Leads Case Study Presentation

BY -

DHAWAL ARORA

HEMANSHU GHADIGAONKAR

SHRINIDHI RAMESH

Your best quote that reflects your approach… "It's one small step for man, one giant leap for mankind."

- NEIL ARMSTRONG

Introduction

In this case study, I examined the poor lead conversion rate of X Education, a company that markets online courses through websites and search engines. My goal was to create a logistic regression machine learning model using data provided by the company to identify key factors that could potentially increase the lead conversion rate. This would allow the company to focus on communicating with leads who are more likely to choose a course.

Step 1: Library Upload

I began by uploading the necessary libraries such as pandas, seaborn, matplotlib, sklearn, and statsmodels, as shown in the attached diagram.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFE
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn import metrics
from sklearn.metrics import precision_score, recall_score
```

✓ 6.9s

Step 2: CSV Data Import

Next, I used the pandas library to import the CSV data into the Jupyter notebook using the read_csv function. I then used leads.head() to view the first few lines of the imported data.

```
[5] leads_csv = pd.read_csv("D:/Upgrad Class/Lead Case Study/leads.csv")
✓ 0.0s
```

```
[6] leads_csv.head()
✓ 0.0s
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	660727	Landing Page Submission	Direct Traffic	No	No	1	2.0
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	No	0	1.0
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	No	1	2.0

5 rows × 37 columns

Step 3: Data Operations

I performed various operations to gain a deeper understanding of the data, such as using leads.shape() to determine the number of rows and columns, and leads.dtypes() to identify the data types of different columns. I also used leads.info() to view additional information about the columns.

```
leads_csv.describe()
```

[9] ✓ 0.0s

	Lead Number	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	As A
count	9240.000000	9240.000000	9103.000000	9240.000000	9103.000000	
mean	617188.435606	0.385390	3.445238	487.698268	2.362820	
std	23405.995698	0.486714	4.854853	548.021466	2.161418	
min	579533.000000	0.000000	0.000000	0.000000	0.000000	
25%	596484.500000	0.000000	1.000000	12.000000	1.000000	
50%	615479.000000	0.000000	3.000000	248.000000	2.000000	
75%	637387.250000	1.000000	5.000000	936.000000	3.000000	
max	660737.000000	1.000000	251.000000	2272.000000	55.000000	

+ Code + Markdown

```
leads_csv.info()
```

[8] ✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 37 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Prospect ID                                   9240 non-null   object
 1   Lead Number                                   9240 non-null   int64
 2   Lead Origin                                   9240 non-null   object
 3   Lead Source                                   9204 non-null   object
 4   Do Not Email                                  9240 non-null   object
 5   Do Not Call                                   9240 non-null   object
 6   Converted                                     9240 non-null   int64
 7   TotalVisits                                   9103 non-null   float64
 8   Total Time Spent on Website                   9240 non-null   int64
 9   Page Views Per Visit                          9103 non-null   float64
 10  Last Activity                                 9137 non-null   object
 11  Country                                       6779 non-null   object
 12  Specialization                                7802 non-null   object
 13  How did you hear about X Education            7033 non-null   object
 14  What is your current occupation               6550 non-null   object
 15  What matters most to you in choosing a course 6531 non-null   object
 16  Search                                        9240 non-null   object
 17  Magazine                                      9240 non-null   object
 18  Newspaper Article                             9240 non-null   object
```

Step 4: Data Cleaning

I conducted data cleaning on the lead dataset. First, I identified columns with null values and calculated the percentage of null values using is_null().sum(). Next, I dropped columns with null values greater than 45% using the drop function. I then replaced null values in the remaining columns with the most frequent value. I also removed columns with unbalanced data, such as Do Not Call, Search, Magazine, etc., resulting in a balanced dataset with no null values.

checking the null values in terms of percentage

