LEAD SCORING CASE STUDY

By
Arjun Singh Baghel

AIM

- To build a Logistic Regression Model to predict whether a lead for online courses for an education company named X Education would be successfully converted or not

OBJECTIVE:

- To Help X Education system to select most promising leads(Hot Leads), leads which are most likely to convert into paying customers
- To build a logistic regression model to assign a lead score value between 0 to 100 to each of the leads which can be used by the company to target potential leads

BUSINESS OBJECTIVE

Business Objective can be classified into 3 main sub-goals

Create a logistic regression model to predict the leads conversion

Multiply the Lead conversion probability to arrive at the lead score

Decide a probability threshold value above which a lead will be predicted as converted

PROBLEM SOLVING METHODOLOGY

- The approach adopted to resolve this group case study can be achieved by following the below methods sequentially

:- Understanding the dataset & Data preparation

:- Applying Recursive feature elimination to identify the best performing subset of features for building the model.

:- Building the model with features selected by RFE.

:- Eliminating all features with high p-value and VIF values and finalize the model

:- Performed model evaluation with various metrics like sensitivity, specificity, precision ,recall etc.

PROBLEM SOLVING METHODOLOGY CONTD..

- Decide on the probability threshold value based on the optimal cutoff point predicted the dependent variable for the training data
- Using the model for the prediction of the dataset and perform model evaluation for the test set

DATA PREPARATION & FEATURE ENGINEERING

- Followed the below steps:

Removed columns which has one unique value	Deleting the following columns as they have only one unique value and hence cannot be responsible in predicting a successful lead case – 'Magazine', 'Receive More Updates About Our Courses' , 'Update me on Supply Chain Content' , 'Update me on Supply Chain Content' and 'I agree to pay the amount through cheque'.
Removing rows where a particular column has high missing values	'Lead Source' is an important column for analysis. Hence all the rows that have null values for it were dropped.
Imputing Null values with Median	The columns 'TotalVisits' and 'Page Views Per Visit' are continuous variables with outliers. Hence the null values for these columns were imputed with the column median values..
Imputing Null Values with mode	The columns 'Country' is a categorical variable with some null values. Also majority of the records belong to the Country 'India'. Thus imputed the null values for this with mode(most occuring value). Then binned rest of category into 'Outside India'..

DATA PREPARATION & FEATURE ENGINEERING

Handling 'Select' values in some columns

- There are some columns in dataset which have a level/value called 'Select'. This might have happened because these fields in the website might be non mandatory fields with drop downs options for the customer to choose from. Amongst the dropdown values, the default option is probably 'Select' and since these aren't mandatory fields, many customer might have have chosen to leave it as the default value 'Select'.
- The Select values in columns were converted to Nulls.

Assigning a Unique Category to NULL/SELECT values

- All the nulls in the columns were binned into a separate column 'Other_Specialization'.
- Instead of deleting columns with huge null value percentage(which results in loss of data), this strategy adds more information into the dataset and results in the change of variance.
- Few Select value has been imputed with the present values like NotSure

Outlier Treatment

- The outliers present in the columns 'TotalVisits' & 'Page Views Per Visit' were finally removed based on interquartile range analysis.

Binary Encoding

- Converting the following binary variables (Yes/No) to 0/1:
- 'Search','Do Not Email', 'Do Not Call', 'Newspaper Article', 'X Education Forums', 'Newspaper', 'Digital Advertisement','Through Recommendations' and 'A free copy of Mastering The Interview'

DATA PREPARATION

- Reading & Visualizing the dataset

```
#displaying the few rows of data set
leads.head()
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	...	Get updates on DM Content	Lead Profile	City	Asymmetrique Activity Index	Asymmetriqu Profile Inde
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	...	No	Select	Select	02.Medium	02.Mediu
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0	674	2.5	...	No	Select	Select	02.Medium	02.Mediu
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	660727	Landing Page Submission	Direct Traffic	No	No	1	2.0	1532	2.0	...	No	Potential Lead	Mumbai	02.Medium	01.Hig
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	No	0	1.0	305	1.0	...	No	Select	Mumbai	02.Medium	01.Hig
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	No	1	2.0	1428	1.0	...	No	Select	Mumbai	02.Medium	01.Hig

5 rows × 37 columns

DATA PREPARATION

Shape of the dataset

```
1  #looking at the shape of data
2  lead_df.shape
```

```
(9240, 37)
```

Total number of rows: 9240

Total number of columns: 37

Information of Dataset(Colum types & Rows)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 37 columns):
Prospect ID                                      9240 non-null object
Lead Number                                      9240 non-null int64
Lead Origin                                      9240 non-null object
Lead Source                                      9204 non-null object
Do Not Email                                     9240 non-null object
Do Not Call                                      9240 non-null object
Converted                                        9240 non-null int64
TotalVisits                                      9103 non-null float64
Total Time Spent on Website                      9240 non-null int64
Page Views Per Visit                             9103 non-null float64
Last Activity                                    9137 non-null object
Country                                          6779 non-null object
Specialization                                   7802 non-null object
How did you hear about X Education               7033 non-null object
What is your current occupation                  6550 non-null object
What matters most to you in choosing a course    6531 non-null object
Search                                           9240 non-null object
Magazine                                         9240 non-null object
Newspaper Article                                9240 non-null object
X Education Forums                               9240 non-null object
Newspaper                                        9240 non-null object
Digital Advertisement                            9240 non-null object
Through Recommendations                          9240 non-null object
Receive More Updates About Our Courses           9240 non-null object
Tags                                             5887 non-null object
Lead Quality                                     4473 non-null object
```

DATA PREPARATION

- Describing the dataset

```
# looking at teh mean and deviation of the in each coulumns and rows
leads.describe()
```

	Lead Number	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Asymmetrique Activity Score	Asymmetrique Profile Score
count	9240.000000	9240.000000	9103.000000	9240.000000	9103.000000	5022.000000	5022.000000
mean	617188.435606	0.385390	3.445238	487.698268	2.362820	14.306252	16.344883
std	23405.995698	0.486714	4.854853	548.021466	2.161418	1.386694	1.811395
min	579533.000000	0.000000	0.000000	0.000000	0.000000	7.000000	11.000000
25%	596484.500000	0.000000	1.000000	12.000000	1.000000	14.000000	15.000000
50%	615479.000000	0.000000	3.000000	248.000000	2.000000	14.000000	16.000000
75%	637387.250000	1.000000	5.000000	936.000000	3.000000	15.000000	18.000000
max	660737.000000	1.000000	251.000000	2272.000000	55.000000	18.000000	20.000000

