LEAD SCORING CASE STUDY

LE THANH DAT

DATA PREPROCESSING, FEATURE SELECTION, MODEL OPTIMIZATION

NECESSARY LIBRARIES

- pandas, numpy
- seaborn, mathplotlib
- sklearn.metrics
- optbinning
- xgboost

```
import pandas as pd
from optbinning import BinningProcess
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.metrics import accuracy_score, classification_report
```

```
from xgboost import XGBClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import accuracy_score, classification_report, roc_curve, roc_auc_score
import matplotlib.pyplot as plt
```

DATA EXLORATION

- Check duplicated/missing values
- Handle outliers

Numeric & categorical variables

```
def choose_numeric_categorical(data):
    numeric_vars = data.select_dtypes(include=['number']).columns.tolist()
    categorical_vars = data.select_dtypes(include=['object', 'category']).columns.tolist()
    return numeric_vars, categorical_vars
numeric_vars, categorical_vars = choose_numeric_categorical(data)
```

Check missing values

```
missing_values_percentage = round(100 * (data.isna().sum() / len(data)), 2)
sorted_missing_values_percentage = missing_values_percentage.sort_values(ascending=False)
sorted_missing_values_percentage
```

Data exploration

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

def plot_numeric_variable_analysis(numeric_columns, dataframe):
    for numeric_column in numeric_columns:
        plt.figure(figsize=(12, 6))
        print(numeric_column)

        # Calculate quartiles
        quartiles = np.percentile(dataframe[numeric_column].dropna(), [25, 50, 75, 90, 99])
        print(f'25th percentile: {quartiles[0]}')
        print(f'50th percentile: {quartiles[1]}')
        print(f'75th percentile: {quartiles[2]}')
        print(f'90th percentile: {quartiles[3]}')
        print(f'99th percentile: {quartiles[4]}')

        # Plot Violin plot
        plt.subplot(1, 2, 1)
        sns.violinplot(y=dataframe[numeric_column])
        plt.title(f'Violin Plot of {numeric_column}')
        plt.ylabel(numeric_column)

        # Plot CDF (Cumulative Distribution Function)
        plt.subplot(1, 2, 2)
        sorted_values = np.sort(dataframe[numeric_column].dropna())
        cdf_values = np.arange(1, len(sorted_values) + 1) / len(sorted_values)
        plt.plot(sorted_values, cdf_values, marker='.', linestyle='none')
        plt.title(f'CDF of {numeric_column}')
        plt.xlabel(numeric_column)
        plt.ylabel('CDF')

        plt.tight_layout()
        plt.show()

