By Seyed Javidh & Vivek Chowdhury (PGDDS – C28)

Lead Scoring Case Study

This is a case study where a company named "X Education" is trying to increase their Lead conversion rate from 30% to a sensible value. For this purpose, they have hired a data scientist to assist them in understanding the top features which the marketing team must pursue in order to maximize the conversion.

In this case study, we have been given a dataset consisting of 37 columns and a metadata document containing the information about these columns. As a first step, we begin by importing the various different libraries, sub-libraries and packages, like- NumPy, Pandas, Matplotlib, Seaborn, Sklearn and Statsmodel. We will use these packages to perform various operations.

Once we have all the packages, we then proceed with importing the dataframe and analysing the same using df.head(), df.info(), df.shape, df.describe() and all the other methods that we have in our arsenal to perform a dataframe inspection. We have also used df.isnull().sum() to understand the number of null values that we have in the dataframe in each columns.

Once we are done with analysing the dataframe, we then begin with the exploratory data analysis. In the EDA, we checked performed univariate analysis and segmented analysis along with the visualizations to draw inferences from the graphs. These inferences helped us understand the data better and provided more insights.

While performing the exploratory data analysis for the categorical columns, we found sever null values, and the best way to tackle most of them were to either create a new category along with values which had low frequency of occurrence or to coalesce it a value which occurs the most. Some of the categorical columns had values which was almost more than 95% of the other values. In this case, these values were dominating and hence, it was a good idea to drop these columns as they will not add any value to our model.

Next up, we performed EDA on the numerical columns. In-case of numerical columns, we saw many outlier values. For this scenario, we took the help of df.describe() method and checked various different percentile values. This helped us to identify that there are very high values above 99^{th} percentile and also, no values under 5^{th} percentile. So, in this case, it was best to drop the 1% above the 99^{th} percentile and 5% below the 5^{th} percentile.

Once we were done with all the data cleaning and analysis, we then start with the preparing the model for modelling. So as a first step, we had performed One Hot encoding using pd.dummies() and dropping the categorical columns afterwards. After that, we performed the train_test_split(). Once we have the X_train and y_train, we performed the standardization on the numerical columns in the X_train. In-case of X_train, we used fit_transform(). With this, we are done with the initial model preparation.

Now, for the fun part, we start building the Logistic Regression model. So, to build this model, we need to identify the features, which are most important for our model. To do exactly that, we took the help of RFE (Recursive Feature Elimination). With RFE, we gave the logistic regression as one argument and number of features required as another. This led RFE to give

us the top 15 features that are important as per RFE. Using these 15 features, we build the model with the help of Statsmodel. On building the first model and observing the summary, we noticed that there were no columns with very high p-value. So, in this case, we went ahead and found out the Variation Inflation Factor (VIF) for all the features that we have. Once we generated the VIF for all the variables, we saw two variables which had a high VIF of above 5. So, we dropped one of them and generated the VIF again. To our surprise all the VIFs were under 2.

With these 14 columns, we went ahead and built our second model on the Train Data. After building this model, we checked for the accuracy, sensitivity, specificity, precision score and the recall score from the confusion matrix. After this, we drew the ROC curve and got a very high area under the curve (AUC) of 0.97.

Once the ROC was plotted, we then had to find the optimal cut-off value. To find that, we plotted a graph with the accuracy, sensitivity, specificity versus the probability. And we observed that 0.28 was a very good cut-off for our model. We had also seen the same using the recall curve.

Now that we had all the information, we applied the model on the Test Data. And using the confusion matrix found- that the Accuracy, Sensitivity and Specificity are 93%, 93% and 92% respectively. The precision score was 88% and the recall score was 93%. Hence, we concluded that our model was performing extremely well and can be used to drive business insights and business decisions.