Lead case study using Decision Trees

Step1: Reading and Understanding the data

In [1]:

```
#Supressing the warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
#Importing the Numpy and Pandas
import numpy as np
import pandas as pd
```

In [3]:

```
#Importing the data and see the head of our dataset
lead=pd.read_csv('E:\Leads.csv')

lead.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	A	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

In [4]:

```
#Let's check the dimensions of our dataframe
lead.shape
```

Out[4]:

```
(9240, 37)
```

In [5]:

```
#Let's look at the sttistical aspects of our dataframe
lead.describe()
```

Out[5]:

	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

In [6]:

```
#Let's see the type of each column
lead.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9240 entries, 0 to 9239
Data columns (total 37 columns):
Prospect ID                                    9240 non-null object
Lead Number                                    9240 non-null int64
Lead Origin                                    9240 non-null object
Lead Source                                    9204 non-null object
Do Not Email                                   9240 non-null object
Do Not Call                                    9240 non-null object
Converted                                      9240 non-null int64
TotalVisits                                    9103 non-null float64
Total Time Spent on Website                    9240 non-null int64
Page Views Per Visit                           9103 non-null float64
Last Activity                                  9137 non-null object
Country                                        6779 non-null object
Specialization                                 7802 non-null object
How did you hear about X Education             7033 non-null object
What is your current occupation                6550 non-null object
What matters most to you in choosing a course  6531 non-null object
Search                                         9240 non-null object
Magazine                                       9240 non-null object
Newspaper Article                              9240 non-null object
X Education Forums                             9240 non-null object
Newspaper                                      9240 non-null object
Digital Advertisement                          9240 non-null object
Through Recommendations                        9240 non-null object
Receive More Updates About Our Courses         9240 non-null object
Tags                                           5887 non-null object
Lead Quality                                   4473 non-null object
Update me on Supply Chain Content              9240 non-null object
Get updates on DM Content                      9240 non-null object
Lead Profile                                   6531 non-null object
City                                           7820 non-null object
Asymmetrique Activity Index                    5022 non-null object
Asymmetrique Profile Index                     5022 non-null object
Asymmetrique Activity Score                    5022 non-null float64
Asymmetrique Profile Score                     5022 non-null float64
I agree to pay the amount through cheque       9240 non-null object
A free copy of Mastering The Interview         9240 non-null object
Last Notable Activity                          9240 non-null object
dtypes: float64(4), int64(3), object(30)
memory usage: 2.6+ MB
```

In [7]:

```
# To check the sum of missing values
lead.isnull().sum()
```

Out[7]:

```
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
```

Step2:Data Cleaning

In [8]:

```
# Convert Select to nan
lead=lead.replace('Select',np.nan)
```

In [9]:

```
#Adding up the missing values(Column wise)
lead.isnull().sum()
```

Out[9]:

```
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                                 3380
How did you hear about X Education             7250
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                                  6855
City                                          3669
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
```

In [10]:

```
#Checking the percentage of missing values
round(100*(lead.isnull().sum()/len(lead.index)),2)
```

Out[10]:

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

In [11]:

```
#Dropping the column which has 70% greater than the nan values i.e. Lead Profile
lead=lead.drop('Lead Profile',1)
```

In [12]:

```
lead=lead.drop('How did you hear about X Education',1)
```

In [13]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [14]:

```
lead['Country'].describe()
```

Out[14]:

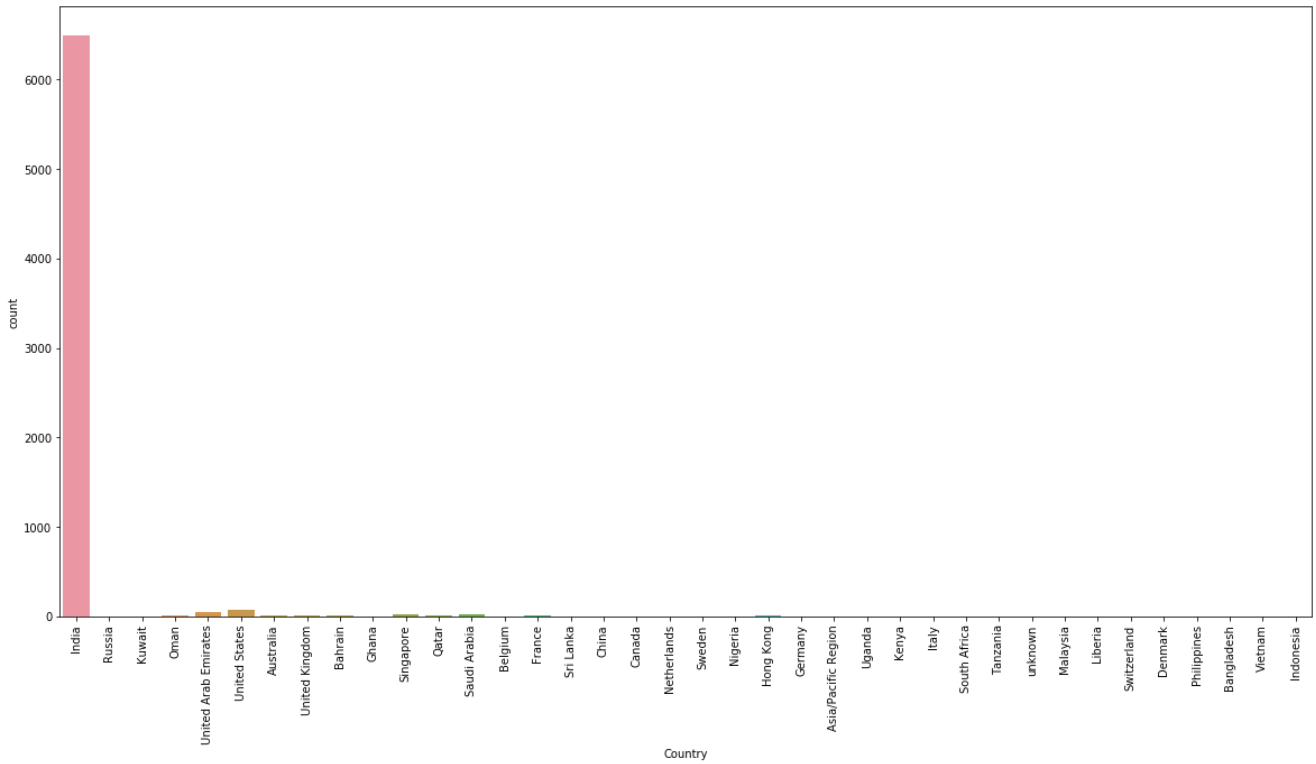
```
count       6779
unique        38
top        India
freq        6492
Name: Country, dtype: object
```

In [15]:

```
plt.figure(figsize=(20,10))
sns.countplot(lead['Country'])
plt.xticks(rotation=90)
```

Out[15]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37]), <a list of 38 Text xticklabel objects>)
```



In [16]:

```
lead['Country']=lead['Country'].replace(np.nan,'India')
```

In [17]:

```
lead['Specialization'].describe()
```

Out[17]:

```
count                   5860
unique                    18
```

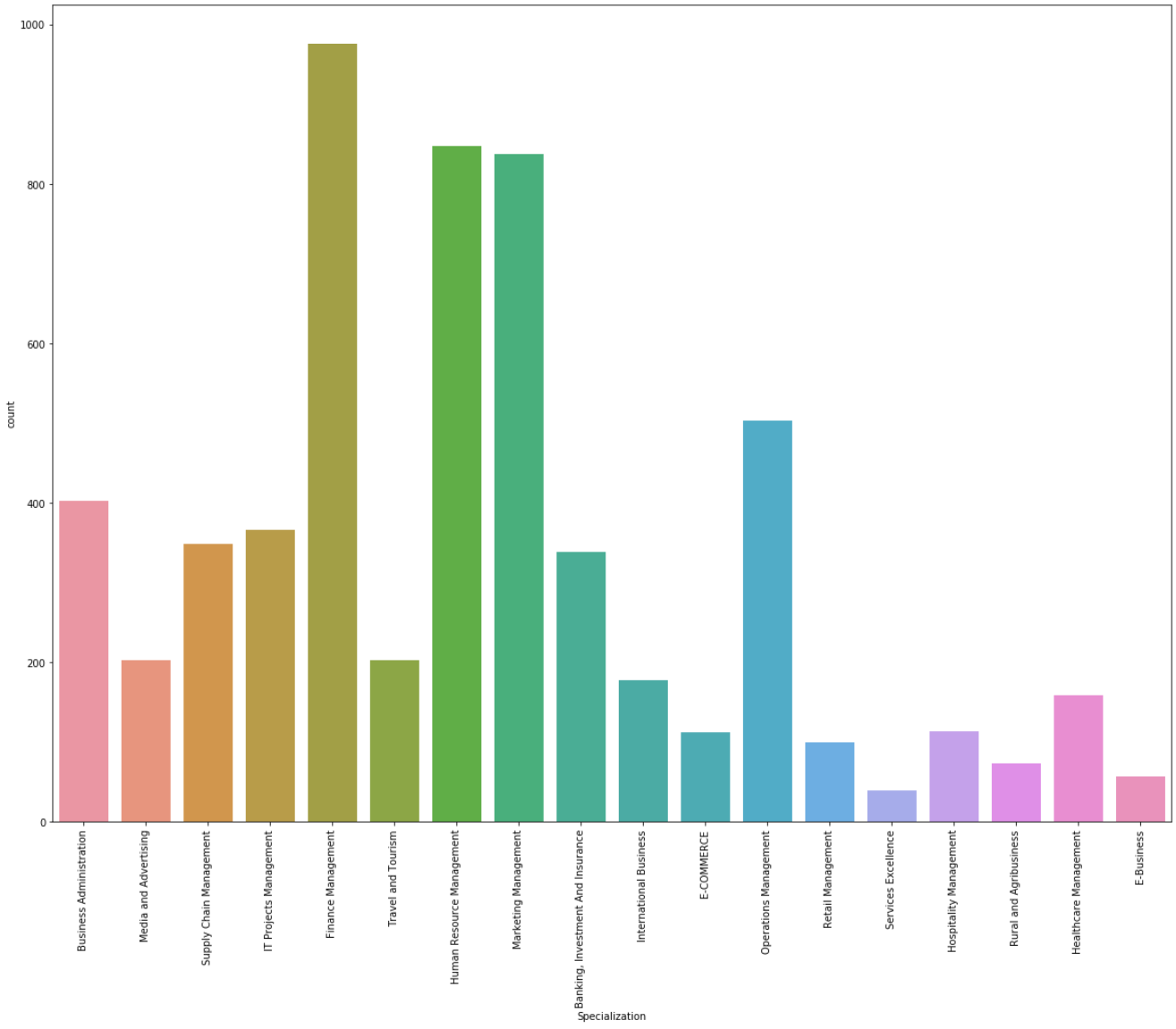
```
top       Finance Management
freq                     976
Name: Specialization, dtype: object
```

In [18]:

```
plt.figure(figsize=(20,15))
sns.countplot(lead['Specialization'])
plt.xticks(rotation=90)
```

Out[18]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17]), <a list of 18 Text xticklabel objects>)
```



In [19]:

```
lead['Specialization']=lead['Specialization'].replace(np.nan,'Others')
```

In [20]:

```
lead['What is your current occupation'].describe()
```

Out[20]:

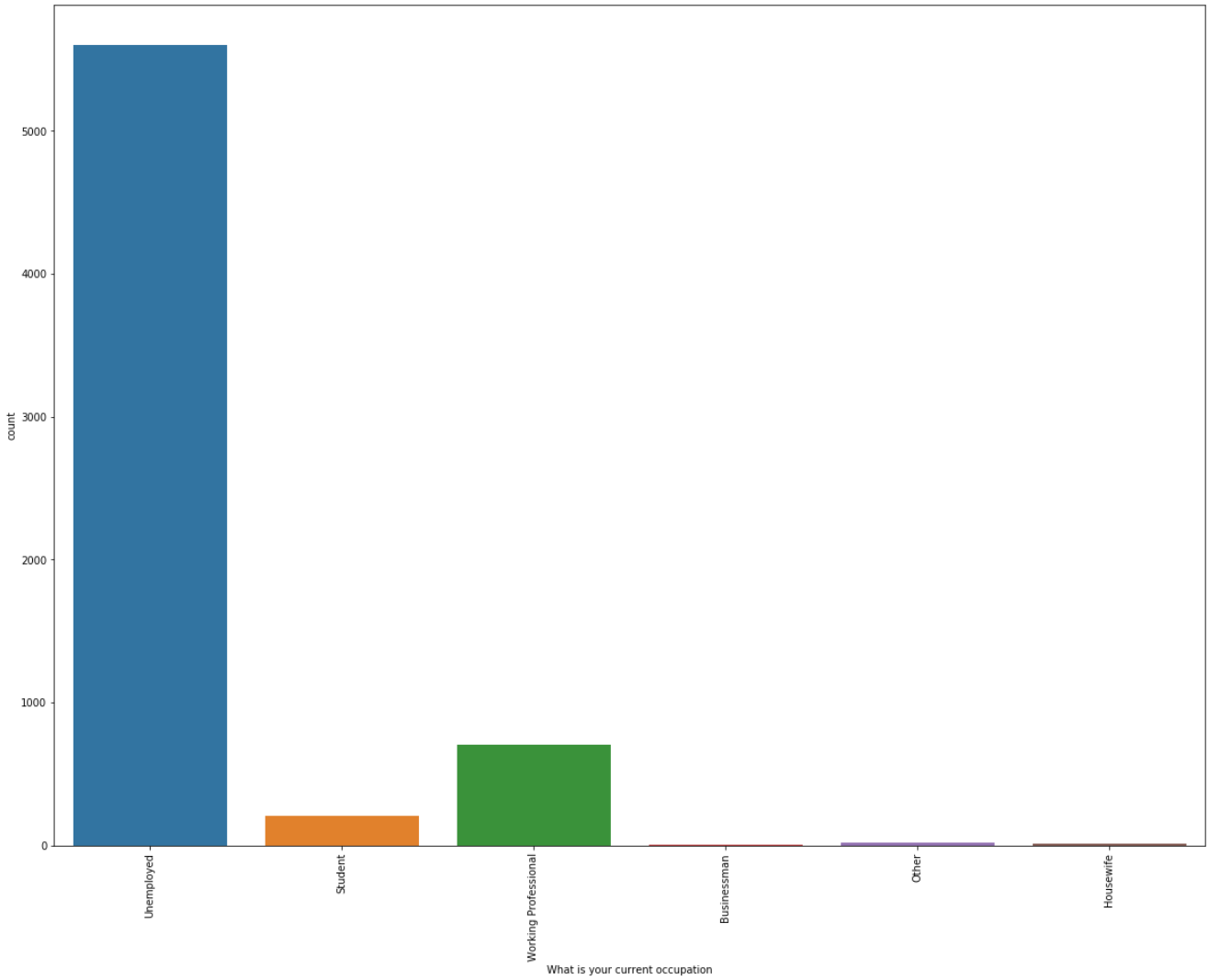
```
count           6550
unique             6
top        Unemployed
freq            5600
Name: What is your current occupation, dtype: object
```

In [21]:

```
plt.figure(figsize=(20,15))
sns.countplot(lead['What is your current occupation'])
plt.xticks(rotation=90)
```

Out[21]:

```
(array([0, 1, 2, 3, 4, 5]), <a list of 6 Text xticklabel objects>)
```



In [22]:

```
lead['What is your current occupation']=lead['What is your current occupation'].replace(np.nan,'Un
employed')
```

In [23]:

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

Out[23]:

```
count                          6531
unique                            3
top       Better Career Prospects
freq                           6528
Name: What matters most to you in choosing a course, dtype: object
```

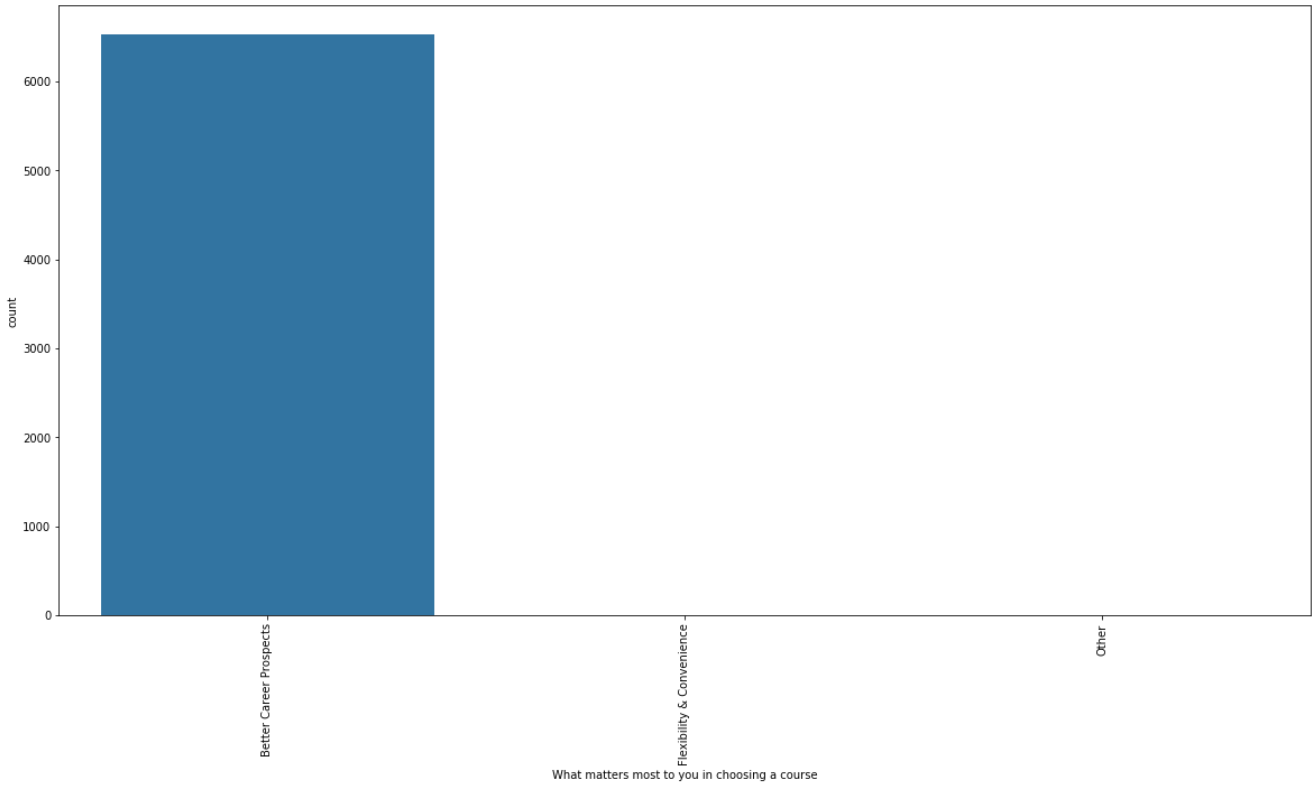
In [24]:

```
plt.figure(figsize=(20,10))
```

```
sns.countplot(lead['What matters most to you in choosing a course'])
plt.xticks(rotation=90)
```

Out[24]:

```
(array([0, 1, 2]), <a list of 3 Text xticklabel objects>)
```



In [25]:

```
lead['What matters most to you in choosing a course']=lead['What matters most to you in choosing a course'].replace(np.nan,'Better Career Prospects')
```

In [26]:

```
lead['Tags'].describe()
```

Out[26]:

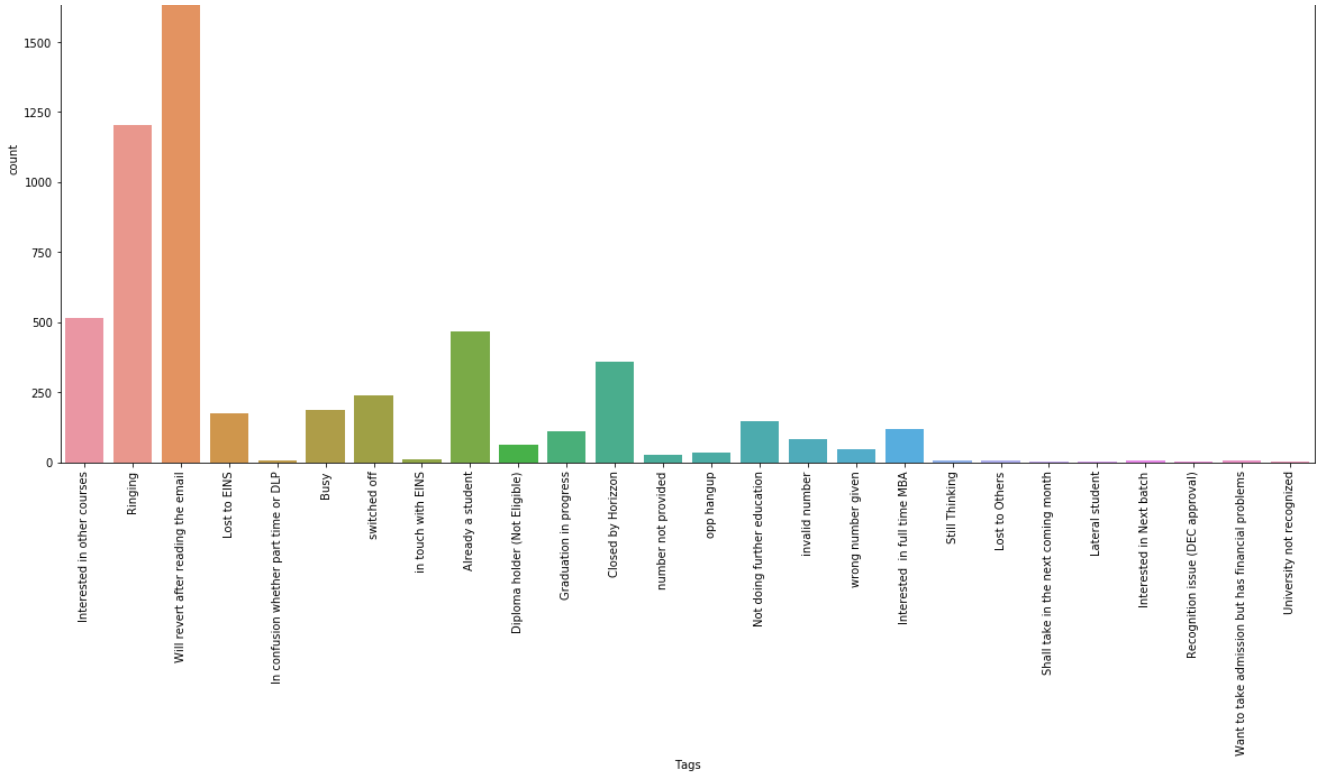
```
count                                   5887
unique                                    26
top       Will revert after reading the email
freq                                    2072
Name: Tags, dtype: object
```

In [27]:

```
plt.figure(figsize=(20,10))
sns.countplot(lead['Tags'])
plt.xticks(rotation=90)
```

Out[27]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25]),
 <a list of 26 Text xticklabel objects>)
```



In [28]:

```
lead['Tags']=lead['Tags'].replace(np.nan,'Will revert after reading the email')
```

In [29]:

```
lead['Lead Quality'].describe()
```

Out[29]:

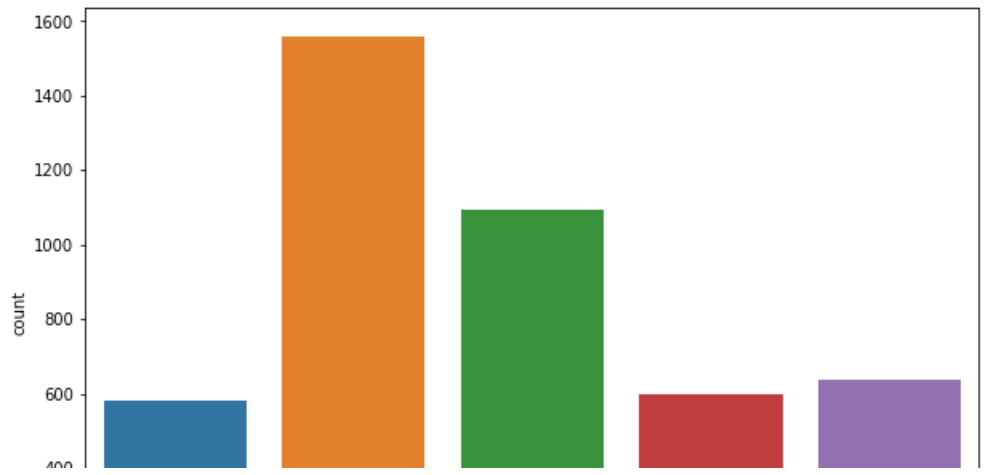
```
count          4473
unique            5
top       Might be
freq           1560
Name: Lead Quality, dtype: object
```

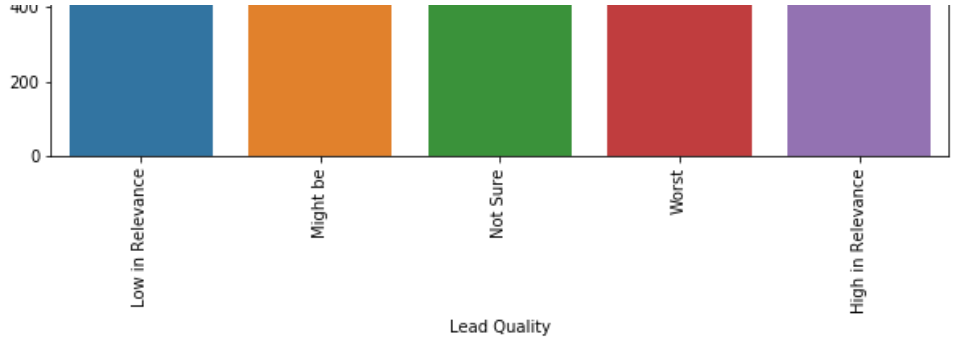
In [30]:

```
plt.figure(figsize=(10,7))
sns.countplot(lead['Lead Quality'])
plt.xticks(rotation=90)
```

Out[30]:

```
(array([0, 1, 2, 3, 4]), <a list of 5 Text xticklabel objects>)
```





In [31]:

```
lead['Lead Quality']=lead['Lead Quality'].replace(np.nan,'Not Sure')
```

In [32]:

```
lead['City'].describe()
```

Out[32]:

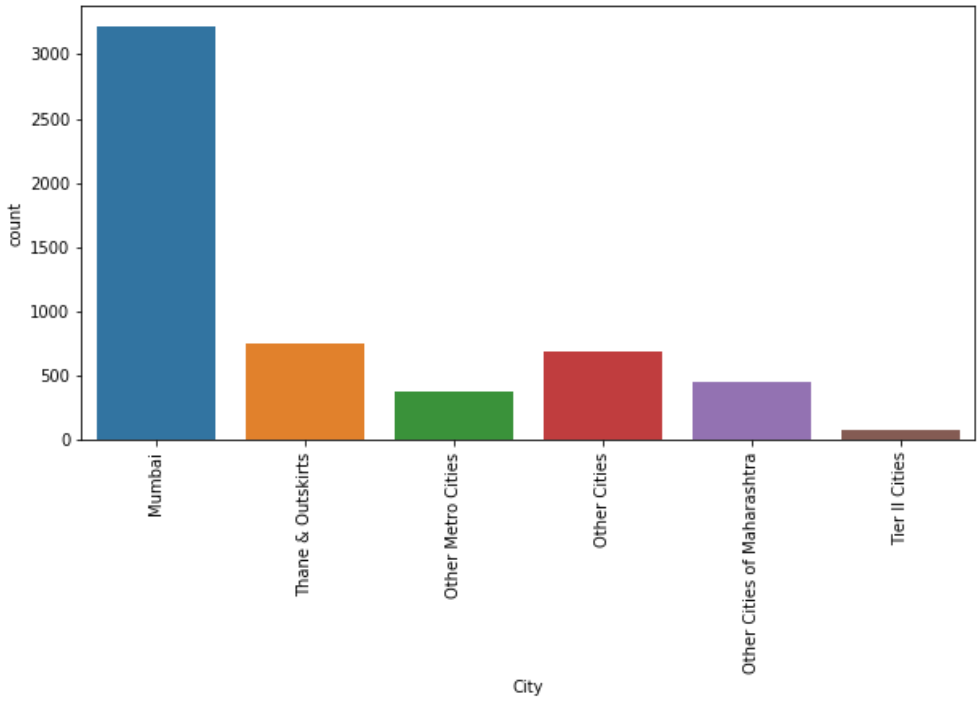
```
count        5571
unique          6
top        Mumbai
freq         3222
Name: City, dtype: object
```

In [33]:

```
plt.figure(figsize=(10,5))
sns.countplot(lead['City'])
plt.xticks(rotation=90)
```

Out[33]:

```
(array([0, 1, 2, 3, 4, 5]), <a list of 6 Text xticklabel objects>)
```



In [34]:

```
lead['City']=lead['City'].replace(np.nan,'Mumbai')
```

In [35]:

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

Out[35]:

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

In [36]:

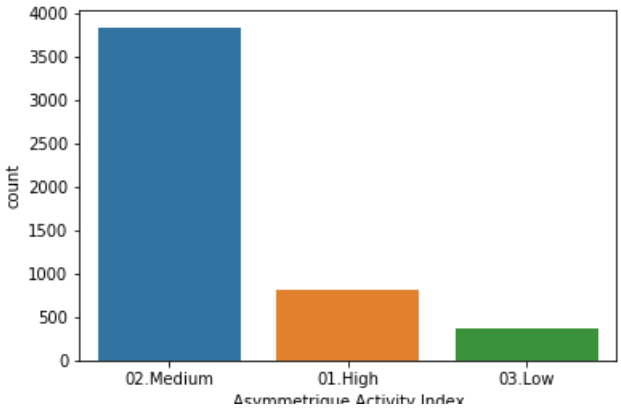
```
lead['Asymmetrique Activity Index'].describe()
```

Out[36]:

```
count           5022
unique             3
top       02.Medium
freq            3839
Name: Asymmetrique Activity Index, dtype: object
```

In [37]:

```
plt1=sns.countplot(lead['Asymmetrique Activity Index'])
```



In [38]:

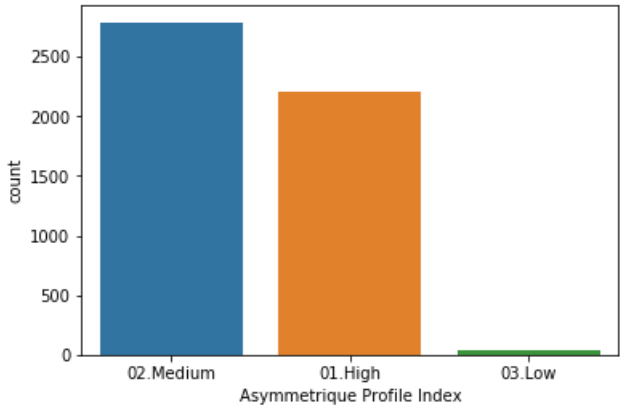
```
lead['Asymmetrique Profile Index'].describe()
```

Out[38]:

```
count           5022
unique             3
top        02.Medium
freq            2788
Name: Asymmetrique Profile Index, dtype: object
```

In [39]:

```
plt2=sns.countplot(lead['Asymmetrique Profile Index'])
```



In [40]:

```
lead['Asymmetrique Activity Score'].describe()
```

Out[40]:

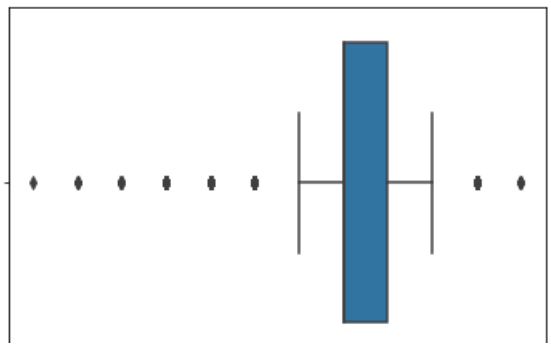
```
count     5022.000000
mean        14.306252
std          1.386694
min          7.000000
25%         14.000000
50%         14.000000
75%         15.000000
max         18.000000
Name: Asymmetrique Activity Score, dtype: float64
```

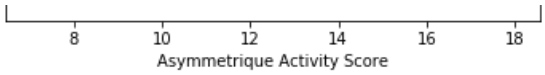
In [41]:

```
sns.boxplot(lead['Asymmetrique Activity Score'])
```

Out[41]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c32a5b4a8>
```





In [42]:

```
lead['Asymmetrique Profile Score'].describe()
```

Out[42]:

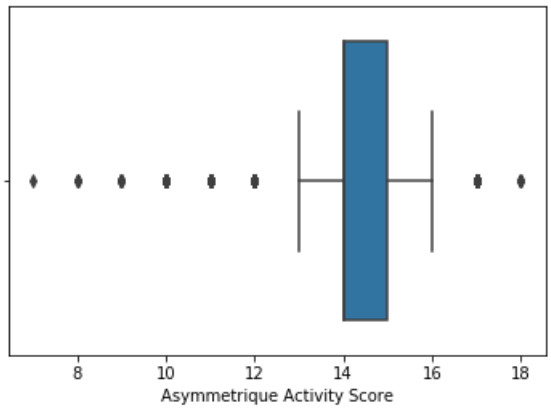
```
count     5022.000000
mean        16.344883
std          1.811395
min         11.000000
25%         15.000000
50%         16.000000
75%         18.000000
max         20.000000
Name: Asymmetrique Profile Score, dtype: float64
```

In [43]:

```
sns.boxplot(lead['Asymmetrique Activity Score'])
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c32aa1d30>
```



In [44]:

```
lead=lead.drop(['Asymmetrique Activity Index','Asymmetrique Profile Index','Asymmetrique Activity Score','Asymmetrique Profile Score'],1)
```

In [45]:

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

Out[45]:

```
Prospect ID                                    0.00
Lead Number                                    0.00
Lead Origin                                    0.00
Lead Source                                    0.39
Do Not Email                                   0.00
Do Not Call                                    0.00
Converted                                      0.00
TotalVisits                                    1.48
Total Time Spent on Website                    0.00
Page Views Per Visit                           1.48
Last Activity                                  1.11
Country                                        0.00
Specialization                                 0.00
What is your current occupation                0.00
What matters most to you in choosing a course  0.00
Search                                         0.00
```

```
Magazine                                        0.00
Newspaper Article                               0.00
X Education Forums                              0.00
Newspaper                                       0.00
Digital Advertisement                           0.00
Through Recommendations                         0.00
Receive More Updates About Our Courses          0.00
Tags                                            0.00
Lead Quality                                    0.00
Update me on Supply Chain Content               0.00
Get updates on DM Content                       0.00
City                                            0.00
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
```

In [46]:

```
lead.dropna(inplace=True)
```

In [47]:

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

Out[47]:

```
Prospect ID                                     0.0
Lead Number                                     0.0
Lead Origin                                     0.0
Lead Source                                     0.0
Do Not Email                                    0.0
Do Not Call                                     0.0
Converted                                       0.0
TotalVisits                                     0.0
Total Time Spent on Website                     0.0
Page Views Per Visit                            0.0
Last Activity                                   0.0
Country                                         0.0
Specialization                                  0.0
What is your current occupation                 0.0
What matters most to you in choosing a course   0.0
Search                                          0.0
Magazine                                        0.0
Newspaper Article                               0.0
X Education Forums                              0.0
Newspaper                                       0.0
Digital Advertisement                           0.0
Through Recommendations                         0.0
Receive More Updates About Our Courses          0.0
Tags                                            0.0
Lead Quality                                    0.0
Update me on Supply Chain Content               0.0
Get updates on DM Content                       0.0
City                                            0.0
I agree to pay the amount through cheque        0.0
A free copy of Mastering The Interview          0.0
Last Notable Activity                           0.0
dtype: float64
```

In [48]:

```
lead.shape
```

Out[48]:

```
(9074, 31)
```

Step3:Analyzing the Data

In [49]:

```
Converted=(sum(lead['Converted'])/len(lead['Converted'].index))*100
Converted
```

Out[49]:

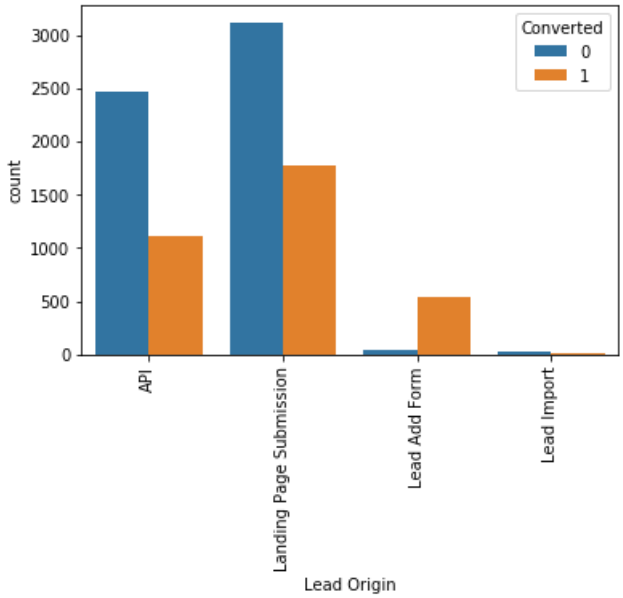
```
37.85541106458012
```

In [50]:

```
sns.countplot(x='Lead Origin',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[50]:

```
(array([0, 1, 2, 3]), <a list of 4 Text xticklabel objects>)
```

