LEAD SCORING CASE STUDY

MODELLING AND VISUALIZATION

By- Siddharth & Tanay

PROBLEM STATEMENT-I

An education company named X Education sells online courses to industry professionals. On any given day, many professionals who are interested in the courses land on their website and browse for courses. The company markets its courses on several websites and search engines like Google. Once these people land on the website, they might browse the courses or fill up a form for the course or watch some videos. When these people fill up a form providing their email address or phone number, they are classified to be a lead. Moreover, the company also gets leads through past referrals. Once these leads are acquired, employees from the sales team start making calls, writing emails, etc. Through this process, some of the leads get converted while most do not. The typical lead conversion rate at X education is around 30%.

Now, although X Education gets a lot of leads, its lead conversion rate is very poor. For example, if, say, they acquire 100 leads in a day, only about 30 of them are converted. To make this process more efficient, the company wishes 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. A typical lead conversion process can be represented using the following funnel:

PROBLEM STATEMENT-II

Lead Conversion Process - Demonstrated as a funnel

As you can see, there are a lot of leads generated in the initial stage (top) but only a few of them come out as paying customers from the bottom. In the middle stage, you need to nurture the potential leads well (i.e. educating the leads about the product, constantly communicating etc.) in order to get a higher lead conversion.

X Education has appointed you to help them select the most promising leads, i.e. the leads that are most likely to convert into paying customers. The company requires you to build a model wherein you need to assign a lead score to each of the leads such that the customers with higher lead score have a higher conversion chance and the customers with lower lead score have a lower conversion chance. The CEO, in particular, has given a ballpark of the target lead conversion rate to be around 80%.

Lead Conversion Process - Demonstrated as a funnel

DATASET PROVIDED

We have been provided with a leads dataset from the past with around 9000 data points. This dataset consists of various attributes such as Lead Source, Total Time Spent on Website, Total Visits, Last Activity, etc. which may or may not be useful in ultimately deciding whether a lead will be converted or not. The target variable, in this case, is the column 'Converted' which tells whether a past lead was converted or not wherein 1 means it was converted and 0 means it wasn't converted. You can learn more about the dataset from the data dictionary provided in the zip folder at the end of the page. Another thing that you also need to check out for are the levels present in the categorical variables. Many of the categorical variables have a level called 'Select' which needs to be handled because it is as good as a null value (think why?)

GOALS OF THE CASE STUDY

There are quite a few goals for this case study.

I. 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. A higher score would mean that the lead is hot, i.e. is most likely to convert whereas a lower score would mean that the lead is cold and will mostly not get converted.

II. There are some more problems presented by the company which your model should be able to adjust to if the company's requirement changes in the future so you will need to handle these as well. These problems are provided in a separate doc file. Please fill it based on the logistic regression model you got in the first step. Also, make sure you include this in your final PPT where you'll make recommendations.

LOGISTIC REGRESSION

- **Logistic Regression** is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.
- Logistic Regression is one of the most popular ways to fit models for categorical data, especially for binary response data in Data Modeling. It is the most important (and probably most used) member of a class of models called generalized linear models. Unlike linear regression, logistic regression can directly predict probabilities (values that are restricted to the (0,1) interval); furthermore, those probabilities are well-calibrated when compared to the probabilities predicted by some other classifiers, such as Naive Bayes. Logistic regression preserves the marginal probabilities of the training data. The coefficients of the model also provide some hint of the relative importance of each input variable

LOGISTIC REGRESSION- ASSUMPTIONS

- Binary logistic regression requires the dependent variable to be binary.
- For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
- Only the meaningful variables should be included.
- The independent variables should be independent of each other. That is, the model should have little or no multi-collinearity.
- The independent variables are linearly related to the log odds.
- Logistic regression requires quite large sample sizes.

PROCEDURE IN THE ASSIGNMENT-I

1.Data Understanding.

1. Looked at the type of columns using .head()
2. Used functions like .describe() and .info() to know about the dataset.
3. This gave us a sense of what type of dataset we were dealing with.

II. Data Cleaning and Univariate analysis of select columns

1. From the information gathered from above steps, figured out the redundant columns.
2. Also figured out the columns which were having significantly high values of either ('Yes' or 'No').
3. Decided if we need to drop null values or impute data into them (on the basis of missing value % of 30).
4. Conversion of categorical variables into dummy variables.

