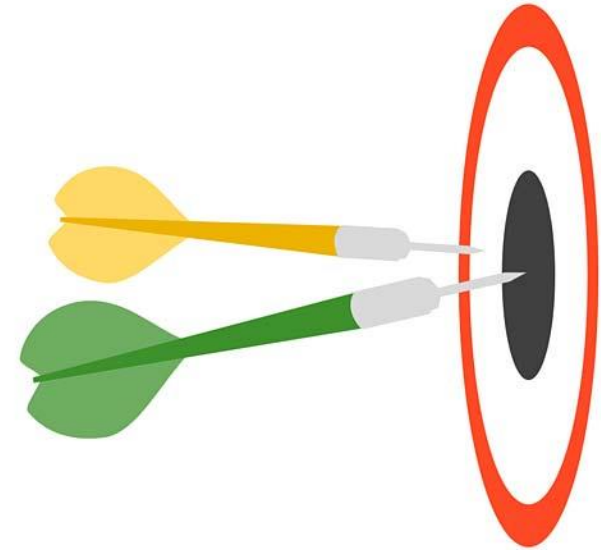


Lead Scoring Case Study



Business Objectives

- ❑ 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.



Goals and Objectives

Data

- ☐ You 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.
- ☐ 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.

Main Column

- ❑ 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.



Lead Conversion Process - Demonstrated as a funnel

Problem Solving Methodology

- ☐ Reading and Understanding the Data
- ☐ Data Cleaning and Preparation
- ☐ EDA(Exploratory Data Analytics)
- ☐ Data Preparation for Modelling
- ☐ Logistics Regression Model Building with help of RFE
- ☐ Model Prediction and Evaluation
- ☐ Calculating Lead score for the entire dataset



Data understanding and Preparation

Remove columns which has more NULL values

- Deleting the following columns as they have more percentage of NULL values and hence cannot be responsible in predicting a successful lead case :
 - How did you hear about X Education - 78.46 % NULL value
 - Lead Profile - 74.18 % NULL value

Outliers Treatment and Combining some parameter in columns as Others..

- Below are columns are treated for outliers where we have capped the value
 - TotalVisits, Page Views Per Visit
 - Tags, Last Activity and Lead Source columns parameter which are not giving any inferences will clubbed as others

Data understanding and Preparation

Remove columns which has only one unique value

- Deleting the following columns as they have only one unique value and hence cannot be responsible in predicting a successful lead case :
 - Magazine
 - Receive More updates about the course
 - Update me on Supply chain content
 - Get updates on DM content
 - I agree to pay the amount through cheque

Imputing NULL values with Mode

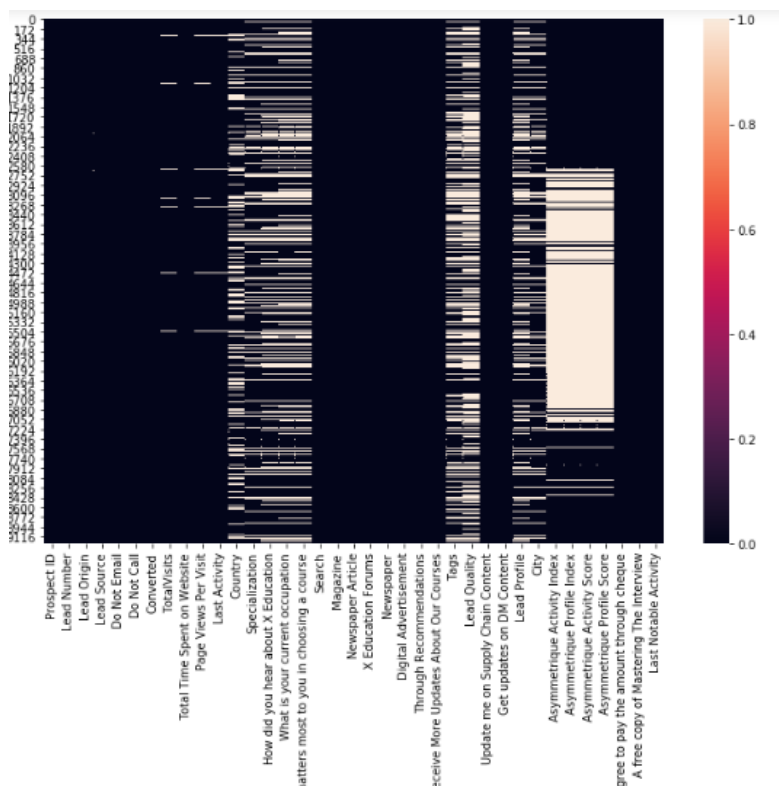
- Below are columns are imputed with Mode:
 - City
 - Tags
 - Country
 - What matters most to you in choosing a course

