	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

Expt, No.

Page No. 20
Date 25/04/2024

Write a python code to implement Random Forest algorithm.

Random forest algorithm is a powerful tree learning technique in Machine Learning, which works by creating a no. of decision trees during the training phase.
Each tree is constructed using a random subset of dataset to measure a random subset of feature in each partition.
This randomness introduces a variability among individual trees, reducing the risk of overfitting and improving overall prediction performance. In prediction, the algorithm aggregates the results of all trees, either by voting (classification tasks) or by averaging (for regression tasks).

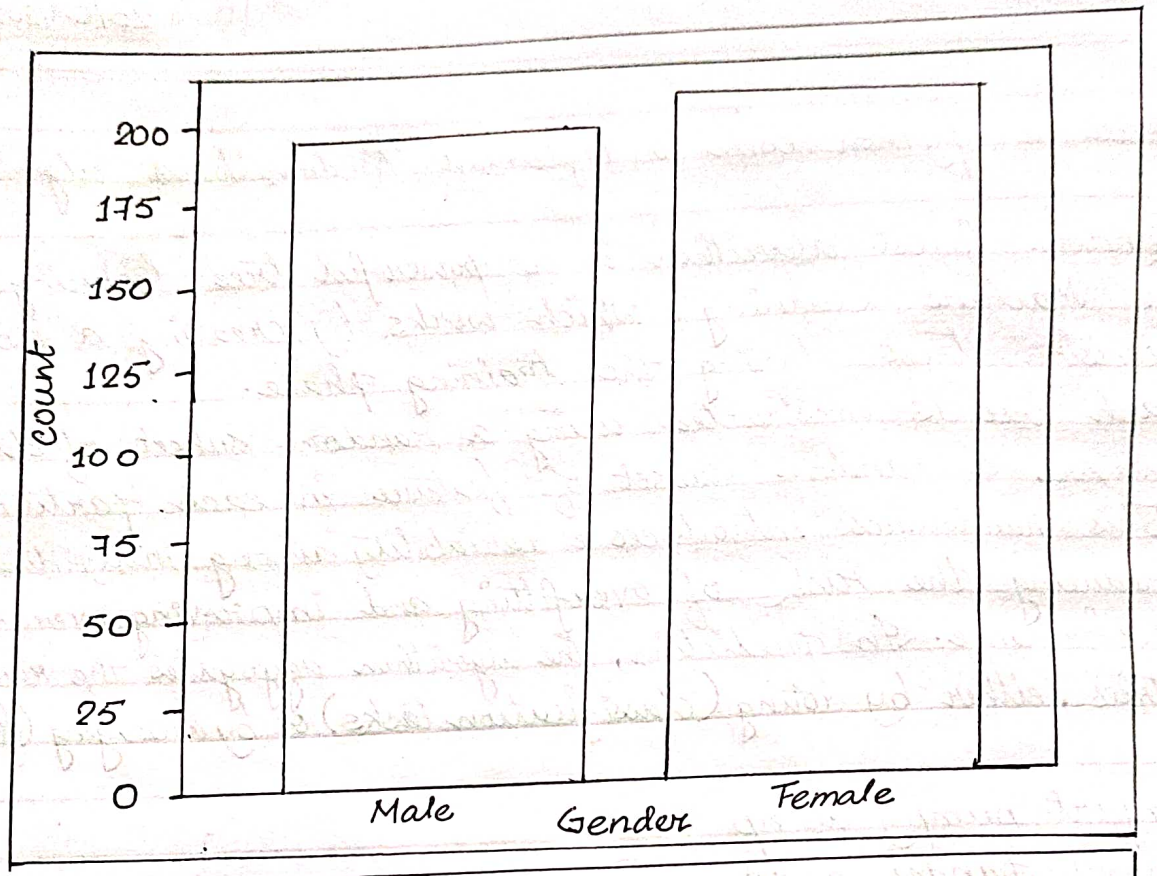
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.matrix import accuracy_score, classification_report,
                                                confusion_matrix
```

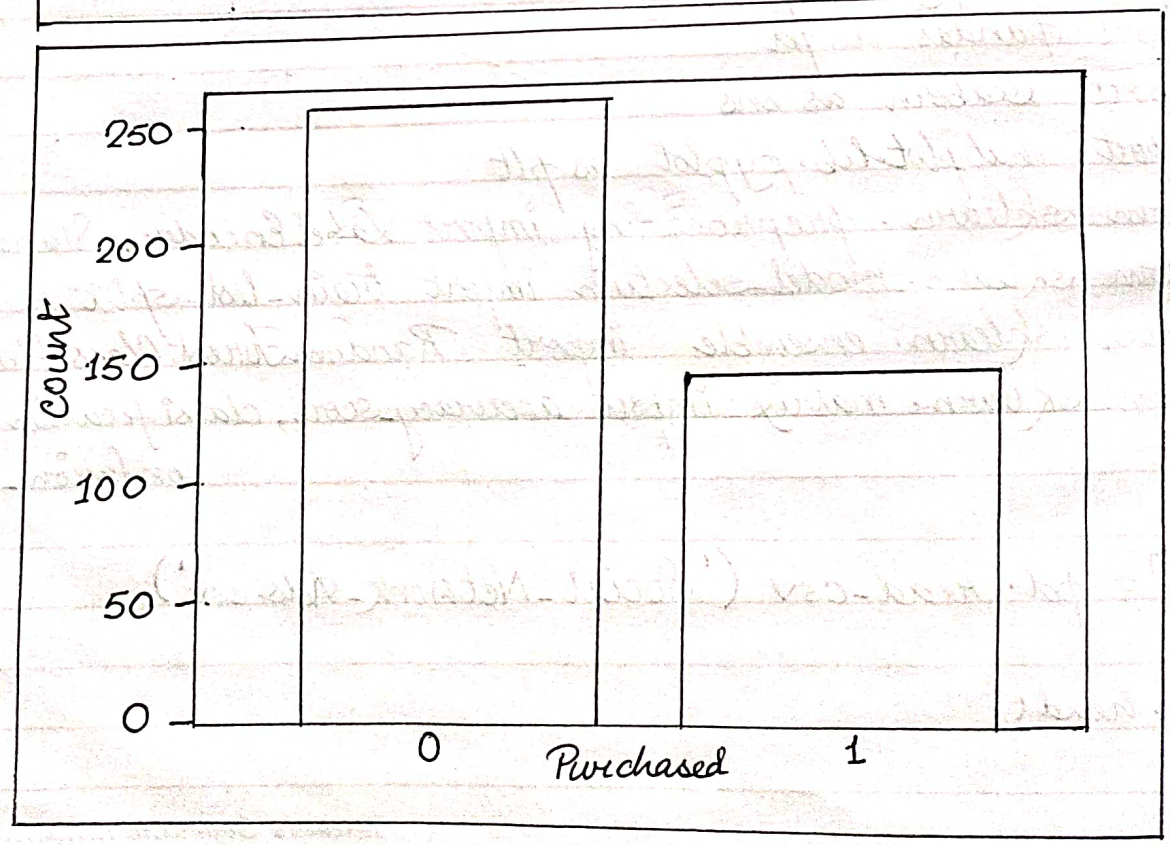
Avijit®

