LEAD CASE STUDY

step1: Importing and Merging Data

In [1]:

```
# Suppressing Warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
# Importing Pandas and NumPy
import pandas as pd, numpy as np
```

In [3]:

```
# Importing all datasets
leads = pd.read_csv("E:/301/Leads.csv")
leads.head()
```

Out[3]:

	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	As A
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	...	No	Select	Select	
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0	674	2.5	...	No	Select	Select	
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	660727	Landing Page Submission	Direct Traffic	No	No	1	2.0	1532	2.0	...	No	Potential Lead	Mumbai	
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	No	0	1.0	305	1.0	...	No	Select	Mumbai	
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	No	1	2.0	1428	1.0	...	No	Select	Mumbai	

5 rows × 37 columns

Step 2: Inspecting the Dataframe¶

In [4]:

```
leads.dtypes
```

Out[4]:

```
Prospect ID                                      object
Lead Number                                       int64
Lead Origin                                      object
Lead Source                                      object
Do Not Email                                     object
Do Not Call                                      object
```

```
Converted                                          int64
TotalVisits                                      float64
Total Time Spent on Website                        int64
Page Views Per Visit                             float64
Last Activity                                     object
Country                                           object
Specialization                                    object
How did you hear about X Education                object
What is your current occupation                   object
What matters most to you in choosing a course     object
Search                                            object
Magazine                                          object
Newspaper Article                                 object
X Education Forums                                object
Newspaper                                         object
Digital Advertisement                             object
Through Recommendations                           object
Receive More Updates About Our Courses            object
Tags                                              object
Lead Quality                                      object
Update me on Supply Chain Content                 object
Get updates on DM Content                         object
Lead Profile                                      object
City                                              object
Asymmetrique Activity Index                       object
Asymmetrique Profile Index                        object
Asymmetrique Activity Score                      float64
Asymmetrique Profile Score                       float64
I agree to pay the amount through cheque          object
A free copy of Mastering The Interview            object
Last Notable Activity                             object
dtype: object
```

In [5]:

```
leads.shape
```

Out[5]:

```
(9240, 37)
```

Step 3: Data Preparation

In [6]:

```
# removing duplicate rows
leads.drop_duplicates(subset='Lead Number')
leads.shape
```

Out[6]:

```
(9240, 37)
```

In [7]:

```
# Checking for total count and percentage of null values in all columns of the dataframe.

total = pd.DataFrame(leads.isnull().sum().sort_values(ascending=False), columns=['Total'])
percentage = pd.DataFrame(round(100*(leads.isnull().sum()/leads.shape[0]),2).sort_values(ascending=
False)\
                          ,columns=['Percentage'])
pd.concat([total, percentage], axis = 1)
```

Out[7]:

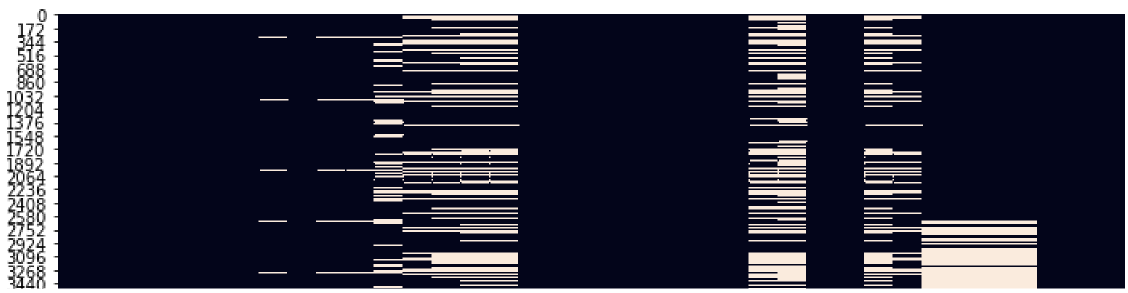
	Total	Percentage
Lead Quality	4767	51.59
Asymmetrique Profile Score	4218	45.65
Asymmetrique Activity Score	4218	45.65

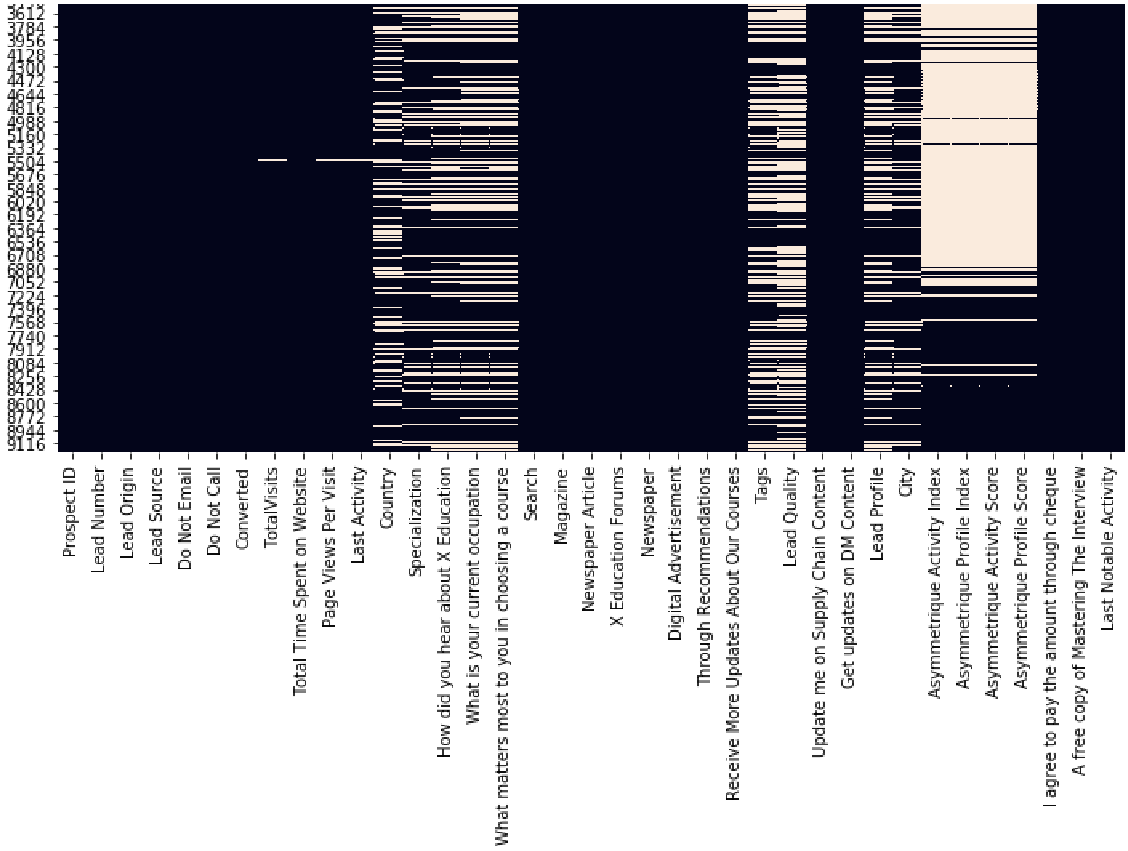
	Total	Percentage
Asymmetrique Activity Score	4218	45.65
Asymmetrique Profile Index	4218	45.65
Asymmetrique Activity Index	4218	45.65
Tags	3353	36.29
What matters most to you in choosing a course	2709	29.32
Lead Profile	2709	29.32
What is your current occupation	2690	29.11
Country	2461	26.63
How did you hear about X Education	2207	23.89
Specialization	1438	15.56
City	1420	15.37
TotalVisits	137	1.48
Page Views Per Visit	137	1.48
Last Activity	103	1.11
Lead Source	36	0.39
Do Not Email	0	0.00
Do Not Call	0	0.00
Converted	0	0.00
Total Time Spent on Website	0	0.00
Lead Origin	0	0.00
Lead Number	0	0.00
Last Notable Activity	0	0.00
Newspaper Article	0	0.00
Search	0	0.00
Magazine	0	0.00
A free copy of Mastering The Interview	0	0.00
X Education Forums	0	0.00
Newspaper	0	0.00
Digital Advertisement	0	0.00
Through Recommendations	0	0.00
Receive More Updates About Our Courses	0	0.00
Update me on Supply Chain Content	0	0.00
Get updates on DM Content	0	0.00
I agree to pay the amount through cheque	0	0.00
Prospect ID	0	0.00

Visualizing occurence of Null values in the columns based on rows

In [8]:

```python
import matplotlib.pyplot as plt,seaborn as sns
plt.figure(figsize=(10,10))
sns.heatmap(leads.isnull(), cbar=False)

plt.tight_layout()
plt.show()
```





Dropping Unnecessary Columns

In [9]:

```
# Identifying if any column exists with only null values
leads.isnull().all(axis=0).any()
```

Out[9]:

```
False
```

In [10]:

```
# Dropping all columns with only 0 values
leads.loc[:, (leads != 0).any(axis=0)]
leads.shape
```

Out[10]:

```
(9240, 37)
```

In [11]:

```
leads= leads.loc[:,leads.nunique()!=1]
leads.shape
```

Out[11]:

```
(9240, 32)
```

In [12]:

```
# Deleting the columns 'Asymmetrique Activity Score' & 'Asymmetrique Profile Score'
# as they will be represented by their corresponding index columns
leads = leads.drop('Asymmetrique Activity Score', axis=1)
leads = leads.drop('Asymmetrique Profile Score', axis=1)
leads.shape
```

Out[12]:

(9240, 30)

In [13]:

```
# Deleting the columns 'Prospect ID' as it will not have any effect in the predicting model
leads = leads.drop('Prospect ID', axis=1)
#leads = leads.drop('Lead Number', axis=1)
leads.shape
```

Out[13]:

(9240, 29)

In [14]:

```
#Deleting the columns 'What matters most to you in choosing a course' as it mostly has unique valu
es and some null values.
leads = leads.drop('What matters most to you in choosing a course', axis=1)
leads.shape
```

Out[14]:

(9240, 28)

In [15]:

```
# Deleting the columns 'How did you hear about X Education' as it mostly has null values or 'Selec
t' values
# that contribute to the 'Converted' percentage.
leads = leads.drop('How did you hear about X Education', axis=1)
leads.shape
```

Out[15]:

(9240, 27)

Removing rows where a particular column has high missing values

In [16]:

```
leads['Lead Source'].isnull().sum()
```

Out[16]:

36

In [17]:

```
# removing rows where a particular column has high missing values because the column cannot be rem
oved because of its importance
leads = leads[~pd.isnull(leads['Lead Source'])]
leads.shape
```

Out[17]:

(9204, 27)

Imputing with Median values because the continuous variables have outliers

In [18]:

```
leads['TotalVisits'].replace(np.NaN, leads['TotalVisits'].median(), inplace =True)
```

In [19]:

In [19]:

```
leads['Page Views Per Visit'].replace(np.NaN, leads['Page Views Per Visit'].median(), inplace
=True)
```

Imputing with Mode values

In [20]:

```
leads['Country'].mode()
```

Out[20]:

```
0    India
dtype: object
```

In [21]:

```
leads.loc[pd.isnull(leads['Country']), ['Country']] = 'India'
```

In [22]:

```
leads['Country'] = leads['Country'].apply(lambda x: 'India' if x=='India' else 'Outside India')
leads['Country'].value_counts()
```

Out[22]:

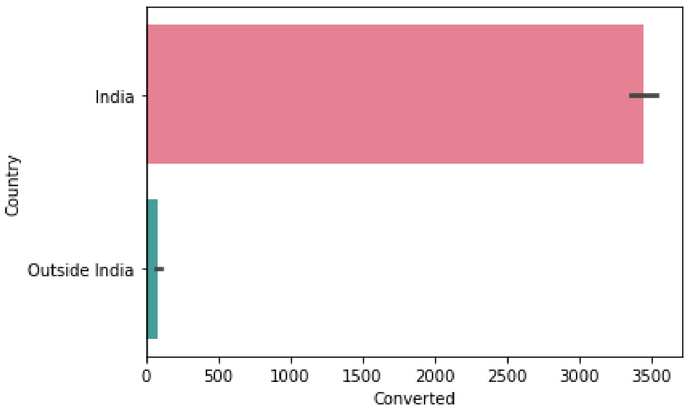
```
India             8917
Outside India      287
Name: Country, dtype: int64
```

In [23]:

```
sns.barplot(y='Country', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[23]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e3b6e5e0>
```



Assigning An Unique Category to NULL/SELECT values

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'

In [24]:

```
leads['Lead Quality'].value_counts()
```

Out[24]:

```
Might be               1545
Not Sure               1090
High in Relevance       632
Worst                   601
Low in Relevance        583
Name: Lead Quality, dtype: int64
```

In [25]:

```
leads['Lead Quality'].isnull().sum()
```

Out[25]:

```
4753
```

In [26]:

```
leads['Lead Quality'].fillna("Unknown", inplace = True)
leads['Lead Quality'].value_counts()
```

Out[26]:

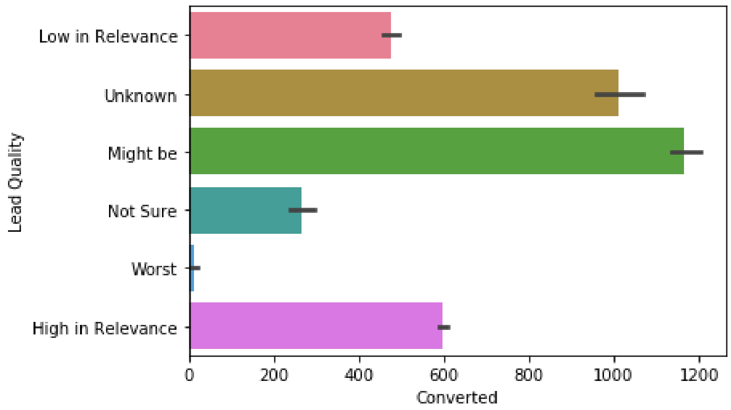
```
Unknown                4753
Might be               1545
Not Sure               1090
High in Relevance       632
Worst                   601
Low in Relevance        583
Name: Lead Quality, dtype: int64
```

In [27]:

```
sns.barplot(y='Lead Quality', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[27]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e3b6ec40>
```



Creating a new category consisting on NULL/Select values for the field Asymmetrique Profile Index

In [28]:

```
leads['Asymmetrique Profile Index'].value_counts()
```

Out[28]:

```
02.Medium    2771
01.High      2201
03.Low         31
Name: Asymmetrique Profile Index, dtype: int64
```

In [29]:

```
leads['Asymmetrique Profile Index'].isnull().sum()
```

Out[29]:

```
4201
```

In [30]:

```
leads['Asymmetrique Profile Index'].fillna("Unknown", inplace = True)
leads['Asymmetrique Profile Index'].value_counts()
```

Out[30]:

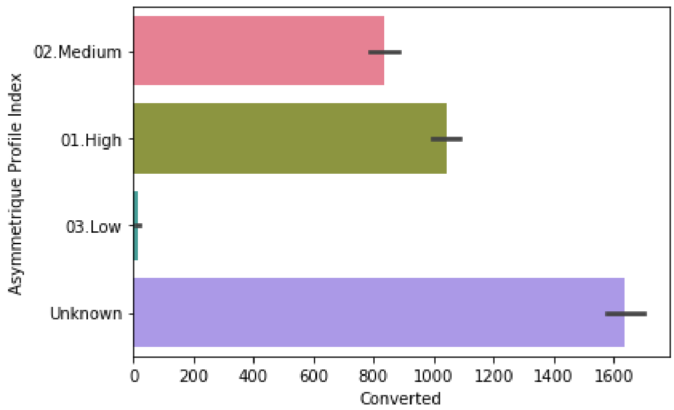
```
Unknown      4201
02.Medium    2771
01.High      2201
03.Low         31
Name: Asymmetrique Profile Index, dtype: int64
```

In [31]:

```
sns.barplot(y='Asymmetrique Profile Index', x='Converted', palette='husl', data=leads, estimator=np
.sum)
```

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e3954580>
```



In [32]:

```
#for Asymmetrique Activity Index
```

In [33]:

```
leads['Asymmetrique Activity Index'].value_counts()
leads['Asymmetrique Activity Index'].isnull().sum()
leads['Asymmetrique Activity Index'].fillna("Unknown", inplace = True)
leads['Asymmetrique Activity Index'].value_counts()
```

Out[33]:

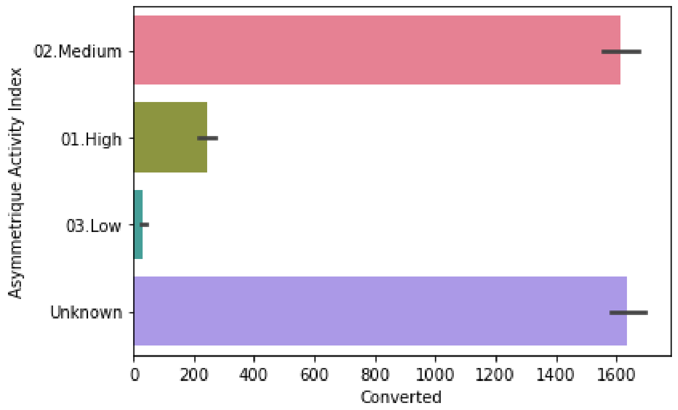
```
Unknown      4201
02.Medium    3820
01.High       821
03.Low        362
Name: Asymmetrique Activity Index, dtype: int64
```

In [34]:

```
sns.barplot(y='Asymmetrique Activity Index', x='Converted', palette='husl', data=leads, estimator=n
p.sum)
```

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e39c02b0>
```