Data Handling 'Select' values in some columns

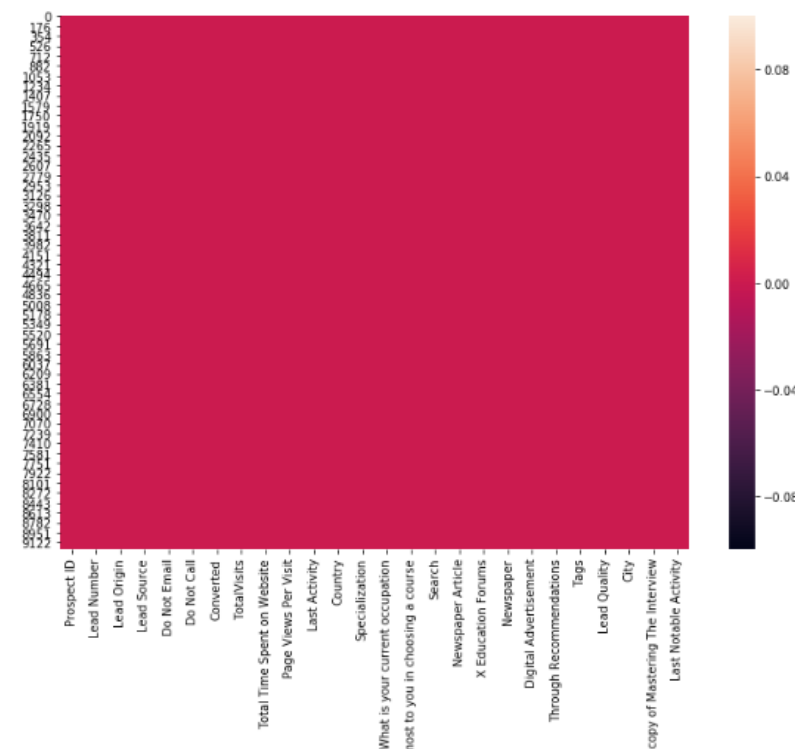
- There are some columns in dataset which have a level/value called 'Select'.
 - This might have happened because these fields in the website might be non-mandatory fields with drop downs options for the customer to choose from.
- Amongst the dropdown values, the default option is probably 'Select' and since these aren't mandatory fields, many customer might have chosen to leave it as the default value 'Select'.
 - The Select values in columns were **converted to NULL**.

NULL Value Heatmap before and after the treatment

Before NULL Treatment



After NULL Treatment



Data Preparation for Modelling

Converting some binary variables (Yes/No) to 0/1

Converting the following binary variables (Yes/No) to 0/1:

- ☐ 'Do Not Email'
- ☐ 'Do Not Call',

Converting some binary variables (Yes/No) to 0/1

```
# Converting Yes to 1 and No to 0
lead['Do Not Email'] = lead['Do Not Email'].map({'Yes': 1, 'No': 0})
lead['Do Not Call'] = lead['Do Not Call'].map({'Yes': 1, 'No': 0})
```

For categorical variables with multiple levels, creating dummy Creation

For the following categorical variables with multiple levels, dummy features were created:

'Lead Quality', 'Tags', 'Lead Profile', 'Lead Origin', 'What is your current occupation', 'Specialization', 'City', 'Last Activity', 'Country' and 'Lead Source', 'Last Notable Activity'

For categorical variables with multiple levels, creating dummy Creation

```
### Dummy Variable Creation
dummy_var = pd.get_dummies(lead[['Lead Origin', 'Lead Source', 'Last Activity', 'Specialization', 'What i
                                'Tags', 'Lead Quality', 'City', 'Last Notable Activity']], drop_first=True)
dummy_var.head()
```

Data Preparation for Modelling

Test- Train Split

- ❑ The original data frame is split into train and test dataset.
- ❑ The train dataset was used to train the model.
- ❑ The test dataset was used to evaluate the model.

Feature Scaling

- ❑ Scaling helps in interpretation. It is important to have all variables (specially categorical ones which have values 0 and on the same scale for the model to be easily interpretable).
- ❑ '**Standardisation**' was used to scale the data for modelling. It basically brings all of the data into a standard normal distribution with mean at zero and standard deviation one.

Model Building – Logistics Regression

Logistics Regression

- ❑ **Logistic regression** is a statistical model that in its basic form uses a **logistic** function to model a binary dependent variable, although many more complex extensions exist. In **regression** analysis, **logistic regression** (or **logit regression**) is estimating the parameters of a **logistic** model (a form of binary **regression**).

Feature Selection using RFE

- ❑ We built your first model based on the summary statistics, we inferred that many of the variables might be insignificant and hence, we need to do some feature elimination. Since the number of features is huge, let's first start off with an automated feature selection technique (RFE) and then move to manual feature elimination (using p-values and VIFs) - this is exactly the same process that you did in linear regression.