```
df = pd.read_csv('Social_Network_Ads.csv')
```

```
df.head()
```

Teacher's Signature :.....


200
175
150
125
100
75
50
25
0
count
Male
Gender
Female


250
200
150
100
50
0
count
0
Purchased
1

Expt, No.

Page No. 21
Date

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

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

X=df.iloc[:, [1,2,3]].values
y = df.iloc [:,4].values

label_encoder = LabelEncoder()
X[:,0] = label_encoder.fit_transform (X[:,0])

X_train, X_test, y_train, y_test = train_test_split (X, y, test_size=0.2,
                                   random_state =42)
```

Avijit®

Teacher's Signature :.....

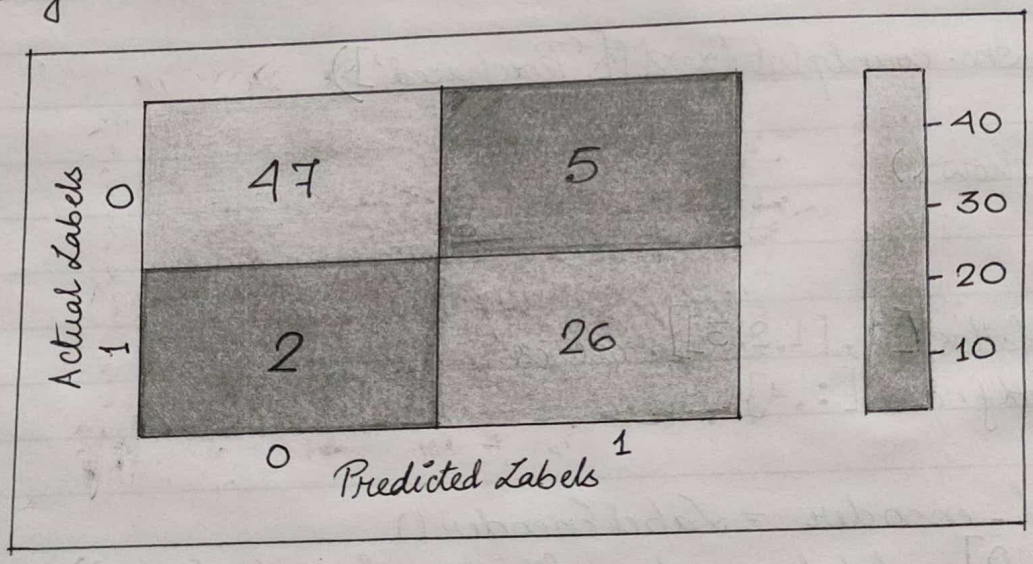
RF Accuracy : 0.9125

RF Confusion Matrix :

```
[[47  5]
 [ 2 26]]
```

RF Classification Report :

	precision	recall	f1-score	support
0	0.96	0.90	0.93	52
1	0.84	0.93	0.88	28
accuracy			0.91	80
macro avg	0.90	0.92	0.91	80
weighted avg	0.92	0.91	0.91	80



The targeted audience is predicted not to purchase the product.

