Lead Scoring Case Study

Problem Statement :
An education company named X Education sells online courses to industry professionals. On any given day, many professionals who are interested in the courses land on their website and browse for courses.
The company markets its courses on several websites and search engines like Google. Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos. When these people fill up a form providing their email address or phone number, they are classified to be a lead. Moreover, the company also gets leads through past referrals. Once these leads are acquired, employees from the sales team start making calls, writing emails, etc. Through this process, some of the leads get converted while most do not. The typical lead conversion rate at X education is around 30%.
There are a lot of leads generated in the initial stage (top) but only a few of them come out as paying customers from the bottom. In the middle stage, you need to nurture the potential leads well (i.e. educating the leads about the product, constantly communicating etc.) in order to get a higher lead conversion.
X Education has appointed you to help them select the most promising leads, i.e. the leads that are most likely to convert into paying customers. The company requires you to build a model wherein you need to assign a lead score to each of the leads such that the customers with higher lead score have a higher conversion chance and the customers with lower lead score have a lower conversion chance. The CEO, in particular, has given a ballpark of the target lead conversion rate to be around 80%.

Our Goals of the Case Study:

- To **build a logistic regression model to assign a lead score** between 0 and 100 to each of the leads which can be used by the company to target potential leads.
- To **adjust to if the company's requirement changes** in the future so you will need to handle these as well.

The steps are broadly:

1. Read and understand the data
2. Clean the data
3. Prepare the data for Model Building
4. Model Building
5. Model Evaluation
6. Making Predictions on the Test Set

Import modules

```
# Supress unnecessary warnings
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd

import time, warnings
import datetime as dt
```

```
from IPython.display import display
pd.options.display.max_columns = None
```

Read and understand the data

```
xleads = pd.read_csv(r'C:\Users\indranil1\Desktop\Indranil-Personal\Case Study\Machine

# Look at the first few entries
xleads.head()
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	C
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	Page Visited on Website	
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0	674	2.5	Email Opened	
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	660727	Landing Page Submission	Direct Traffic	No	No	1	2.0	1532	2.0	Email Opened	
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	No	0	1.0	305	1.0	Unreachable	
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	No	1	2.0	1428	1.0	Converted to Lead	

```
# Inspect the shape of the dataset

xleads.shape
```

```
(9240, 37)
```

```
# Inspect the different columsn in the dataset

xleads.columns
```

```
Index(['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', 'Last Activity',
       'Country', 'Specialization', 'How did you hear about X Education',
       'What is your current occupation',
       'What matters most to you in choosing a course', 'Search', 'Magazine',
       'Newspaper Article', 'X Education Forums', 'Newspaper',
```

```
       'Digital Advertisement', 'Through Recommendations',
       'Receive More Updates About Our Courses', 'Tags', 'Lead Quality',
       'Update me on Supply Chain Content', 'Get updates on DM Content',
       'Lead Profile', 'City', 'Asymmetrique Activity Index',
       'Asymmetrique Profile Index', 'Asymmetrique Activity Score',
       'Asymmetrique Profile Score',
       'I agree to pay the amount through cheque',
       'A free copy of Mastering The Interview', 'Last Notable Activity'],
      dtype='object')
```

```
xleads.describe()
```

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

```
# Check the summary of the dataset

xleads.describe(include='all')
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website
count	9240	9240.000000	9240	9204	9240	9240	9240.000000	9103.000000	9240.000000
unique	9240	NaN	5	21	2	2	NaN	NaN	NaN
top	d7896afe-20f6-4494-9a09-50d47321a9fb	NaN	Landing Page Submission	Google	No	No	NaN	NaN	NaN
freq	1	NaN	4886	2868	8506	9238	NaN	NaN	NaN
mean	NaN	617188.435606	NaN	NaN	NaN	NaN	0.385390	3.445238	487.698268
std	NaN	23405.995698	NaN	NaN	NaN	NaN	0.486714	4.854853	548.021466
min	NaN	579533.000000	NaN	NaN	NaN	NaN	0.000000	0.000000	0.000000
25%	NaN	596484.500000	NaN	NaN	NaN	NaN	0.000000	1.000000	12.000000
50%	NaN	615479.000000	NaN	NaN	NaN	NaN	0.000000	3.000000	248.000000
75%	NaN	637387.250000	NaN	NaN	NaN	NaN	1.000000	5.000000	936.000000

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website
max	NaN	660737.000000	NaN	NaN	NaN	NaN	1.000000	251.000000	2272.000000

```
# Check the info to see the types of the feature variables and the null values present

xleads.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 37 columns):
 #   Column                                         Non-Null Count  Dtype  
---  ------                                         --------------  -----  
 0   Prospect ID                                    9240 non-null   object 
 1   Lead Number                                    9240 non-null   int64  
 2   Lead Origin                                    9240 non-null   object 
 3   Lead Source                                    9204 non-null   object 
 4   Do Not Email                                   9240 non-null   object 
 5   Do Not Call                                    9240 non-null   object 
 6   Converted                                      9240 non-null   int64  
 7   TotalVisits                                    9103 non-null   float64
 8   Total Time Spent on Website                    9240 non-null   int64  
 9   Page Views Per Visit                           9103 non-null   float64
 10  Last Activity                                  9137 non-null   object 
 11  Country                                        6779 non-null   object 
 12  Specialization                                 7802 non-null   object 
 13  How did you hear about X Education             7033 non-null   object 
 14  What is your current occupation                6550 non-null   object 
 15  What matters most to you in choosing a course  6531 non-null   object 
 16  Search                                         9240 non-null   object 
 17  Magazine                                       9240 non-null   object 
 18  Newspaper Article                              9240 non-null   object 
 19  X Education Forums                             9240 non-null   object 
 20  Newspaper                                      9240 non-null   object 
 21  Digital Advertisement                          9240 non-null   object 
 22  Through Recommendations                        9240 non-null   object 
 23  Receive More Updates About Our Courses         9240 non-null   object 
 24  Tags                                           5887 non-null   object 
 25  Lead Quality                                   4473 non-null   object 
 26  Update me on Supply Chain Content              9240 non-null   object 
 27  Get updates on DM Content                      9240 non-null   object 
 28  Lead Profile                                   6531 non-null   object 
```

```
 29  City                                          7820 non-null   object
 30  Asymmetrique Activity Index                   5022 non-null   object
 31  Asymmetrique Profile Index                    5022 non-null   object
 32  Asymmetrique Activity Score                   5022 non-null   float64
 33  Asymmetrique Profile Score                    5022 non-null   float64
 34  I agree to pay the amount through cheque      9240 non-null   object
 35  A free copy of Mastering The Interview        9240 non-null   object
 36  Last Notable Activity                         9240 non-null   object
dtypes: float64(4), int64(3), object(30)
memory usage: 2.6+ MB
```

Looks like there are quite a few categorical variables present in this dataset for which we will need to create dummy variables. Also, there are a lot of null values present as well, so we will need to treat them accordingly.

Data Cleaning and Preparation

```
# Check the number of missing values in each column

xleads.isnull().sum()
```

```
Prospect ID                                       0
Lead Number                                       0
Lead Origin                                       0
Lead Source                                      36
Do Not Email                                      0
Do Not Call                                       0
Converted                                         0
TotalVisits                                     137
Total Time Spent on Website                       0
Page Views Per Visit                            137
Last Activity                                   103
Country                                        2461
Specialization                                 1438
How did you hear about X Education             2207
What is your current occupation                2690
What matters most to you in choosing a course  2709
Search                                            0
Magazine                                          0
Newspaper Article                                 0
X Education Forums                                0
Newspaper                                         0
Digital Advertisement                             0
Through Recommendations                           0
Receive More Updates About Our Courses            0
Tags                                           3353
Lead Quality                                   4767
Update me on Supply Chain Content                 0
Get updates on DM Content                         0
```

```
Lead Profile                                     2709
City                                             1420
Asymmetrique Activity Index                      4218
Asymmetrique Profile Index                       4218
Asymmetrique Activity Score                      4218
Asymmetrique Profile Score                       4218
I agree to pay the amount through cheque            0
A free copy of Mastering The Interview              0
Last Notable Activity                               0
dtype: int64
```

As you can see there are a lot of column which have high number of missing values. Clearly, these columns are not useful. Since, there are 9000 datapoints in our dataframe, let's eliminate the columns having greater than 3000 missing values as they are of no use to us.

```
# Drop all the columns in which greater than 3000 missing values are present

for col in xleads.columns:
    if xleads[col].isnull().sum() > 3000:
        xleads.drop(col, 1, inplace=True)
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                      0
Lead Number                                      0
Lead Origin                                      0
Lead Source                                     36
Do Not Email                                     0
Do Not Call                                      0
Converted                                        0
TotalVisits                                    137
Total Time Spent on Website                      0
Page Views Per Visit                           137
Last Activity                                  103
Country                                       2461
Specialization                                1438
How did you hear about X Education            2207
What is your current occupation               2690
What matters most to you in choosing a course 2709
Search                                           0
Magazine                                         0
Newspaper Article                                0
X Education Forums                               0
Newspaper                                        0
Digital Advertisement                            0
Through Recommendations                          0
Receive More Updates About Our Courses           0
```

```
Update me on Supply Chain Content                 0
Get updates on DM Content                         0
Lead Profile                                   2709
City                                           1420
I agree to pay the amount through cheque          0
A free copy of Mastering The Interview            0
Last Notable Activity                             0
dtype: int64
```

As you might be able to interpret, the variable City won't be of any use in our analysis. So it's best that we drop it.

```
xleads.drop(['City'], axis = 1, inplace = True)
```

```
# Same goes for the variable 'Country'

xleads.drop(['Country'], axis = 1, inplace = True)
```

```
# Let's now check the percentage of missing values in each column

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