Final Model Logistics Regression

Model Building - 4

```
X_train_sm = sm.add_constant(X_train[col])
LRM_4 = sm.GLM(y_train,X_train_sm,family = sm.families.Binomial())
res = LRM_4.fit()
res.summary()
```

```
# Let's check the overall accuracy of this model on train set
print(metrics.accuracy_score(y_train_pred_final.Converted, y_train_pred_final.predicted))

0.9193827743662415
```

Generalized Linear Model Regression Results

Dep. Variable:	Converted	No. Observations:	6351
Model:	GLM	Df Residuals:	6337
Model Family:	Binomial	Df Model:	13
Link Function:	logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-1588.8
Date:	Sun, 19 Apr 2020	Deviance:	3177.6
Time:	23:29:06	Pearson chi2:	3.08e+04
No. Iterations:	8		
Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
const	-2.0888	0.216	-9.654	0.000	-2.513	-1.665
Do Not Email	-1.3012	0.212	-6.134	0.000	-1.717	-0.885
Lead Origin_Lead Add Form	1.0894	0.363	3.001	0.003	0.378	1.801
Lead Source_Welingak Website	3.4138	0.818	4.173	0.000	1.810	5.017
What is your current occupation_Working Professional	1.3403	0.291	4.602	0.000	0.769	1.911
Tags_Busy	3.8040	0.330	11.532	0.000	3.157	4.450
Tags_Closed by Horizon	7.9562	0.763	10.433	0.000	6.461	9.451
Tags_Lost to EINS	9.1785	0.754	12.177	0.000	7.701	10.656
Tags_Ringing	-1.6947	0.337	-5.036	0.000	-2.354	-1.035
Tags_Will revert after reading the email	3.9665	0.229	17.311	0.000	3.517	4.416
Tags_switched off	-2.2882	0.587	-3.900	0.000	-3.438	-1.138
Lead Quality_Not Sure	-3.3406	0.128	-26.026	0.000	-3.592	-3.089
Lead Quality_Worst	-3.7624	0.850	-4.426	0.000	-5.428	-2.096
Last Notable Activity_SMS Sent	2.7406	0.120	22.847	0.000	2.506	2.976