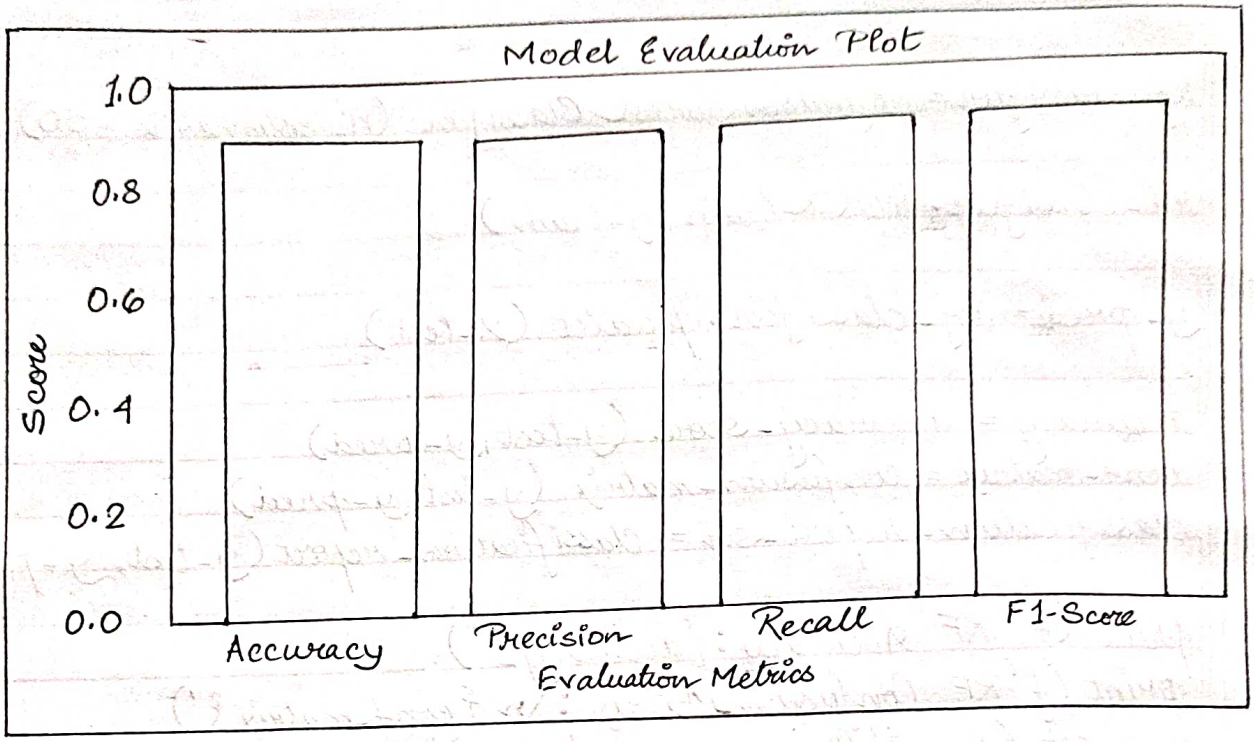
Expt, No.

Page No. 22
Date

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

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 = np.array([[0,30,50000]])
predicted_purchase = rf_classifier(new_data)