plot_numeric_variable_analysis(numeric_vars, data)
```

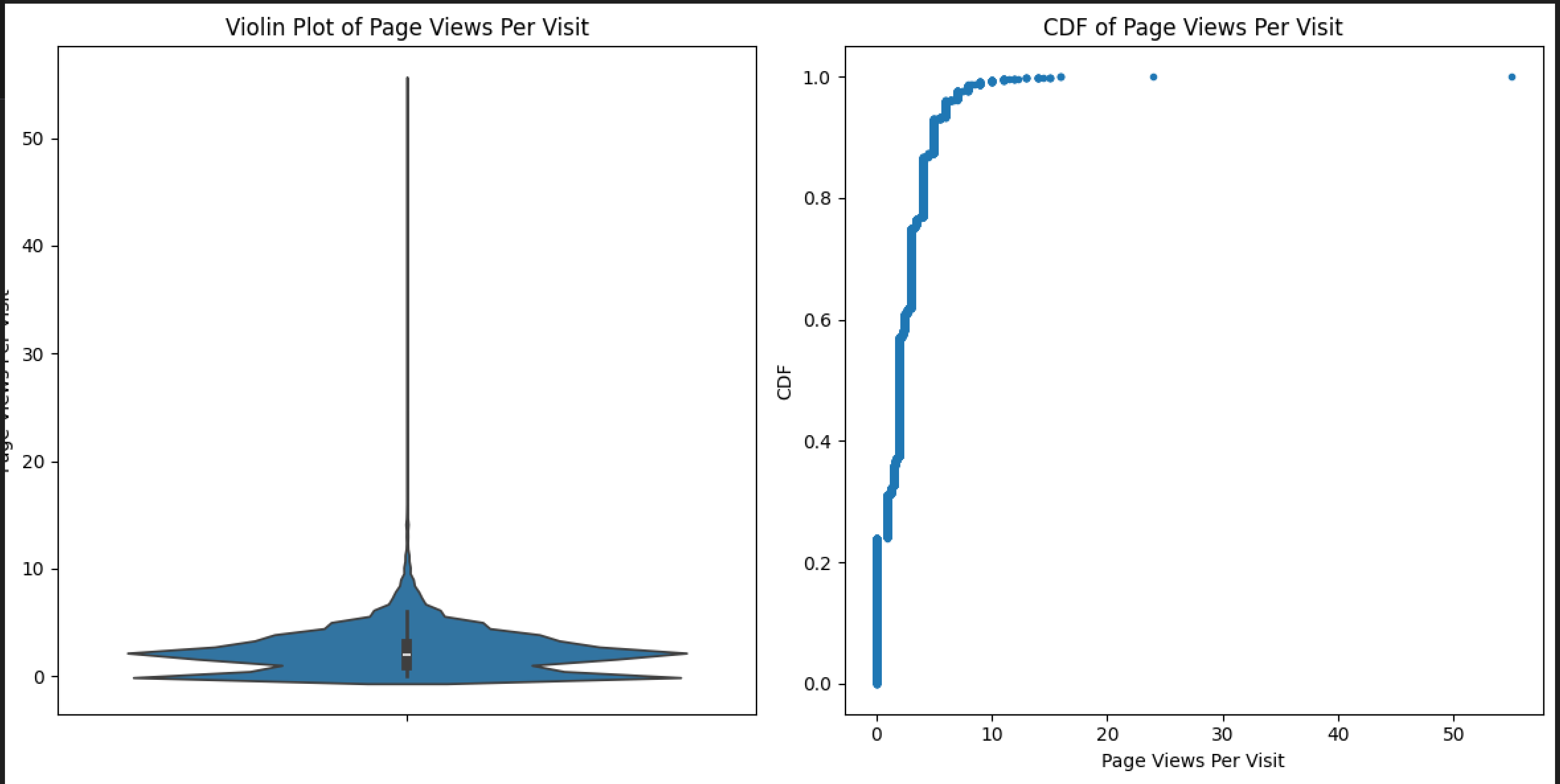
DATA EXLORATION

The images show that there are numerous NaN values across the columns. Notably, the Page Views Per Visit column contains some outliers.

```
Page Views Per Visit
25th percentile: 1.0
50th percentile: 2.0
75th percentile: 3.0
90th percentile: 5.0
99th percentile: 9.0
```