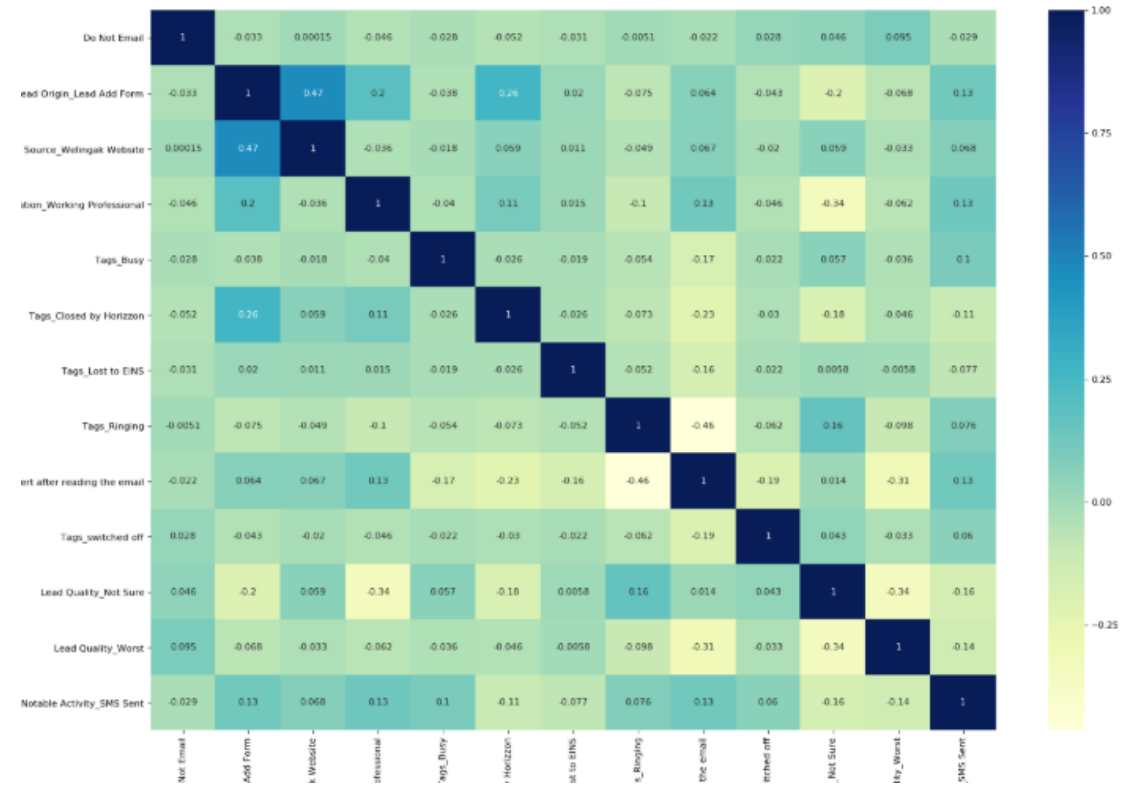
VIFs

Let's see about eliminating the insignificant variables based on the VIFs, and the p-values.

```
# defined the vif check funtion to check the VIF for the model
def vif_check(Y):
    vif = pd.DataFrame()
    vif['Features'] = X_train[Y].columns
    vif['VIF'] = [variance_inflation_factor
                  (X_train[Y].values, i) for i in range(X_train[Y].shape[1])]
    vif['VIF'] = round(vif['VIF'], 2)
    vif = vif.sort_values(by = "VIF", ascending = False)
    return(vif)
```

```
# Checking the Variance Inflation Factor(VIF)
vif_check(col)
```

	Features	VIF
8	Tags_Will revert after reading the email	2.81
10	Lead Quality_Not Sure	2.78
1	Lead Origin_Lead Add Form	1.82
7	Tags_Ringing	1.54
12	Last Notable Activity_SMS Sent	1.52
2	Lead Source_Welingak Website	1.38
3	What is your current occupation_Working Profes...	1.28
5	Tags_Closed by Horizon	1.15
4	Tags_Busy	1.11
0	Do Not Email	1.10
9	Tags_switched off	1.10
6	Tags_Lost to EINS	1.05
11	Lead Quality_Worst	1.03



- ❑ Generalized Linear Models from StatsModels is used to build the Logistic Regression model. The model is built initially with the 15 variables selected by RFE.
- ❑ Unwanted features are dropped serially after checking p values (< 0.5) and VIF (< 5) and model is built multiple times.
- ❑ The final model with 13 features.

Predicting the Converted Probability and Predicted Column

	Converted	Converted_prob	Lead ID
0	0	0.188037	3009
1	0	0.194070	1012
2	0	0.000805	9228
3	1	0.782077	4750
4	1	0.977003	7987

- ❑ Creating a data frame with the actual Converted flag and the predicted probabilities.
- ❑ Showing top 5 records of the data frame in the picture above.

	Converted	Converted_prob	Lead ID	predicted
0	0	0.188037	3009	0
1	0	0.194070	1012	0
2	0	0.000805	9228	0
3	1	0.782077	4750	1
4	1	0.977003	7987	1

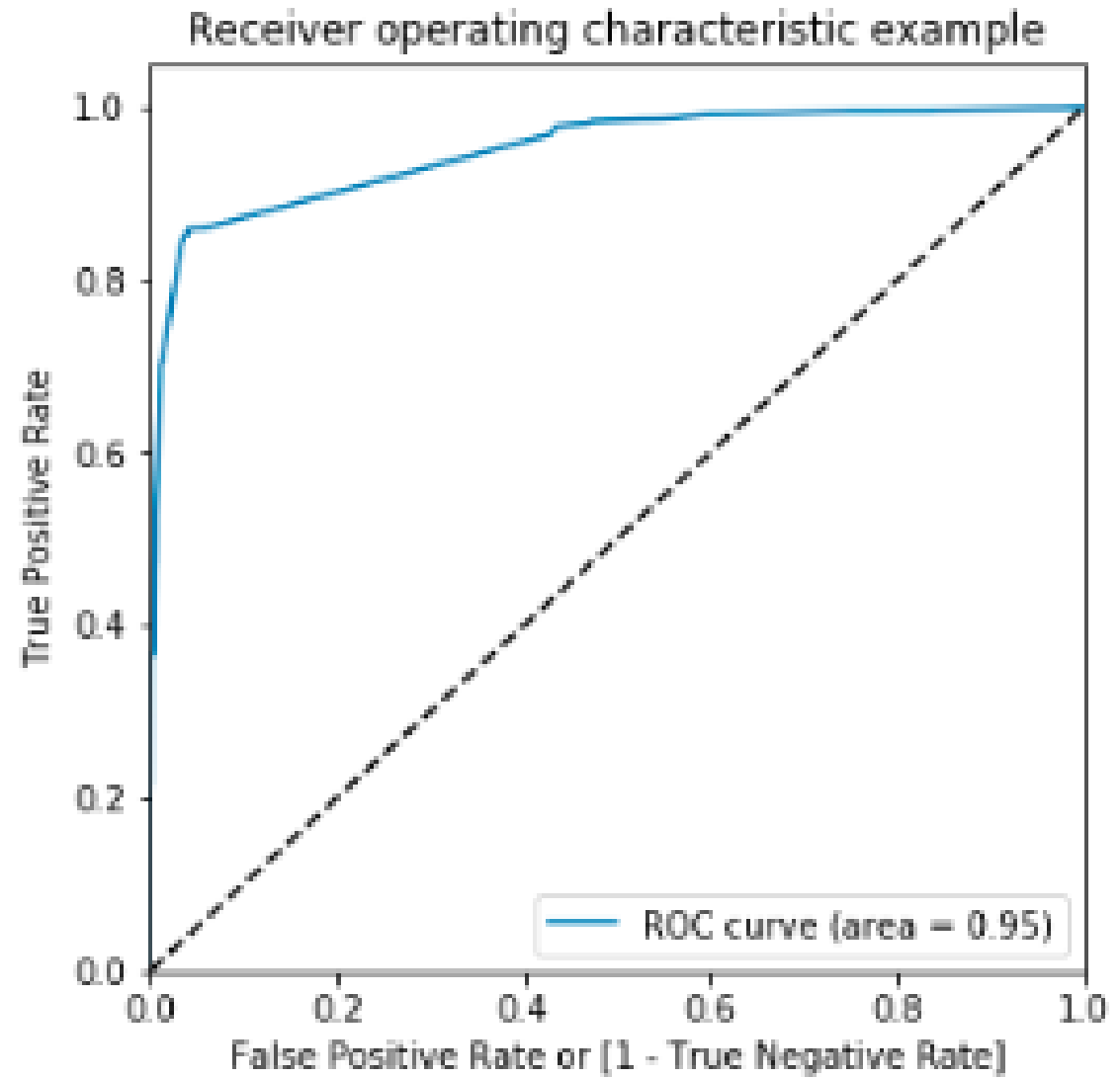
- ❑ Creating new column 'predicted' with 1 if $\text{Converted_Prob} > 0.5$ else 0
- ❑ Showing top 5 records of the data frame in the picture above.

