LEAD SCORING

CASE STUDY

Business Understanding and problem statement

An education company named X Education sells online courses to industry professionals. On any given day, many professionals who are interested in the courses land on their website and browse for courses. The company markets its courses on several websites and search engines like Google. Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos.

The company wants to -

- Identify the most potential leads, also known as 'Hot Leads'.
- If they successfully identify this set of leads, the lead conversion rate should go up as the sales team will now be focusing more on communicating with the potential leads rather than making calls to everyone.
- Build a model wherein lead score needed to be assigned to each of the leads.The customers with a higher lead score have a higher conversion chance. The customers with a lower lead score have a lower conversion chance.

Business objective:

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

Steps undertaken:

1.Importing various libraries like pandas, seaborn, etc and loading the dataset thereafter
2.Data cleaning and manipulation (eg. checking null percentage in features , missing value treatment, recognizing irrelevant features)
3. Checking for data imbalance in the dataset with respect to target variable here.
4. Outliers detection and their proper treatment
5. Data visualization
6 Model building
7. Conclusion based on our outputs and insights drawn from the above mentioned steps

Data importing and reading

Read the Data -

In [3]:
```
# Read the data from csv file

lead_scoring_df = pd.read_csv('Leads.csv')
```

In [4]:
```
# Check the head of the dataset

lead_scoring_df.head()
```

Out[4]:

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

```
# Check the value counts for 'Page Views Per Visit'

lead_scoring_df['Page Views Per Visit'].value_counts()
```

```
0.00     2189
2.00     1795
3.00     1196
4.00      896
1.00      651
         ...
3.43        1
2.56        1
6.33        1
1.64        1
2.08        1
Name: Page Views Per Visit, Length: 114, dtype: int64
```

```
# Check the median of column

lead_scoring_df['Page Views Per Visit'].median()
```

```
2.0
```

```
# Impute value

lead_scoring_df['Page Views Per Visit'] = lead_scoring_df['Page Views Per Visit'].replace(np.NaN, lead_scoring_df['Page Views
```


-- Removing redundant and unwanted columns -

1. Columns which have only one value 'NO' in all the rows.
2. Seems these columns will not provide any insights to the lead.
3. Dropping these columns.
    A. Magazine
    B. Receive More Updates About Our Courses
    C. Update me on Supply Chain Content
    D. Get updates on DM Content
    E. I agree to pay the amount through cheque

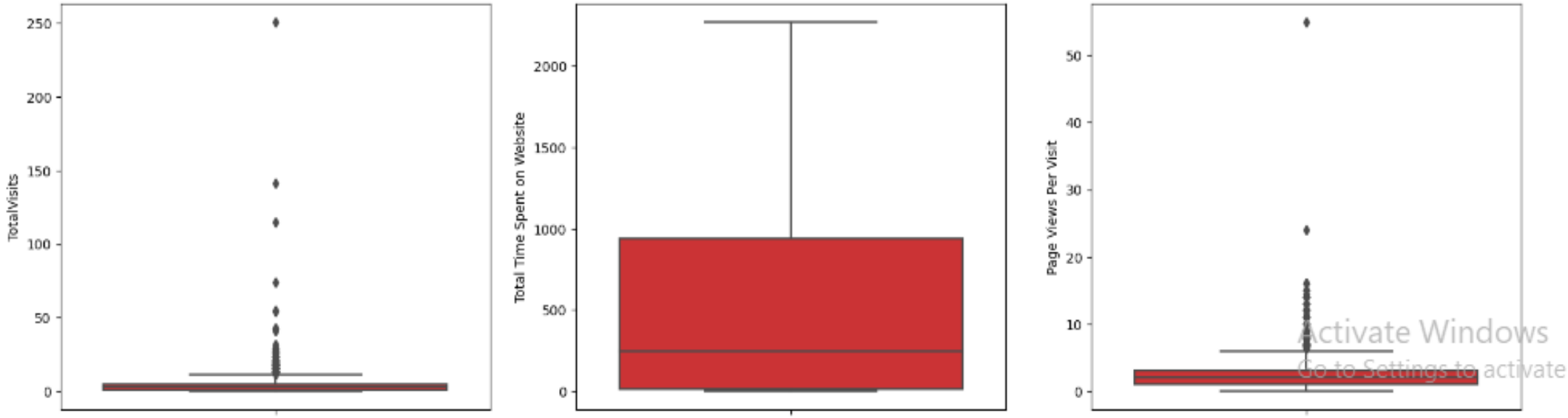
```
lead_scoring_df = lead_scoring_df.drop(['Magazine', 'Receive More Updates About Our Courses',
                                        'Update me on Supply Chain Content', 'Get updates on DM Content',
                                        'I agree to pay the amount through cheque', 'Newspaper Article',
                                        'X Education Forums', 'Newspaper',
                                        'Digital Advertisement',
                                        'Through Recommendations'], axis=1)
```