III. Data Modelling

1. Used library sklearn and statsmodel for logistic regression.
2. Created the first iteration of the logistic regression model.
3. Calculated VIF and checked for multi collinearity.
4. Used p-values to drop the unnecessary columns (0.05).
5. Once the model is finalised, used .predict() to calculate the probabilities for the train dataset and the test dataset.
6. Use confusion matrix to calculate accuracy, sensitivity and specificity.
7. Also calculated precision.

PROCEDURE IN THE ASSIGNMENT-II

IV. Conclusion

1. Listed out top prospects (Hot lead) on the basis of probability of converted point.
2. Listed out the percentage of accuracy, sensitivity and specificity.
3. And have also talked about the precision of the dataset after the analysis.
4. Listed out top variables and top dummy variables.

DATA UNDERSTANDING-II

```
# to read data set
df=pd.read_csv('Leads.csv')
df.head()
```

	Prospect ID	Lead Number	Lead Origin	Lead Source	Do Not Email	Do Not Call	Converted	TotalVisits	Total Time Spent on Website	Page Views Per Visit	...	Get updates on DM Content	Lead Profile	City	Asymmetrique Activity Index
0	7927b2df-8bba-4d29-b9a2-b6e0beafe620	660737	API	Olark Chat	No	No	0	0.0	0	0.0	...	No	Select	Select	02.Medium
1	2a272436-5132-4136-86fa-dcc88c88f482	660728	API	Organic Search	No	No	0	5.0	674	2.5	...	No	Select	Select	02.Medium
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	0cc2df48-7cf4-4e39-9de9-19797f9b38cc	660719	Landing Page Submission	Direct Traffic	No	No	0	1.0	305	1.0	...	No	Select	Mumbai	02.Medium
4	3256f628-e534-4826-9d63-4a8b88782852	660681	Landing Page Submission	Google	No	No	1	2.0	1428	1.0	...	No	Select	Mumbai	02.Medium

5 rows × 37 columns

Used df.head() to take a look at the columns we are dealing with.

DATA UNDERSTANDING-II

```
# to get the shape of the data
df.shape
```

```
(9240, 37)
```

To determine the shape of the dataframe.

```
# to get the decription about the data set
df.describe()
```

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

To get the statistic description about the data.

DATA CLEANING AND UNIVARIATE ANALYSIS(SELECT COLUMNS)

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

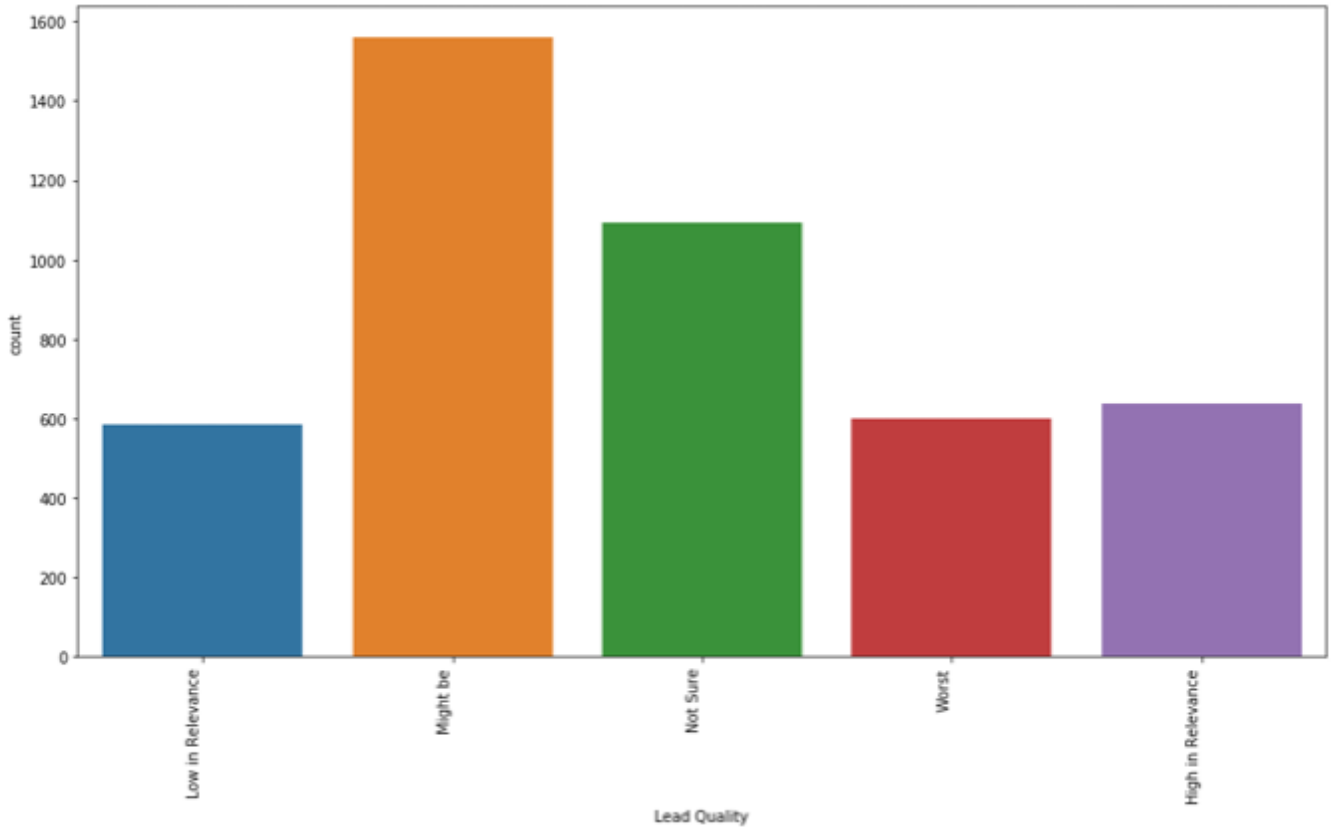
```
Prospect ID                                      0.00
Lead Number                                      0.00
Lead Origin                                      0.00
Lead Source                                      0.39
Do Not Email                                     0.00
Do Not Call                                      0.00
Converted                                        0.00
TotalVisits                                      1.48
Total Time Spent on Website                      0.00
Page Views Per Visit                             1.48
Last Activity                                    1.11
Country                                         26.63
Specialization                                  15.56
How did you hear about X Education              23.89
What is your current occupation                 29.11
What matters most to you in choosing a course   29.32
Search                                           0.00
Magazine                                         0.00
Newspaper Article                                0.00
X Education Forums                               0.00
Newspaper                                        0.00
Digital Advertisement                            0.00
Through Recommendations                          0.00
Receive More Updates About Our Courses           0.00
Tags                                            36.29
Lead Quality                                    51.59
Update me on Supply Chain Content                0.00
Get updates on DM Content                        0.00
Lead Profile                                    29.32
City                                            15.37
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
```