```
Prospect ID                                    0.00
Lead Number                                    0.00
Lead Origin                                    0.00
Lead Source                                    0.39
Do Not Email                                   0.00
Do Not Call                                    0.00
Converted                                      0.00
TotalVisits                                    1.48
Total Time Spent on Website                    0.00
Page Views Per Visit                           1.48
Last Activity                                  1.11
Specialization                                15.56
How did you hear about X Education            23.89
What is your current occupation               29.11
What matters most to you in choosing a course 29.32
Search                                         0.00
Magazine                                       0.00
Newspaper Article                              0.00
X Education Forums                             0.00
Newspaper                                      0.00
Digital Advertisement                          0.00
Through Recommendations                        0.00
Receive More Updates About Our Courses         0.00
Update me on Supply Chain Content              0.00
Get updates on DM Content                      0.00
Lead Profile                                  29.32
I agree to pay the amount through cheque       0.00
A free copy of Mastering The Interview         0.00
```

```
Last Notable Activity                          0.00
dtype: float64
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                    0
Lead Number                                    0
Lead Origin                                    0
Lead Source                                   36
Do Not Email                                   0
Do Not Call                                    0
Converted                                      0
TotalVisits                                  137
Total Time Spent on Website                    0
Page Views Per Visit                         137
Last Activity                                103
Specialization                              1438
How did you hear about X Education          2207
What is your current occupation             2690
What matters most to you in choosing a course  2709
Search                                         0
Magazine                                       0
Newspaper Article                              0
X Education Forums                             0
Newspaper                                      0
Digital Advertisement                          0
Through Recommendations                        0
Receive More Updates About Our Courses         0
Update me on Supply Chain Content              0
Get updates on DM Content                      0
Lead Profile                                2709
I agree to pay the amount through cheque       0
A free copy of Mastering The Interview         0
Last Notable Activity                          0
dtype: int64
```

Now recall that there are a few columns in which there is a level called 'Select' which basically means that the student had not selected the option for that particular column which is why it shows 'Select'. These values are as good as missing values and hence we need to identify the value counts of the level 'Select' in all the columns that it is present.

```
# Get the value counts of all the columns

for column in xleads:
    print(xleads[column].astype('category').value_counts())
    print('_____________________________________________________')
```

```
fffb0e5e-9f92-4017-9f42-781a69da4154    1
```

```
56453aec-3f7b-4f30-870c-8f966d393100    1
53ac14bd-2bb2-4315-a21c-94562d1b6b2d    1
53aabd84-5dcc-4299-bbe3-62f3764b07b1    1
539ffa32-1be7-4fe1-b04c-faf1bab763cf    1
                                      ..
aa4180a5-84f1-4e67-8d90-0c8403070a59    1
aa405742-17ac-4c65-b19e-ab91c241cc53    1
aa30ebb2-8476-41ce-9258-37cc025110d3    1
aa27a0af-eeab-4007-a770-fa8a93fa53c8    1
000104b9-23e4-4ddc-8caa-8629fe8ad7f4    1
Name: Prospect ID, Length: 9240, dtype: int64
--------------------------------------------------
660737    1
603303    1
602561    1
602557    1
602540    1
         ..
630422    1
630405    1
630403    1
630390    1
579533    1
Name: Lead Number, Length: 9240, dtype: int64
--------------------------------------------------
Landing Page Submission    4886
API                        3580
Lead Add Form               718
Lead Import                  55
Quick Add Form                1
Name: Lead Origin, dtype: int64
--------------------------------------------------
Google               2868
Direct Traffic       2543
Olark Chat           1755
Organic Search       1154
Reference             534
Welingak Website      142
Referral Sites        125
Facebook               55
bing                    6
google                  5
Click2call              4
```

```
Press_Release           2
Social Media            2
Live Chat               2
Pay per Click Ads       1
welearnblog_Home        1
NC_EDM                  1
WeLearn                 1
blog                    1
testone                 1
youtubechannel          1
Name: Lead Source, dtype: int64
--------------------------------------------------
No     8506
Yes     734
Name: Do Not Email, dtype: int64
--------------------------------------------------
No     9238
Yes       2
Name: Do Not Call, dtype: int64
--------------------------------------------------
0    5679
1    3561
Name: Converted, dtype: int64
--------------------------------------------------
0.0      2189
2.0      1680
3.0      1306
4.0      1120
5.0       783
6.0       466
1.0       395
7.0       309
8.0       224
9.0       164
10.0      114
11.0       86
13.0       48
12.0       45
14.0       36
16.0       21
15.0       18
17.0       16
18.0       15
```

```
20.0       12
19.0        9
21.0        6
23.0        6
24.0        5
25.0        5
27.0        5
22.0        3
29.0        2
26.0        2
28.0        2
43.0        1
115.0       1
74.0        1
55.0        1
54.0        1
141.0       1
42.0        1
41.0        1
32.0        1
30.0        1
251.0       1
Name: TotalVisits, dtype: int64
--------------------------------------------------
0      2193
60       19
127      18
74       18
75       18
        ...
919       1
915       1
911       1
909       1
934       1
Name: Total Time Spent on Website, Length: 1731, dtype: int64
--------------------------------------------------
0.00     2189
2.00     1795
3.00     1196
4.00      896
1.00      651
         ...
```

```
1.86        1
3.80        1
3.82        1
3.83        1
55.00       1
Name: Page Views Per Visit, Length: 114, dtype: int64
--------------------------------------------------
Email Opened                    3437
SMS Sent                        2745
Olark Chat Conversation          973
Page Visited on Website          640
Converted to Lead                428
Email Bounced                    326
Email Link Clicked               267
Form Submitted on Website        116
Unreachable                       93
Unsubscribed                      61
Had a Phone Conversation          30
Approached upfront                 9
View in browser link Clicked       6
Email Marked Spam                  2
Email Received                     2
Resubscribed to emails             1
Visited Booth in Tradeshow         1
Name: Last Activity, dtype: int64
--------------------------------------------------
Select                            1942
Finance Management                 976
Human Resource Management          848
Marketing Management               838
Operations Management              503
Business Administration            403
IT Projects Management             366
Supply Chain Management            349
Banking, Investment And Insurance  338
Travel and Tourism                 203
Media and Advertising              203
International Business             178
Healthcare Management              159
Hospitality Management             114
E-COMMERCE                         112
Retail Management                  100
Rural and Agribusiness              73
```

```
E-Business                             57
Services Excellence                    40
Name: Specialization, dtype: int64
--------------------------------------------------
Select                   5043
Online Search             808
Word Of Mouth             348
Student of SomeSchool     310
Other                     186
Multiple Sources          152
Advertisements             70
Social Media               67
Email                      26
SMS                        23
Name: How did you hear about X Education, dtype: int64
--------------------------------------------------
Unemployed              5600
Working Professional     706
Student                  210
Other                     16
Housewife                 10
Businessman                8
Name: What is your current occupation, dtype: int64
--------------------------------------------------
Better Career Prospects      6528
Flexibility & Convenience       2
Other                           1
Name: What matters most to you in choosing a course, dtype: int64
--------------------------------------------------
No     9226
Yes      14
Name: Search, dtype: int64
--------------------------------------------------
No    9240
Name: Magazine, dtype: int64
--------------------------------------------------
No     9238
Yes       2
Name: Newspaper Article, dtype: int64
--------------------------------------------------
No     9239
Yes       1
Name: X Education Forums, dtype: int64
```

```
--------------------------------------------------
No     9239
Yes       1
Name: Newspaper, dtype: int64
--------------------------------------------------
No     9236
Yes       4
Name: Digital Advertisement, dtype: int64
--------------------------------------------------
No     9233
Yes       7
Name: Through Recommendations, dtype: int64
--------------------------------------------------
No    9240
Name: Receive More Updates About Our Courses, dtype: int64
--------------------------------------------------
No    9240
Name: Update me on Supply Chain Content, dtype: int64
--------------------------------------------------
No    9240
Name: Get updates on DM Content, dtype: int64
--------------------------------------------------
Select                         4146
Potential Lead                 1613
Other Leads                     487
Student of SomeSchool           241
Lateral Student                  24
Dual Specialization Student      20
Name: Lead Profile, dtype: int64
--------------------------------------------------
No    9240
Name: I agree to pay the amount through cheque, dtype: int64
--------------------------------------------------
No     6352
Yes    2888
Name: A free copy of Mastering The Interview, dtype: int64
--------------------------------------------------
Modified                       3407
Email Opened                   2827
SMS Sent                       2172
Page Visited on Website         318
Olark Chat Conversation         183
Email Link Clicked              173
```

```
Email Bounced                   60
Unsubscribed                    47
Unreachable                     32
Had a Phone Conversation        14
Email Marked Spam                2
View in browser link Clicked     1
Resubscribed to emails           1
Form Submitted on Website        1
Email Received                   1
Approached upfront               1
Name: Last Notable Activity, dtype: int64
```

The following three columns now have the level 'Select'. Let's check them once again.

```
xleads['Lead Profile'].astype('category').value_counts()
```

```
Select                         4146
Potential Lead                 1613
Other Leads                     487
Student of SomeSchool           241
Lateral Student                  24
Dual Specialization Student      20
Name: Lead Profile, dtype: int64
```

```
xleads['How did you hear about X Education'].value_counts()
```

```
Select                   5043
Online Search             808
Word Of Mouth             348
Student of SomeSchool     310
Other                     186
Multiple Sources          152
Advertisements             70
Social Media               67
Email                      26
SMS                        23
Name: How did you hear about X Education, dtype: int64
```

```
xleads['Specialization'].value_counts()
```

```
Select                           1942
Finance Management                976
Human Resource Management         848
Marketing Management              838
Operations Management             503
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                   159
Hospitality Management                  114
E-COMMERCE                              112
Retail Management                       100
Rural and Agribusiness                   73
E-Business                               57
Services Excellence                      40
Name: Specialization, dtype: int64
```

Clearly the levels Lead Profile and How did you hear about X Education have a lot of rows which have the value Select which is of no use to the analysis so it's best that we drop them.

```
xleads.drop(['Lead Profile', 'How did you hear about X Education'], axis = 1, inplace =
```

Also notice that when you got the value counts of all the columns, there were a few columns in which only one value was majorly present for all the data points. These include `Do Not Call`, `Search`, `Magazine`, `Newspaper Article`, `X Education Forums`, `Newspaper`, `Digital Advertisement`, `Through Recommendations`, `Receive More Updates About Our Courses`, `Update me on Supply Chain Content`, `Get updates on DM Content`, `I agree to pay the amount through cheque`. Since practically all of the values for these variables are `No`, it's best that we drop these columns as they won't help with our analysis.