Outlier treatment

1. Following columns have outliers.
   A. TotalVisits
   B. Page Views Per Visit

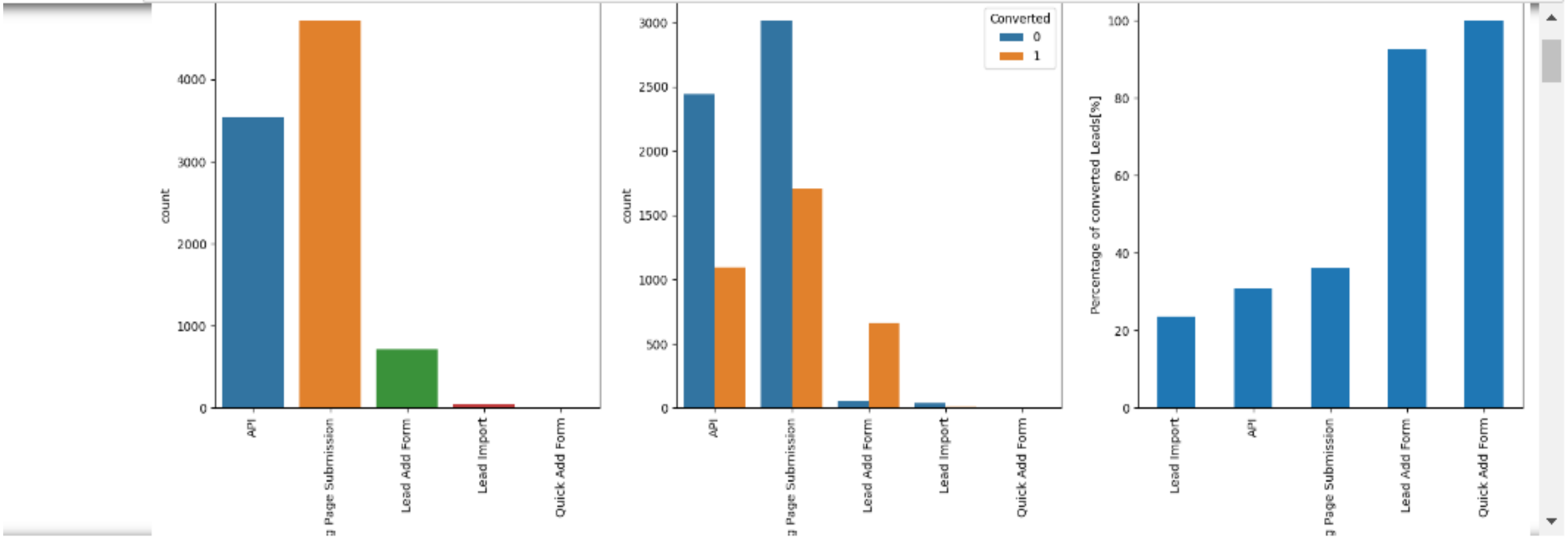
In [57]:

```
# Check outliers in those 2 numerical column

plt.figure(figsize=(20, 25))
plt.subplot(4,3,1)
sns.boxplot(y = 'TotalVisits', palette='Set1', data = lead_scoring_df)
plt.subplot(4,3,2)
sns.boxplot(y = 'Total Time Spent on Website', palette='Set1', data = lead_scoring_df)
plt.subplot(4,3,3)
sns.boxplot(y = 'Page Views Per Visit', palette='Set1', data = lead_scoring_df)
plt.show()
```



