LEADS SCORING CASE STUDY

CASSADY MCDONALD & MANICKAVEL

BUSINESS OBJECTIVES

X Education needs a model to select the most promising leads, i.e. the leads that are most likely to convert into paying customers. The company requires to build a model wherein a lead score will be assigned 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%.

HANDLING MISSING VALUES

- NAN (missing value) which if below 50% was imputed based on median value if its continuous variable. For categorical values mode was used to replace the missing value
- ‘Select’ Option was grouped as Other/Unknown based on the column details

HANDLING MISSING VALUES (CONT.)

Missing Value Analysis

In [196]:
```
(Leads.isnull().sum().sort_values(ascending = False)/len(Leads.index))*100
```

Out[196]:
```
What is your current occupation           29.112554
Country                                   26.634199
Specialization                            15.562771
City                                      15.367965
Page Views Per Visit                       1.482684
TotalVisits                                1.482684
Lead Source                                0.389610
A free copy of Mastering The Interview     0.000000
Total Time Spent on Website                0.000000
Converted                                  0.000000
Lead Origin                                0.000000
dtype: float64
```

SPECIALIZATION

```
# Specialization with NAN and its corresponding current occupation
Leads[Leads['Specialization'].isnull()] ['What is your current occupation'].value_counts(dropna=False)
```

```
NaN                     1420
Student                   13
Working Professional       4
Businessman                1
Name: What is your current occupation, dtype: int64
```

- When 'Specialization' is with NAN and its corresponding 'current occupation' is also NAN then we can replace those with Other.
- For 'Specialization' with NAN and has value in 'current occupation' then we can find mode in such a way that- Specialization which has higest count for that 'current occupation'.

```
# Specialization with Select and its corresponding current occupation
Leads[Leads['Specialization'] == "Select"] ['What is your current occupation'].value_counts(dropna=False)
```

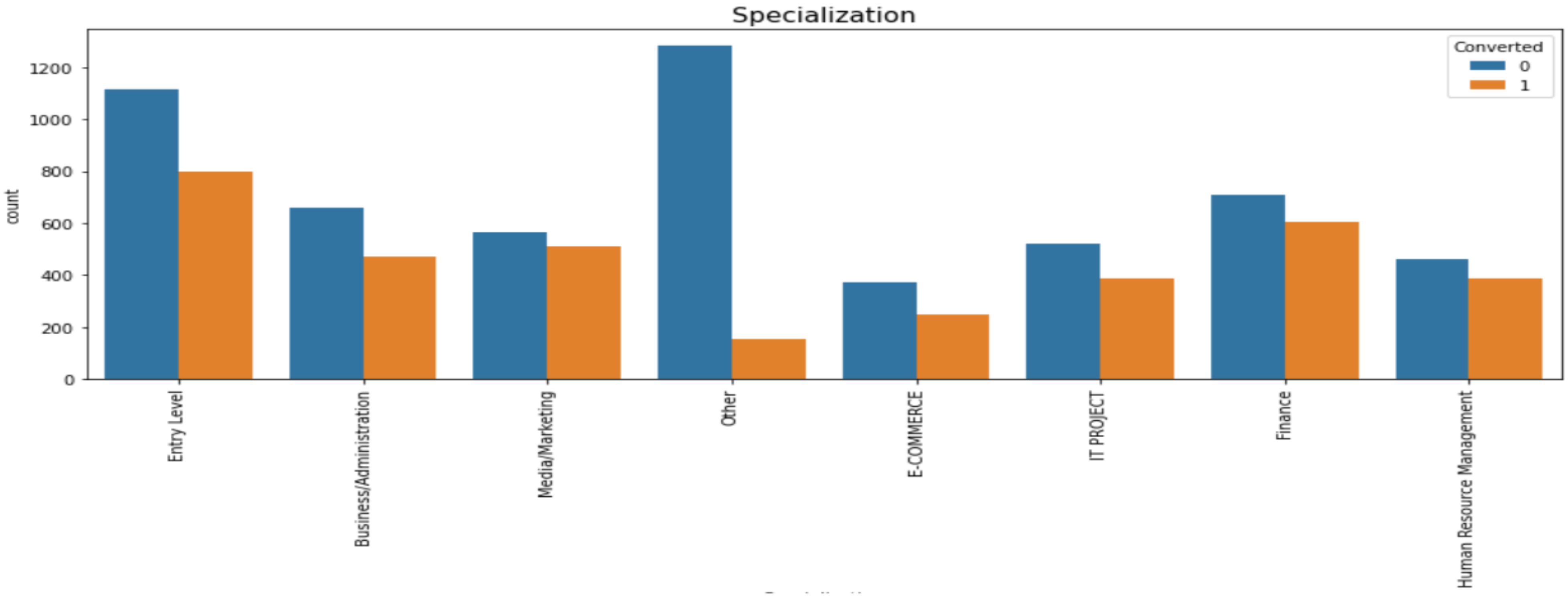
```
Unemployed              1828
Student                   72
Working Professional      27
NaN                       13
Businessman                1
Other                      1
Name: What is your current occupation, dtype: int64
```

- Unemployed & Student arent employed and hence they dont have work specalization. They can be gropued under new name 'entry level'.
- When 'Specialization' is with Select and its corresponding 'current occupation' is also NAN then we can replace those with Other.
- For 'Specialization' with Select and has value in 'current occupation' then we can find mode in such a way that- filter data for that occupation and then find mode for Specialization.