DATA PREPARATION

- Imputing Select/Null values from the dataset

In [456]:
```
# Mode of city in 'Mumbai' so imputing NaN to 'Mumbai'
lead_df.City = lead_df.City.replace(np.nan, 'Mumbai')
```

Specailization : Column

In [457]:
```
lead_df.Specialization.describe()
```

Out[457]:
```
count                    5860
unique                     18
top       Finance Management
freq                      976
Name: Specialization, dtype: object
```

Top used value is 'Finance Management'

In [458]:
```
#  The lead don't have any specialization ie NaN (36%) value, So we can create new specialization that is 'Others_Specializa
# So imputing NaN to 'Others_Specialization'
lead_df.Specialization = lead_df.Specialization.replace(np.nan, 'Others_Specialization')

```

DATA PREPARATION

- Imputing Select/Null values from the dataset

'What is your current occupation' : Column

In [464]:
```
lead_df['What is your current occupation'].describe()
```

Out[464]:
```
count              6550
unique                6
top          Unemployed
freq               5600
Name: What is your current occupation, dtype: object
```

'Unemployed' is the top used value.

In [465]:
```
# Imputing NaN to 'Unemployed'.
lead_df['What is your current occupation'] = lead_df['What is your current occupation'].replace(np.nan, 'Unemploye
```

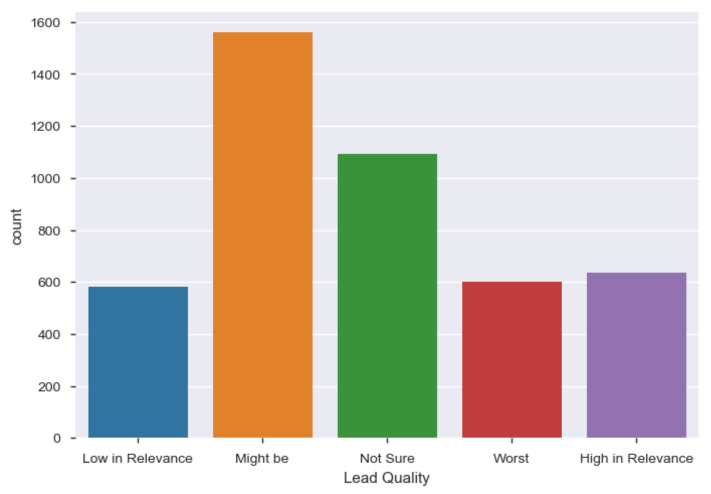
Country : Column

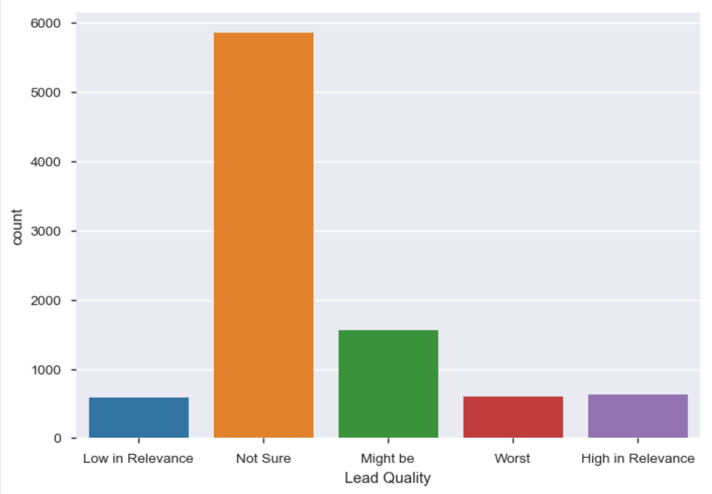
In [466]:
```
lead_df.Country.describe()
```

Out[466]:
```
count      6779
unique       38
top       India
freq       6492
Name: Country, dtype: object
```

DATA CLEANING

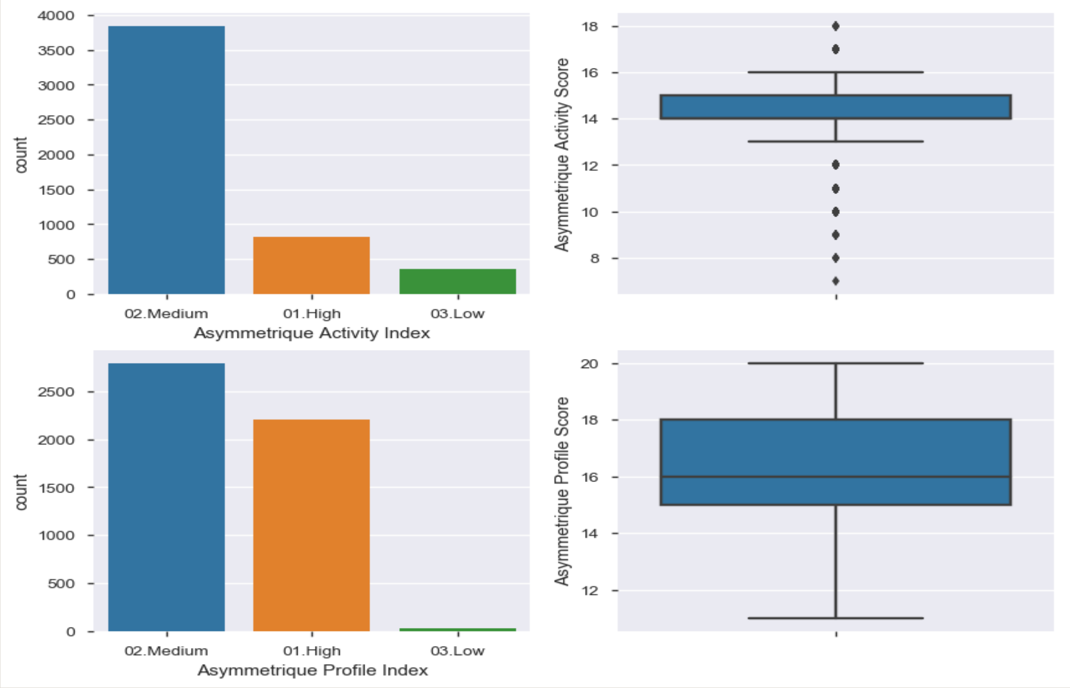
- Imputing Nulls: with Not Sure for Lead Quality





DATA CLEANING

- Identifying outlier:



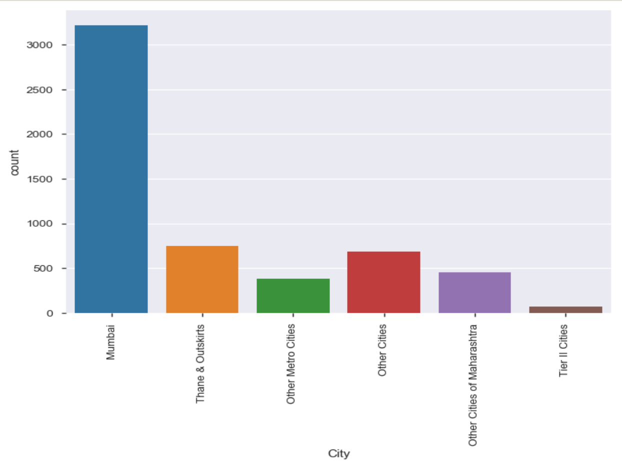
DATA CLEANING

- Imputing Nulls in City:

Here 'Mumbai' on top

```
In [454]:  1  lead_df.City.head()

Out[454]: 0       NaN
          1       NaN
          2    Mumbai
          3    Mumbai
          4    Mumbai
          Name: City, dtype: object
```



DATA CLEANING

- Created a new column “Other Specialization” to impute the value of Specialization

```
2 lead_df.City = lead_df.City.replace(np.nan, 'Mumbai')
```

Specailization : Column

```
In [457]: lead_df.Specialization.describe()

Out[457]: count                   5860
          unique                    18
          top       Finance Management
          freq                     976
          Name: Specialization, dtype: object
```

Top used value is 'Finance Management'

```
In [458]: # The lead don't have any specialization ie NaN (36%) value, So we can create new specialization that is 'Others_Specializa
          # So imputing NaN to 'Others_Specialization'
          lead_df.Specialization = lead_df.Specialization.replace(np.nan, 'Others_Specialization')
```

DATA CLEANING

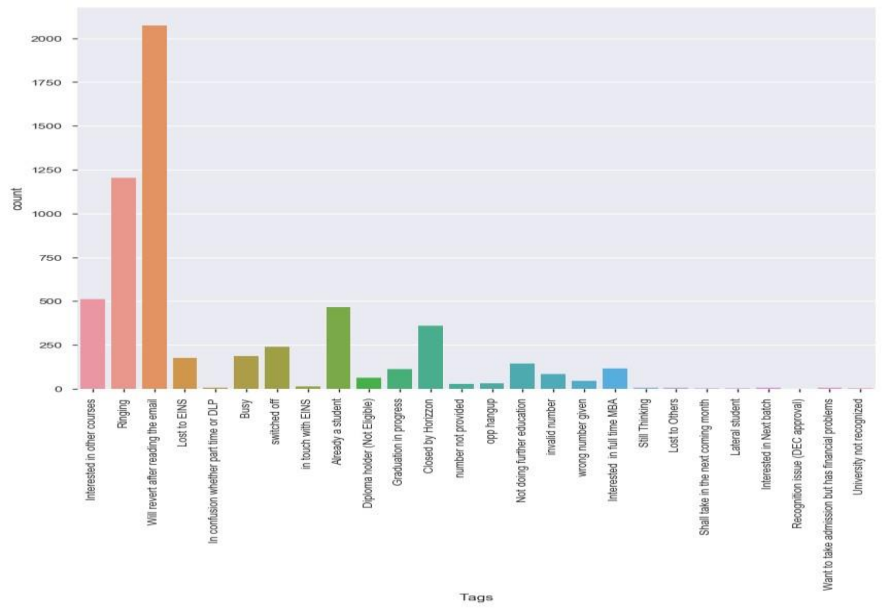
Imputed NAN with most frequent option" Will revert after reading the email')

Tag : Column

In [459]:
```
lead_df.Tags.describe()
```

Out[459]:
```
count                            5887
unique                             26
top       Will revert after reading the email
freq                             2072
Name: Tags, dtype: object
```

'Will revert after reading the email' top used value.



DATA CLEANING

- Dropped the variable having 2% missing values

Now here we can see that columns having missing values is less that 2%. Se can drop it.

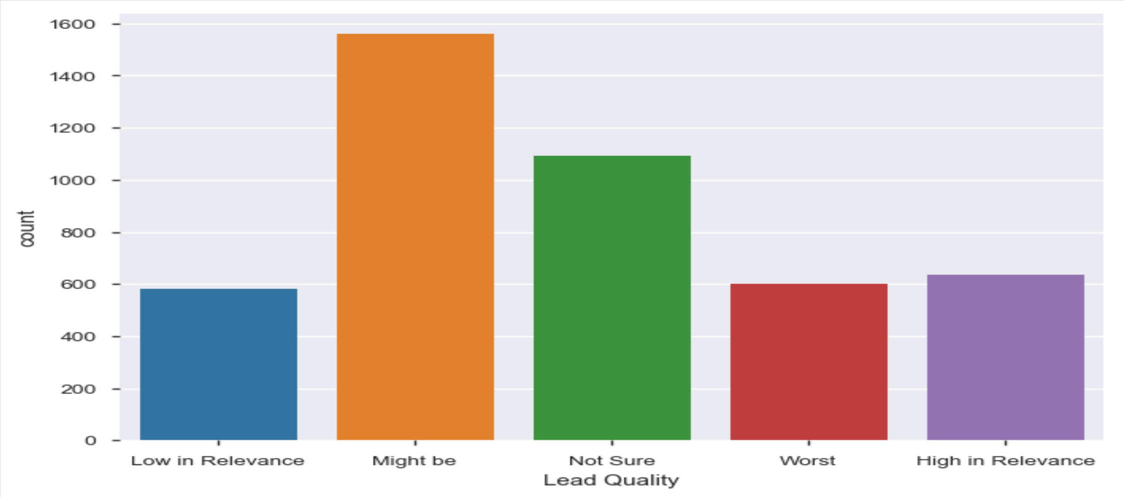
In [469]:
```
# Drop missing values
lead_df.dropna(inplace = True)
```

In [470]:
```
# Checking missing values %.
round(100*(lead_df.isnull().sum()/len(lead_df.index)), 2)
```