```
xleads.drop(['Do Not Call', 'Search', 'Magazine', 'Newspaper Article', 'X Education For
             'Digital Advertisement', 'Through Recommendations', 'Receive More Updates A
             'Update me on Supply Chain Content', 'Get updates on DM Content',
             'I agree to pay the amount through cheque'], axis = 1, inplace = True)
```

Also, the variable `What matters most to you in choosing a course` has the level `Better Career Prospects` `6528` times while the other two levels appear once twice and once respectively. So we should drop this column as well.

```
xleads['What matters most to you in choosing a course'].value_counts()
```

```
Better Career Prospects      6528
Flexibility & Convenience       2
Other                           1
Name: What matters most to you in choosing a course, dtype: int64
```

```
# Drop the null value rows present in the variable 'What matters most to you in choosir

xleads.drop(['What matters most to you in choosing a course'], axis = 1, inplace=True)
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                    0
Lead Number                                    0
Lead Origin                                    0
Lead Source                                   36
Do Not Email                                   0
Converted                                      0
TotalVisits                                  137
Total Time Spent on Website                    0
Page Views Per Visit                         137
Last Activity                                103
Specialization                              1438
What is your current occupation             2690
A free copy of Mastering The Interview         0
Last Notable Activity                          0
dtype: int64
```

Now, there's the column `What is your current occupation` which has a lot of null values. Now you can drop the entire row but since we have already lost so many feature variables, we choose not to drop it as it might turn out to be significant in the analysis. So let's just drop the null rows for the column `What is you current occupation` .

```
xleads = xleads[~pd.isnull(xleads['What is your current occupation'])]
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                    0
Lead Number                                    0
Lead Origin                                    0
Lead Source                                   36
Do Not Email                                   0
Converted                                      0
TotalVisits                                  130
Total Time Spent on Website                    0
Page Views Per Visit                         130
Last Activity                                103
Specialization                                18
What is your current occupation                0
A free copy of Mastering The Interview         0
Last Notable Activity                          0
dtype: int64
```

Since now the number of null values present in the columns are quite small we can simply drop the rows in which these null values are present.

```
# Drop the null value rows in the column 'TotalVisits'

xleads = xleads[~pd.isnull(xleads['TotalVisits'])]
```

```
# Check the null values again

xleads.isnull().sum()
```

```
Prospect ID                                 0
Lead Number                                 0
Lead Origin                                 0
Lead Source                                29
Do Not Email                                0
Converted                                   0
TotalVisits                                 0
Total Time Spent on Website                 0
Page Views Per Visit                        0
Last Activity                               0
Specialization                             18
What is your current occupation             0
A free copy of Mastering The Interview      0
Last Notable Activity                       0
dtype: int64
```

```
# Drop the null values rows in the column 'Lead Source'

xleads = xleads[~pd.isnull(xleads['Lead Source'])]
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                 0
Lead Number                                 0
Lead Origin                                 0
Lead Source                                 0
Do Not Email                                0
Converted                                   0
TotalVisits                                 0
Total Time Spent on Website                 0
Page Views Per Visit                        0
Last Activity                               0
Specialization                             18
What is your current occupation             0
A free copy of Mastering The Interview      0
Last Notable Activity                       0
dtype: int64
```

```
# Drop the null values rows in the column 'Specialization'

xleads = xleads[~pd.isnull(xleads['Specialization'])]
```

```
# Check the number of null values again

xleads.isnull().sum()
```

```
Prospect ID                                    0
Lead Number                                    0
Lead Origin                                    0
Lead Source                                    0
Do Not Email                                   0
Converted                                      0
TotalVisits                                    0
Total Time Spent on Website                    0
Page Views Per Visit                           0
Last Activity                                  0
Specialization                                 0
What is your current occupation                0
A free copy of Mastering The Interview         0
Last Notable Activity                          0
dtype: int64
```

Now your data doesn't have any null values. Let's now check the percentage of rows that we have retained.

```
print(len(xleads.index))
print(len(xleads.index)/9240)
```

```
6373
0.6897186147186147
```

We still have around 69% of the rows which seems good enough.

```
# Let's look at the dataset again

xleads.head()
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Speci
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	0	0.0	0	0.0	Page Visited on Website	
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	0	5.0	674	2.5	Email Opened	

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Speci
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	660727	Landing Page Submission	Direct Traffic	No	1	2.0	1532	2.0	Email Opened	E Admin
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	0	1.0	305	1.0	Unreachable	Me Ad
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	1	2.0	1428	1.0	Converted to Lead	

Now, clearly the variables `Prospect ID` and `Lead Number` won't be of any use in the analysis, so it's best that we drop these two variables.

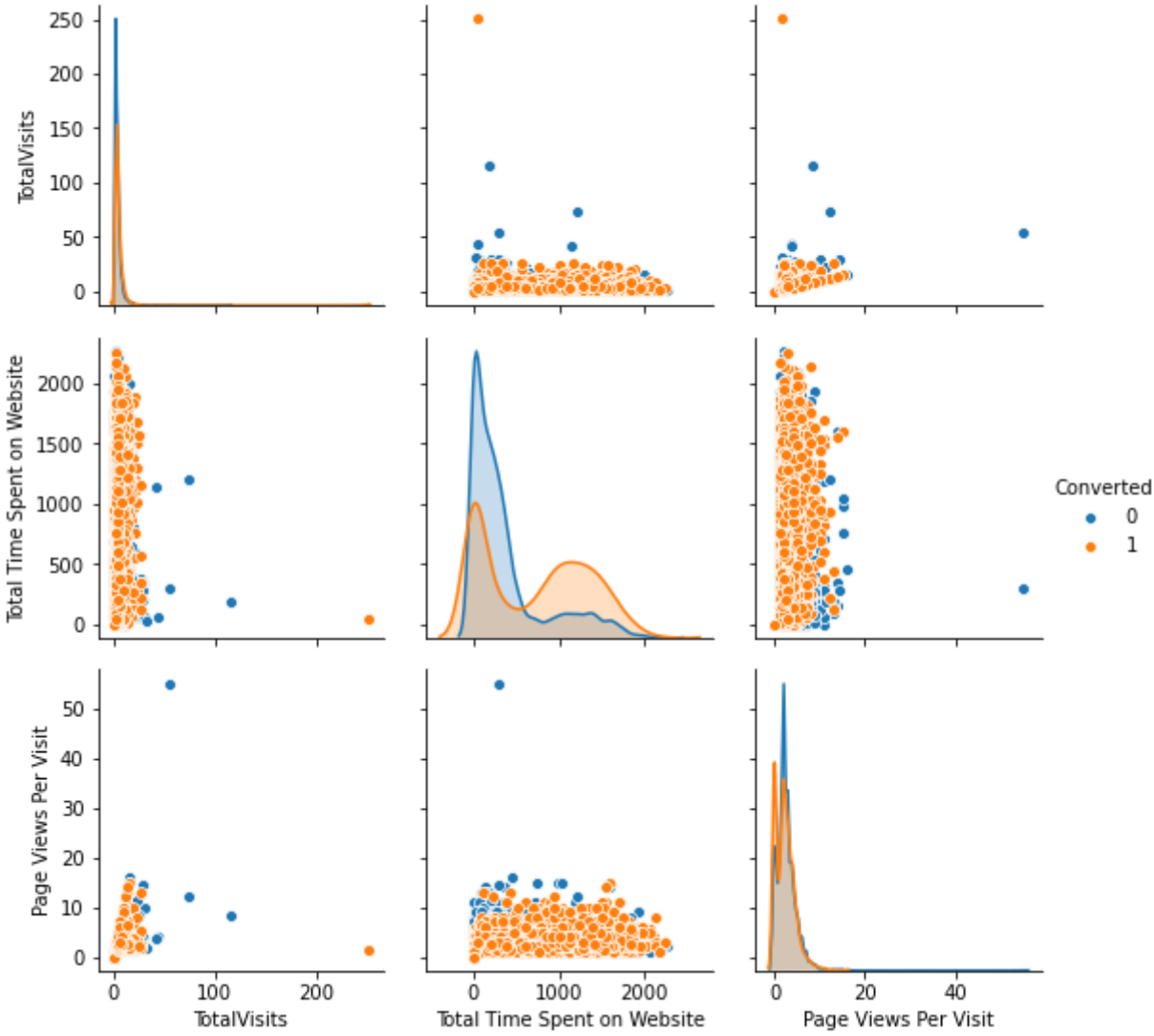
```
xleads.drop(['Prospect ID', 'Lead Number'], 1, inplace = True)
```

```
xleads.head()
```