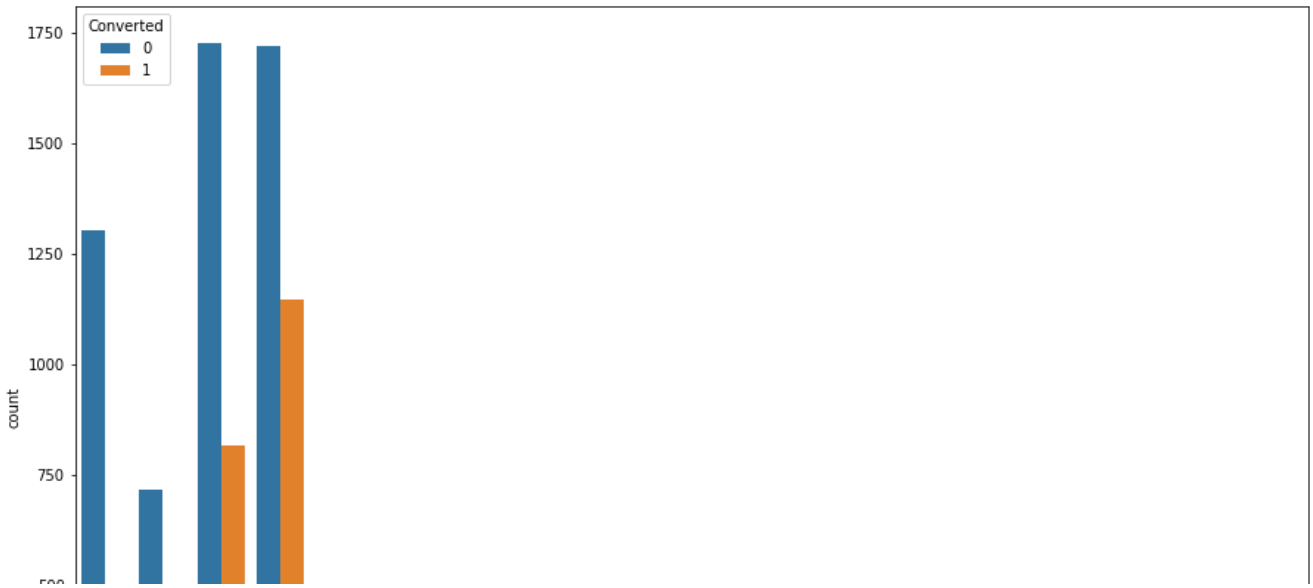


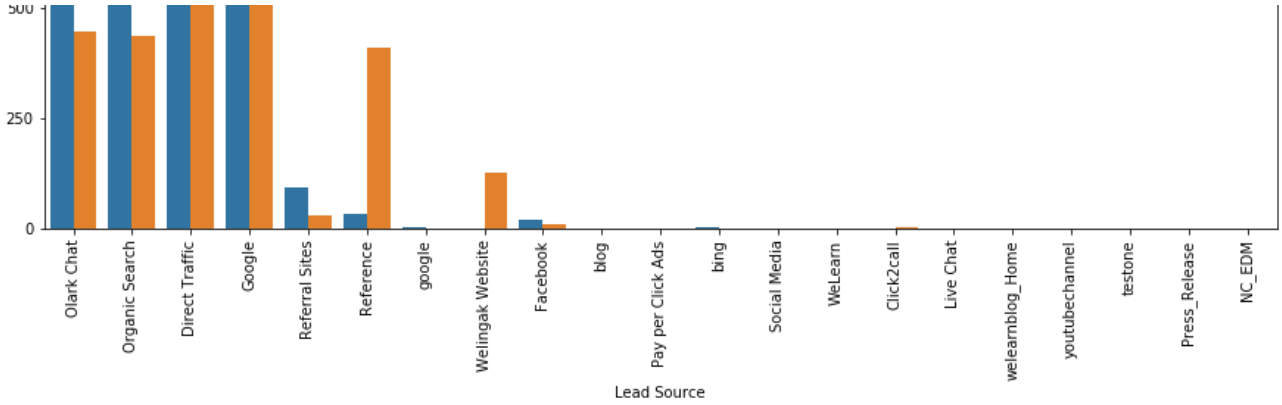
In [51]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Lead Source',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[51]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20]), <a list of 21 Text xticklabel objects>)
```





In [52]:

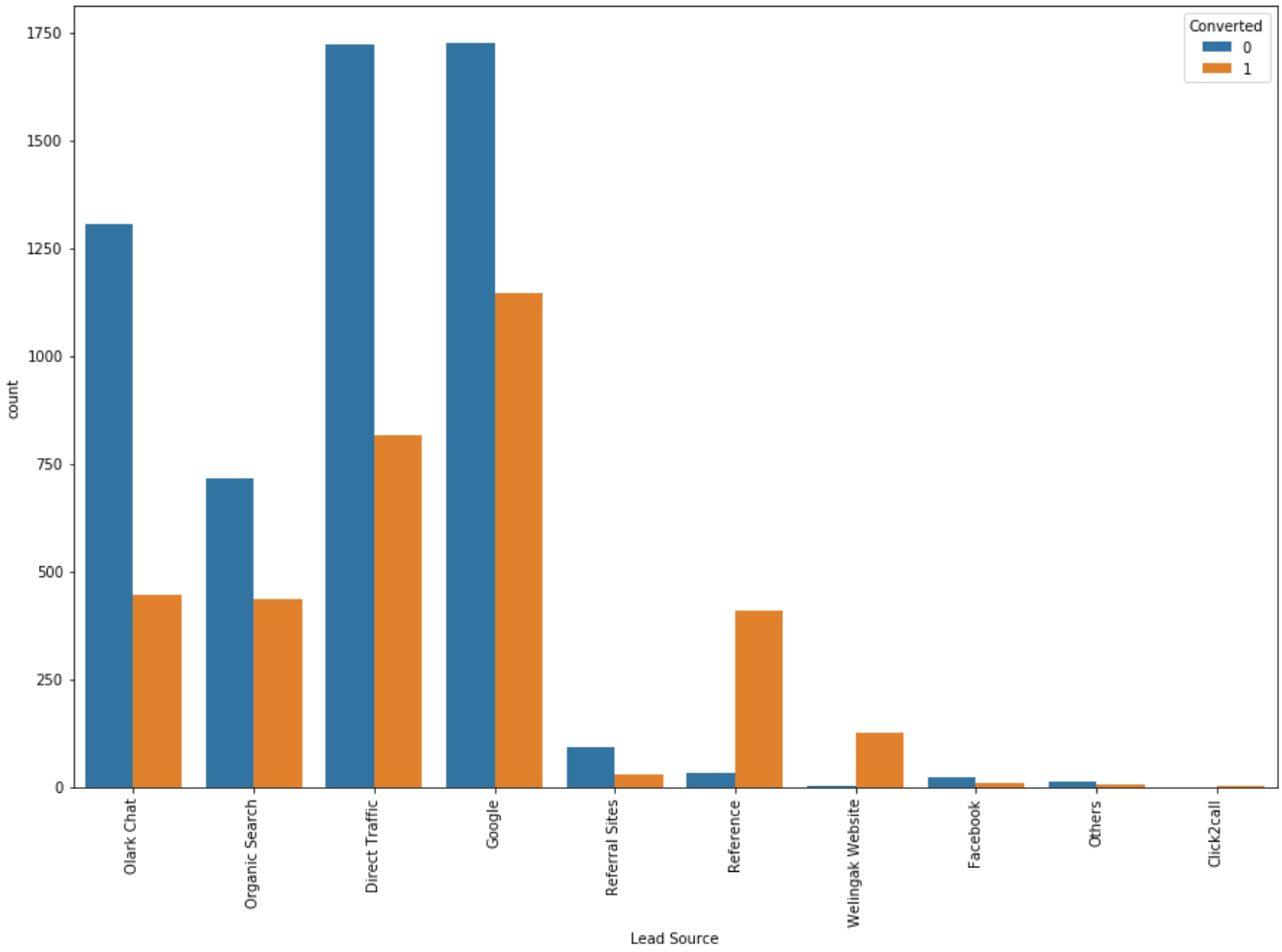
```
lead['Lead Source']=lead['Lead Source'].replace(['google'],'Google')
lead['Lead Source']=lead['Lead Source'].replace(['blog','Pay per Click Ads','bing','Social Media',
'WeLearn','Click2cal','Live     Chat','welearnblog_Home','youtubechannel','testone','Press_Release','N
C_EDM'],'Others')
```

In [53]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Lead Source',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[53]:

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]), <a list of 10 Text xticklabel objects>)
```

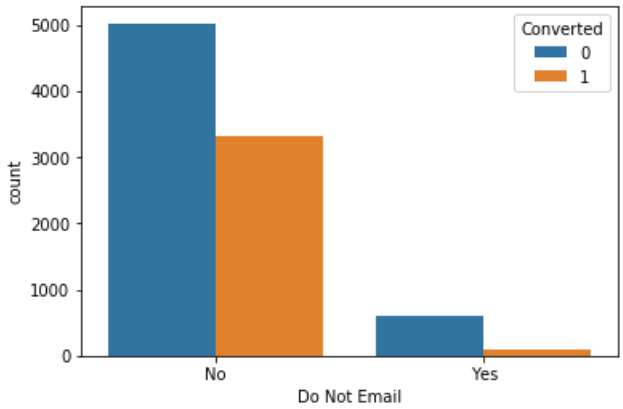


In [54]:

```
sns.countplot(x='Do Not Email',hue='Converted',data=lead)
```

Out[54]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c32b63be0>
```

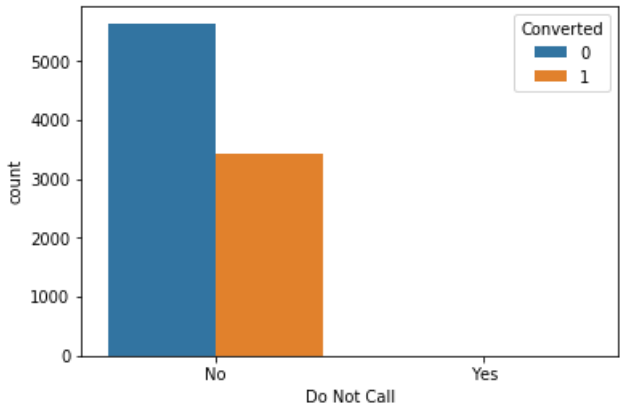


In [55]:

```
sns.countplot(x='Do Not Call',hue='Converted',data=lead)
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c32d48be0>
```



In [56]:

```
lead['TotalVisits'].describe(percentiles=(.05,.10,.25,.50,.75,.90,.95,.99))
```

Out[56]:

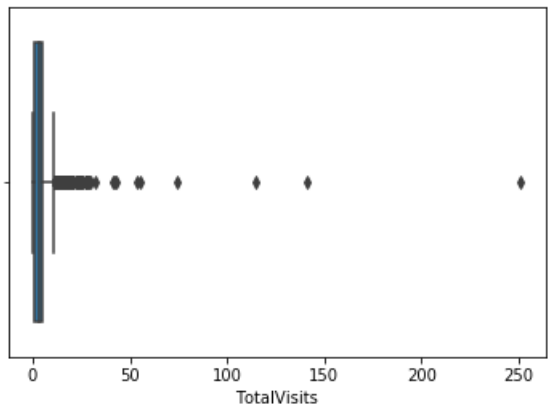
```
count     9074.000000
mean         3.456028
std          4.858802
min          0.000000
5%           0.000000
10%          0.000000
25%          1.000000
50%          3.000000
75%          5.000000
90%          7.000000
95%         10.000000
99%         17.000000
max        251.000000
Name: TotalVisits, dtype: float64
```

In [57]:

```
sns.boxplot(lead['TotalVisits'])
```

Out[57]:

<matplotlib.axes._subplots.AxesSubplot at 0x29c32d927f0>



In [58]:

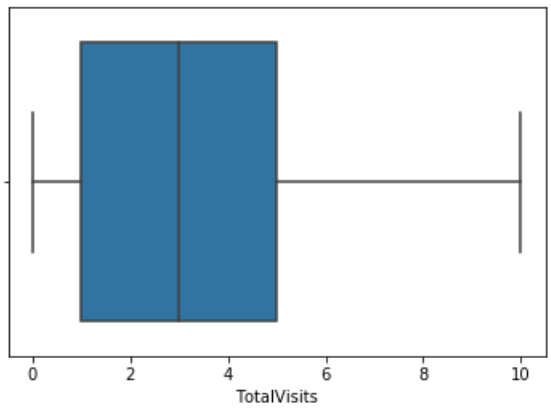
```
percentiles=lead['TotalVisits'].quantile([0.05,0.95]).values
lead['TotalVisits'][lead['TotalVisits']<=percentiles[0]]=percentiles[0]
lead['TotalVisits'][lead['TotalVisits']>=percentiles[1]]=percentiles[1]
```

In [59]:

```
sns.boxplot(lead['TotalVisits'])
```

Out[59]:

<matplotlib.axes._subplots.AxesSubplot at 0x29c32def518>

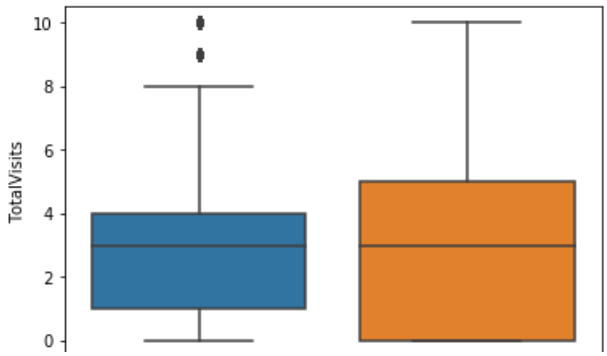


In [60]:

```
sns.boxplot(y='TotalVisits',x='Converted',data=lead)
```

Out[60]:

<matplotlib.axes._subplots.AxesSubplot at 0x29c32b150b8>



In [61]:

```
lead['Total Time Spent on Website'].describe()
```

Out[61]:

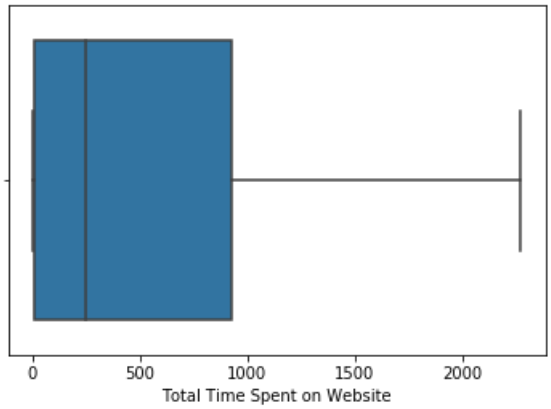
```
count     9074.000000
mean       482.887481
std        545.256560
min          0.000000
25%         11.000000
50%        246.000000
75%        922.750000
max       2272.000000
Name: Total Time Spent on Website, dtype: float64
```

In [62]:

```
sns.boxplot(lead['Total Time Spent on Website'])
```

Out[62]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c33f0b470>
```

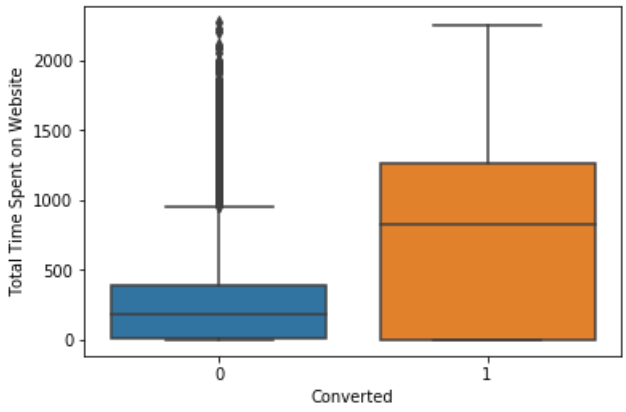


In [63]:

```
sns.boxplot(y='Total Time Spent on Website',x='Converted',data=lead)
```

Out[63]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c33f677f0>
```



In [64]:

```
lead['Page Views Per Visit'].describe()
```

Out[64]:

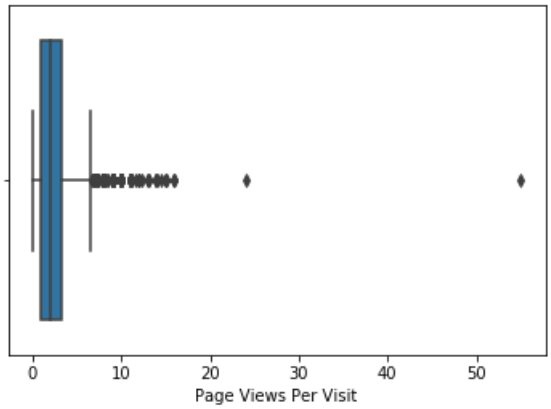
```
count     9074.000000
mean         2.370151
std          2.160871
min          0.000000
25%          1.000000
50%          2.000000
75%          3.200000
max         55.000000
Name: Page Views Per Visit, dtype: float64
```

In [65]:

```
sns.boxplot(lead['Page Views Per Visit'])
```

Out[65]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c33fc0470>
```



In [66]:

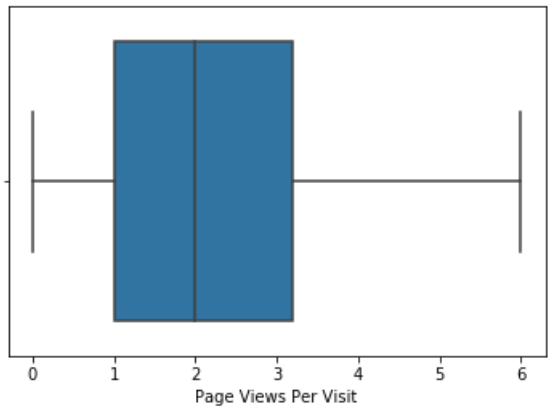
```
percentiles=lead['Page Views Per Visit'].quantile([0.05,0.95]).values
lead['Page Views Per Visit'][lead['Page Views Per Visit']<=percentiles[0]]=percentiles[0]
lead['Page Views Per Visit'][lead['Page Views Per Visit']>=percentiles[1]]=percentiles[1]
```

In [67]:

```
sns.boxplot(lead['Page Views Per Visit'])
```

Out[67]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c340302b0>
```

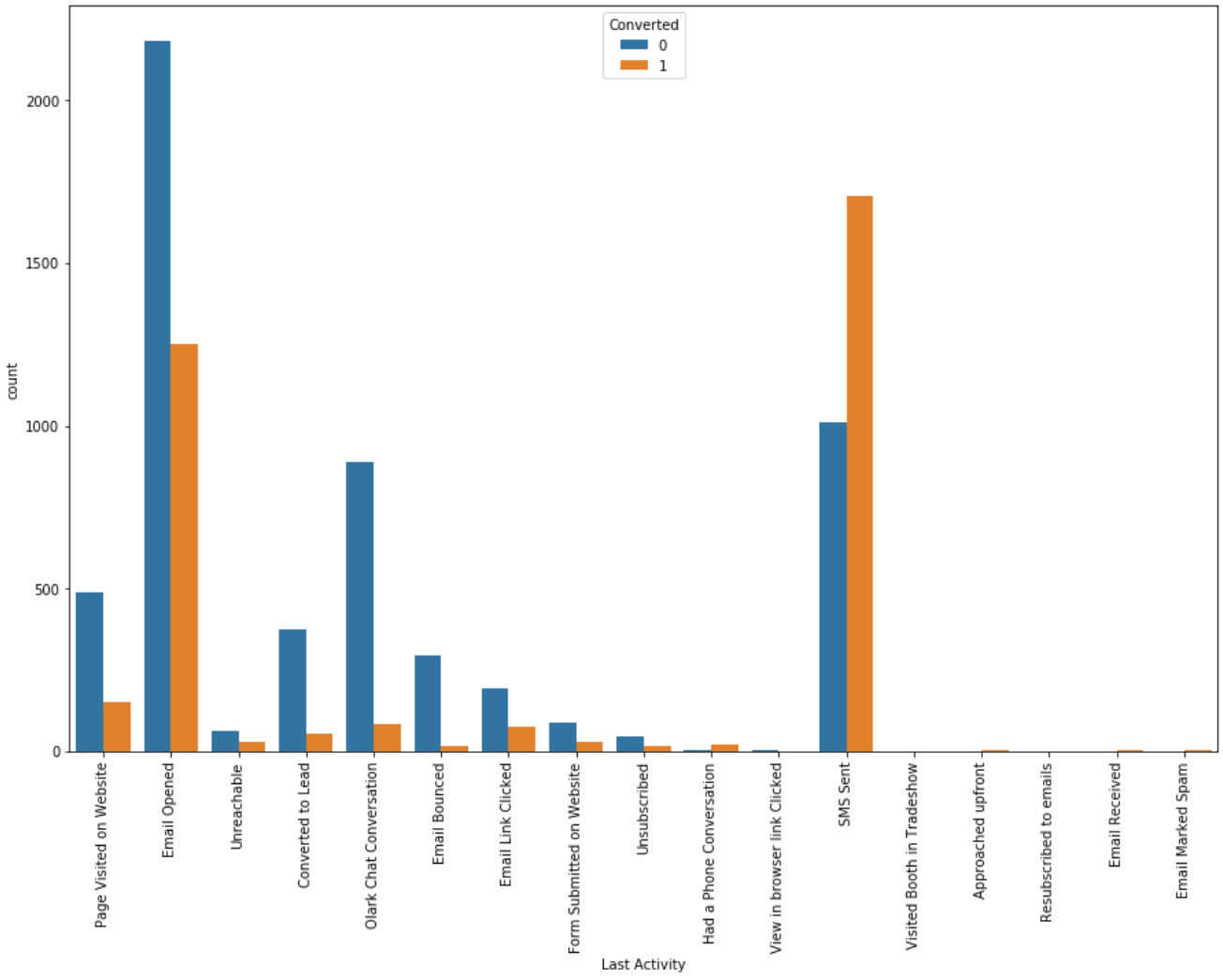


In [68]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Last Activity',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[68]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16]),
 <a list of 17 Text xticklabel objects>)
```



