

---

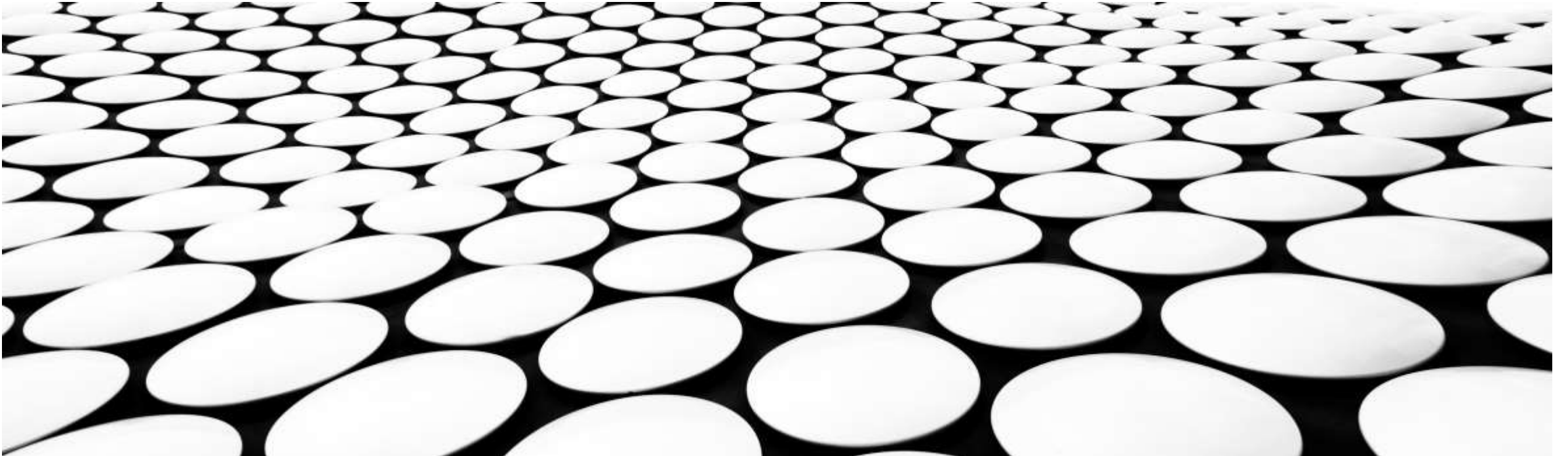
# LEAD SCORING CASE STUDY

SUBMITTED BY –

SONALI JAIN

SREEJIT BANERJEE

SRIDHAR





# PROBLEM STATEMENT

An education company named X Education sells online courses to industry professionals. On any given day, many professionals who are interested in the courses land on their website and browse for courses.

The company markets its courses on several websites and search engines like Google. Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos. When these people fill up a form providing their email address or phone number, they are classified to be a lead. Moreover, the company also gets leads through past referrals. Once these leads are acquired, employees from the sales team start making calls, writing emails, etc. Through this process, some of the leads get converted while most do not. The typical lead conversion rate at X education is around 30%.

Now, although X Education gets a lot of leads, its lead conversion rate is very poor. For example, if, say, they acquire 100 leads in a day, only about 30 of them are converted. To make this process more efficient, the company wishes to identify the most potential leads, also known as 'Hot Leads'. If they successfully identify this set of leads, the lead conversion rate should go up as the sales team will now be focusing more on communicating with the potential leads rather than making calls to everyone.

A typical lead conversion process can be represented using the following funnel:

As we can see, there are a lot of leads generated in the initial stage (top) but only a few of them come out as paying customers from the bottom. In the middle stage, you need to nurture the potential leads well (i.e. educating the leads about the product, constantly communicating etc. ) in order to get a higher lead conversion.

X Education has appointed you to help them select the most promising leads, i.e. the leads that are most likely to convert into paying customers. The company requires you to build a model wherein you need to assign a lead score to each of the leads such that the customers with a higher lead score have a higher conversion chance and the customers with a lower lead score have a lower conversion chance. The CEO, in particular, has given a ballpark of the target lead conversion rate to be around 80%.



# GOALS OF THE ANALYSIS

There are quite a few goals for this case study:

1. Build a logistic regression model to assign a lead score between 0 and 100 to each of the leads which can be used by the company to target potential leads. A higher score would mean that the lead is hot, i.e. is most likely to convert whereas a lower score would mean that the lead is cold and will mostly not get converted.
2. There are some more problems presented by the company which your model should be able to adjust to if the company's requirement changes in the future so you will need to handle these as well. These problems are provided in a separate doc file. Please fill it based on the logistic regression model you got in the first step. Also, make sure you include this in your final PPT where you'll make recommendations.