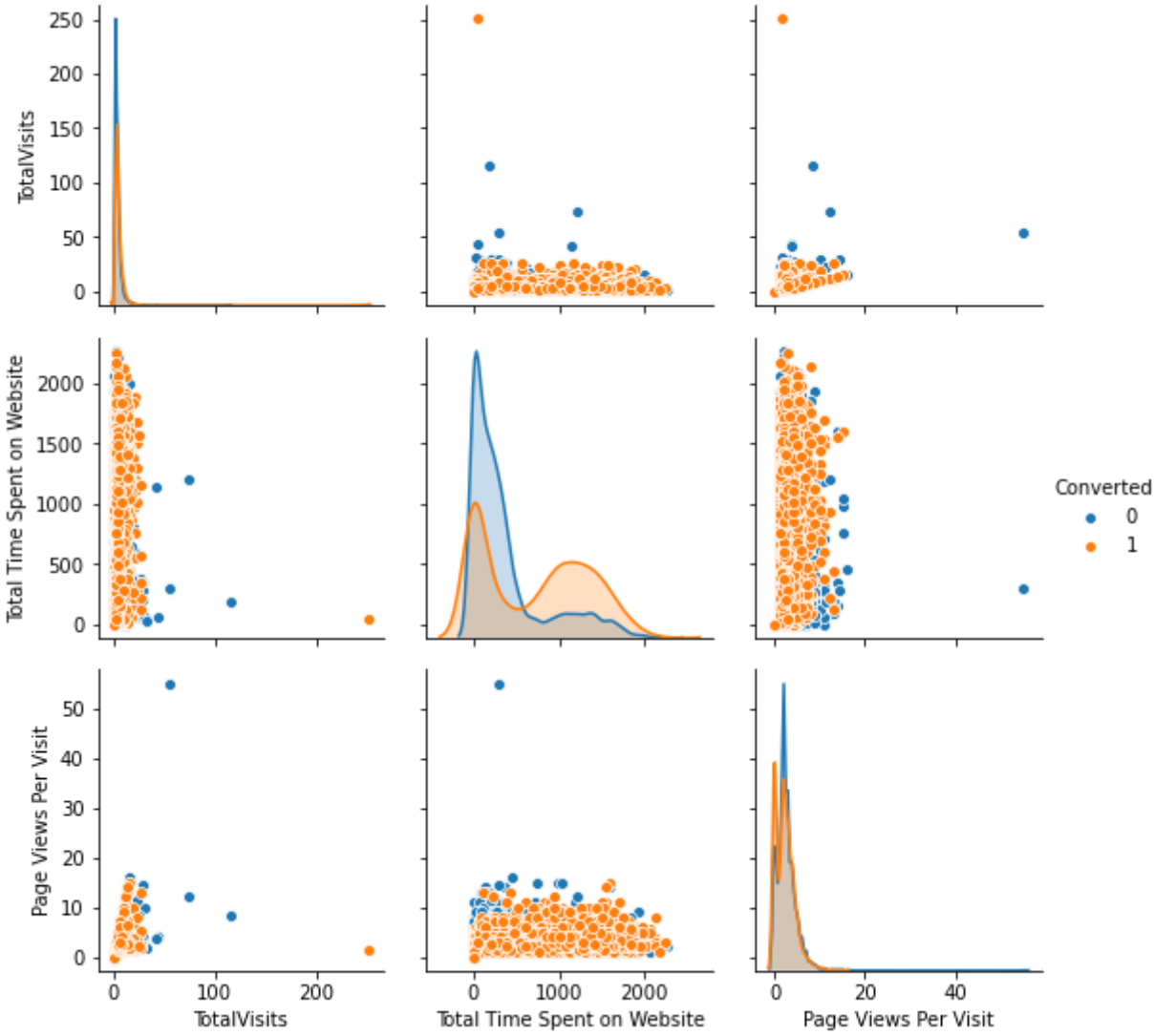
	Lead Origin	Lead Source	Do Not Email	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Specialization	What is your current occupation	Ma In
0	API	Olark Chat	No	0	0.0	0	0.0	Page Visited on Website	Select	Unemployed	
1	API	Organic Search	No	0	5.0	674	2.5	Email Opened	Select	Unemployed	
2	Landing Page Submission	Direct Traffic	No	1	2.0	1532	2.0	Email Opened	Business Administration	Student	
3	Landing Page Submission	Direct Traffic	No	0	1.0	305	1.0	Unreachable	Media and Advertising	Unemployed	
4	Landing Page Submission	Google	No	1	2.0	1428	1.0	Converted to Lead	Select	Unemployed	

Prepare the data for modelling

```
from matplotlib import pyplot as plt
import seaborn as sns
sns.pairplot(xleads,diag_kind='kde',hue='Converted')
plt.show()
```



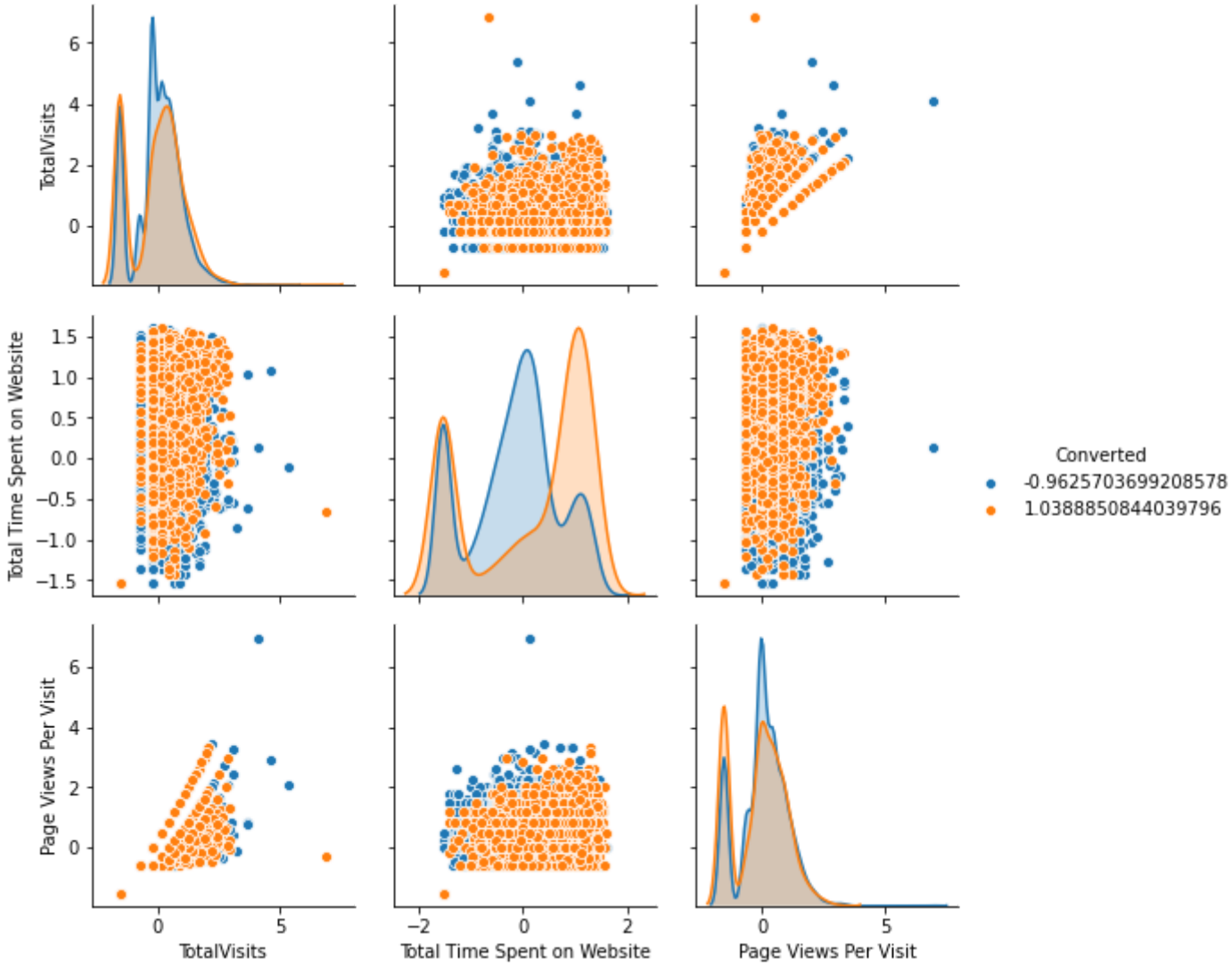

```
xedu = xleads[['TotalVisits','Total Time Spent on Website','Page Views Per Visit','Conv
sns.pairplot(xedu,diag_kind='kde',hue='Converted')
plt.show()
```




```
from sklearn.preprocessing import PowerTransformer
pt = PowerTransformer()
transformedxedu = pd.DataFrame(pt.fit_transform(xedu))
transformedxedu.columns = xedu.columns
transformedxedu.head()
```

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Converted
0	-1.539988	-1.532509	-1.534722	-0.962570
1	0.690854	0.641870	0.230818	-0.962570
2	-0.219742	1.262512	-0.019004	1.038885
3	-0.723932	0.153656	-0.629842	-0.962570
4	-0.219742	1.204175	-0.629842	1.038885

```
sns.pairplot(transformedxedu,diag_kind='kde',hue='Converted')
plt.show()
```



Dummy variable creation

The next step is to deal with the categorical variables present in the dataset. So first take a look at which variables are actually categorical variables.

```
# Check the columns which are of type 'object'

temp = xleads.loc[:, xleads.dtypes == 'object']
temp.columns
```

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

```
# Create dummy variables using the 'get_dummies' command
dummy = pd.get_dummies(xleads[['Lead Origin', 'Lead Source', 'Do Not Email', 'Last Acti
                               'What is your current occupation','A free copy of Masteri
                               'Last Notable Activity']], drop_first=True)

# Add the results to the master dataframe
xleads = pd.concat([xleads, dummy], axis=1)
```

```
# Creating dummy variable separately for the variable 'Specialization' since it has the
# drop that level by specifying it explicitly

dummy_spl = pd.get_dummies(xleads['Specialization'], prefix = 'Specialization')
dummy_spl = dummy_spl.drop(['Specialization_Select'], 1)
xleads = pd.concat([xleads, dummy_spl], axis = 1)
```

```
# Drop the variables for which the dummy variables have been created

xleads = xleads.drop(['Lead Origin', 'Lead Source', 'Do Not Email', 'Last Activity',
                      'Specialization', 'What is your current occupation',
                      'A free copy of Mastering The Interview', 'Last Notable Activity'],
```

```
# Let's take a look at the dataset again

xleads.head()
```

	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Direct Traffic	Lead Source_Facebook	S
0	0	0.0	0	0.0	0	0	0	0	0	
1	0	5.0	674	2.5	0	0	0	0	0	
2	1	2.0	1532	2.0	1	0	0	1	0	
3	0	1.0	305	1.0	1	0	0	1	0	
4	1	2.0	1428	1.0	1	0	0	0	0	

Test-Train Split

The next step is to split the dataset into training an testing sets.

```
# Import the required library

from sklearn.model_selection import train_test_split
```

```
# Put all the feature variables in X

X = xleads.drop(['Converted'], 1)
X.head()
```