Out[470]:
```
Prospect ID                                       0.0
```

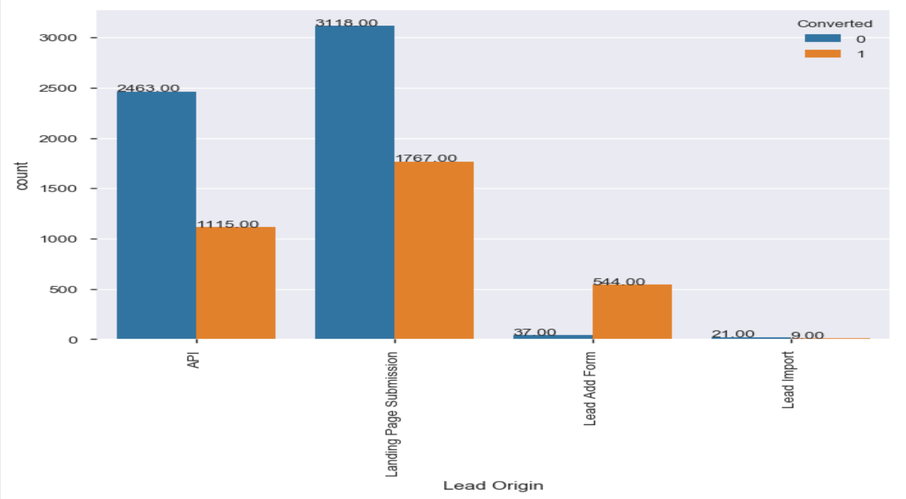
EDA: UNIVARIATE ANALYSIS

- Lead Quality



EDA: UNIVARIATE ANALYSIS

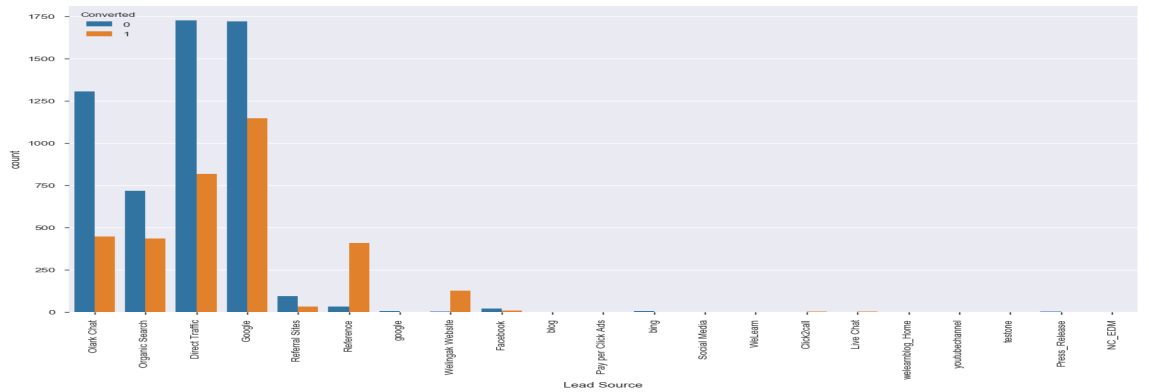
- Lead Origin



[{"metadata":{},"cell_type":"markdown","source":"Inference:\n\n - 'Landing Page Submission' is the top conversion count, than second is 'API' variable.\n - 'Lead Add From' aprox 90% conversion rate but count of lead are not very high.\n - 'Lead Import' having only 9 count of conversion rate.\n \n- So according to the above visualization and Inferences we say that, we need to target 'Landing Page Submission' and 'API' variables to improve the overall lead conversion rate."}]

EDA: UNIVARIATE ANALYSIS

- Lead Source

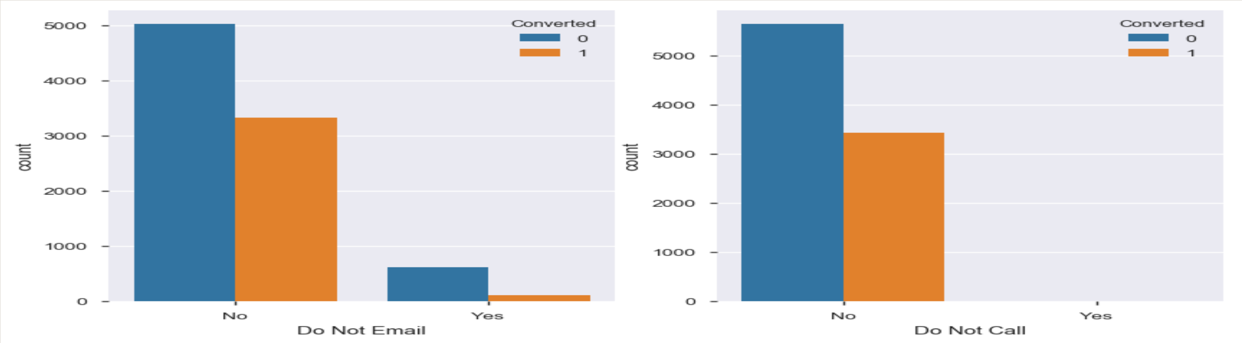


Inference:
- In Lead Source 'Google' having top most conversion count of leads. - Second highest conversation count of 'Direct Traffic' Lead Source.

•So according to the above visualization and Inferences we say that, we need to target Google, Direct, Organic Search, and Olark Chat variables to improve the overall lead conversion rate and generate more leads from reference and welingak website.