Data visualization

In [69]:
```
for i in new_category_column.columns:
    catplot(i)
```



Inferences:

Figure Name : Lead Origin

1. Most Leads originated from landing page submission and mimimum is from Quick Add Form
2. Lead received from Quick Add Form are 100% Converted, but there is only 1 such lead

```
for i in new_category_column.columns:
    catplot(i)
```

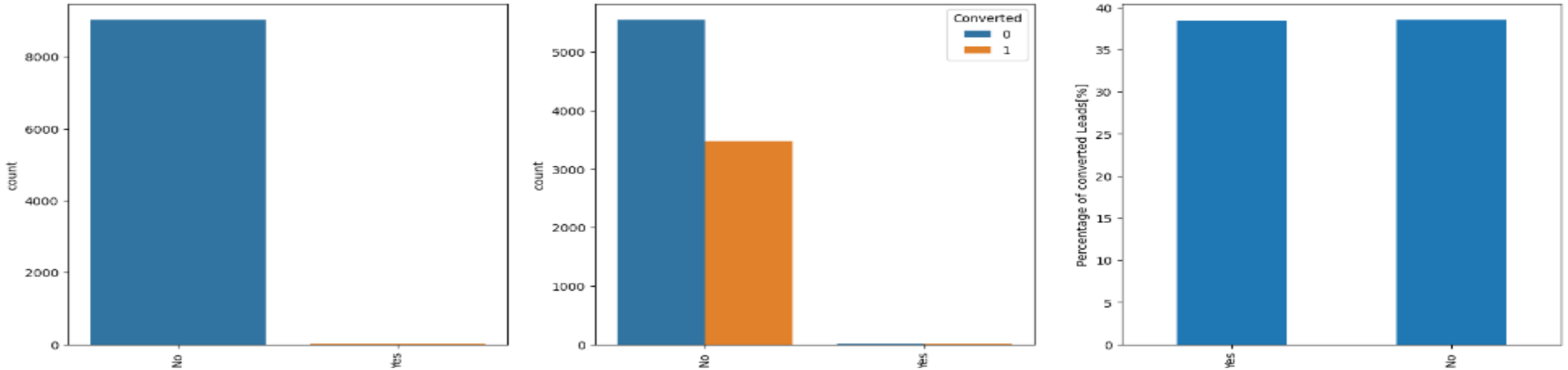


Figure Name: Last Activity

1. Most of the prospects have 'Email Opened' as their last activity
2. Conversion rate of leads with last activity as email opened is about 35% and leads with last activity as SMS Sent is almost 60%
3. Conversion rate of leads with last activity as not sure is 80%

Figure Name:What is your current occupation

1. Unemployed people are highest in number with 45%(approx.) as a conversion rate
2. Housewife are lesser in number but their conversion rate is 100% followed by working professional and others