In [69]:

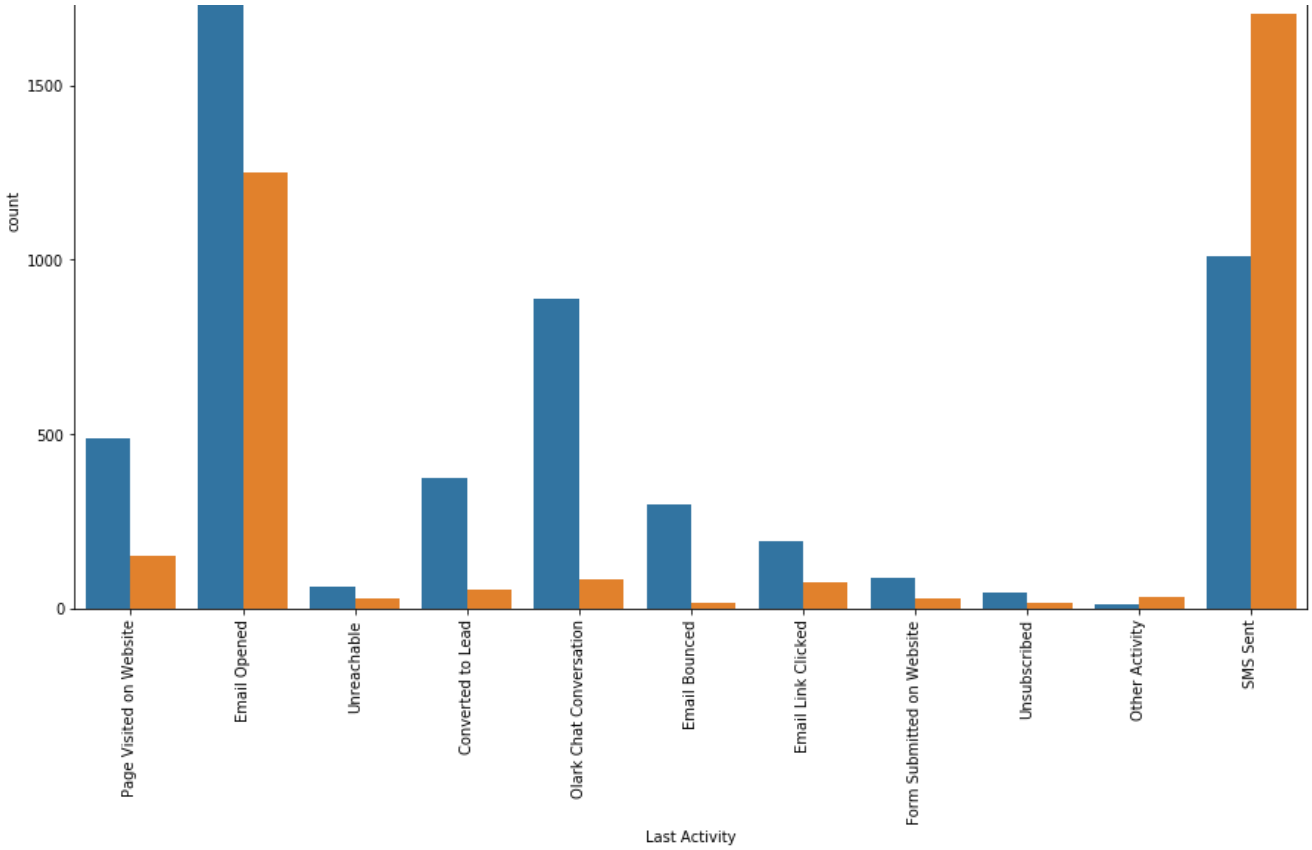
```
lead['Last Activity']=lead['Last Activity'].replace(['Had a Phone Conversation','View in browser l
ink Clicked','Visited Booth in Tradeshow','Approached upfront','Resubscribed to emails','Email
Received','Email Marked Spam'],'Other Activity')
```

In [70]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Last Activity',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[70]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10]),
 <a list of 11 Text xticklabel objects>)
```

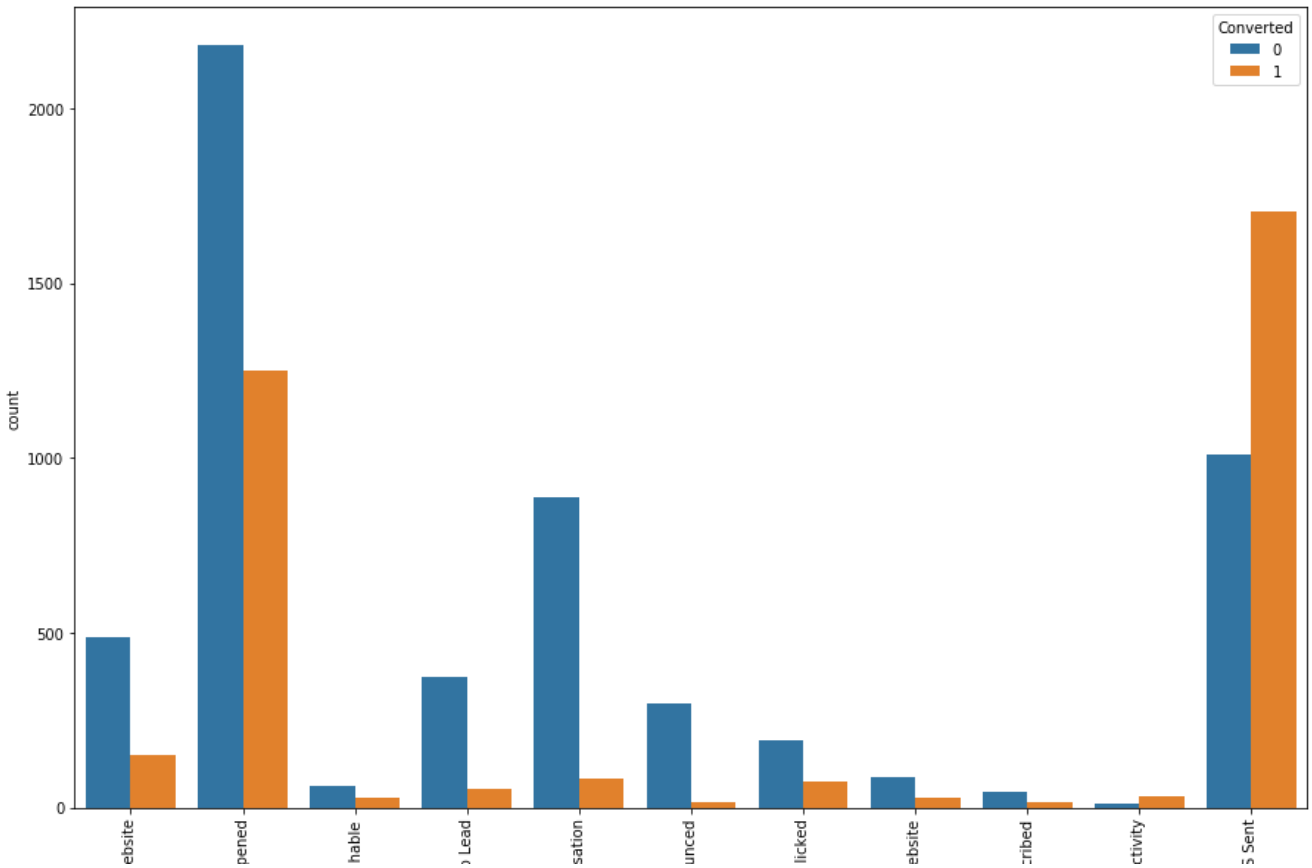


In [71]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Last Activity',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[71]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10]),
 <a list of 11 Text xticklabel objects>)
```

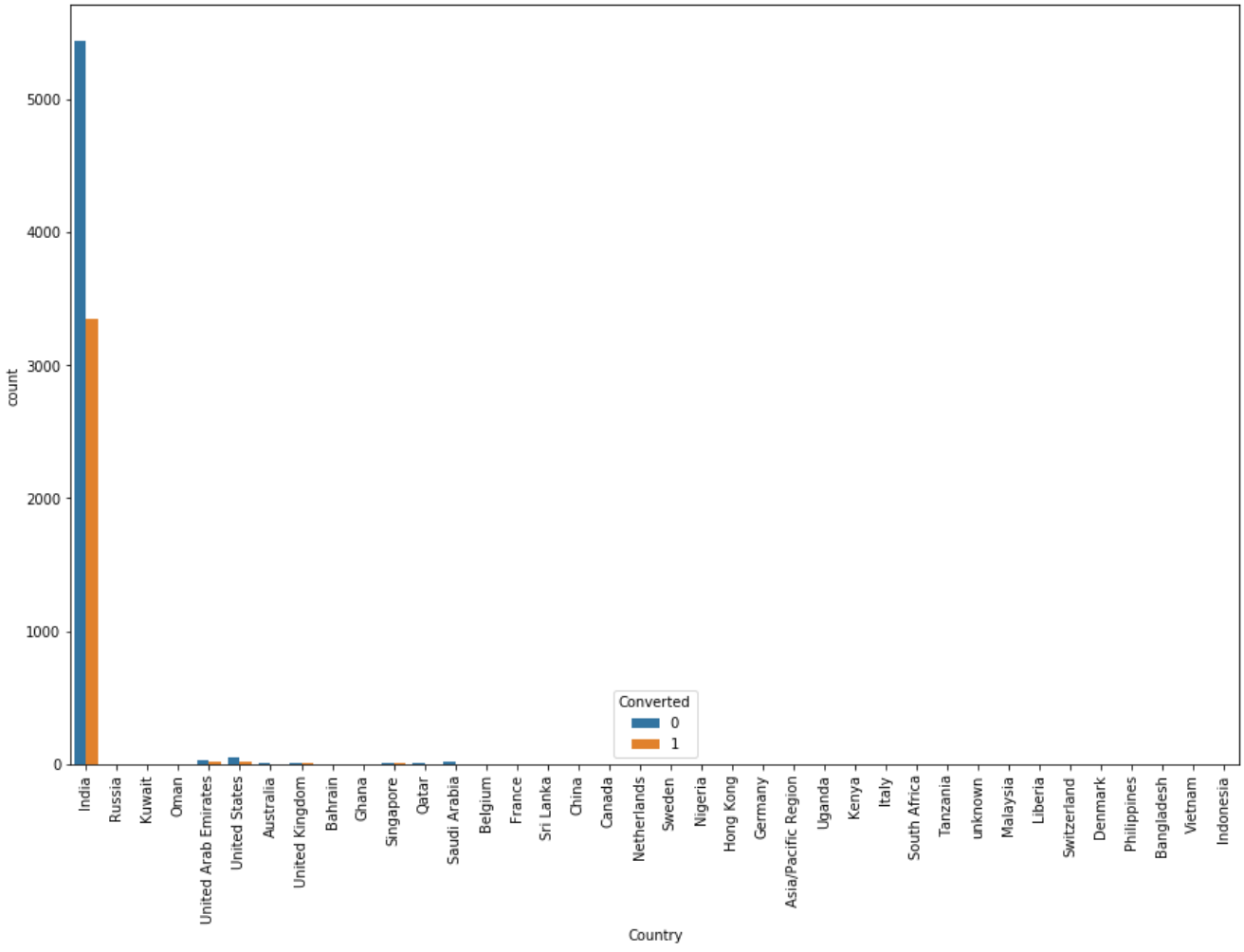


In [72]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Country',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[72]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37]), <a list of 38 Text xticklabel objects>)
```



In [73]:

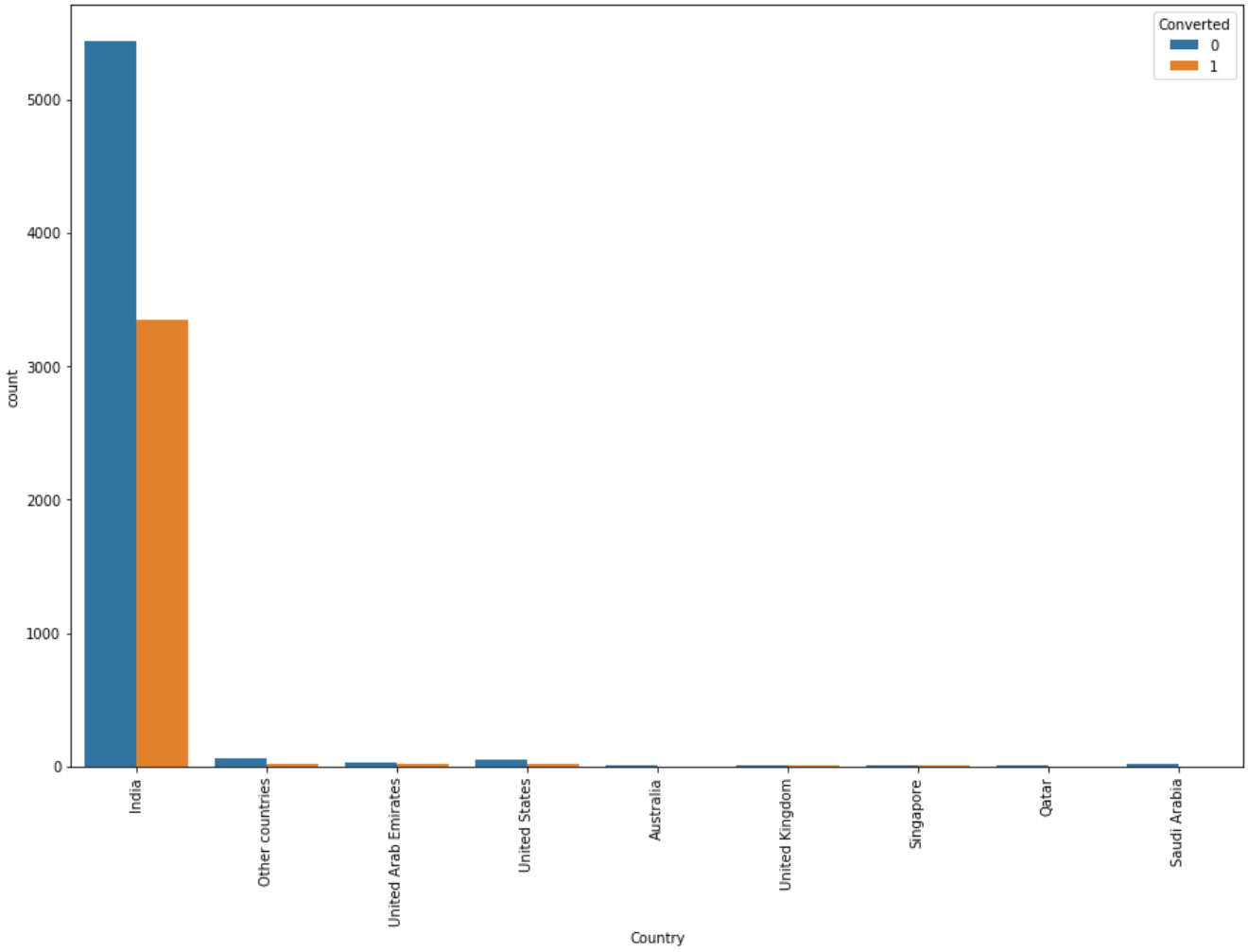
```
lead['Country']=lead['Country'].replace(['Russia','Kuwait','Oman','Bahrain','Ghana','Belgium','Fra
nce','Sri Lanka','China','Canada','Netherlands','Sweden','Nigeria','Hong
Kong','Germany','Asia/Pacific Region','Uganda','Kenya','Italy','South  Africa','Tanzania','unknown'
,'Malaysia','Liberia','Switzerland','Denmark','Philippines','Bangladesh','Vietnam','Indonesia'],'O
ther countries')
```

In [74]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Country',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[74]:

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8]), <a list of 9 Text xticklabel objects>)
```



In [75]:

```
lead['Specialization'].describe()
```

Out[75]:

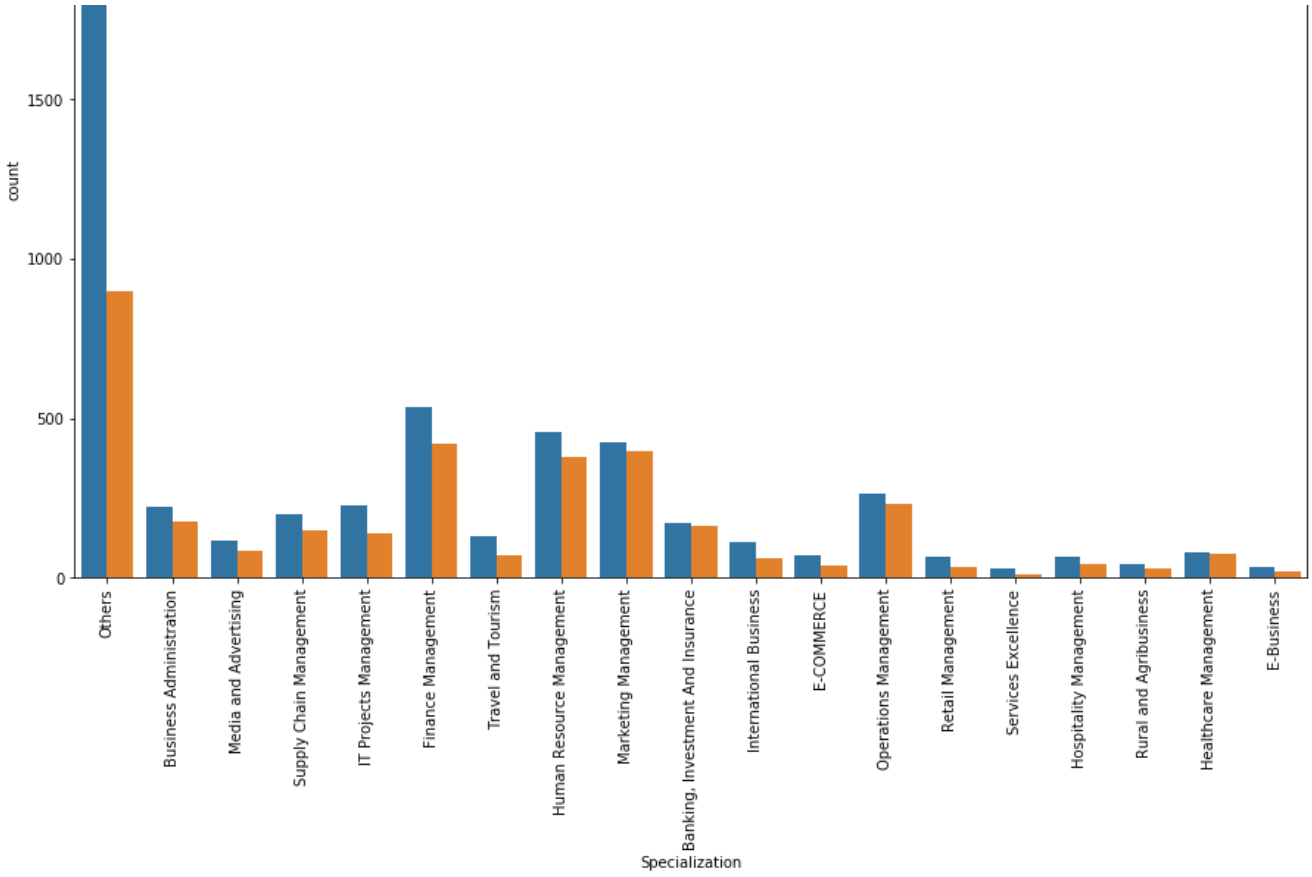
```
count        9074
unique         19
top        Others
freq         3282
Name: Specialization, dtype: object
```