SPECIALIZATION

As few category item count are very low and to avoid model bias, we could club catgory as mentioned below

- Finance = ['Finance Management', 'Banking, Investment And Insurance']
- ECOMMERCE = ['E-COMMERCE', 'Retail Management', 'Supply Chain Management','E-Business',. 'Services Excellence']
- Business/Administration = ['Business Administration', 'International Business', 'Travel and Tourism','Rural and Agribusiness','Healthcare Management','Hospitality Management']
- Media/Marketing = ['Marketing Management', 'Media and Advertising']



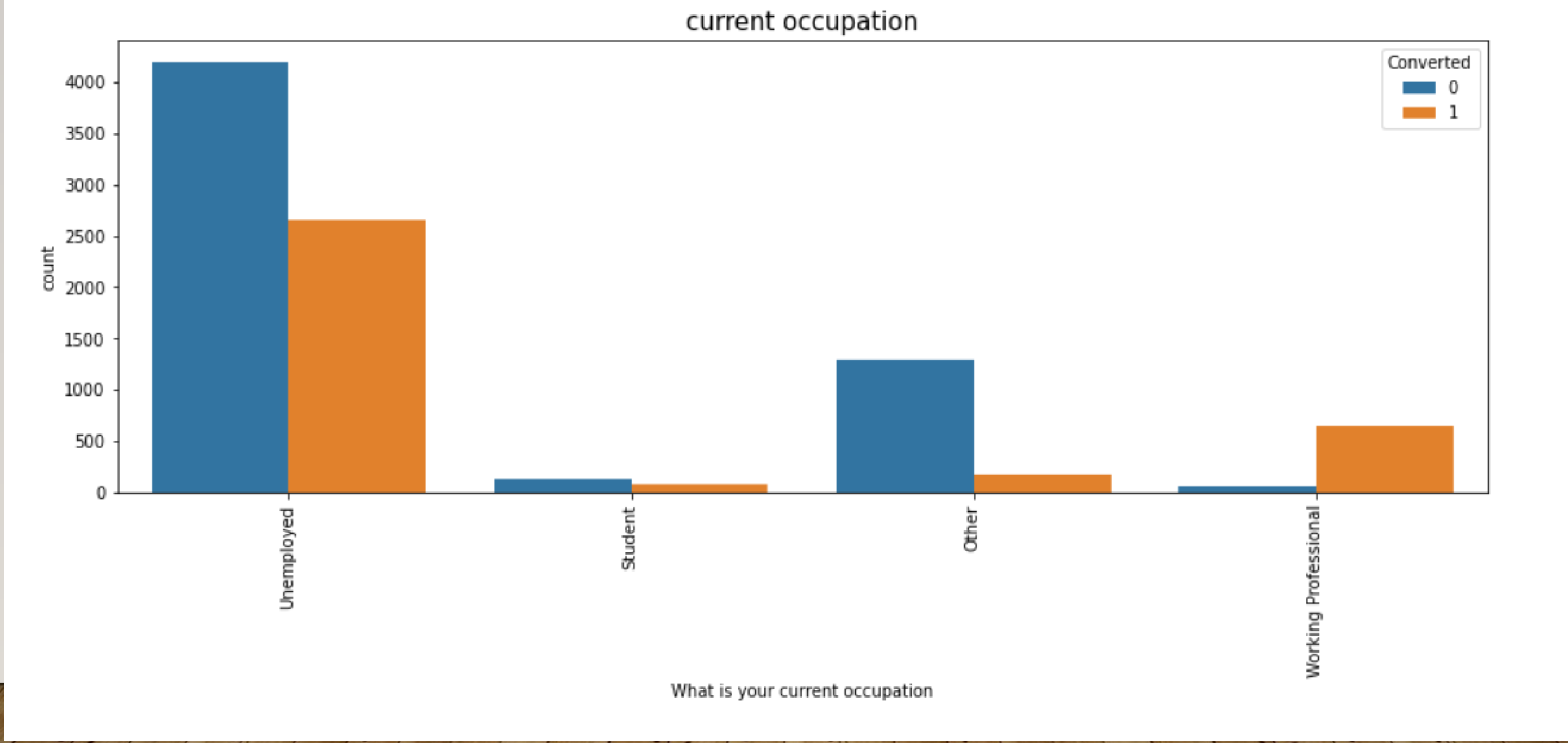
WHAT IS YOUR CURRENT OCCUPATION

```
#Check specalization for NAN value of current occupation to see if we could get more information on missing value
Leads[Leads['What is your current occupation'].isnull()].Specialization.value_counts(dropna=False)
```

```
Other                                1433
Finance Management                    214
Human Resource Management             172
Marketing Management                  160
Operations Management                 108
Business Administration                89
IT Projects Management                 88
Supply Chain Management                71
Banking, Investment And Insurance      69
Travel and Tourism                     53
Media and Advertising                  41
International Business                 40
Healthcare Management                  34
E-COMMERCE                             31
Retail Management                      22
Hospitality Management                 21
Rural and Agribusiness                 15
Services Excellence                    15
E-Business                             14
Name: Specialization, dtype: int64
```

- NAN Occupation for which has some Specialization as 'Other' can be grouped into new category Other
- current occupation which NAN and have some in 'Specialization' then we can find mode in such a way that- filter data for that Specialization and then find mode for current occupation.