Figure Name:Search

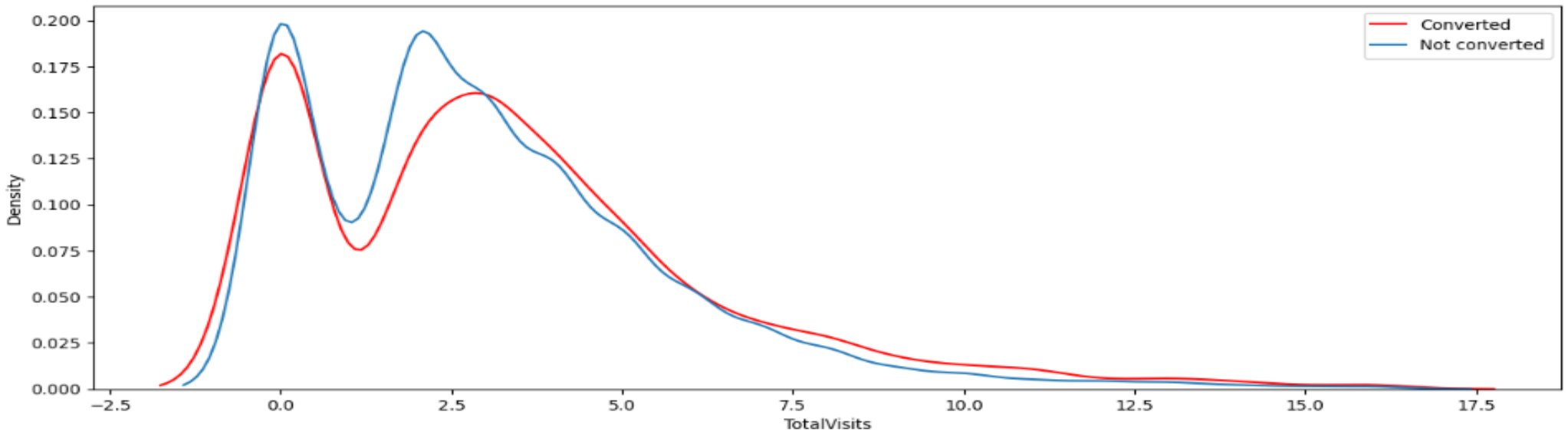
1. Through Search (yes) 14 whereas Search(No) 9226
2. Conversion rate for both through search and not search are almost same

Figure Name:A free copy of Mastering The Interview

1. Conversion rate is high for who do not want a free copy of mastering interview

In [72]:
```
for i in new_numerical_column.columns:
    numplot(i)
```

```
--------------------------------------------------------------------------------------------------------
Figure Name : TotalVisits
--------------------------------------------------------------------------------------------------------
```



Inferences:

Figure Name : TotalVisits ::The total number of visits made by the customer on the website.

1. Large number(2189) of customers have not visited X Education website
2. As we can infer from the figure once the number of visit is more than 2.5 the conversion rate increases (From here conversion rate increase with increase in visits). So if the number of visits is less then they are less likely to convert as compared to customers who visits website

Figure Name : Total Time Spent on Website

1. We can seen that maximum number (2193) of customers has spent close to 0 time on website
2. If customer has spent less time on website then they less likely to convert as compared to customers who spend more time on website

Figure Name : Page Views Per Visit

1. There are about customers who have visited 0.0 pages per visit

Create Dummy Variable -

1. Converting some binary variables (Yes/No) to (1/0).
2. Creating Dummy Variables for categorical features.
3. Drop first dummy variable as p-1 dummies can explain the p category.

```
# Convert some binary variables (Yes/No) to (1/0)

vars = ['Do Not Email', 'Do Not Call', 'Search', 'A free copy of Mastering The Interview']

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

# Call the function to map those columns
lead_scoring_df[vars] = lead_scoring_df[vars].apply(binary_map)
```

Training and Test set -

[86]:
```
# Add feature variables to X

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

Out[86]:

	Do Not Email	Do Not Call	TotalVisits	Total Time Spent on Website	Page Views Per Visit	Search	A free copy of Mastering The Interview	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	...	Last Notable Activity_Email Received	Last Notable Activity_Form Submitted on Website	Last Notable Activity_Had a Phone Conversation
0	0	0	0.0	0	0.0	0	0	0	0	0	...	0	0	0
1	0	0	5.0	674	2.5	0	0	0	0	0	...	0	0	0
2	0	0	2.0	1532	2.0	0	1	1	0	0	...	0	0	0
3	0	0	1.0	305	1.0	0	0	1	0	0	...	0	0	0
4	0	0	2.0	1428	1.0	0	0	1	0	0	...	0	0	0

5 rows × 64 columns

[87]:
```
# Add target variable to y

y = lead_scoring_df['Converted']
y.head()
```

Step:4 -Building a Logistic Regression Model

1. Mixed Approach.
2. Automated coarse tunning with manual fine tunning slection.
3. For Automated coarse tunning we use RFE for feature selection.
4. Then use statmodel for building the model.

RFE - Recursive feature elimination -

1. Start with 20 features.
2. Using the LogisticRegressionfunction from sklearn.