In [76]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Specialization',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[76]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18]), <a list of 19 Text xticklabel objects>)
```



In [77]:

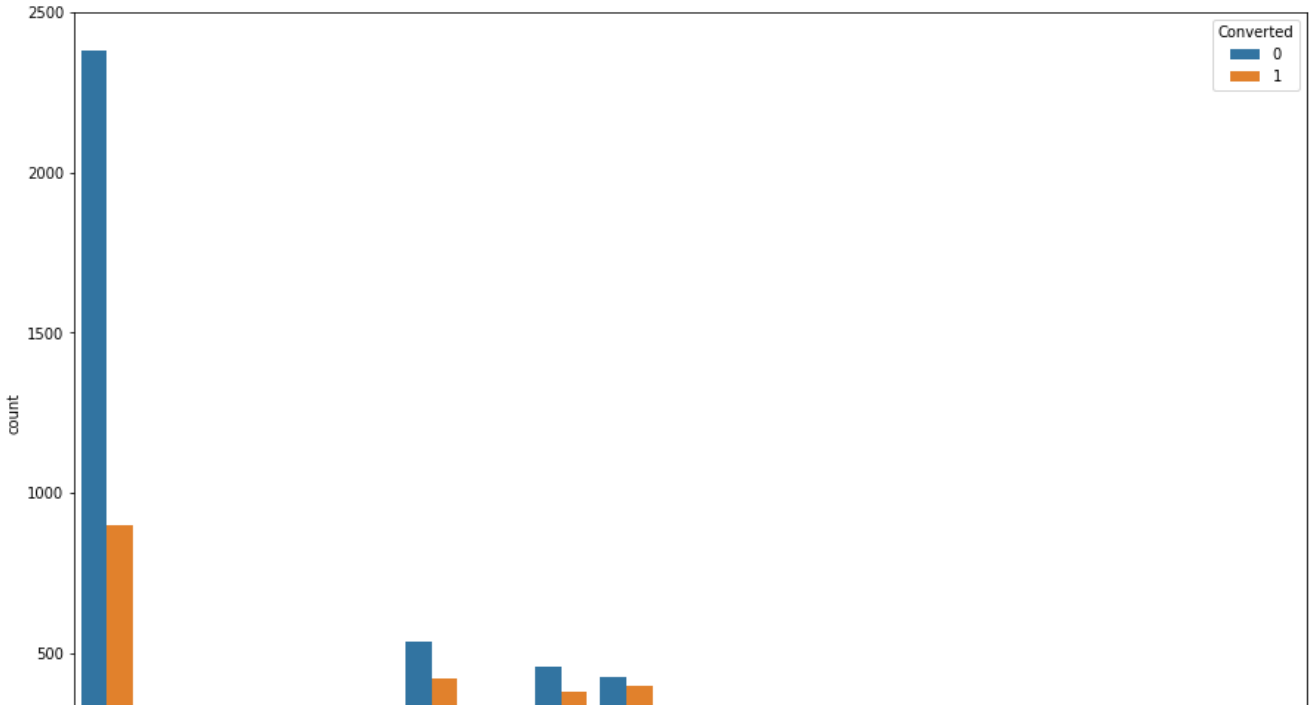
```
lead['Specialization']=lead['Specialization'].replace('Others','Other Specialization')
```

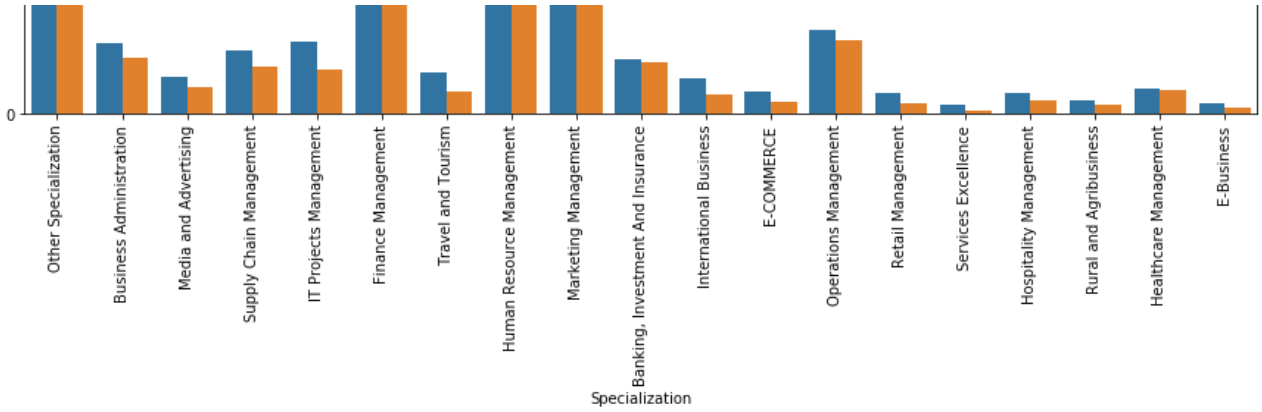
In [78]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Specialization',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[78]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18]), <a list of 19 Text xticklabel objects>)
```





In [79]:

```
lead['What is your current occupation'].describe()
```

Out[79]:

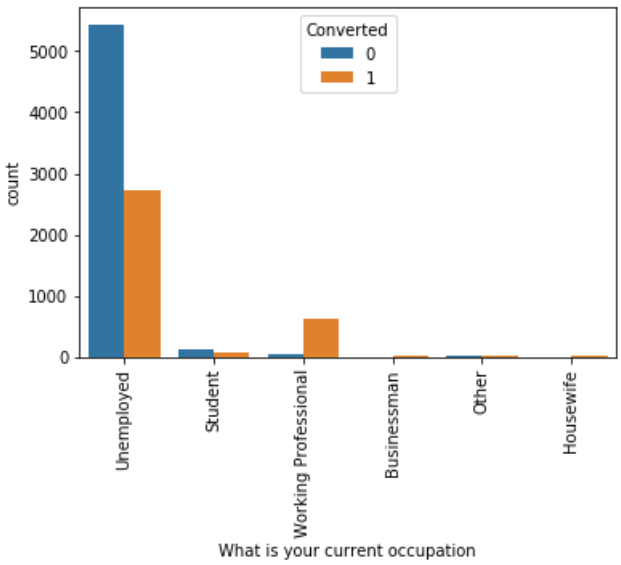
```
count            9074
unique              6
top        Unemployed
freq             8159
Name: What is your current occupation, dtype: object
```

In [80]:

```
sns.countplot(x='What is your current occupation',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[80]:

```
(array([0, 1, 2, 3, 4, 5]), <a list of 6 Text xticklabel objects>)
```



In [81]:

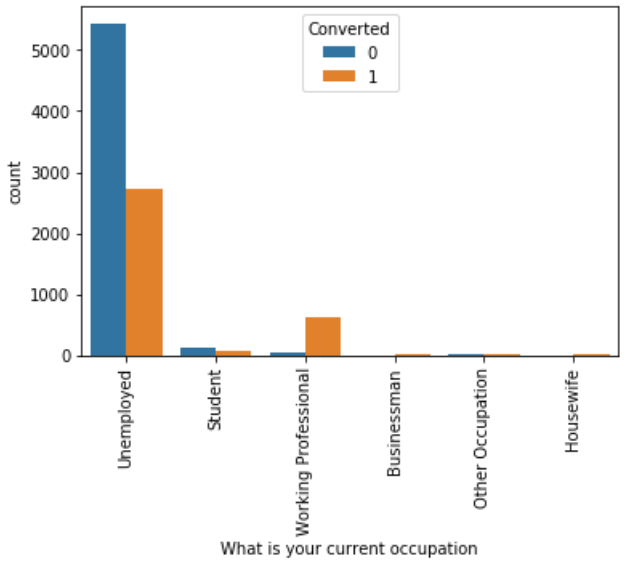
```
lead['What is your current occupation']=lead['What is your current occupation'].replace('Other','O
ther Occupation')
```

In [82]:

```
sns.countplot(x='What is your current occupation',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[82]:

```
(array([0, 1, 2, 3, 4, 5]), <a list of 6 Text xticklabel objects>)
```



In [83]:

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

Out[83]:

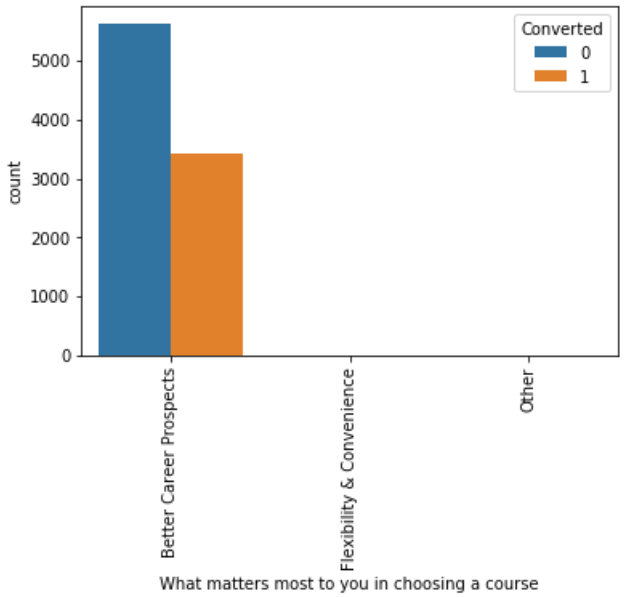
```
count                          9074
unique                            3
top       Better Career Prospects
freq                           9072
Name: What matters most to you in choosing a course, dtype: object
```

In [84]:

```
sns.countplot(x='What matters most to you in choosing a course',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[84]:

```
(array([0, 1, 2]), <a list of 3 Text xticklabel objects>)
```



In [85]:

```
lead['Search'].describe()
```

Out[85]:

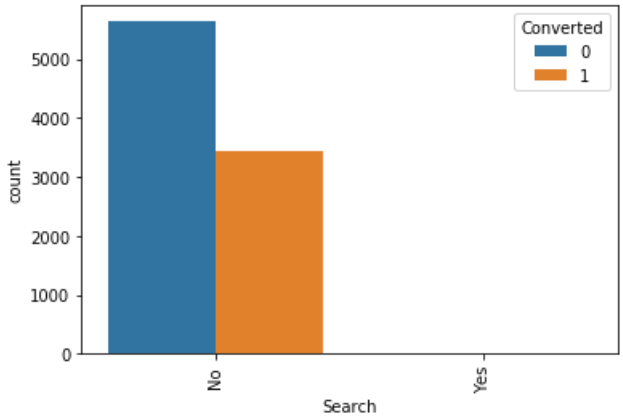
```
count      9074
unique        2
top          No
freq       9060
Name: Search, dtype: object
```

In [86]:

```
sns.countplot(x='Search',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[86]:

```
(array([0, 1]), <a list of 2 Text xticklabel objects>)
```



In [87]:

```
lead['Magazine'].describe()
```

Out[87]:

```
count      9074
unique        1
top          No
freq       9074
Name: Magazine, dtype: object
```

In [88]:

```
lead['Newspaper Article'].describe()
```

Out[88]:

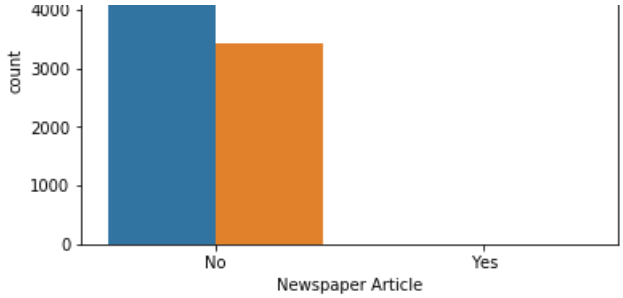
```
count      9074
unique        2
top          No
freq       9072
Name: Newspaper Article, dtype: object
```

In [89]:

```
sns.countplot(x='Newspaper Article',hue='Converted',data=lead)
```

Out[89]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c36432550>
```



In [90]:

```
lead['X Education Forums'].describe()
```

Out[90]:

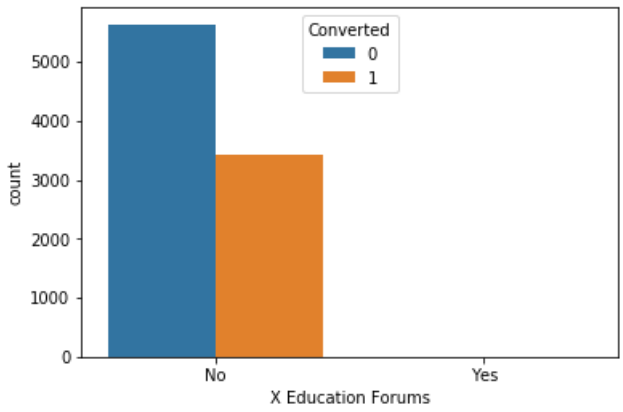
```
count     9074
unique       2
top         No
freq      9073
Name: X Education Forums, dtype: object
```

In [91]:

```
sns.countplot(x='X Education Forums',hue='Converted',data=lead)
```

Out[91]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c3647b550>
```



In [92]:

```
lead['Newspaper'].describe()
```

Out[92]:

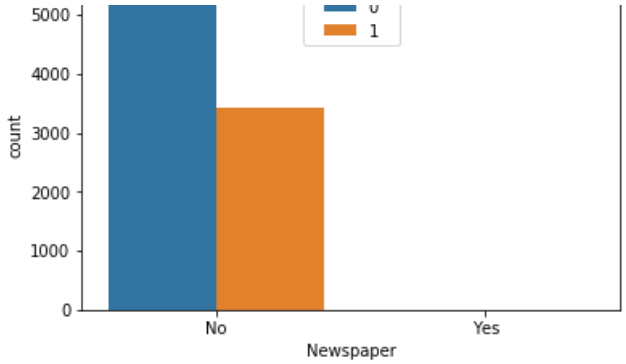
```
count     9074
unique       2
top         No
freq      9073
Name: Newspaper, dtype: object
```

In [93]:

```
sns.countplot(x='Newspaper',hue='Converted',data=lead)
```

Out[93]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c364d6c50>
```



In [94]:

```
lead['Digital Advertisement'].describe()
```

Out[94]:

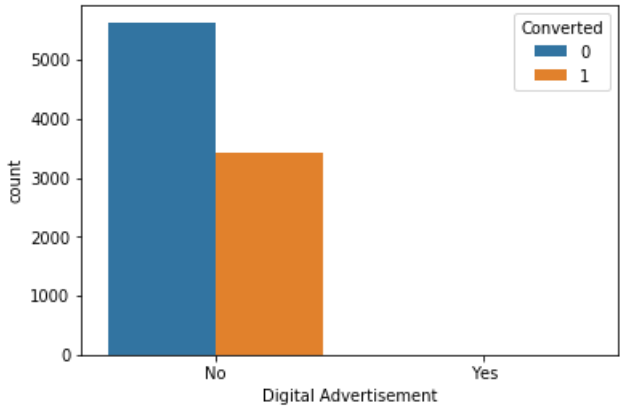
```
count      9074
unique        2
top          No
freq       9070
Name: Digital Advertisement, dtype: object
```

In [95]:

```
sns.countplot(x='Digital Advertisement',hue='Converted',data=lead)
```

Out[95]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c36cc30f0>
```



In [96]:

```
lead['Through Recommendations'].describe()
```

Out[96]:

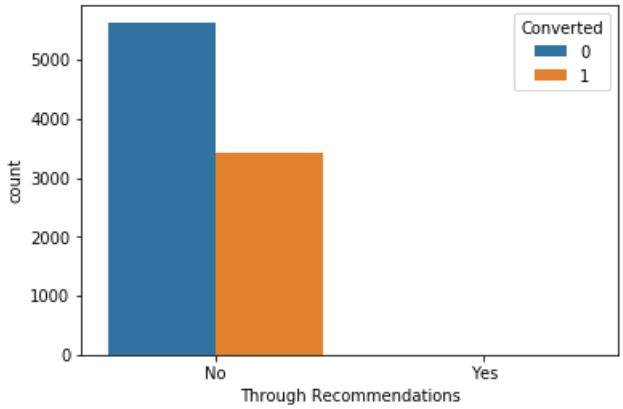
```
count      9074
unique        2
top          No
freq       9067
Name: Through Recommendations, dtype: object
```

In [97]:

```
sns.countplot(x='Through Recommendations',hue='Converted',data=lead)
```

Out[97]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29c36d168d0>
```



In [98]:

```
lead['Receive More Updates About Our Courses'].describe()
```

Out[98]:

```
count      9074
unique        1
top          No
freq       9074
Name: Receive More Updates About Our Courses, dtype: object
```

In [99]:

```
lead['Tags'].describe()
```

Out[99]:

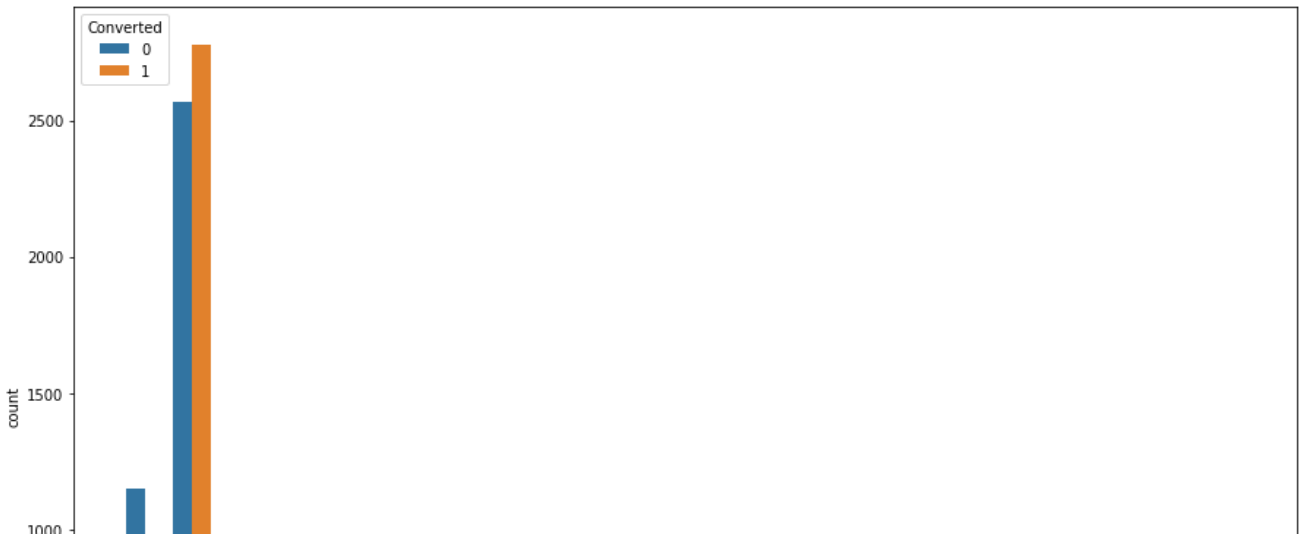
```
count                                     9074
unique                                      26
top       Will revert after reading the email
freq                                      5343
Name: Tags, dtype: object
```

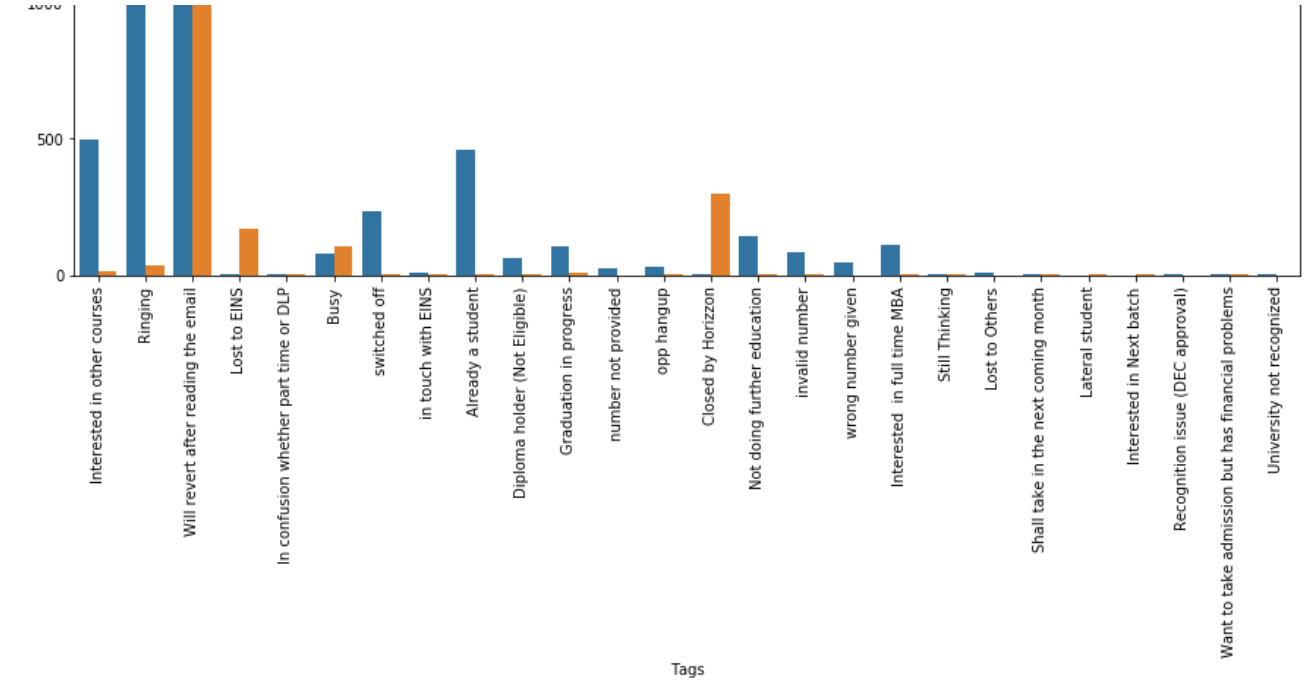
In [100]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Tags',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[100]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25]),
 <a list of 26 Text xticklabel objects>)
```





In [101]:

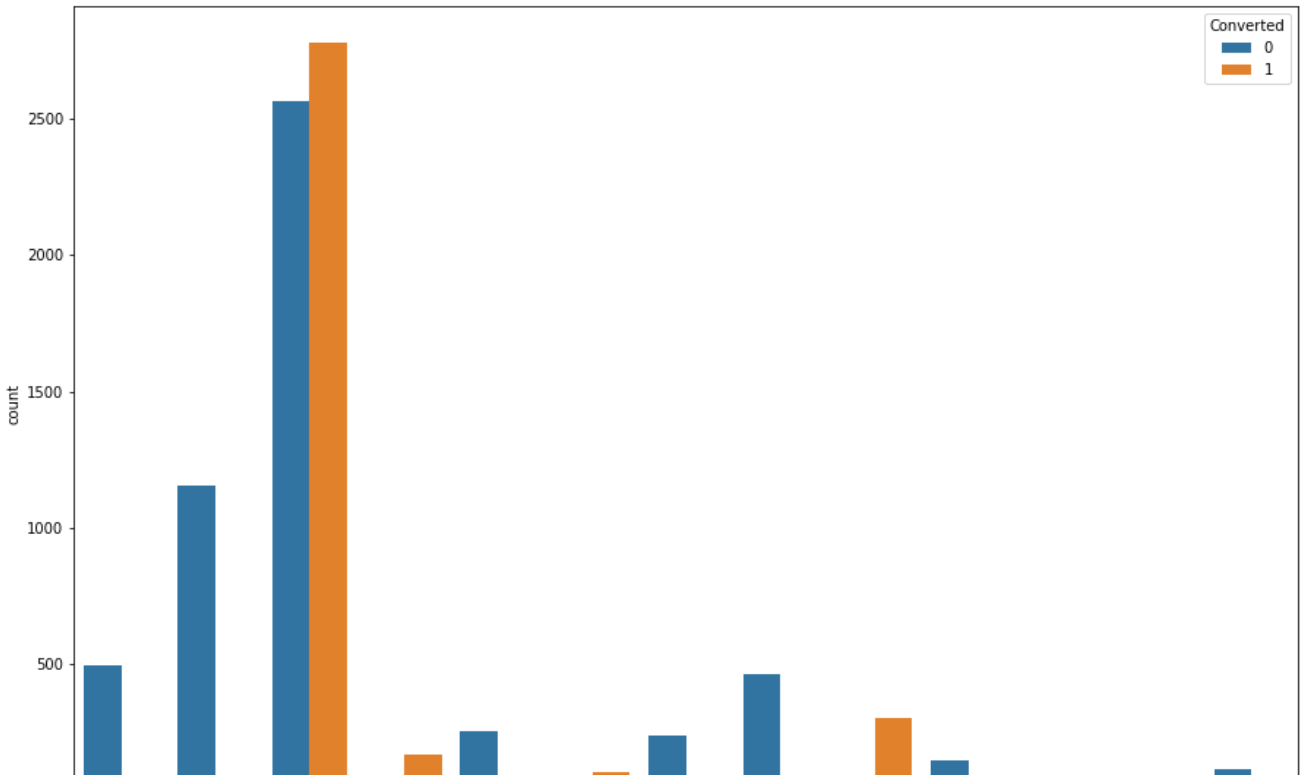
```
lead['Tags']=lead['Tags'].replace(['In confusion whether part time or DLP','in touch with
EINS','Diploma holder (Not Eligible)','Graduation in progress','number not provided','opp hangup',
'Still Thinking','Lost to Others','Shall take in the next coming month','Lateral
student','Interested in Next batch','Recognition issue (DEC approval)','Want to take admission but
has financial problems','University not recognized'],'Other Tags')
```

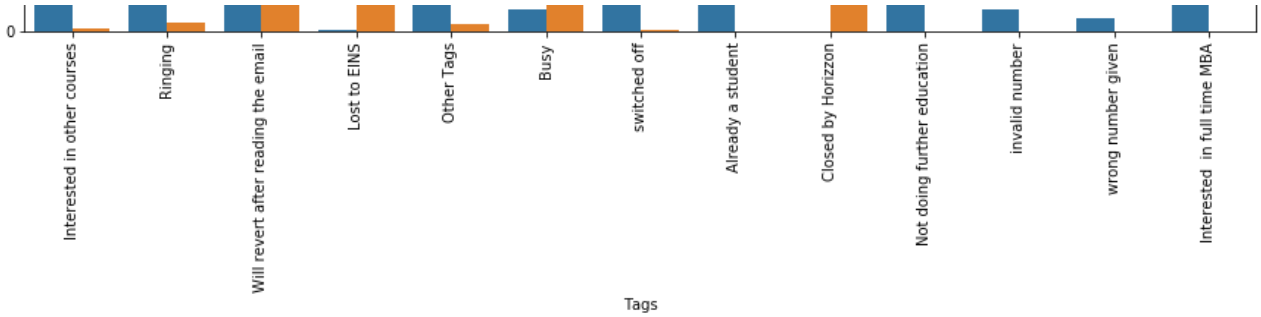
In [102]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Tags',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[102]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12]),
 <a list of 13 Text xticklabel objects>)
```





In [103]:

```
lead['Lead Quality'].describe()
```

Out[103]:

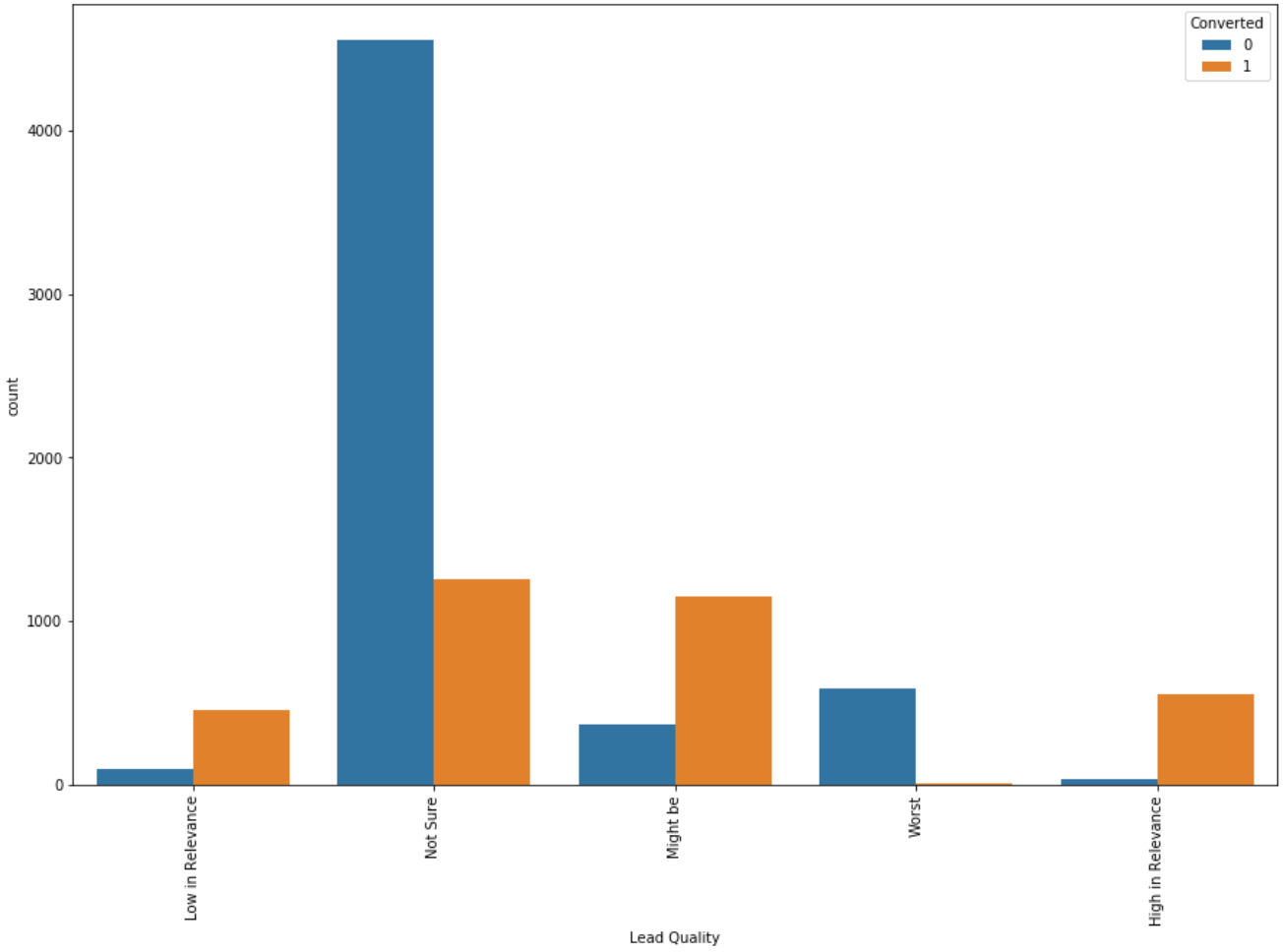
```
count          9074
unique            5
top       Not Sure
freq           5806
Name: Lead Quality, dtype: object
```

In [104]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Lead Quality',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[104]:

```
(array([0, 1, 2, 3, 4]), <a list of 5 Text xticklabel objects>)
```



In [105]:

```
lead['Update me on Supply Chain Content'].describe()
```

Out[105]:

```
count     9074
unique       1
top         No
freq      9074
Name: Update me on Supply Chain Content, dtype: object
```

In [106]:

```
lead['Get updates on DM Content'].describe()
```

Out[106]:

```
count     9074
unique       1
top         No
freq      9074
Name: Get updates on DM Content, dtype: object
```

In [107]:

```
lead['City'].describe()
```

Out[107]:

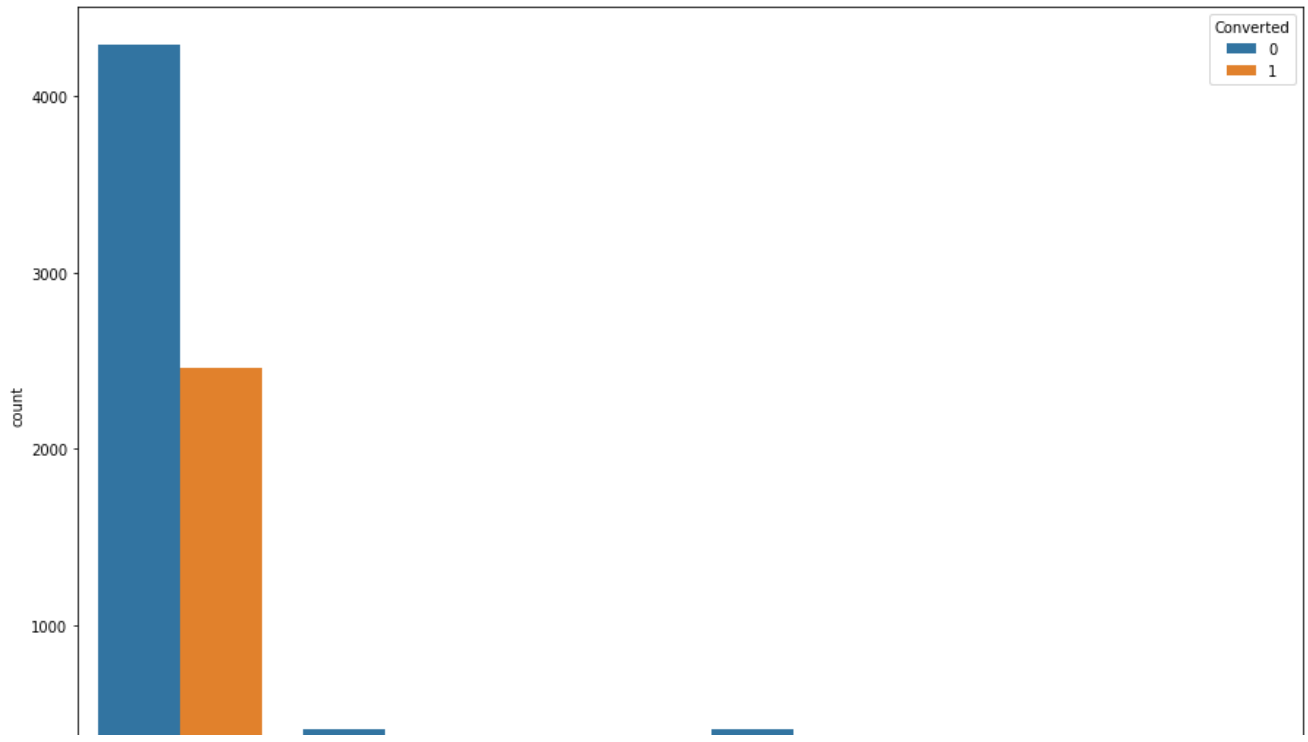
```
count       9074
unique         6
top       Mumbai
freq        6752
Name: City, dtype: object
```

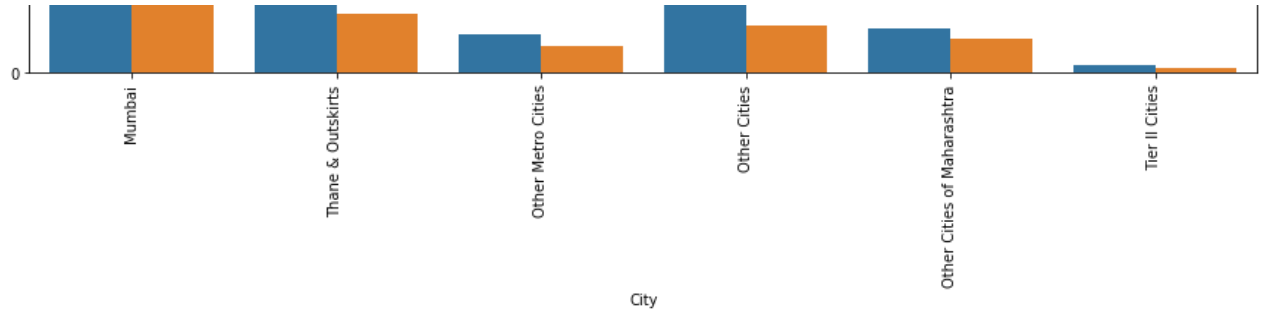
In [108]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='City',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[108]:

```
(array([0, 1, 2, 3, 4, 5]), <a list of 6 Text xticklabel objects>)
```





In [109]:

```
lead['I agree to pay the amount through cheque'].describe()
```

Out[109]:

```
count      9074
unique        1
top          No
freq       9074
Name: I agree to pay the amount through cheque, dtype: object
```

In [110]:

```
lead['A free copy of Mastering The Interview'].describe()
```

Out[110]:

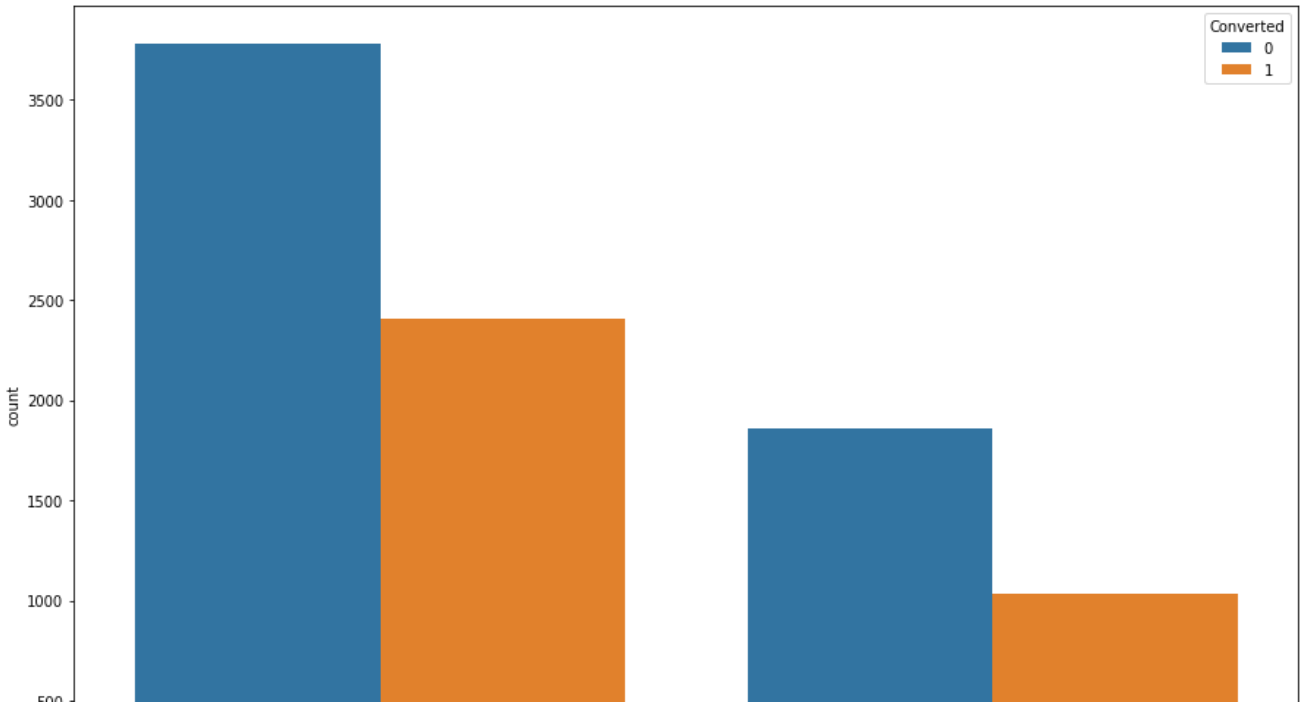
```
count      9074
unique        2
top          No
freq       6186
Name: A free copy of Mastering The Interview, dtype: object
```

In [111]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='A free copy of Mastering The Interview',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[111]:

```
(array([0, 1]), <a list of 2 Text xticklabel objects>)
```



In [112]:

```
lead['Last Notable Activity'].describe()
```

Out[112]:

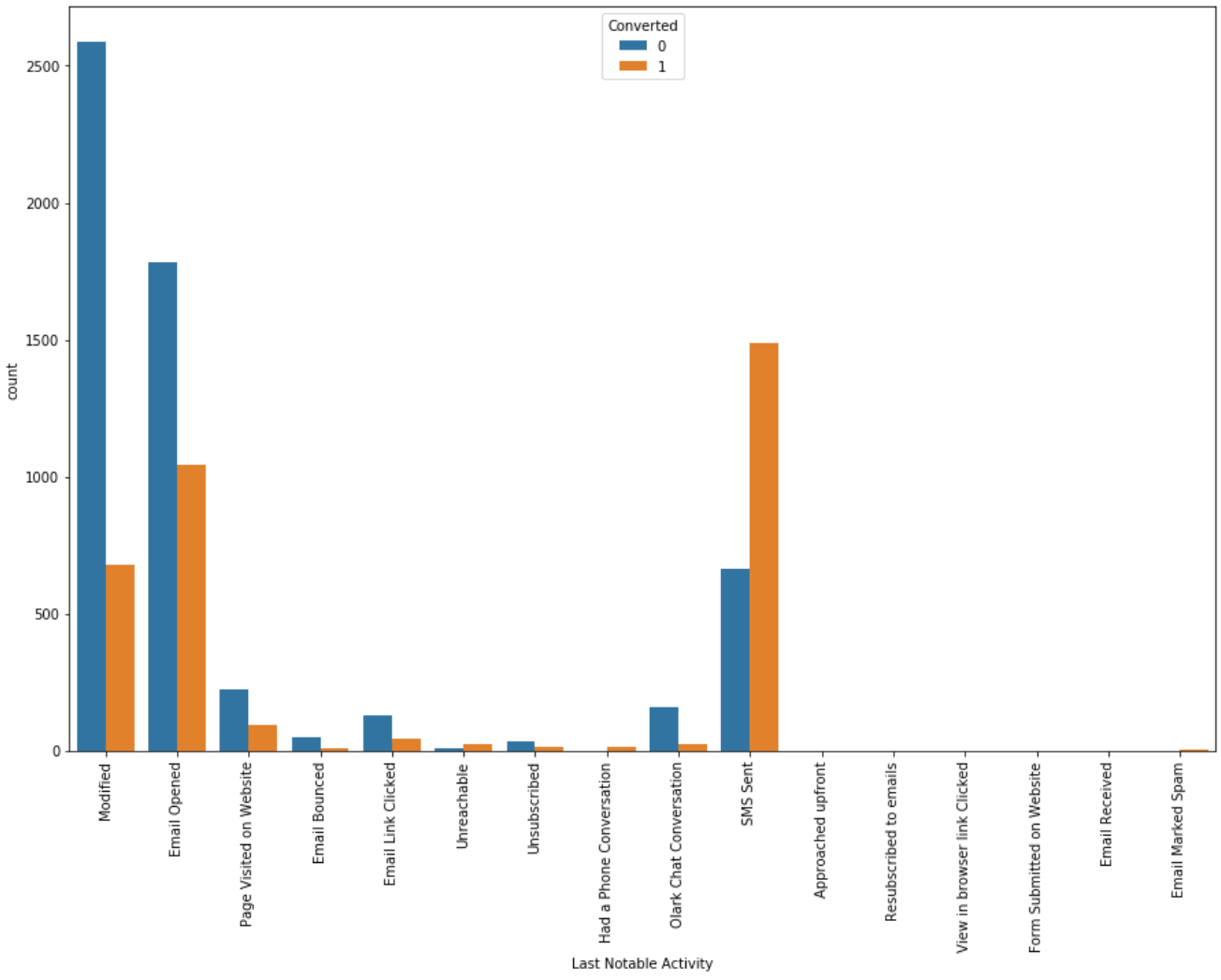
```
count          9074
unique           16
top        Modified
freq           3267
Name: Last Notable Activity, dtype: object
```

In [113]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Last Notable Activity',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[113]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15]),
 <a list of 16 Text xticklabel objects>)
```



In [114]:

```
lead['Last Notable Activity']=lead['Last Notable Activity'].replace(['Approached
```

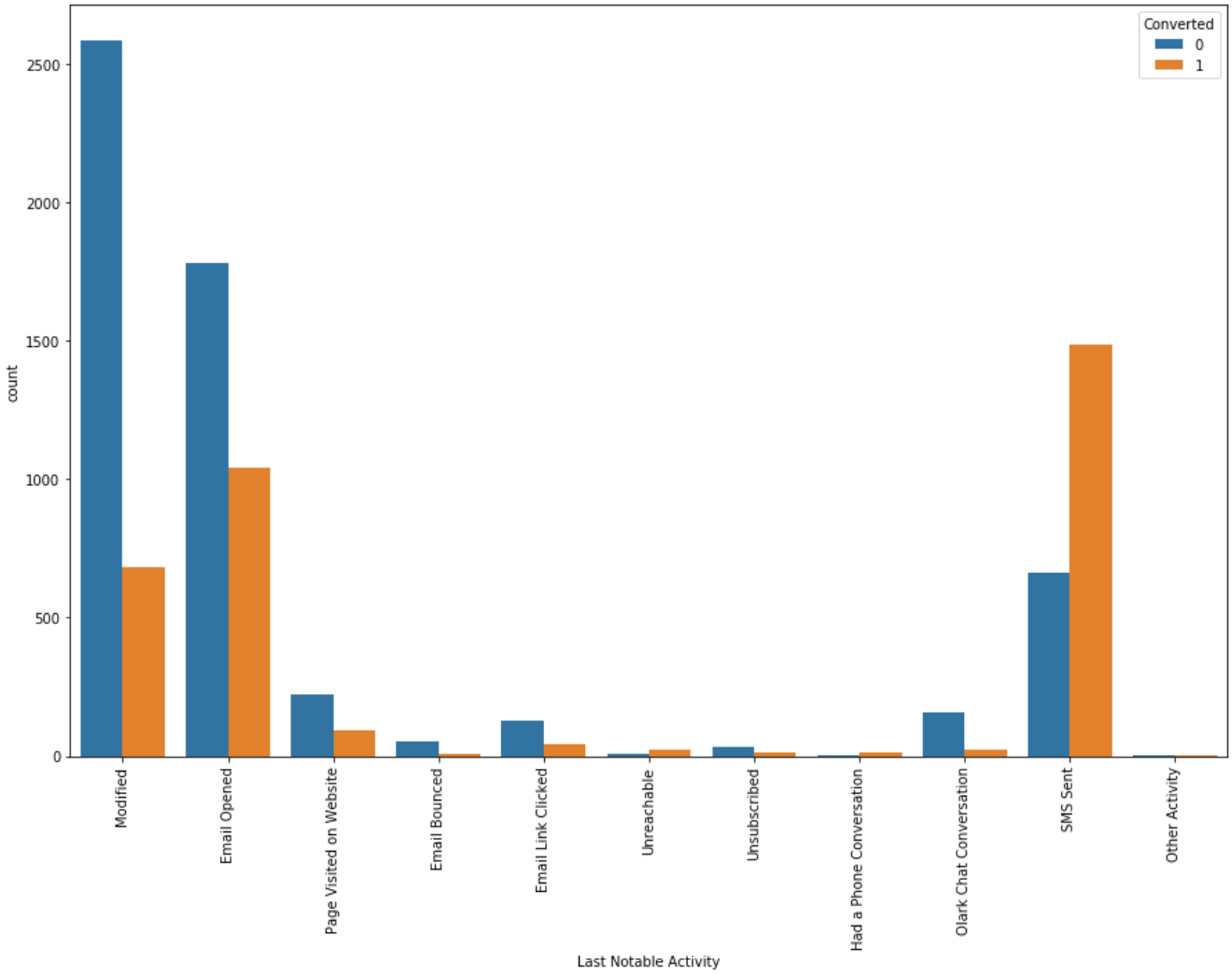
```
upfront','Resubscribed to emails','View in browser link Clicked','Form Submitted on Website','Emai
l Received','Email Marked Spam'],'Other Activity')
```

In [115]:

```
plt.figure(figsize=(15,10))
sns.countplot(x='Last Notable Activity',hue='Converted',data=lead)
plt.xticks(rotation=90)
```

Out[115]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10]),
 <a list of 11 Text xticklabel objects>)
```



In [116]:

```
lead=lead.drop(['Lead Number','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','Update me on Supply Chain Content','Get
updates on DM Content','Country','I agree to pay the amount through cheque','A free copy of Master
ing The Interview'],1)
```

In [117]:

```
lead.shape
```

Out[117]:

```
(9074, 16)
```

In [118]:

```
lead.head()
```

Out[118]:

	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Specialization	What is your current occupation	
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	API	Olark Chat	No	No	0	0.0	0	0.0	Page Visited on Website	Other Specialization	Unemployed	In
1	2a272436-5132-4136-86fa-dcc88c88f482	API	Organic Search	No	No	0	5.0	674	2.5	Email Opened	Other Specialization	Unemployed	
2	8cc8c611-a219-4f35-ad23-fdfd2656bd8a	Landing Page Submission	Direct Traffic	No	No	1	2.0	1532	2.0	Email Opened	Business Administration	Student	W th
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	Landing Page Submission	Direct Traffic	No	No	0	1.0	305	1.0	Unreachable	Media and Advertising	Unemployed	
4	3256f628-e534-4826-9d63-4a8b88782852	Landing Page Submission	Google	No	No	1	2.0	1428	1.0	Converted to Lead	Other Specialization	Unemployed	W th

Step4:Split the data into test and train data

In [119]:

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

In [120]:

```
X=lead.drop(['Prospect ID','Converted'],axis=1)
```

In [121]:

```
X.head()
```

Out[121]:

	Lead Origin	Lead Source	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City
0	API	Olark Chat	No	No	0.0	0	0.0	Page Visited on Website	Other Specialization	Unemployed	Interested in other courses	Low in Relevance	Mumba
1	API	Organic Search	No	No	5.0	674	2.5	Email Opened	Other Specialization	Unemployed	Ringing	Not Sure	Mumba
2	Landing Page Submission	Direct Traffic	No	No	2.0	1532	2.0	Email Opened	Business Administration	Student	Will revert after reading the email	Might be	Mumba
3	Landing Page Submission	Direct Traffic	No	No	1.0	305	1.0	Unreachable	Media and Advertising	Unemployed	Ringing	Not Sure	Mumba
4	Landing Page Submission	Google	No	No	2.0	1428	1.0	Converted to Lead	Other Specialization	Unemployed	Will revert after reading the email	Might be	Mumba

In [122]:

```
y=lead['Converted']
```

In [123]:

```
y.head()
```

Out[123]:

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

In [124]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.7,test_size=0.3,random_state=100)
```

In [125]:

```
X_train.head()
```

Out[125]:

	Lead Origin	Lead Source	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City
3009	Landing Page Submission	Direct Traffic	No	No	2.0	397	2.0	Email Opened	Finance Management	Unemployed	Will revert after reading the email	Not Sure	Mumbai
1012	Landing Page Submission	Direct Traffic	Yes	No	2.0	190	2.0	Email Bounced	Other Specialization	Working Professional	Will revert after reading the email	Not Sure	Mumbai
9226	API	Olark Chat	No	No	0.0	0	0.0	SMS Sent	Other Specialization	Unemployed	Ringing	Not Sure	Mumbai
4750	Landing Page Submission	Direct Traffic	No	No	2.0	1380	2.0	SMS Sent	Marketing Management	Unemployed	Will revert after reading the email	Not Sure	Other Cities
7987	Landing Page Submission	Direct Traffic	No	No	5.0	1584	2.5	SMS Sent	Finance Management	Unemployed	Lost to EINS	Not Sure	Other Cities of Maharashtra

In [126]:

```
X.head()
```

Out[126]:

	Lead Origin	Lead Source	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City
0	API	Olark Chat	No	No	0.0	0	0.0	Page Visited on Website	Other Specialization	Unemployed	Interested in other courses	Low in Relevance	Mumba
1	API	Organic Search	No	No	5.0	674	2.5	Email Opened	Other Specialization	Unemployed	Ringing	Not Sure	Mumba
	Landing										Will revert		

2	Landing Page Submission	Direct Traffic	No	No	2.0	1532	2.0	Email Opened	Business Administration	Student	after reading the email	Might be	Mumba
	Lead Origin	**Lead Source**	**Do Not Email**	**Do Not Call**	**TotalVisits**	**Total Time Spent on Website**	**Page Views Per Visit**	**Last Activity**	**Specialization**	**What is your current occupation**	**Tags**	**Lead Quality**	**City**
3	Landing Page Submission	Direct Traffic	No	No	1.0	305	1.0	Unreachable	Media and Advertising	Unemployed	Ringing	Not Sure	Mumba
4	Landing Page Submission	Google	No	No	2.0	1428	1.0	Converted to Lead	Other Specialization	Unemployed	Will revert after reading the email	Might be	Mumba

In [127]:

```
lead.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9074 entries, 0 to 9239
Data columns (total 16 columns):
Prospect ID                              9074 non-null object
Lead Origin                              9074 non-null object
Lead Source                              9074 non-null object
Do Not Email                             9074 non-null object
Do Not Call                              9074 non-null object
Converted                                9074 non-null int64
TotalVisits                              9074 non-null float64
Total Time Spent on Website              9074 non-null int64
Page Views Per Visit                     9074 non-null float64
Last Activity                            9074 non-null object
Specialization                           9074 non-null object
What is your current occupation  9074 non-null object
Tags                                     9074 non-null object
Lead Quality                             9074 non-null object
City                                     9074 non-null object
Last Notable Activity                    9074 non-null object
dtypes: float64(2), int64(2), object(12)
memory usage: 1.5+ MB
```

In [128]:

```
lead.shape
```

Out[128]:

```
(9074, 16)
```

Step5: Data Preparation

In [129]:

```
from sklearn import preprocessing

# encode categorical variables using Label Encoder

# select all categorical variables
lead_categorical=lead.select_dtypes(include=['object'])
lead_categorical.head()
```

Out[129]:

	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City	Last Notable Activity
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	API	Olark Chat	No	No	Page Visited on Website	Other Specialization	Unemployed	Interested in other courses	Low in Relevance	Mumbai	Modified
1	2a272436-5132-4136-86fa-dcc88c88f482	API	Organic Search	No	No	Email Opened	Other Specialization	Unemployed	Ringing	Not Sure	Mumbai	Email Opened
	8cc8c611-								Will revert			

	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City	Last Notable Activity
2	a219-4f35-ad23-fdfd2656bd8a	Landing Page Submission	Direct Traffic	No	No	Email Opened	Business Administration	Student	after reading the email	Might be	Mumbai	Email Opened
3	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	Landing Page Submission	Direct Traffic	No	No	Unreachable	Media and Advertising	Unemployed	Ringing	Not Sure	Mumbai	Modified
4	3256f628-e534-4826-9d63-4a8b88782852	Landing Page Submission	Google	No	No	Converted to Lead	Other Specialization	Unemployed	Will revert after reading the email	Might be	Mumbai	Modified

In [130]:

```
# appky label encoder to lead_categorical

le=preprocessing.LabelEncoder()
lead_categorical=lead_categorical.apply(le.fit_transform)
lead_categorical.head()
```

Out[130]:

	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City	Last Notable Activity
0	4332	0	4	0	0	7	13	4	4	1	0	4
1	1527	0	5	0	0	3	13	4	8	3	0	2
2	5034	1	1	0	0	3	1	3	9	2	0	2
3	462	1	1	0	0	9	11	4	8	3	0	4
4	1842	1	3	0	0	0	13	4	9	2	0	4

In [131]:

```
# concat lead_categorical with original lead

lead = lead.drop(lead_categorical.columns, axis=1)
lead = pd.concat([lead, lead_categorical], axis=1)
lead.head()
```

Out[131]:

	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality
0	0	0.0	0	0.0	4332	0	4	0	0	7	13	4	4	1
1	0	5.0	674	2.5	1527	0	5	0	0	3	13	4	8	3
2	1	2.0	1532	2.0	5034	1	1	0	0	3	1	3	9	2
3	0	1.0	305	1.0	462	1	1	0	0	9	11	4	8	3
4	1	2.0	1428	1.0	1842	1	3	0	0	0	13	4	9	2

In [132]:

```
lead.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9074 entries, 0 to 9239
Data columns (total 16 columns):
Converted                          9074 non-null int64
TotalVisits                        9074 non-null float64
Total Time Spent on Website        9074 non-null int64
Page Views Per Visit               9074 non-null float64
Prospect ID                        9074 non-null int32
Lead Origin                        9074 non-null int32
Lead Source                        9074 non-null int32
Do Not Email                       9074 non-null int32
```

```
Do Not Call                           9074 non-null int32
Last Activity                         9074 non-null int32
Specialization                        9074 non-null int32
What is your current occupation  9074 non-null  int32
Tags                                  9074 non-null int32
Lead Quality                          9074 non-null int32
City                                  9074 non-null int32
Last Notable Activity                 9074 non-null int32
dtypes: float64(2), int32(12), int64(2)
memory usage: 1.1 MB
```

