Abstract

The case study of a bank (referred to as Bank A) that provides an EMI option to its credit card clients is described in the problem statement. Customers may divide large purchases into smaller, more manageable monthly payments using the EMI feature while avoiding revolving interest. The bank provides EMIs at many points, including the point of sale, after a transaction, and as a balance-on-EMI plan for certain customers. While it offers a lucrative income stream through processing fees, interest, and foreclosure penalties, credit card issuers devote a lot of time and effort into ensuring that a high percentage of expenditures are turned into EMI. The promotion of Bank A's post-transaction EMI proposal to clients, however, was difficult. The bank established a contact center where representatives would make cold calls to consumers and promote the EMI facility because conversion rates from digital ads were low. This led to higher conversion rates but also higher expenses. To solve this problem, Bank A made the decision to only contact clients who were more likely to turn their transactions into EMIs, saving money on the expense of specialized agents. To assist Bank A in prioritizing its credit card transactions for EMI calling is the goal of the problem statement. In order to properly allocate its resources and increase income, the bank needs to determine which clients are most likely to choose EMI repayment.

Problem Statement

The goal is to create a precise and efficient prediction model that can assist the bank in deciding which Credit Card transactions to prioritize for EMI calls. The challenge at hand is to create a predictive model that can precisely estimate the likelihood that a specific credit card transaction would turn into an EMI. The model may be constructed using the development data, and using the same reasoning, it is possible to estimate the likelihood that transactions in the validation data will convert to EMI.

Approach to the solution

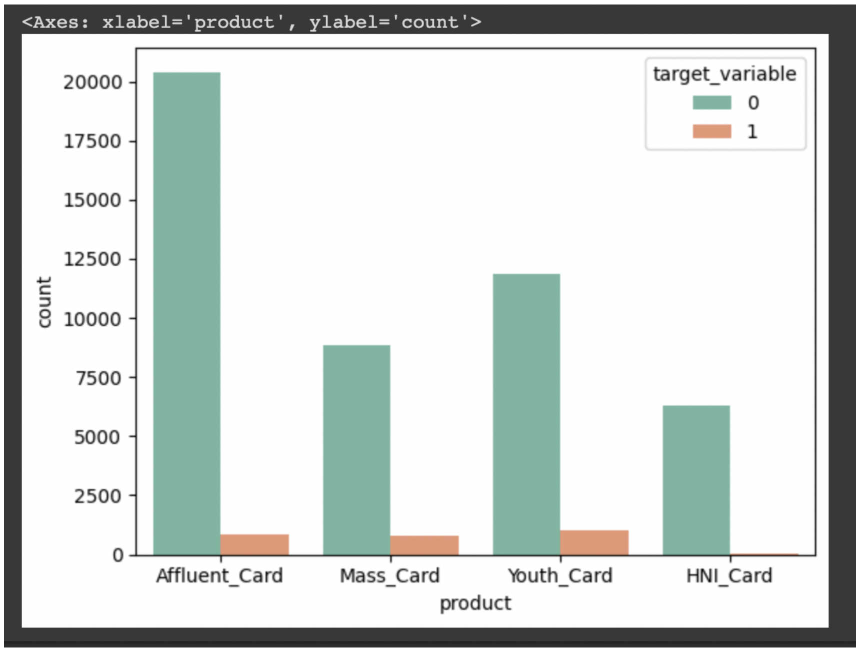
Data exploration and cleaning

At first, it is very crucial to understand the dataset with which we are dealing with, understanding the meaning of each column from the given **data_dictionary**. After that, column by column I had analysed the following data parameters :

- **Data type** : For training the model, it is necessary to know the type of data in dataframe. If they are 'categorical' type, then by using one- hot encoding, we convert the categorical data to numerical values.
- **NULL values:** The number of null values present in each column is determined, The columns with large number of null values are dropped, while those which contains the null values values comparatively less, are been filled either by zero , or by replacing it to the mean value of the column being filled. However a thorough understanding of the meaning of each column is necessary, and selectively choose to fill the null values, there are some columns where we cannot replace it with either 0 or mean value.
- **Determining negative and positive values in columns:** Determining negative values in a column is important as it can impact the analysis and interpretation of the data. Negative values in a column where they are not expected can indicate data entry errors or inconsistencies, which can impact the accuracy of the data.

  Negative values can affect the interpretation of the data and results of analysis or modeling. For example, if the correlation between target_variable and age was sightly high, indicating the age can be considered for training the model, however the age of the customer was given as negative.
- **Column Scaling** : The range of values for each feature in the dataset may be normalized by scaling the columns, which is crucial. This is crucial when working with numerical data since some machine learning algorithms may perform poorly when various characteristics have varying sizes, units, or magnitudes. In order to determine which columns need to be scaled, one can use the statistics of each column, indicating minimum value, maximum value, mean value , 25th percentile , 50 percentile, etc using "df.info".

Insights



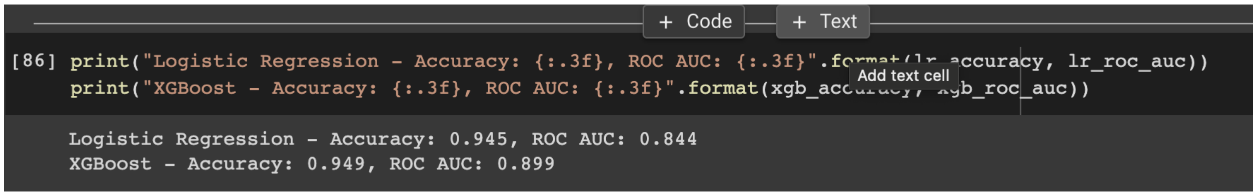
```
<Axes: xlabel='product', ylabel='count'>
```

As the above count plot gives us the information that, **youth_card** product of the EMI option was highly opted by the customers. While the Affluent_Card being less preferred.

Evaluation of the Model

This code evaluates two different classification models **Logistic Regression** and **XGBoos**t using a dev data set. For each model, the predicted probabilities of the positive class (EMI conversion) are obtained for the test set. Then, three performance metrics are calculated:
Accuracy score: the proportion of correct predictions (both true positives and true negatives) out of all predictions made.
ROC AUC score: a metric that evaluates the ability of the model to distinguish between positive and negative classes, taking into account all possible decision thresholds.
Precision-Recall AUC score: another metric that evaluates the ability of the model to distinguish between positive and negative classes, but with a focus on the trade-off between precision and recall.

These metrics are stored in the variables **lr_accuracy**, **lr_roc_auc**, **xgb_accuracy**, **xgb_roc_auc**, **lr_prc_auc**, and **xgb_prc_auc**



```
[86] print("Logistic Regression - Accuracy: {:.3f}, ROC AUC: {:.3f}".format(lr_accuracy, lr_roc_auc))
     print("XGBoost - Accuracy: {:.3f}, ROC AUC: {:.3f}".format(xgb_accuracy, xgb_roc_auc))

     Logistic Regression - Accuracy: 0.945, ROC AUC: 0.844
     XGBoost - Accuracy: 0.949, ROC AUC: 0.899
```

As the above code snippet suggests, **Logistic regression** gives 0.945 accuracy with ROC AUC score being 0.844. While the Training with **XGBOOST** the outcoming accuracy is 0.949 with ROC AUC value as 0.899.
Conclusion: It suggests that, XGboost model appears to give higher accuracy as compared to the Logistic regression model with higher ROC AUC accuracy score. Hence

we further predicted the probability of EMI conversion for validation data using the XGBoost model.

Made by: Richa Sachan