```
logreg = LogisticRegression()

# Start with 20 features

rfe = RFE(logreg, n_features_to_select=20)
rfe = rfe.fit(X_train, y_train)
```

Model 1 -

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

Out[94]:

Generalized Linear Model Regression Results

Dep. Variable:	Converted	**No. Observations:**	6320
Model:	GLM	**Df Residuals:**	6299
Model Family:	Binomial	**Df Model:**	20
Link Function:	Logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-2531.0
Date:	Fri, 14 Jul 2023	**Deviance:**	5062.0
Time:	18:00:44	**Pearson chi2:**	6.53e+03
No. Iterations:	21	**Pseudo R-squ. (CS):**	0.4106
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.3855	0.073	5.269	0.000	0.242	0.529

```
y_train_pred_final = pd.DataFrame({'Converted':y_train.values, 'Converted_Prob':y_train_pred})
y_train_pred_final['LeadId'] = y_train.index
y_train_pred_final.head()
```

	Converted	Converted_Prob	LeadId
0	0	0.115864	5493
1	0	0.115449	8064
2	0	0.014459	4716
3	0	0.350863	9117
4	1	0.453145	2402

ting new column 'predicted' with 1 if Converted_Prob > 0.5 else 0

```
y_train_pred_final['predicted'] = y_train_pred_final.Converted_Prob.map(lambda x: 1 if x > 0.5 else 0)
y_train_pred_final.head()
```

	Converted	Converted_Prob	LeadId	predicted
0	0	0.115864	5493	0
1	0	0.115449	8064	0

Inference - P value of all variables is < 0.05. So all variables seem to be significant.

```
# Get predicted values on the train set

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

```
3]: 5493    0.115864
    8064    0.115449
    4716    0.014459
    9117    0.350863
    2402    0.453145
    1796    0.035380
    1120    0.035380
    253     0.057627
    1491    0.083259
    2004    0.415186
    dtype: float64
```

```
# Reshape

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