In [35]:

```
#for City
```

In [36]:

```
leads['City'].isnull().sum()
leads['City'].fillna("Unknown", inplace = True)
leads['City'].value_counts()
leads['City'].replace('Select', 'Unknown', inplace =True)
leads['City'].value_counts()
```

Out[36]:

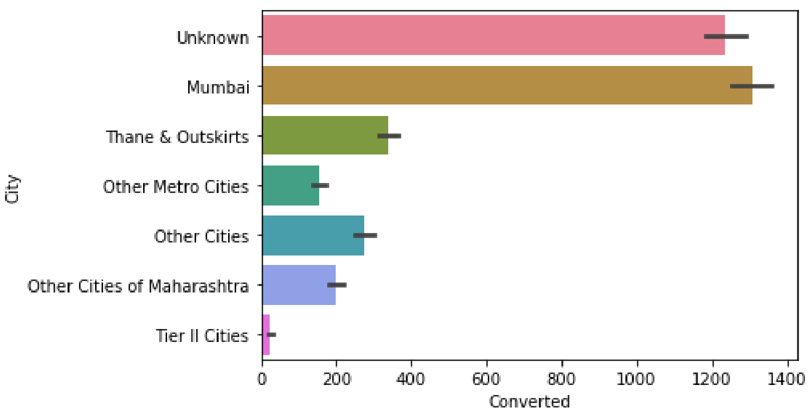
```
Unknown                        3638
Mumbai                         3220
Thane & Outskirts               751
Other Cities                    686
Other Cities of Maharashtra     456
Other Metro Cities              379
Tier II Cities                   74
Name: City, dtype: int64
```

In [37]:

```
sns.barplot(y='City', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[37]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e3aa0310>
```



In [38]:

```
#FOR Last Activity
```

In [39]:

```
leads['Last Activity'].value_counts()
leads['Last Activity'].isnull().sum()
leads['Last Activity'].fillna("Unknown", inplace = True)
leads['Last Activity'].value_counts()
```

Out[39]:

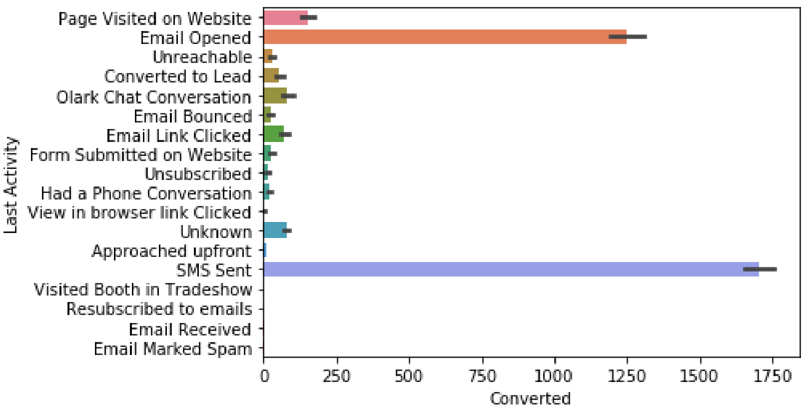
```
Email Opened                    3432
SMS Sent                        2723
Olark Chat Conversation          973
Page Visited on Website          640
Converted to Lead                428
Email Bounced                    321
Email Link Clicked               267
Form Submitted on Website        116
Unknown                          101
Unreachable                       93
Unsubscribed                      59
Had a Phone Conversation          30
Approached upfront                 9
View in browser link Clicked       6
Email Marked Spam                  2
Email Received                     2
Visited Booth in Tradeshow         1
Resubscribed to emails             1
Name: Last Activity, dtype: int64
```

In [40]:

```
sns.barplot(y='Last Activity', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[40]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e43514c0>
```



In [41]:

```
#for What is your current occupation
```

In [42]:

```
leads['What is your current occupation'].value_counts()
leads['What is your current occupation'].isnull().sum()
leads['What is your current occupation'].fillna("Unknown", inplace = True)
leads['What is your current occupation'].value_counts()
```

Out[42]:

```
Unemployed              5567
Unknown                 2690
Working Professional     704
Student                  209
Other                     16
```

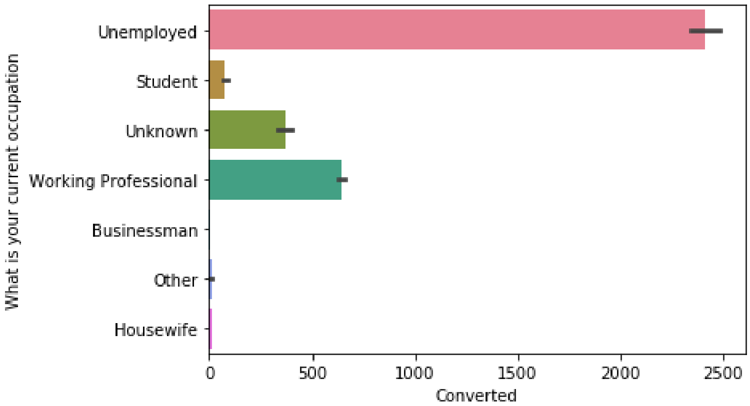
```
Housewife                  10
Businessman                 8
Name: What is your current occupation, dtype: int64
```

In [43]:

```
sns.barplot(y='What is your current occupation', x='Converted', palette='husl', data=leads, estimat
or=np.sum)
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e4412040>
```



In [44]:

```
#for Lead Profile
```

In [45]:

```
leads['Lead Profile'].value_counts()
leads['Lead Profile'].isnull().sum()
leads['Lead Profile'].fillna("Unknown", inplace = True)
leads['Lead Profile'].value_counts()
```

Out[45]:

```
Select                         4115
Unknown                        2709
Potential Lead                 1608
Other Leads                     487
Student of SomeSchool           241
Lateral Student                  24
Dual Specialization Student      20
Name: Lead Profile, dtype: int64
```

In [46]:

```
leads['Lead Profile'].replace('Select', 'Unknown', inplace =True)
leads['Lead Profile'].value_counts()
```

Out[46]:

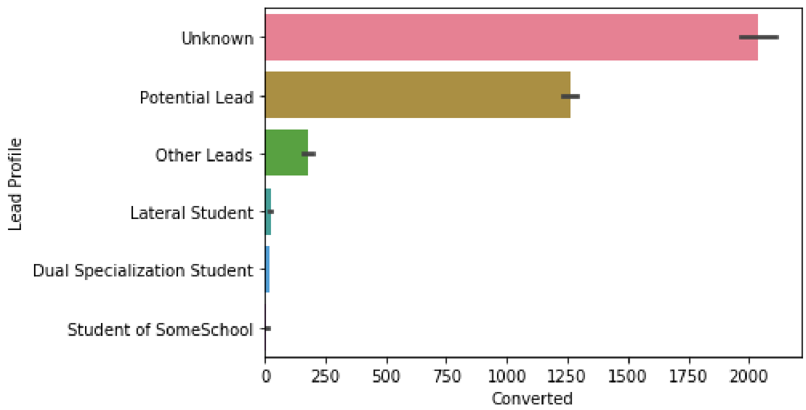
```
Unknown                        6824
Potential Lead                 1608
Other Leads                     487
Student of SomeSchool           241
Lateral Student                  24
Dual Specialization Student      20
Name: Lead Profile, dtype: int64
```

In [47]:

```
sns.barplot(y='Lead Profile', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[47]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e4485b20>
```



In [48]:

```
# for Specialization
```

In [49]:

```
leads['Specialization'].value_counts()
leads['Specialization'].isnull().sum()
leads['Specialization'].fillna("Unknown", inplace = True)
leads['Specialization'].value_counts()
```

Out[49]:

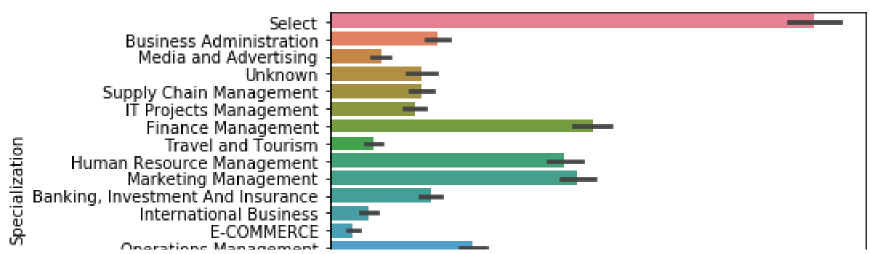
```
Select                               1914
Unknown                              1438
Finance Management                    973
Human Resource Management             847
Marketing Management                  837
Operations Management                 502
Business Administration               403
IT Projects Management                366
Supply Chain Management               349
Banking, Investment And Insurance     338
Media and Advertising                 203
Travel and Tourism                    203
International Business                178
Healthcare Management                 158
Hospitality Management                114
E-COMMERCE                            111
Retail Management                     100
Rural and Agribusiness                 73
E-Business                             57
Services Excellence                    40
Name: Specialization, dtype: int64
```

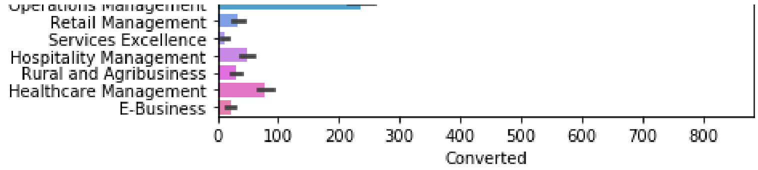
In [50]:

```
sns.barplot(y='Specialization', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[50]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e46ce730>
```





In [51]:

```
# for tags
```

In [52]:

```
leads['Tags'].value_counts()
leads['Tags'].isnull().sum()
leads['Tags'].fillna("Unknown", inplace = True)
leads['Tags'].value_counts()
```

Out[52]:

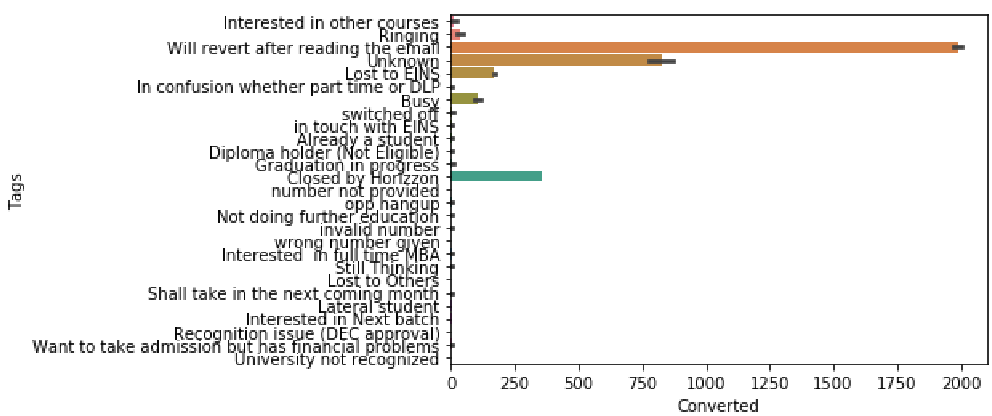
```
Unknown                                              3342
Will revert after reading the email                  2052
Ringing                                              1200
Interested in other courses                           513
Already a student                                     465
Closed by Horizzon                                    358
switched off                                          240
Busy                                                  186
Lost to EINS                                          174
Not doing further education                           145
Interested  in full time MBA                          117
Graduation in progress                                111
invalid number                                         83
Diploma holder (Not Eligible)                          63
wrong number given                                     47
opp hangup                                             33
number not provided                                    26
in touch with EINS                                     12
Lost to Others                                          7
Still Thinking                                          6
Want to take admission but has financial problems       6
Interested in Next batch                                5
In confusion whether part time or DLP                   5
Lateral student                                         3
Shall take in the next coming month                     2
University not recognized                               2
Recognition issue (DEC approval)                        1
Name: Tags, dtype: int64
```

In [53]:

```
sns.barplot(y='Tags', x='Converted', palette='husl', data=leads, estimator=np.sum)
```

Out[53]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x211e3b23b20>
```



In [54]:

```
leads['Lead Quality'].value_counts()
leads['Lead Quality'].isnull().sum()
leads['Lead Quality'].fillna("Unknown", inplace = True)
leads['Lead Quality'].value_counts()
```

Out[54]:

```
Unknown              4753
Might be             1545
Not Sure             1090
High in Relevance     632
Worst                 601
Low in Relevance      583
Name: Lead Quality, dtype: int64
```

Reinspecting Null Values

In [55]:

```
# Checking for total count and percentage of null values in all columns of the dataframe.

total = pd.DataFrame(leads.isnull().sum().sort_values(ascending=False), columns=['Total'])
percentage = pd.DataFrame(round(100*(leads.isnull().sum()/leads.shape[0]),2).sort_values(ascending=
False)\
                          ,columns=['Percentage'])
pd.concat([total, percentage], axis = 1).head()
```

Out[55]:

	Total	Percentage
Last Notable Activity	0	0.0
What is your current occupation	0	0.0
Lead Origin	0	0.0
Lead Source	0	0.0
Do Not Email	0	0.0

Checking for Outliers

In [56]:

```
# Checking outliers at 25%,50%,75%,90%,95% and 99%
leads.describe(percentiles=[.25,.5,.75,.90,.95,.99]).T
```

Out[56]:

	count	mean	std	min	25%	50%	75%	90%	95%	99%	max
Lead Number	9204.0	617194.608648	23418.830233	579533.0	596484.5	615479.0	637409.25	650513.1	655405.85	659599.46	660737.0
Converted	9204.0	0.383746	0.486324	0.0	0.0	0.0	1.00	1.0	1.00	1.00	1.0
TotalVisits	9204.0	3.449587	4.824662	0.0	1.0	3.0	5.00	7.0	10.00	17.00	251.0
Total Time Spent on Website	9204.0	489.005541	547.980340	0.0	14.0	250.0	938.00	1380.0	1562.00	1839.97	2272.0
Page Views Per Visit	9204.0	2.364923	2.145999	0.0	1.0	2.0	3.00	5.0	6.00	9.00	55.0

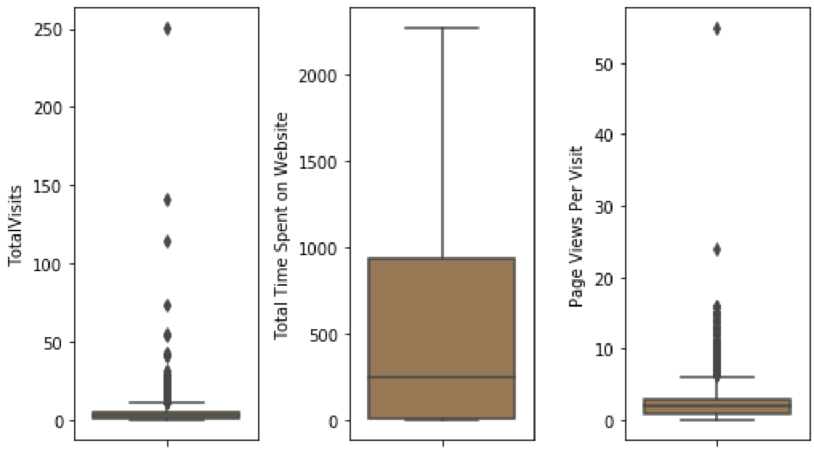
In [57]:

```
numeric_variables = ['TotalVisits','Total Time Spent on Website','Page Views Per Visit']
print(numeric_variables)
```

```
['TotalVisits', 'Total Time Spent on Website', 'Page Views Per Visit']
```

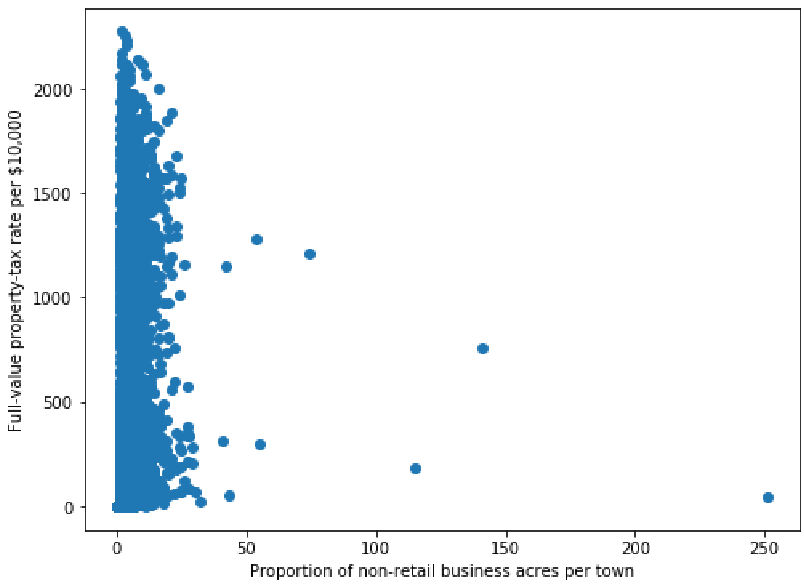
In [58]:

```
numeric_variables = ['TotalVisits','Total Time Spent on Website','Page Views Per Visit']
def boxplot(var_list):
    plt.figure(figsize=(12,8))
    for var in var_list:
        plt.subplot(2,5,var_list.index(var)+1)
        #plt.boxplot(country[var])
        sns.boxplot(y=var,palette='cubehelix', data=leads)
    # Automatically adjust subplot params so that the subplotS fits in to the figure area.
    plt.tight_layout()
    # display the plot
    plt.show()

boxplot(numeric_variables)
```



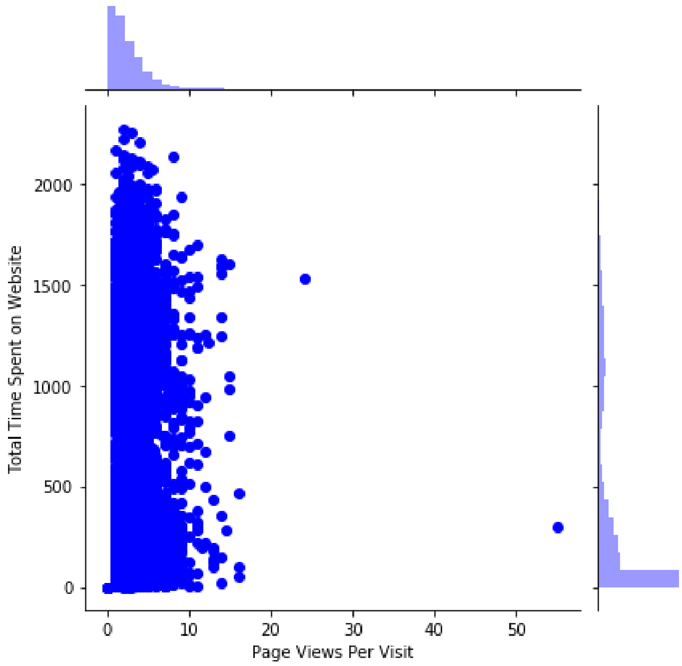
In [59]:

```
fig, ax = plt.subplots(figsize=(8,6))
ax.scatter(leads['TotalVisits'], leads['Total Time Spent on Website'])
ax.set_xlabel('Proportion of non-retail business acres per town')
ax.set_ylabel('Full-value property-tax rate per $10,000')
plt.show()
```



In [60]:

```
sns.jointplot(leads['Page Views Per Visit'],leads['Total Time Spent on Website'], color="b")
plt.show()
```



Removing outlier values based on the Interquartile distance

In [61]:

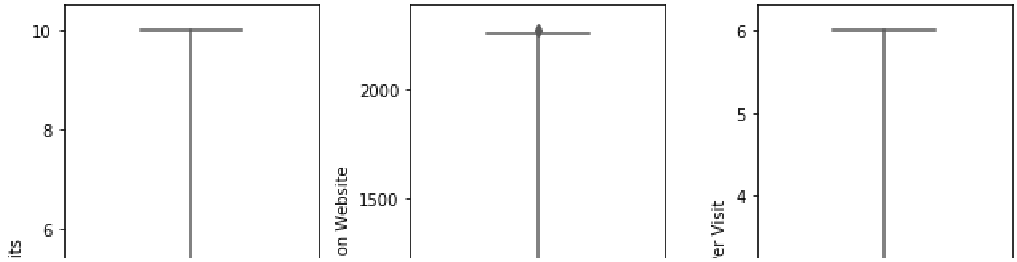
```
Q1 = leads['TotalVisits'].quantile(0.25)
Q3 = leads['TotalVisits'].quantile(0.75)
IQR = Q3 - Q1
leads=leads.loc[(leads['TotalVisits'] >= Q1 - 1.5*IQR) & (leads['TotalVisits'] <= Q3 + 1.4*IQR)]

Q1 = leads['Page Views Per Visit'].quantile(0.25)
Q3 = leads['Page Views Per Visit'].quantile(0.75)
IQR = Q3 - Q1
leads=leads.loc[(leads['Page Views Per Visit'] >= Q1 - 1.5*IQR) & (leads['Page Views Per Visit'] <=
Q3 + 1.5*IQR)]

leads.shape
```

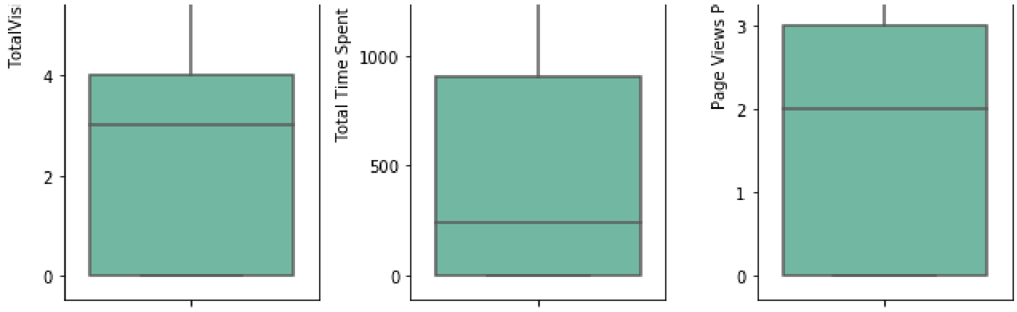
Out[61]:

```
(8575, 27)
```

In [62]:

```
def boxplot(var_list):
    plt.figure(figsize=(15,10))
    for var in var_list:
        plt.subplot(2,5,var_list.index(var)+1)
        #plt.boxplot(country[var])
        sns.boxplot(y=var,palette='BuGn_r', data=leads)
    # Automatically adjust subplot params so that the subplotS fits in to the figure area.
    plt.tight_layout()
    # display the plot
    plt.show()

boxplot(numeric_variables)
```





In [63]:

```
leads.shape
```

Out[63]:

```
(8575, 27)
```

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

In [64]:

```
# 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[64]:

	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	...	Digital Advertisement	Throug Recommendation
0	660737	API	Olark Chat	0	0	0	0.0	0	0.0	Page Visited on Website	...	0	
1	660728	API	Organic Search	0	0	0	5.0	674	2.5	Email Opened	...	0	
2	660727	Landing Page Submission	Direct Traffic	0	0	1	2.0	1532	2.0	Email Opened	...	0	
3	660719	Landing Page Submission	Direct Traffic	0	0	0	1.0	305	1.0	Unreachable	...	0	
4	660681	Landing Page Submission	Google	0	0	1	2.0	1428	1.0	Converted to Lead	...	0	

5 rows × 27 columns

For categorical variables with multiple levels, creating dummy features

In [65]:

```
# 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']]
```

```
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[65]:

```
(8575, 66)
```

In [66]:

```
# Creating dummy variables for the remaining categorical variables and
# dropping the level called 'Unknown' which represents null/select values.

# Creating dummy variables for the variable 'Lead Quality'
ml = pd.get_dummies(leads['Lead Quality'], prefix='Lead Quality')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Lead Quality_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Asymmetrique Profile Index'
ml = pd.get_dummies(leads['Asymmetrique Profile Index'], prefix='Asymmetrique Profile Index')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Asymmetrique Profile Index_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Asymmetrique Activity Index'
ml = pd.get_dummies(leads['Asymmetrique Activity Index'], prefix='Asymmetrique Activity Index')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Asymmetrique Activity Index_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Tags'
ml = pd.get_dummies(leads['Tags'], prefix='Tags')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Tags_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Lead Profile'
ml = pd.get_dummies(leads['Lead Profile'], prefix='Lead Profile')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Lead Profile_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'What is your current occupation'
ml = pd.get_dummies(leads['What is your current occupation'], prefix='What is your current
occupation')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['What is your current occupation_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Specialization'
ml = pd.get_dummies(leads['Specialization'], prefix='Specialization')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['Specialization_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'City'
ml = pd.get_dummies(leads['City'], prefix='City')
# Dropping the level called 'Unknown' which represents null/select values
ml1 = ml.drop(['City_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------
# Creating dummy variables for the variable 'Last Activity'
ml = pd.get_dummies(leads['Last Activity'], prefix='Last Activity')
# Dropping the level called 'Unknown' which represents null/select values
```

```
ml1 = ml.drop(['Last Activity_Unknown'], 1)
#Adding the results to the master dataframe
leads = pd.concat([leads,ml1], axis=1)
#-------------------------------------------------------------------------------------
leads.shape
```

Out[66]:

```
(8575, 156)
```

Dropping the repeated variables

In [67]:

```
leads = leads.drop(['Lead Quality','Asymmetrique Profile Index','Asymmetrique Activity Index','Tag
s','Lead Profile',
                    'Lead Origin','What is your current occupation', 'Specialization', 'City','Last
Activity', 'Country',
                    'Lead Source','Last Notable Activity'], 1)
leads.shape
```

Out[67]:

```
(8575, 143)
```

In [68]:

```
leads.head()
```

Out[68]:

	Lead Number	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	Newspaper Article	X Education Forums	...	Last Activity_Form Submitted on Website	Last Activity_Had a Phone Conversation	Acti Cor
0	660737	0	0	0	0.0	0	0.0	0	0	0	...	0	0	
1	660728	0	0	0	5.0	674	2.5	0	0	0	...	0	0	
2	660727	0	0	1	2.0	1532	2.0	0	0	0	...	0	0	
3	660719	0	0	0	1.0	305	1.0	0	0	0	...	0	0	
4	660681	0	0	1	2.0	1428	1.0	0	0	0	...	0	0	

5 rows × 143 columns

In [69]:

```
# Ensuring there are no categorical columns left in the dataframe
cols = leads.columns
num_cols = leads._get_numeric_data().columns
list(set(cols) - set(num_cols))
```

Out[69]:

```
[]
```

In [70]:

```
# Creating a copy of this origial variable in case if needed later on
original_leads = leads.copy()
print(original_leads.shape)
print(leads.shape)
```

```
(8575, 143)
(8575, 143)
```

Step 4: Test-Train Split

In [71]:

```
from sklearn.model_selection import train_test_split
```

In [72]:

```
# Putting feature variable to X
X = leads.drop(['Converted','Lead Number'], axis=1)

X.head()
```

Out[72]:

	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	...	Last Activity_Form Submitted on Website	Las Activity_Ha a Phon Conversatio
0	0	0	0.0	0	0.0	0	0	0	0	0	...	0	
1	0	0	5.0	674	2.5	0	0	0	0	0	...	0	
2	0	0	2.0	1532	2.0	0	0	0	0	0	...	0	
3	0	0	1.0	305	1.0	0	0	0	0	0	...	0	
4	0	0	2.0	1428	1.0	0	0	0	0	0	...	0	

5 rows × 141 columns

In [73]:

```
# Putting response variable to y
y = leads['Converted']

y.head()
```

Out[73]:

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

In [74]:

```
# Splitting the data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, test_size=0.3,
random_state=100)
```

Step 5: Feature Scaling

In [75]:

```
from sklearn.preprocessing import StandardScaler
```

In [76]:

```
scaler = StandardScaler()

X_train[['TotalVisits','Total Time Spent on Website','Page Views Per Visit']] =
scaler.fit_transform(X_train[['TotalVisits','Total Time Spent on Website','Page Views Per Visit']]
)

X_train.head()
```

Out[76]:

	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	...	Last Activity_Form Submitted on Website	Acti Conv
8529	0	0	0.969969	-0.864724	1.785283	0	0	0	0	0	...	0	
7331	0	0	0.102087	-0.215257	0.562949	0	0	0	0	0	...	0	
7688	0	0	0.102087	1.523992	0.562949	0	0	0	0	0	...	0	
92	0	0	0.536028	-0.686762	1.174116	0	0	0	0	0	...	0	
4908	0	0	-1.199737	-0.872062	-1.270553	0	0	0	0	0	...	0	

5 rows × 141 columns

In [77]:

```
X_train.describe()
```

Out[77]:

	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	Newspaper Article	X Education Forums	Newspaper	Digit Advertisemen
count	6002.000000	6002.0	6.002000e+03	6.002000e+03	6.002000e+03	6002.000000	6002.0	6002.0	6002.000000	6002.00000
mean	0.076308	0.0	1.047701e-16	6.392754e-17	2.308494e-17	0.001000	0.0	0.0	0.000167	0.00033
std	0.265512	0.0	1.000083e+00	1.000083e+00	1.000083e+00	0.031604	0.0	0.0	0.012908	0.01825
min	0.000000	0.0	-1.199737e+00	-8.720622e-01	-1.270553e+00	0.000000	0.0	0.0	0.000000	0.00000
25%	0.000000	0.0	-7.657957e-01	-8.683929e-01	-6.593854e-01	0.000000	0.0	0.0	0.000000	0.00000
50%	0.000000	0.0	1.020868e-01	-4.381673e-01	-4.821826e-02	0.000000	0.0	0.0	0.000000	0.00000
75%	0.000000	0.0	5.360281e-01	7.846274e-01	5.629489e-01	0.000000	0.0	0.0	0.000000	0.00000
max	1.000000	0.0	3.139676e+00	3.296264e+00	2.396450e+00	1.000000	0.0	0.0	1.000000	1.00000

8 rows × 141 columns

Checking the Lead Conversion Rate

In [78]:

```
### Checking the Lead Conversion Rate
converted = (sum(leads['Converted'])/len(leads['Converted'].index))*100
converted
```

Out[78]:

38.04081632653061

We have almost 38% lead conversion rate

Step 6: Model Building

Running Your First Training Model

In [79]:

```python
import statsmodels.api as sm
```

In [80]:

```python
# Logistic regression model
logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
logm1.fit().summary()
```

Out[80]:

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:	Tue, 12 May 2020	**Deviance:**	nan
Time:	13:37:35	**Pearson chi2:**	3.48e+18
No. Iterations:	100		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.415e+15	1.08e+08	-3.15e+07	0.000	-3.42e+15	-3.42e+15
Do Not Email	-4.083e+14	4.66e+06	-8.76e+07	0.000	-4.08e+14	-4.08e+14
Do Not Call	27.2194	1.44e-06	1.89e+07	0.000	27.219	27.219
TotalVisits	1.001e+14	1.51e+06	6.62e+07	0.000	1e+14	1e+14
Total Time Spent on Website	3.069e+14	1.07e+06	2.87e+08	0.000	3.07e+14	3.07e+14
Page Views Per Visit	-1.283e+14	1.64e+06	-7.83e+07	0.000	-1.28e+14	-1.28e+14
Search	4.048e+14	2.9e+07	1.39e+07	0.000	4.05e+14	4.05e+14
Newspaper Article	47.1124	1.35e-06	3.49e+07	0.000	47.112	47.112
X Education Forums	40.5135	1.13e-06	3.59e+07	0.000	40.513	40.513
Newspaper	-4.434e+15	6.76e+07	-6.55e+07	0.000	-4.43e+15	-4.43e+15
Digital Advertisement	3.389e+14	4.85e+07	6.98e+06	0.000	3.39e+14	3.39e+14
Through Recommendations	7.425e+14	5e+07	1.48e+07	0.000	7.43e+14	7.43e+14
A free copy of Mastering The Interview	-3.6e+13	2.94e+06	-1.23e+07	0.000	-3.6e+13	-3.6e+13
Country_Outside India	1.04e+14	4.99e+06	2.09e+07	0.000	1.04e+14	1.04e+14
Lead Source_Direct Traffic	1.561e+15	7.95e+07	1.96e+07	0.000	1.56e+15	1.56e+15
Lead Source_Facebook	6.292e+14	4.01e+07	1.57e+07	0.000	6.29e+14	6.29e+14
Lead Source_Google	1.606e+15	7.95e+07	2.02e+07	0.000	1.61e+15	1.61e+15
Lead Source_Live Chat	2.602e+15	6.31e+07	4.12e+07	0.000	2.6e+15	2.6e+15
Lead Source_NC_EDM	5.976e+15	1.04e+08	5.74e+07	0.000	5.98e+15	5.98e+15
Lead Source_Olark Chat	1.854e+15	7.94e+07	2.34e+07	0.000	1.85e+15	1.85e+15
Lead Source_Organic Search	1.527e+15	7.96e+07	1.92e+07	0.000	1.53e+15	1.53e+15
Lead Source_Pay per Click Ads	-1.37e+15	1.04e+08	-1.31e+07	0.000	-1.37e+15	-1.37e+15
Lead Source_Press_Release	-11.2505	7.86e-07	-1.43e+07	0.000	-11.250	-11.250
Lead Source_Reference	6.212e+14	4.15e+07	1.5e+07	0.000	6.21e+14	6.21e+14
Lead Source_Referral Sites	1.487e+15	7.99e+07	1.86e+07	0.000	1.49e+15	1.49e+15
Lead Source_Social Media	2.895e+15	1.06e+08	2.73e+07	0.000	2.89e+15	2.89e+15
Lead Source_WeLearn	14.0831	5.06e-07	2.78e+07	0.000	14.083	14.083
Lead Source_Welingak Website	2.199e+15	4.2e+07	5.23e+07	0.000	2.2e+15	2.2e+15
Lead Source_bing	-1.352e+15	9.28e+07	-1.46e+07	0.000	-1.35e+15	-1.35e+15
Lead Source_blog	-3.303e+15	1.04e+08	-3.17e+07	0.000	-3.3e+15	-3.3e+15

Lead Source_blog	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Lead Source_google	-2.875e+15	9.31e+07	-3.09e+07	0.000	-2.88e+15	-2.88e+15
Lead Source_testone	1.956e+15	1.04e+08	1.87e+07	0.000	1.96e+15	1.96e+15
Lead Source_welearnblog_Home	-3.107e+15	1.04e+08	-2.98e+07	0.000	-3.11e+15	-3.11e+15
Lead Source_youtubechannel	-1.7245	3.68e-07	-4.68e+06	0.000	-1.724	-1.724
Lead Origin_Landing Page Submission	-1.301e+13	4.28e+06	-3.04e+06	0.000	-1.3e+13	-1.3e+13
Lead Origin_Lead Add Form	1.186e+15	6.77e+07	1.75e+07	0.000	1.19e+15	1.19e+15
Lead Origin_Lead Import	6.292e+14	4.01e+07	1.57e+07	0.000	6.29e+14	6.29e+14
Last Notable Activity_Email Bounced	1.837e+15	7.42e+07	2.48e+07	0.000	1.84e+15	1.84e+15
Last Notable Activity_Email Link Clicked	6.687e+14	7.39e+07	9.05e+06	0.000	6.69e+14	6.69e+14
Last Notable Activity_Email Marked Spam	2.078e+15	4.39e+07	4.74e+07	0.000	2.08e+15	2.08e+15
Last Notable Activity_Email Opened	1.697e+15	7.32e+07	2.32e+07	0.000	1.7e+15	1.7e+15
Last Notable Activity_Email Received	2.305e+15	1.2e+08	1.92e+07	0.000	2.3e+15	2.3e+15
Last Notable Activity_Form Submitted on Website	-2.768e+15	9.98e+07	-2.77e+07	0.000	-2.77e+15	-2.77e+15
Last Notable Activity_Had a Phone Conversation	1.372e+15	8.18e+07	1.68e+07	0.000	1.37e+15	1.37e+15
Last Notable Activity_Modified	1.22e+15	7.31e+07	1.67e+07	0.000	1.22e+15	1.22e+15
Last Notable Activity_Olark Chat Conversation	9.611e+14	7.35e+07	1.31e+07	0.000	9.61e+14	9.61e+14
Last Notable Activity_Page Visited on Website	1.507e+15	7.35e+07	2.05e+07	0.000	1.51e+15	1.51e+15
Last Notable Activity_Resubscribed to emails	1.1245	1.24e-06	9.09e+05	0.000	1.124	1.124
Last Notable Activity_SMS Sent	2.032e+15	7.32e+07	2.77e+07	0.000	2.03e+15	2.03e+15
Last Notable Activity_Unreachable	1.097e+15	7.54e+07	1.45e+07	0.000	1.1e+15	1.1e+15
Last Notable Activity_Unsubscribed	1.785e+15	7.68e+07	2.32e+07	0.000	1.79e+15	1.79e+15
Last Notable Activity_View in browser link Clicked	1.136e+15	1.2e+08	9.43e+06	0.000	1.14e+15	1.14e+15
Lead Quality_High in Relevance	-9.861e+13	5.63e+06	-1.75e+07	0.000	-9.86e+13	-9.86e+13
Lead Quality_Low in Relevance	-1.369e+14	5.45e+06	-2.51e+07	0.000	-1.37e+14	-1.37e+14
Lead Quality_Might be	-1.809e+14	4.06e+06	-4.46e+07	0.000	-1.81e+14	-1.81e+14
Lead Quality_Not Sure	1.411e+14	3.68e+06	3.83e+07	0.000	1.41e+14	1.41e+14
Lead Quality_Worst	-3.773e+14	5.57e+06	-6.77e+07	0.000	-3.77e+14	-3.77e+14
Asymmetrique Profile Index_01.High	-1.975e+14	3.86e+06	-5.12e+07	0.000	-1.97e+14	-1.97e+14
Asymmetrique Profile Index_02.Medium	4.509e+13	3.34e+06	1.35e+07	0.000	4.51e+13	4.51e+13
Asymmetrique Profile Index_03.Low	-2.621e+14	1.44e+07	-1.82e+07	0.000	-2.62e+14	-2.62e+14
Asymmetrique Activity Index_01.High	1.058e+14	4.13e+06	2.56e+07	0.000	1.06e+14	1.06e+14
Asymmetrique Activity Index_02.Medium	9.429e+13	3.34e+06	2.82e+07	0.000	9.43e+13	9.43e+13
Asymmetrique Activity Index_03.Low	-6.145e+14	5.07e+06	-1.21e+08	0.000	-6.15e+14	-6.15e+14
Tags_Already a student	-9.694e+13	6.49e+06	-1.49e+07	0.000	-9.69e+13	-9.69e+13
Tags_Busy	-8.193e+14	7.61e+06	-1.08e+08	0.000	-8.19e+14	-8.19e+14
Tags_Closed by Horizzon	1.535e+15	7.01e+06	2.19e+08	0.000	1.53e+15	1.53e+15
Tags_Diploma holder (Not Eligible)	-3.443e+15	1.11e+07	-3.11e+08	0.000	-3.44e+15	-3.44e+15
Tags_Graduation in progress	5.546e+14	9.08e+06	6.11e+07	0.000	5.55e+14	5.55e+14
Tags_In confusion whether part time or DLP	8.335e+14	3.04e+07	2.74e+07	0.000	8.33e+14	8.33e+14
Tags_Interested in full time MBA	8.53e+13	8.87e+06	9.62e+06	0.000	8.53e+13	8.53e+13
Tags_Interested in Next batch	4.038e+15	3.92e+07	1.03e+08	0.000	4.04e+15	4.04e+15
Tags_Interested in other courses	1.492e+14	5.13e+06	2.91e+07	0.000	1.49e+14	1.49e+14
Tags_Lateral student	4.871e+15	4.79e+07	1.02e+08	0.000	4.87e+15	4.87e+15
Tags_Lost to EINS	1.764e+15	7.42e+06	2.38e+08	0.000	1.76e+15	1.76e+15
Tags_Lost to Others	2.015e+14	3.08e+07	6.55e+06	0.000	2.01e+14	2.01e+14
Tags_Not doing further education	9.149e+13	8.38e+06	1.09e+07	0.000	9.15e+13	9.15e+13
Tags_Recognition issue (DEC approval)	-3.995e+15	6.89e+07	-5.79e+07	0.000	-3.99e+15	-3.99e+15
Tags_Ringing	-5.595e+14	4.4e+06	-1.27e+08	0.000	-5.6e+14	-5.6e+14
Tags_Shall take in the next coming month	4.963e+15	6.78e+07	7.32e+07	0.000	4.96e+15	4.96e+15
Tags_Still Thinking	3.827e+14	3.42e+07	1.12e+07	0.000	3.83e+14	3.83e+14

Tags_University not recognized	-3.339e+15	4.79e+07	-6.97e+07	0.000	-3.34e+15	-3.34e+15
Tags_Want to take admission but has financial problems	6.589e+14	4.15e+07	1.59e+07	0.000	6.59e+14	6.59e+14
Tags_Will revert after reading the email	1.177e+15	5.07e+06	2.32e+08	0.000	1.18e+15	1.18e+15
Tags_in touch with EINS	1.033e+15	2.42e+07	4.27e+07	0.000	1.03e+15	1.03e+15
Tags_invalid number	-3.133e+15	9.98e+06	-3.14e+08	0.000	-3.13e+15	-3.13e+15
Tags_number not provided	-2.948e+15	1.66e+07	-1.78e+08	0.000	-2.95e+15	-2.95e+15
Tags_opp hangup	-5.167e+14	1.62e+07	-3.2e+07	0.000	-5.17e+14	-5.17e+14
Tags_switched off	-7.601e+14	6.61e+06	-1.15e+08	0.000	-7.6e+14	-7.6e+14
Tags_wrong number given	-1.428e+15	1.27e+07	-1.12e+08	0.000	-1.43e+15	-1.43e+15
Lead Profile_Dual Specialization Student	2.325e+14	2.16e+07	1.08e+07	0.000	2.33e+14	2.33e+14
Lead Profile_Lateral Student	1.482e+15	1.79e+07	8.29e+07	0.000	1.48e+15	1.48e+15
Lead Profile_Other Leads	4.112e+14	4.7e+06	8.74e+07	0.000	4.11e+14	4.11e+14
Lead Profile_Potential Lead	3.739e+14	3.28e+06	1.14e+08	0.000	3.74e+14	3.74e+14
Lead Profile_Student of SomeSchool	-1.21e+14	8.03e+06	-1.51e+07	0.000	-1.21e+14	-1.21e+14
What is your current occupation_Businessman	-1.1e+15	4.82e+07	-2.28e+07	0.000	-1.1e+15	-1.1e+15
What is your current occupation_Housewife	5.129e+14	2.45e+07	2.09e+07	0.000	5.13e+14	5.13e+14
What is your current occupation_Other	-1.056e+15	1.95e+07	-5.42e+07	0.000	-1.06e+15	-1.06e+15
What is your current occupation_Student	-7.487e+14	7.46e+06	-1e+08	0.000	-7.49e+14	-7.49e+14
What is your current occupation_Unemployed	-7.881e+14	4.32e+06	-1.82e+08	0.000	-7.88e+14	-7.88e+14
What is your current occupation_Working Professional	-6.604e+14	5.71e+06	-1.16e+08	0.000	-6.6e+14	-6.6e+14
Specialization_Banking, Investment And Insurance	2.857e+14	6.78e+06	4.22e+07	0.000	2.86e+14	2.86e+14
Specialization_Business Administration	2.646e+14	6.5e+06	4.07e+07	0.000	2.65e+14	2.65e+14
Specialization_E-Business	4.691e+14	1.29e+07	3.63e+07	0.000	4.69e+14	4.69e+14
Specialization_E-COMMERCE	-7.826e+12	9.61e+06	-8.14e+05	0.000	-7.83e+12	-7.83e+12
Specialization_Finance Management	1.622e+14	5.75e+06	2.82e+07	0.000	1.62e+14	1.62e+14
Specialization_Healthcare Management	1.491e+14	8.91e+06	1.67e+07	0.000	1.49e+14	1.49e+14
Specialization_Hospitality Management	1.895e+14	9.42e+06	2.01e+07	0.000	1.9e+14	1.9e+14
Specialization_Human Resource Management	1.723e+14	5.74e+06	3e+07	0.000	1.72e+14	1.72e+14
Specialization_IT Projects Management	1.061e+14	6.98e+06	1.52e+07	0.000	1.06e+14	1.06e+14
Specialization_International Business	9.411e+13	8.12e+06	1.16e+07	0.000	9.41e+13	9.41e+13
Specialization_Marketing Management	3.037e+14	5.67e+06	5.35e+07	0.000	3.04e+14	3.04e+14
Specialization_Media and Advertising	1.365e+14	7.95e+06	1.72e+07	0.000	1.37e+14	1.37e+14
Specialization_Operations Management	2.337e+14	6.22e+06	3.76e+07	0.000	2.34e+14	2.34e+14
Specialization_Retail Management	9.467e+13	1.02e+07	9.32e+06	0.000	9.47e+13	9.47e+13
Specialization_Rural and Agribusiness	-4.408e+13	1.12e+07	-3.92e+06	0.000	-4.41e+13	-4.41e+13
Specialization_Select	1.498e+14	4.18e+06	3.58e+07	0.000	1.5e+14	1.5e+14
Specialization_Services Excellence	1.708e+14	1.66e+07	1.03e+07	0.000	1.71e+14	1.71e+14
Specialization_Supply Chain Management	4.478e+13	6.74e+06	6.64e+06	0.000	4.48e+13	4.48e+13
Specialization_Travel and Tourism	-1.048e+14	8.3e+06	-1.26e+07	0.000	-1.05e+14	-1.05e+14
City_Mumbai	-7.043e+13	4.61e+06	-1.53e+07	0.000	-7.04e+13	-7.04e+13
City_Other Cities	-8.994e+13	5.4e+06	-1.66e+07	0.000	-8.99e+13	-8.99e+13
City_Other Cities of Maharashtra	-1.554e+13	5.87e+06	-2.65e+06	0.000	-1.55e+13	-1.55e+13
City_Other Metro Cities	-3.015e+14	6.29e+06	-4.79e+07	0.000	-3.02e+14	-3.02e+14
City_Thane & Outskirts	-1.033e+14	5.26e+06	-1.96e+07	0.000	-1.03e+14	-1.03e+14
City_Tier II Cities	2.696e+14	1.1e+07	2.46e+07	0.000	2.7e+14	2.7e+14
Last Activity_Approached upfront	5.231e+15	2.92e+07	1.79e+08	0.000	5.23e+15	5.23e+15
Last Activity_Converted to Lead	-7.838e+13	1.06e+07	-7.4e+06	0.000	-7.84e+13	-7.84e+13
Last Activity_Email Bounced	3.338e+13	1.17e+07	2.84e+06	0.000	3.34e+13	3.34e+13
Last Activity_Email Link Clicked	7.522e+14	1.25e+07	6.03e+07	0.000	7.52e+14	7.52e+14
Last Activity_Email Marked Spam	2.078e+15	4.39e+07	4.74e+07	0.000	2.08e+15	2.08e+15

Last Activity_Email Opened	1.106e+14	9.92e+06	1.11e+07	0.000	1.11e+14	1.11e+14
Last Activity_Email Received	4.186e+15	6.8e+07	6.16e+07	0.000	4.19e+15	4.19e+15
Last Activity_Form Submitted on Website	2.751e+14	1.21e+07	2.27e+07	0.000	2.75e+14	2.75e+14
Last Activity_Had a Phone Conversation	6.294e+14	2.31e+07	2.72e+07	0.000	6.29e+14	6.29e+14
Last Activity_Olark Chat Conversation	-8.611e+13	1.01e+07	-8.55e+06	0.000	-8.61e+13	-8.61e+13
Last Activity_Page Visited on Website	4.133e+13	1.06e+07	3.91e+06	0.000	4.13e+13	4.13e+13
Last Activity_Resubscribed to emails	0	0	nan	nan	0	0
Last Activity_SMS Sent	6.132e+14	1e+07	6.12e+07	0.000	6.13e+14	6.13e+14
Last Activity_Unreachable	3.205e+14	1.43e+07	2.24e+07	0.000	3.2e+14	3.2e+14
Last Activity_Unsubscribed	7.866e+14	2.28e+07	3.44e+07	0.000	7.87e+14	7.87e+14
Last Activity_View in browser link Clicked	-2.8e+15	6.81e+07	-4.11e+07	0.000	-2.8e+15	-2.8e+15
Last Activity_Visited Booth in Tradeshow	-7.005e+14	6.9e+07	-1.01e+07	0.000	-7.01e+14	-7.01e+14

Step 7: Feature Selection Using RFE

In [81]:

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

In [82]:

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

In [83]:

```
rfe.support_
```

Out[83]:

```
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,  True,
       False,  True,  True, False, False,  True, False,  True, False,
        True, False,  True, False,  True, False, False, False, False,
        True, False,  True,  True,  True,  True,  True, False, False,
       False, False, False, False, False, False, False,  True,  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, False, False, False, False,
       False,  True, False, False, False, False])
```

In [85]:

```
list(zip(X_train.columns, rfe.support_, rfe.ranking_))
```

Out[85]:

```
[('Do Not Email', False, 8),
 ('Do Not Call', False, 122),
 ('TotalVisits', False, 73),
 ('Total Time Spent on Website', False, 12),
 ('Page Views Per Visit', False, 55),
 ('Search', False, 24),
 ('Newspaper Article', False, 116),
 ('X Education Forums', False, 115),
 ('Newspaper', False, 91),
```

```
('Digital Advertisement', False, 89),
('Through Recommendations', False, 102),
('A free copy of Mastering The Interview', False, 90),
('Country_Outside India', False, 78),
('Lead Source_Direct Traffic', False, 53),
('Lead Source_Facebook', False, 54),
('Lead Source_Google', False, 94),
('Lead Source_Live Chat', False, 110),
('Lead Source_NC_EDM', False, 16),
('Lead Source_Olark Chat', False, 11),
('Lead Source_Organic Search', False, 70),
('Lead Source_Pay per Click Ads', False, 111),
('Lead Source_Press_Release', False, 120),
('Lead Source_Reference', False, 35),
('Lead Source_Referral Sites', False, 63),
('Lead Source_Social Media', False, 114),
('Lead Source_WeLearn', False, 119),
('Lead Source_Welingak Website', True, 1),
('Lead Source_bing', False, 99),
('Lead Source_blog', False, 76),
('Lead Source_google', False, 75),
('Lead Source_testone', False, 108),
('Lead Source_welearnblog_Home', False, 80),
('Lead Source_youtubechannel', False, 117),
('Lead Origin_Landing Page Submission', False, 72),
('Lead Origin_Lead Add Form', False, 10),
('Lead Origin_Lead Import', False, 52),
('Last Notable Activity_Email Bounced', False, 33),
('Last Notable Activity_Email Link Clicked', False, 19),
('Last Notable Activity_Email Marked Spam', False, 83),
('Last Notable Activity_Email Opened', False, 88),
('Last Notable Activity_Email Received', False, 106),
('Last Notable Activity_Form Submitted on Website', False, 87),
('Last Notable Activity_Had a Phone Conversation', False, 40),
('Last Notable Activity_Modified', False, 2),
('Last Notable Activity_Olark Chat Conversation', False, 6),
('Last Notable Activity_Page Visited on Website', False, 98),
('Last Notable Activity_Resubscribed to emails', False, 121),
('Last Notable Activity_SMS Sent', False, 14),
('Last Notable Activity_Unreachable', False, 74),
('Last Notable Activity_Unsubscribed', False, 32),
('Last Notable Activity_View in browser link Clicked', False, 105),
('Lead Quality_High in Relevance', False, 29),
('Lead Quality_Low in Relevance', False, 82),
('Lead Quality_Might be', False, 38),
('Lead Quality_Not Sure', False, 50),
('Lead Quality_Worst', True, 1),
('Asymmetrique Profile Index_01.High', False, 64),
('Asymmetrique Profile Index_02.Medium', False, 85),
('Asymmetrique Profile Index_03.Low', False, 84),
('Asymmetrique Activity Index_01.High', False, 65),
('Asymmetrique Activity Index_02.Medium', False, 66),
('Asymmetrique Activity Index_03.Low', True, 1),
('Tags_Already a student', True, 1),
('Tags_Busy', False, 27),
('Tags_Closed by Horizzon', True, 1),
('Tags_Diploma holder (Not Eligible)', True, 1),
('Tags_Graduation in progress', False, 4),
('Tags_In confusion whether part time or DLP', False, 37),
('Tags_Interested  in full time MBA', True, 1),
('Tags_Interested in Next batch', False, 48),
('Tags_Interested in other courses', True, 1),
('Tags_Lateral student', False, 34),
('Tags_Lost to EINS', True, 1),
('Tags_Lost to Others', False, 36),
('Tags_Not doing further education', True, 1),
('Tags_Recognition issue (DEC approval)', False, 30),
('Tags_Ringing', True, 1),
('Tags_Shall take in the next coming month', False, 45),
('Tags_Still Thinking', False, 9),
('Tags_University not recognized', False, 39),
('Tags_Want to take admission but has financial problems', False, 28),
('Tags_Will revert after reading the email', True, 1),
('Tags_in touch with EINS', False, 49),
('Tags_invalid number', True, 1),
('Tags_number not provided', True, 1),
('Tags_opp hangup', True, 1),
```

