In [35]:
```
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.metrics import recall_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.tree import DecisionTreeClassifier
```

In [36]:
```
df = pd.read_csv("customer_churn.csv")
```

In [37]:
```
df.head(5)
```

Out[37]:

	Unnamed: 0	SeniorCitizen	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_
0	0	0	29.85	29.85	0	1	
1	1	0	56.95	1889.50	0	0	
2	2	0	53.85	108.15	1	0	
3	3	0	42.30	1840.75	0	0	
4	4	0	70.70	151.65	1	1	

5 rows × 52 columns

In [38]:
```
df = df.drop("Unnamed: 0",axis = 1)
```

In [39]:
```
df.head(5)
```

Out[39]:

	SeniorCitizen	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_Male	Partn
0	0	29.85	29.85	0	1	0	
1	0	56.95	1889.50	0	0	1	
2	0	53.85	108.15	1	0	1	
3	0	42.30	1840.75	0	0	1	
4	0	70.70	151.65	1	1	0	

5 rows × 51 columns

creating x and y variables

In [40]:
```
x = df.drop("Churn", axis = 1)
```

In [41]:
```
x
```

Out[41]:

	SeniorCitizen	MonthlyCharges	TotalCharges	gender_Female	gender_Male	Partner_N
0	0	29.85	29.85	1	0	
1	0	56.95	1889.50	0	1	
2	0	53.85	108.15	0	1	
3	0	42.30	1840.75	0	1	
4	0	70.70	151.65	1	0	
...	...	...	...	...	...	
7027	0	84.80	1990.50	0	1	
7028	0	103.20	7362.90	1	0	
7029	0	29.60	346.45	1	0	
7030	1	74.40	306.60	0	1	
7031	0	105.65	6844.50	0	1	

7032 rows × 50 columns

In [42]:
```
y = df["Churn"]
```

In [43]:
```
y
```

Out[43]:
```
0       0
1       0
2       1
3       0
4       1
       ..
7027    0
7028    0
7029    0
7030    1
7031    0
Name: Churn, Length: 7032, dtype: int64
```

In [44]:
```
df["Churn"].value_counts()
```

Out[44]:
```
0    5163
1    1869
Name: Churn, dtype: int64
```

In [45]:
```
df
```

Out[45]:

	SeniorCitizen	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_Male	P
0	0	29.85	29.85	0	1	0	
1	0	56.95	1889.50	0	0	1	
2	0	53.85	108.15	1	0	1	
3	0	42.30	1840.75	0	0	1	
4	0	70.70	151.65	1	1	0	
...	...	...	...	...	...	...	
7027	0	84.80	1990.50	0	0	1	
7028	0	103.20	7362.90	0	1	0	
7029	0	29.60	346.45	0	1	0	
7030	1	74.40	306.60	1	0	1	
7031	0	105.65	6844.50	0	0	1	

7032 rows × 51 columns

In [ ]:
```
```

In [46]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25
```

Decision Tree classifier

In [47]:
```
model_dt = DecisionTreeClassifier(criterion = "gini", max_depth = 6, min_s
```

In [48]:
```
model_dt.fit(x_train,y_train)
```

Out[48]:
```
DecisionTreeClassifier(max_depth=6, min_samples_leaf=8)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [50]:
```
y_pred = model_dt.predict(x_test)
```

In [51]:
```
y_pred
```

Out[51]:
```
array([1, 0, 0, ..., 0, 0, 0], dtype=int64)
```

In [57]:
```
print(classification_report(y_test,y_pred,labels = [0,1]))
```

```
              precision    recall  f1-score   support

           0       0.83      0.89      0.86      1283
           1       0.63      0.50      0.56       475

    accuracy                           0.79      1758
   macro avg       0.73      0.70      0.71      1758
weighted avg       0.77      0.79      0.78      1758
```

In [55]:
```
confusion_matrix(y_test,y_pred)
```

Out[55]:
```
array([[1143,  140],
       [ 237,  238]], dtype=int64)
```

Recall = TP/(TP+FN)

Accuracy = (TP+TN)/TOTAL

Precision = TP/(TP+FP)

In [ ]:
```

```

Random Forest Classifier

In [82]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [87]:
```
model_rf = RandomForestClassifier(n_estimators = 100, criterion = "gini",
```

In [88]:
```
model_rf.fit(x_train,y_train)
y_pred_rf = model_rf.predict(x_test)
```

In [89]:
```
y_pred_rf
```

Out[89]:
```
array([1, 0, 0, ..., 0, 0, 0], dtype=int64)
```

In [ ]:
```

```

In [90]:
```
print(classification_report(y_test,y_pred_rf,labels = [0,1]))
```

```
              precision    recall  f1-score   support

           0       0.82      0.92      0.87      1283
           1       0.68      0.45      0.54       475

    accuracy                           0.79      1758
   macro avg       0.75      0.69      0.70      1758
weighted avg       0.78      0.79      0.78      1758
```

In [ ]:
```

```

In [91]:
```
# saving model
import pickle
```

In [93]:
```
filename = "model.sav"
pickle.dump(model_rf, open(filename, "wb"))
```

In [96]:
```
load_model = pickle.load(open(filename, "rb"))
```

In [ ]:
```

```

In [ ]:

```python
from flask import Flask, request, jsonify
import joblib
import numpy as np

app = Flask(__name__)

# Load the trained models
rf_model = joblib.load('random_forest_model.pkl')
dt_model = joblib.load('decision_tree_model.pkl')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()

    # Extract features from request data
    features = np.array([data['feature1'], data['feature2'], data['feature

    # Predict using both models
    rf_prediction = rf_model.predict(features)
    dt_prediction = dt_model.predict(features)

    # Return predictions as JSON
    return jsonify({
        'RandomForestPrediction': int(rf_prediction[0]),
        'DecisionTreePrediction': int(dt_prediction[0])
    })

if __name__ == '__main__':
    app.run(debug=True)
```