if predicted_purchase[0] == 1:
    print("The targeted audience is predicted to purchase the product.")
else:
    print("The targeted audience is predicted not to purchase the product.")
```

vijit®

Teacher's Signature :.....



Cross-Validation Results:
Individual Accuracies: [0.875 0.9 0.975 0.95 0.9 0.8 0.8 0.9 0.9 0.875]
Average Accuracy: 0.8875

Expt. No.

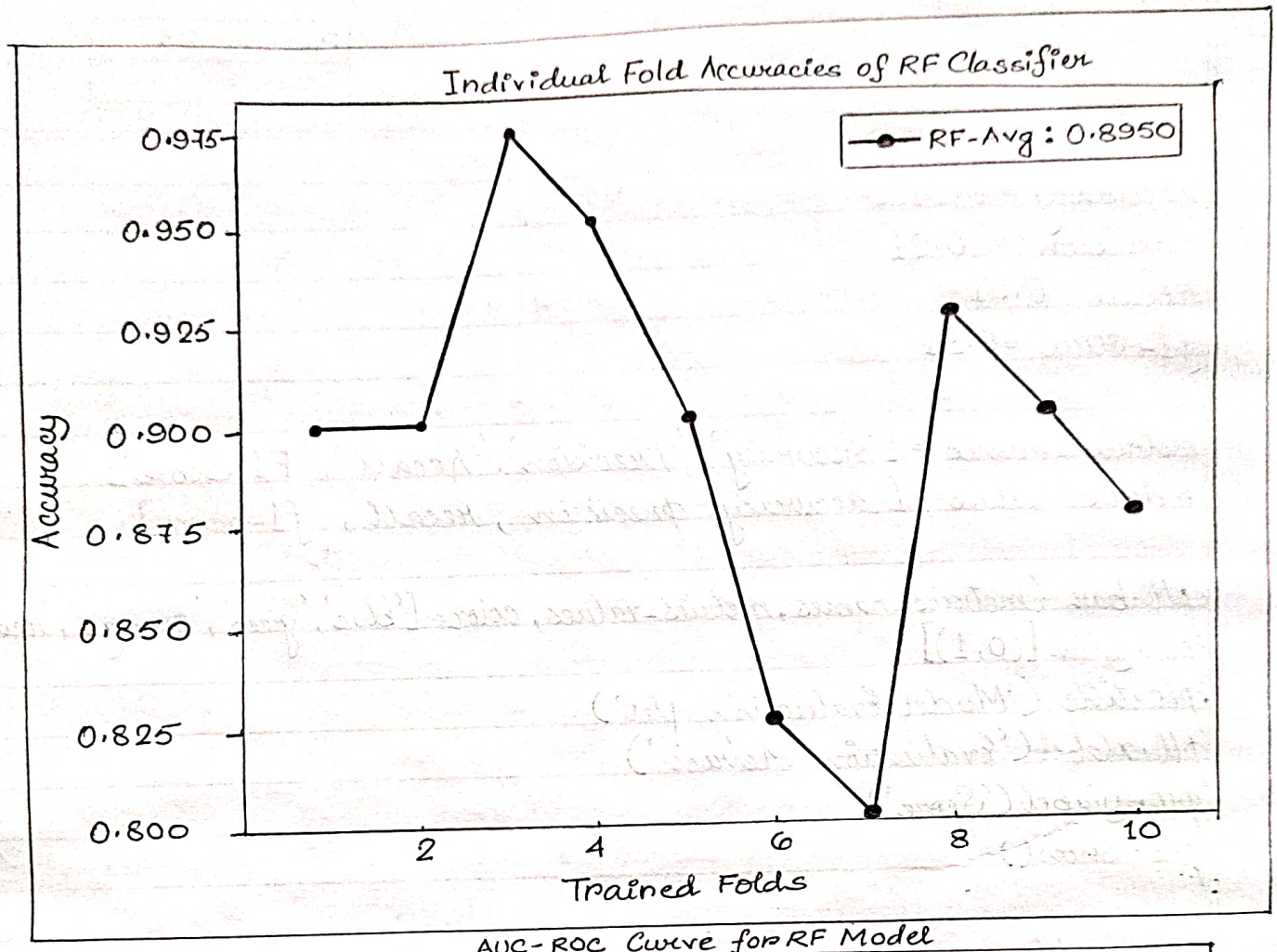
Page No. 23
Date

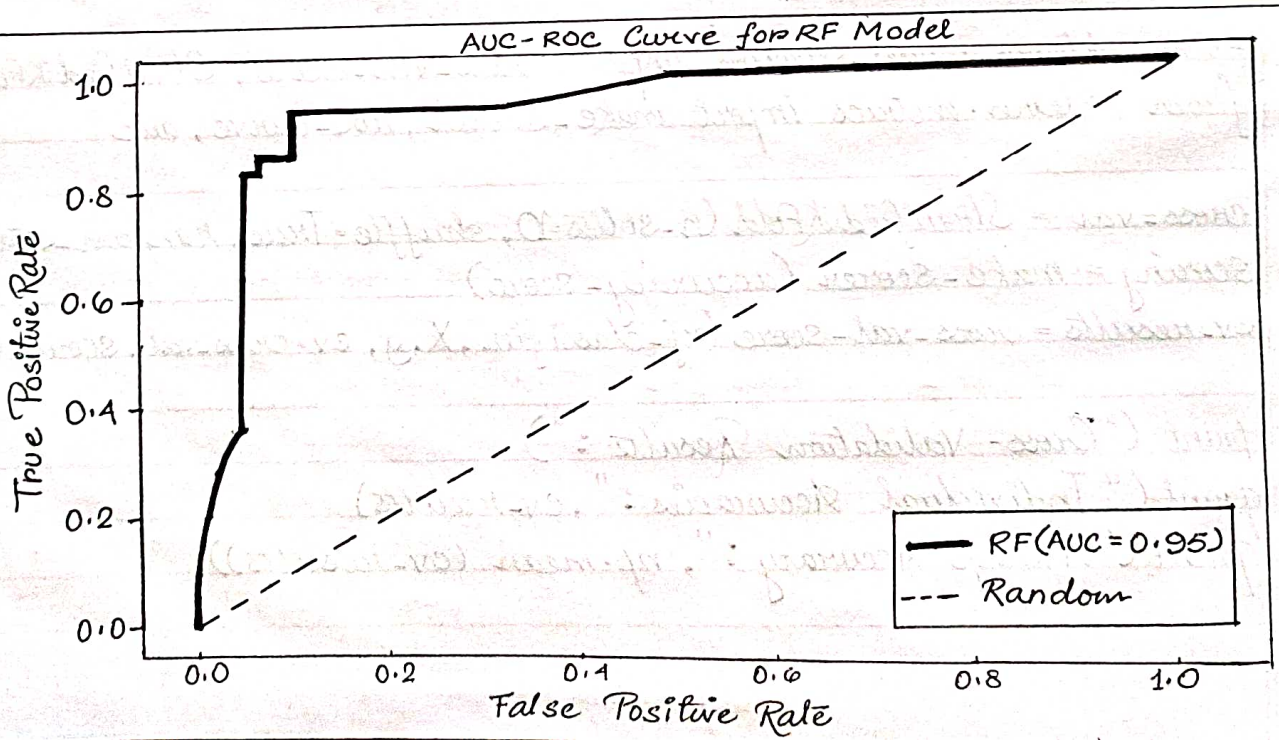
accuracy - 0.9125
precision = 0.91
recall - 0.91
f1-score = 0.91