```python
missing_values_percentage = round(100 * (data.isna().sum() / len(data)), 2)
sorted_missing_values_percentage = missing_values_percentage.sort_values(ascending=False)
sorted_missing_values_percentage
```

Lead Quality	51.59
Asymmetrique Activity Index	45.65
Asymmetrique Profile Score	45.65
Asymmetrique Profile Index	45.65
Asymmetrique Activity Score	45.65
Tags	36.29
Lead Profile	29.32
What matters most to you in choosing a course	29.32
What is your current occupation	29.11
Country	26.63
How did you hear about X Education	23.89
Specialization	15.56
City	15.37
Page Views Per Visit	1.48
TotalVisits	1.48
Last Activity	1.11
Lead Source	0.39
I agree to pay the amount through cheque	0.00
A free copy of Mastering The Interview	0.00
Get updates on DM Content	0.00
Update me on Supply Chain Content	0.00
Lead Origin	0.00
X Education Forums	0.00
Receive More Updates About Our Courses	0.00
Through Recommendations	0.00
...	
Converted	0.00
Do Not Call	0.00
Do Not Email	0.00
Last Notable Activity	0.00



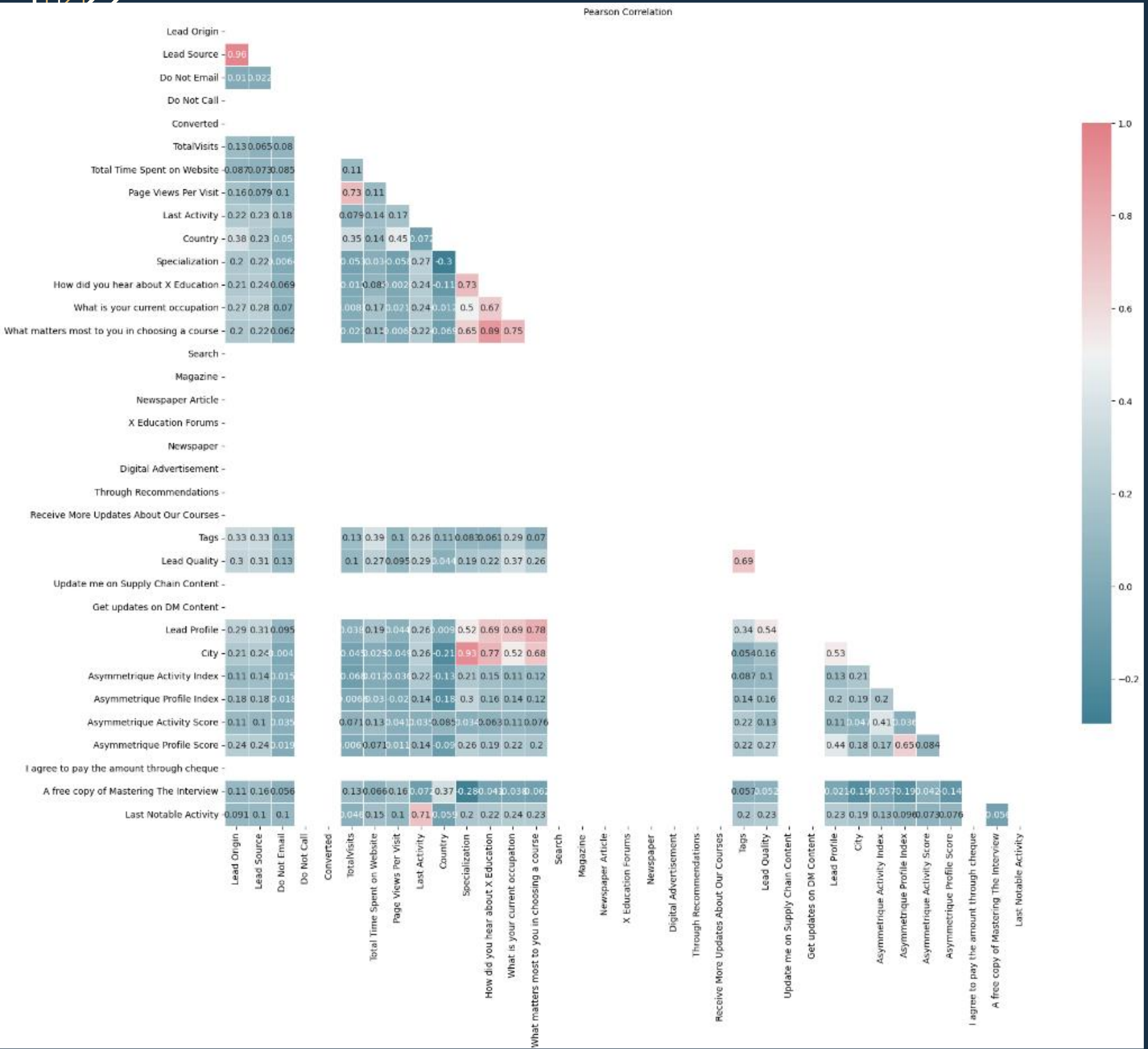
FEATURE SELECTION USING CORRELATION AND IV

- Pearson correlation to check multi-collinearity: from the Lead data, export to iv_table.csv and binning_table.csv to identify which variables have the highest IV and should be included

A	B	C	D	E	F
	variable	iv	unique_bin	top_bin	freq_bin
0	Lead Origin	0.609526843	3	[1]	3903
1	Lead Source	0.658412573	5	[3]	2295