# DATA EXPLORATION AND PRE-PROCESSING


## IMPORTING THE REQUIRED LIBRARIES

```
# Importing the necessary libraries:

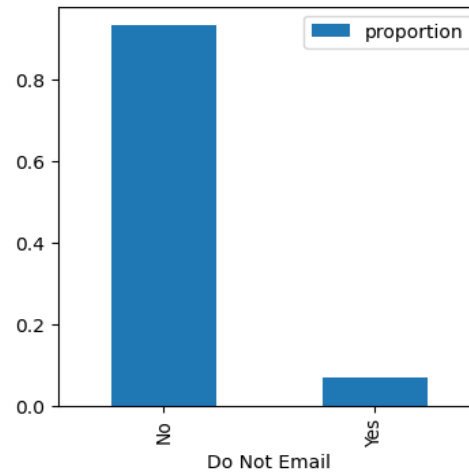
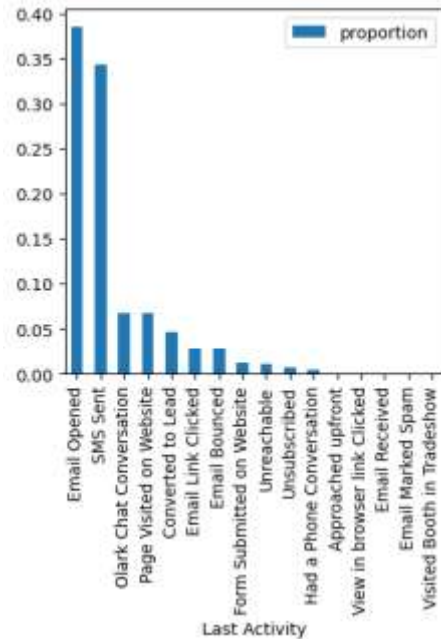
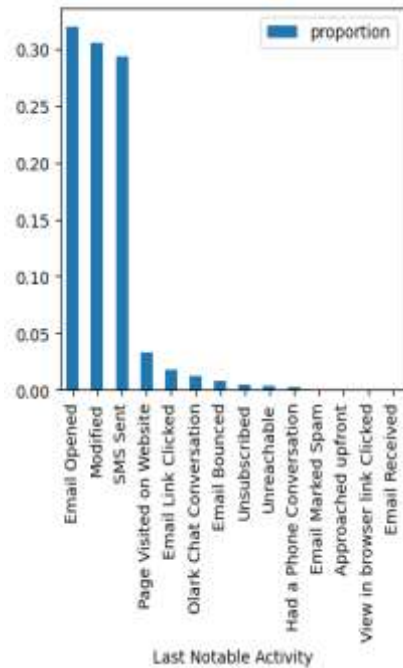
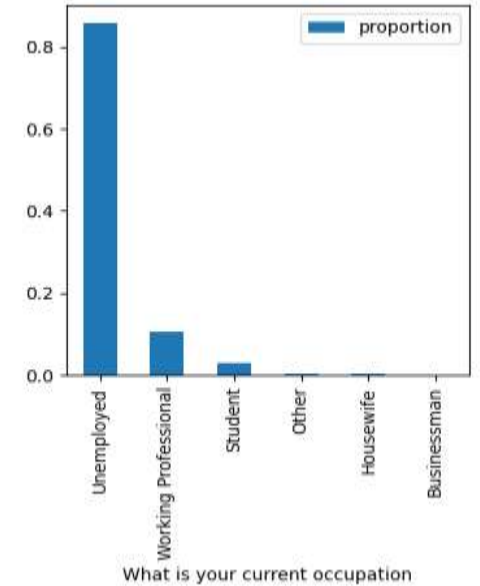
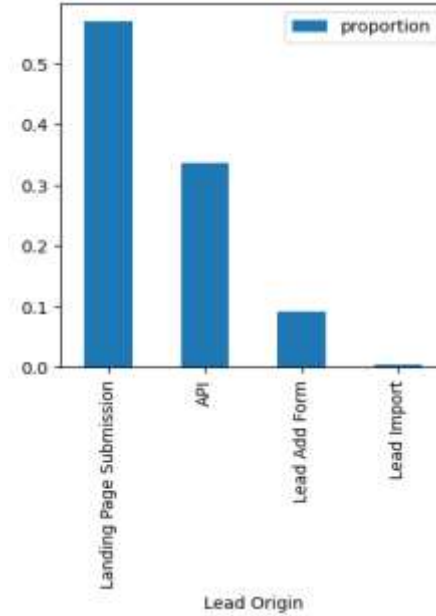
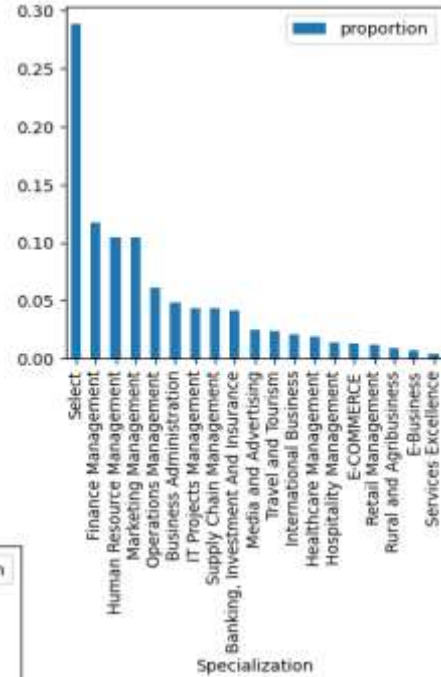
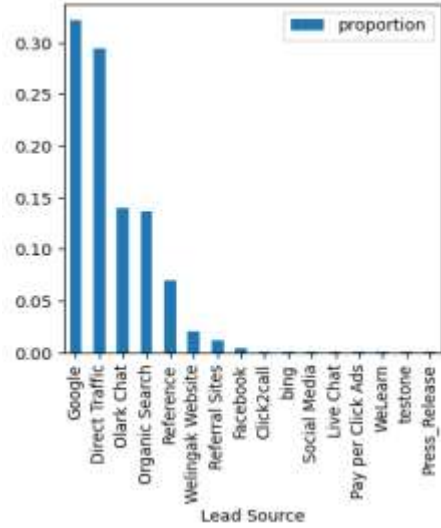
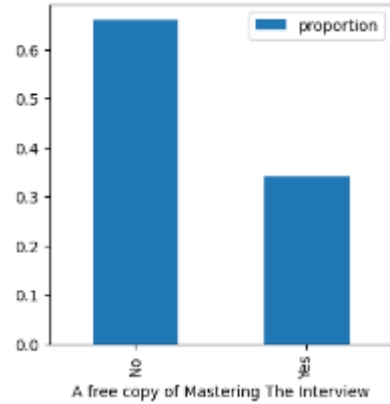
