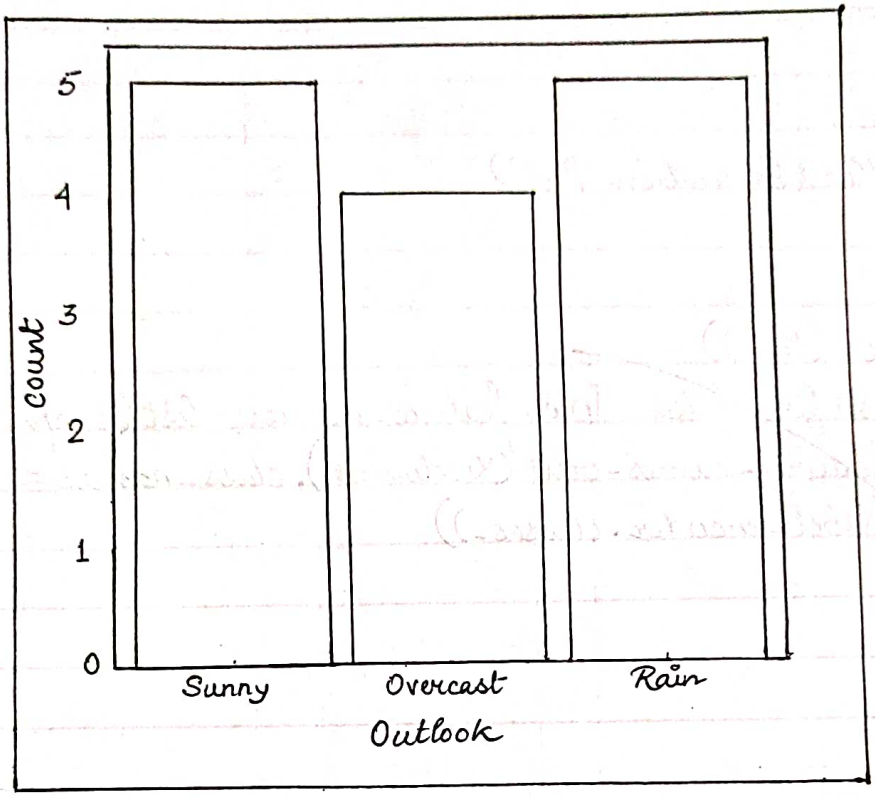
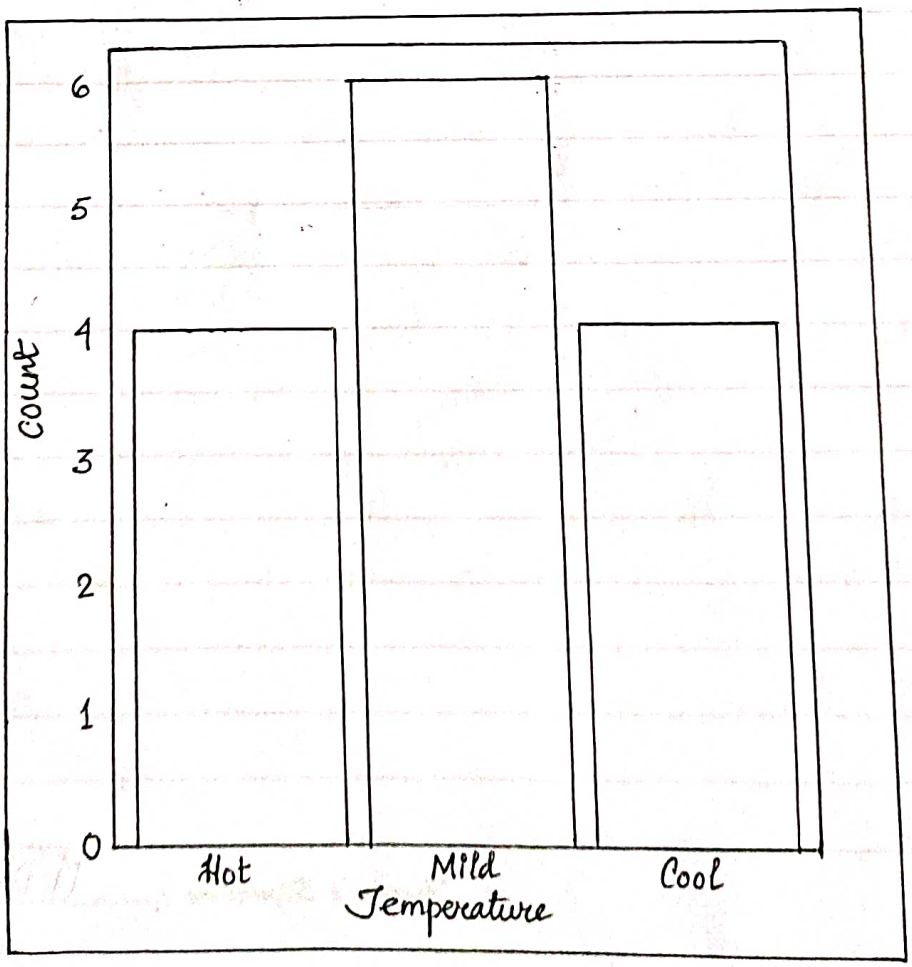