In [135]:

```
lead['Converted']=lead['Converted'].astype('category')
```

Step6: model building and evaluation

In [136]:

```
# Importing train-test-split
from sklearn.model_selection import train_test_split
```

In [138]:

```
# Putting feature variable to X
X = lead.drop('Converted',axis=1)

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

In [139]:

```
# Splitting the data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.30,
                                                    random_state = 99)

X_train.head()
```

Out[139]:

	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Prospect ID	Lead Origin	Lead Source	Do Not Email	Do Not Call	Last Activity	Specialization	What is your current occupation	Tags	Lead Quality	City	Not Act
3534	3.0	129	3.0	2985	1	5	0	0	3	12	4	11	3	0	
2358	2.0	240	2.0	3059	1	3	0	0	8	4	5	9	2	0	
1830	3.0	226	1.5	6074	0	4	0	0	3	7	4	8	3	0	
1647	2.0	1184	2.0	1821	1	6	0	0	3	10	4	9	1	4	
6254	2.0	31	2.0	8917	0	3	0	0	5	13	4	4	3	0	

In [140]:

```
# Importing decision tree classifier from sklearn library
from sklearn.tree import DecisionTreeClassifier

# Fitting the decision tree with default hyperparameters, apart from
# max_depth which is 5 so that we can plot and read the tree.
dt_default = DecisionTreeClassifier(max_depth=5)
dt_default.fit(X_train, y_train)
```

Out[140]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=5,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
```

```
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [141]:

```
# Let's check the evaluation metrics of our default model

# Importing classification report and confusion matrix from sklearn metrics
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

# Making predictions
y_pred_default = dt_default.predict(X_test)

# Printing classification report
print(classification_report(y_test, y_pred_default))
```

```
              precision    recall  f1-score   support

           0       0.88      0.95      0.92      1699
           1       0.91      0.79      0.85      1024

   micro avg       0.89      0.89      0.89      2723
   macro avg       0.90      0.87      0.88      2723
weighted avg       0.89      0.89      0.89      2723
```

In [142]:

```
# Printing confusion matrix and accuracy
print(confusion_matrix(y_test,y_pred_default))
print(accuracy_score(y_test,y_pred_default))
```

```
[[1619   80]
 [ 214  810]]
0.8920308483290489
```

Step7: Plotting the Decision Tree

In [143]:

```
from IPython.display import Image
from sklearn.externals.six import StringIO
from sklearn.tree import export_graphviz
import pydotplus, graphviz

# Putting features
features =list(lead.columns[1:])
features
```

Out[143]:

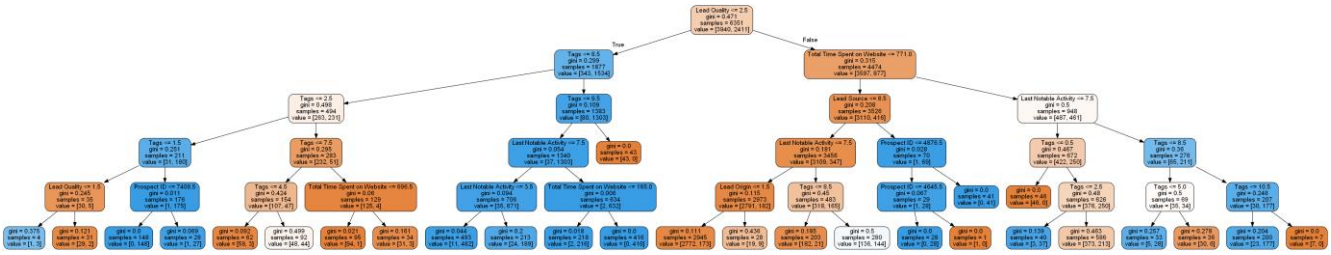
```
['TotalVisits',
 'Total Time Spent on Website',
 'Page Views Per Visit',
 'Prospect ID',
 'Lead Origin',
 'Lead Source',
 'Do Not Email',
 'Do Not Call',
 'Last Activity',
 'Specialization',
 'What is your current occupation',
 'Tags',
 'Lead Quality',
 'City',
 'Last Notable Activity']
```

In [144]:

```
# Plotting tree with max_depth=3
dot_data = StringIO()
export_graphviz(dt_default,  out_file=dot_data,
                feature_names=features, filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

Out[144]:



Step8: Hyperparameter Tuning

In [145]:

```
#The default tree is quite complex, and we need to simplify it by tuning the hyperparameters.
#criterion
#splitter
#max_features
#max_depth
#min_samples_split
#min_sapmles_leaf
#max_leaf_nodes
#min_impurity_split
```

Tuning max_deapth

In [146]:

```
# GridSearchCV to find optimal max_depth
from sklearn.model_selection import KFold
from sklearn.model_selection import GridSearchCV


# specify number of folds for k-fold CV
n_folds = 5

# parameters to build the model on
parameters = {'max_depth': range(1, 40)}

# instantiate the model
dtree = DecisionTreeClassifier(criterion = "gini",
                               random_state = 100)

# fit tree on training data
tree = GridSearchCV(dtree, parameters,
                    cv=n_folds,
                   scoring="accuracy")
tree.fit(X_train, y_train)
```

Out[146]:

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=100,
            splitter='best'),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'max_depth': range(1, 40)}, pre_dispatch='2*n_jobs',
       refit=True, return_train_score='warn', scoring='accuracy',
       verbose=0)
```

In [148]:

```
scores=tree.cv_results_
pd.DataFrame(scores).head()
```

Out[148]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_max_depth	params	split0_test_score	split1_test_score
0	0.008744	0.000608	0.001598	0.001957	1	{'max_depth': 1}	0.826121	0.800787
1	0.011261	0.002148	0.002644	0.002199	2	{'max_depth': 2}	0.825334	0.811024
2	0.013794	0.002558	0.002024	0.001787	3	{'max_depth': 3}	0.885130	0.884252
3	0.015114	0.002902	0.003373	0.006746	4	{'max_depth': 4}	0.888277	0.883465
4	0.021309	0.005463	0.000000	0.000000	5	{'max_depth': 5}	0.892998	0.903937

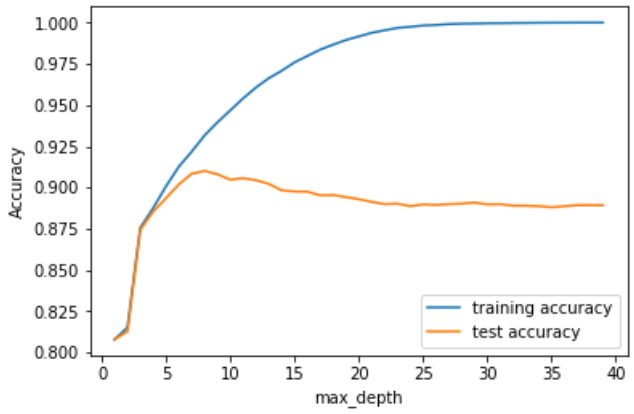
5 rows × 21 columns

In [149]:

```
#now lets visualize how tarin and test score changes with max_depth
```

In [150]:

```
# plotting accuracies with max_depth
plt.figure()
plt.plot(scores["param_max_depth"],
         scores["mean_train_score"],
         label="training accuracy")
plt.plot(scores["param_max_depth"],
         scores["mean_test_score"],
         label="test accuracy")
plt.xlabel("max_depth")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```



Tuning min_samples_leaf

In [151]:

```
from sklearn.model_selection import KFold
from sklearn.model_selection import GridSearchCV


# specify number of folds for k-fold CV
n_folds = 5
```

```
# parameters to build the model on
parameters = {'min_samples_leaf': range(5, 200, 20)}

# instantiate the model
dtree = DecisionTreeClassifier(criterion = "gini",
                               random_state = 100)

# fit tree on training data
tree = GridSearchCV(dtree, parameters,
                    cv=n_folds,
                   scoring="accuracy")
tree.fit(X_train, y_train)
```

Out[151]:

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=100,
            splitter='best'),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'min_samples_leaf': range(5, 200, 20)},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring='accuracy', verbose=0)
```

In [152]:

```
# scores of GridSearch CV
scores = tree.cv_results_
pd.DataFrame(scores).head()
```

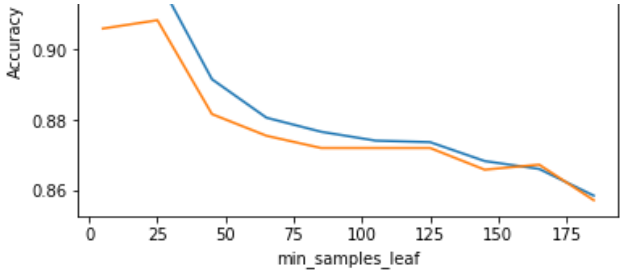
Out[152]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_min_samples_leaf	params	split0_test_score	split1
0	0.034250	0.003295	0.002566	0.002542	5	{'min_samples_leaf': 5}	0.901652	
1	0.034372	0.009674	0.006375	0.007411	25	{'min_samples_leaf': 25}	0.905586	
2	0.025424	0.007160	0.009374	0.007654	45	{'min_samples_leaf': 45}	0.896145	
3	0.028206	0.008745	0.003123	0.006247	65	{'min_samples_leaf': 65}	0.881196	
4	0.019212	0.006076	0.009621	0.007869	85	{'min_samples_leaf': 85}	0.874902	

5 rows × 21 columns

In [153]:

```
# plotting accuracies with min_samples_leaf
plt.figure()
plt.plot(scores["param_min_samples_leaf"],
         scores["mean_train_score"],
         label="training accuracy")
plt.plot(scores["param_min_samples_leaf"],
         scores["mean_test_score"],
         label="test accuracy")
plt.xlabel("min_samples_leaf")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```



Tuning min_samples_split

In [154]:

```
# GridSearchCV to find optimal min_samples_split
from sklearn.model_selection import KFold
from sklearn.model_selection import GridSearchCV


# specify number of folds for k-fold CV
n_folds = 5

# parameters to build the model on
parameters = {'min_samples_split': range(5, 200, 20)}

# instantiate the model
dtree = DecisionTreeClassifier(criterion = "gini",
                               random_state = 100)

# fit tree on training data
tree = GridSearchCV(dtree, parameters,
                    cv=n_folds,
                   scoring="accuracy")
tree.fit(X_train, y_train)
```

Out[154]:

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=100,
            splitter='best'),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'min_samples_split': range(5, 200, 20)},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring='accuracy', verbose=0)
```

In [155]:

```
# scores of GridSearch CV
scores = tree.cv_results_
pd.DataFrame(scores).head()
```

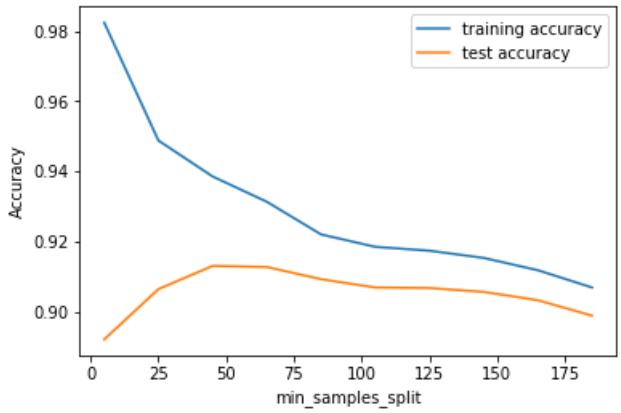
Out[155]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_min_samples_split	params	split0_test_score	split1
0	0.037003	0.004879	0.002812	0.001522	5	{'min_samples_split': 5}	0.892998	
1	0.042120	0.007504	0.000479	0.000959	25	{'min_samples_split': 25}	0.910307	
2	0.034365	0.006248	0.000000	0.000000	45	{'min_samples_split': 45}	0.913454	
3	0.035995	0.005619	0.000198	0.000396	65	{'min_samples_split': 65}	0.906373	
4	0.038323	0.007054	0.000000	0.000000	85	{'min_samples_split': 85}	0.910307	

5 rows × 21 columns

In [156]:

```
# plotting accuracies with min_samples_leaf
plt.figure()
plt.plot(scores["param_min_samples_split"],
         scores["mean_train_score"],
         label="training accuracy")
plt.plot(scores["param_min_samples_split"],
         scores["mean_test_score"],
         label="test accuracy")
plt.xlabel("min_samples_split")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```



Step9:Grid search to find optimal Hyperparameters

In [157]:

```
# Create the parameter grid
param_grid = {
    'max_depth': range(5, 15, 5),
    'min_samples_leaf': range(50, 150, 50),
    'min_samples_split': range(50, 150, 50),
    'criterion': ["entropy", "gini"]
}

n_folds = 5

# Instantiate the grid search model
dtree = DecisionTreeClassifier()
grid_search = GridSearchCV(estimator = dtree, param_grid = param_grid,
                           cv = n_folds, verbose = 1)

# Fit the grid search to the data
grid_search.fit(X_train,y_train)
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  80 out of  80 | elapsed:    2.5s finished
```

Out[157]:

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best'),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'max_depth': range(5, 15, 5), 'min_samples_leaf': range(50, 150, 50),
```

```
'min_samples_split': range(50, 150, 50), 'criterion': ['entropy', 'gini']},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=1)
```

In [158]:

```
# cv results
cv_results = pd.DataFrame(grid_search.cv_results_)
cv_results
```

Out[158]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_criterion	param_max_depth	param_min_samples_leaf	param
0	0.022641	0.001993	0.002417	0.002282	entropy	5	50	
1	0.023984	0.001949	0.003856	0.002312	entropy	5	50	
2	0.015973	0.000563	0.007790	0.007316	entropy	5	100	
3	0.031233	0.000020	0.000000	0.000000	entropy	5	100	
4	0.032712	0.007801	0.004202	0.006507	entropy	10	50	
5	0.038530	0.008480	0.003125	0.006249	entropy	10	50	
6	0.034590	0.003071	0.000369	0.000739	entropy	10	100	
7	0.027876	0.006559	0.003925	0.006048	entropy	10	100	
8	0.023289	0.004937	0.000204	0.000409	gini	5	50	
9	0.018750	0.006248	0.003118	0.006235	gini	5	50	
10	0.025696	0.006065	0.000000	0.000000	gini	5	100	
11	0.016314	0.000901	0.003122	0.006244	gini	5	100	
12	0.025053	0.005502	0.004247	0.005555	gini	10	50	
13	0.026555	0.003824	0.004078	0.002085	gini	10	50	

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_criterion	param_max_depth	param_min_samples_leaf	param
14	0.024874	0.002681	0.001630	0.001236	gini	10	100	
15	0.023378	0.001043	0.000200	0.000399	gini	10	100	

16 rows × 24 columns

In [159]:

```
# printing the optimal accuracy score and hyperparameters
print("best accuracy", grid_search.best_score_)
print(grid_search.best_estimator_)
```

```
best accuracy 0.8784443394740986
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=10,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=50, min_samples_split=50,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

Running the model with best parameter obtained from grid search

In [160]:

```
# model with optimal hyperparameters
clf_gini = DecisionTreeClassifier(criterion = "gini",
                                  random_state = 100,
                                  max_depth=10,
                                  min_samples_leaf=50,
                                  min_samples_split=50)
clf_gini.fit(X_train, y_train)
```

Out[160]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=10,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=50, min_samples_split=50,
            min_weight_fraction_leaf=0.0, presort=False, random_state=100,
            splitter='best')
```

In [161]:

```
# accuracy score
clf_gini.score(X_test,y_test)
```

Out[161]:

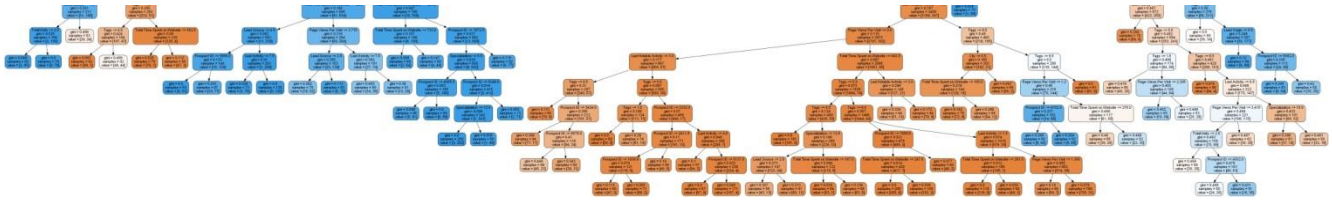
```
0.8872567021667279
```

In [162]:

```
# plotting the tree
dot_data = StringIO()
export_graphviz(clf_gini, out_file=dot_data,feature_names=features,filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

Out[162]:



You can see that this tree is too complex to understand. Let's try reducing the max_depth and see how the tree looks.

In [163]:

```
# tree with max_depth = 3
clf_gini = DecisionTreeClassifier(criterion = "gini",
                                  random_state = 100,
                                  max_depth=3,
                                  min_samples_leaf=50,
                                  min_samples_split=50)
clf_gini.fit(X_train, y_train)

# score
print(clf_gini.score(X_test,y_test))
```

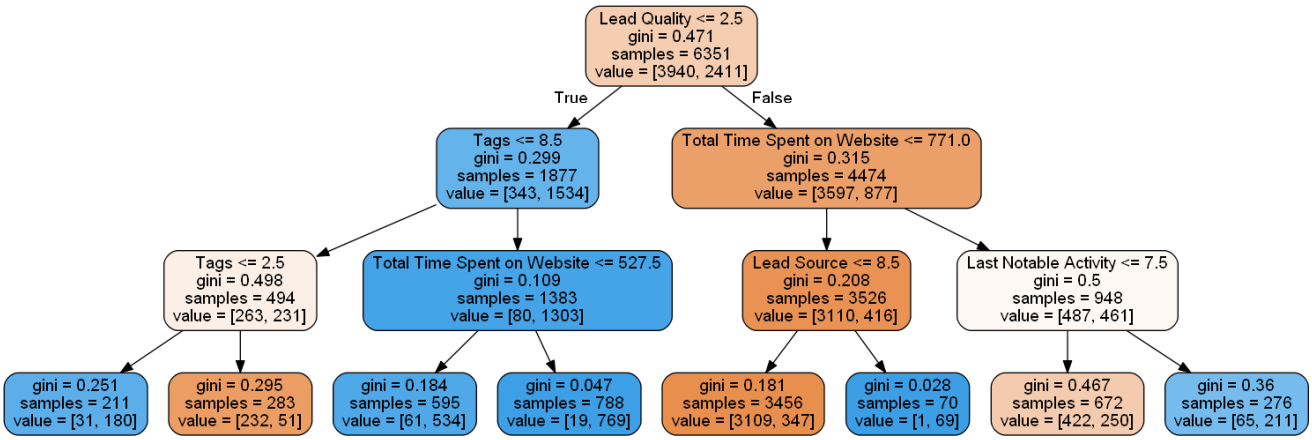
```
0.8615497612926919
```

In [164]:

```
# plotting tree with max_depth=3
dot_data = StringIO()
export_graphviz(clf_gini, out_file=dot_data,feature_names=features,filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

Out[164]:



In [165]:

```
# classification metrics
from sklearn.metrics import classification_report,confusion_matrix
y_pred = clf_gini.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.84      0.95      0.90      1699
           1       0.90      0.71      0.79      1024

   micro avg       0.86      0.86      0.86      2723
   macro avg       0.87      0.83      0.84      2723
weighted avg       0.87      0.86      0.86      2723
```

In [166]:

```
# confusion matrix
print(confusion_matrix(y_test,y_pred))
```

```
[[1620   79]
 [ 298  726]]
```

In [ ]:

```

```