2	Do Not Email	0.108353799	2	[0]	6794
3	Do Not Call	0	1	[0 1]	7392
4	Last Activity	0.845715513	4	[3 13 5]	3054
5	Country	0.018164662	2	[0 36 32 30 29 28 27	5338
6	Specialization	0.384737403	5	[14 15 17 6 1 11]	2486
7	How did you hear about X Education	0.478848744	4	[6 1 0]	4122
8	What is your current occupation	1.007207474	3	[3 4 0 2]	4694
9	What matters most to you in choosing a course	0.572587389	2	[0 1]	5249
10	Search	0	1	[1 0]	7392
11	Magazine	0	1	[0]	7392
12	Newspaper Article	0	1	[1 0]	7392
13	X Education Forums	0	1	[1 0]	7392
14	Newspaper	0	1	[1 0]	7392
15	Digital Advertisement	0	1	[1 0]	7392
16	Through Recommendations	0	1	[0 1]	7392
17	Receive More Updates About Our Courses	0	1	[0]	7392
18	Tags	4.824129566	5	[16 26 20 5 18 15 1]	2842
19	Lead Quality	2.008333608	5	[5 3]	4664
20	Update me on Supply Chain Content	0	1	[0]	7392
21	Get updates on DM Content	0	1	[0]	7392
22	Lead Profile	1.088576106	4	[4]	3314
23	City	0.356866941	5	[6 0]	2645
24	Asymmetrique Activity Index	0.0756598	3	[3]	3355
25	Asymmetrique Profile Index	0.073583573	3	[3]	3355
26	I agree to pay the amount through cheque	0	1	[0]	7392
27	A free copy of Mastering The Interview	0.005845037	2	[0]	5086
28	Last Notable Activity	0.661165701	4	[6 9 1 8]	2931
29	Converted	0	1	(-inf, inf)	7392
30	TotalVisits	0.081661091	6	[3.50, 7.50)	2161
31	Total Time Spent on Website	1.065929153	5	[1.50, 416.50)	2860