5
4
3
2
1
0
Count
Sunny
Overcast
Rain
Outlook


6
5
4
3
2
1
0
Count
Hot
Mild
Cool
Temperature

Expt, No.

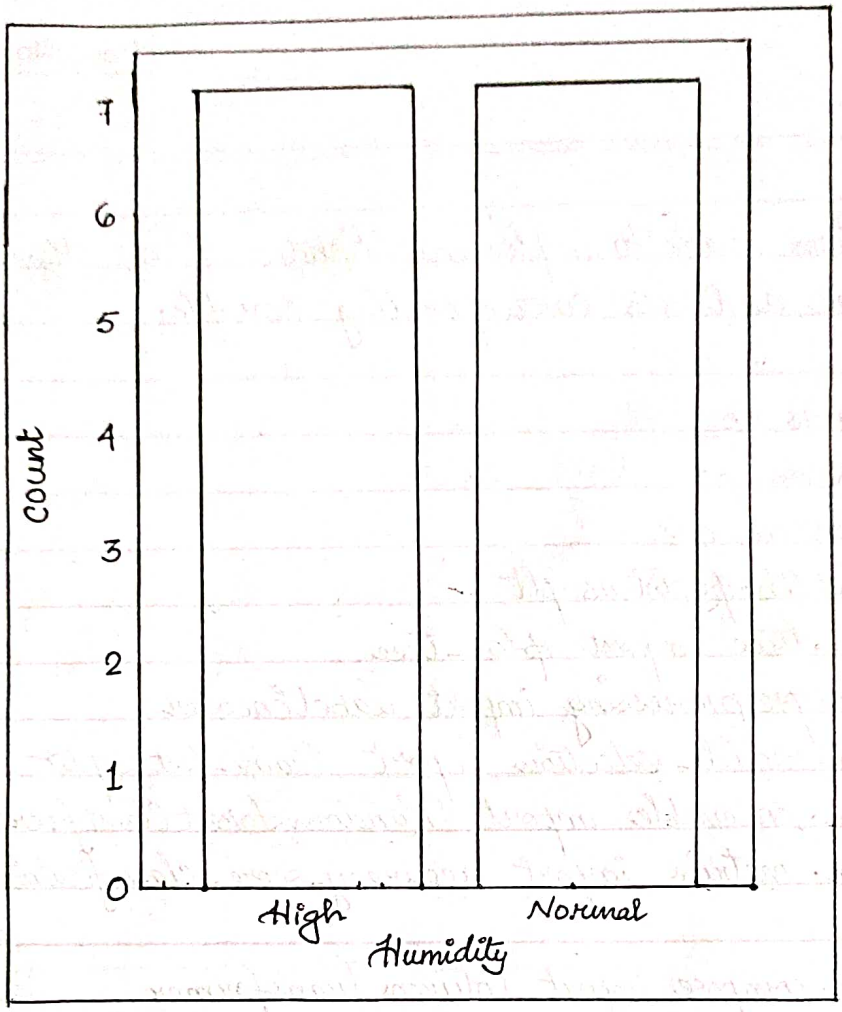
Page No. 14
Date

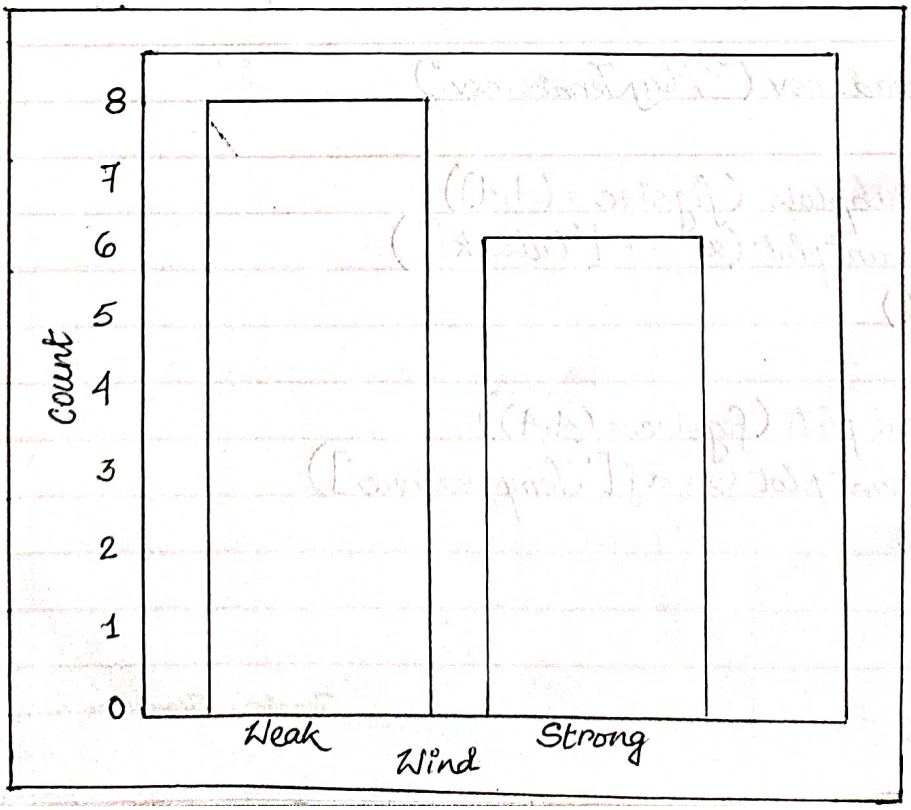
Write a python code to implement Random Forest Algorithm using LabelEncoder and its corresponding csv file.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
                            confusion_matrix
from sklearn.compose import ColumnTransformer

df = pd.read_csv("PlayTennis.csv")

ax = plt.subplots(figsize = (4,4))
ax = sns.countplot(x = df['Outlook'])
plt.show()

ax = plt.subplots(figsize = (4,4))
ax = sns.countplot(x = df['Temperature'])
plt.show()
```

Avijit®

Teacher's Signature :.....


7
6
5
4
3
2
1
0
Count
High
Normal
Humidity


8
7
6
5
4
3
2
1
0
Count
Weak
Strong
Wind

Expt, No.

Page No. 15
Date

```
ax = plt.subplots (figsize = (4,4))
ax = sns.countplot (df ['Humidity'])
plt.show()

ax = plt.subplots (figsize = (4,4))
ax = sns.countplot (x = df ['Wind'])
plt.show()

X = df.iloc [:, :-1]
y = df.iloc [:, -1]

label_encoder = LabelEncoder()
X_encoded = X.copy()
for column in X.columns:
    if X[column].dtype == 'object':
        X_encoded [column] = label_encoder.fit_transform
                                (X[column])
y_encoded = label_encoder.fit_transform (y)