```
round(100*(leads_csv.isnull().sum()/len(leads_csv.index)),2)
```

[14] 0.0s

```
Prospect ID                                      0.00
Lead Number                                      0.00
Lead Origin                                      0.00
Lead Source                                      0.39
Do Not Email                                     0.00
Do Not Call                                      0.00
Converted                                        0.00
TotalVisits                                      1.48
Total Time Spent on Website                      0.00
Page Views Per Visit                             1.48
Last Activity                                    1.11
Country                                         26.63
Specialization                                  36.58
How did you hear about X Education              78.46
What is your current occupation                 29.11
What matters most to you in choosing a course   29.32
Search                                           0.00
Magazine                                         0.00
Newspaper Article                                0.00
X Education Forums                               0.00
Newspaper                                        0.00
Digital Advertisement                            0.00
Through Recommendations                          0.00
Receive More Updates About Our Courses           0.00
Tags                                            36.29
Lead Quality                                    51.59
Update me on Supply Chain Content                0.00
Get updates on DM Content                        0.00
Lead Profile                                    74.19
City                                            39.71
Asymmetrique Activity Index                     45.65
Asymmetrique Profile Index                      45.65
Asymmetrique Activity Score                     45.65
Asymmetrique Profile Score                      45.65
I agree to pay the amount through cheque         0.00
A free copy of Mastering The Interview           0.00
Last Notable Activity                            0.00
dtype: float64
```

Step 5: Handling Outliers

I addressed outliers in numeric data columns such as TotalVisits, Total Time Spent on Website, and Page Views Per Visit to ensure data accuracy.

We are examining the numerical variables for any outliers.

```
leads_csv.info()
```

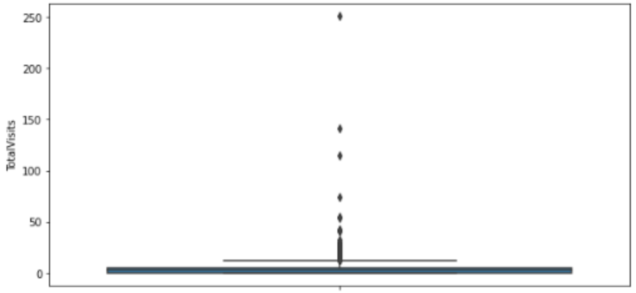
[174] ✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 14 columns):
 #   Column                                  Non-Null Count  Dtype  
---  ------                                  --------------  -----  
 0   Lead Origin                             9240 non-null   object 
 1   Lead Source                             9240 non-null   object 
 2   Do Not Email                            9240 non-null   object 
 3   Converted                               9240 non-null   int64  
 4   TotalVisits                             9240 non-null   float64
 5   Total Time Spent on Website             9240 non-null   int64  
 6   Page Views Per Visit                    9240 non-null   float64
 7   Last Activity                           9240 non-null   object 
 8   Specialization                          9240 non-null   object 
 9   What is your current occupation         9240 non-null   object 
 10  Tags                                    9240 non-null   object 
 11  City                                    9240 non-null   object 
 12  A free copy of Mastering The Interview  9240 non-null   object 
 13  Last Notable Activity                   9240 non-null   object 
dtypes: float64(2), int64(2), object(10)
memory usage: 1010.8+ KB
```

Total Visits

```
plt.figure(figsize=(10,5))
sns.boxplot(y=leads_csv['TotalVisits'])
plt.show()
```

[175] ✓ 0.0s



Step 6: Creating Dummy Variables

I created dummy variables for categorical columns with more than two categories. This resulted in 90 columns, preparing the data for the machine learning algorithm.

Dummy Variable Creation

```
leads_csv.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 14 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Lead Origin                             9240 non-null   object
 1   Lead Source                             9240 non-null   object
 2   Do Not Email                            9240 non-null   object
 3   Converted                               9240 non-null   int64
 4   TotalVisits                             9240 non-null   float64
 5   Total Time Spent on Website             9240 non-null   int64
 6   Page Views Per Visit                    9240 non-null   float64
 7   Last Activity                           9240 non-null   object
 8   Specialization                          9240 non-null   object
 9   What is your current occupation         9240 non-null   object
 10  Tags                                    9240 non-null   object
 11  City                                    9240 non-null   object
 12  A free copy of Mastering The Interview  9240 non-null   object
 13  Last Notable Activity                   9240 non-null   object
dtypes: float64(2), int64(2), object(10)
memory usage: 1010.8+ KB
```

```
# categorical columns
categorical_cols= leads_csv.select_dtypes(include=['object']).columns
categorical_cols
```

✓ 0.0s

```
Index(['Lead Origin', 'Lead Source', 'Do Not Email', 'Last Activity',
       'Specialization', 'What is your current occupation', 'Tags', 'City',
       'A free copy of Mastering The Interview', 'Last Notable Activity'],
      dtype='object')
```

```
leads_csv["Do Not Email"].value_counts()
```

✓ 0.0s

```
No     8506
Yes     734
Name: Do Not Email, dtype: int64
```

```
leads_csv["A free copy of Mastering The Interview"].value_counts()
```

Step 7: Train-Test Data Split

I split the data into train and test sets using the train-test library, and scaled the numeric variables using the StandardScaler method.