Null Value Check (%)

We can see from this snapshot that some columns like Lead Quality, Asymmetrique Activity Index etc., have significantly higher % of missing values. Hence will drop them before starting with the analysis.

And the ones which have missing values less than 30%, we have tried to impute those missing value.

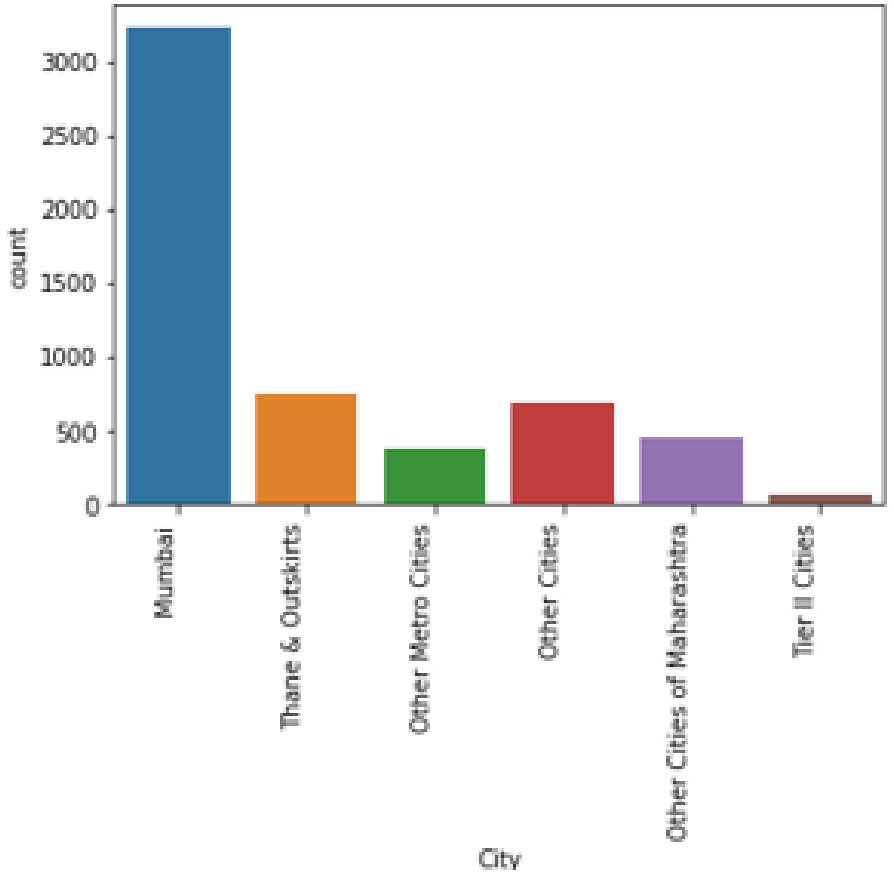
DATA CLEANING AND UNIVARIATE ANALYSIS(SELECT COLUMNS)