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Direct Traffic	Lead Source_Facebook	Lea Source_Goog
0	0.0	0	0.0	0	0	0	0	0	
1	5.0	674	2.5	0	0	0	0	0	
2	2.0	1532	2.0	1	0	0	1	0	

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Direct Traffic	Lead Source_Facebook	Lea Source_Goog
3	1.0	305	1.0	1	0	0	1	0	
4	2.0	1428	1.0	1	0	0	0	0	

```
# Put the target variable in y

y = xleads['Converted']

y.head()
```

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

```
# Split the dataset into 70% train and 30% test

X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, test_size=0.3
```

Scaling

Now there are a few numeric variables present in the dataset which have different scales. So let's go ahead and scale these variables.

```
# Import MinMax scaler

from sklearn.preprocessing import MinMaxScaler
```

```
# Scale the three numeric features present in the dataset

scaler = MinMaxScaler()

X_train[['TotalVisits', 'Page Views Per Visit', 'Total Time Spent on Website']] = scale

X_train.head()
```

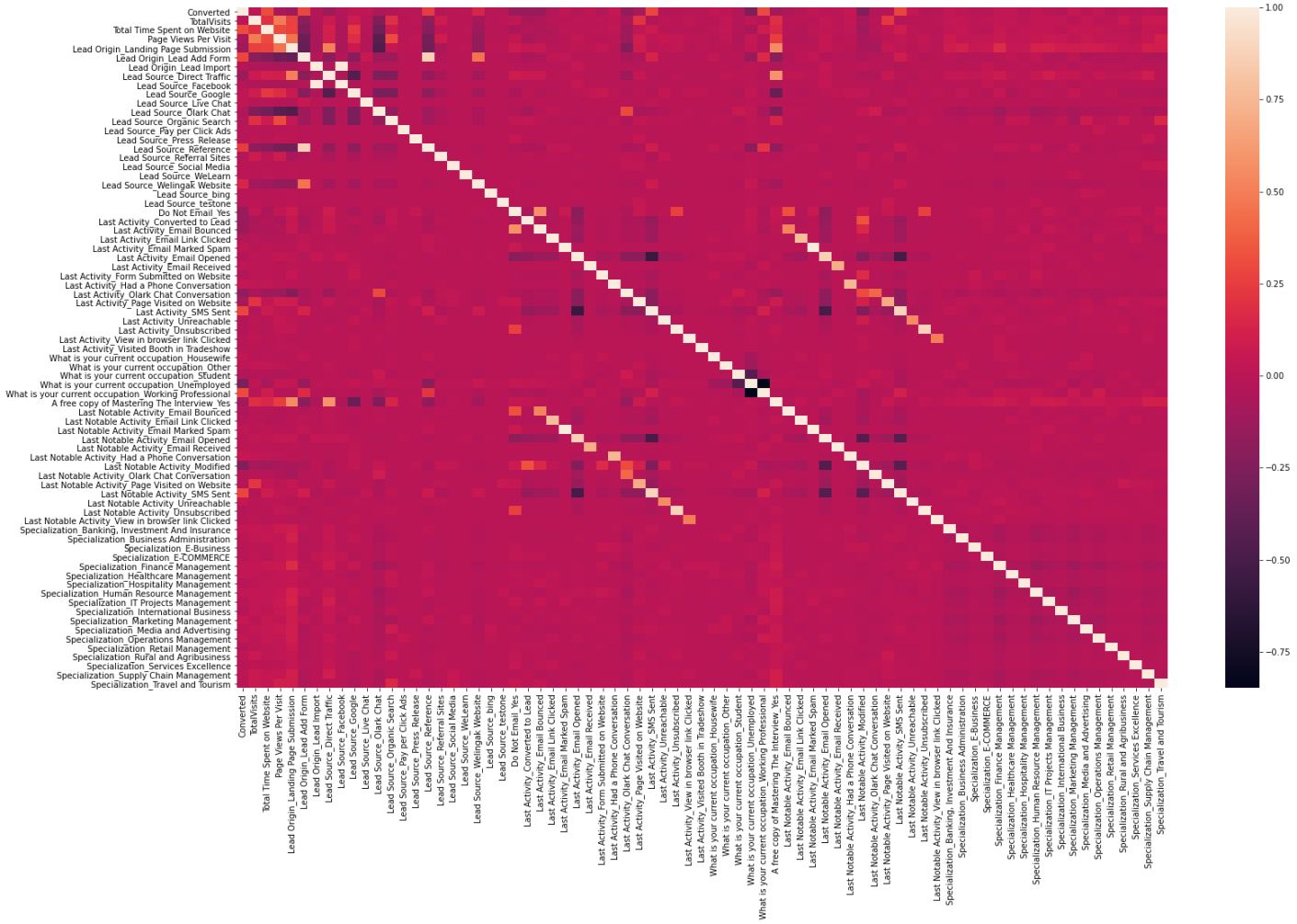
	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Direct Traffic	Lead Source_Facebook	Source_
8003	0.015936	0.029489	0.125	1	0	0	1	0	
218	0.015936	0.082306	0.250	1	0	0	1	0	
4171	0.023904	0.034331	0.375	1	0	0	1	0	
4037	0.000000	0.000000	0.000	0	0	0	0	0	

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Source_Direct Traffic	Lead Source_Facebook	Source_
3660	0.000000	0.000000	0.000	0	1	0	0	0	

Looking at the correlations

Let's now look at the correlations. Since the number of variables are pretty high, it's better that we look at the table instead of plotting a heatmap

```
# Looking at the correlation table
plt.figure(figsize = (25,15))
sns.heatmap(xleads.corr())
plt.show()
```



Model Building

Let's now move to model building. As you can see that there are a lot of variables present in the dataset which we cannot deal with. So the best way to approach this is to select a small set of features from this pool of variables using RFE.

```
# Import 'LogisticRegression' and create a LogisticRegression object

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

```
# Import RFE and select 15 variables

from sklearn.feature_selection import RFE
rfe = RFE(logreg, 15)             # running RFE with 15 variables as output
rfe = rfe.fit(X_train, y_train)
```

```
# Let's take a look at which features have been selected by RFE

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

```
[('TotalVisits', True, 1),
 ('Total Time Spent on Website', True, 1),
 ('Page Views Per Visit', False, 23),
 ('Lead Origin_Landing Page Submission', False, 8),
 ('Lead Origin_Lead Add Form', True, 1),
 ('Lead Origin_Lead Import', False, 52),
 ('Lead Source_Direct Traffic', False, 24),
 ('Lead Source_Facebook', False, 51),
 ('Lead Source_Google', False, 36),
 ('Lead Source_Live Chat', False, 44),
 ('Lead Source_Olark Chat', True, 1),
 ('Lead Source_Organic Search', False, 35),
 ('Lead Source_Pay per Click Ads', False, 43),
 ('Lead Source_Press_Release', False, 53),
 ('Lead Source_Reference', True, 1),
 ('Lead Source_Referral Sites', False, 37),
 ('Lead Source_Social Media', False, 58),
 ('Lead Source_WeLearn', False, 42),
 ('Lead Source_Welingak Website', True, 1),
 ('Lead Source_bing', False, 33),
 ('Lead Source_testone', False, 38),
 ('Do Not Email_Yes', True, 1),
 ('Last Activity_Converted to Lead', False, 25),
 ('Last Activity_Email Bounced', False, 4),
 ('Last Activity_Email Link Clicked', False, 49),
 ('Last Activity_Email Marked Spam', False, 57),
 ('Last Activity_Email Opened', False, 41),
 ('Last Activity_Email Received', False, 54),
 ('Last Activity_Form Submitted on Website', False, 28),
 ('Last Activity_Had a Phone Conversation', True, 1),
 ('Last Activity_Olark Chat Conversation', False, 5),
 ('Last Activity_Page Visited on Website', False, 26),
 ('Last Activity_SMS Sent', True, 1),
 ('Last Activity_Unreachable', False, 47),
 ('Last Activity_Unsubscribed', False, 40),
 ('Last Activity_View in browser link Clicked', False, 34),
 ('Last Activity_Visited Booth in Tradeshow', False, 48),
 ('What is your current occupation_Housewife', True, 1),
```

```
('What is your current occupation_Other', False, 46),
('What is your current occupation_Student', True, 1),
('What is your current occupation_Unemployed', True, 1),
('What is your current occupation_Working Professional', True, 1),
('A free copy of Mastering The Interview_Yes', False, 50),
('Last Notable Activity_Email Bounced', False, 3),
('Last Notable Activity_Email Link Clicked', False, 20),
('Last Notable Activity_Email Marked Spam', False, 59),
('Last Notable Activity_Email Opened', False, 27),
('Last Notable Activity_Email Received', False, 60),
('Last Notable Activity_Had a Phone Conversation', True, 1),
('Last Notable Activity_Modified', False, 2),
('Last Notable Activity_Olark Chat Conversation', False, 32),
('Last Notable Activity_Page Visited on Website', False, 31),
('Last Notable Activity_SMS Sent', False, 45),
('Last Notable Activity_Unreachable', True, 1),
('Last Notable Activity_Unsubscribed', False, 39),
('Last Notable Activity_View in browser link Clicked', False, 29),
('Specialization_Banking, Investment And Insurance', False, 6),
('Specialization_Business Administration', False, 15),
('Specialization_E-Business', False, 11),
('Specialization_E-COMMERCE', False, 9),
('Specialization_Finance Management', False, 14),
('Specialization_Healthcare Management', False, 10),
('Specialization_Hospitality Management', False, 55),
('Specialization_Human Resource Management', False, 16),
('Specialization_IT Projects Management', False, 18),
('Specialization_International Business', False, 22),
('Specialization_Marketing Management', False, 12),
('Specialization_Media and Advertising', False, 21),
('Specialization_Operations Management', False, 19),
('Specialization_Retail Management', False, 30),
('Specialization_Rural and Agribusiness', False, 7),
('Specialization_Services Excellence', False, 56),
('Specialization_Supply Chain Management', False, 13),
('Specialization_Travel and Tourism', False, 17)]
```

```
# Put all the columns selected by RFE in the variable 'col'

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

Now you have all the variables selected by RFE and since we care about the statistics part, i.e. the p-values and the VIFs, let's use these variables to create a logistic regression model using statsmodels.