CURRENT OCCUPATION



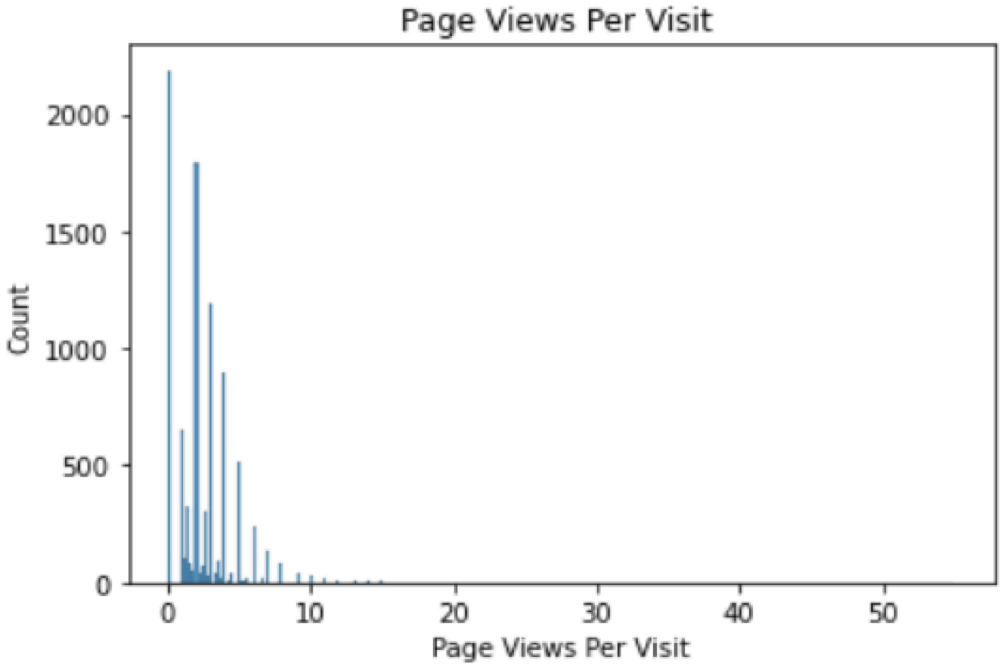
CITY

```
Leads['City'].value_counts(  dropna=False)*100
```

```
Mumbai                         322200
Select                         224900
NaN                            142000
Thane & Outskirts               75200
Other Cities                    68600
Other Cities of Maharashtra     45700
Other Metro Cities              38000
Tier II Cities                   7400
Name: City, dtype: int64
```

- Select & NAN can be grouped as 'Unknown City'
- Tier II City can be combined with other cities as count is very low

PAGE VIEWS PER VISIT



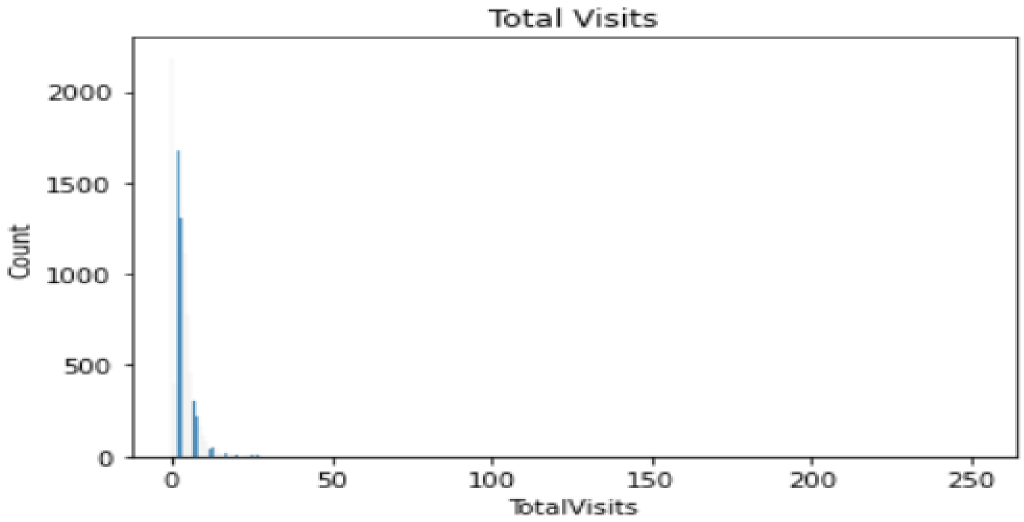
This looks to be right skewed and median can be used

```
#Replace the missing value with median
Leads['Page Views Per Visit'].fillna(value=Leads['Page Views Per Visit'].median(),inplace=True)
```

TOTAL VISITS

TotalVisits

```
sns.histplot(data =Leads, x = "TotalVisits" )
plt.title("Total Visits")
plt.show()
```