no of null value Lead Quality 51.59

Since the % of missng values is 51.59, we have dropped this column.

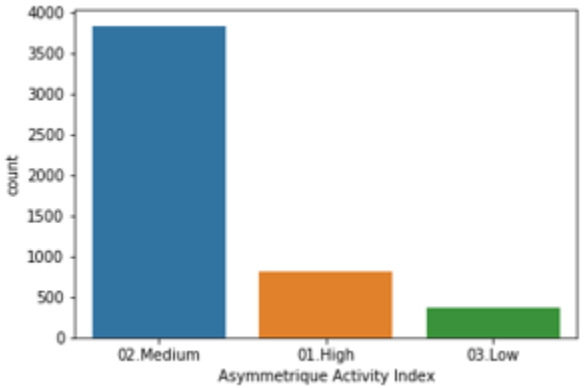
DATA CLEANING AND UNIVARIATE ANALYSIS(SELECT COLUMNS)



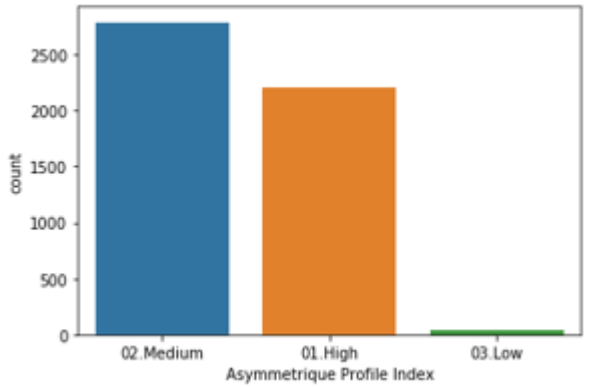

```
null value in City:  39.71
```

Since the % of missing values is 39.71, we have dropped this column.

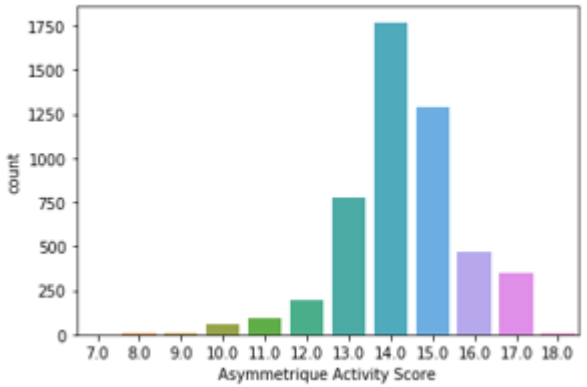
DATA CLEANING AND UNIVARIATE ANALYSIS(SELECT COLUMNS)



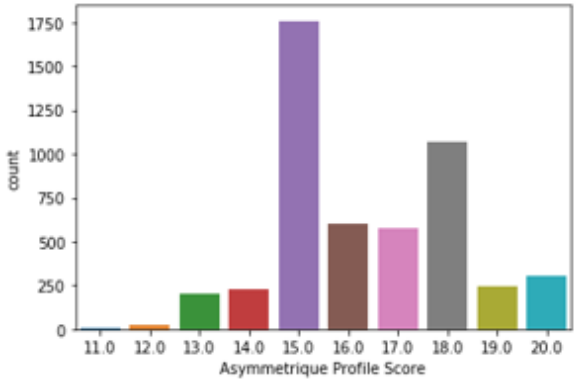
null value in Asymmetrique Activity Index: 45.65



null value in Asymmetrique Profile Index: 45.65



null value in Asymmetrique Activity Score: 45.65



null value in Asymmetrique Profile Score: 45.65

Since all 4 coulmns have significantly high values of missing (45.65). We have dropped them.