X_train, X_test, y_train, y_test = train_test_split (X_encoded,
                                    y_encoded, test_size = 0.2, random_state = 42)

rf_classifier = RandomForestClassifier (n_estimators = 50)
rf_classifier.fit (X_train, y_train)
y_pred = rf_classifier.predict (X_test)
```

Avijit®

Teacher's Signature :.....

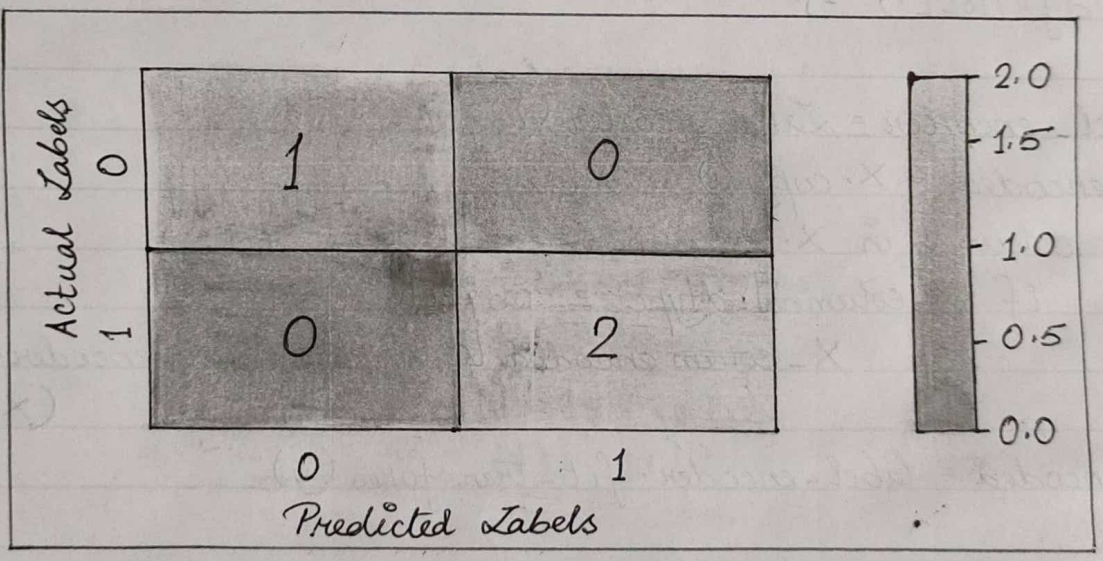