Plotting ROC Curve

- ❑ An ROC curve demonstrates several things:
 - ❑ It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).
 - ❑ The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.
 - ❑ The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

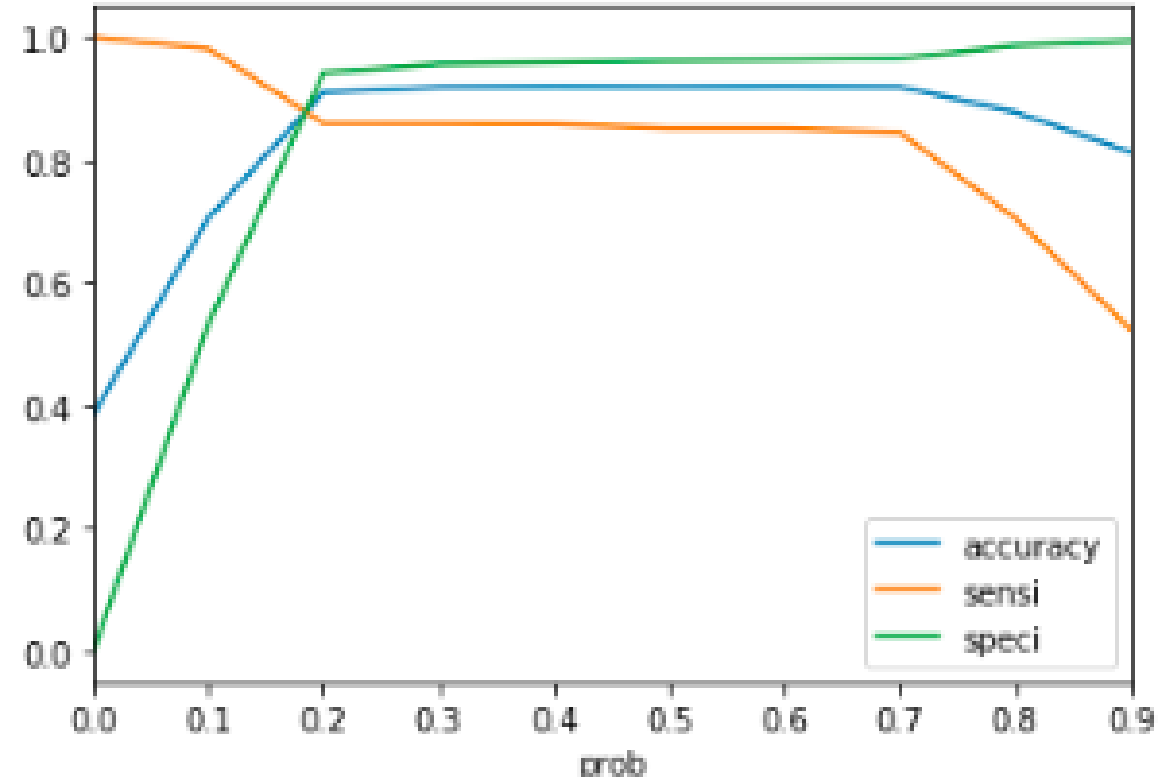
As a rule of thumb, an AUC can be classed as follows,

- 0.90 - 1.00 = excellent
- 0.80 - 0.90 = good
- 0.70 - 0.80 = fair
- 0.60 - 0.70 = poor
- 0.50 - 0.60 = fail



Finding Optimal Cutoff Point

- ❑ The accuracy sensitivity and specificity was calculated for various values of probability threshold and plotted in the graph to the right.
- ❑ From the curve above, **0.2** is found to be the optimum point for cutoff probability.
- ❑ At this threshold value, all the 3 metrics i.e. accuracy, sensitivity and specificity was found to be well above 80% which is a well acceptable value the significance test and the multi-collinearity test.