EDA: UNIVARIATE ANALYSIS

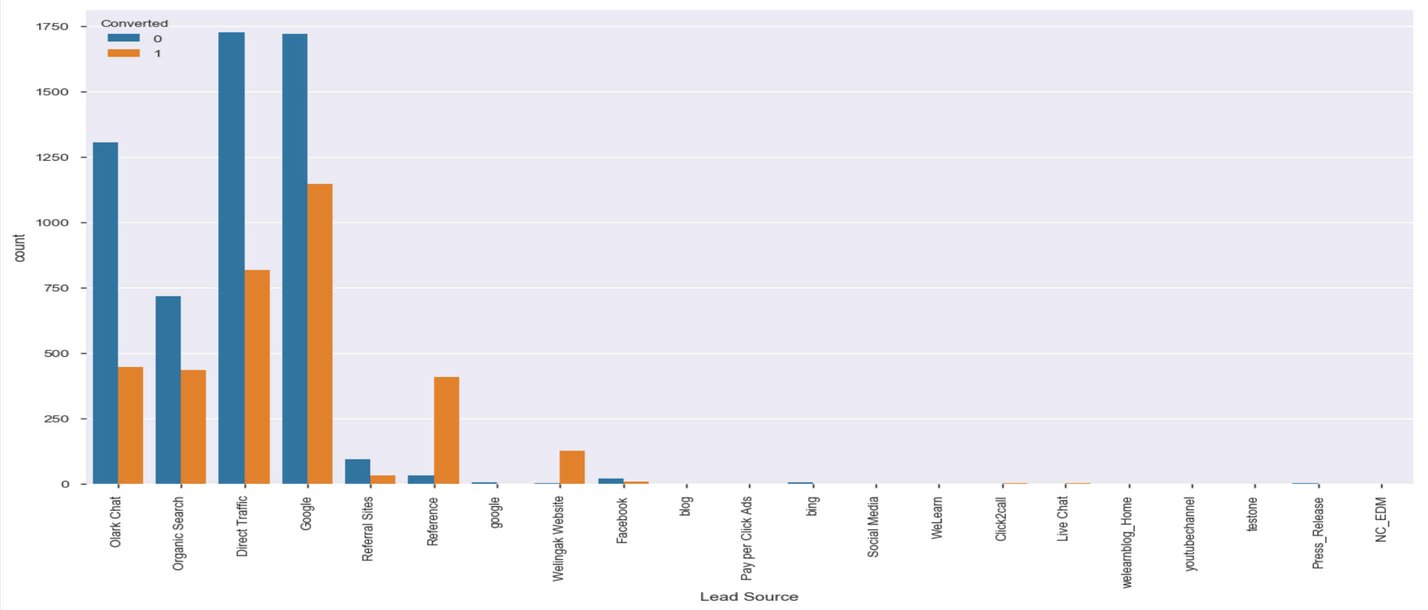
- Do Not Call & Do not Email



Inference:
- In 'Do Not Email' if lead sending email than converstion count is high. - In 'Do Not Call' if lead called than converstion count is high.

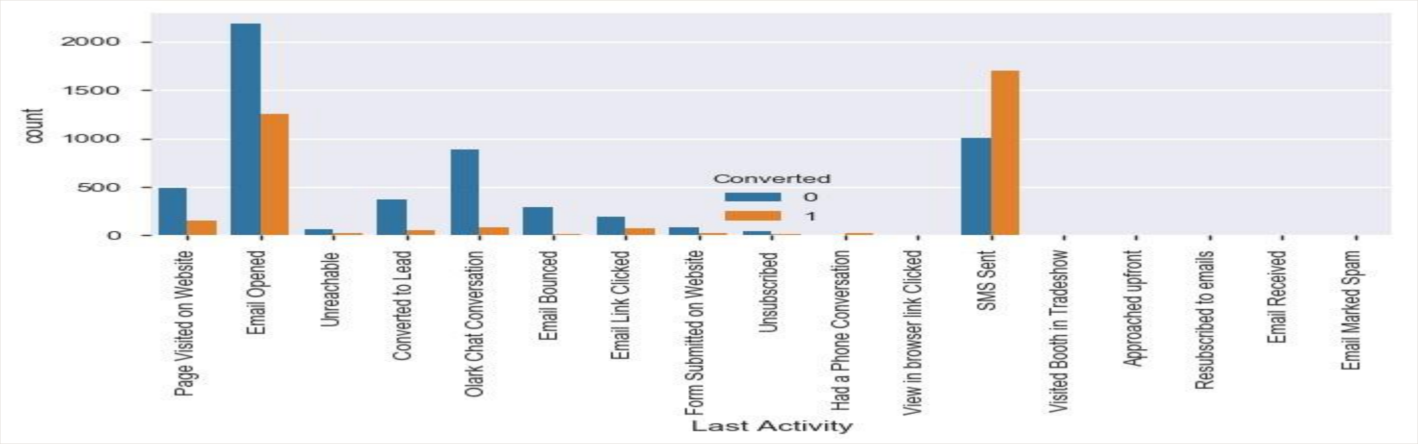
EDA: UNIVARIATE ANALYSIS

- Lead Source



EDA: UNIVARIATE ANALYSIS

- LastActivity



EDA CONTD...

City : Column

In [453]:
```
lead_df.City.describe()
```

Out[453]:
```
count       5571
unique         6
top       Mumbai
freq        3222
Name: City, dtype: object
```

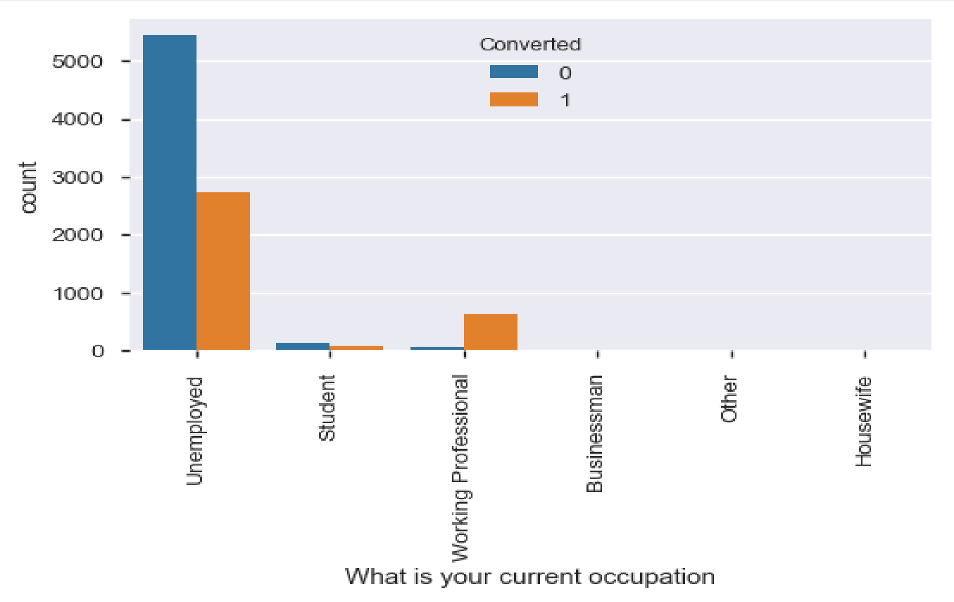
Here 'Mumbai' on top

In [454]:
```
lead_df.City.head()
```

Out[454]:
```
0       NaN
1       NaN
2    Mumbai
3    Mumbai
4    Mumbai
Name: City, dtype: object
```

In [455]:
```
sns.countplot(lead_df.City)
plt.xticks(rotation = 90)
```

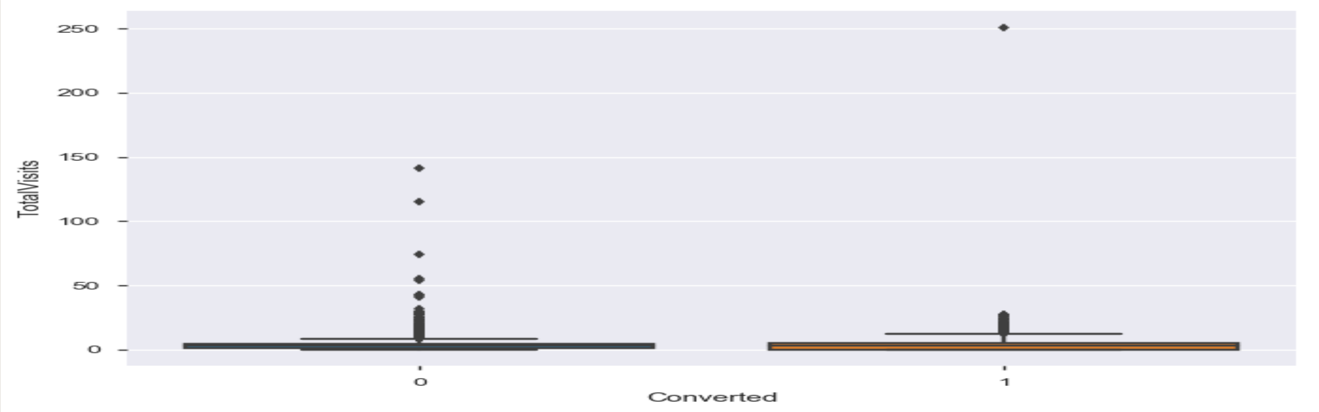
EDA CONTD…



Converted
0
1
count
5000
4000
3000
2000
1000
0
Unemployed
Student
Working Professional
Businessman
Other
Housewife
What is your current occupation