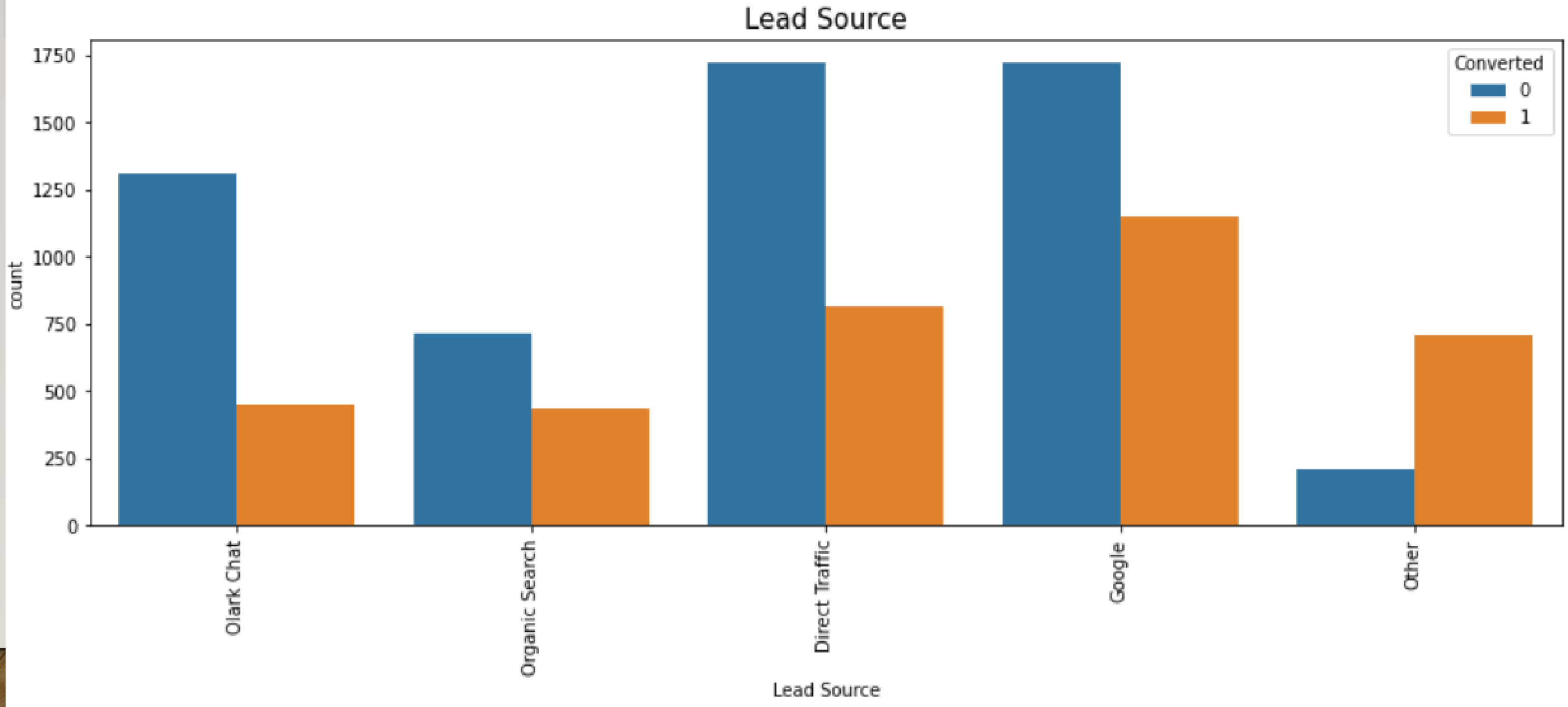


This looks to be right skewed and median can be used

```
#Replace the missing value with median
Leads['TotalVisits'].fillna(value=Leads['TotalVisits'].median(),inplace=True)
```

LEAD SOURCE

```
Other_Src = ['Direct Traffic', 'Olark Chat' ,'Google','Organic Search']
Leads['Lead Source'] = Leads['Lead Source'].apply(lambda x: 'Other' if x not in Other_Src else x  )
```



DUMMIES FOR CATEGORICAL VARIABLES

```
#Create Dummy variable for occupation
current_occupation_Dummy =  pd.get_dummies(Leads['What is your current occupation'],prefix = 'Curr_Occ_', drop_first = True)
Leads = pd.concat([Leads, current_occupation_Dummy], axis = 1)
```

```
#Create Dummy variable for Specialization
Specialization_Dummy =  pd.get_dummies(Leads['Specialization'],prefix = 'Spec_', drop_first = True)
Leads = pd.concat([Leads, Specialization_Dummy], axis = 1)
```

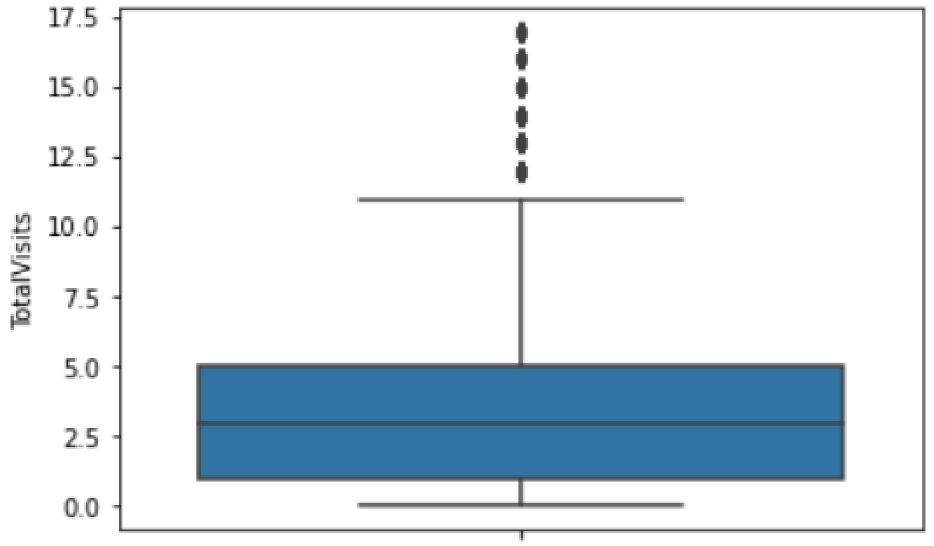
```
#Create Dummy variable for City
City_Dummy =  pd.get_dummies(Leads['City'],prefix = 'City_', drop_first = True)
Leads = pd.concat([Leads, City_Dummy], axis = 1)
```

```
#Create Dummy variable for Lead Source
LeadSource_Dummy =  pd.get_dummies(Leads['Lead Source'],prefix = 'LS_', drop_first = True)
Leads = pd.concat([Leads, LeadSource_Dummy], axis = 1)
```

```
Leads['Lead Origin'] = Leads['Lead Origin'].apply(lambda x: 'NON-API' if x not in 'API' else x)
#Create Dummy variable for Lead Origin
LeadOrigin_Dummy =  pd.get_dummies(Leads['Lead Origin'],prefix = 'LO_', drop_first = True)
Leads = pd.concat([Leads, LeadOrigin_Dummy], axis = 1)
```