Train-Test Split

```python
y = leads_csv['Converted']
X=leads_csv.drop('Converted', axis=1)

y.head()
```

0.1s Python

```
0    0
1    0
2    1
3    0
4    1
Name: Converted, dtype: int64
```

```python
X.head()
```

0.0s Python

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Origin_Quick Add Form	What is your current occupation_Student	What is your current occupation_Unemployed	What is your c occupation_W Profe
0	0.0	0	0.0	0	0	0	0	0	1	
1	5.0	674	2.5	0	0	0	0	0	1	
2	2.0	1532	2.0	1	0	0	0	1	0	
3	1.0	305	1.0	1	0	0	0	0	1	
4	2.0	1428	1.0	1	0	0	0	0	1	

5 rows × 59 columns

```python
# Splitting the data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=100)
```

0.0s Python

```python
X_train.info()
```

0.0s Python

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6468 entries, 1871 to 5640
Data columns (total 59 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
```

Step 8: Model Creation

I created models using Recursive Feature Elimination (RFE) until all the p-values of the columns were less than 0.05. I also checked for VIF values less than 3. I eliminated columns with high p-values and VIF one by one, and identified the most suitable model for further analysis. I then made predictions on the target variable and evaluated the model's accuracy, precision, recall, and specificity.

BUILDING MODEL 1

```
X_train_sm = sm.add_constant(X_train[col])
l_model1 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = l_model1.fit()
res.summary()
```

✓ 0.0s

Generalized Linear Model Regression Results

Dep. Variable:	Converted	No. Observations:	6468
Model:	GLM	Df Residuals:	6452
Model Family:	Binomial	Df Model:	15
Link Function:	Logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-1396.4
Date:	Sun, 16 Apr 2023	Deviance:	2792.7
Time:	01:49:15	Pearson chi2:	1.06e+04
No. Iterations:	8	Pseudo R-squ. (CS):	0.5924
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.5631	0.088	-29.114	0.000	-2.736	-2.391
What is your current occupation_Unemployed	2.2634	0.117	19.370	0.000	2.034	2.492
What is your current occupation_Working Professional	2.5416	0.356	7.149	0.000	1.845	3.238
Lead Source_Welingak Website	2.9473	0.733	4.021	0.000	1.511	4.384
Last Activity_SMS Sent	2.0669	0.109	18.950	0.000	1.853	2.281
Tags_Already a student	-4.8662	0.718	-6.774	0.000	-6.274	-3.458
Tags_Closed by Horizzon	5.8904	1.010	5.829	0.000	3.910	7.871
Tags_Graduation in progress	-2.5781	0.493	-5.228	0.000	-3.545	-1.612
Tags_Interested in full time MBA	-3.8136	0.736	-5.184	0.000	-5.255	-2.372
Tags_Interested in other courses	-3.4071	0.325	-10.479	0.000	-4.044	-2.770
Tags_Lost to EINS	5.5233	0.727	7.601	0.000	4.099	6.948
Tags_Not doing further education	-4.6677	1.015	-4.599	0.000	-6.657	-2.678
Tags_Other_tags	-3.0611	0.271	-11.300	0.000	-3.592	-2.530
Tags_Ringing	-4.4302	0.233	-19.033	0.000	-4.886	-3.974
Tags_Will revert after reading the email	3.3450	0.183	18.256	0.000	2.986	3.704
Tags_switched off	-4.8710	0.522	-9.334	0.000	-5.894	-3.848

```
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

✓ 0.1s

	Features	VIF
5	Tags_Closed by Horizzon	1.33
11	Tags_Other_tags	1.29
14	Tags_switched off	1.24
10	Tags_Not doing further education	1.14
7	Tags_Interested in full time MBA	1.10
2	Lead Source_Welingak Website	1.09
6	Tags_Graduation in progress	1.09
9	Tags_Lost to EINS	1.06
1	What is your current occupation_Working Profes...	0.95
8	Tags_Interested in other courses	0.44
13	Tags_Will revert after reading the email	0.32
4	Tags_Already a student	0.29
12	Tags_Ringing	0.20
3	Last Activity_SMS Sent	0.12
0	What is your current occupation_Unemployed	0.11

