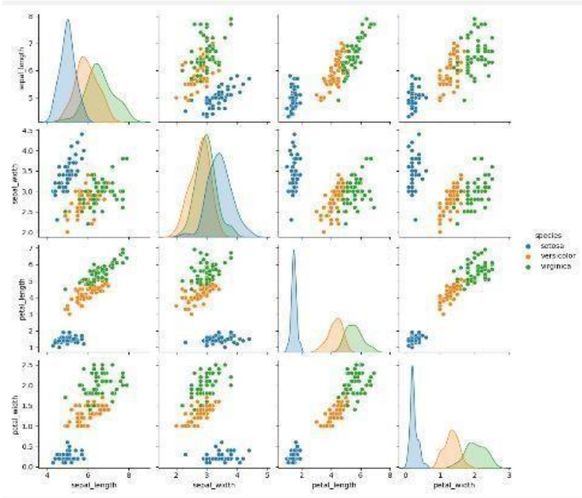


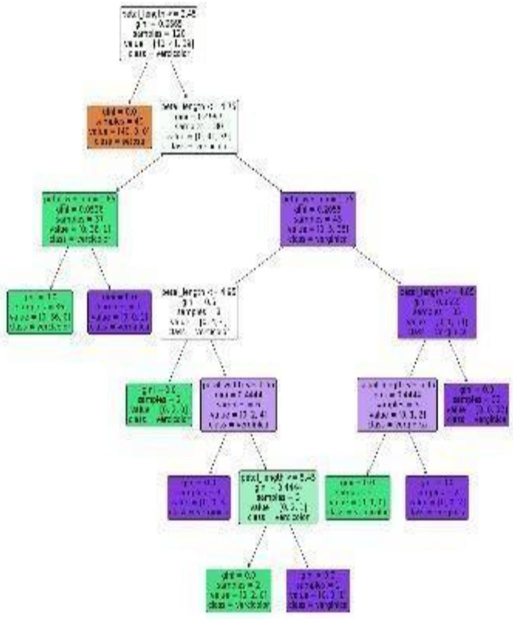
```
C:\Users\DELL\AppData\Local\Programs\Python\Python310\python.exe "
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           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
```

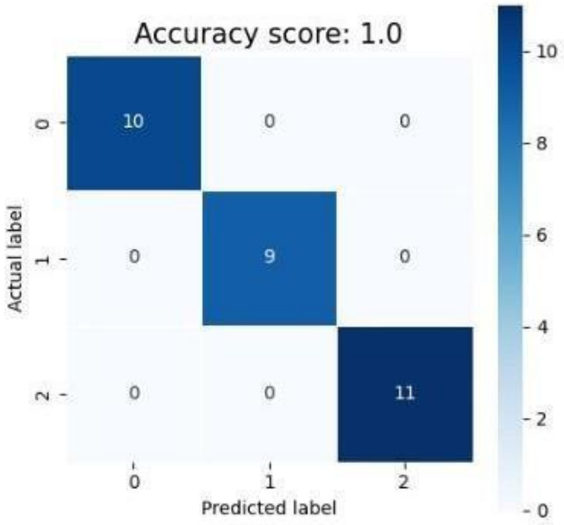


```
(150, 4)
   sepal_length  sepal_width  petal_length  petal_width
0           5.1          3.5           1.4          0.2
1           4.9          3.0           1.4          0.2
2           4.7          3.2           1.3          0.2
3           4.6          3.1           1.5          0.2
4           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
```









Result

The program was executed and the result was successfully obtained. Thus CO3 was obtained.