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFE
from sklearn import metrics
from sklearn.metrics import precision_recall_curve
from statsmodels.stats.outliers_influence import variance_inflation_factor
import statsmodels.api as sm

import warnings
warnings.filterwarnings('ignore')

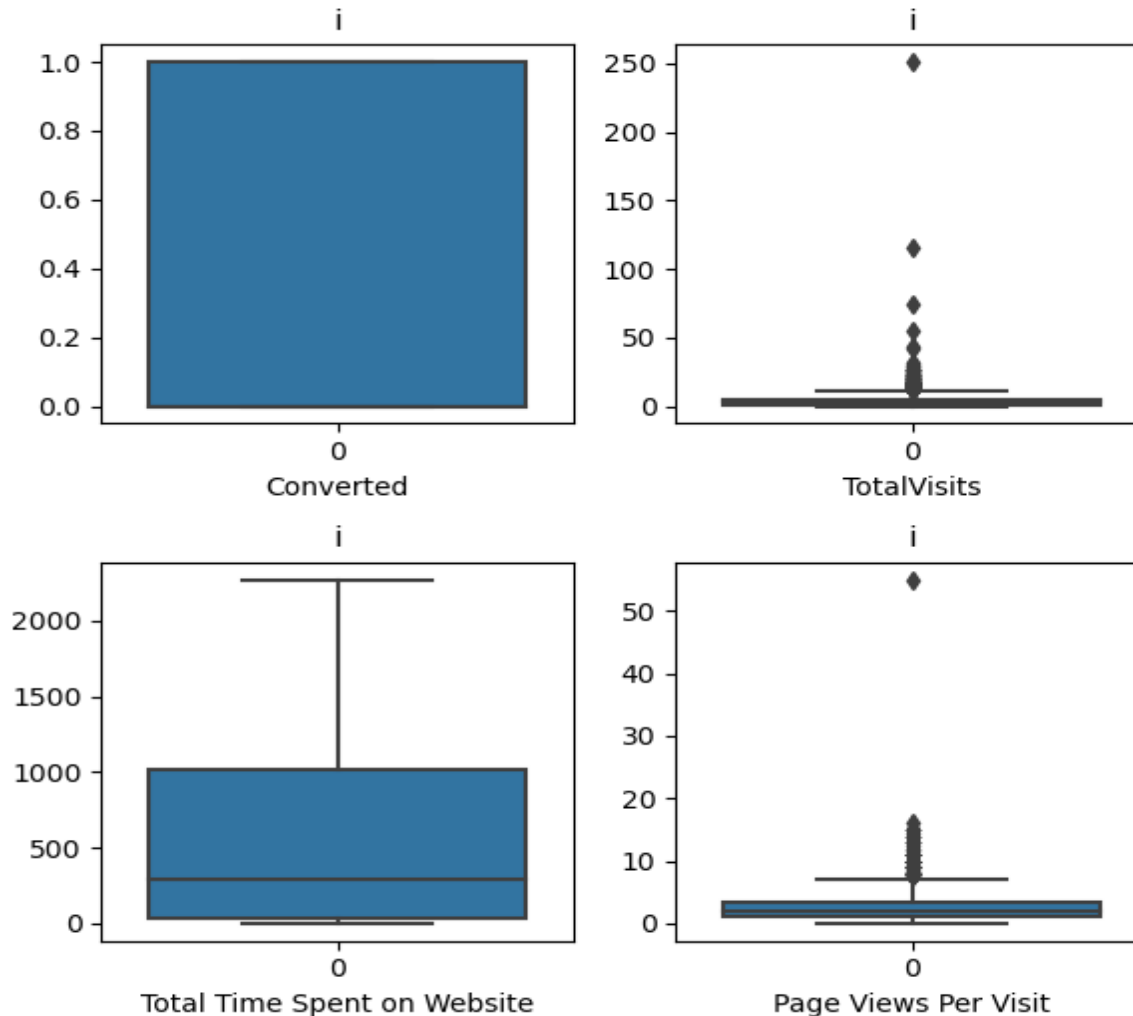
pd.options.display.max_columns = None
pd.options.display.max_rows = None
```