```
# Select only the columns selected by RFE

X_train = X_train[col]
```

```
# Import statsmodels

import statsmodels.api as sm
```

```
# Fit a logistic Regression model on X_train after adding a constant and output the sum

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

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	4461
Model:	GLM	**Df Residuals:**	4445
Model Family:	Binomial	**Df Model:**	15
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2072.8
Date:	Mon, 23 Nov 2020	**Deviance:**	4145.5
Time:	20:44:21	**Pearson chi2:**	4.84e+03
No. Iterations:	22		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.0061	0.600	-1.677	0.094	-2.182	0.170
TotalVisits	11.3439	2.682	4.230	0.000	6.088	16.600
Total Time Spent on Website	4.4312	0.185	23.924	0.000	4.068	4.794
Lead Origin_Lead Add Form	2.9483	1.191	2.475	0.013	0.614	5.283
Lead Source_Olark Chat	1.4584	0.122	11.962	0.000	1.219	1.697
Lead Source_Reference	1.2994	1.214	1.070	0.285	-1.080	3.679
Lead Source_Welingak Website	3.4159	1.558	2.192	0.028	0.362	6.470
Do Not Email_Yes	-1.5053	0.193	-7.781	0.000	-1.884	-1.126
Last Activity_Had a Phone Conversation	1.0397	0.983	1.058	0.290	-0.887	2.966
Last Activity_SMS Sent	1.1827	0.082	14.362	0.000	1.021	1.344
What is your current occupation_Housewife	22.6492	2.45e+04	0.001	0.999	-4.8e+04	4.8e+04
What is your current occupation_Student	-1.1544	0.630	-1.831	0.067	-2.390	0.081
What is your current occupation_Unemployed	-1.3395	0.594	-2.254	0.024	-2.505	-0.175
What is your current occupation_Working Professional	1.2743	0.623	2.045	0.041	0.053	2.496
Last Notable Activity_Had a Phone Conversation	23.1932	2.08e+04	0.001	0.999	-4.08e+04	4.08e+04
Last Notable Activity_Unreachable	2.7868	0.807	3.453	0.001	1.205	4.369

There are quite a few variable which have a p-value greater than `0.05`. We will need to take care of them. But first, let's also look at the VIFs.

```
# Import 'variance_inflation_factor'

from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
# Make a VIF dataframe for all the variables present

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

	Features	VIF
2	Lead Origin_Lead Add Form	84.19
4	Lead Source_Reference	65.18
5	Lead Source_Welingak Website	20.03
11	What is your current occupation_Unemployed	3.65
7	Last Activity_Had a Phone Conversation	2.44
13	Last Notable Activity_Had a Phone Conversation	2.43
1	Total Time Spent on Website	2.38
0	TotalVisits	1.62
8	Last Activity_SMS Sent	1.59
12	What is your current occupation_Working Profes...	1.56
3	Lead Source_Olark Chat	1.44
6	Do Not Email_Yes	1.09
10	What is your current occupation_Student	1.09
9	What is your current occupation_Housewife	1.01
14	Last Notable Activity_Unreachable	1.01

VIFs seem to be in a decent range except for three variables.

Let's first drop the variable `Lead Source_Reference` since it has a high p-value as well as a high VIF.

```
X_train.drop('Lead Source_Reference', axis = 1, inplace = True)
```

```
# Refit the model with the new set of features

logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
logm1.fit().summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	4461
Model:	GLM	**Df Residuals:**	4446
Model Family:	Binomial	**Df Model:**	14

Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2073.2
Date:	Mon, 23 Nov 2020	**Deviance:**	4146.5
Time:	20:45:45	**Pearson chi2:**	4.82e+03
No. Iterations:	22		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.0057	0.600	-1.677	0.094	-2.181	0.170
TotalVisits	11.3428	2.682	4.229	0.000	6.086	16.599
Total Time Spent on Website	4.4312	0.185	23.924	0.000	4.068	4.794
Lead Origin_Lead Add Form	4.2084	0.259	16.277	0.000	3.702	4.715
Lead Source_Olark Chat	1.4583	0.122	11.960	0.000	1.219	1.697
Lead Source_Welingak Website	2.1557	1.037	2.079	0.038	0.124	4.188
Do Not Email_Yes	-1.5036	0.193	-7.779	0.000	-1.882	-1.125
Last Activity_Had a Phone Conversation	1.0398	0.983	1.058	0.290	-0.887	2.966
Last Activity_SMS Sent	1.1827	0.082	14.362	0.000	1.021	1.344
What is your current occupation_Housewife	22.6511	2.45e+04	0.001	0.999	-4.8e+04	4.8e+04
What is your current occupation_Student	-1.1537	0.630	-1.830	0.067	-2.389	0.082
What is your current occupation_Unemployed	-1.3401	0.594	-2.255	0.024	-2.505	-0.175
What is your current occupation_Working Professional	1.2748	0.623	2.046	0.041	0.053	2.496
Last Notable Activity_Had a Phone Conversation	23.1934	2.08e+04	0.001	0.999	-4.08e+04	4.08e+04
Last Notable Activity_Unreachable	2.7872	0.807	3.454	0.001	1.205	4.369

The variable `Lead Profile_Dual Specialization Student` also needs to be dropped.

```
# Make a VIF dataframe for all the variables present

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

	Features	VIF
10	What is your current occupation_Unemployed	3.65
6	Last Activity_Had a Phone Conversation	2.44
12	Last Notable Activity_Had a Phone Conversation	2.43
1	Total Time Spent on Website	2.38
2	Lead Origin_Lead Add Form	1.71
0	TotalVisits	1.62
7	Last Activity_SMS Sent	1.59

	Features	VIF
11	What is your current occupation_Working Profes...	1.56
3	Lead Source_Olark Chat	1.44
4	Lead Source_Welingak Website	1.33
5	Do Not Email_Yes	1.09
9	What is your current occupation_Student	1.09
8	What is your current occupation_Housewife	1.01
13	Last Notable Activity_Unreachable	1.01

The VIFs are now all less than 5. So let's drop the ones with the high p-values beginning with `Last Notable Activity_Had a Phone Conversation` .

```
X_train.drop('Last Notable Activity_Had a Phone Conversation', axis = 1, inplace = True
```

```
# Refit the model with the new set of features

logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
logm1.fit().summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	4461
Model:	GLM	**Df Residuals:**	4447
Model Family:	Binomial	**Df Model:**	13
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2076.1
Date:	Mon, 23 Nov 2020	**Deviance:**	4152.2
Time:	20:46:57	**Pearson chi2:**	4.82e+03
No. Iterations:	21		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-1.0069	0.600	-1.679	0.093	-2.182	0.168
TotalVisits	11.4551	2.686	4.265	0.000	6.191	16.720
Total Time Spent on Website	4.4237	0.185	23.900	0.000	4.061	4.787
Lead Origin_Lead Add Form	4.2082	0.259	16.276	0.000	3.701	4.715
Lead Source_Olark Chat	1.4581	0.122	11.958	0.000	1.219	1.697
Lead Source_Welingak Website	2.1557	1.037	2.079	0.038	0.124	4.188
Do Not Email_Yes	-1.5037	0.193	-7.780	0.000	-1.882	-1.125
Last Activity_Had a Phone Conversation	2.7502	0.802	3.430	0.001	1.179	4.322
Last Activity_SMS Sent	1.1826	0.082	14.364	0.000	1.021	1.344
What is your current occupation_Housewife	21.6525	1.49e+04	0.001	0.999	-2.91e+04	2.91e+04
What is your current occupation_Student	-1.1520	0.630	-1.828	0.068	-2.387	0.083
What is your current occupation_Unemployed	-1.3385	0.594	-2.253	0.024	-2.503	-0.174

What is your current occupation_Working Professional	1.2743	0.623	2.045	0.041	0.053	2.495
Last Notable Activity_Unreachable	2.7862	0.807	3.453	0.001	1.205	4.368

Drop What is your current occupation_Housewife .

```
X_train.drop('What is your current occupation_Housewife', axis = 1, inplace = True)
```

```
# Refit the model with the new set of features

logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
logm1.fit().summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	4461
Model:	GLM	**Df Residuals:**	4448
Model Family:	Binomial	**Df Model:**	12
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2078.3
Date:	Mon, 23 Nov 2020	**Deviance:**	4156.7
Time:	20:47:36	**Pearson chi2:**	4.83e+03
No. Iterations:	7		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.4528	0.554	-0.818	0.413	-1.538	0.632
TotalVisits	11.2586	2.672	4.214	0.000	6.023	16.495
Total Time Spent on Website	4.4217	0.185	23.898	0.000	4.059	4.784
Lead Origin_Lead Add Form	4.2057	0.258	16.274	0.000	3.699	4.712
Lead Source_Olark Chat	1.4530	0.122	11.930	0.000	1.214	1.692
Lead Source_Welingak Website	2.1541	1.037	2.078	0.038	0.122	4.186
Do Not Email_Yes	-1.5063	0.193	-7.785	0.000	-1.886	-1.127
Last Activity_Had a Phone Conversation	2.7515	0.802	3.432	0.001	1.180	4.323
Last Activity_SMS Sent	1.1823	0.082	14.362	0.000	1.021	1.344
What is your current occupation_Student	-1.7017	0.588	-2.893	0.004	-2.855	-0.549
What is your current occupation_Unemployed	-1.8879	0.550	-3.435	0.001	-2.965	-0.811
What is your current occupation_Working Professional	0.7246	0.581	1.248	0.212	-0.413	1.862
Last Notable Activity_Unreachable	2.7834	0.807	3.448	0.001	1.201	4.365

Drop What is your current occupation_Working Professional .