```
4]: array([0.11586368, 0.11544875, 0.01445895, 0.35086334, 0.4531454 ,
           0.03537972, 0.03537972, 0.05762721, 0.08325887, 0.41518643])
```

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

Accuracy rate is around 81% which is good. But we will also need to calculate other metrics as we cannot depend only on the accuracy metrics only.

Checking VIFs

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

0]:

	Features	VIF
13	Last Notable Activity_Modified	2.22
8	Last Activity_Olark Chat Conversation	2.07
0	Do Not Email	1.84
6	Last Activity_Email Bounced	1.82
3	Lead Source_Olark Chat	1.73
2	Lead Origin_Lead Add Form	1.62
9	What is your current occupation_No Information	1.58
14	Last Notable Activity_Olark Chat Conversation	1.34

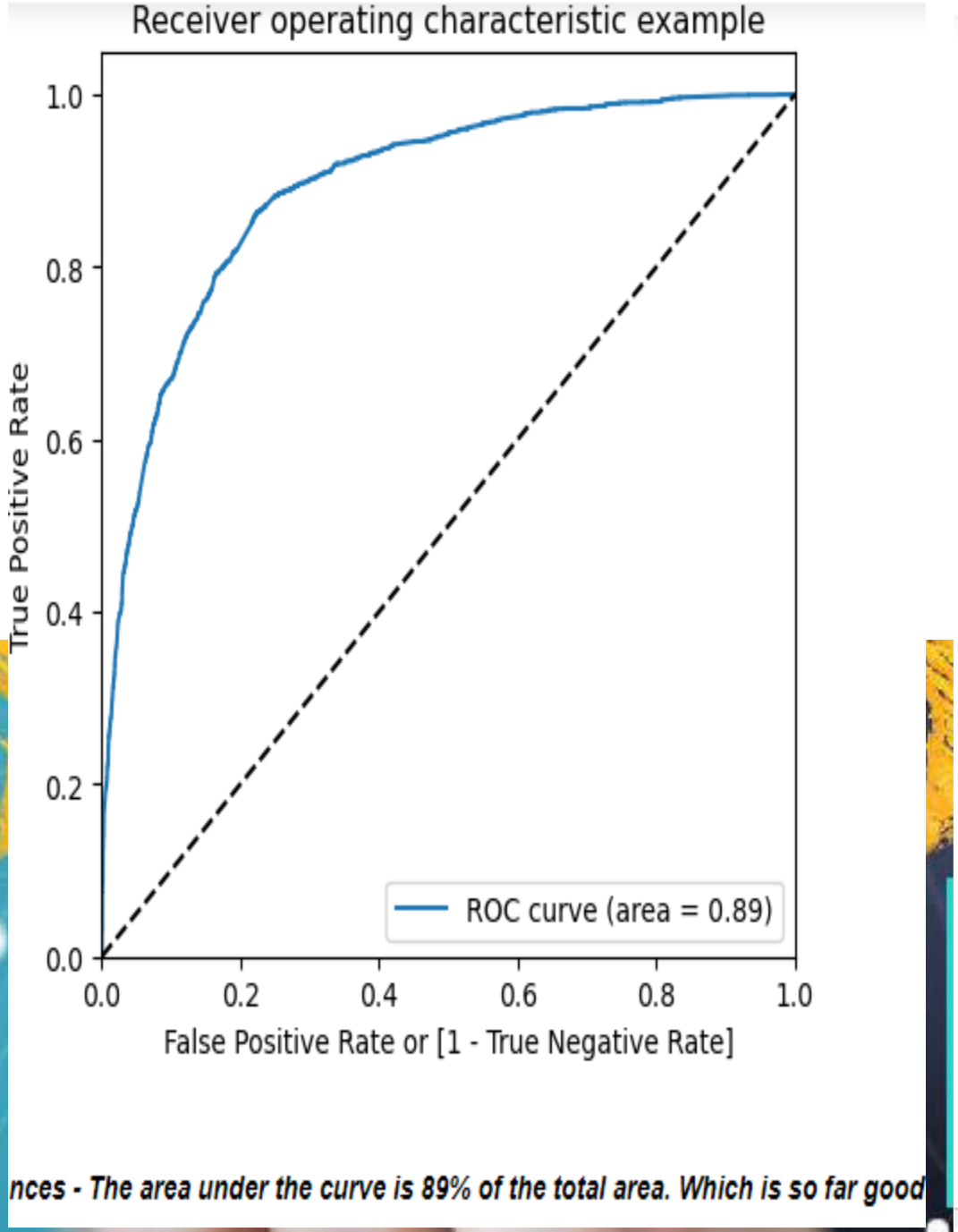

```
TP = confusion1[1,1] # true positive
TN = confusion1[0,0] # true negatives
FP = confusion1[0,1] # false positives
FN = confusion1[1,0] # false negatives
```

```
# Sensitivity/TPR(True Positive Rate)