- 
- . READING THE DATASET.**
  - . CHECKING THE INFO, HEAD AND SHAPE.**
  - . CHECKING FOR NULL VALUES.**
  - . DROPPING COLUMNS WITH MORE THAN 3000 NULL VALUES.**
  - . DROPPING COLUMNS WHICH ARE NOT USEFUL TO THE MODEL.**
  - . HANDLING THE COLUMNS WITH THE VALUE “SELECT”**
  - . FINAL CHECK FOR NULL VALUES AND IF THE TOTAL DATASET IS CLEAN OR NOT.**
  - . CREATING DUMMY VARIABLES FOR CATEGORICAL DATA.**
  - . SPLITTING DATA INTO TRAIN AND TEST SETS.**
  - . SCALING THE NUMERIC VARIABLES**

# UNIVARIATE ANALYSIS

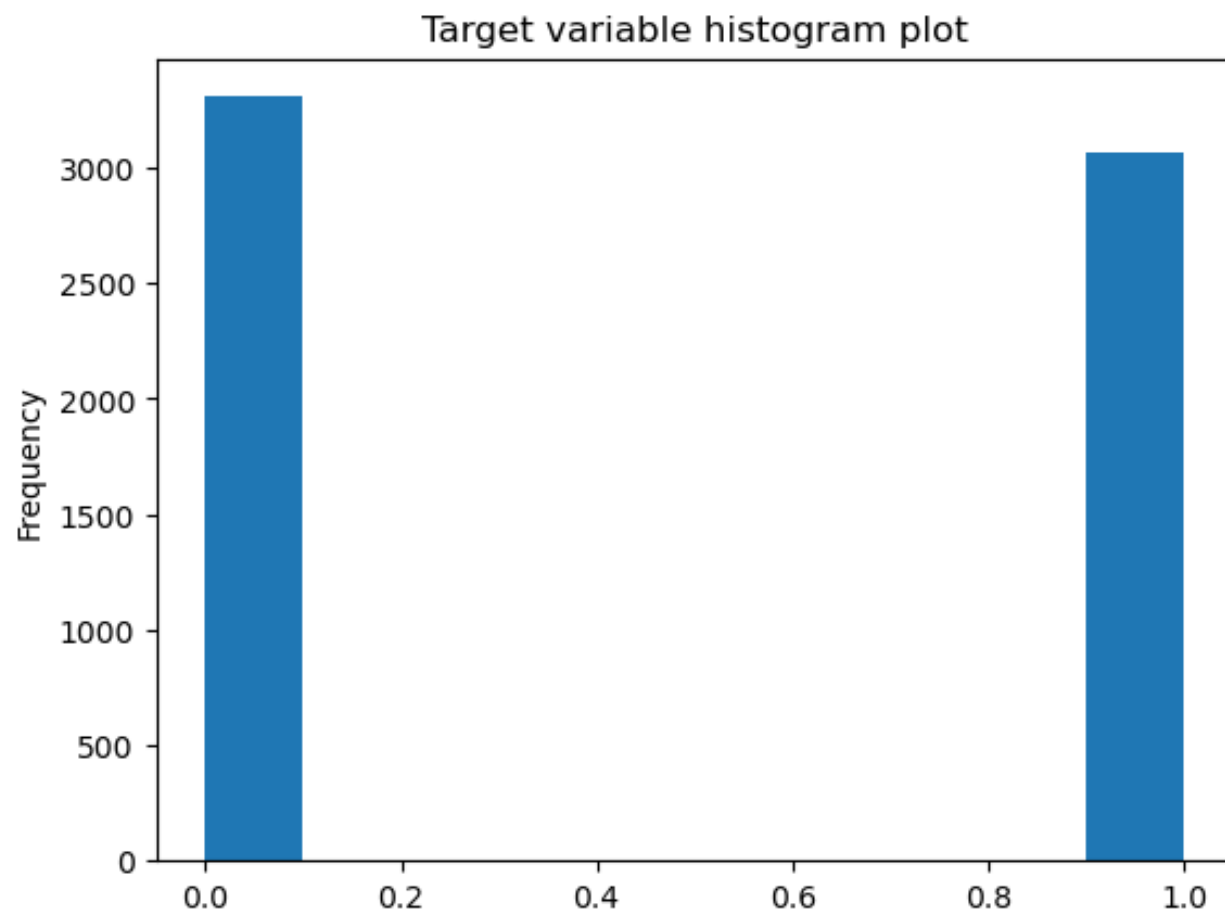


## BOXPLOTS : OUTLIERS ARE PRESENT IN 'TOTALVISITS' AND 'TOTAL TIME SPENT ON WEBSITE'

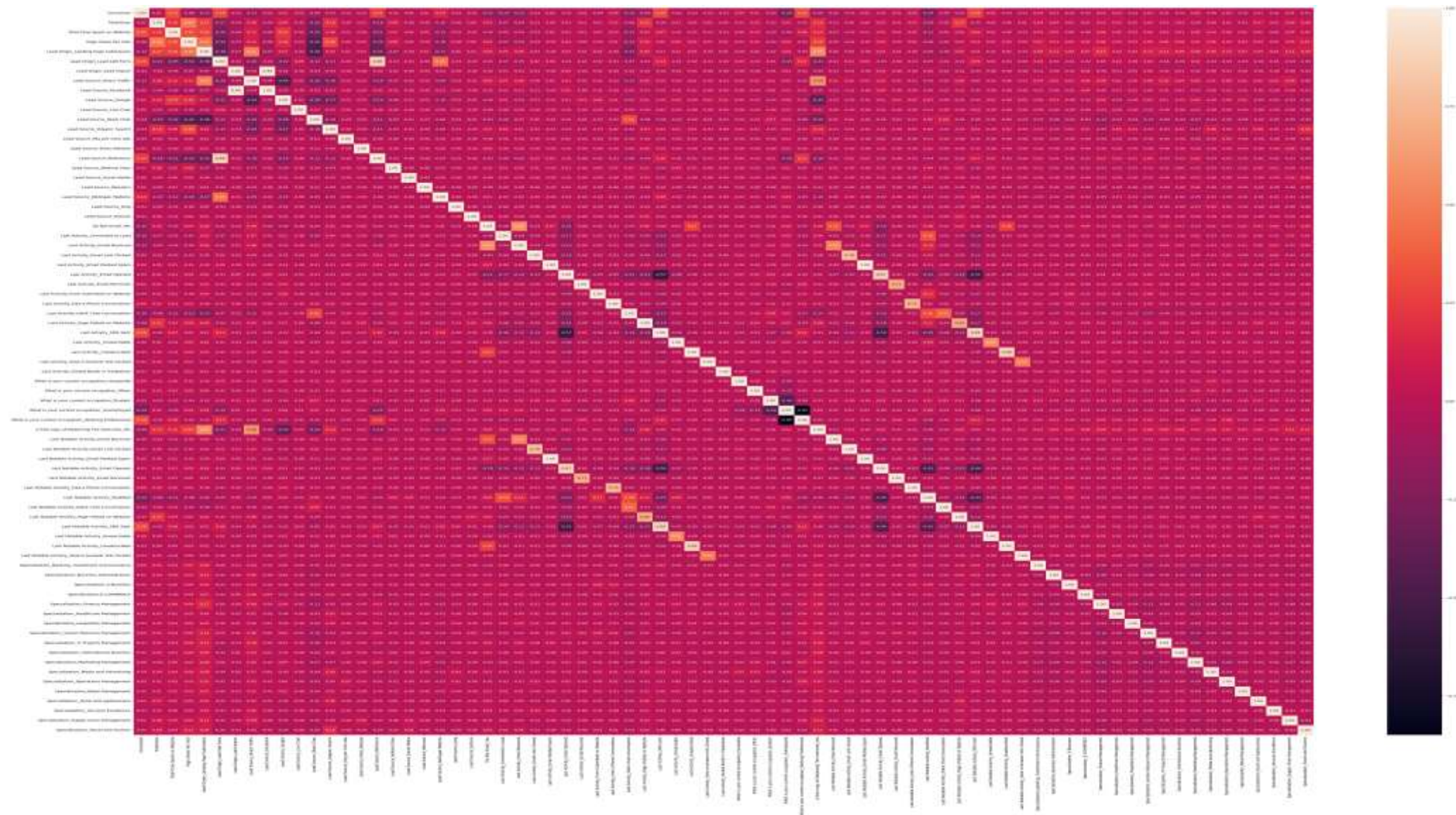




# ANALYSIS OF TARGET COLUMN TO CHECK FOR DATA IMBALANCE



## HEATMAP FOR VISUALISING CORRELATIONS



## MODEL BUILDING:

- . USING STATSMODELS AND REF WE BUILT A MODEL WITH THE BEST 15 FEATURES
- . CHECKED P-VALUES AND VIFS AND ITERATED ON THE MODEL.
- . CAME TO A FINAL MODEL WITH ALL P-VALUES AND VIFS IN RANGE.

Generalized Linear Model Regression Results

Dep. Variable:	Converted	No. Observations:	4461
Model:	GLM	Df Residuals:	4449
Model Family:	Binomial	Df Model:	11
Link Function:	Logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-2043.7
Date:	Tue, 19 Nov 2024	Deviance:	4087.3
Time:	19:13:26	Pearson chi2:	4.65e+03
No. iterations:	7	Pseudo R-squ. (CS):	0.3742
Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
const	1.7491	0.188	9.296	0.000	1.380	2.118
Total Time Spent on Website	1.0931	0.046	23.647	0.000	1.003	1.184
Lead Origin_Lead Add Form	4.0410	0.256	15.799	0.000	3.540	4.542