```
('Tags_switched off', True, 1),
('Tags_wrong number given', True, 1),
('Lead Profile_Dual Specialization Student', False, 51),
('Lead Profile_Lateral Student', False, 13),
('Lead Profile_Other Leads', False, 18),
('Lead Profile_Potential Lead', False, 17),
('Lead Profile_Student of SomeSchool', False, 46),
('What is your current occupation_Businessman', False, 92),
('What is your current occupation_Housewife', False, 20),
('What is your current occupation_Other', False, 25),
('What is your current occupation_Student', False, 3),
('What is your current occupation_Unemployed', True, 1),
('What is your current occupation_Working Professional', True, 1),
('Specialization_Banking, Investment And Insurance', False, 62),
('Specialization_Business Administration', False, 59),
('Specialization_E-Business', False, 47),
('Specialization_E-COMMERCE', False, 81),
('Specialization_Finance Management', False, 68),
('Specialization_Healthcare Management', False, 101),
('Specialization_Hospitality Management', False, 58),
('Specialization_Human Resource Management', False, 67),
('Specialization_IT Projects Management', False, 104),
('Specialization_International Business', False, 109),
('Specialization_Marketing Management', False, 56),
('Specialization_Media and Advertising', False, 107),
('Specialization_Operations Management', False, 57),
('Specialization_Retail Management', False, 71),
('Specialization_Rural and Agribusiness', False, 95),
('Specialization_Select', False, 15),
('Specialization_Services Excellence', False, 60),
('Specialization_Supply Chain Management', False, 100),
('Specialization_Travel and Tourism', False, 23),
('City_Mumbai', False, 97),
('City_Other Cities', False, 77),
('City_Other Cities of Maharashtra', False, 69),
('City_Other Metro Cities', False, 44),
('City_Thane & Outskirts', False, 96),
('City_Tier II Cities', False, 21),
('Last Activity_Approached upfront', False, 61),
('Last Activity_Converted to Lead', False, 41),
('Last Activity_Email Bounced', False, 42),
('Last Activity_Email Link Clicked', False, 31),
('Last Activity_Email Marked Spam', False, 79),
('Last Activity_Email Opened', False, 113),
('Last Activity_Email Received', False, 93),
('Last Activity_Form Submitted on Website', False, 26),
('Last Activity_Had a Phone Conversation', False, 5),
('Last Activity_Olark Chat Conversation', False, 22),
('Last Activity_Page Visited on Website', False, 43),
('Last Activity_Resubscribed to emails', False, 118),
('Last Activity_SMS Sent', True, 1),
('Last Activity_Unreachable', False, 86),
('Last Activity_Unsubscribed', False, 7),
('Last Activity_View in browser link Clicked', False, 103),
('Last Activity_Visited Booth in Tradeshow', False, 112)]
```

In [86]:

```
col = X_train.columns[rfe.support_]
col
```

Out[86]:

```
Index(['Lead Source_Welingak Website', 'Lead Quality_Worst',
       'Asymmetrique Activity Index_03.Low', 'Tags_Already a student',
       'Tags_Closed by Horizzon', 'Tags_Diploma holder (Not Eligible)',
       'Tags_Interested  in full time MBA', 'Tags_Interested in other courses',
       'Tags_Lost to EINS', 'Tags_Not doing further education', 'Tags_Ringing',
       'Tags_Will revert after reading the email', 'Tags_invalid number',
       'Tags_number not provided', 'Tags_opp hangup', 'Tags_switched off',
       'Tags_wrong number given', 'What is your current occupation_Unemployed',
       'What is your current occupation_Working Professional',
       'Last Activity_SMS Sent'],
      dtype='object')
```

In [87]:

```
X_train.columns[~rfe.support_]
```

Out[87]:

```
Index(['Do Not Email', 'Do Not Call', 'TotalVisits',
       'Total Time Spent on Website', 'Page Views Per Visit', 'Search',
       'Newspaper Article', 'X Education Forums', 'Newspaper',
       'Digital Advertisement',
       ...
       'Last Activity_Email Received',
       'Last Activity_Form Submitted on Website',
       'Last Activity_Had a Phone Conversation',
       'Last Activity_Olark Chat Conversation',
       'Last Activity_Page Visited on Website',
       'Last Activity_Resubscribed to emails', 'Last Activity_Unreachable',
       'Last Activity_Unsubscribed',
       'Last Activity_View in browser link Clicked',
       'Last Activity_Visited Booth in Tradeshow'],
      dtype='object', length=121)
```

Assessing the model with StatsModels

In [88]:

```
X_train_sm = sm.add_constant(X_train[col])
logm2 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = logm2.fit()
res.summary()
```

Out[88]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5981
Model Family:	Binomial	**Df Model:**	20
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-1264.7
Date:	Tue, 12 May 2020	**Deviance:**	2529.4
Time:	13:38:21	**Pearson chi2:**	8.56e+03
No. Iterations:	24		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.4929	0.090	-27.836	0.000	-2.668	-2.317
Lead Source_Welingak Website	3.2281	0.731	4.414	0.000	1.795	4.662
Lead Quality_Worst	-2.5504	0.761	-3.354	0.001	-4.041	-1.060
Asymmetrique Activity Index_03.Low	-2.4592	0.358	-6.869	0.000	-3.161	-1.758
Tags_Already a student	-3.8785	0.726	-5.344	0.000	-5.301	-2.456
Tags_Closed by Horizzon	5.1421	0.722	7.120	0.000	3.727	6.558
Tags_Diploma holder (Not Eligible)	-24.1871	2.82e+04	-0.001	0.999	-5.52e+04	5.52e+04
Tags_Interested in full time MBA	-3.0545	0.742	-4.117	0.000	-4.509	-1.600
Tags_Interested in other courses	-3.0288	0.330	-9.183	0.000	-3.675	-2.382
Tags_Lost to EINS	6.3792	0.831	7.677	0.000	4.751	8.008
Tags_Not doing further education	-3.7904	1.032	-3.674	0.000	-5.813	-1.768
Tags_Ringing	-4.2659	0.249	-17.107	0.000	-4.755	-3.777
Tags_Will revert after reading the email	3.5963	0.194	18.561	0.000	3.217	3.976
Tags_invalid number	-25.7192	2.7e+04	-0.001	0.999	-5.3e+04	5.29e+04

Tags_number not provided	-25.9733	4.5e+04	-0.001	1.000	-8.82e+04	8.82e+04
Tags_opp hangup	-3.5152	1.063	-3.308	0.001	-5.598	-1.433
Tags_switched off	-5.1620	0.724	-7.126	0.000	-6.582	-3.742
Tags_wrong number given	-26.1206	3.49e+04	-0.001	0.999	-6.84e+04	6.84e+04
What is your current occupation_Unemployed	2.0649	0.119	17.357	0.000	1.832	2.298
What is your current occupation_Working Professional	2.1458	0.364	5.903	0.000	1.433	2.858
Last Activity_SMS Sent	2.0390	0.112	18.174	0.000	1.819	2.259

In [89]:

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

Out[89]:

```
8529    0.065692
7331    0.009069
7688    0.833555
92      0.076360
4908    0.076360
451     0.009069
4945    0.009069
2844    0.994975
4355    0.076360
7251    0.001051
dtype: float64
```

In [90]:

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

Out[90]:

```
array([0.06569164, 0.00906869, 0.83355546, 0.07635965, 0.07635965,
       0.00906869, 0.00906869, 0.99497496, 0.07635965, 0.00105118])
```

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

In [91]:

```
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()
```

Out[91]:

	Converted	Conversion_Prob	LeadID
0	0	0.065692	8529
1	0	0.009069	7331
2	1	0.833555	7688
3	0	0.076360	92
4	0	0.076360	4908

Creating new column 'predicted' with 1 if Churn_Prob > 0.5 else 0

In [92]:

```
y_train_pred_final['predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 els
e 0)
```

```
# Let's see the head
y_train_pred_final.head()
```

Out[92]:

	Converted	Conversion_Prob	LeadID	predicted
0	0	0.065692	8529	0
1	0	0.009069	7331	0
2	1	0.833555	7688	1
3	0	0.076360	92	0
4	0	0.076360	4908	0

In [93]:

```
from sklearn import metrics
```

Creating Confusion Metrics

In [94]:

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.predicted )
print(confusion)
```

```
[[3647   89]
 [ 409 1857]]
```

In [95]:

```
# Let's check the overall accuracy.
print(metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.predicted))
```

```
0.9170276574475175
```

Checking VIFs

In [96]:

```
# Check for the VIF values of the feature variables.
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [97]:

```
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].values, i) for i in range(X_train[col].shape[1
])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[97]:

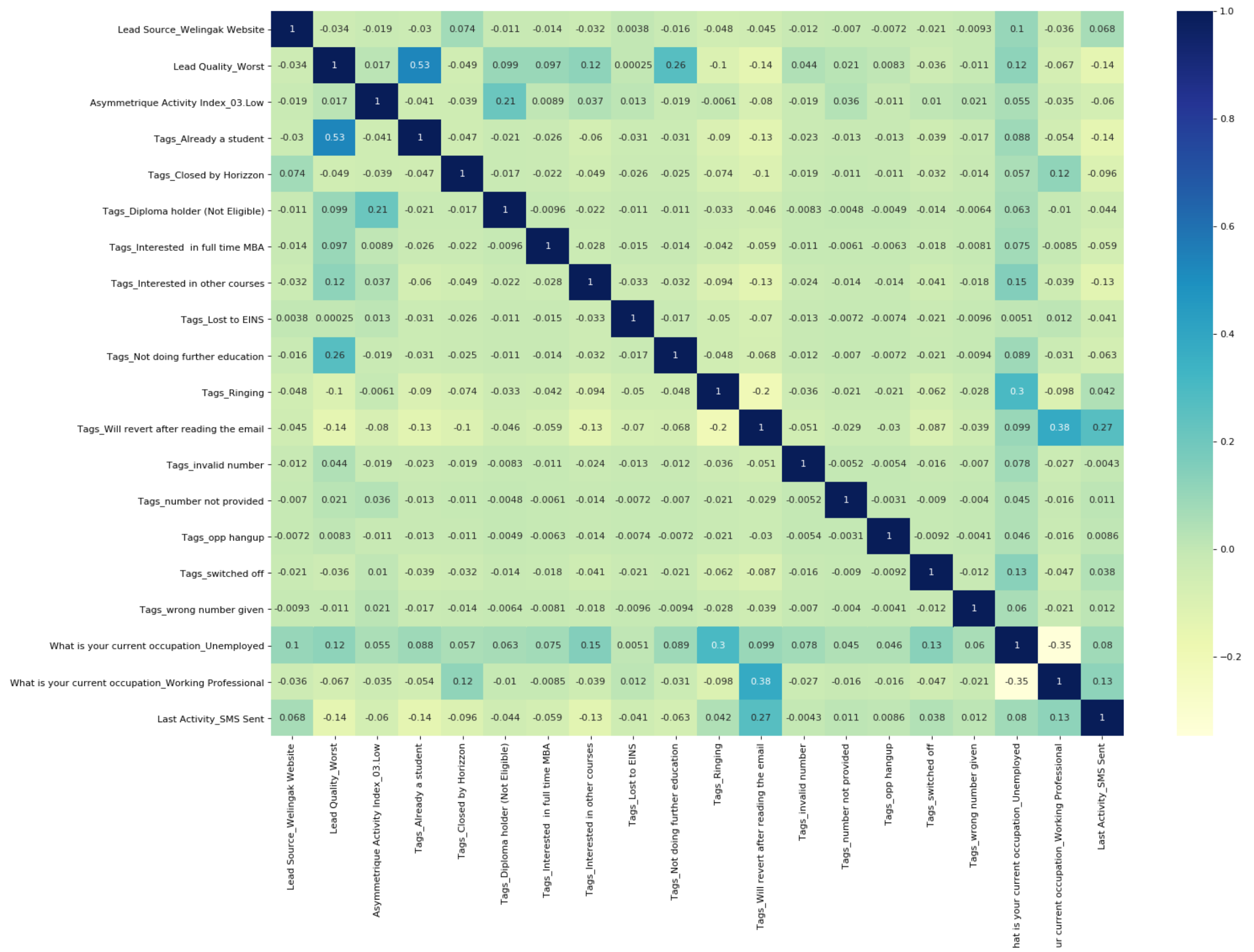
	Features	VIF
4	Tags_Closed by Horizzon	1.30
9	Tags_Not doing further education	1.27
15	Tags_switched off	1.20
5	Tags_Diploma holder (Not Eligible)	1.12
6	Tags_Interested in full time MBA	1.12

	Features	VIF
2	Asymmetrique Activity Index_03.Low	1.11
0	Lead Source_Welingak Website	1.09
12	Tags_invalid number	1.08
8	Tags_Lost to EINS	1.07
16	Tags_wrong number given	1.04
14	Tags_opp hangup	1.03
13	Tags_number not provided	1.03
18	What is your current occupation_Working Profes...	0.80
1	Lead Quality_Worst	0.69
10	Tags_Ringing	0.62
7	Tags_Interested in other courses	0.40
3	Tags_Already a student	0.38
11	Tags_Will revert after reading the email	0.09
17	What is your current occupation_Unemployed	0.01
19	Last Activity_SMS Sent	0.00

Clearly there is not much multicollinearity present in our model among the selected features as per their VIF values.

In [98]:

```
# Slightly alter the figure size to make it more horizontal.
plt.figure(figsize=(20,15), dpi=80, facecolor='w', edgecolor='k', frameon='True')

cor = X_train[col].corr()
sns.heatmap(cor, annot=True, cmap="YlGnBu")

plt.tight_layout()
plt.show()
```



Dropping the Variable and Updating the Model

In [99]:

```
col = col.drop('Tags_number not provided', 1)
col
```

Out[99]:

```
Index(['Lead Source_Welingak Website', 'Lead Quality_Worst',
       'Asymmetrique Activity Index_03.Low', 'Tags_Already a student',
       'Tags_Closed by Horizzon', 'Tags_Diploma holder (Not Eligible)',
       'Tags_Interested  in full time MBA', 'Tags_Interested in other courses',
       'Tags_Lost to EINS', 'Tags_Not doing further education', 'Tags_Ringing',
       'Tags_Will revert after reading the email', 'Tags_invalid number',
       'Tags_opp hangup', 'Tags_switched off', 'Tags_wrong number given',
       'What is your current occupation_Unemployed',
       'What is your current occupation_Working Professional',
       'Last Activity_SMS Sent'],
      dtype='object')
```

In [100]:

```
X_train_sm = sm.add_constant(X_train[col])
logm3 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = logm3.fit()
res.summary()
```

Out[100]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5982
Model Family:	Binomial	**Df Model:**	19
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-1278.7
Date:	Tue, 12 May 2020	**Deviance:**	2557.4
Time:	13:38:42	**Pearson chi2:**	8.49e+03
No. Iterations:	24		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.4804	0.089	-27.881	0.000	-2.655	-2.306
Lead Source_Welingak Website	3.2918	0.731	4.503	0.000	1.859	4.725
Lead Quality_Worst	-2.7112	0.739	-3.668	0.000	-4.160	-1.263
Asymmetrique Activity Index_03.Low	-2.4342	0.357	-6.817	0.000	-3.134	-1.734
Tags_Already a student	-3.8015	0.724	-5.247	0.000	-5.221	-2.382
Tags_Closed by Horizzon	5.1851	0.722	7.184	0.000	3.770	6.600
Tags_Diploma holder (Not Eligible)	-24.1120	2.81e+04	-0.001	0.999	-5.51e+04	5.51e+04
Tags_Interested in full time MBA	-2.9855	0.741	-4.028	0.000	-4.438	-1.533
Tags_Interested in other courses	-2.9603	0.329	-8.996	0.000	-3.605	-2.315
Tags_Lost to EINS	6.4382	0.838	7.684	0.000	4.796	8.080
Tags_Not doing further education	-3.7070	1.031	-3.596	0.000	-5.727	-1.687
Tags_Ringing	-4.1829	0.248	-16.855	0.000	-4.669	-3.696
Tags_Will revert after reading the email	3.6368	0.193	18.834	0.000	3.258	4.015
Tags_invalid number	-25.6348	2.7e+04	-0.001	0.999	-5.3e+04	5.29e+04

Tags_opp hangup	-3.4305	1.062	-3.231	0.001	-5.512	-1.349
Tags_switched off	-5.0770	0.724	-7.013	0.000	-6.496	-3.658
Tags_wrong number given	-26.0375	3.49e+04	-0.001	0.999	-6.85e+04	6.84e+04
What is your current occupation_Unemployed	1.9949	0.118	16.969	0.000	1.764	2.225
What is your current occupation_Working Professional	2.1030	0.363	5.788	0.000	1.391	2.815
Last Activity_SMS Sent	2.0063	0.111	18.069	0.000	1.789	2.224

In [101]:

