Lead Scoring Case Study

Presented by Aman Rai and Saurabh Singh

Problem Statement

Summary: An Education Company (company X) has a poor lead conversion rate of ~30%. To make the process more efficient, they want to identify the most promising leads - a.k.a "Hot Leads" - through whom they'll be able to attain a high conversion rate. The CEO would like to increase the conversion rate of target customers to 80%

Solution Approach

The approach to solving this would the aforementioned problem statement would be divided into the following **broad** steps

1. Data Preparation: Tasks like cleaning the data, removing outliers, imputing missing values and creating dummy variables
2. Data Modelling: Feature Scaling, logistic regression on train data set and further refining through RFE, VIFs and p-statistics value
3. Performing sensitivity and specificity analysis: mapping ROC curve and finding optimal cut-off and resetting predicted conversion
4. Making predictions on test data and ensuring analysis results (sensitivity, specificity and accuracy) are high and similar to train data results
5. As an additional analysis- generate the AUC Score which will identify how accurately is the model able to distinguish between all the Positive and the Negative class points correctly
6. With a high AUC score (close to 1) - you can confidently predict your HOT LEADS - or promising leads that will definitely get converted

Check point for Analysis

1. Import required libraries and packages

2. Read given datasets “Leads”

3. Perform routine checks

4. Perform Data Prep steps:

```
In [192]:
# checking the percentage of missing values for Asymmetrique Activity Score and Asymmetrique Profile Score

print(round(100*(Leadscore_df['Asymmetrique Activity Score'].isnull().sum()/len(Leadscore_df.index)),2))
print(round(100*(Leadscore_df['Asymmetrique Profile Score'].isnull().sum()/len(Leadscore_df.index)),2))
```

```
45.78
45.78
```

Observation on null check

a. Identifying binary variables and converting them to 0 and 1

b. For categorical variables with multiple levels, create dummy features (one-hot encoded)

- Note: In the data description it was mentioned that many categorical variables will have a 'Select' value which should be treated as a null

- Since in many such variables, the % of select values is high, we're first converting them to dummy variables to ensure we can remove them from data without needing us to remove any rows

- dropping the repeated variables

c. Checking for outliers: and removing outlier cases for 'Tota Visits' cases that are above 99 percentile and below 5 percentile

d. Removing missing values: removing columns 'Asymmetrique Activity Score','Asymmetrique Profile Score' which have ~45% missing values

e. removing 'Select' columns after converting them to dummy variables

Steps before Data Modelling

- Predictor Variables (X): all except 'Prospect ID', 'Lead Number','Converted'
- Target Variable (y): ‘Converted’
- Splitting the data into train and test: 70% and 30% respectively
- Checking the conversion rate for class imbalance: since we have almost 38% conversion rate it ensures that there is no class imbalance

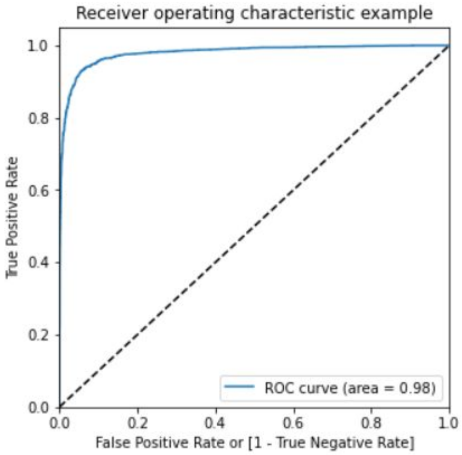
```
In [217]:
# Getting the predicted values on the train set
y_train_pred = res.predict(X_train_sm)
y_train_pred[:10]

Out[217]:
3069    0.992434
1963    0.052035
1567    0.982650
7059    0.043660
6861    0.087001
6235    0.207356
309     0.916686
6089    0.016391
1050    0.987741
5242    0.328476
dtype: float64
```

Generating Prediction values

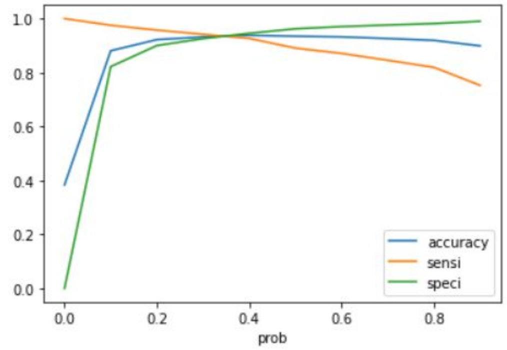
Model Building

1. Initiate with an RFE analysis on predictor variables:
   a. This is to reduce the #of predictor variables
   b. Through RFE we have have only focussing on top 1/3rd variables
   c. Remaining would be removed through p-statistics and VIF
2. Through the above techniques we are now left with 32 predictor variables, out of 183
3. Generate the predicted values on the train set
4. Merge the Predicted data with Actual ‘Converted’ variable
5. Creating new column 'predicted' with 1 if Converted_Prob > 0.5 else 0

Sensitivity & Specificity analysis



1. Initial results based on target vs predicted values - accuracy: 93.4%, sensitivity: 89%, specificity: 96%
2. Plot accuracy, sensitivity and specificity at different probability thresholds. Interaction of all these lines give us the new threshold of 0.35
3. Final results based on target vs predicted values - Accuracy : 93.5%, Sensitivity : 93.4%, Specificity : 93.6%
4. Precision Score: 90.1% and Recall Score: 93.4%



Predictions on Test data

1. Feature Scale the test using using transfer function
2. Filter to keep only 32 predictor columns like the train data
3. Run regression, generate predicted values and perform sensitivity analysis.

FINAL OBSERVATION:

Train Data: Accuracy : 93.5% Sensitivity : 93.4% Specificity : 93.6%

Test Data: Accuracy : 93.3% Sensitivity : 93.1% Specificity : 93.45%

AUC Score:93.3%

Summary: The Model seems to predict the Conversion Rate very well and we should be able to confidently inform the CEO of company X that the set of predicted conversion would categorise as HOT LEADS and they would have a high chance of converting and give the CEO confidence in making good calls based on this model