LABEL ENCODING

```
df['Do Not Email']=df['Do Not Email'].map({'Yes':1,'No':0})
df['Do Not Call']=df['Do Not Call'].map({'Yes':1,'No':0})
df['Search']=df['Search'].map({'Yes':1,'No':0})
df['Magazine']=df['Magazine'].map({'Yes':1,'No':0})
df['Newspaper Article']=df['Newspaper Article'].map({'Yes':1,'No':0})
df['X Education Forums']=df['X Education Forums'].map({'Yes':1,'No':0})
df['Newspaper']=df['Newspaper'].map({'Yes':1,'No':0})
df['Digital Advertisement']=df['Digital Advertisement'].map({'Yes':1,'No':0})
df['Through Recommendations']=df['Through Recommendations'].map({'Yes':1,'No':0})
df['Receive More Updates About Our Courses']=df['Receive More Updates About Our Courses'].map({'Yes':1,'No':0})
df['Update me on Supply Chain Content']=df['Update me on Supply Chain Content'].map({'Yes':1,'No':0})
df['Get updates on DM Content']=df['Get updates on DM Content'].map({'Yes':1,'No':0})
df['I agree to pay the amount through cheque']=df['I agree to pay the amount through cheque'].map({'Yes':1,'No':0})
df['A free copy of Mastering The Interview']=df['A free copy of Mastering The Interview'].map({'Yes':1,'No':0})
```

DUMMY VARIABLE CREATION

```
# to create dummy variable
#instead of creating dummy singly created dummy for required variable in one go
pd.options.display.max_columns=40
status=pd.get_dummies(df[[ 'Lead Origin', 'Lead Source', 'Last Activity',
       'Specialization', 'What is your current occupation',
       'What matters most to you in choosing a course', 'Tags',
         'Last Notable Activity']],dummy_na=True,drop_first=True)
status.head()
```

	Lead Origin_Landing Page Submission	Lead Origin_Lead Add Form	Lead Origin_Lead Import	Lead Origin_nan	Lead Source_Direct Traffic	Lead Source_Facebook	Lead Source_Google	Lead Source_Live Chat	Lead Source_NC_EDM	Lead Source_Olark Chat	Sourc
0	0	0	0	0	0	0	0	0	0	1	
1	0	0	0	0	0	0	0	0	0	0	
2	1	0	0	0	1	0	0	0	0	0	
3	1	0	0	0	1	0	0	0	0	0	
4	1	0	0	0	0	0	1	0	0	0	

5 rows × 111 columns

We have created dummy variable for categorical columns.

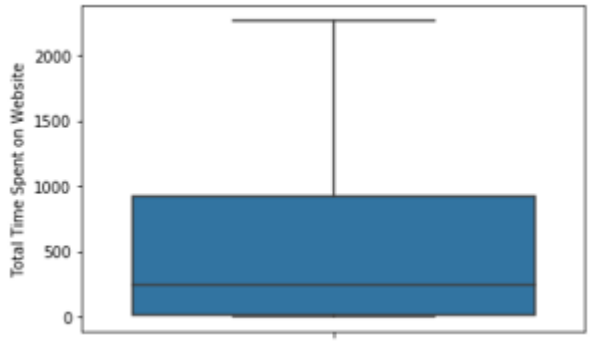
```
# Add the results to the original dataframe
df=pd.concat([df,status],axis=1)
df.head()
```

OUTLIER ANALYSIS (NUMERICAL COLUMNS

```
# Checking outliers at 25%,50%,75%,90%,95% and 99%
df1.describe(percentiles=[.25,.5,.75,.90,.95,.99])
```

	Lead Number	Total Time Spent on Website	TotalVisits	Page Views Per Visit
count	9074.000000	9074.000000	9074.000000	9074.000000
mean	617032.619352	482.887481	3.456028	2.370151
std	23348.029512	545.256560	4.858802	2.160871
min	579533.000000	0.000000	0.000000	0.000000
25%	596406.000000	11.000000	1.000000	1.000000
50%	615278.500000	246.000000	3.000000	2.000000
75%	637176.500000	922.750000	5.000000	3.200000
90%	650276.800000	1373.000000	7.000000	5.000000
95%	655344.450000	1557.000000	10.000000	6.000000
99%	659563.350000	1839.000000	17.000000	9.000000
max	660737.000000	2272.000000	251.000000	55.000000

```
#Columns considered for outlier ananlysis
df1=df[['Lead Number','Total Time Spent on Website','TotalVisits','Page Views Per Visit']]
```



Since the values are increasing significantly at 99%, we have taken it as a cutoff.

```
q1=df['Page Views Per Visit'].quantile(0.99)
q2=df['TotalVisits'].quantile(0.99)
df=df[df['Page Views Per Visit']<=q1]
df=df[df['TotalVisits']<=q2]
df.shape
```

```
(8924, 131)
```

TRAIN-TEST SPLIT

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

	Converted
0	0
1	0
2	1
3	0
4	1

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

```
print(X_train.shape)
print(X_test.shape)
```

```
(6246, 128)
(2678, 128)
```

We have spit the data into train and test. Train size = 70% and test size = 30% at random state =100.