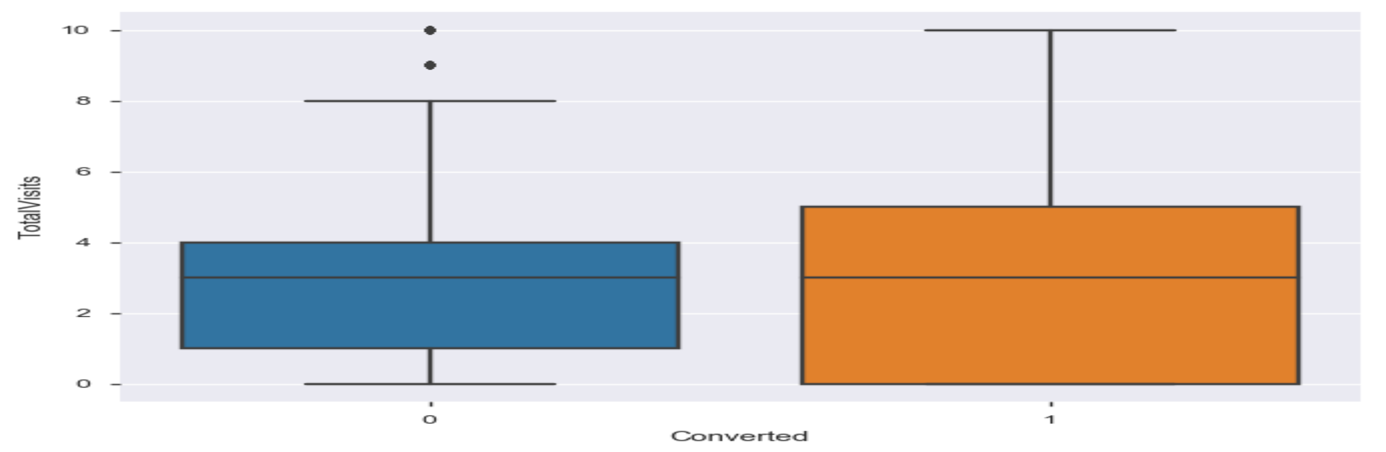
EDA: OUTLIERIDENTIFICATION

- Lead Source




```
# 17.00 data is 99%
# So for analysis 5% to 95%
percentiles = lead_df['TotalVisits'].quantile([0.05,0.95]).values
lead_df['TotalVisits'][lead_df['TotalVisits'] <= percentiles[0]] = percentiles[0]
lead_df['TotalVisits'][lead_df['TotalVisits'] >= percentiles[1]] = percentiles[1]
```

EDA CONTD…



Inference:
- Median of both converted and not converted are equal.

DATA PREPARATION & FEATURE ENGINEERING

Dummy Encoding

- For the following categorical variables with multiple levels, dummy features (one-hot encoded) were created:
- 'Lead Quality','Asymmetrique Profile Index','Asymmetrique Activity Index','Tags','Lead Profile', 'Lead Origin','What is your current occupation', 'Specialization', 'City','Last Activity', 'Country‘ and 'Lead Source','Last Notable Activity'

Test-Train Split

- The original data frame was split into train and test dataset. The train dataset was used to train the model and test dataset was used to evaluate the model.

Feature Scaling

- Scaling helps in interpretation. It is important to have all variables(specially categorical ones which has values 0 and 1) on the same scale for the model to be easily interpretable.
- ‘Standardisation’ was used to scale the data for modelling. It basically brings all of the data into a standard normal distribution with mean at zero and standard deviation one.

DATA PREPARATION & FEATURE ENGINEERING

- Dummy Encoding

For categorical variables with multiple levels, creating dummy features (one-hot encoded)

In [71]:
```
# Creating a dummy variable for some of the categorical variables and dropping the first one.
dummy1 = pd.get_dummies(leads[['Country', 'Lead Source','Lead Origin','Last Notable Activity']], drop_first=True)

# Adding the results to the master dataframe
leads = pd.concat([leads, dummy1], axis=1)
leads.shape
```

Out[71]: (8575, 66)

DATA PREPARATION & FEATURE ENGINEERING

- Binary Encoding

Converting some binary variables (Yes/No) to 0/1

In [70]:

```python
# List of variables to map

varlist =  ['Search','Do Not Email', 'Do Not Call', 'Newspaper Article', 'X Education Forums', 'Newspaper',
             'Digital Advertisement','Through Recommendations','A free copy of Mastering The Interview']

# Defining the map function
def binary_map(x):
    return x.map({'Yes': 1, "No": 0})

# Applying the function to the housing list
leads[varlist] = leads[varlist].apply(binary_map)
leads.head()
```

Out[70]:

Total Time

FEATURE SELECTION USING RFE

- **Recursive feature elimination** is an optimization technique for finding the best performing subset of features. It is based on the idea of repeatedly constructing a model and choosing either the best (based on coefficients), setting the feature aside and then repeating the process with the rest of the features.This process is applied until all the features in the dataset are exhausted. Features are then ranked according to when they were eliminated.