Evaluating the model on Train Dataset

Confusion Matrix

# Predicted	Not Converted	Converted
# Actual		
Not Converted	3679	226
Converted	343	2103



Probability
Threshold
= 0.2

Accuracy
 $TP + TN / (TP + TN + FN + FP)$

0.910

Sensitivity
 $TP / (TP + FN)$

0.859

Specificity
 $TN / (TN + FP)$

0.942

False Positive
Rate
 $FP / (TN + FP)$

0.057

Positive
Predictive Value
 $TP / (TP + FP)$

0.902

Negative
Predictive Value
 $TN / (TN + FN)$

0.914

Precision
 $TP / TP + FP$

0.902

Recall
 $TP / TP + FN$

0.859

F1 score =
 $2 \times (Precision * Recall) / (Precision + Recall)$

0.880

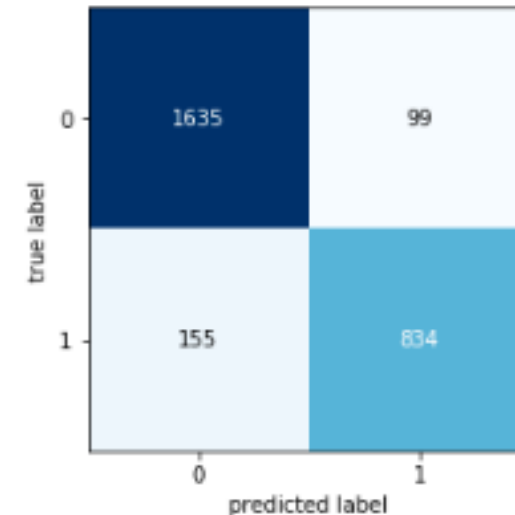
Area under
the curve

0.949

Making predictions on the test dataset

- ❑ The final model on the train dataset is used to make predictions for the test dataset
- ❑ The train data set was scaled using the `scaler.transform` function that was used to scale the train dataset.
- ❑ The Predicted probabilities were added to the leads in the test dataframe.
- ❑ Using the probability threshold value of 0.2, the leads from the test dataset were predicted if they will convert or not.
- ❑ The Conversion Matrix was calculated based on the Actual and Predicted 'Converted' columns

Accuracy Score: 0.906720528828498



	Lead ID	Converted	Converted_prob	final_predicted
0	3271	0	0.188037	0
1	1490	1	0.961508	1
2	7938	0	0.188037	0
3	4216	1	0.999049	1
4	3830	0	0.188037	0

Evaluating the model on Test Dataset

Accuracy
 $\frac{TP + TN}{TP + TN + FN + FP}$

0.906

Sensitivity
 $\frac{TP}{TP + FN}$

0.843

Specificity
 $\frac{TN}{TN + FP}$

0.942

Area under
the curve

0.939

Negative
Predictive Value
 $\frac{TN}{TN + FN}$

0.913

Precision
 $\frac{TP}{TP + FP}$

0.893

Recall
 $\frac{TP}{TP + FN}$

0.843

$F1 \text{ score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$

0.867

False Positive
Rate
 $\frac{FP}{TN + FP}$

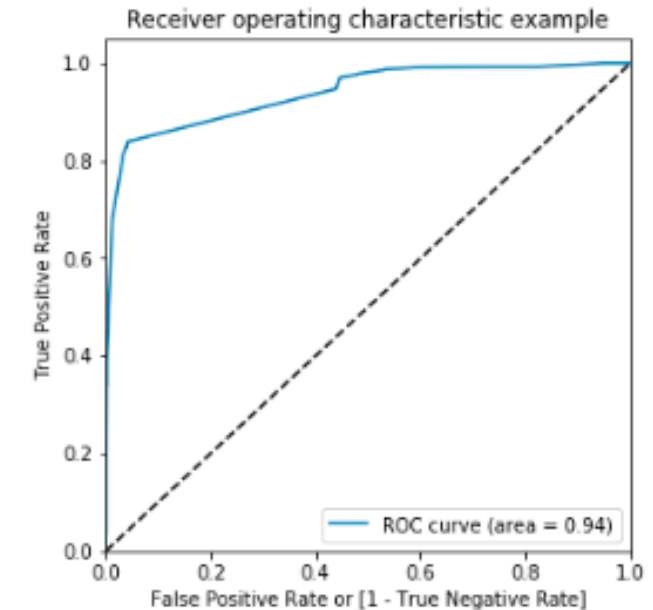
0.057

Positive
Predictive Value
 $\frac{TP}{TP + FP}$

0.893

Cross Validation
Score

0.896



Classification Report

```
print(classification_report(y_pred_final.Converted, y_pred_final.final_predicted))
```

	precision	recall	f1-score	support
0	0.91	0.95	0.93	1734
1	0.91	0.84	0.88	989
accuracy			0.91	2723
macro avg	0.91	0.90	0.90	2723
weighted avg	0.91	0.91	0.91	2723