APPLYING LOGISTIC REGRESSION

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

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

```
from sklearn.feature_selection import RFE
rfe = RFE(logreg, 35)
rfe = rfe.fit(X_train, y_train)
```

```
rfe.support_
```

```
rfe.ranking_
```

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

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

	coef	std err	z	P>|z|	[0.025	0.975]
const	-2.0735	0.477	-4.349	0.000	-3.008	-1.139
Do Not Email	-0.7355	0.303	-2.431	0.015	-1.328	-0.143
Total Time Spent on Website	1.0794	0.063	17.252	0.000	0.957	1.202
Lead Origin_Landing Page Submission	-0.7227	0.140	-5.179	0.000	-0.996	-0.449
Lead Origin_Lead Add Form	1.0703	0.431	2.483	0.013	0.226	1.915
Lead Source_Olark Chat	0.8341	0.170	4.900	0.000	0.501	1.168
Lead Source_Welingak Website	3.8421	0.845	4.547	0.000	2.186	5.498
Last Activity_Converted to Lead	-0.9338	0.392	-2.385	0.017	-1.701	-0.166
Last Activity_Email Bounced	-1.0199	0.620	-1.645	0.100	-2.235	0.195
Last Activity_Had a Phone Conversation	3.0203	2.000	1.510	0.131	-0.899	6.939
Last Activity_Olark Chat Conversation	-0.9799	0.317	-3.089	0.002	-1.602	-0.358
Last Activity_Page Visited on Website	-0.6862	0.249	-2.755	0.006	-1.174	-0.198
Last Activity_SMS Sent	0.9772	0.266	3.674	0.000	0.456	1.499
What is your current occupation_Student	-0.9949	0.624	-1.595	0.111	-2.217	0.228
What is your current occupation_Unemployed	-0.8619	0.370	-2.327	0.020	-1.588	-0.136
Tags_Busy	2.6727	0.381	7.006	0.000	1.925	3.420
Tags_Closed by Horizzon	8.0961	0.795	10.184	0.000	6.538	9.654
Tags_Diploma holder (Not Eligible)	-0.7514	1.118	-0.672	0.501	-2.942	1.439
Tags_Interested in other courses	-0.3683	0.501	-0.735	0.462	-1.350	0.613
Tags_Lateral student	26.5639	8.72e+04	0.000	1.000	-1.71e+05	1.71e+05
Tags_Lost to EINS	7.3426	0.615	11.934	0.000	6.137	8.548

We can clearly see that there are some columns with high probability values. Hence we will drop them one at a time.

FEATURE SELECTION-VIF

```
# 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
```

	Features	VIF
9	Last Activity_SMS Sent	7.13
22	Last Notable Activity_SMS Sent	7.06
10	What is your current occupation_Unemployed	6.36
2	Lead Origin_Landing Page Submission	3.07
21	Last Notable Activity_Modified	2.95
19	Tags_nan	2.91
16	Tags_Will revert after reading the email	2.30
4	Lead Source_Olark Chat	2.25
3	Lead Origin_Lead Add Form	1.95
7	Last Activity_Olark Chat Conversation	1.74
14	Tags_Ringing	1.74
1	Total Time Spent on Website	1.46
6	Last Activity_Converted to Lead	1.35
5	Lead Source_Welingak Website	1.32
12	Tags_Closed by Horizzon	1.31
0	Do Not Email	1.19
8	Last Activity_Page Visited on Website	1.17
18	Tags_switched off	1.14
11	Tags_Busy	1.14
13	Tags_Lost to EINS	1.13
20	Last Notable Activity_Email Link Clicked	1.06
17	Tags_in touch with EINS	1.01
15	Tags_Shall take in the next coming month	1.00

We dropped the features that were having VIF more than 5 at a time after each iteration. Last Activity_SMS Sent and What is your current occupation_Unemployed were dropped.