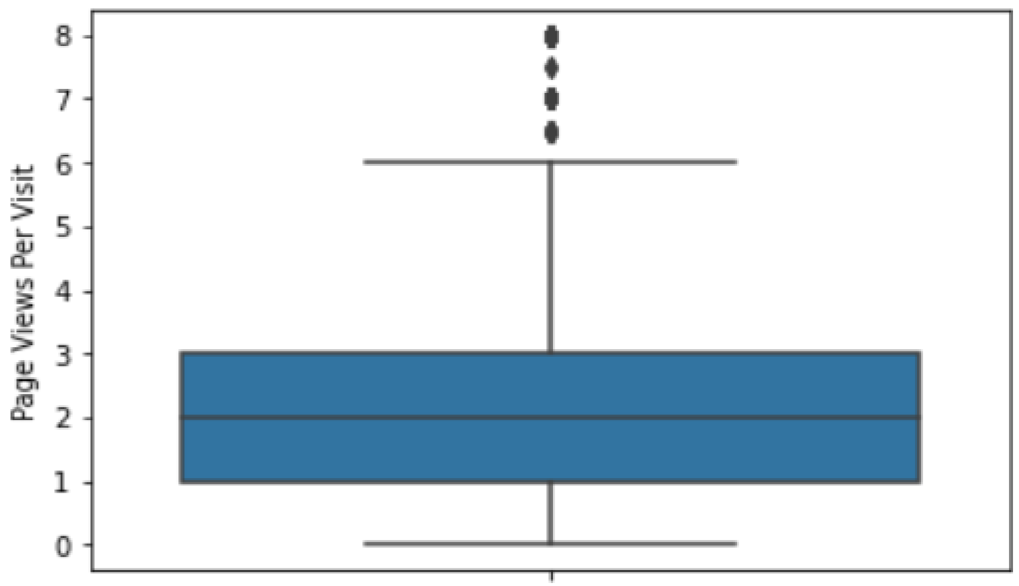
HANDLING OUTLIERS – TOTAL VISITS

```
# Handling Outliers in Total Visits
Q3 = Leads['TotalVisits'].quantile(0.99)
Leads = Leads[(Leads['TotalVisits'] <= Q3)]
Q1 = Leads['TotalVisits'].quantile(0.01)
Leads = Leads[(Leads['TotalVisits'] >= Q1)]
sns.boxplot(y=Leads['TotalVisits'])
plt.show()
```