Lead Source_Olark Chat	1.2964	0.114	11.330	0.000	1.072	1.521
Lead Source_Welingak Website	2.0508	1.037	1.977	0.048	0.018	4.084
Do Not Email_Yes	-1.4175	0.193	-7.328	0.000	-1.797	-1.038
Last Activity_Had a Phone Conversation	2.8940	0.799	3.624	0.000	1.329	4.459
Last Activity_SMS Sent	0.9945	0.084	11.810	0.000	0.829	1.159
What is your current occupation_Student	-2.4199	0.284	-8.513	0.000	-2.977	-1.863
What is your current occupation_Unemployed	-2.5468	0.188	-13.542	0.000	-2.915	-2.178
Last Notable Activity_Modified	-0.8510	0.090	-9.504	0.000	-1.026	-0.675
Last Notable Activity_Unreachable	2.4662	0.807	3.056	0.002	0.885	4.048

	Features	VIF
8	What is your current occupation_Unemployed	2.19
6	Last Activity_SMS Sent	1.52
1	Lead Origin_Lead Add Form	1.51
9	Last Notable Activity_Modified	1.50
2	Lead Source_Olark Chat	1.34
3	Lead Source_Welingak Website	1.30
0	Total Time Spent on Website	1.24
4	Do Not Email_Yes	1.09
7	What is your current occupation_Student	1.04
5	Last Activity_Had a Phone Conversation	1.01
10	Last Notable Activity_Unreachable	1.01

# MODEL PERFORMANCE ON TRAIN SET USING RANDOM CUTOFF OF 0.5

```
# Check the accuracy
```

```
print(metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.Predicted))
```

```
0.7910782335799148
```

```
: # Let's evaluate the other metrics as well
```

```
TP = confusion[1,1] # true positive  
TN = confusion[0,0] # true negatives  
FP = confusion[0,1] # false positives  
FN = confusion[1,0] # false negatives
```

```
: # Calculating the sensitivity
```

```
TP/(TP+FN)
```

```
: 0.7398790134946487
```