TP / float(TP+FN)
```

3]: 0.7013687266694317

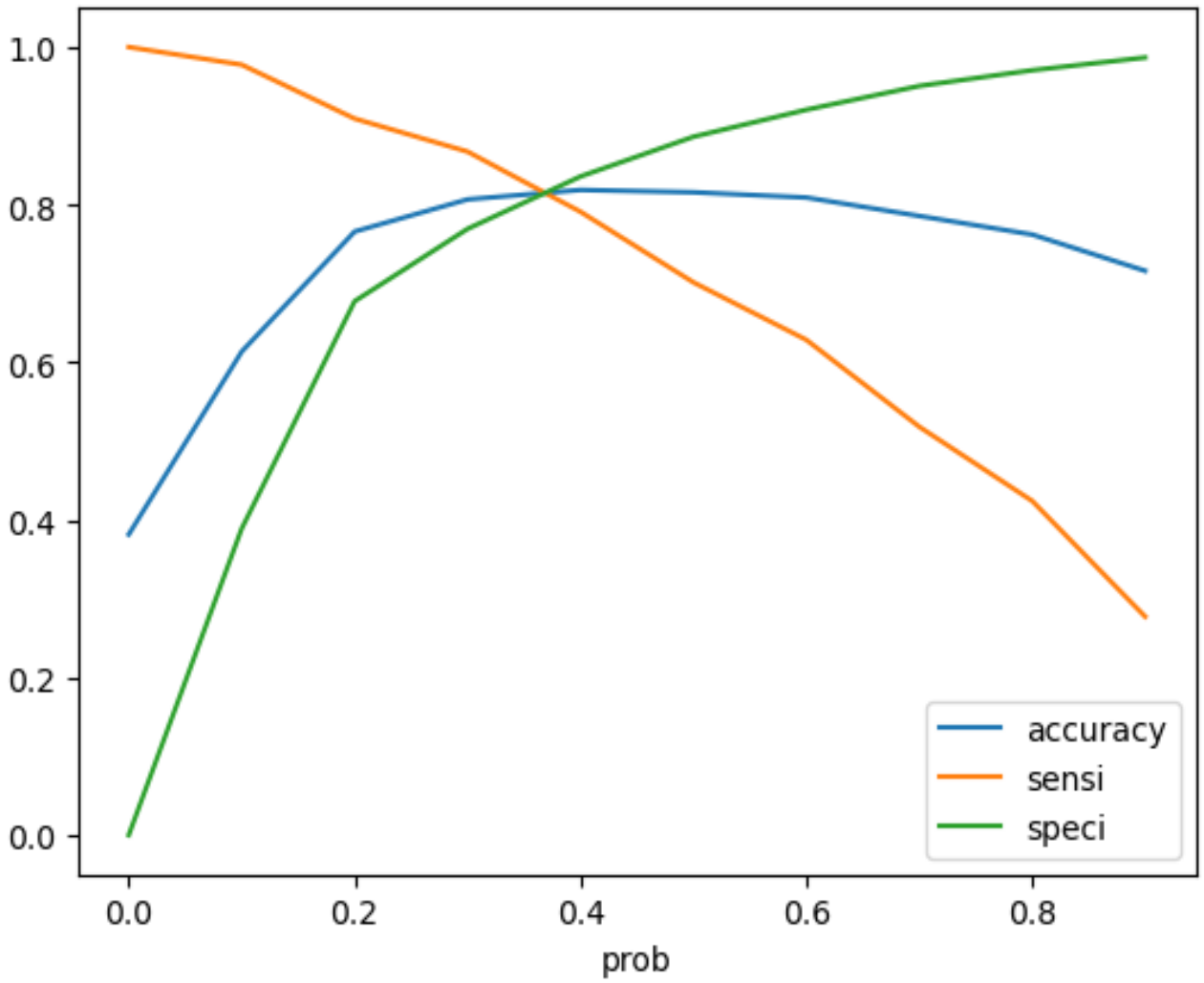
```
# Specificity

TN / float(TN+FP)
```

4]: 0.8861601432591456



nces - The area under the curve is 89% of the total area. Which is so far good

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

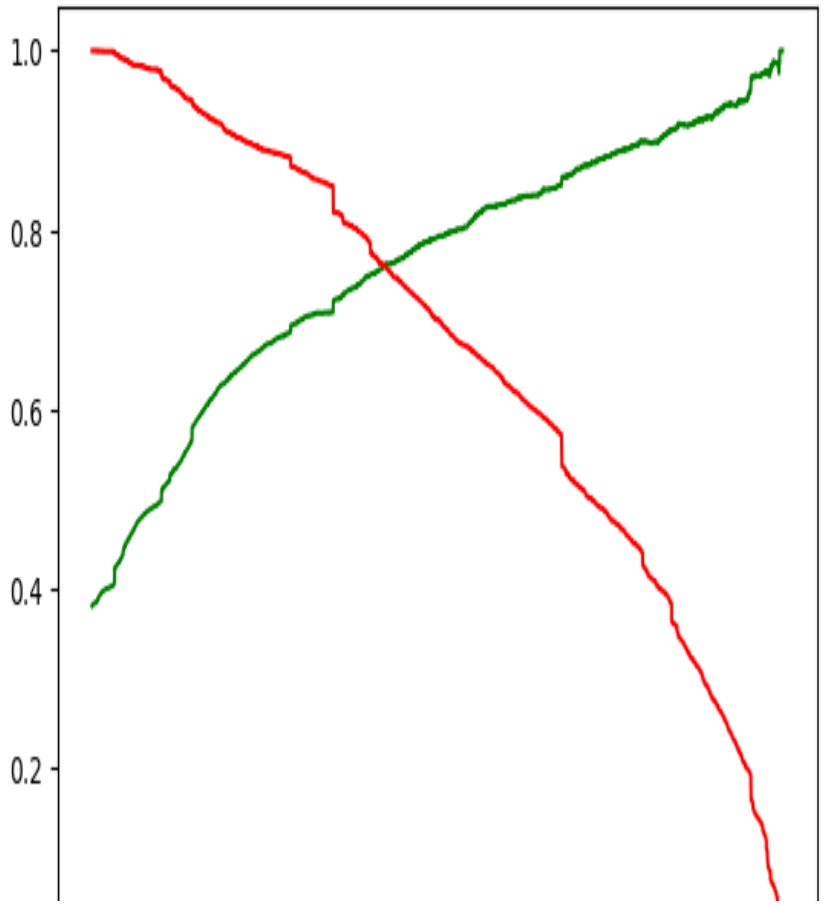


Curve is saying 0.37 is the optimum point to take it as a cutoff probability.

Precision and recall tradeoff

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

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