FEATURE SELECTION USING RFE

Step 7: Feature Selection Using RFE

[88]:

```python
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
```

[89]:

```python
from sklearn.feature_selection import RFE
rfe = RFE(logreg, 20)             # running RFE with 20 variables as output
rfe = rfe.fit(X_train, y_train)
```

[90]:

```python
rfe.support_
```

[90]:

```
array([False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False,  True,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False,  True, False, False, False, False, False,  True, False,
       False,  True, False, False, False, False, False,  True, False,
        True, False, False, False,  True, False, False, False, False,
        True, False,  True,  True, False,  True,  True, False, False,
       False, False, False, False, False, False, False, False, False,
```

RECURSIVE FEATURE ELIMINATION

In [558]:

```python
#  Using RFE for variables selection

# Importing LogisticRegression
from sklearn.linear_model import LogisticRegression

# Creating Object
logreg = LogisticRegression()

# Importing RFE
from sklearn.feature_selection import RFE
rfe = RFE(logreg, 15)
rfe = rfe.fit(X_train, y_train)
```

In [559]:

```python
rfe.support_
```

Out[559]:

```
array([ True, False, False, False, False, False,  True, False, False,
       False, False, False, False, False, False,  True, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
       False, False, False,  True,  True,  True, False, False,  True,
       False, False,  True,  True,  True,  True,  True, False, False,
        True,  True, False, False, False, False, False, False, False,
       False, False, False, False, False, False, False, False, False,
        True, False, False, False])
```

BUILDING MODEL

- Generalized Linear Models from Stats Models is used to build the Logistic Regression model.
- The model is built initially with the 15 variables selected by RFE.
- Unwanted features are dropped serially after checking p values (<0.5) and VIF (< 5) and model is built multiple times.
- The final model with 16 features, passes both the significance test and the multi-collinearity test.

BUILDING MODEL

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5871
Model Family:	Binomial	**Df Model:**	130
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	nan
Date:	Mon, 26 Aug 2019	**Deviance:**	nan
Time:	13:13:45	**Pearson chi2:**	3.54e+18
No. Iterations:	100	**Covariance Type:**	nonrobust

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.282e+15	1.08e+08	-3.03e+07	0.000	-3.28e+15	-3.28e+15
Do Not Email	-5.112e+14	4.66e+06	-1.1e+08	0.000	-5.11e+14	-5.11e+14

PREDICTING THE CONVERSION PROBABILITY & PREDICTIVE COLUMNS

In [95]:
```
# Getting the predicted values on the train set
y_train_pred = res.predict(X_train_sm)
y_train_pred[:10]
```

Out[95]:
```
8529    0.0
7331    0.0
7688    1.0
92      0.0
4908    0.0
451     0.0
4945    0.0
2844    1.0
4355    0.0
7251    0.0
dtype: float64
```

In [96]:
```
# reshaping the numpy array containing predicted values
y_train_pred = y_train_pred.values.reshape(-1)
y_train_pred[:10]
```

Out[96]:
```
array([0., 0., 1., 0., 0., 0., 0., 1., 0., 0.])
```

Creating a dataframe with the actual churn flag and the predicted probabilities

In [97]:
```
y_train_pred_final = pd.DataFrame({'Converted':y_train.values, 'Conversion_Prob':y_train_pred})
y_train_pred_final['LeadID'] = y_train.index
y_train_pred_final.head()
```

FINDING OPTIMAL PROBABILITY THRESHOLD

Step 10: Finding Optimal Cutoff Point

Optimal cutoff probability is that prob where we get balanced sensitivity and specificity

In [174]:

```
# Let's create columns with different probability cutoffs
numbers = [float(x)/10 for x in range(10)]
for i in numbers:
    y_train_pred_final[i]= y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > i else 0)
y_train_pred_final.head()
```

Out[174]:

	Converted	Conversion_Prob	LeadID	predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	0	0.064688	8529	0	1	0	0	0	0	0	0	0	0	0
1	0	0.009566	7331	0	1	0	0	0	0	0	0	0	0	0
2	1	0.762190	7688	1	1	1	1	1	1	1	1	1	0	0
3	0	0.077626	92	0	1	0	0	0	0	0	0	0	0	0
4	0	0.077626	4908	0	1	0	0	0	0	0	0	0	0	0

In [175]:

```
# Now let's calculate accuracy sensitivity and specificity for various probability cutoffs.
cutoff_df = pd.DataFrame( columns = ['prob','accuracy','sensi','speci'])
from sklearn.metrics import confusion_matrix

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

PLOTTING ROC AND CALCULATING AUC

Using the probability threshold value 0f 0.33 on the test dataset to predict if a lead will convert

In [213]:
```
y_pred_final['final_predicted'] = y_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.33 else 0)
```

In [214]:
```
y_pred_final.head()
```

Out[214]:

	LeadID	Converted	Conversion_Prob	final_predicted
0	6190	0	0.000591	0
1	7073	0	0.077626	0
2	4519	0	0.309185	0
3	607	1	0.999825	1
4	440	0	0.077626	0

In [215]:
```
# Let's check the overall accuracy.
acc_score=metrics.accuracy_score(y_pred_final.Converted, y_pred_final.final_predicted)
acc_score
```

Out[215]: 0.9055577147298873

In [216]:
```
confusion_test = metrics.confusion_matrix(y_pred_final.Converted, y_pred_final.final_predicted )
print(confusion_test)
```