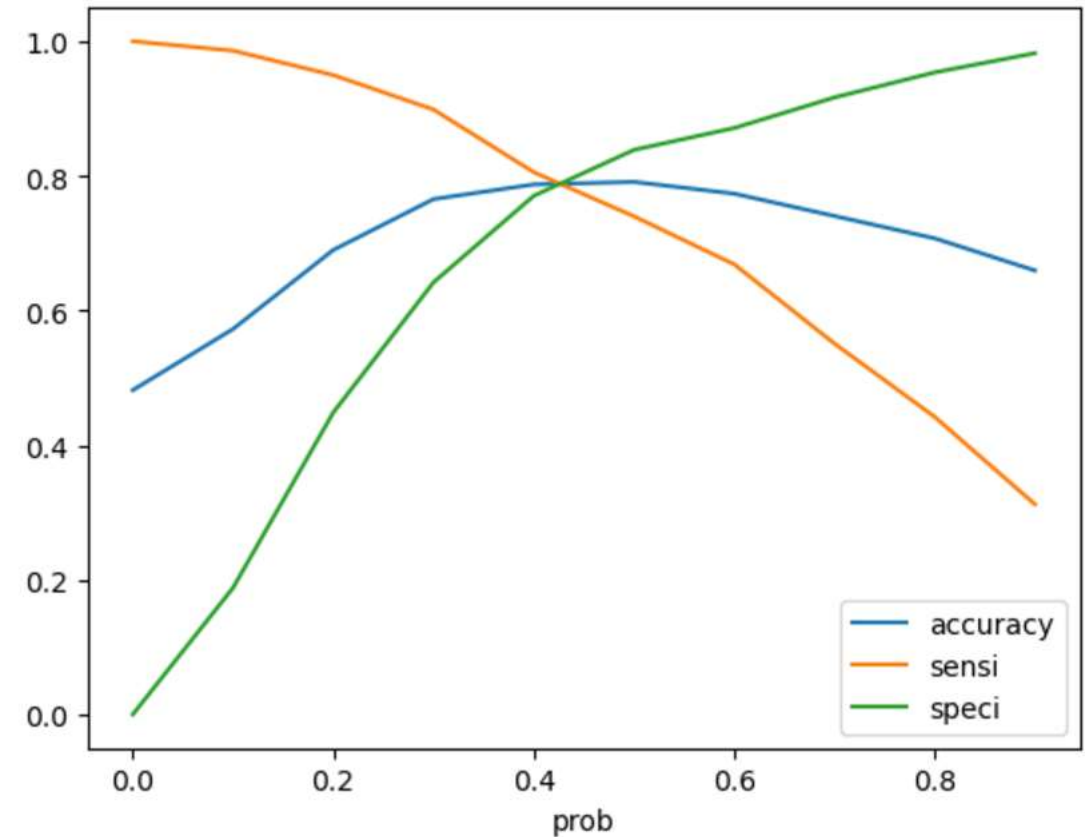
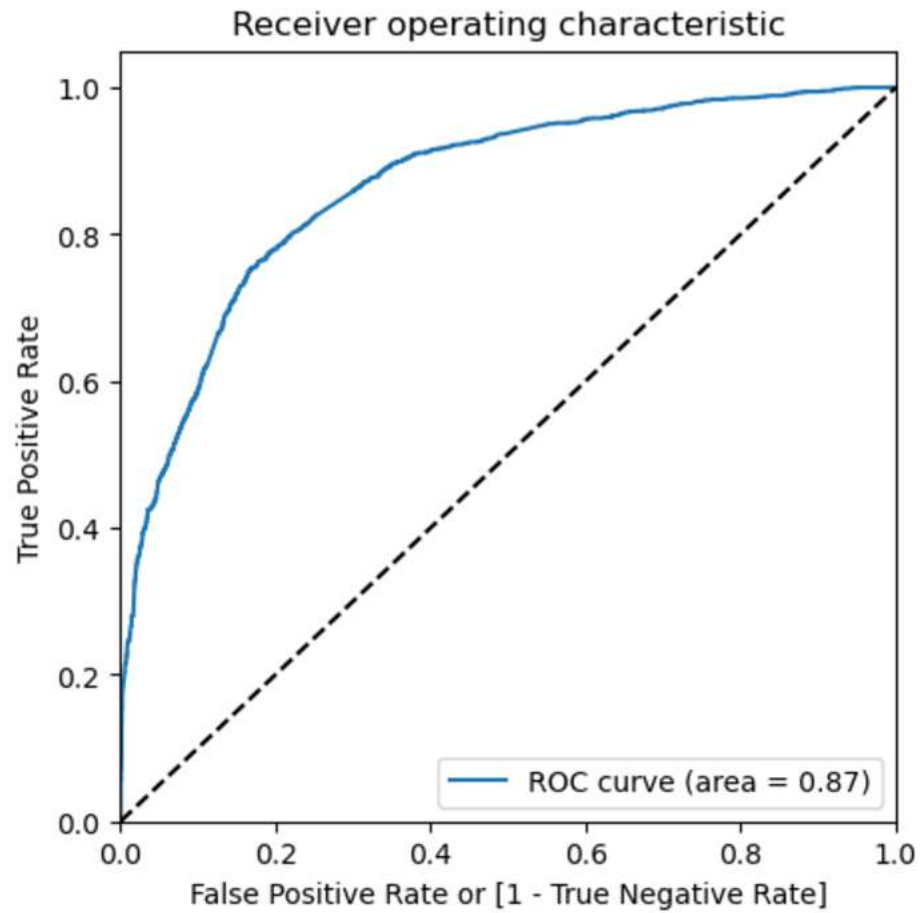
```
: # Calculate the specificity
```

```
TN/(TN+FP)
```

```
: 0.8386678200692042
```