```
X_train.drop('What is your current occupation_Working Professional', axis = 1, inplace
```

```
# Refit the model with the new set of features

logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
res = logm1.fit()
res.summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	4461
Model:	GLM	**Df Residuals:**	4449
Model Family:	Binomial	**Df Model:**	11
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2079.1
Date:	Mon, 23 Nov 2020	**Deviance:**	4158.1
Time:	20:48:17	**Pearson chi2:**	4.80e+03
No. Iterations:	7		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.2040	0.196	1.043	0.297	-0.179	0.587
TotalVisits	11.1489	2.665	4.184	0.000	5.926	16.371
Total Time Spent on Website	4.4223	0.185	23.899	0.000	4.060	4.785
Lead Origin_Lead Add Form	4.2051	0.258	16.275	0.000	3.699	4.712
Lead Source_Olark Chat	1.4526	0.122	11.934	0.000	1.214	1.691
Lead Source_Welingak Website	2.1526	1.037	2.076	0.038	0.121	4.185
Do Not Email_Yes	-1.5037	0.193	-7.774	0.000	-1.883	-1.125
Last Activity_Had a Phone Conversation	2.7552	0.802	3.438	0.001	1.184	4.326
Last Activity_SMS Sent	1.1856	0.082	14.421	0.000	1.024	1.347
What is your current occupation_Student	-2.3578	0.281	-8.392	0.000	-2.908	-1.807
What is your current occupation_Unemployed	-2.5445	0.186	-13.699	0.000	-2.908	-2.180
Last Notable Activity_Unreachable	2.7846	0.807	3.449	0.001	1.202	4.367

All the p-values are now in the appropriate range. Let's also check the VIFs again in case we had missed something.

```
# Make a VIF dataframe for all the variables present

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

	Features	VIF
9	What is your current occupation_Unemployed	2.82

	Features	VIF
1	Total Time Spent on Website	2.00
0	TotalVisits	1.54
7	Last Activity_SMS Sent	1.51
2	Lead Origin_Lead Add Form	1.45
3	Lead Source_Olark Chat	1.33
4	Lead Source_Welingak Website	1.30
5	Do Not Email_Yes	1.08
8	What is your current occupation_Student	1.06
6	Last Activity_Had a Phone Conversation	1.01
10	Last Notable Activity_Unreachable	1.01

Model Evaluation

Now, both the p-values and VIFs seem decent enough for all the variables. So let's go ahead and make predictions using this final set of features.

```
# Use 'predict' to predict the probabilities on the train set

y_train_pred = res.predict(sm.add_constant(X_train))
y_train_pred[:10]
```

```
8003    0.300117
218     0.142002
4171    0.127629
4037    0.291558
3660    0.954795
207     0.194426
2044    0.178073
6411    0.949460
6498    0.075995
2085    0.982316
dtype: float64
```

```
# Reshaping it into an array

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

```
array([0.30011695, 0.14200165, 0.12762885, 0.29155814, 0.95479546,
       0.19442563, 0.17807328, 0.94946006, 0.07599465, 0.98231619])
```

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

```
# Create a new dataframe containing the actual conversion flag and the probabilities pr
```

```
y_train_pred_final = pd.DataFrame({'Converted':y_train.values, 'Conversion_Prob':y_trai
y_train_pred_final.head()
```

	Converted	Conversion_Prob
0	0	0.300117
1	0	0.142002
2	1	0.127629
3	1	0.291558
4	1	0.954795

Creating new column 'Predicted' with 1 if Paid_Prob > 0.5 else 0

```
y_train_pred_final['Predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x: 1 if

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

	Converted	Conversion_Prob	Predicted
0	0	0.300117	0
1	0	0.142002	0
2	1	0.127629	0
3	1	0.291558	0
4	1	0.954795	1

Now that you have the probabilities and have also made conversion predictions using them, it's time to evaluate the model.

```
# Import metrics from sklearn for evaluation

from sklearn import metrics
```

```
# Create confusion matrix

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

```
[[1929  383]
 [ 560 1589]]
```

```
# Predicted     not_churn    churn
# Actual
# not_churn        2543      463
# churn            692       1652
```

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

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

```
0.7886124187401928
```

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

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

```
# Calculate the sensitivity

TP/(TP+FN)
```

```
0.739413680781759
```

```
# Calculate the specificity

TN/(TN+FP)
```

```
0.8343425605536332
```

Finding the Optimal Cutoff

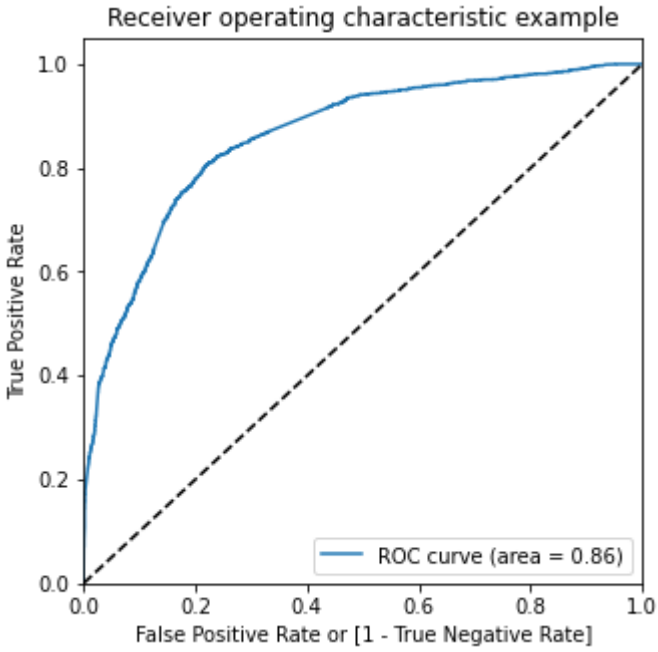
Now 0.5 was just arbitrary to loosely check the model performace. But in order to get good results, you need to optimise the threshold. So first let's plot an ROC curve to see what AUC we get.

```
# ROC function

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 None
```

```
fpr, tpr, thresholds = metrics.roc_curve( y_train_pred_final.Converted, y_train_pred_fi
```

```
# Import matplotlib to plot the ROC curve

import matplotlib.pyplot as plt
```

```
# Call the ROC function

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



The area under the curve of the ROC is 0.86 which is quite good. So we seem to have a good model. Let's also check the sensitivity and specificity tradeoff to find the optimal cutoff point.

```
# Let's create columns with different probability cutoffs

numbers = [float(x)/10 for x in range(10)]
for i in numbers:
    y_train_pred_final[i]= y_train_pred_final.Conversion_Prob.map(lambda x: 1 if x > i
y_train_pred_final.head()
```

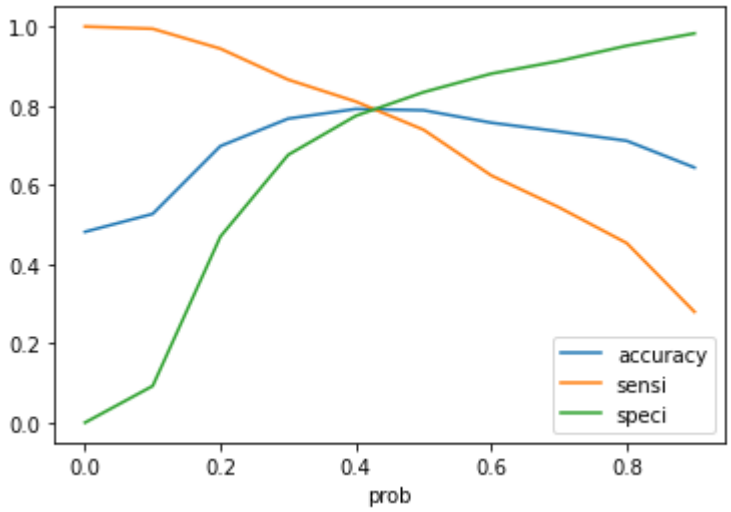
	Converted	Conversion_Prob	Predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	0	0.300117	0	1	1	1	1	0	0	0	0	0	0
1	0	0.142002	0	1	1	0	0	0	0	0	0	0	0
2	1	0.127629	0	1	1	0	0	0	0	0	0	0	0
3	1	0.291558	0	1	1	1	0	0	0	0	0	0	0
4	1	0.954795	1	1	1	1	1	1	1	1	1	1	1

```
# Let's create a dataframe to see the values of accuracy, sensitivity, and specificity

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	0.481731	1.000000	0.000000
0.1	0.1	0.527012	0.994416	0.092561
0.2	0.2	0.698274	0.944160	0.469723
0.3	0.3	0.767541	0.865984	0.676038
0.4	0.4	0.791975	0.810610	0.774654
0.5	0.5	0.788612	0.739414	0.834343
0.6	0.6	0.757229	0.624011	0.881055
0.7	0.7	0.735037	0.543509	0.913062
0.8	0.8	0.711500	0.453234	0.951557
0.9	0.9	0.644026	0.279665	0.982699

```
# Let's plot it as well

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



As you can see that around 0.42, you get the optimal values of the three metrics. So let's choose 0.42 as our cutoff now.

```
y_train_pred_final['final_predicted'] = y_train_pred_final.Conversion_Prob.map( lambda

y_train_pred_final.head()
```

	Converted	Conversion_Prob	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	0.300117	0	1	1	1	1	0	0	0	0	0	0	0
1	0	0.142002	0	1	1	0	0	0	0	0	0	0	0	0
2	1	0.127629	0	1	1	0	0	0	0	0	0	0	0	0
3	1	0.291558	0	1	1	1	0	0	0	0	0	0	0	0
4	1	0.954795	1	1	1	1	1	1	1	1	1	1	1	1

```
# Let's check the accuracy now

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

```
0.7908540685944856
```

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

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

```
array([[1823,  489],
       [ 444, 1705]], dtype=int64)
```

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

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

```
# Calculate Sensitivity

TP/(TP+FN)
```

```
0.793392275476966
```

```
# Calculate Specificity

TN/(TN+FP)
```

```
0.7884948096885813
```

This cutoff point seems good to go!