```
[[1445  132]
 [ 111  885]]
```

PRECISION & RECALL

F1 = 2×(Precision*Recall)/(Precision+Recall)

In [227]:
```
F1 = 2*(Precision*Recall)/(Precision+Recall)
F1
```

Out[227]: 0.879284649776453

Classification Report

In [228]:
```
from sklearn.metrics import classification_report
print(classification_report(y_pred_final.Converted, y_pred_final.final_predicted))
```

	precision	recall	f1-score	support
0	0.93	0.92	0.92	1577
1	0.87	0.89	0.88	996
avg / total	0.91	0.91	0.91	2573

LEAD SCORE CALCULATION

Step 13: Calculating Lead score for the entire dataset

Lead Score = 100 * ConversionProbability

This needs to be calculated for all the leads from the original dataset (train + test)

[235]:
```
# Selecting the test dataset along with the Conversion Probability and final predicted value
leads_test_pred = y_pred_final.copy()
leads_test_pred.head()
```

[235]:

	LeadID	Converted	Conversion_Prob	final_predicted
0	6190	0	0.000591	0
1	7073	0	0.077626	0
2	4519	0	0.309185	0
3	607	1	0.999825	1
4	440	0	0.077626	0

[236]:
```
# Selecting the train dataset along with the Conversion Probability and final predicted valu
leads_train_pred = y_train_pred_final.copy()
leads_train_pred.head()
```

DETERMINING FEATURE IMPORTANCE

Selecting the coefficients of the selected features from our final model excluding the intercept

In [249]:
```
pd.options.display.float_format = '{:.2f}'.format
new_params = res.params[1:]
new_params
```

Out[249]:
```
Lead Source_Welingak Website                          3.61
Lead Quality_Worst                                   -3.18
Asymmetrique Activity Index_03.Low                   -2.34
Tags_Already a student                               -3.45
Tags_Closed by Horizzon                               5.44
Tags_Interested  in full time MBA                    -2.66
Tags_Interested in other courses                     -2.63
Tags_Lost to EINS                                     6.71
Tags_Not doing further education                     -3.35
Tags_Ringing                                         -3.84
Tags_Will revert after reading the email              3.87
Tags_opp hangup                                      -3.08
Tags_switched off                                    -4.73
What is your current occupation_Unemployed            1.67
What is your current occupation_Working Professional  1.89
Last Activity_SMS Sent                                1.97
dtype: float64
```

Getting a relative coeffient value for all the features wrt the feature with the highest coefficient

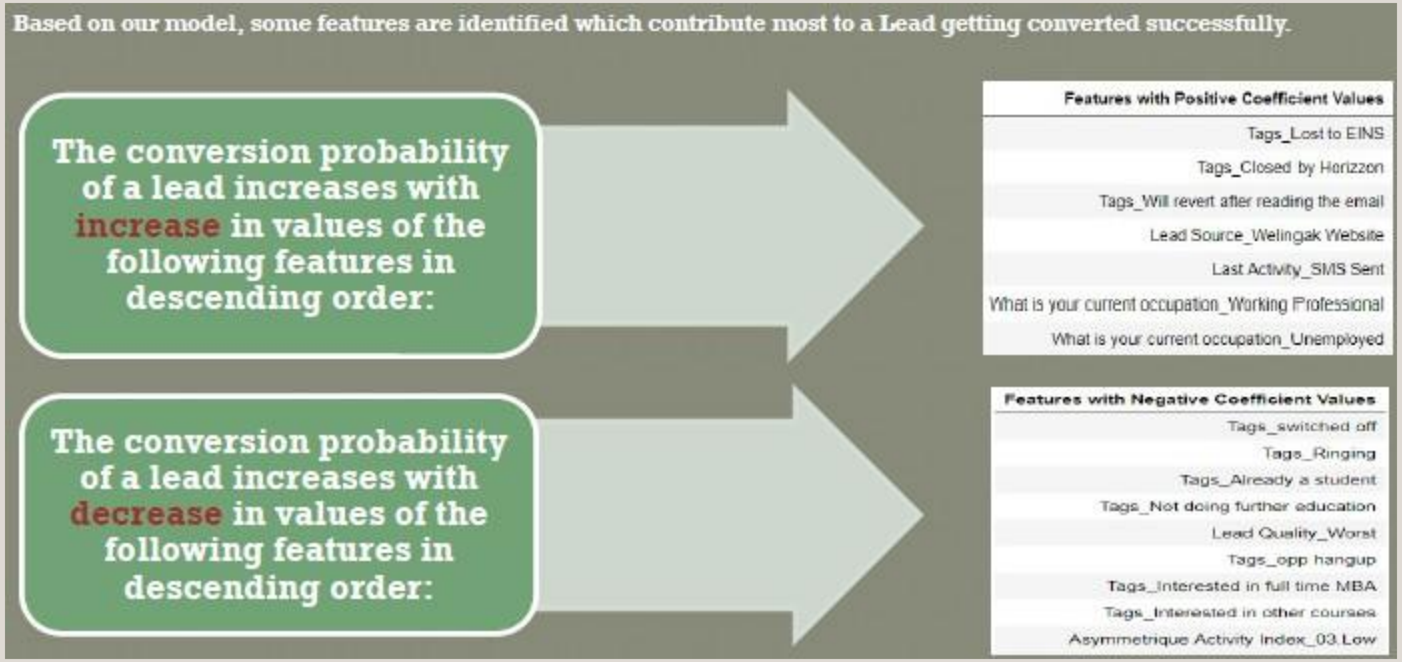
INFERENCE

- We tried several models, can conclude below points:
  - All variables have p-value < 0.5
  - All features have very low vif values meaning there is hardly any multicollinearity among the features
  - This is also evident from the heat map
  - The overall accuracy of 0.905 at a probability threshold of 0.33 on test dataset is also acceptable

INFERENCE CONTD….

- Using mo… of
  Conversi…

INFERENCE CONTD….



RECOMMENDATIONS

- Top three variable in the model that contribute most towards the probability of lead getting converted
  - Tags_Lost to EINS
  - Tags_Closed by Horizon
  - Tags will revert after reading the email

RECOMMENDATION

- Top three categorical /dummy variables
  - Tags_Lost to EINS
  - Tags_Closed by Horizon
  - Tags will revert after reading the email

RECOMMENDATION CONTD…

- X Education has a period of 2 months every year during which they hire few interns. The sales team, in particular, has around 10 interns allotted to them. So, during this phase, they wish to make the lead conversion more aggressive. So they want almost all of the potential leads (i.e. the customers who have been predicted as 1 by the model) to be converted and hence, want to make phone calls to as much of such people as possible. Suggest a good strategy they should employ at this stage.
    - We will choose a lower threshold value for Conversion Probability. This will ensure the Sensitivity rating is very high which in turn will make sure almost all leads that are likely to Convert are identified correctly and the agents can make phone calls to as much of such people as possible.

RECOMMENDATION CONTD…

- **Similarly, at times, the company reaches its target for a quarter before the deadline. During this time, the company wants the sales team to focus on some new work as well. So during this time, the company's aim is to not make phone calls unless it's extremely necessary, i.e. they want to minimize the rate of useless phone calls. Suggest a strategy they should employ at this stage.**
  - We will choose a higher threshold value for Conversion Probability.This will ensure the Specificity rating is very high, which in turn will make sure almost all leads that are on the brink of the probability of getting Converted or not are not selected. As a result the agents won't have to make unnecessary phone calls and can focus on some new work.

CONCLUSION

Selecting `Top 3` features which contribute most towards the probability of a lead getting converted

[257]:
```
pd.DataFrame(feature_importance).reset_index().sort_values(by=0,ascending=False).head(3)
```

[257]:

	index	0
7	Tags_Lost to EINS	100.00
4	Tags_Closed by Horizzon	81.12
10	Tags_Will revert after reading the email	57.67

Step 15: Conclusion

After trying several models, we finally chose a model with the following characteristics:

- All variables have p-value < `0.05`.
- All the features have very low VIF values, meaning, there is `hardly any muliticollinearity` among the features. This is also evident from the heat map.
- The overall accuracy of `0.9056` at a probability threshold of 0.33 on the test dataset is also very acceptable.

Using this model, the dependent variable value was predicted as per the following threshold values of Conversion probability:

Dataset	Threshhold value	Accuracy	Sensitivity	Specificity	False Postive Rate	Positive Predictive Value	Negative Predictive value	Precision	Recall	F1 value	Cross Validation Score	AUC
train	0.50	0.9125	0.8195	0.9690	0.0310	0.9412	0.8985	|	|	0.9624		