RF Accuracy : 1.0

RF Confusion Matrix :

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

RF Classification Report :

	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



The person is predicted not to play tennis.

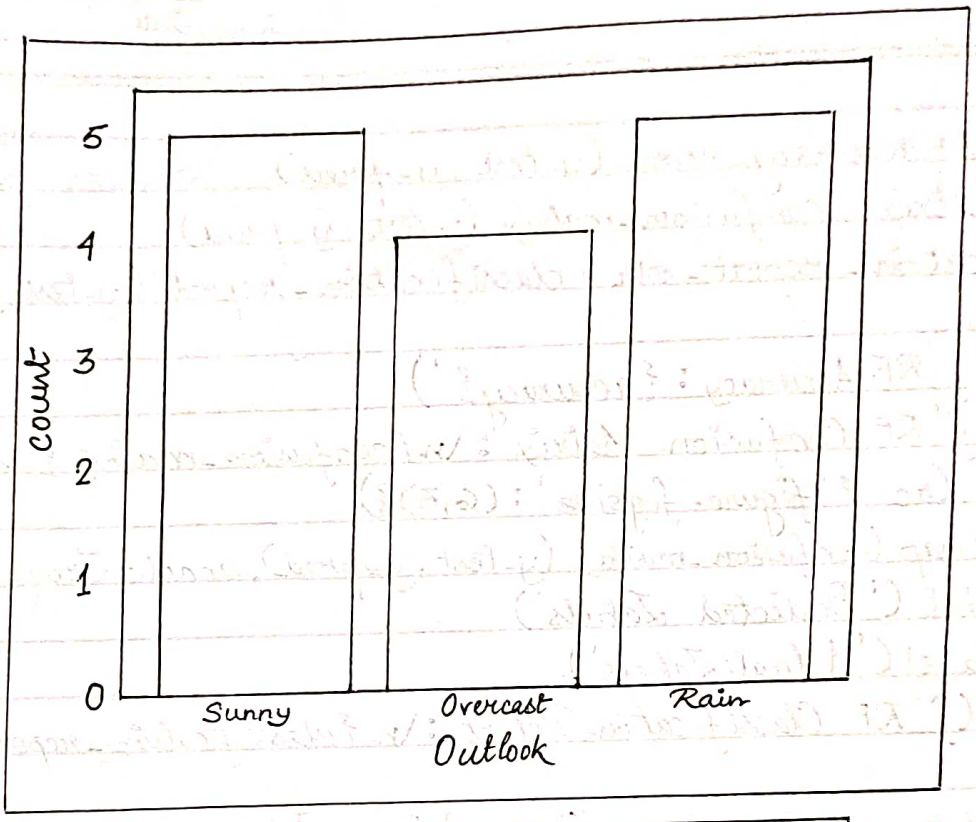
Expt, No.

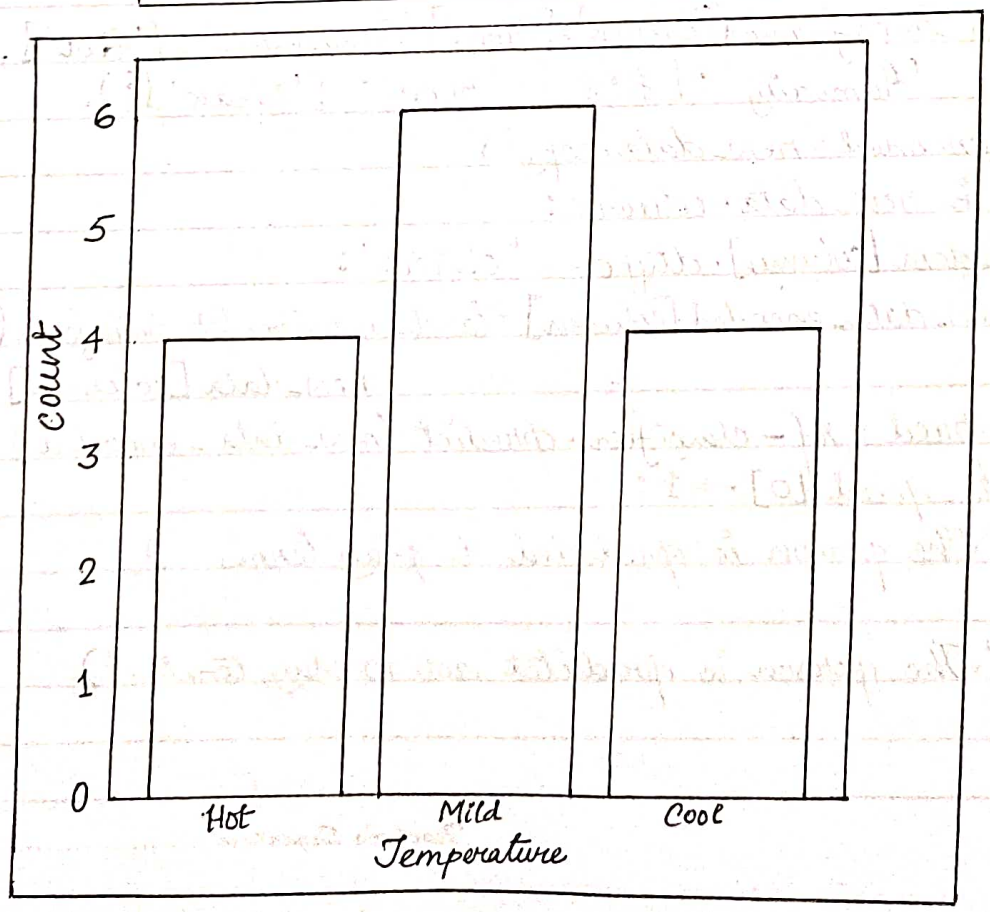
Page No. 16
Date