32	Page Views Per Visit	0.059326374	5	[2.04, inf)	3115
33	Asymmetrique Activity Score	0.383067713	5	Missing	3355
34	Asymmetrique Profile Score	0.182606501	5	Missing	3355

		Bin	Count	Count (%)	Non-event	Event	Event rate	WoE	IV	JS
Lead Origin	0	[3 0]	2911	0.393804113	2020	891	0.306080385	0.351888832	0.046413712	0.005771965
Lead Origin	1	[1]	3903	0.528003247	2484	1419	0.363566487	0.093298235	0.004544161	0.000567814
Lead Origin	2	[2 4]	578	0.078192641	39	539	0.932525952	-3.092773456	0.558568969	0.050879756
Lead Origin	Totals		7392	1	4543	2849	0.385416667		0.609526843	0.057219535
Lead Source	0	[18 9 19 17 2 6 11 15]	1554	0.210227273	1176	378	0.243243243	0.668360402	0.084334819	0.01034992
Lead Source	1	[1]	2049	0.277191558	1377	672	0.327964861	0.250784627	0.016860612	0.002102071
Lead Source	2	[7]	913	0.123511905	569	344	0.376779847	0.036619245	0.000164915	2.06E-05
Lead Source	3	[3]	2295	0.310470779	1381	914	0.398257081	-0.053886949	0.000906926	0.000113352
Lead Source	4	[21 10 14 13 4 0]	581	0.078598485	40	541	0.931153184	-3.071159356	0.5561453	0.050839466
Lead Source	Totals		7392	1	4543	2849	0.385416667		0.658412573	0.063425422
Do Not Email	0	[1]	598	0.080898268	508	90	0.150501672	1.264052246	0.101415356	0.011895085
Do Not Email	1	[0]	6794	0.919101732	4035	2759	0.406093612	-0.086481525	0.006938443	0.000867035
Do Not Email	Totals		7392	1	4543	2849	0.385416667		0.108353799	0.01276212