FEATURE SELECTION-VIF

These were the final feature we selected after VIF

	Features	VIF
2	Lead Origin_Landing Page Submission	2.59
19	Last Notable Activity_Modified	2.24
14	Tags_Will revert after reading the email	2.14
4	Lead Source_Olark Chat	2.11
17	Tags_nan	2.06
3	Lead Origin_Lead Add Form	1.93
20	Last Notable Activity_SMS Sent	1.74
7	Last Activity_Olark Chat Conversation	1.60
1	Total Time Spent on Website	1.46
12	Tags_Ringing	1.42
5	Lead Source_Welingak Website	1.32
10	Tags_Closed by Horizzon	1.28
6	Last Activity_Converted to Lead	1.25
0	Do Not Email	1.17
8	Last Activity_Page Visited on Website	1.12
11	Tags_Lost to EINS	1.11
9	Tags_Busy	1.09
16	Tags_switched off	1.09
18	Last Notable Activity_Email Link Clicked	1.05
13	Tags_Shall take in the next coming month	1.00
15	Tags_in touch with EINS	1.00

We dropped the features that were having VIF more than 5 at a time after each iteration. Last Activity_SMS Sent and What is your current occupation_Unemployed were dropped.

LOGISTIC REGRESSION

```
y_train_pred_final = pd.DataFrame({'Converted':y_train.values.reshape(-1), 'Converted_points':y_train_pred})
#y_train_pred_final['CustID'] = y_train.index
y_train_pred_final.head()
```

	Converted	Converted_points
0	1	0.985318
1	1	0.996063
2	1	0.981540
3	0	0.032752
4	0	0.224880

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

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

	Converted	Converted_points	predicted
0	1	0.985318	1
1	1	0.996063	1
2	1	0.981540	1
3	0	0.032752	0
4	0	0.224880	0

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

```
array([[3707,  164],
       [ 282, 2093]], dtype=int64)
```

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

```
0.9285943003522255
```

Accuracy of the model is 0.9285

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

LOGISTIC REGRESSION

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

```
0.8812631578947369
```

```
# Let us calculate specificity
TN / float(TN+FP)
```

```
0.9576336863859468
```

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

```
0.042366313614053214
```

```
# positive predictive value
print (TP / float(TP+FP))
```

```
0.9273371732388126
```

```
# Negative predictive value
print (TN / float(TN+ FN))
```

```
0.9293055903735272
```

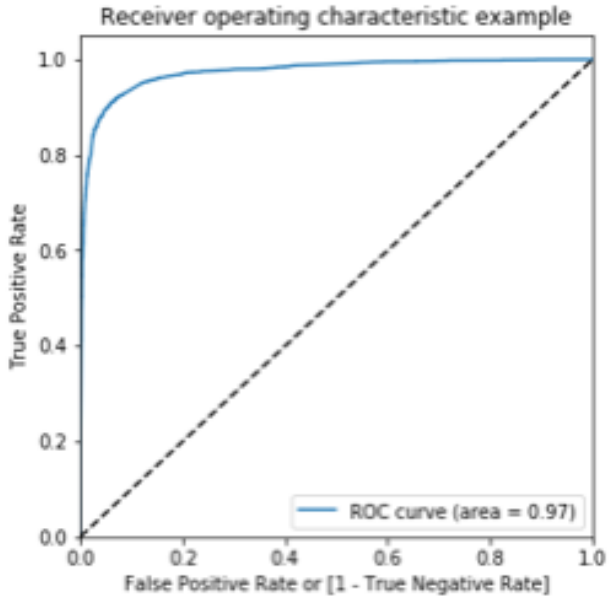
Sensitivity=0.881

Specificity =0.957

ROC

ROC Curve

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



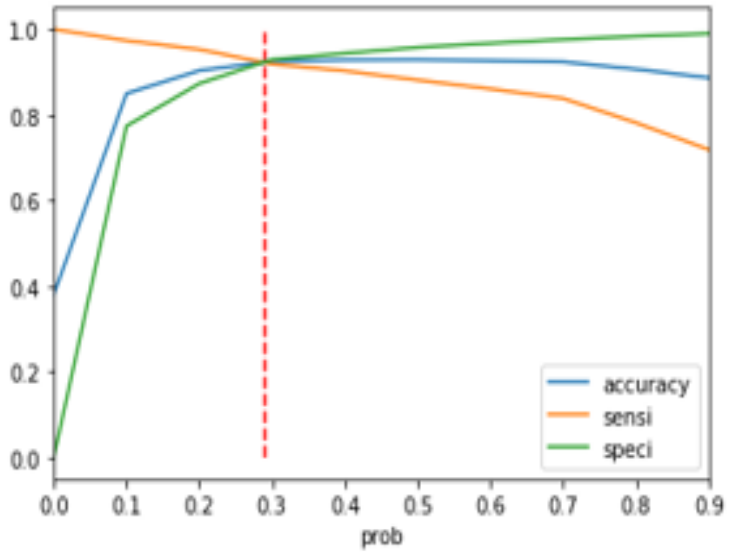

```
# Let's create columns with different probability cutoffs
numbers = [float(x)/10 for x in range(10)]
for i in numbers:
    y_train_pred_final[i]= y_train_pred_final.Converted_points.map(lambda x: 1 if x > i else 0)
y_train_pred_final.head()
```