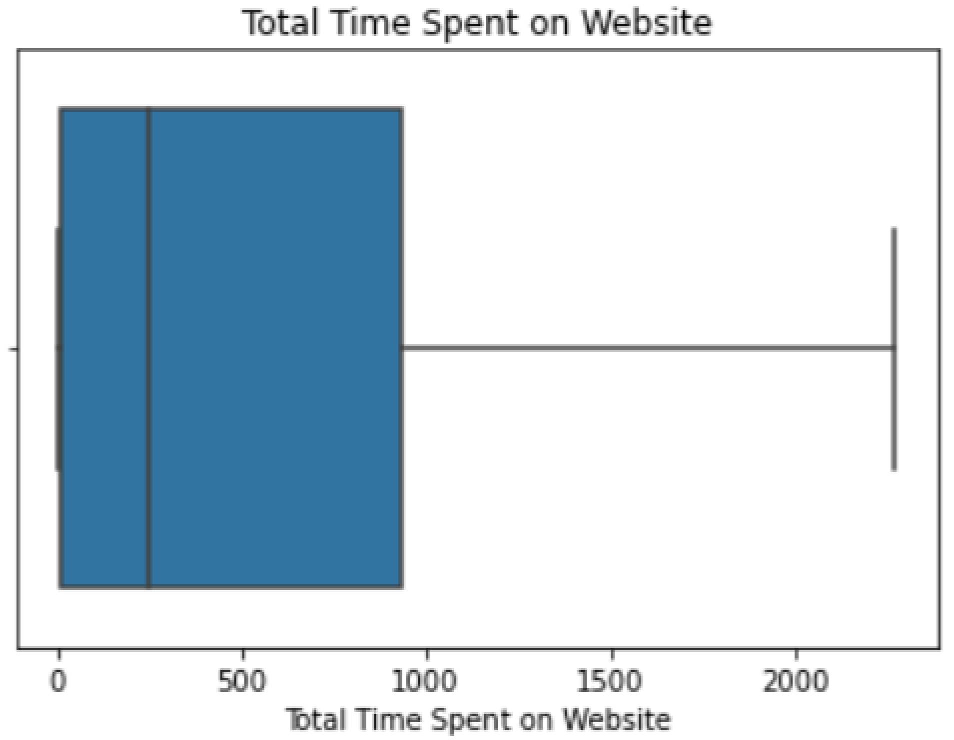


HANDLING OUTLIERS - PAGE VIEWS PER VISIT

```
# Handling Outliers in Page Views Per Visit
Q3 = Leads['Page Views Per Visit'].quantile(0.99)
Leads = Leads[(Leads['Page Views Per Visit'] <= Q3)]
Q1 = Leads['Page Views Per Visit'].quantile(0.01)
Leads = Leads[(Leads['Page Views Per Visit'] >= Q1)]
sns.boxplot(y=Leads['Page Views Per Visit'])
plt.show()
```