Do Not Call	0	[0 1]	7392	1	4543	2849	0.385416667	-2.22E-16	0	0
Do Not Call	Totals		7392	1	4543	2849	0.385416667		0	0
Last Activity	0	[16 2 9 1]	1375	0.186011905	1248	127	0.092363636	1.818490931	0.418491678	0.046118078
Last Activity	1	[15 7 10 14]	679	0.091856061	528	151	0.222385862	0.785196916	0.04964149	0.006050539
Last Activity	2	[3 13 5]	3054	0.413149351	1965	1089	0.356581532	0.12361287	0.006217009	0.000776632
Last Activity	3	[12 8 17 6 11 0]	2284	0.308982684	802	1482	0.648861646	-1.080658729	0.371365337	0.044286038
Last Activity	Totals		7392	1	4543	2849	0.385416667		0.845715513	0.097231286
Country	0	[0 36 32 30 29 28 27 24 22 20 18 16 15 14 19 10 13 6 5 4 25 23 1 35	5338	0.722132035	3387	1951	0.365492694	0.084982981	0.005161943	0.000645049
Country	1	[33 38 26 2 8 21 11 3 7]	2054	0.277867965	1156	898	0.437195716	-0.21406855	0.013002718	0.001622243
Country	Totals		7392	1	4543	2849	0.385416667		0.018164662	0.002267292
Specialization	0	[19]	1136	0.153679654	1022	114	0.100352113	1.726698791	0.31934852	0.03559808
Specialization	1	[16 2 18 13 3 8 9]	855	0.115665584	547	308	0.360233918	0.107729488	0.00132474	0.000165512
Specialization	2	[14 15 17 6 1 11]	2486	0.336309524	1445	1041	0.418744972	-0.138691999	0.006562851	0.000819699
Specialization	3	[4 7 0]	1734	0.234577922	936	798	0.460207612	-0.307112652	0.02274692	0.002832243
Specialization	4	[12 10 5]	1181	0.159767316	593	588	0.49788315	-0.45815208	0.034754372	0.004306696