# FINDING THE OPTIMAL CUTOFF



## We are getting 0.42 as the optimum cutoff.

# USING NEW CUTOFF FOR THE METRICS ON TRAIN DATA

```
# Let's check the accuracy now
```

```
metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.final_predicted)
```

```
0.7899574086527684
```

```
# Creating confusion matrix
```

```
confusion2 = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.Predicted )  
print(confusion2)
```

```
[[1939  373]  
 [ 559 1590]]
```

```
# Let's evaluate the other metrics as well
```

```
TP = confusion2[1,1] # true positive  
TN = confusion2[0,0] # true negatives  
FP = confusion2[0,1] # false positives  
FN = confusion2[1,0] # false negatives
```

```
# Calculate Sensitivity
```

```
TP/(TP+FN)
```

```
0.7398790134946487
```

```
# Calculate Specificity
```

```
TN/(TN+FP)
```

```
0.8386678200692042
```

# MAKING PREDICTIONS ON TEST SET

```
# Let's check the overall accuracy
```

```
metrics.accuracy_score(y_pred_final['Converted'], y_pred_final.final_predicted)
```

```
0.7892259414225942
```

```
confusion2 = metrics.confusion_matrix(y_pred_final['Converted'], y_pred_final.final_predicted )  
confusion2
```

```
array([[787, 209],  
       [194, 722]], dtype=int64)
```

```
TP = confusion2[1,1] # true positive  
TN = confusion2[0,0] # true negatives  
FP = confusion2[0,1] # false positives  
FN = confusion2[1,0] # false negatives
```

```
# Calculate sensitivity
```

```
TP / float(TP+FN)
```

```
0.7882096069868996
```

```
# Calculate specificity
```

```
TN / float(TN+FP)
```

```
0.7901606425702812
```

# PRECISION AND RECALL VIEW (TRAIN)

*#Looking at the confusion matrix again*

```
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.Predicted )  
confusion
```

```
array([[1939,  373],  
       [ 559, 1590]], dtype=int64)
```

## Precision

$TP / TP + FP$

```
confusion[1,1]/(confusion[0,1]+confusion[1,1])
```

0.8099847172694855

## Recall

$TP / TP + FN$

```
confusion[1,1]/(confusion[1,0]+confusion[1,1])
```

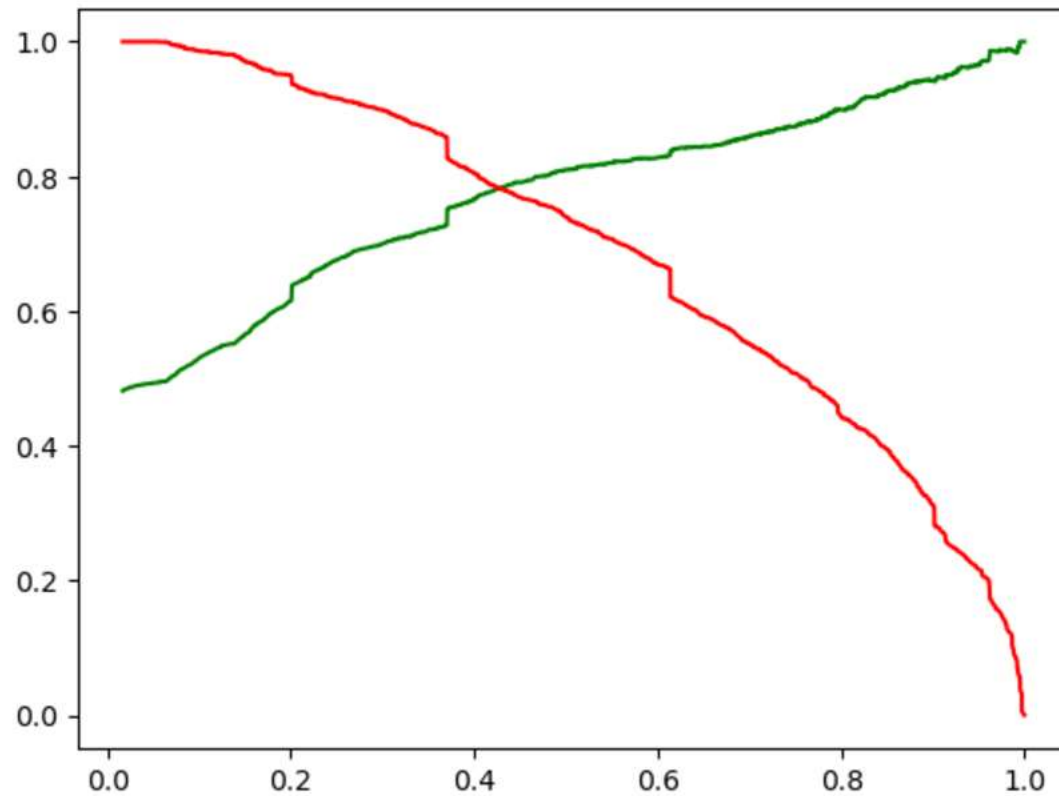
0.7398790134946487



# PRECISION RECALL CURVE FOR FINDING THE OPTIMUM CUTOFF(0.44)

```
p, r, thresholds = precision_recall_curve(y_train_pred_final.Converted, y_train_pred_final.Conversion_Prob)
```

```
plt.plot(thresholds, p[:-1], "g-")  
plt.plot(thresholds, r[:-1], "r-")  
plt.show()
```