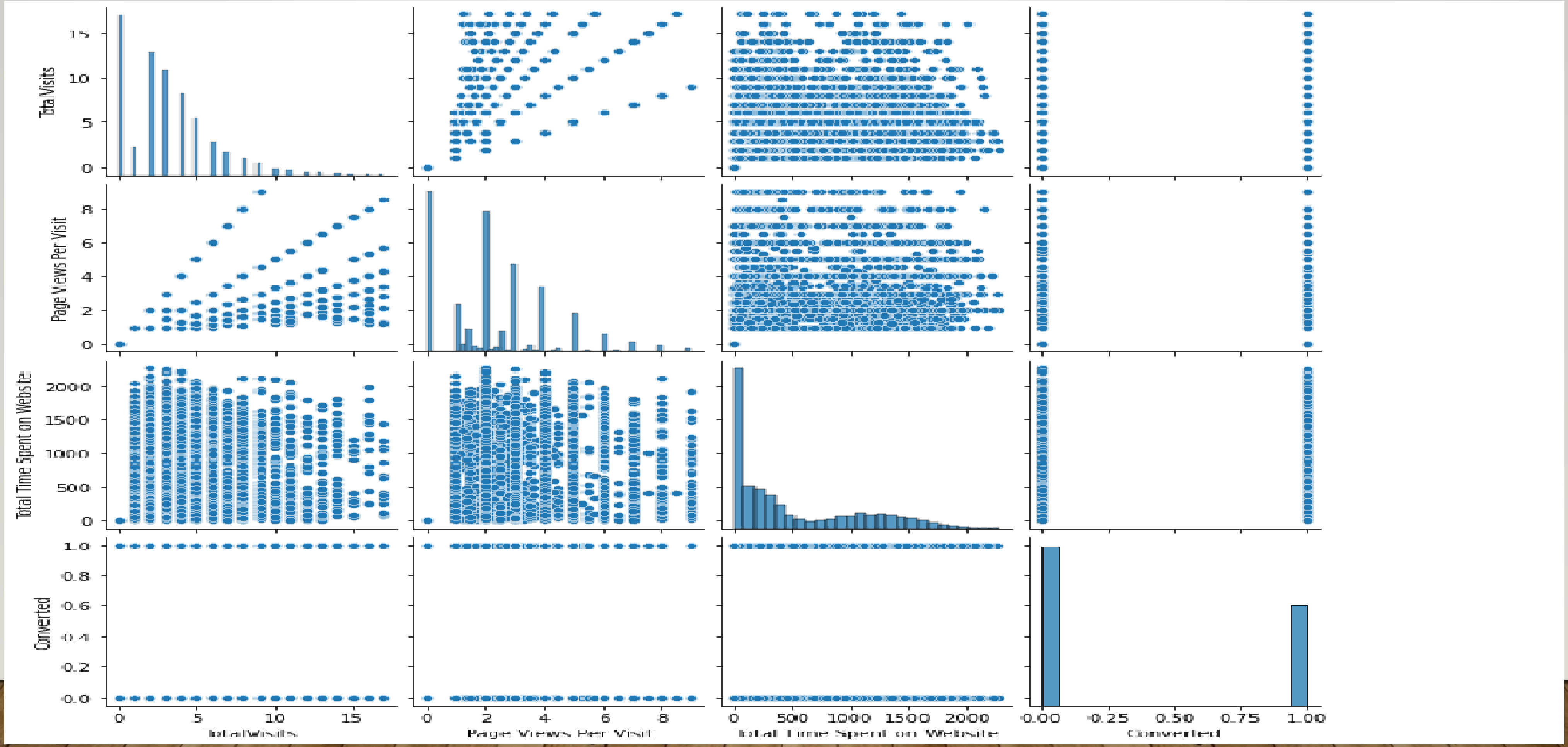


HANDLING OUTLIERS - TOTAL TIME SPENT ON WEBSITE

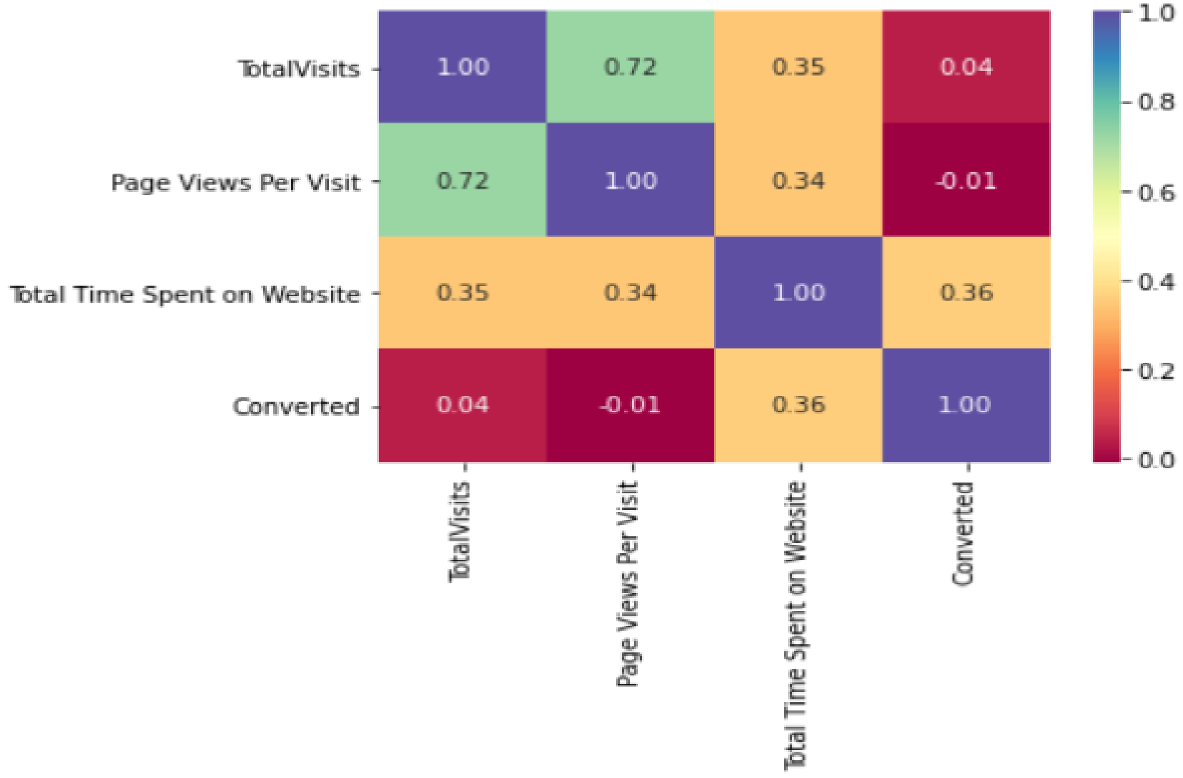


There is no outliers in Total Time Spent on Website

MULTIVARIATE ANALYSIS



CORRELATION ANALYSIS FOR NUMERICAL VALUE



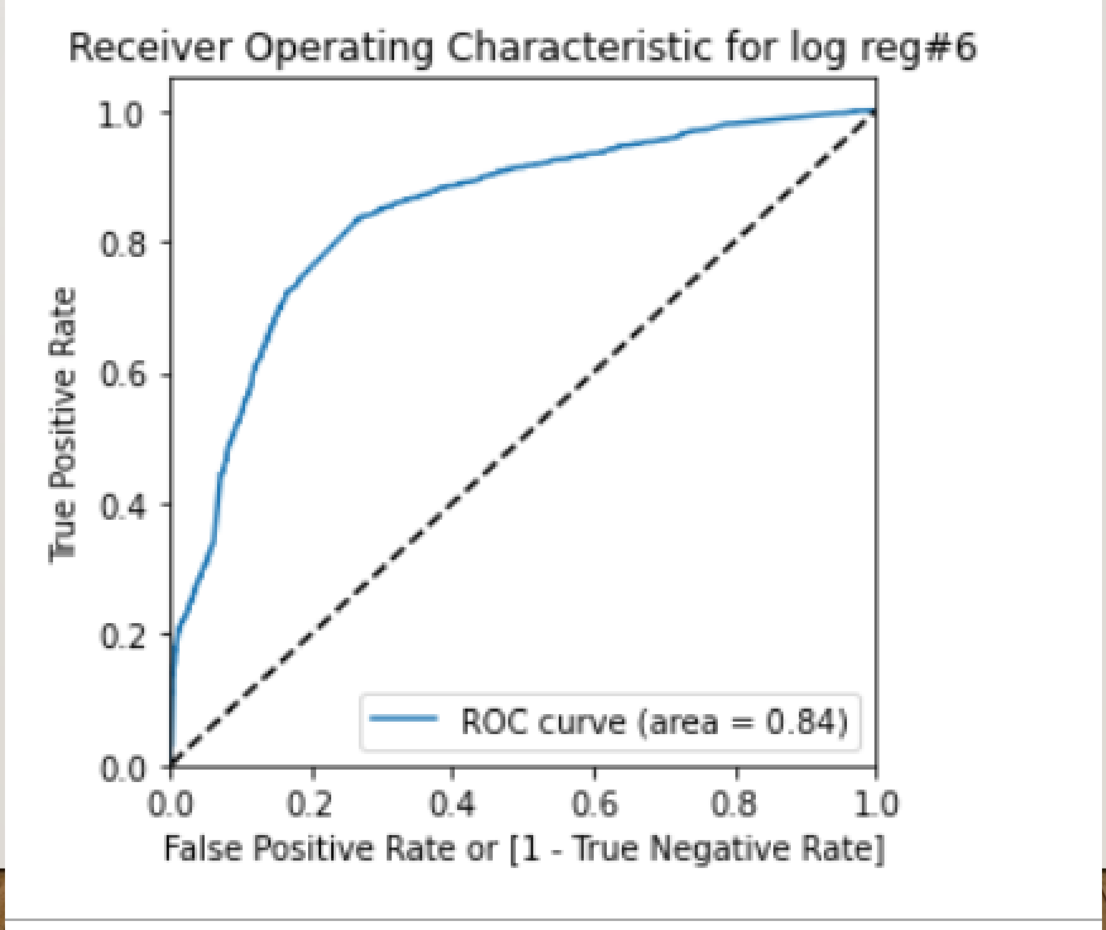
- Page view per visit and Total Visits have good correlation
- Total Time spent have better correlation with targetr variable when compared to Page view per visit and Total Visits