Specialization	Totals		7392	1	4543	2849	0.385416667		0.384737403	0.04372223
How did you hear about X Education	0	[10 5]	1761	0.238230519	1538	223	0.126632595	1.464446847	0.381151193	0.043796385
How did you hear about X Education	1	[2 4 7 9]	601	0.081304113	360	241	0.400998336	-0.065312433	0.00034931	4.37E-05
How did you hear about X Education	2	[3 8]	908	0.122835498	517	391	0.43061674	-0.187284217	0.004389883	0.000547935
How did you hear about X Education	3	[6 1 0]	4122	0.55762987	2128	1994	0.483745754	-0.401579631	0.092958359	0.011542341
How did you hear about X Education	Totals		7392	1	4543	2849	0.385416667		0.478848744	0.055930317

FEATURE SELECTION USING CORRELATION AND IV

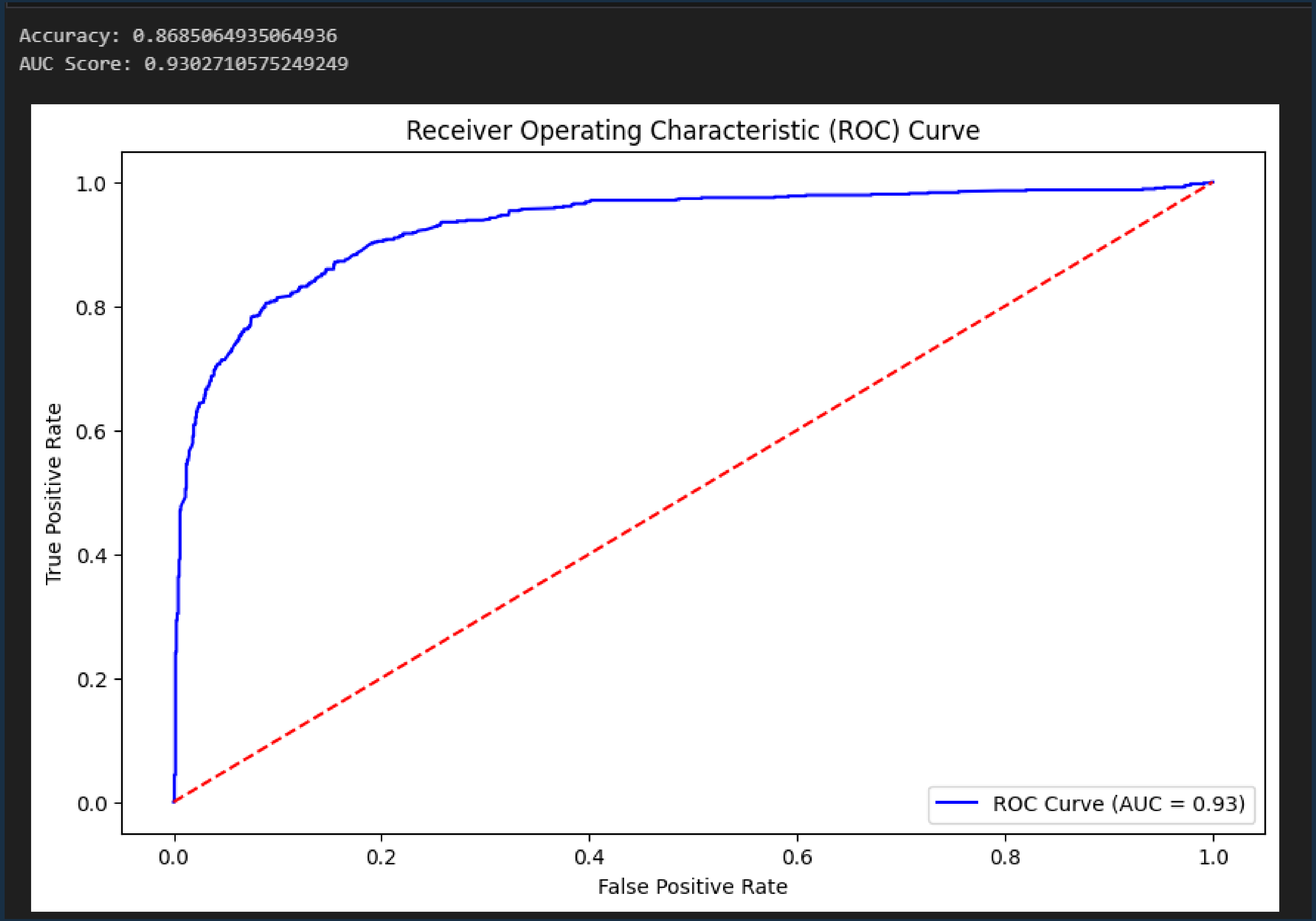


Using heatmap visualization and remove highly correlated with low IV feature values:

Certain conditions must be applied to filter for useful variables: correlation should be less than 0.6, and if it exceeds 0.6, the variable with the lower IV should be removed. Additionally, only variables with an IV greater than 0.6 should be selected.

MODEL PERFORMANCE SUMMARY

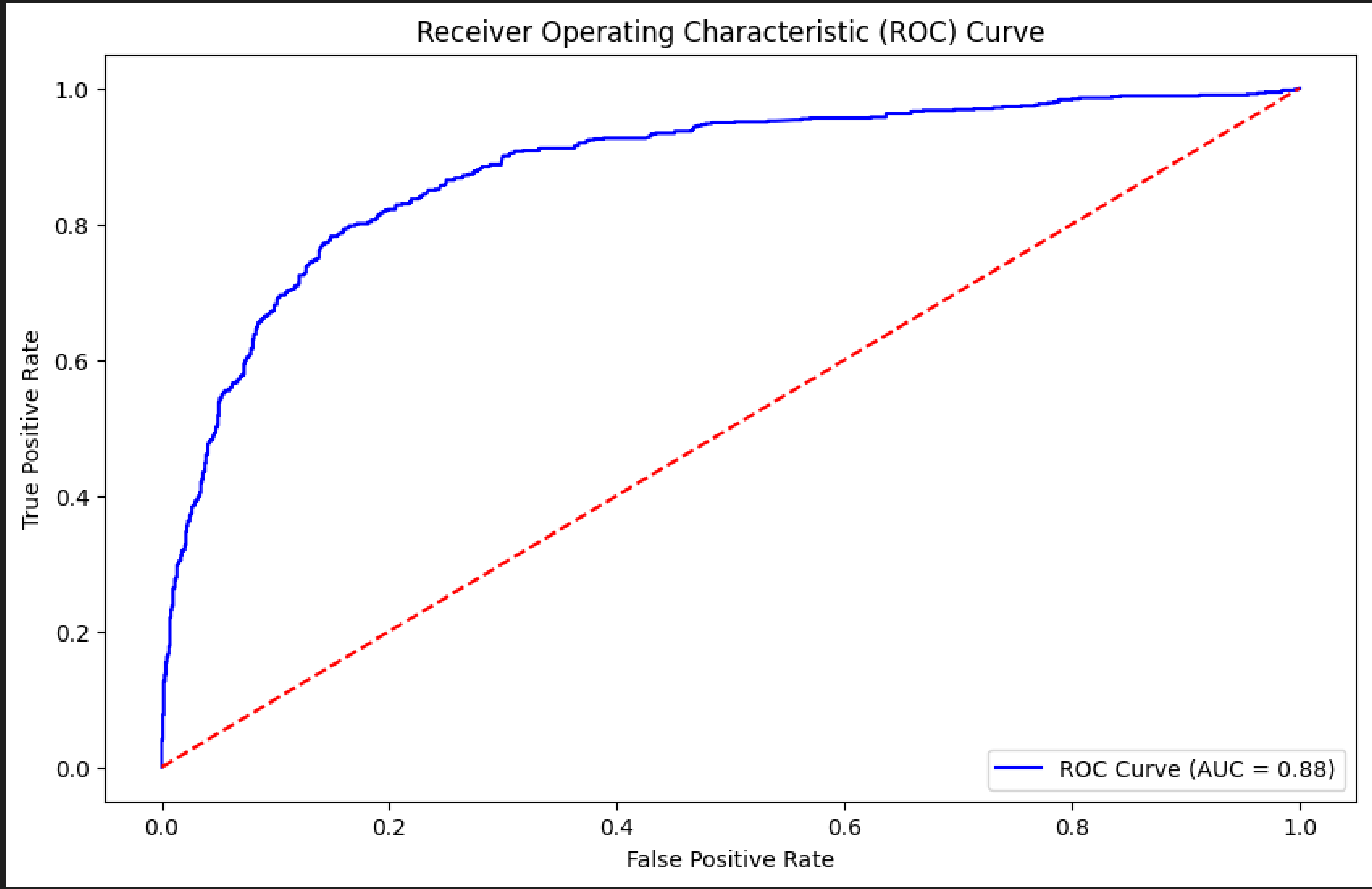
- Optimized Logistic Regression
  - Accuracy: 0.86851
  - AUC Score: 0.93027



MODEL PERFORMANCE SUMMARY

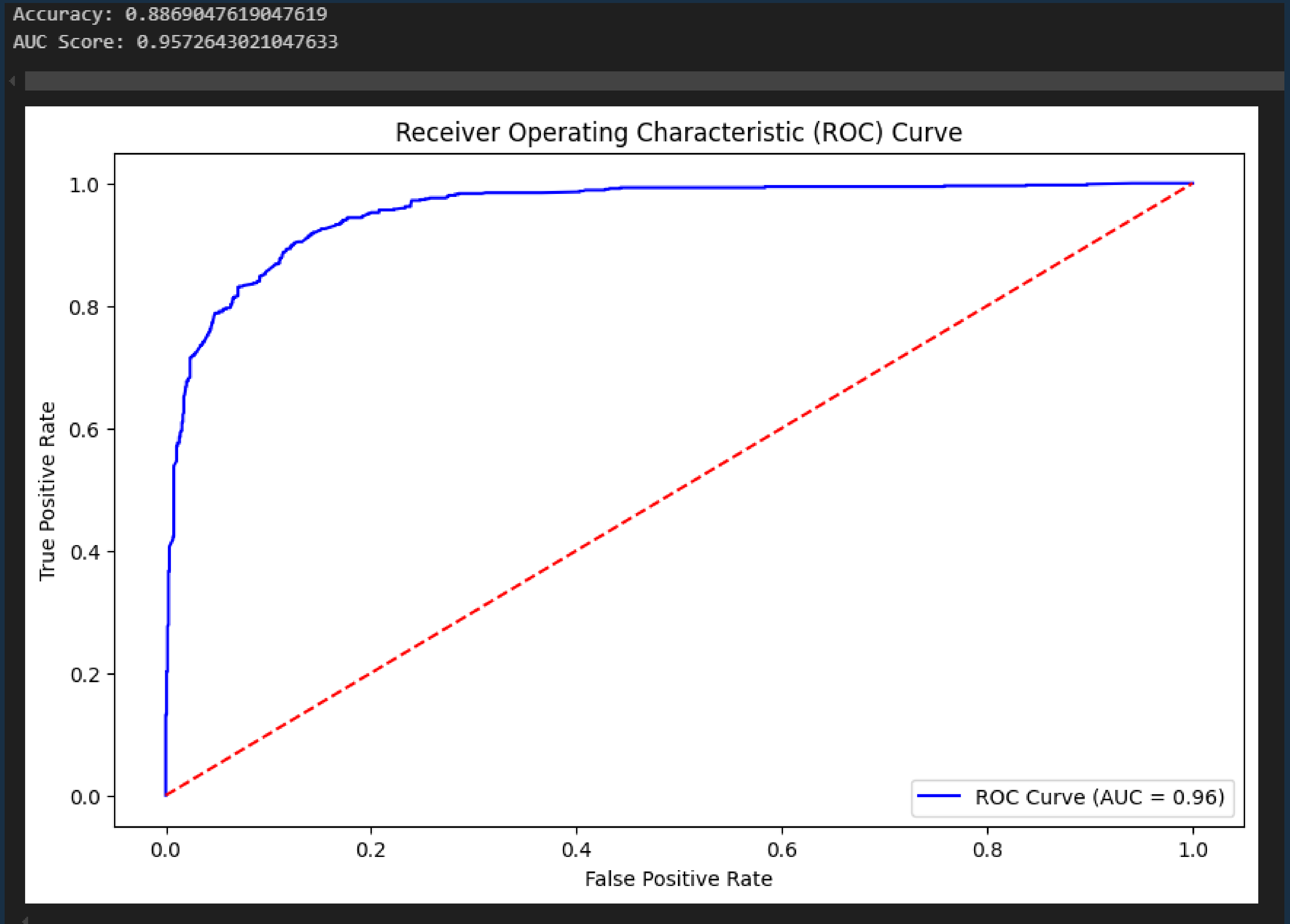
- Baseline Logistic Regression
  - Accuracy: 0.81818
  - AUC Score: 0.88224