```
# Checking the accuracy of the test dataset.

from sklearn import metrics    # Imprting metrics from sklearn

print('Accuracy score in predicting test dataset :',metrics.accuracy_score(y_test_pred_final.Conv
```

```
Accuracy score in predicting test dataset : 0.8069398301956442
```

```
from sklearn.metrics import precision_score, recall_score     # Importing precision and recall sc

print('Precision score in predicting test dataset:',precision_score(y_test_pred_final.Converted,

print('Recall score in predicting test dataset:',recall_score(y_test_pred_final.Converted, y_test
```

```
Precision score in predicting test dataset: 0.7142857142857143
Recall score in predicting test dataset: 0.848968105065666
```

```
#Lead Score assigning
# Creating new columns lead score

y_test_pred_final['Lead Score']=y_test_pred_final['Converted_Probability'].apply(lambda x:round(x

y_test_pred_final.head()
```

:

	Converted	Converted_Probability	ID	Predicted	Lead Score
4664	0	0.442622	4664	1	44

```
Last Activity_Converted to Lead                    -0.95
Last Activity_Email Bounced                        -1.15
Last Activity_Not Sure                             -1.58
Last Activity_Olark Chat Conversation              -1.27
What is your current occupation_No Information     -1.14
What is your current occupation_Working Professional  2.34
Last Notable Activity_Email Link Clicked           -1.60
Last Notable Activity_Email Opened                 -1.37
Last Notable Activity_Modified                     -1.65
Last Notable Activity_Olark Chat Conversation      -1.34
Last Notable Activity_Page Visited on Website      -1.67
dtype: float64
```

Conclusion:

- The Accuracy, Precision and Recall score we have obtained is in acceptable range.
- We have high recall than precision which we were aiming to obtain. In business terms, this model has an ability to adjust with the company's requirements in coming future. This concludes that the model is in stable state.

Final Equation Logit Function:-

logit(p) = log(p/(1-p)) = 0.40 + (Do Not Email * -1.31) + (Total Time Spent on Website * 1.11) +(Lead Origin_Lead Add Form * 3.60) + (Lead Source_Olark Chat * 1.33) + (Lead Source_Welingak Website * 2.12) + (Last Activity_Converted to Lead * -0.95) + (Last Activity_Email Bounced * -1.15) + (Last Activity_Not Sure *-1.58)+ (Last Activity_Olark Chat Conversation * -1.27) + (What is your current occupation_No Information * -1.14)+ (What is your current occupation_Working Professional * 2.34) + (Last Notable Activity_Email Link Clicked * -1.60) + (Last Notable Activity_Email Opened * -1.37) + (Last Notable Activity_Modified * -1.65) + (Last Notable Activity_Olark Chat Conversation *-1.34) + (Last Notable Activity_Page Visited on Website * -1.67)

Thankyou