TOP 8 FEATURE WHICH WILL BE USED FOR MODEL BUILDING

Dep. Variable:	Converted	**No. Observations:**	6363
Model:	GLM	**Df Residuals:**	6353
Model Family:	Binomial	**Df Model:**	9
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-3043.8
Date:	Wed, 21 Jul 2021	**Deviance:**	6087.6
Time:	18:18:23	**Pearson chi2:**	6.79e+03
No. Iterations:	6		
Covariance Type:	nonrobust		

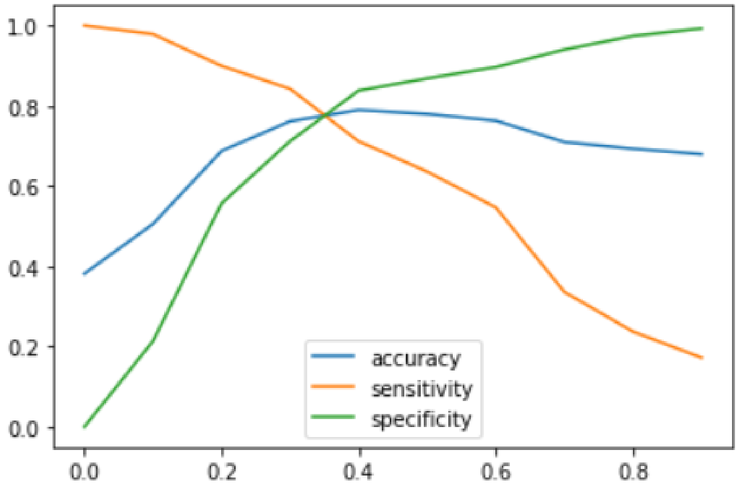
	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.8618	0.523	-1.646	0.100	-1.888	0.164
Total Time Spent on Website	4.3095	0.155	27.786	0.000	4.006	4.614
Curr_Occ__Student	-1.4903	0.555	-2.684	0.007	-2.578	-0.402
Curr_Occ__Unemployed	-1.3546	0.520	-2.606	0.009	-2.373	-0.336
Curr_Occ__Working Professional	1.3427	0.546	2.458	0.014	0.272	2.413
Spec__Other	-3.1574	0.532	-5.930	0.000	-4.201	-2.114
LS__Google	0.5182	0.082	6.292	0.000	0.357	0.680
LS__Olark Chat	1.5267	0.112	13.577	0.000	1.306	1.747
LS__Organic Search	0.3104	0.109	2.861	0.004	0.098	0.523
LS__Other	3.0478	0.130	23.411	0.000	2.793	3.303

ROC CURVE


Receiver Operating Characteristic for log reg#6
True Positive Rate
False Positive Rate or [1 - True Negative Rate]
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.84)

PROBABILITY DISTRIBUTION OF ACCURACY, SENSITIVITY, AND SPECIFICITY

	probability	accuracy	sensitivity	specificity
0.0	0.0	0.381581	1.000000	0.000000
0.1	0.1	0.505422	0.978583	0.213469
0.2	0.2	0.687412	0.899506	0.556544
0.3	0.3	0.761590	0.842257	0.711817
0.4	0.4	0.789565	0.711285	0.837865
0.5	0.5	0.779664	0.635914	0.868361
0.6	0.6	0.763005	0.546540	0.896569
0.7	0.7	0.709571	0.336079	0.940025
0.8	0.8	0.692755	0.237232	0.973825
0.9	0.9	0.679554	0.172158	0.992630



looks to be right before 0.3 is good as it has good sensitivity rate and decent accuracy & specificity

TRAIN DATASET VALIDATION

```
# confusion Matrix
confusion2 = metrics.confusion_matrix(y_train_pred_final.Converted, y_train_pred_final.final_Predicted )
confusion2
```

```
array([[2897, 1038],
       [ 412, 2016]], dtype=int64)
```

new cutoff yielded a slightly less accurate model, increased sensitivity, and decreased specificity

accuracy = 0.77 (rounded)

sensitivity = 0.83 (rounded)

specificity = 0.74 (rounded)

FPR = 0.26 (rounded)

PPR = 0.66 (rounded)

NPV = 0.88 (rounded)

precision = 0.73 (rounded)

recall = 0.71 (rounded)

TEST DATASET VALIDATION

```
## Confusion matrix
confusion = metrics.confusion_matrix(y_pred_final.Converted, y_pred_final.Predicted )
print(confusion)
```

```
[[1211  449]
 [ 195  872]]
```

accuracy = 0.76 (rounded)

sensitivity = 0.81 (rounded)

specificity = 0.72 (rounded)

FPR = 0.27

PPR = 0.66 (rounded)

NPV = 0.86 (rounded)

precision = 0.66 (rounded)

recall = 0.81 (rounded)

Both in Traning and Test dataset Model have predicted almost with same accuracy, sensitivity,specificity