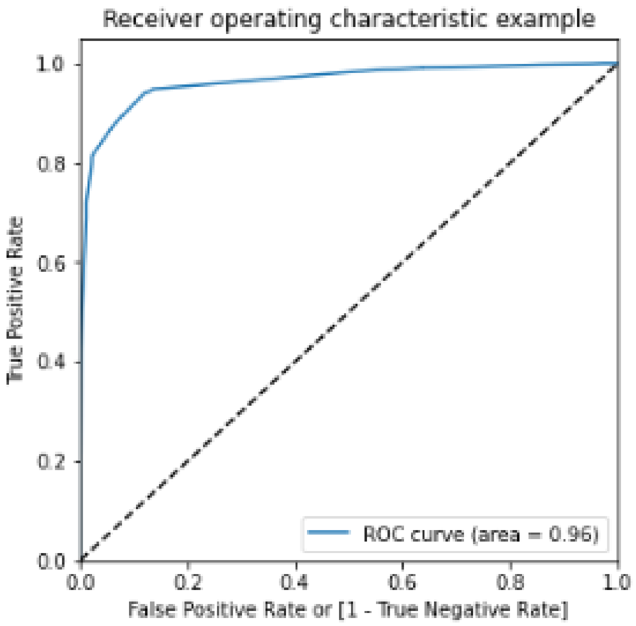
Step 9: ROC Curve

I used the ROC Curve to determine the best cutoff value for observations, instead of the previous 0.5 cutoff. Using this cutoff, I made predictions on the test data and calculated accuracy, precision, and specificity as the final solution for the train data. I also applied the model to the test data, which had not been seen by the model before, to assess its performance.

Calling the ROC function

```
draw_roc(y_train_pred_final.Converted, y_train_
```

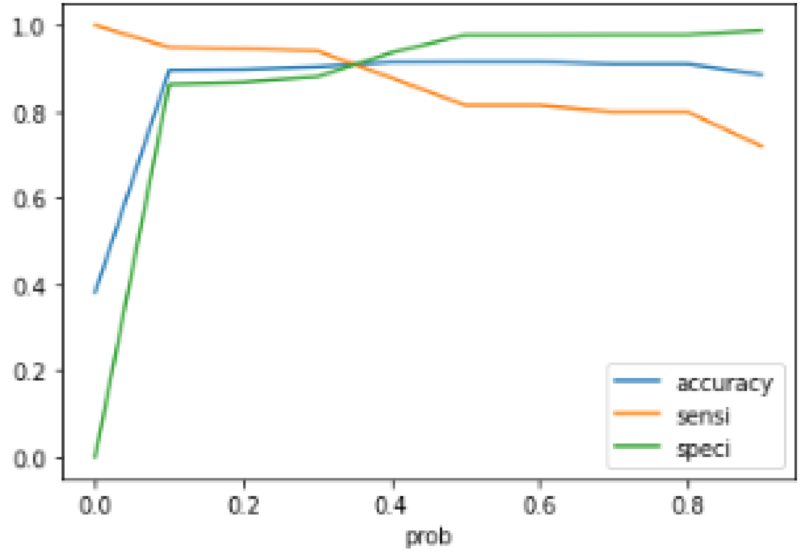
✓ 0.1s



Plotting it

```
cutoff_df.plot.line(x='prob', y=['accuracy','sensi','speci'])
plt.show()
```

✓ 0.1s



Results and Conclusion

Based on the final machine learning model, I achieved an accuracy of 91%, sensitivity of around 81%, and specificity of around 97%. After selecting the best cutoff value of 0.3, I calculated an accuracy of 90%, sensitivity of 94%, and specificity of around 88%. Applying the model to the test data set, I concluded that the model performed well with an accuracy of 90%, sensitivity of 96%, specificity of around 87%, precision score of 83%, and recall score of 96%.

Based on these results, I recommend that the company focus on leads that are closed by 'Horizzon', lost to EINS, and avoid leads that are too