	Converted	Converted_points	predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	1	0.985318	1	1	1	1	1	1	1	1	1	1	1
1	1	0.996063	1	1	1	1	1	1	1	1	1	1	1
2	1	0.981540	1	1	1	1	1	1	1	1	1	1	1
3	0	0.032752	0	1	0	0	0	0	0	0	0	0	0
4	0	0.224880	0	1	1	1	0	0	0	0	0	0	0

We have created columns with different probability cutoff.

SPECIFICITY VS SENSITIVITY

```
# Let's plot accuracy sensitivity and specificity for various probabilities.
cutoff_df.plot.line(x='prob', y=['accuracy','sensi','speci'])
plt.vlines(x=0.29, ymax=1, ymin=0, colors="r", linestyles="--")
plt.show()
```



All three intersect at x = 0.3. So now we will take 0.3 as the new cutoff.

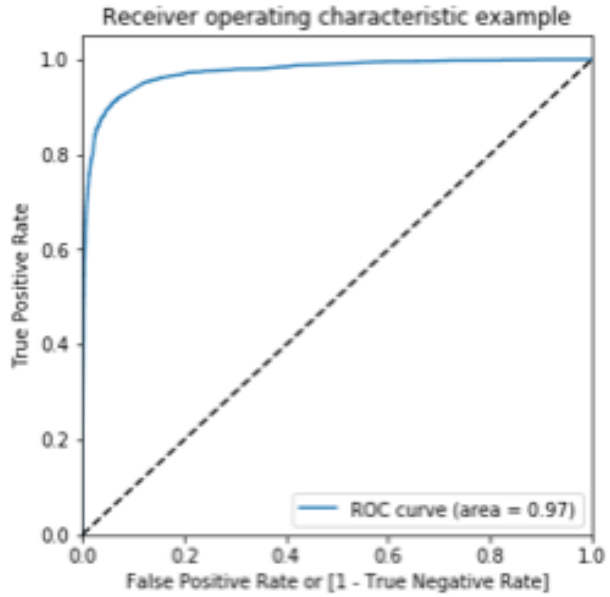
```
y_train_pred_final['final_predicted'] = y_train_pred_final.Converted_points.map( lambda x: 1 if x > 0.3 else 0)

y_train_pred_final.head()
```

	Converted	Converted_points	predicted	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	final_predicted
0	1	0.985318	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0.996063	1	1	1	1	1	1	1	1	1	1	1	1
2	1	0.981540	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0.032752	0	1	0	0	0	0	0	0	0	0	0	0
4	0	0.224880	0	1	1	1	0	0	0	0	0	0	0	0

NEW ROC

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



Accuracy=0.9252

Sensitivity=0.9187

Specificity=0.9292

Precesion=0.8884

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

```
array([[3597,  274],
       [ 193, 2182]], dtype=int64)
```

MAKING PREDICTION ON TEST DATA

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

```
0.9253174010455564
```

```
confusion3=metrics.confusion_matrix(y_test.Converted, y_test.final_converted)
confusion3
```

```
array([[1563,  121],
       [  79,  915]], dtype=int64)
```

```
TP = confusion3[1,1] # true positive
TN = confusion3[0,0] # true negatives
FP = confusion3[0,1] # false positives
FN = confusion3[1,0] # false negatives
```

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

```
0.920523138832998
```

```
# Let us calculate specificity
TN / float(TN+FP)
```

```
0.9281472684085511
```

```
#precesion of the model
confusion3[1,1]/(confusion3[0,1]+confusion3[1,1])
```

```
0.8832046332046332
```

We can clearly see that the overall accuracy is 92.5% and precision is 88%. We can conclude by saying that we have built a fairly good model.

CONCLUSION

- The top three variable which contribute most toward the probability of a lead getting converted are as :-
  1. Total time spent on website
  2. Do not email
  3. Tags
- The top 3 categorical/dummy variable in the model one should focus the most in order to increase the probability of lead conversion:
  1. Tags_Closed by Horizzon
  2. Tags_Lost to EINS
  3. Tags_Will revert after reading the email