```
accuracy = accuracy_score (y_test, y_pred)
conf_matrix = confusion_matrix (y_test, y_pred)
classification_report_str = classification_report (y_test, y_pred)

print (f'RF Accuracy : {accuracy}')
print (f'RF Confusion Matrix : \n{confusion_matrix}')
sns.set (rc={'figure.figsize' : (6,3)})
sns.heatmap (confusion_matrix (y_test, y_pred), annot = True, fmt = 'd')
plt.xlabel ('Predicted Labels')
plt.ylabel ('Actual Labels')
print (f'RF Classification Report : \n{classification_report_str}')

new_data = pd.DataFrame ({'Outlook' : ['Sunny'], 'Temperature' : ['Hot'],
                          'Humidity' : ['High'], 'Wind' : ['Weak']})
new_data_encoded = new_data.copy()
for column in new_data.columns:
    if new_data[column].dtype == 'object':
        new_data_encoded[column] = label_encoder.fit_transform(
                                   new_data[column])
play_tennis_pred = rf_classifier.predict (new_data_encoded)
if play_tennis_pred[0] == 1:
    print ("The person is predicted to play tennis.")
else:
    print ("The person is predicted not to play tennis.")
```

Avijit®

Teacher's Signature :.....


5
4
3
2
1
0
Count
Sunny
Overcast
Rain
Outlook


6
5
4
3
2
1
0
Count
Hot
Mild
Cool
Temperature

Expt, No.

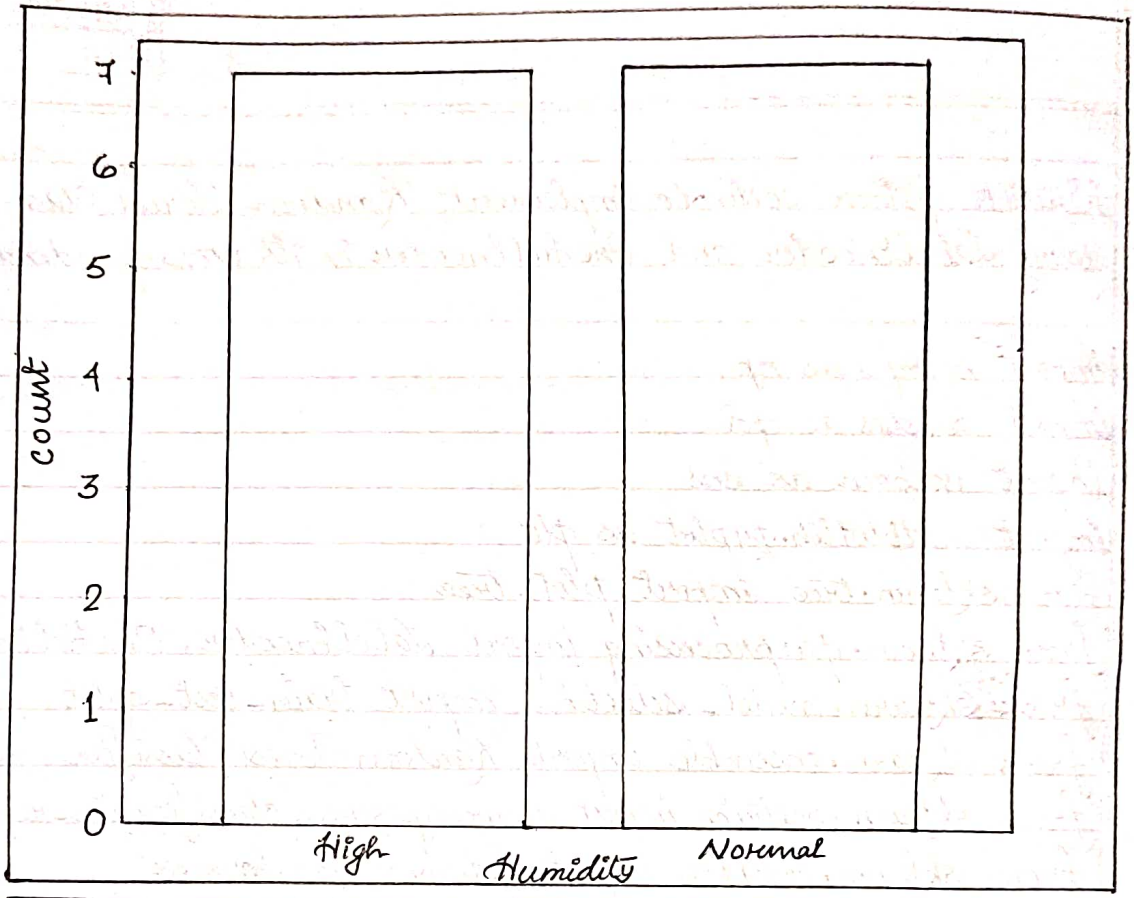
Page No. 17
Date

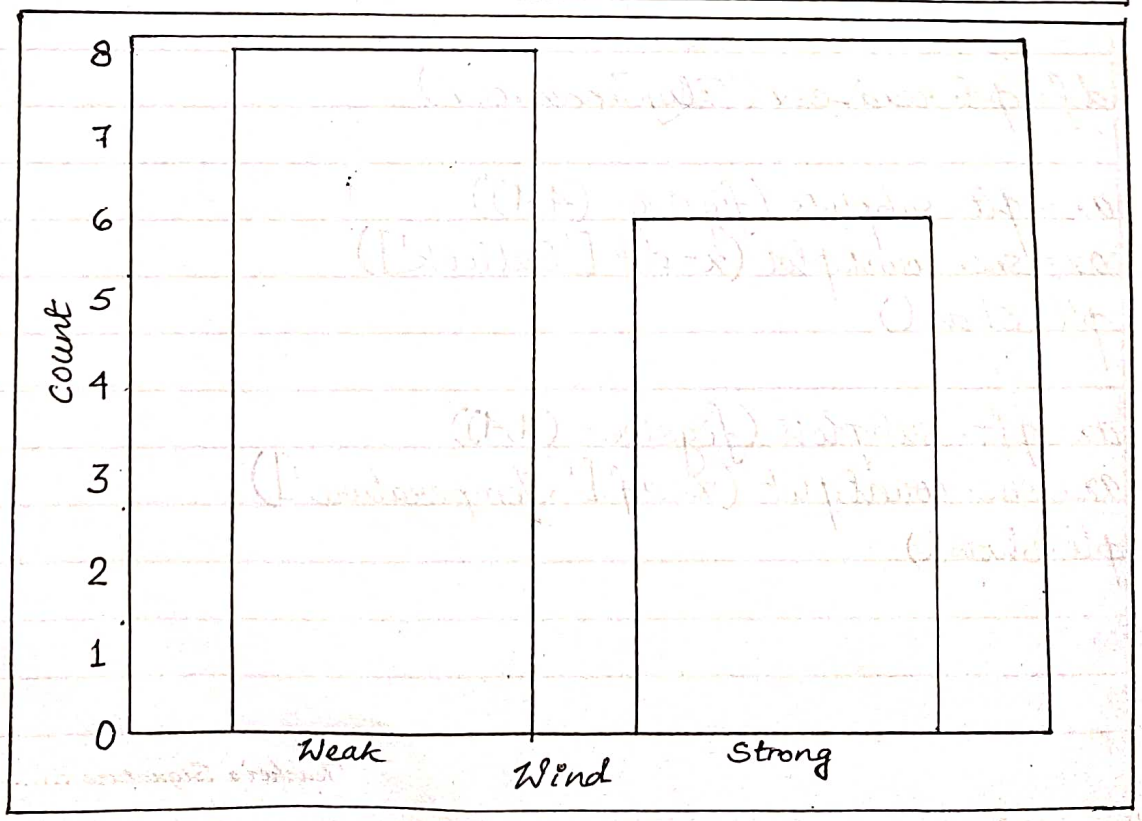
Write a python code to implement Random Forest Classifier algorithm using LabelEncoder and OneHotEncoder & its corresponding csv file.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.compose import ColumnTransformer

df = pd.read_csv('PlayTennis.csv')

ax = plt.subplots(figsize=(4,4))
ax = sns.countplot(x=df['Outlook'])
plt.show()

ax = plt.subplots(figsize=(4,4))
ax = sns.countplot(x=df['Temperature'])
plt.show()
```

Avijit®

Teacher's Signature :.....


7
6
5
4
3
2
1
0
count
High
Humidity
Normal


8
7
6
5
4
3
2
1
0
count
Weak
Wind
Strong

Expt, No.