```
Accuracy: 0.8181818181818182
AUC Score: 0.8822468695600569
```



MODEL PERFORMANCE SUMMARY

- Optimized XGBoost
  - Accuracy: 0.88690
  - AUC Score: 0.95726



XGBOOST THRESHOLD OPTIMIZATION

Update threshold for accurarcy optimzation

- Highest optimal threshold: 0.649999
- Highest accuracy: 0.888528

```python
# Define the range of thresholds to evaluate
threshold_values = np.arange(0.4, 0.9, 0.05)

# Get the predicted probabilities from the best model
y_test_pred_probabilities = best_xgb_model.predict_proba(x_test_data)[:, 1]

# Initialize variables to store the best threshold and the corresponding accuracy
optimal_threshold = 0.0
highest_accuracy = 0.0

# Iterate through each threshold value
for threshold in threshold_values:
    # Apply the threshold to get binary predictions
    y_threshold_predictions = (y_test_pred_probabilities >= threshold).astype(int)

    # Calculate accuracy for the current threshold
    current_accuracy = accuracy_score(y_test_labels, y_threshold_predictions)

    # Update the best threshold if the current accuracy is higher
    if current_accuracy > highest_accuracy:
        highest_accuracy = current_accuracy
        optimal_threshold = threshold

# Output the best threshold and corresponding accuracy
print(f"Optimal Threshold: {optimal_threshold}")
print(f"Highest Accuracy: {highest_accuracy}")
```

[27]

```
Optimal Threshold: 0.6499999999999999
Highest Accuracy: 0.8885281385281385
```