# CHECKING METRICS USING NEW CUTOFF

```
# Let's check the accuracy now
```

```
metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.final_predicted)
```

```
0.7917507285362027
```

```
# Let's create the confusion matrix once again
```

```
confusion2 = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.final_predicted )  
confusion2
```

```
array([[1865,  447],  
       [ 482, 1667]], dtype=int64)
```

```
# Let's evaluate the other metrics as well
```

```
TP = confusion2[1,1] # true positive  
TN = confusion2[0,0] # true negatives  
FP = confusion2[0,1] # false positives  
FN = confusion2[1,0] # false negatives
```

```
# Calculating Precision
```

```
TP/(TP+FP)
```

```
0.7885525070955535
```

```
# Calculating Recall
```

```
TP/(TP+FN)
```

```
0.7757096323871568
```

# MAKING PREDICTIONS ON TEST SET AGAIN

```
: # Let's check the overall accuracy
```

```
metrics.accuracy_score(y_pred_final['Converted'], y_pred_final.final_predicted)
```

```
: 0.7866108786610879
```

```
: confusion2 = metrics.confusion_matrix(y_pred_final['Converted'], y_pred_final.final_predicted )  
confusion2
```

```
: array([[799, 197],  
        [211, 705]], dtype=int64)
```

```
: TP = confusion2[1,1] # true positive  
TN = confusion2[0,0] # true negatives  
FP = confusion2[0,1] # false positives  
FN = confusion2[1,0] # false negatives
```

```
: # Calculate Precision
```

```
TP/(TP+FP)
```

```
: 0.7815964523281597
```

```
: # Calculate Recall
```

```
TP/(TP+FN)
```

```
: 0.769650655021834
```



## MAJOR DRIVERS OF LEAD CONVERSION

- Total Time Spent on Website
- Last Activity had a phone conversation
- Lead Origin - Lead Add Form

# STRATEGIES FOR AGGRESSIVE PERIODS

## Recommended Strategy

### 1. Adjust Model Cutoff to Increase Sensitivity

- Lower the cutoff probability (e.g., from **0.5 to 0.3**) to classify more leads as potential converters.
- At a **0.3 cutoff**, your metrics show:
  - Sensitivity = **0.8426** (84.26% of potential leads are identified).
  - Specificity = **0.7524**, which is reasonable given the focus on aggressive outreach.
- This will ensure more potential leads are contacted without ignoring too many high-value leads.

### 2. Prioritize Leads by Conversion Probability

- Rank leads predicted as 1 by their **probabilities** (model output).
- Start with the top-scoring leads (highest conversion probability) and progressively move down the list.
- This helps allocate time and resources effectively to maximize returns.

### 3. Leverage Interns for Outreach

- **Divide Leads:** Distribute leads among interns based on their conversion probabilities.
- **Assign Scripts:** Use personalized scripts based on key features (e.g., occupation, last activity) to enhance engagement.
  - Example: Leads with "Working Professional" as occupation or "Had a Phone Conversation" as last activity should receive tailored pitches.

### 4. Optimize Call Strategy

- Use data insights to time calls:
  - Leads with "**SMS Sent**" or "**Phone Conversation**" activities may respond better during similar time slots.
- Experiment with calling windows (e.g., lunch hours for working professionals).

### 5. Monitor and Reallocate Daily

- Track conversion rates in real-time to measure intern performance and lead quality.
- Redistribute unresponsive leads to other interns to maximize coverage.

### 6. Additional Engagement Channels

- Supplement phone calls with:
  - **SMS/Emails:** Send follow-ups post-call for those who express interest.
  - **Chatbots:** Engage low-priority leads via automated chats to convert them passively.

# STRATEGIES FOR LOW- PRIORITY PERIODS

## Recommended Strategy

### 1. Adjust Model Cutoff to Increase Specificity

- Raise the cutoff probability (e.g., from **0.5 to 0.6 or 0.7**) to focus on leads with the highest likelihood of conversion.
- At a **0.7 cutoff**, your metrics show:
  - Specificity = **0.9483** (94.83% of leads classified as 0 are correctly identified).
  - Sensitivity = **0.5094**, which is acceptable since fewer leads need to be called.

### 2. Focus on High-Probability Leads

- Use the model's probabilities to rank and segment leads.
- Call only those leads with probabilities above the cutoff.

### 3. Leverage Other Engagement Channels

- For leads below the cutoff, avoid phone calls and use cost-effective engagement strategies:
  - **Email Campaigns**: Send personalized follow-ups or offers.
  - **Automated Chatbots**: Use chat tools to passively gauge interest.
  - **SMS/WhatsApp**: Send quick reminders or updates to low-probability leads.

### 4. Analyze Past Conversion Patterns

- Identify patterns in previous data to refine high-conversion segments:
  - For example, prioritize leads with specific "Last Activities" like "**SMS Sent**" or "**Phone Conversation**", which indicate higher intent.

### 5. Minimal Contact Workflow

- Introduce a tiered workflow:
  - **High-probability leads**: Phone calls + follow-ups.

- **Moderate-probability leads**: Automated emails/SMS only.
- **Low-probability leads**: No contact until the next cycle.

### 6. Implement a Feedback Loop

- Continuously monitor outcomes to adjust the cutoff dynamically if unnecessary calls persist.



**THE END**