Page No. 18
Date

```
ax = plt.subplots (figsize=(4,4))
ax = sns.countplot (x=df['Humidity'])
plt.show()

ax=plt.subplots (figsize= (4,4))
ax= sns.countplot (x=df['Wind'])
plt.show()

X = df.iloc [:, :-1]
y = df.iloc [:, -1]

categorical_features = ['Outlook', 'Temperature', 'Humidity', 'Wind']
preprocessor = ColumnTransformer ('transformers = [('encoder', OneHotEncoder(),
                    categorical_features)], remainder = 'passthrough')
X_encoded = preprocessor.fit_transform (X)

label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform (y)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y_encoded,
                                    test_size= 0.2, random_state =42)
rf_classifier = RandomForestClassifier (n_estimators=50)
rf_classifier.fit (X_train, y_train)

y_pred =rf_classifier.predict (X_test)
```

Avijit®

Teacher's Signature :.....

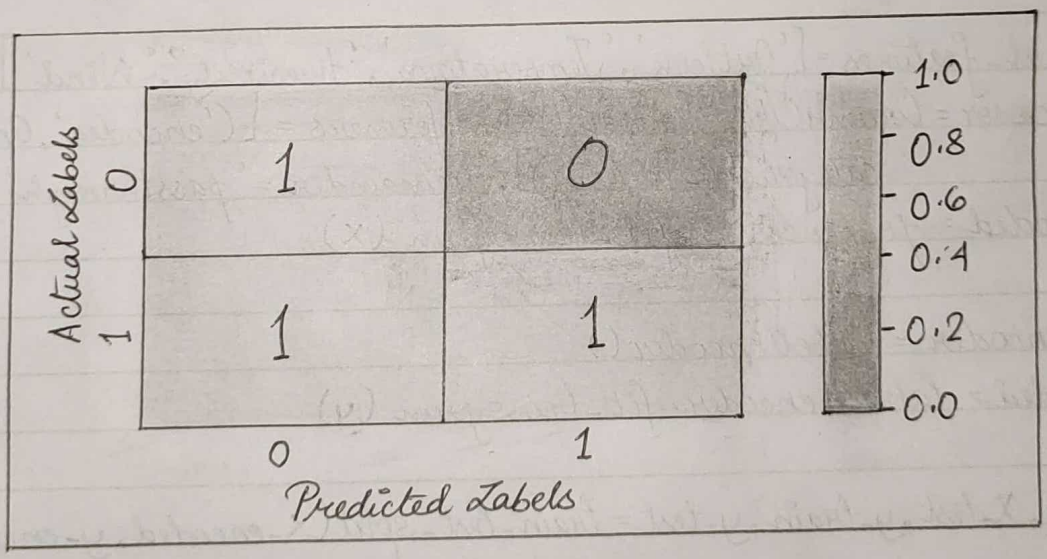
RF Accuracy : 0.6666666666666666

RF Confusion Matrix :

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

RF Classification Report :

	precision	recall	f1-score	support
0	0.50	1.00	0.67	1
1	1.00	0.50	0.67	2
accuracy			0.67	3
macro avg	0.75	0.75	0.67	3
weighted avg	0.83	0.67	0.67	3



The person is predicted not to play tennis.

```
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
classification_report_str = classification_report(y_test, y_pred)

print(f'RF Accuracy: {accuracy}')
print(f'RF Confusion Matrix:\n{conf_matrix}')
sns.set(rc={'figure.figsize': (6,3)})
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d')
plt.xlabel('Predicted Labels')
plt.ylabel('Actual Labels')
print(f'RF Classification Report:\n{classification_report_str}')

new_data = pd.DataFrame({'Outlook': ['Sunny'], 'Temperature': ['Hot'],
                         'Humidity': ['High'], 'Wind': ['Weak']})

new_data_encoded = preprocessor.transform(new_data)

predicted_play_tennis = rf_classifier.predict(new_data_encoded)

if predicted_play_tennis[0] == 1:
    print("The person is predicted to play tennis.")

else:
    print("The person is predicted not to play tennis.")
```


Teacher's Signature :.....