In [10]:

```
import pickle
import numpy as np
import warnings
import pandas as pd
import time
warnings.filterwarnings("ignore")
from sklearn.metrics import roc_curve,auc
import matplotlib.pyplot as plt
def final_1(X):

    '''This function take the input and preprocess it and then make predictions on it using pretrained models
       and return the predictions. '''
    start = time.time()
    if isinstance(X, pd.core.series.Series):
        values= list((X.to_frame()).iloc[:, 0])
        cols= list(X.to_frame().index)
        X= pd.DataFrame([values], columns = cols)
    column_names = pickle.load(open('column_names.pkl', 'rb'))
    X = X[column_names]
    X['count_zeros'] = (X == 0).astype(int).sum()
    X['var38'] = np.log(X['var38'])

    #predicting using 100 trained models
    df_test = pd.DataFrame()
    for i in range(1,101):
        df_test[f'pred_{i}'] = model_approach_1[f'model_{i}'].predict_proba(X)[:,1]

    #predicting using the meta model
    pred_meta = meta_model.predict_proba(df_test)[:,1]

    #predicting using 10 trained models
    pred = pd.DataFrame()
    for i in range(1,11):
        pred[f'pred_{i}'] = model_approach_2[f'model_{i}_xgb'].predict_proba(X)[:,1]

    i = 0
    for col in df_test.columns:
        if i == 0:
            ensemble = df_test[col]
            i += 1
        else:
            ensemble += df_test[col]
            i += 1
    ensemble /= 100

    prediction_1 = (0.3*pred_meta+ensemble*0.7)

    i = 0
    for col in pred.columns:
        if i == 0:
            prediction_2 = pred[col]
            i += 1
        else:
            prediction_2 += pred[col]
            i += 1
    prediction_2 /= 10

    prediction_2[X['var15'] < 23] = 0
    prediction_2[X['saldo_var30'] > 500000] = 0
    prediction_2[X['saldo_medio_var5_hace2'] > 160000] = 0

    final_prediction = (0.3*prediction_1+0.7*prediction_2)

    #Reference:
    #https://www.youtube.com/watch?v=D8HcmzYnBv0
    #Here the threshold value is selected after analyzing the ROC Curve on train data
    threshold = 0.049654096364974976

    if len(final_prediction) == 1:
        if final_prediction[0] > threshold:
            pred_final = 'Unsatisfied'
        else:
            pred_final = 'Satisfied'
        end = time.time()
        print(f"Time : {end-start} seconds")
        return pred_final
    else:
        pred_final = []
        for val in final_prediction:
            if val > threshold:
                pred_final.append('Unsatisfied')
            else:
                pred_final.append('Satisfied')
        end = time.time()
        print(f"Time : {end-start} seconds")
        return pred_final
```

In [11]:

```
from sklearn.metrics import roc_curve,auc
def final_2(X,y):

    '''This function takes the input and preprocess it and then make predictions on it using pretrained models
       and return the AUC score. '''
    start = time.time()
    column_names = pickle.load(open('column_names.pkl', 'rb'))
    X = X[column_names]
    X['count_zeros'] = (X == 0).astype(int).sum(axis=1)
    X['var38'] = np.log(X['var38'])

    #predicting using 100 trained models
    df_test = pd.DataFrame()
    for i in range(1,101):
        df_test[f'pred_{i}'] = model_approach_1[f'model_{i}'].predict_proba(X)[:,1]

    #predicting using the meta model
    pred_meta = meta_model.predict_proba(df_test)[:,1]

    #predicting using 10 trained models
    pred = pd.DataFrame()
    for i in range(1,11):
        pred[f'pred_{i}'] = model_approach_2[f'model_{i}_xgb'].predict_proba(X)[:,1]

    i = 0
    for col in df_test.columns:
        if i == 0:
            ensemble = df_test[col]
            i += 1
        else:
            ensemble += df_test[col]
            i += 1
    ensemble /= 100
    prediction_1 = (0.3*pred_meta+ensemble*0.7)

    i = 0
    for col in pred.columns:
        if i == 0:
            prediction_2 = pred[col]
            i += 1
        else:
            prediction_2 += pred[col]
            i += 1
    prediction_2 /= 10

    prediction_2[X['var15'] < 23] = 0
    prediction_2[X['saldo_var30'] > 500000] = 0
    prediction_2[X['saldo_medio_var5_hace2'] > 160000] = 0

    final_prediction = (0.3*prediction_1+0.7*prediction_2)
    test_fpr, test_tpr, tr_thresholds = roc_curve(y, final_prediction)
    auc_score = auc(test_fpr, test_tpr)
    end = time.time()
    print(f"Time : {end-start} seconds")
    return auc_score
```

In [12]:

```
import pickle
if __name__ == '__main__':

    #loading 100 trained models
    model_approach_1 = {}
    for i in range(1,101):
        model_approach_1[f'model_{i}'] = pickle.load(open(f'model_{i}.pkl', 'rb'))

    #loading meta model
    meta_model = pickle.load(open('meta_model.pkl', 'rb'))

    #loading 10 trained models
    model_approach_2 = {}
    for i in range(1,11):
        model_approach_2[f'model_{i}_xgb'] = pickle.load(open(f'model_{i}_xgb.pkl', 'rb'))

    print("----------------------------Output From final_1 function----------------------------------")
    df = pd.read_csv('train.csv')
    sample_unsatisfied = df[df['TARGET'] == 1].iloc[0]
    print(f'Customer is {final_1(sample_unsatisfied)}')
    sample_satisfied = df[df['TARGET'] == 0].iloc[0]
    print(f'Customer is {final_1(sample_satisfied)}')
    print()
    print("----------------------------Output From final_2 function----------------------------------")
    X = pd.read_csv('train.csv')
    y = X['TARGET']
    X = X.drop(['TARGET'],axis=1)
    print(f'AUC Score: {final_2(X,y)}')
```

```
----------------------------Output From final_1 function----------------------------------
Time : 0.840299129486084 seconds
Customer is Unsatisfied
Time : 0.7592098712921143 seconds
Customer is Satisfied

----------------------------Output From final_2 function----------------------------------
Time : 66.61568021774292 seconds
AUC Score: 0.8910933473956772
```

In [ ]: