Name: Saloni Vishwakarma

Roll No: C1-13

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn import metrics
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

```
d1=pd.read_csv('tennis.csv')
print(d1)
```

```
     outlook  temp humidity  windy play
0      sunny   hot     high  False   no
1      sunny   hot     high   True   no
2   overcast   hot     high  False  yes
3      rainy  mild     high  False  yes
4      rainy  cool   normal  False  yes
5      rainy  cool   normal   True   no
6   overcast  cool   normal   True  yes
7      sunny  mild     high  False   no
8      sunny  cool   normal  False  yes
9      rainy  mild   normal  False  yes
10     sunny  mild   normal   True  yes
11  overcast  mild     high   True  yes
12  overcast   hot   normal  False  yes
13     rainy  mild     high   True   no
```

```
d1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   outlook   14 non-null     object
 1   temp      14 non-null     object
 2   humidity  14 non-null     object
 3   windy     14 non-null     bool  
 4   play      14 non-null     object
dtypes: bool(1), object(4)
memory usage: 594.0+ bytes
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
d1.iloc[:,0] = le.fit_transform(d1.iloc[:,0]) # outlook column - index no. 0- will encode c
d1.iloc[:,1] = le.fit_transform(d1.iloc[:,1]) # temp
d1.iloc[:,2] = le.fit_transform(d1.iloc[:,2]) # humidity
d1.iloc[:,3] = le.fit_transform(d1.iloc[:,3]) # windy
d1.iloc[:,4] = le.fit_transform(d1.iloc[:,4]) # play
print(d1)
d1.info()
```

```
    outlook  temp  humidity  windy  play
0         2     1         0      0     0
1         2     1         0      1     0
2         0     1         0      0     1
3         1     2         0      0     1
4         1     0         1      0     1
5         1     0         1      1     0
6         0     0         1      1     1
7         2     2         0      0     0
8         2     0         1      0     1
9         1     2         1      0     1
10        2     2         1      1     1
11        0     2         0      1     1
12        0     1         1      0     1
13        1     2         0      1     0
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14 entries, 0 to 13
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   outlook   14 non-null     int32
 1   temp      14 non-null     int32
 2   humidity  14 non-null     int32
 3   windy     14 non-null     int64
 4   play      14 non-null     int32
dtypes: int32(4), int64(1)
memory usage: 468.0 bytes
C:\Users\sharv\AppData\Local\Temp\ipykernel_13348\1076206382.py:3: DeprecationWarning
  d1.iloc[:,0] = le.fit_transform(d1.iloc[:,0]) # outlook column - index no. 0- will
C:\Users\sharv\AppData\Local\Temp\ipykernel_13348\1076206382.py:4: DeprecationWarning
  d1.iloc[:,1] = le.fit_transform(d1.iloc[:,1]) # temp
C:\Users\sharv\AppData\Local\Temp\ipykernel_13348\1076206382.py:5: DeprecationWarning
  d1.iloc[:,2] = le.fit_transform(d1.iloc[:,2]) # humidity
C:\Users\sharv\AppData\Local\Temp\ipykernel_13348\1076206382.py:7: DeprecationWarning
  d1.iloc[:,4] = le.fit_transform(d1.iloc[:,4]) # play
```

for i in range(5): d1.iloc[:,i] = le.fit_transform(d1.iloc[:,i])

```
x=d1.iloc[:,0:4]
y=d1.iloc[:,4]
print(x.shape)
print(y.shape)
```

```
print(y.shape)
```

```
(14, 4)
(14,)
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(11, 4)
(11,)
(3, 4)
(3,)
```

```
rfclassifier=RandomForestClassifier()

rfclassifier=rfclassifier.fit(x_train,y_train)
y_pred=rfclassifier.predict(x_test)
```

```
print("Actual Values:")
print(y_test)
print("Predicted values:")
print(y_pred)
```

```
Actual Values:
3    1
7    0
6    1
Name: play, dtype: int32
Predicted values:
[1 0 1]
```

Model Evaluation

```
print("accuracy",metrics.accuracy_score(y_test,y_pred))
```

```
accuracy 1.0
```

```
cf=confusion_matrix(y_test,y_pred)
```

```
print(cf)
```

```
[[1 0]
```

```
[[1 0]
 [0 2]]
```

```
print(classification_report(y_test,y_pred))
```

```
                  precision    recall  f1-score   support

               0       1.00      1.00      1.00         1
               1       1.00      1.00      1.00         2

        accuracy                           1.00         3
       macro avg       1.00      1.00      1.00         3
    weighted avg       1.00      1.00      1.00         3
```

x[0]=outlook x[1]=temp x[2]=humidity x[3]=windy