Making Predictions on the Test Set

Let's now make predicitons on the test set.

```
# Scale the test set as well using just 'transform'

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

```
# Select the columns in X_train for X_test as well

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

	TotalVisits	Total Time Spent on Website	Lead Origin_Lead Add Form	Lead Source_Olark Chat	Lead Source_Reference	Lead Source_Welingak Website	Do Not Email_Yes	Last Activity_Had a Phone Conversation	A
4771	0.000000	0.000000	1	0	1	0	0	0	
6122	0.027888	0.029049	0	0	0	0	0	0	
9202	0.015936	0.416813	0	0	0	0	0	0	
6570	0.011952	0.378961	0	0	0	0	1	0	
2668	0.031873	0.395246	0	0	0	0	0	0	

```
# Add a constant to X_test

X_test_sm = sm.add_constant(X_test[col])
```

```
# Check X_test_sm

X_test_sm
```

	const	TotalVisits	Total Time Spent on Website	Lead Origin_Lead Add Form	Lead Source_Olark Chat	Lead Source_Reference	Lead Source_Welingak Website	Do Not Email_Yes	Activity a Pl Conversa
4771	1.0	0.000000	0.000000	1	0	1	0	0	
6122	1.0	0.027888	0.029049	0	0	0	0	0	
9202	1.0	0.015936	0.416813	0	0	0	0	0	
6570	1.0	0.011952	0.378961	0	0	0	0	1	
2668	1.0	0.031873	0.395246	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	
5828	1.0	0.011952	0.027289	0	0	0	0	0	
6583	1.0	0.011952	0.152289	0	0	0	0	0	
5531	1.0	0.055777	0.702025	0	0	0	0	0	
3056	1.0	0.011952	0.417694	0	0	0	0	1	
4088	1.0	0.019920	0.530370	0	0	0	0	0	

1912 rows × 16 columns

```
# Drop the required columns from X_test as well

X_test.drop(['Lead Source_Reference', 'What is your current occupation_Housewife',
             'What is your current occupation_Working Professional', 'Last Notable Acti
```

```
# Make predictions on the test set and store it in the variable 'y_test_pred'

y_test_pred = res.predict(sm.add_constant(X_test))
```

```
y_test_pred[:10]
```

```
4771    0.996296
6122    0.129992
9202    0.703937
6570    0.299564
2668    0.720796
4233    0.792250
3368    0.704038
9091    0.464521
5972    0.282978
3631    0.786460
dtype: float64
```

```
# Converting y_pred to a dataframe

y_pred_1 = pd.DataFrame(y_test_pred)
```

```
# Let's see the head

y_pred_1.head()
```

	0
4771	0.996296
6122	0.129992
9202	0.703937
6570	0.299564
2668	0.720796

```
# Converting y_test to dataframe

y_test_df = pd.DataFrame(y_test)
```

```
# Remove index for both dataframes to append them side by side

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

```
# Append y_test_df and y_pred_1

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

```
# Check 'y_pred_final'

y_pred_final.head()
```

	Converted	0
0	1	0.996296
1	0	0.129992
2	0	0.703937
3	1	0.299564
4	1	0.720796

```
# Rename the column

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

```
# Let's see the head of y_pred_final

y_pred_final.head()
```

	Converted	Conversion_Prob
0	1	0.996296
1	0	0.129992
2	0	0.703937
3	1	0.299564
4	1	0.720796

```
# Make predictions on the test set using 0.45 as the cutoff

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

```
# Check y_pred_final

y_pred_final.head()
```

	Converted	Conversion_Prob	final_predicted
0	1	0.996296	1
1	0	0.129992	0
2	0	0.703937	1
3	1	0.299564	0

	Converted	Conversion_Prob	final_predicted
4	1	0.720796	1

```
# Let's check the overall accuracy

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

```
0.7845188284518828
```

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

```
array([[786, 210],
       [202, 714]], dtype=int64)
```

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

```
# Calculate sensitivity
TP / float(TP+FN)
```

```
0.7794759825327511
```

```
# Calculate specificity
TN / float(TN+FP)
```

```
0.7891566265060241
```

Precision-Recall View

Let's now also build the training model using the precision-recall view

```
#Looking at the confusion matrix again

confusion = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.P
confusion
```

```
array([[1929,  383],
       [ 560, 1589]], dtype=int64)
```

Precision

TP / TP + FP

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

```
0.8057809330628803
```

Recall

TP / TP + FN

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

```
0.739413680781759
```

Precision and recall tradeoff

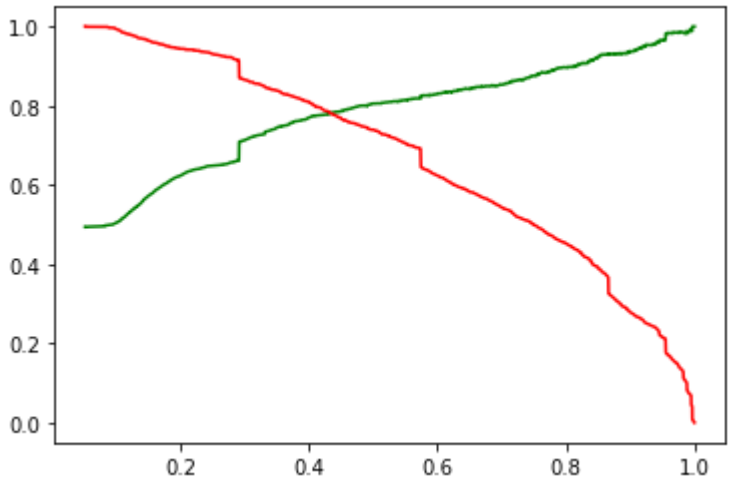
```
from sklearn.metrics import precision_recall_curve
```

```
y_train_pred_final.Converted, y_train_pred_final.Predicted
```

```
(0       0
 1       0
 2       1
 3       1
 4       1
        ..
 4456    1
 4457    0
 4458    0
 4459    0
 4460    0
 Name: Converted, Length: 4461, dtype: int64,
 0       0
 1       0
 2       0
 3       0
 4       1
        ..
 4456    1
 4457    1
 4458    1
 4459    0
 4460    0
 Name: Predicted, Length: 4461, dtype: int64)
```

```
p, r, thresholds = precision_recall_curve(y_train_pred_final.Converted, y_train_pred_fi
```

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




```
y_train_pred_final['final_predicted'] = y_train_pred_final.Conversion_Prob.map(lambda x

y_train_pred_final.head()
```

	Converted	Conversion_Prob	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	0.300117	0	1	1	1	1	0	0	0	0	0	0	0
1	0	0.142002	0	1	1	0	0	0	0	0	0	0	0	0
2	1	0.127629	0	1	1	0	0	0	0	0	0	0	0	0
3	1	0.291558	0	1	1	1	0	0	0	0	0	0	0	0
4	1	0.954795	1	1	1	1	1	1	1	1	1	1	1	1

```
# Let's check the accuracy now

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

```
0.7895090786819099
```

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

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

```
array([[1852,  460],
       [ 479, 1670]], dtype=int64)
```

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

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

```
# Calculate Precision

TP/(TP+FP)
```

```
0.784037558685446
```

```
# Calculate Recall

TP/(TP+FN)
```

```
0.7771056305258259
```

This cutoff point seems good to go!

Making Predictions on the Test Set

Let's now make predicitons on the test set.

```
# Make predictions on the test set and store it in the variable 'y_test_pred'

y_test_pred = res.predict(sm.add_constant(X_test))
```

```
y_test_pred[:10]
```

```
4771    0.996296
6122    0.129992
9202    0.703937
6570    0.299564
2668    0.720796
4233    0.792250
3368    0.704038
9091    0.464521
5972    0.282978
3631    0.786460
dtype: float64
```

```
# Converting y_pred to a dataframe

y_pred_1 = pd.DataFrame(y_test_pred)
```

```
# Let's see the head

y_pred_1.head()
```

	0
4771	0.996296
6122	0.129992
9202	0.703937
6570	0.299564
2668	0.720796

```
# Converting y_test to dataframe

y_test_df = pd.DataFrame(y_test)
```

```
# Remove index for both dataframes to append them side by side

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

```
# Append y_test_df and y_pred_1

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

```
# Check 'y_pred_final'

y_pred_final.head()
```

	Converted	0
0	1	0.996296
1	0	0.129992
2	0	0.703937
3	1	0.299564
4	1	0.720796

```
# Rename the column

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

```
# Let's see the head of y_pred_final

y_pred_final.head()
```

	Converted	Conversion_Prob
0	1	0.996296
1	0	0.129992
2	0	0.703937
3	1	0.299564
4	1	0.720796

```
# Make predictions on the test set using 0.44 as the cutoff

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

```
# Check y_pred_final

y_pred_final.head()
```

	Converted	Conversion_Prob	final_predicted
0	1	0.996296	1
1	0	0.129992	0
2	0	0.703937	1
3	1	0.299564	0
4	1	0.720796	1

```
# Let's check the overall accuracy

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

```
0.7866108786610879
```

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

```
array([[801, 195],
       [213, 703]], dtype=int64)
```

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

```
# Calculate Precision

TP/(TP+FP)
```

```
0.7828507795100222
```

```
# Calculate Recall

TP/(TP+FN)
```

```
0.767467248908297
```

Summary

There are a lot of leads generated in the initial stage (top) but only a few of them come out as paying customers from the bottom. In the middle stage, you need to nurture the potential leads well (i.e. educating the leads about the product, constantly communicating etc.) in order to get a higher lead conversion.
First, sort out the best prospects from the leads you have generated. 'TotalVisits' , 'Total Time Spent on Website' , 'Page Views Per Visit' which contribute most towards the probability of a lead getting converted.

Then, You must keep a list of leads handy so that you can inform them about new courses, services, job offers and future higher studies. Monitor each lead carefully so that you can tailor the information you send to them. Carefully provide job offerings, information or courses that suits best according to the interest of the leads. A proper plan to chart the needs of each lead will go a long way to capture the leads as prospects.
Focus on converted leads. Hold question-answer sessions with leads to extract the right information you need about them. Make further inquiries and appointments with the leads to determine their intention and mentality to join online courses.