```
metrics_names = ['Accuracy', 'Precision', 'Recall', 'F1-Score']
metrics_values = [accuracy, precision, recall, f1-score]

plt.bar(metrics_names, metrics_values, color=['blue', 'green', 'orange', 'red'])
plt.ylim[(0,1)]
plt.title('Model Evaluation plot')
plt.xlabel('Evaluation Metrics')
plt.ylabel('Score')
plt.show()

from sklearn.model_selection import cross_val_score, StraifiedKFold
from sklearn.metrics import make_scorer, roc_curve, auc

cross_val = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
scoring = make_scorer(accuracy_score)
cv_results = cross_val_score(rf_classifier, X, y, cv=cross_val, scoring=scoring)

print("Cross- Validation Results :")
print("Individual Accuracies :", cv_results)
print("Average Accuracy :", np.mean(cv_results))
```

Avijit®

Teacher's Signature :.....


Individual Fold Accuracies of RF Classifier
RF-Avg : 0.8950
Accuracy
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
2
4
6
8
10
Trained Folds


AUC-ROC Curve for RF Model
True Positive Rate
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate
RF(AUC = 0.95)
Random

```
model = ['RF']
accuracies = {'RF': [0.9, 0.9, 0.975, 0.95, 0.9, 0.825, 0.8, 0.925, 0.9, 0.875],}

plt.figure(figsize=(8,4))
for model in model:
    plt.plot(range(1,11), accuracies[model], marker='o', label=f'{model} -
    Avg: {sum(accuracies[model])/10:.4f}')

plt.title('Individual Fold Accuracies of RF Classifier')
plt.xlabel('Trained Folds')
plt.ylabel('Accuracy')
plt.legend(bbox_to_anchor=(1.05,1), loc='upper left')
plt.show()
label_encoder = LabelEncoder()
y_test_binary = label_encoder.fit_transform(y_test)
rf_predicted_scores = rf_classifier.predict_proba(X_test)[:,1]
rf_fpr, rf_tpr, _ = roc_curve(y_test_binary, rf_predicted_scores)
rf_roc_auc = auc(rf_fpr, rf_tpr)
plt.figure(figsize=(8,6))
sns.set(style='darkgrid')
plt.plot(rf_fpr, rf_tpr, color='purple', lw=2, label=f'RF (AUC = {rf_roc_auc:.2f})')
plt.plot([0,1],[0,1], linestyle='--', color='gray', label='Random')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('AUC-ROC Curve for RF Model')
plt.legend(loc='lower right')
plt.show()
```


Teacher's Signature :.....