Finding the Feature Importance

- ❑ 13 features have been used by our model to successfully predict if a lead will get converted or not.
- ❑ The Coefficient (beta) values for each of these features from the model parameters are used to determine the order of importance of these features.
- ❑ Features with high positive beta values are the ones that contribute most towards the probability of a lead getting converted.
- ❑ Similarly, features with high negative beta values contribute the least.
- ❑ The Relative Importance of each feature is determined on a scale of 100 with the feature with highest importance having a score of 100.
- ❑ $\text{feature_importance} = 100.0 * (\text{feature_importance} / \text{feature_importance.max()})$
- ❑ The features are then sorted.

```
feature_importance.sort_values(ascending=False)
```

Tags_Lost to EINS	100.00
Tags_Closed by Horizzon	86.68
Tags_Will revert after reading the email	43.22
Tags_Busy	41.44
Lead Source_Welingak Website	37.19
Last Notable Activity_SMS Sent	29.86
What is your current occupation_Working Professional	14.60
Lead Origin_Lead Add Form	11.87
Do Not Email	-14.18
Tags_Ringing	-18.46
Tags_switched off	-24.93
Lead Quality_Not Sure	-36.40
Lead Quality_Worst	-40.99

dtype: float64

Lead Score Calculation

Lead Score is calculated for all the leads in the original dataframe. Formula for Lead Score calculation is:
Lead Score = 100 * Conversion Probability

	Lead Number	Converted	Converted_prob	final_predicted	Lead_Score
0	660737	0.0	0.110195	0.0	11.0
1	660728	0.0	0.000805	0.0	0.0
2	660727	1.0	0.867357	1.0	87.0
3	660719	0.0	0.000805	0.0	0.0
4	660681	1.0	0.867357	1.0	87.0
5	660680	0.0	0.188037	0.0	19.0
6	660673	1.0	0.867357	1.0	87.0
7	660664	0.0	0.188037	0.0	19.0
8	660624	0.0	0.188037	0.0	19.0
9	660616	0.0	0.188037	0.0	19.0

- ❑ The train and test dataset is concatenated to get the entire list of leads available.
- ❑ The Converted Probability is multiplied by 100 to obtain the Lead Score for each ID.
- ❑ Higher the lead score, higher is the probability of a lead getting converted and vice versa,
- ❑ Since, we had used 0.2 as our final Probability threshold for deciding if a lead will convert or not, any lead with a lead score of 20 or above will have a value of '1' in the final_predicted column.

Based on our model, below features are identified which contribute most to a Lead getting converted successfully.

The conversion probability of a lead increases with increase in values of the following features in descending order

Below are Positive coeff. features:

- Tags_Lost to EINS 100.00
- Tags_Closed by Horizon 86.68
- Tags_Will revert after reading the email 43.22
- Tags_Busy 41.44
- Lead Source_Welingak Website 37.19
- Last Notable Activity_SMS Sent 29.86
- What is your current occupation_Working Professional 14.60
- Lead Origin_Lead Add Form 11.87

The conversion probability of a lead increases with decrease in values of the following features in descending order:

Below are Negative coeff. features:

- Do Not Email -14.18
- Tags_Ringing -18.46
- Tags_switched off -24.93
- Lead Quality_Not Sure -36.40
- Lead Quality_Worst -40.99



Conclusion and Recommendation

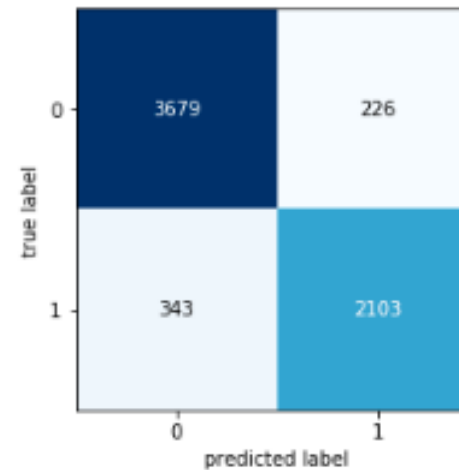
- ❑ We have build the model for leads and got some features which is important for leads conversion and value of some feature is positive coefficients and some negative.
- ❑ We have selected the probability threshold value as per model result and we can select as per the business requirement and so it will decrease and increase the Sensitivity and Specificity.
- ❑ High sensitivity will ensure that almost all leads with a probability of conversion are correctly predicted where high specificity will ensure that leads that are on the verge of converting or not are selected.

Below are TOP 3 coff. features:

- **Tags_Lost** to EINS 100.00
- **Tags_Closed** by Horizzon 86.68
- **Tags_Will** revert after reading the email 43.22

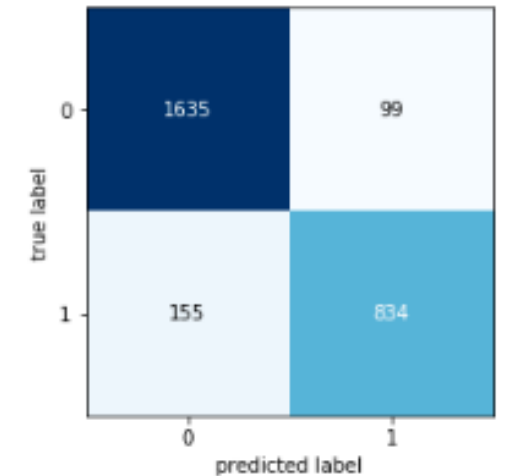
Confusion Matrix

Accuracy Score: 0.9104078097937333



Train Data

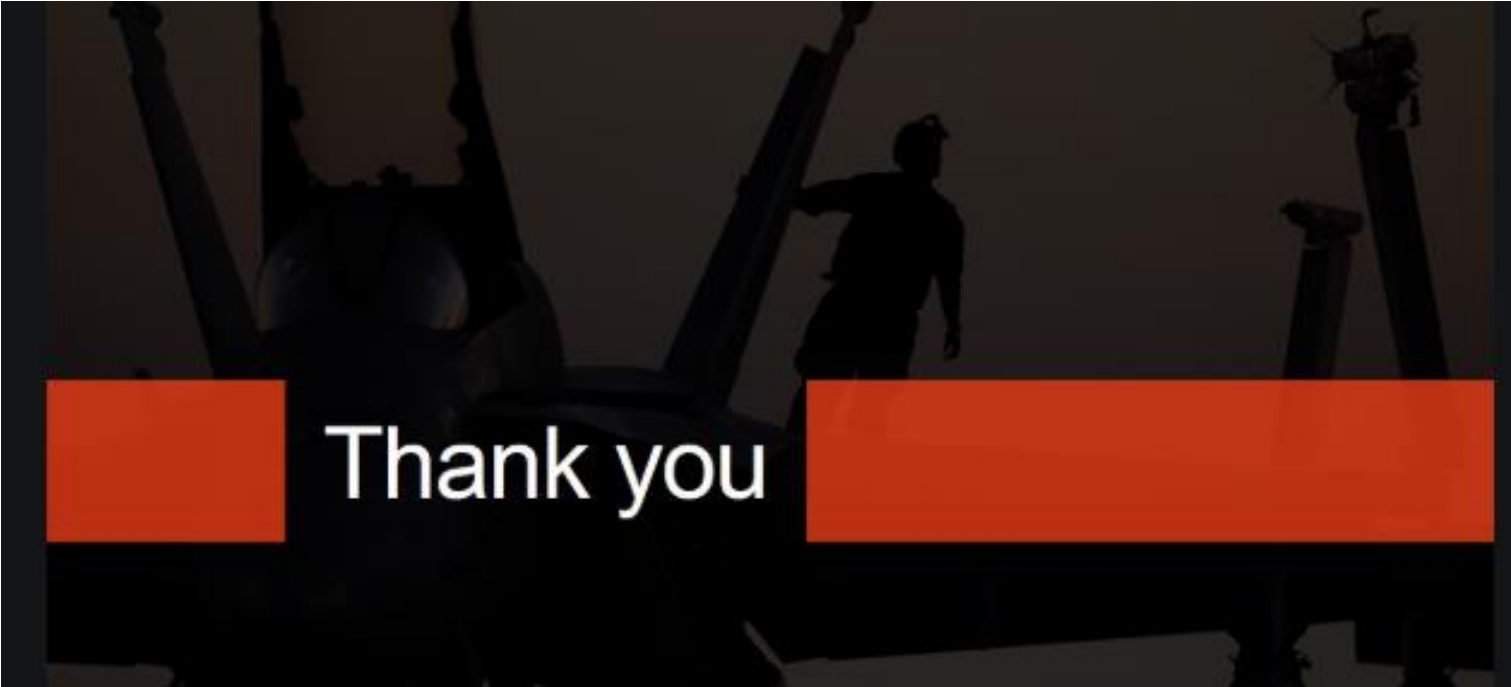
Accuracy Score: 0.906720528828498



Test Data

Rohit Keshari
&
Rahul
Choudhary

upGrad



Thank you