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

Out[101]:

```
8529    0.065249
7331    0.009300
7688    0.820658
92      0.077242
4908    0.077242
451     0.009300
4945    0.009300
2844    0.994861
4355    0.077242
7251    0.000913
dtype: float64
```

In [102]:

```
y_train_pred = y_train_pred.values.reshape(-1)
y_train_pred[:10]
```

Out[102]:

```
array([6.52492255e-02, 9.29987842e-03, 8.20658174e-01, 7.72422324e-02,
       7.72422324e-02, 9.29987842e-03, 9.29987842e-03, 9.94861183e-01,
       7.72422324e-02, 9.12704851e-04])
```

In [103]:

```
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()
```

Out[103]:

	Converted	Conversion_Prob	LeadID
0	0	0.065249	8529
1	0	0.009300	7331
2	1	0.820658	7688
3	0	0.077242	92
4	0	0.077242	4908

In [104]:

```
y_train_pred_final['predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 els
e 0)

# Let's see the head
y_train_pred_final.head()
```

Out[104]:

	Converted	Conversion_Prob	LeadID	predicted

	Converted	Conversion_Prob	LeadID	predicted
0	0	0.065249	8529	0
1	0	0.009300	7331	0
2	1	0.820658	7688	1
3	0	0.077242	92	0
4	0	0.077242	4908	0

In [105]:

```
from sklearn import metrics
```

In [106]:

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.predicted )
print(confusion)
```

```
[[3641   95]
 [ 409 1857]]
```

In [107]:

```
# checking accuracy
print(metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.predicted))
```

```
0.9160279906697767
```

In [108]:

```
#checking VIFS
```

In [109]:

```
# Check for the VIF values of the feature variables.
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [110]:

```
# Check for the VIF values of the feature variables.
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [111]:

```
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].values, i) for i in range(X_train[col].shape[1
])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[111]:

	Features	VIF
4	Tags_Closed by Horizzon	1.29
9	Tags_Not doing further education	1.27
14	Tags_switched off	1.19
6	Tags_Interested in full time MBA	1.12
5	Tags_Diploma holder (Not Eligible)	1.12
2	Asymmetrique Activity Index_03.Low	1.11
0	Lead Source_Welingak Website	1.09

	Features	VIF
12	Tags_invalid number	1.08
8	Tags_Lost to EINS	1.07
15	Tags_wrong number given	1.04
13	Tags_opp hangup	1.03
17	What is your current occupation_Working Profes...	0.79
1	Lead Quality_Worst	0.69
10	Tags_Ringing	0.62
7	Tags_Interested in other courses	0.39
3	Tags_Already a student	0.38
11	Tags_Will revert after reading the email	0.09
16	What is your current occupation_Unemployed	0.01
18	Last Activity_SMS Sent	0.00

Dropping the Variable and Updating the Model

In [112]:

```
col = col.drop('Tags_wrong number given', 1)
col
```

Out[112]:

```
Index(['Lead Source_Welingak Website', 'Lead Quality_Worst',
       'Asymmetrique Activity Index_03.Low', 'Tags_Already a student',
       'Tags_Closed by Horizzon', 'Tags_Diploma holder (Not Eligible)',
       'Tags_Interested  in full time MBA', 'Tags_Interested in other courses',
       'Tags_Lost to EINS', 'Tags_Not doing further education', 'Tags_Ringing',
       'Tags_Will revert after reading the email', 'Tags_invalid number',
       'Tags_opp hangup', 'Tags_switched off',
       'What is your current occupation_Unemployed',
       'What is your current occupation_Working Professional',
       'Last Activity_SMS Sent'],
      dtype='object')
```

In [113]:

```
X_train_sm = sm.add_constant(X_train[col])
logm4 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = logm4.fit()
res.summary()
```

Out[113]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5983
Model Family:	Binomial	**Df Model:**	18
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-1305.1
Date:	Tue, 12 May 2020	**Deviance:**	2610.1
Time:	13:38:55	**Pearson chi2:**	8.25e+03
No. Iterations:	23		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.4653	0.088	-27.969	0.000	-2.638	-2.293
Lead Source_Welingak Website	3.4161	0.731	4.676	0.000	1.984	4.848
Lead Quality_Worst	-2.7568	0.728	-3.787	0.000	-4.184	-1.330

Asymmetrique Activity Index_03.Low	-2.3688	0.357	-6.637	0.000	-3.068	-1.669
Tags_Already a student	-3.6760	0.724	-5.080	0.000	-5.094	-2.258
Tags_Closed by Horizzon	5.2742	0.721	7.314	0.000	3.861	6.687
Tags_Diploma holder (Not Eligible)	-22.9881	1.71e+04	-0.001	0.999	-3.35e+04	3.35e+04
Tags_Interested in full time MBA	-2.8602	0.740	-3.866	0.000	-4.310	-1.410
Tags_Interested in other courses	-2.8332	0.328	-8.641	0.000	-3.476	-2.191
Tags_Lost to EINS	6.4558	0.839	7.692	0.000	4.811	8.101
Tags_Not doing further education	-3.5698	1.030	-3.467	0.001	-5.588	-1.552
Tags_Ringing	-4.0320	0.246	-16.378	0.000	-4.515	-3.550
Tags_Will revert after reading the email	3.7184	0.192	19.386	0.000	3.342	4.094
Tags_invalid number	-24.4886	1.64e+04	-0.001	0.999	-3.22e+04	3.21e+04
Tags_opp hangup	-3.2794	1.061	-3.092	0.002	-5.358	-1.201
Tags_switched off	-4.9237	0.723	-6.809	0.000	-6.341	-3.506
What is your current occupation_Unemployed	1.8623	0.115	16.189	0.000	1.637	2.088
What is your current occupation_Working Professional	2.0226	0.363	5.570	0.000	1.311	2.734
Last Activity_SMS Sent	1.9628	0.109	17.982	0.000	1.749	2.177

In [114]:

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

Out[114]:

```
8529    0.064635
7331    0.009613
7688    0.795734
92      0.078329
4908    0.078329
451     0.009613
4945    0.009613
2844    0.994720
4355    0.078329
7251    0.000879
dtype: float64
```

In [115]:

```
y_train_pred = y_train_pred.values.reshape(-1)
y_train_pred[:10]
```

Out[115]:

```
array([6.46349739e-02, 9.61261677e-03, 7.95733870e-01, 7.83285731e-02,
       7.83285731e-02, 9.61261677e-03, 9.61261677e-03, 9.94720023e-01,
       7.83285731e-02, 8.79091579e-04])
```

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

In [116]:

```
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()
```

Out[116]:

	Converted	Conversion_Prob	LeadID
0	0	0.064635	8529
1	0	0.009613	7331

	Converted	Conversion_Prob	LeadID
2	1	0.795734	7688
3	0	0.078329	92
4	0	0.078329	4908

In [117]:

```
y_train_pred_final['predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 els
e 0)

# Let's see the head
y_train_pred_final.head()
```

Out[117]:

	Converted	Conversion_Prob	LeadID	predicted
0	0	0.064635	8529	0
1	0	0.009613	7331	0
2	1	0.795734	7688	1
3	0	0.078329	92	0
4	0	0.078329	4908	0

In [118]:

```
from sklearn import metrics
```

In [119]:

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.predicted )
print(confusion)
```

```
[[3630  106]
 [ 409 1857]]
```

In [120]:

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

```
0.9141952682439187
```

In [121]:

```
# checking VIFs
```

In [122]:

```
# Check for the VIF values of the feature variables.
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [123]:

```
# Create a dataframe that will contain the names of all the feature variables and their respective
VIFs
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].values, i) for i in range(X_train[col].shape[1
])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[123]:

	Features	VIF
4	Tags_Closed by Horizzon	1.29
9	Tags_Not doing further education	1.26
14	Tags_switched off	1.19
6	Tags_Interested in full time MBA	1.12
5	Tags_Diploma holder (Not Eligible)	1.12
2	Asymmetrique Activity Index_03.Low	1.11
0	Lead Source_Welingak Website	1.09
12	Tags_invalid number	1.08
8	Tags_Lost to EINS	1.06
13	Tags_opp hangup	1.02
16	What is your current occupation_Working Profes...	0.79
1	Lead Quality_Worst	0.69
10	Tags_Ringing	0.61
7	Tags_Interested in other courses	0.39
3	Tags_Already a student	0.38
11	Tags_Will revert after reading the email	0.09
15	What is your current occupation_Unemployed	0.01
17	Last Activity_SMS Sent	0.00

Dropping the Variable and Updating the Model

In [124]:

```
col = col.drop('Tags_Diploma holder (Not Eligible)', 1)
col
```

Out[124]:

```
Index(['Lead Source_Welingak Website', 'Lead Quality_Worst',
       'Asymmetrique Activity Index_03.Low', 'Tags_Already a student',
       'Tags_Closed by Horizzon', 'Tags_Interested  in full time MBA',
       'Tags_Interested in other courses', 'Tags_Lost to EINS',
       'Tags_Not doing further education', 'Tags_Ringing',
       'Tags_Will revert after reading the email', 'Tags_invalid number',
       'Tags_opp hangup', 'Tags_switched off',
       'What is your current occupation_Unemployed',
       'What is your current occupation_Working Professional',
       'Last Activity_SMS Sent'],
      dtype='object')
```

In [125]:

```
X_train_sm = sm.add_constant(X_train[col])
logm5 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = logm5.fit()
res.summary()
```

Out[125]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5984
Model Family:	Binomial	**Df Model:**	17
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-1313.2
Date:	Tue, 12 May 2020	**Deviance:**	2626.4

Time:	13:39:08	**Pearson chi2:**	8.42e+03
No. Iterations:	23		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.4750	0.088	-28.020	0.000	-2.648	-2.302
Lead Source_Welingak Website	3.4678	0.731	4.747	0.000	2.036	4.900
Lead Quality_Worst	-2.8883	0.706	-4.092	0.000	-4.272	-1.505
Asymmetrique Activity Index_03.Low	-2.4330	0.351	-6.931	0.000	-3.121	-1.745
Tags_Already a student	-3.6149	0.723	-4.999	0.000	-5.032	-2.198
Tags_Closed by Horizzon	5.3212	0.721	7.382	0.000	3.908	6.734
Tags_Interested in full time MBA	-2.8081	0.740	-3.794	0.000	-4.259	-1.357
Tags_Interested in other courses	-2.7838	0.328	-8.493	0.000	-3.426	-2.141
Tags_Lost to EINS	6.5606	0.846	7.757	0.000	4.903	8.218
Tags_Not doing further education	-3.5144	1.030	-3.412	0.001	-5.533	-1.496
Tags_Ringing	-3.9921	0.246	-16.235	0.000	-4.474	-3.510
Tags_Will revert after reading the email	3.7631	0.192	19.646	0.000	3.388	4.138
Tags_invalid number	-24.4442	1.64e+04	-0.001	0.999	-3.22e+04	3.21e+04
Tags_opp hangup	-3.2379	1.061	-3.052	0.002	-5.317	-1.159
Tags_switched off	-4.8845	0.723	-6.756	0.000	-6.302	-3.467
What is your current occupation_Unemployed	1.8184	0.114	15.893	0.000	1.594	2.043
What is your current occupation_Working Professional	1.9876	0.362	5.486	0.000	1.277	2.698
Last Activity_SMS Sent	1.9808	0.109	18.198	0.000	1.767	2.194

In [126]:

```
y_train_pred = res.predict(X_train_sm)
y_train_pred[:10]
```

Out[126]:

```
8529    0.064888
7331    0.009483
7688    0.789866
92      0.077629
4908    0.077629
451     0.009483
4945    0.009483
2844    0.994813
4355    0.077629
7251    0.000777
dtype: float64
```

In [127]:

```
y_train_pred = y_train_pred.values.reshape(-1)
y_train_pred[:10]
```

Out[127]:

```
array([6.48878261e-02, 9.48266404e-03, 7.89866093e-01, 7.76292105e-02,
       7.76292105e-02, 9.48266404e-03, 9.48266404e-03, 9.94812863e-01,
       7.76292105e-02, 7.76508332e-04])
```

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

In [128]:

```
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()
```

```
y_train_pred_final.head()
```

Out[128]:

	Converted	Conversion_Prob	LeadID
0	0	0.064888	8529
1	0	0.009483	7331
2	1	0.789866	7688
3	0	0.077629	92
4	0	0.077629	4908

In [129]:

```
y_train_pred_final['predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 els
e 0)

# Let's see the head
y_train_pred_final.head()
```

Out[129]:

	Converted	Conversion_Prob	LeadID	predicted
0	0	0.064888	8529	0
1	0	0.009483	7331	0
2	1	0.789866	7688	1
3	0	0.077629	92	0
4	0	0.077629	4908	0

In [130]:

```
from sklearn import metrics
```

In [131]:

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.predicted )
print(confusion)
```

```
[[3629  107]
 [ 409 1857]]
```

In [132]:

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

```
0.9140286571142953
```

In [133]:

```
# checking VIFs
```

In [134]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [135]:

```
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].values, i) for i in range(X_train[col].shape[1
])]
```

```
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[135]:

	Features	VIF
4	Tags_Closed by Horizzon	1.28
8	Tags_Not doing further education	1.25
13	Tags_switched off	1.18
5	Tags_Interested in full time MBA	1.11
0	Lead Source_Welingak Website	1.08
11	Tags_invalid number	1.07
2	Asymmetrique Activity Index_03.Low	1.07
7	Tags_Lost to EINS	1.06
12	Tags_opp hangup	1.02
15	What is your current occupation_Working Profes...	0.78
1	Lead Quality_Worst	0.67
9	Tags_Ringing	0.59
6	Tags_Interested in other courses	0.38
3	Tags_Already a student	0.37
10	Tags_Will revert after reading the email	0.09
14	What is your current occupation_Unemployed	0.01
16	Last Activity_SMS Sent	0.00

Dropping the Variable and Updating the Model

In [136]:

```
col = col.drop('Tags_invalid number', 1)
col
```

Out[136]:

```
Index(['Lead Source_Welingak Website', 'Lead Quality_Worst',
       'Asymmetrique Activity Index_03.Low', 'Tags_Already a student',
       'Tags_Closed by Horizzon', 'Tags_Interested  in full time MBA',
       'Tags_Interested in other courses', 'Tags_Lost to EINS',
       'Tags_Not doing further education', 'Tags_Ringing',
       'Tags_Will revert after reading the email', 'Tags_opp hangup',
       'Tags_switched off', 'What is your current occupation_Unemployed',
       'What is your current occupation_Working Professional',
       'Last Activity_SMS Sent'],
      dtype='object')
```

In [137]:

```
X_train_sm = sm.add_constant(X_train[col])
logm6 = sm.GLM(y_train,X_train_sm, family = sm.families.Binomial())
res = logm6.fit()
res.summary()
```

Out[137]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6002
Model:	GLM	**Df Residuals:**	5985
Model Family:	Binomial	**Df Model:**	16
Link Function:	logit	**Scale:**	1.0000

Method:	IRLS	**Log-Likelihood:**	-1342.4
Date:	Tue, 12 May 2020	**Deviance:**	2684.8
Time:	13:39:20	**Pearson chi2:**	8.52e+03
No. Iterations:	8		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.4751	0.088	-28.144	0.000	-2.647	-2.303
Lead Source_Welingak Website	3.6135	0.730	4.949	0.000	2.182	5.044
Lead Quality_Worst	-3.1794	0.670	-4.742	0.000	-4.494	-1.865
Asymmetrique Activity Index_03.Low	-2.3401	0.354	-6.605	0.000	-3.035	-1.646
Tags_Already a student	-3.4492	0.722	-4.776	0.000	-4.865	-2.034
Tags_Closed by Horizzon	5.4435	0.720	7.559	0.000	4.032	6.855
Tags_Interested in full time MBA	-2.6565	0.740	-3.591	0.000	-4.106	-1.207
Tags_Interested in other courses	-2.6347	0.327	-8.060	0.000	-3.275	-1.994
Tags_Lost to EINS	6.7102	0.862	7.786	0.000	5.021	8.399
Tags_Not doing further education	-3.3472	1.030	-3.250	0.001	-5.366	-1.329
Tags_Ringing	-3.8360	0.244	-15.709	0.000	-4.315	-3.357
Tags_Will revert after reading the email	3.8695	0.190	20.331	0.000	3.497	4.243
Tags_opp hangup	-3.0789	1.061	-2.903	0.004	-5.158	-1.000
Tags_switched off	-4.7274	0.722	-6.544	0.000	-6.143	-3.311
What is your current occupation_Unemployed	1.6711	0.112	14.926	0.000	1.452	1.891
What is your current occupation_Working Professional	1.8944	0.363	5.221	0.000	1.183	2.606
Last Activity_SMS Sent	1.9687	0.107	18.383	0.000	1.759	2.179

In [138]:

```
y_train_pred = res.predict(X_train_sm)
y_train_pred[:10]
```

Out[138]:

```
8529    0.064688
7331    0.009566
7688    0.762190
92      0.077626
4908    0.077626
451     0.009566
4945    0.009566
2844    0.994819
4355    0.077626
7251    0.000591
dtype: float64
```

In [139]:

```
y_train_pred = y_train_pred.values.reshape(-1)
y_train_pred[:10]
```

Out[139]:

```
array([6.46881585e-02, 9.56568869e-03, 7.62190244e-01, 7.76256984e-02,
       7.76256984e-02, 9.56568869e-03, 9.56568869e-03, 9.94818870e-01,
       7.76256984e-02, 5.91337209e-04])
```

In [140]:

```
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()
```

Out[140]:

	Converted	Conversion_Prob	LeadID
0	0	0.064688	8529
1	0	0.009566	7331
2	1	0.762190	7688
3	0	0.077626	92
4	0	0.077626	4908

In [141]:

```
y_train_pred_final['predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > 0.5 els
e 0)

# Let's see the head
y_train_pred_final.head()
```

Out[141]:

	Converted	Conversion_Prob	LeadID	predicted
0	0	0.064688	8529	0
1	0	0.009566	7331	0
2	1	0.762190	7688	1
3	0	0.077626	92	0
4	0	0.077626	4908	0

In [142]:

```
from sklearn import metrics
```

In [143]:

```
confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.predicted )
print(confusion)
```

```
[[3620  116]
 [ 409 1857]]
```

In [144]:

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

```
0.9125291569476841
```

In [145]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [146]:

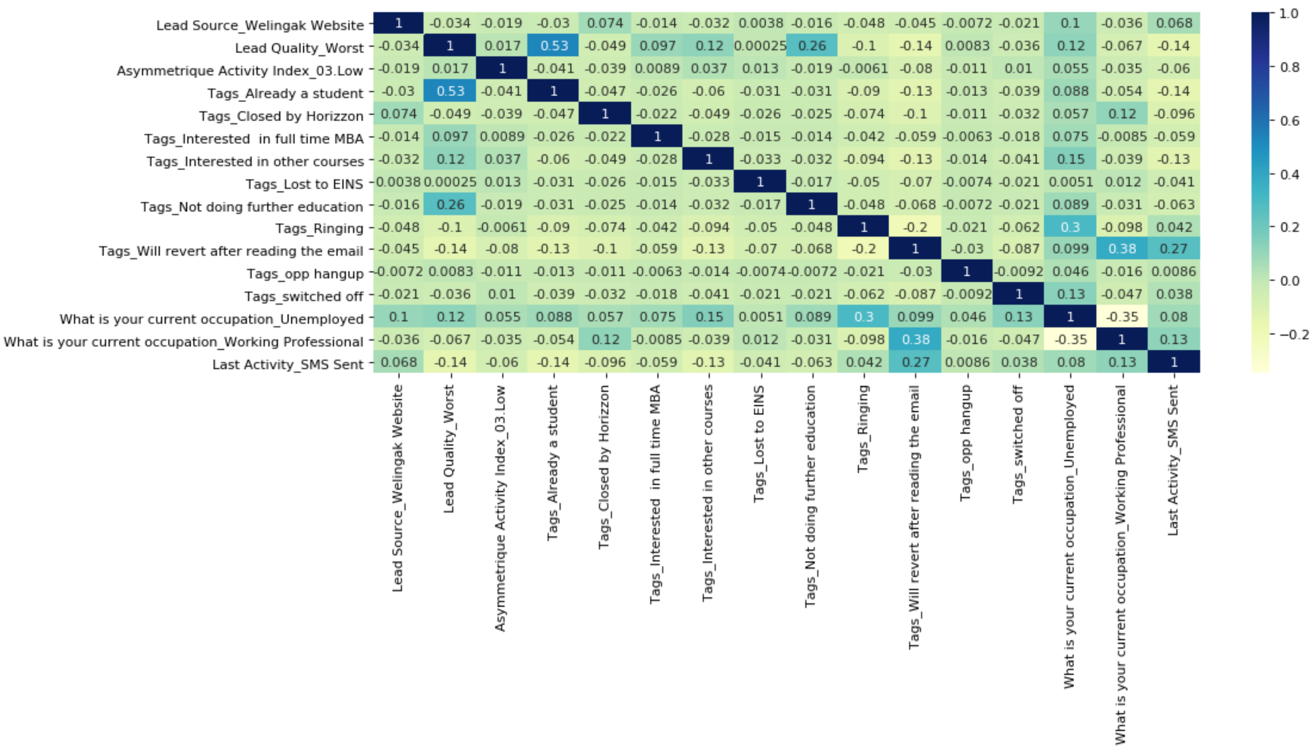
```
vif = pd.DataFrame()
vif['Features'] = X_train[col].columns
vif['VIF'] = [variance_inflation_factor(X_train[col].values, i) for i in range(X_train[col].shape[1
])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[146]:

	Features	VIF
4	Tags_Closed by Horizzon	1.26
8	Tags_Not doing further education	1.23
12	Tags_switched off	1.17
5	Tags_Interested in full time MBA	1.10
0	Lead Source_Welingak Website	1.08
2	Asymmetrique Activity Index_03.Low	1.07
7	Tags_Lost to EINS	1.06
11	Tags_opp hangup	1.02
14	What is your current occupation_Working Profes...	0.77
1	Lead Quality_Worst	0.67
9	Tags_Ringing	0.58
6	Tags_Interested in other courses	0.38
3	Tags_Already a student	0.36
10	Tags_Will revert after reading the email	0.09
13	What is your current occupation_Unemployed	0.01
15	Last Activity_SMS Sent	0.00

In [147]:

```
plt.figure(figsize=(15,8), dpi=80, facecolor='w', edgecolor='k', frameon='True')

cor = X_train[col].corr()
sns.heatmap(cor, annot=True, cmap="YlGnBu")

plt.tight_layout()
plt.show()
```



Step 8: Calculating Metrics beyond Accuracy

In [148]:

```
TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
```

```
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives
```

In [149]:

```
sensitivity=TP / float(TP+FN)
print(sensitivity)
specificity=TN / float(TN+FP)
print(specificity)
print(FP/ float(TN+FP))
print (TP / float(TP+FP))
print (TN / float(TN+ FN))
```

```
0.8195057369814651
0.9689507494646681
0.031049250535331904
0.941206284845413
0.8984859766691486
```

Step 9: Plotting the ROC Curve

It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).

The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.

The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

In [150]:

```
def draw_roc( actual, probs ):
    fpr, tpr, thresholds = metrics.roc_curve( actual, probs,
                                              drop_intermediate = False )
    auc_score = metrics.roc_auc_score( actual, probs )
    plt.figure(figsize=(5, 5))
    plt.plot( fpr, tpr, label='ROC curve (area = %0.2f)' % auc_score )
    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate or [1 - True Negative Rate]')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()

    return fpr,tpr, thresholds
```

In [151]:

```
fpr, tpr, thresholds = metrics.roc_curve( y_train_pred_final.Converted, y_train_pred_final.Conversi
on_Prob, drop_intermediate = False )
```

draw_roc(y_train_pred_final.Converted, y_train_pred_final.Conversion_Prob)

Calculating the area under the curve(GINI)

In [152]:

```
def auc_val(fpr,tpr):
    AreaUnderCurve = 0.
    for i in range(len(fpr)-1):
        AreaUnderCurve += (fpr[i+1]-fpr[i]) * (tpr[i+1]+tpr[i])
    AreaUnderCurve *= 0.5
    return AreaUnderCurve
```

In [153]:

```
auc = auc_val(fpr,tpr)
```

```
auc
```

Out[153]:

```
0.9623860234430959
```

Step 10: Finding Optimal Cutoff Point

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

In [154]:

```
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[154]:

	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 [155]:

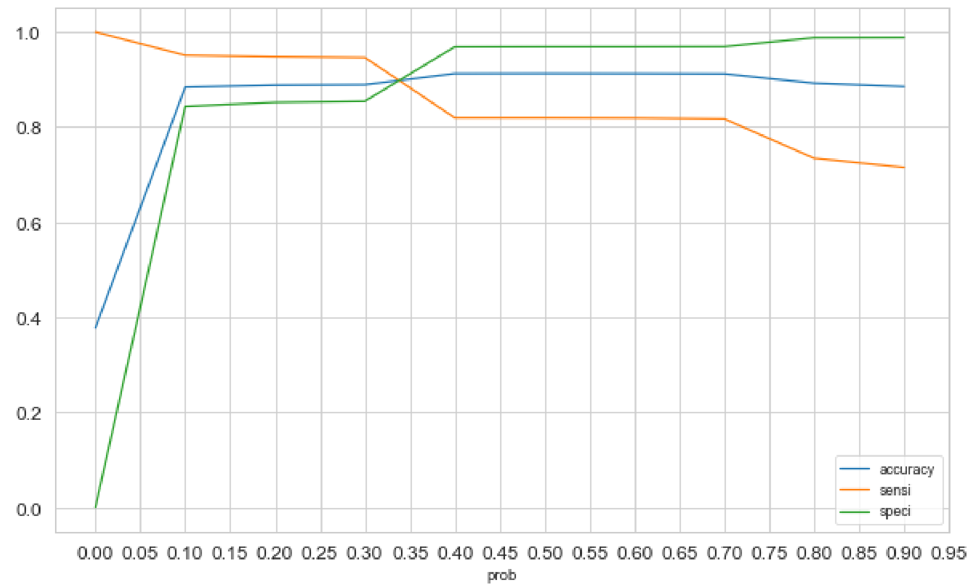
```
cutoff_df = pd.DataFrame( columns = ['prob','accuracy','sensi','speci'])
from sklearn.metrics import confusion_matrix

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

num = [0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
for i in num:
    cm1 = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final[i] )
    total1=sum(sum(cm1))
    accuracy = (cm1[0,0]+cm1[1,1])/total1

    speci = cm1[0,0]/(cm1[0,0]+cm1[0,1])
    sensi = cm1[1,1]/(cm1[1,0]+cm1[1,1])
    cutoff_df.loc[i] =[ i ,accuracy,sensi,speci]
print(cutoff_df)
```

```
     prob  accuracy     sensi     speci
0.0   0.0  0.377541  1.000000  0.000000
0.1   0.1  0.884039  0.951015  0.843415
0.2   0.2  0.888204  0.947926  0.851981
0.3   0.3  0.889037  0.946161  0.854390
0.4   0.4  0.912363  0.819506  0.968683
0.5   0.5  0.912529  0.819506  0.968951
0.6   0.6  0.912363  0.819064  0.968951
0.7   0.7  0.911863  0.817299  0.969218
0.8   0.8  0.892203  0.734334  0.987955
0.9   0.9  0.885205  0.715357  0.988223
```

In [156]:

```
sns.set_style("whitegrid") # white/whitegrid/dark/ticks
sns.set_context("paper") # talk/poster
cutoff_df.plot.line(x='prob', y=['accuracy','sensi','speci'], figsize=(10,6))
# plot x axis limits
plt.xticks(np.arange(0, 1, step=0.05), size = 12)
plt.yticks(size = 12)
```

```
plt.show()
```



In [157]:

```
y_train_pred_final.head()
```

Out[157]:

	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 [158]:

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

y_train_pred_final.head()
```

Out[158]:

	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	final_predicted
0	0	0.064688	8529	0	1	0	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	0
2	1	0.762190	7688	1	1	1	1	1	1	1	1	1	0	0	1
3	0	0.077626	92	0	1	0	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	0

In [159]:

```
# Let's check the overall accuracy.
metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.final_predicted)
```

Out[159]:

0.9031989336887704

In [160]:

In [160]:

```
confusion1 = metrics.confusion_matrix(y_train_pred_final.Converted,
y_train_pred_final.final_predicted)
confusion1
```

Out[160]:

```
array([[3411,  325],
       [ 256, 2010]], dtype=int64)
```

In [161]:

```
TP = confusion1[1,1] # true positive
TN = confusion1[0,0] # true negatives
FP = confusion1[0,1] # false positives
FN = confusion1[1,0] # false negatives
```

In [162]:

```
# Let's see the sensitivity of our logistic regression model
TP / float(TP+FN)
```

Out[162]:

```
0.8870255957634599
```

In [163]:

```
# Let us calculate specificity
TP / float(TP+FN)
```

Out[163]:

```
0.8870255957634599
```

In [164]:

```
# Calculate false postive rate - predicting churn when customer does not have churned
print(FP/ float(TN+FP))
```

```
0.0869914346895075
```

In [165]:

```
print (TP / float(TP+FP))
```

```
0.860813704496788
```

In [166]:

```
print (TN / float(TN+ FN))
```

```
0.9301881647122989
```

Step 11: Precision and Recall

Precision

TP / TP + FP

In [167]:

```
precision = confusion1[1,1]/(confusion1[0,1]+confusion1[1,1])
precision
```

Out[167]:

```
0.860813704496788
```

Recall

TP / TP + FN

In [168]:

```
recall = confusion1[1,1]/(confusion1[1,0]+confusion1[1,1])
recall
```

Out[168]:

```
0.8870255957634599
```

In [169]:

```
from sklearn.metrics import precision_score, recall_score
```

In [170]:

```
precision_score(y_train_pred_final.Converted, y_train_pred_final.final_predicted)
```

Out[170]:

```
0.860813704496788
```

In [171]:

```
recall_score(y_train_pred_final.Converted, y_train_pred_final.final_predicted)
```

Out[171]:

```
0.8870255957634599
```

Precision and recall tradeoff

In [172]:

```
from sklearn.metrics import precision_recall_curve
```

In [173]:

```
y_train_pred_final.Converted, y_train_pred_final.final_predicted
```

Out[173]:

```
(0       0
 1       0
 2       1
 3       0
 4       0
        ..
 5997    0
 5998    0
 5999    0
 6000    1
 6001    0
 Name: Converted, Length: 6002, dtype: int64,
 0       0
 1       0
 2       1
 3       0
 4       0
```

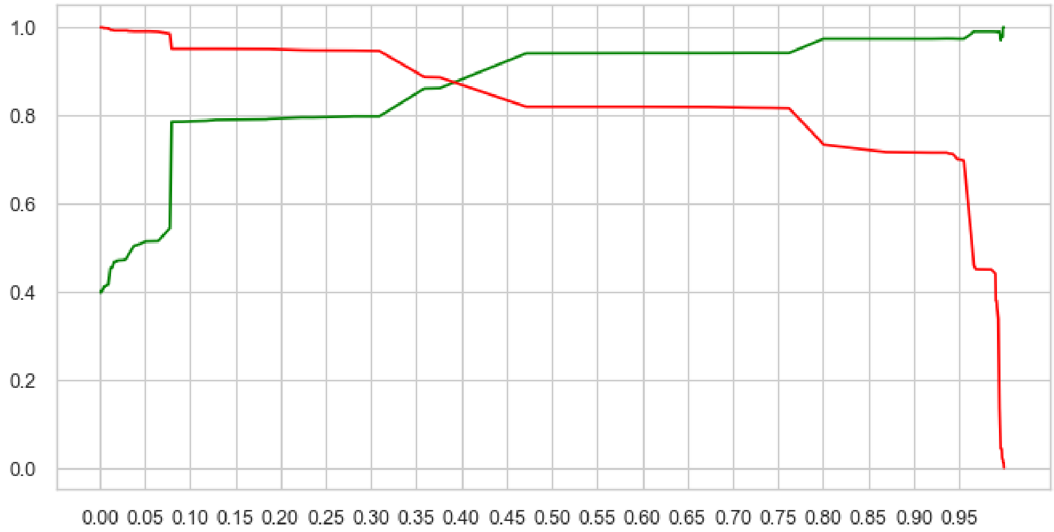
```
        ..
5997    0
5998    0
5999    0
6000    1
6001    0
Name: final_predicted, Length: 6002, dtype: int64)
```

In [174]:

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

In [175]:

```
plt.figure(figsize=(8, 4), dpi=100, facecolor='w', edgecolor='k', frameon='True')
plt.plot(thresholds, p[:-1], "g-")
plt.plot(thresholds, r[:-1], "r-")
plt.xticks(np.arange(0, 1, step=0.05))
plt.show()
```



From the precision-recall graph above, we get the optical threshold value as close to .37. However our business requirement here is to have Lead Conversion Rate around 80%

Calculating the F1 score

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

In [176]:

```
F1 = 2*(precision*recall)/(precision+recall)
F1
```

Out[176]:

```
0.8737231036731146
```

Step 12: Making predictions on the test set

In [177]:

```
X_test[['TotalVisits','Total Time Spent on Website','Page Views Per Visit']] = scaler.transform(X_
test[['TotalVisits','Total Time Spent on Website','Page Views Per Visit']])
X_test.head()
```

Out[177]:

	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	Newspaper Article	X Education Forums	Newspaper	Digital Advertisement	...	Last Activity_Form Submitted on Website	Activ Conv
6190	0	0	-1.199737	-0.872062	-1.270553	0	0	0	0	0	...	0	
7073	0	0	0.969969	-0.615211	1.785283	0	0	0	0	0	...	0	
4519	1	0	-1.199737	-0.872062	-1.270553	0	0	0	0	0	...	0	
607	0	0	-1.199737	-0.872062	-1.270553	0	0	0	0	0	...	0	
440	0	0	1.403911	-0.094170	0.562949	0	0	0	0	0	...	0	

5 rows × 141 columns

In [178]:

```
X_test = X_test[col]
X_test.head()
```

Out[178]:

	Lead Source_Welingak Website	Lead Quality_Worst	Asymmetrique Activity Index_03.Low	Tags_Already a student	Tags_Closed by Horizzon	Tags_Interested in full time MBA	Tags_Interested in other courses	Tags_Lost to EINS	Tags d fu educa
6190	0	1	0	1	0	0	0	0	
7073	0	0	0	0	0	0	0	0	
4519	0	0	0	0	0	0	0	0	
607	1	0	0	0	0	0	0	0	
440	0	0	0	0	0	0	0	0	

In [179]:

```
X_test_sm = sm.add_constant(X_test)
```

In [180]:

```
y_test_pred = res.predict(X_test_sm)
```

In [181]:

```
y_test_pred[:10]
```

Out[181]:

```
6190    0.000591
7073    0.077626
4519    0.309185
607     0.999825
440     0.077626
4247    0.077626
7431    0.008041
726     0.376039
7300    0.008041
4046    0.077626
dtype: float64
```

In [182]:

```
y_pred_1 = pd.DataFrame(y_test_pred)
```

In [183]:

```
y_pred_1.head()
```

Out[183]:

	0
6190	0.000591
7073	0.077626
4519	0.309185
607	0.999825
440	0.077626

In [184]:

```
y_test_df = pd.DataFrame(y_test)
```

In [185]:

```
y_test_df['LeadID'] = y_test_df.index
```

In [186]:

```
y_pred_1.reset_index(drop=True, inplace=True)
y_test_df.reset_index(drop=True, inplace=True)
```

In [187]:

```
y_pred_final = pd.concat([y_test_df, y_pred_1],axis=1)
```

In [188]:

```
y_pred_final.head()
```

Out[188]:

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

In [189]:

```
y_pred_final= y_pred_final.rename(columns={ 0 : 'Conversion_Prob'})
```

In [190]:

```
y_pred_final.head()
```

Out[190]:

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

	Converted	LeadID	Conversion_Prob
4	0	440	0.077626

In [191]:

```
y_pred_final.shape
```

Out[191]:

```
(2573, 3)
```

In [192]:

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

In [193]:

```
y_pred_final.head()
```

Out[193]:

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

In [194]:

```
acc_score=metrics.accuracy_score(y_pred_final.Converted, y_pred_final.final_predicted)
acc_score
```

Out[194]:

```
0.9055577147298873
```

In [195]:

```
confusion_test = metrics.confusion_matrix(y_pred_final.Converted, y_pred_final.final_predicted )
print(confusion_test)
```

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

Confusion Matrix in Visuals

In [196]:

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
```

In [197]:

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

Sensitivity TP / TP + FN

In [198]:

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

Out[198]:

```
0.8885542168674698
```

Specificity TN / TN + FP

In [199]:

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

Out[199]:

```
0.9162967660114141
```

False Postive Rate FP / TN + FP

In [200]:

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

```
0.08370323398858592
```

Positive Predictive Value TP / TP + FP

In [201]:

```
print (TP / float(TP+FP))
```

```
0.8702064896755162
```

Negative Predictive Value TN / TN + FN

In [202]:

```
print (TN / float(TN+ FN))
```

```
0.9286632390745502
```

Precision TP / TP + FP

In [203]:

```
Precision = confusion_test[1,1]/(confusion_test[0,1]+confusion_test[1,1])
Precision
```

Out[203]:

```
0.8702064896755162
```

Recall TP / TP + FN

In [204]:

```
Recall = confusion_test[1,1]/(confusion_test[1,0]+confusion_test[1,1])
Recall
```

Out[204]:

```
0.8885542168674698
```

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

In [205]:

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

Out[205]:

```
0.879284649776453
```

In [206]:

```
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

    accuracy                           0.91      2573
   macro avg       0.90      0.90      0.90      2573
weighted avg       0.91      0.91      0.91      2573
```

In [207]:

```
from sklearn.model_selection import cross_val_score

lr = LogisticRegression(solver = 'lbfgs')
scores = cross_val_score(lr, X, y, cv=10)
scores.sort()
accuracy = scores.mean()

print(scores)
print(accuracy)
```

```
[0.84364061 0.87762238 0.89731622 0.90898483 0.91608392 0.92307692
 0.92540793 0.92648775 0.93006993 0.9369895 ]
0.9085679975411598
```

Plotting the ROC Curve for Test Dataset

In [208]:

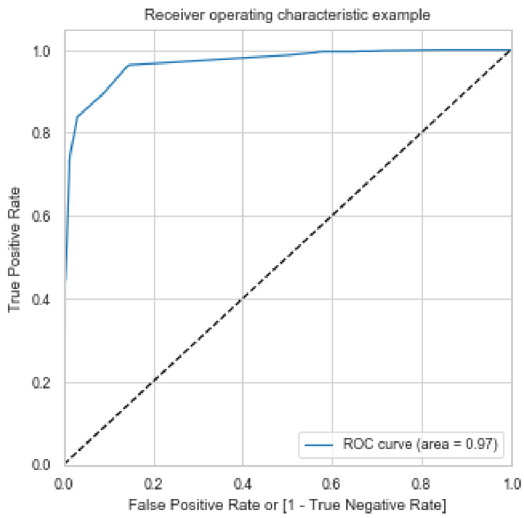
```
def draw_roc( actual, probs ):
    fpr, tpr, thresholds = metrics.roc_curve( actual, probs,
                                              drop_intermediate = False )
    auc_score = metrics.roc_auc_score( actual, probs )
    plt.figure(figsize=(5, 5))
    plt.plot( fpr, tpr, label='ROC curve (area = %0.2f)' % auc_score )
    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate or [1 - True Negative Rate]')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()

    return fpr,tpr, thresholds
```

In [209]:

```
fpr, tpr, thresholds = metrics.roc_curve( y_pred_final.Converted, y_pred_final.Conversion_Prob, dro
p_intermediate = False )
```

In [210]:

```
draw_roc(y_pred_final.Converted, y_pred_final.Conversion_Prob)
```



Out[210]:

```
(array([0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
        0.00000000e+00, 0.00000000e+00, 6.34115409e-04, 2.53646164e-03,
        2.53646164e-03, 2.53646164e-03, 2.53646164e-03, 2.53646164e-03,
        3.80469245e-03, 3.80469245e-03, 6.34115409e-03, 1.26823082e-02,
        1.26823082e-02, 1.26823082e-02, 1.26823082e-02, 1.33164236e-02,
        1.33164236e-02, 1.39505390e-02, 1.39505390e-02, 2.98034242e-02,
        2.98034242e-02, 2.98034242e-02, 8.24350032e-02, 8.37032340e-02,
        1.43310082e-01, 1.43944198e-01, 1.45846544e-01, 1.46480659e-01,
        1.47114775e-01, 1.49651237e-01, 1.50919467e-01, 1.52821814e-01,
        1.53455929e-01, 4.98414711e-01, 5.67533291e-01, 5.68167406e-01,
        5.82752061e-01, 5.85288523e-01, 6.37285986e-01, 6.48065948e-01,
        6.66455295e-01, 6.67089410e-01, 6.72162334e-01, 6.81039949e-01,
        6.81674065e-01, 6.88649334e-01, 7.12111604e-01, 8.44007609e-01,
        8.44641725e-01, 8.59860495e-01, 8.63665187e-01, 8.64299302e-01,
        8.70006341e-01, 8.71908687e-01, 8.72542803e-01, 8.73176918e-01,
        8.91566265e-01, 8.97273304e-01, 8.98541535e-01, 8.99175650e-01,
        9.00443881e-01, 9.01077996e-01, 9.02346227e-01, 9.02980342e-01,
        9.04248573e-01, 9.04882689e-01, 9.16296766e-01, 9.23272036e-01,
        9.23906151e-01, 9.31515536e-01, 9.43563729e-01, 9.74001268e-01,
        9.74635384e-01, 9.75269499e-01, 9.75903614e-01, 9.79074192e-01,
        9.79708307e-01, 9.80342422e-01, 9.98731769e-01, 9.99365885e-01,
        1.00000000e+00]),
 array([0.        , 0.00200803, 0.0060241 , 0.00702811, 0.00903614,
        0.01305221, 0.01506024, 0.02208835, 0.0251004 , 0.02710843,
        0.03413655, 0.03614458, 0.03915663, 0.06024096, 0.14959839,
        0.33032129, 0.35240964, 0.37048193, 0.42670683, 0.4437751 ,
        0.44578313, 0.45180723, 0.51907631, 0.71485944, 0.71987952,
        0.73192771, 0.73393574, 0.73393574, 0.73493976, 0.74497992,
        0.74598394, 0.83333333, 0.83534137, 0.8373494 , 0.88855422,
        0.88855422, 0.96184739, 0.96184739, 0.96285141, 0.96285141,
        0.96285141, 0.96385542, 0.96385542, 0.96385542, 0.96385542,
        0.98694779, 0.9939759 , 0.99497992, 0.99598394, 0.99598394,
        0.99598394, 0.99598394, 0.99698795, 0.99698795, 0.99698795,
        0.99698795, 0.99698795, 0.99698795, 0.99799197, 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        , 1.        , 1.        ,
        1.        , 1.        , 1.        ]),
 array([1.99998975e+00, 9.99989752e-01, 9.99963630e-01, 9.99926618e-01,
        9.99824507e-01, 9.99739607e-01, 9.99696013e-01, 9.99619996e-01,
        9.98921946e-01, 9.98744640e-01, 9.98652624e-01, 9.97982408e-01,
        9.97827196e-01, 9.97285124e-01, 9.94818870e-01, 9.93531147e-01,
        9.92330917e-01, 9.91658985e-01, 9.90430784e-01, 9.85729281e-01,
        9.72513628e-01, 9.66532756e-01, 9.64045300e-01, 9.55451586e-01
```

```
       9.51129211e-01, 9.43188923e-01, 9.38594823e-01, 9.36681743e-01,
       9.08834949e-01, 8.01307904e-01, 8.00272142e-01, 7.62190244e-01,
       7.20870097e-01, 6.73819364e-01, 3.76039363e-01, 3.58780052e-01,
       3.09184642e-01, 2.35885412e-01, 1.86945330e-01, 1.83648482e-01,
       1.28515565e-01, 1.17670662e-01, 1.01339571e-01, 9.24171186e-02,
       7.95826653e-02, 7.76256984e-02, 6.46881585e-02, 5.48630241e-02,
       4.13271293e-02, 3.85913902e-02, 3.11094131e-02, 3.04578364e-02,
       2.75807056e-02, 2.27525240e-02, 2.01774252e-02, 1.82829397e-02,
       1.74663085e-02, 1.55031513e-02, 1.40202683e-02, 9.56568869e-03,
       9.47683387e-03, 8.04083211e-03, 6.61749672e-03, 6.09878155e-03,
       6.00129941e-03, 5.87241677e-03, 4.92713243e-03, 4.21923408e-03,
       3.94509344e-03, 3.08308090e-03, 3.01667910e-03, 2.95232373e-03,
       2.72443128e-03, 2.23748987e-03, 1.79056558e-03, 1.66748292e-03,
       1.51445478e-03, 1.36773766e-03, 1.33426161e-03, 1.30547565e-03,
       1.17880864e-03, 9.29384926e-04, 6.54824520e-04, 5.91337209e-04,
       5.81186973e-04, 4.76695605e-04, 4.01716645e-04, 3.81344283e-04,
       2.45737689e-04, 1.28668971e-04, 1.11246454e-04, 6.31089388e-05,
       5.69870559e-05]))
```

In [211]:

```python
def auc_val(fpr,tpr):
    AreaUnderCurve = 0.
    for i in range(len(fpr)-1):
        AreaUnderCurve += (fpr[i+1]-fpr[i]) * (tpr[i+1]+tpr[i])
    AreaUnderCurve *= 0.5
    return AreaUnderCurve
```

In [212]:

```python
auc = auc_val(fpr,tpr)
auc
```

Out[212]:

```
0.9678947241088641
```

As a rule of thumb, an AUC can be classed as follows,

0.90 - 1.00 = excellent 0.80 - 0.90 = good 0.70 - 0.80 = fair 0.60 - 0.70 = poor 0.50 - 0.60 = fail

Step 13: Calculating Lead score for the entire dataset

In [213]:

```python
leads_test_pred = y_pred_final.copy()
leads_test_pred.head()
```

Out[213]:

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

In [214]:

```python
leads_train_pred = y_train_pred_final.copy()
leads_train_pred.head()
```

Out[214]:

	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	final_predicted
0	0	0.064688	8529	0	1	0	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	0
2	1	0.762190	7688	1	1	1	1	1	1	1	1	1	0	0	1
3	0	0.077626	92	0	1	0	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	0

In [215]:

```
leads_train_pred = leads_train_pred[['LeadID','Converted','Conversion_Prob','final_predicted']]
leads_train_pred.head()
```

Out[215]:

	LeadID	Converted	Conversion_Prob	final_predicted
0	8529	0	0.064688	0
1	7331	0	0.009566	0
2	7688	1	0.762190	1
3	92	0	0.077626	0
4	4908	0	0.077626	0

In [216]:

```
lead_full_pred = leads_train_pred.append(leads_test_pred)
lead_full_pred.head()
```

Out[216]:

	LeadID	Converted	Conversion_Prob	final_predicted
0	8529	0	0.064688	0
1	7331	0	0.009566	0
2	7688	1	0.762190	1
3	92	0	0.077626	0
4	4908	0	0.077626	0

In [217]:

```
print(leads_train_pred.shape)
print(leads_test_pred.shape)
print(lead_full_pred.shape)
```

```
(6002, 4)
(2573, 4)
(8575, 4)
```

In [218]:

```
len(lead_full_pred['LeadID'].unique().tolist())
```

Out[218]:

```
8575
```

In [219]:

```
lead_full_pred['Lead_Score'] = lead_full_pred['Conversion_Prob'].apply(lambda x : round(x*100))
lead_full_pred.head()
```

Out[219]:

	LeadID	Converted	Conversion_Prob	final_predicted	Lead_Score
0	8529	0	0.064688	0	6
1	7331	0	0.009566	0	1
2	7688	1	0.762190	1	76
3	92	0	0.077626	0	8
4	4908	0	0.077626	0	8

In [220]:

```
lead_full_pred.LeadID.max()
```

Out[220]:

9239

In [221]:

```
lead_full_pred = lead_full_pred.set_index('LeadID').sort_index(axis = 0, ascending = True)
lead_full_pred.head()
```

Out[221]:

	Converted	Conversion_Prob	final_predicted	Lead_Score
LeadID				
0	0	0.031109	0	3
1	0	0.009566	0	1
2	1	0.801308	1	80
3	0	0.009566	0	1
4	1	0.955452	1	96

In [222]:

```
original_leads = original_leads[['Lead Number']]
original_leads.head()
```

Out[222]:

	Lead Number
0	660737
1	660728
2	660727
3	660719
4	660681

In [223]:

```
leads_with_score = pd.concat([original_leads, lead_full_pred], axis=1)
leads_with_score.head(10)
```

Out[223]:

	Lead Number	Converted	Conversion_Prob	final_predicted	Lead_Score
0	660737	0	0.031109	0	3
1	660728	0	0.009566	0	1
2	660727	1	0.801308	1	80

	Lead Number	Converted	Conversion_Prob	final_predicted	Lead_Score
3	660719	0	0.009566	0	1
4	660681	1	0.955452	1	96
5	660680	0	0.077626	0	8
6	660673	1	0.955452	1	96
7	660664	0	0.077626	0	8
8	660624	0	0.077626	0	8
9	660616	0	0.077626	0	8

In [224]:

```
leads_with_score.shape
```

Out[224]:

```
(8575, 5)
```

In [224]:

```
total = pd.DataFrame(leads_with_score.isnull().sum().sort_values(ascending=False),
columns=['Total'])
percentage = pd.DataFrame(round(100*(leads_with_score.isnull().sum()/leads_with_score.shape[0]),2).
sort_values(ascending=False)\
                          ,columns=['Percentage'])
pd.concat([total, percentage], axis = 1)
```

Out[224]:

	Total	Percentage
Lead_Score	0	0.0
final_predicted	0	0.0
Conversion_Prob	0	0.0
Converted	0	0.0
Lead Number	0	0.0

Step 14: Determining Feature Importance

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

In [225]:

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

Out[225]:

```
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

In [226]:

```
#feature_importance = abs(new_params)
feature_importance = new_params
feature_importance = 100.0 * (feature_importance / feature_importance.max())
feature_importance
```

Out[226]:

```
Lead Source_Welingak Website                           53.85
Lead Quality_Worst                                    -47.38
Asymmetrique Activity Index_03.Low                    -34.87
Tags_Already a student                                -51.40
Tags_Closed by Horizzon                                81.12
Tags_Interested  in full time MBA                     -39.59
Tags_Interested in other courses                      -39.26
Tags_Lost to EINS                                     100.00
Tags_Not doing further education                      -49.88
Tags_Ringing                                          -57.17
Tags_Will revert after reading the email               57.67
Tags_opp hangup                                       -45.88
Tags_switched off                                     -70.45
What is your current occupation_Unemployed             24.90
What is your current occupation_Working Professional   28.23
Last Activity_SMS Sent                                 29.34
dtype: float64
```

Sorting the feature variables based on their relative coefficient values

In [227]:

```
sorted_idx = np.argsort(feature_importance,kind='quicksort',order='list of str')
sorted_idx
##
```

Out[227]:

```
Lead Source_Welingak Website                           12
Lead Quality_Worst                                      9
Asymmetrique Activity Index_03.Low                      3
Tags_Already a student                                  8
Tags_Closed by Horizzon                                 1
Tags_Interested  in full time MBA                      11
Tags_Interested in other courses                        5
Tags_Lost to EINS                                       6
Tags_Not doing further education                        2
Tags_Ringing                                           13
Tags_Will revert after reading the email               14
Tags_opp hangup                                        15
Tags_switched off                                       0
What is your current occupation_Unemployed             10
What is your current occupation_Working Professional    4
Last Activity_SMS Sent                                  7
dtype: int64
```

Plot showing the feature variables based on their relative coefficient values

In [228]:

```
pos = np.arange(sorted_idx.shape[0]) + .5

featfig = plt.figure(figsize=(10,6))
featax = featfig.add_subplot(1, 1, 1)
featax.barh(pos, feature_importance[sorted_idx], align='center', color = 'tab:red',alpha=0.8)
featax.set_yticks(pos)
featax.set_yticklabels(np.array(X_train[col].columns)[sorted_idx], fontsize=12)
featax.set_xlabel('Relative Feature Importance', fontsize=14)

plt.tight_layout()
```

```
plt.show()
```

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

In [229]:

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

Out[229]:

	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 o

In [ ]: