# DATA 1030 Midterm Report

Manlin Li
*Brown Data Science Initiative*
https://github.com/Marlenahaslee/DATA1030_Project

# 1 INTRODUCTION

Customer churn is when a customer stops using the service of a company's product, and in this case, stop being a paying client for a particular business. This problem is very important, especially for the audience of banks and credit card companies. For credit card companies to be profitable, they need to prevent their clients from gaming the system by churning. This project aims to build a classification tool for banks and credit card companies to help them predict whether a costumer will churn or not based on his/her demographical information as well as the record of how they have used the credit card. As an early warning of whether the existing customers are going to churn, this tool is useful in helping the bank and credit card companies in taking some action to maintain their customer retention.

The dataset obtained consists of a total of 10127 data with 21 valid columns. (There were originally 23 columns, but the author of the dataset suggested the last two columns containing data from Naïve Bayes Classifier Level to be removed.) The target variable `Attrition_Flag` is marked as either "Existing Customer" or "Attrited Customer".

Based on the literature research, all publications have the same goal of classification given the emphasis of this dataset on whether customers churned or not. Luc Chetboun and Thomas Konstantin were both interested in building a classifier to predict the `Attrition_Flag` for users (whether they churned or not). Moreover, they both chose SVM as one of their machine learning models, given that the mechanism of SVM is to find the best hyperplane that separates all data points into two classes. Luc Chetboun found in his project that AdaBoost achieved the best performance with precision of 0.88 and recall of 0.82 [1], and Thomas Konstantin in his project had random forest as the most robust model among all. Thomas Konstantin solves the class imbalance problem by SMOTE (Synthetic Minority Over-sampling Technique), and he also applied a PCA after the one-hot encoding to compress the dimensionality of the dataset [2].

# 2 Exploratory Data Analysis

During exploratory data analysis, the distribution, existence of null values, and unique values for each variable is explored by printing out the summary statistics and visualization. Bivariate visualizations between the target variable and the predictor variables are also conducted.

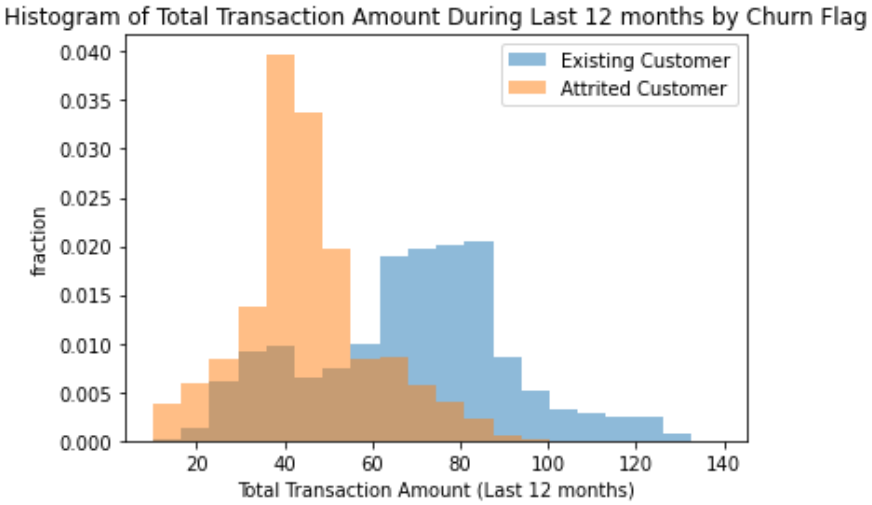


**Figure 1** This category-specific histogram demonstrates the distribution of total transaction amount during the last 12 months by the churn flag of the customer: "Attrited Customer" or "Existing Customer". The bell-shaped pattern for each category indicates that they both approximate a Gaussian distribution. However, it can be easily seen that the distribution of total transaction amount for attrited customers is centered around 40 while that for existing customer is centered around 70. At the meantime, the former category has a smaller variance. This obvious difference in the distribution of transaction amount indicates that this variable is very likely to be an influencing factor in the classification model.

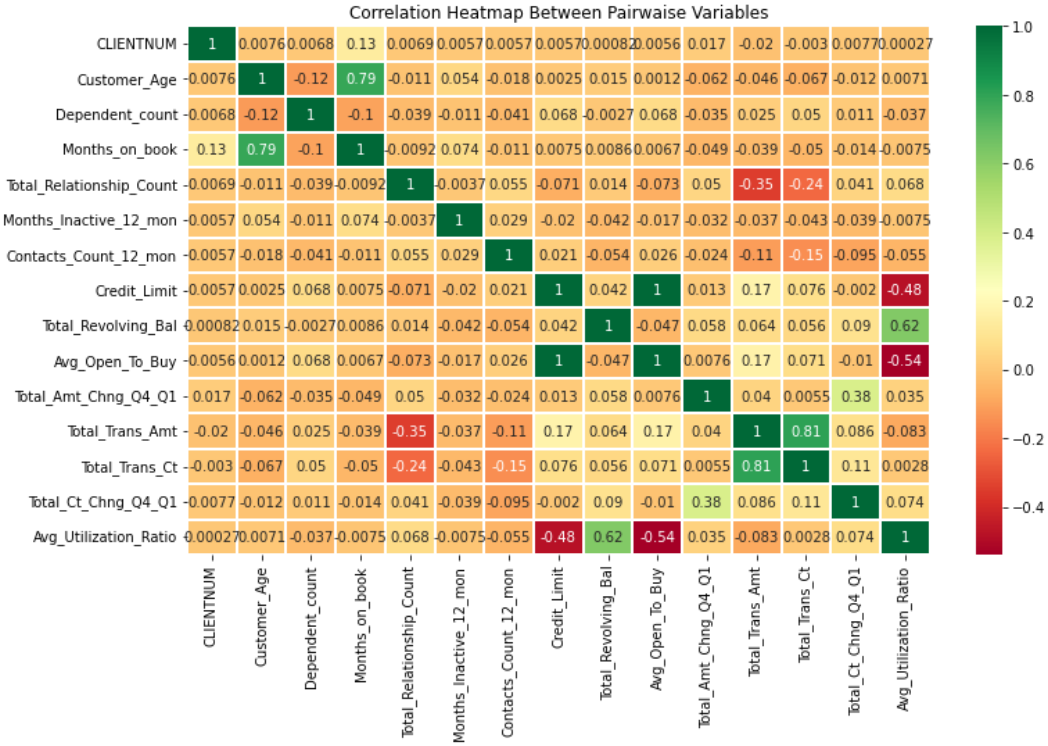


**Figure 2** This correlation matrix heat map gives an overview of the linear relationship between predictor variables that are of continuous type. It can be seen from the graph that a cell with dark

green/red indicates a strong positive/negative correlation. For example, the correlation coefficient between `Months_on_book` and `Customer_Age` is 0.79, indicating that the period of relationship with bank has a positive linear relationship with the age of the customer. This visualization is important to understand because it may indicate the multicollinearity issue between predictor variables, which requests future attention when constructing the models.

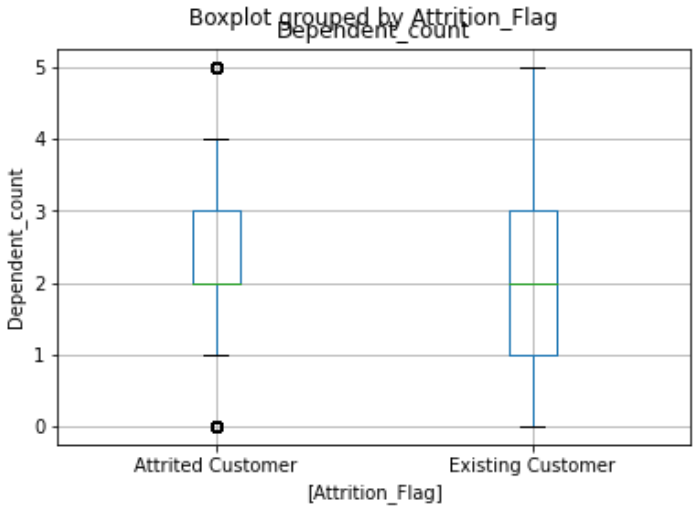


**Figure 3** The above box plot shows the distribution of customers with different number of dependents on their credit cards. From the plot, the range of number of dependents for "Attrited Customer" is smaller than that for "Existing Customer". Also, the overlapping between the mean and the 25% quantile for "Attributed Customer" indicates the skewness for that category comparing to "Existing Customer" which follows a more bell-shaped distribution.

# 3 Methods

## 3.1 Data Splitting & Preprocessing

After EDA, the `CLIENTNUM` column is dropped since it is a unique identifier that is meaningless in our classification model. Given that the target variable is a binary variable, it is also encoded by 0(Existing Customer) or 1(Attrited Customer) before splitting the data.

The data is first split into a testing set with 20% data, and the rest 80% data is used for the cross-validation process in the hyperparameter tuning.

The dataset is assumed to be Independent and Identically Distributed (iid). Each data entry represents the credit card usage pattern and personal information of an individual customer. Also, it was shown during the previous EDA section that each customer only has one unique data entry in the dataset. Therefore, it is neither a group-structured data nor a time-series data.

In the encoding step, a pipeline consisting of StandardScaler, OneHotEncoder, OrdinalEncoder, and MinMaxScaler is implemented to avoid the leaking statistics. StandardScaler is applied to the continuous variables that have no clear pattern of being bounded including `Dependent_count`,

`Months_on_book`, etc. MinMaxScaler is applied to variables such as `Age` and `No. of Contacts in the last 12 months` which have clear bounds of 0 to 100 and 0 to 12, respectively. OrdinalEncoder is applied to orderd categorical variables including `Income_Category`, `Education_Level`, `Card_Category`. OneHotEncoder is applied to categorical variables `Gender` and `Marital_Status`. As a result, there are 23 features included in the preprocessed data.

## 3.2 Model Training & Hyperparameter Tuning

In the developed machine learning pipeline, each model will be trained and tested on 10 different random states to account for the uncertainties due to splitting and non-deterministic machine learning methods. After being preprocessed, the input model will be performed on the training and validation data. An exhaustive search over specified parameter values method is used in finding the best hyperparameters with 4-fold cross validation. After obtaining the best estimator, it will then be applied on the transformed test data. In the end, the final test scores (accuracy scores) and the best estimator for each random state will be returned as the results. The reason for choosing accuracy score as my evaluation metric is as following: given that the problem is a classification problem and that customers who is likely to churn as well those who will be continuing clients are both important, and the dataset is not severely imbalanced, accuracy is a straightforward score to measure.

Using the designed machine learning pipeline, 7 machine learning models were trained and tested together with a baseline model for comparison: a logistic regression with L1 regularization , a logistic regression with L2 regularization, a K-nearest neighbors classifier, a decision tree classifier, a random forest classifier, a support vector machine classifier, and a XGBoost classifier. For each random state and each cross-validation fold, there is an exhaustive search over all specified hyperparameter values for each estimator. Below are the parameters tuned for each model and their corresponding choice of values:

| Model | Parameter Choices |
|---|---|
| Baseline | **strategy**: most_frequent |
| Lasso | **C**: 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3 |
| Ridge | **C**: 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3 |
| KNN | **k_neighbors**: 3, 6, 9, 12, 20 |
| Decision Tree | **max_depth** : 5, 20, 50, 100, **max_features**: 0.5, 0.6, 0.7, 0.8, 0.9 |
| Random Forest | **max_depth** : 20, 40, 50, **max_features**: 0.7, 0.8, 0.9 |
| SVC | **gamma** : 1e-2, 1e-1, 1e0, 1e1, 1e2, **C**: 0.1, 1, 10 |
| XGB | **learning_rate** : 0.03, **colsample_bytree**: 0.9, **subsample**: 0.66, **max_depth**: None, 3, 10, 30, 50 |

**Figure 4** Hyperparameter value choices for each model

# 4 Results

## 4.1 Model Performance Comparison & Selection

After each round of hyperparameter tuning, the best parameter combinations are extracted and used for fitting on the holdout test set. Below are the average accuracy scores for the best estimator in each of the 10 random states:

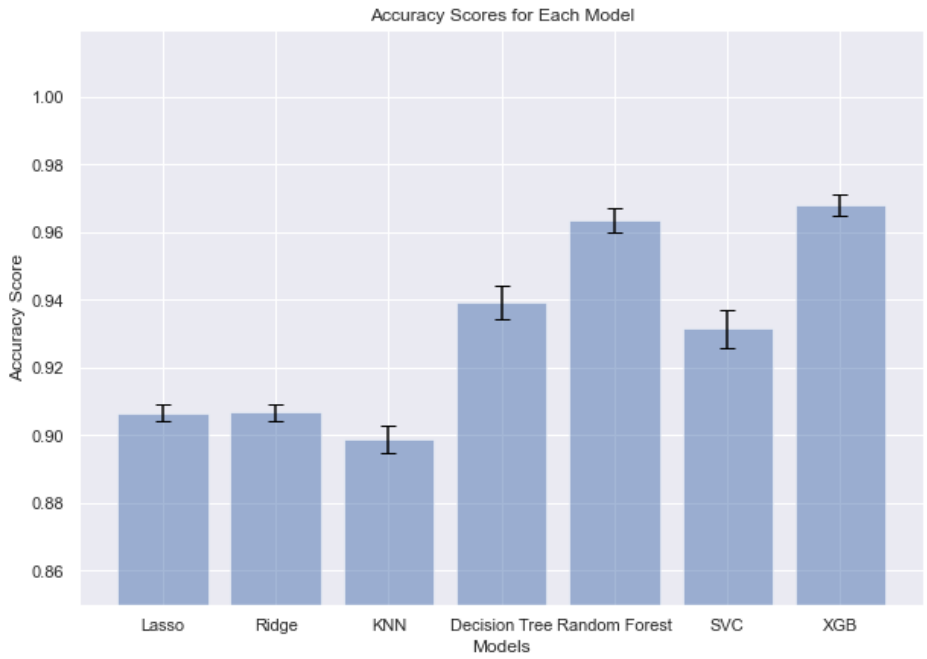


**Figure 5** Average accuracy scores for the best model in 10 random states

As it can be observed, the XGBoost classifier achieves the highest accuracy score among all models which indicates its predictive power over the other classifiers. The best parameter combinations for XGBoost classifier includes a max depth of 10, a learning rate of 0.03, a subsample ratio of 0.66 along with other default parameters setting.

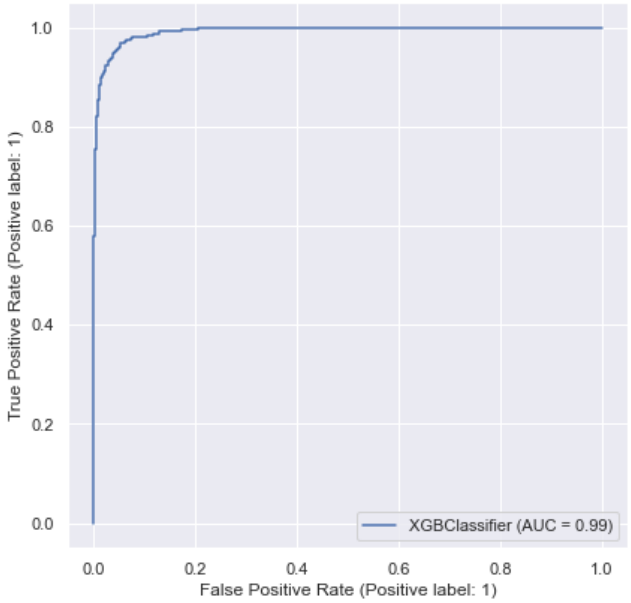


**Figure 6** ROC curve of XGBoost

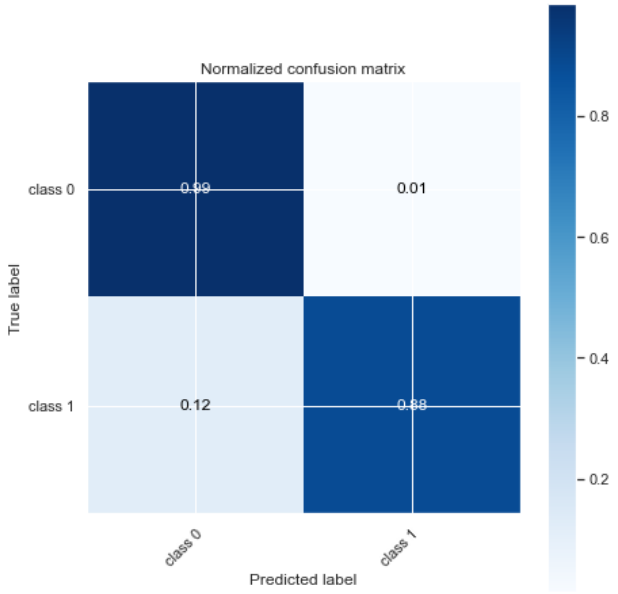


**Figure 7** Normalized Confusion Matrix for XGBoost

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.98 | 0.99 | 0.98 | 1707 |
| 1 | 0.93 | 0.88 | 0.90 | 319 |
| accuracy | | | 0.97 | 2026 |
| macro avg | 0.95 | 0.93 | 0.94 | 2026 |
| weighted avg | 0.97 | 0.97 | 0.97 | 2026 |

**Figure 8** Classification Report for XGBoost

By fitting the final model again with a single random state, its performance can be further shown by the visualizations and classification report. As indicated above, the final XGBoost classifier shows an AUC of 0.99, a recall score for the positive class (attrited customers) of 0.88, a precision score of 0.93, and a f1-score of 0.90.

Comparing it with the baseline model, a dummy classifier that predicts the most frequent class in the training set as the output, it shows apparent improvement in the predictive power. The accuracy scores for the dummy classifier in the 10 random states have a mean of 0.84 and standard deviation of 0.05 while the accuracy scores for the XGBoost classifier have a mean of 0.97 and a standard deviation of 0.003. Therefore, the best model XGBoost classifier achieves an average accuracy that is 2.6 standard deviation above the baseline dummy classifier. By the same token, the baseline model has an average accuracy score that is 4.3 standard deviation below the XGBoost classifier.

## 4.2 Feature Importance & Interpretation

Since interpretability is crucial in real-world cases, the next step is to better understand how the model works on the dataset and which features are most relevant for prediction. Three methods to see the global feature importance will be covered and the first one is the permutation feature importance which is defined to be the decrease in a model score when a single feature value is randomly shuffled. As it can be seen in Figure 9, `Total_Trans_Ct`, `Total_Trans_Amt`, `Total_Relationship_Count`, and `Total_Revolving_Bal` are the ones that are most relevant to the prediction.

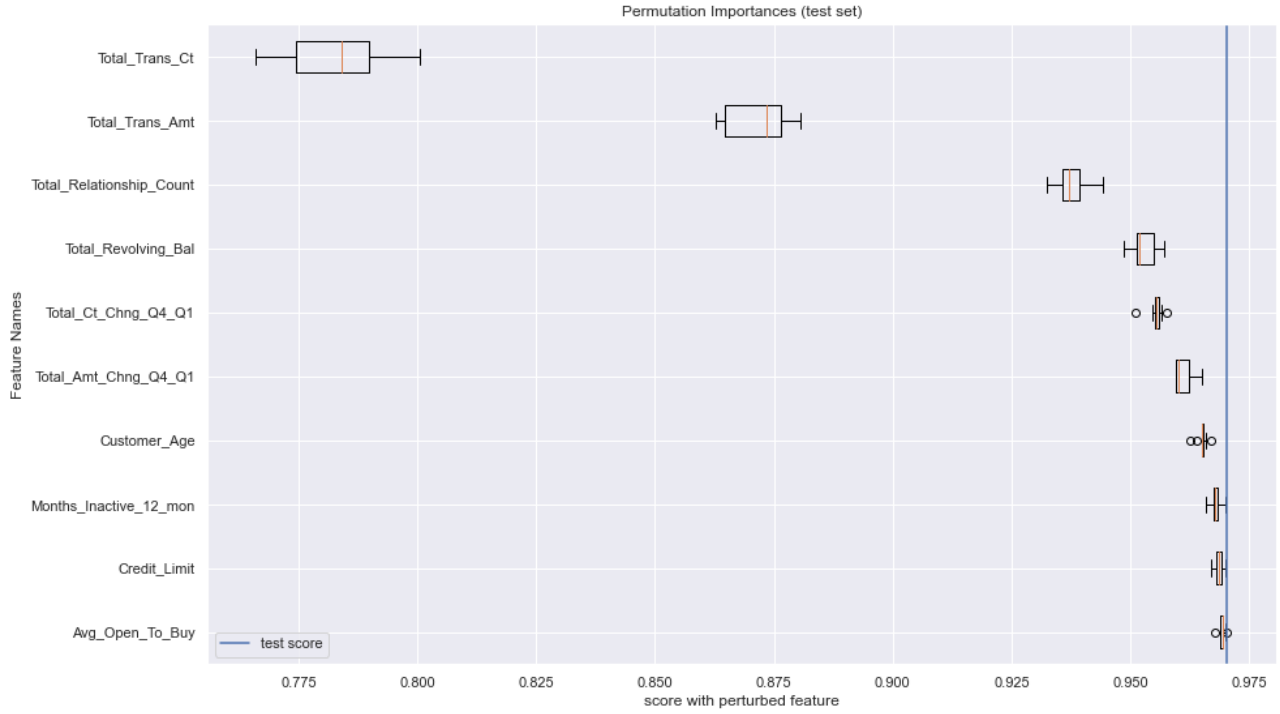


**Figure 9** Top 10 Permutation Importance

Besides that, the top 10 important features that are generated by the built-in feature importance method for tree-based algorithms are shown below. The features that have the highest importance are `Total_Trans_Ct`, `Total_Revolving_Bal`, `Total_Relationship_Count`, and `Total_Trans_Amt`.

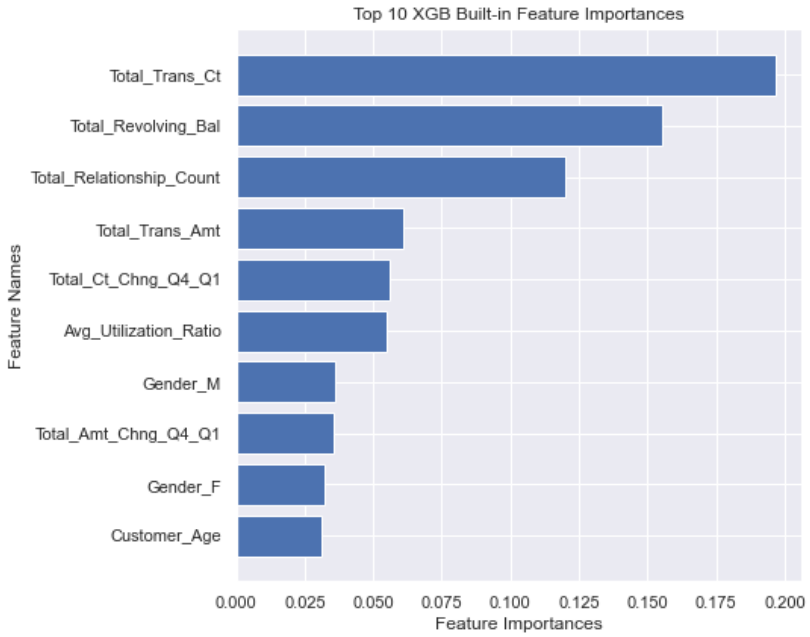


**Figure 10** Top 10 XGB Built-in Feature Importance

Lastly, the global feature importance can also be shown in Figure 10 by applying SHAP values. The most relevant features are `Total_Trans_Ct`, `Total_Trans_Amt`. `Total_Revolving_Bal`, and `Total_Relationship_Count`.


**Figure 11** Top 10 Permutation Importance

As it can be seen, the global feature importance returned by the three different methods are consistent with each other. The most important features are `Total_Trans_Ct`, `Total_Trans_Amt`. `Total_Revolving_Bal`, and `Total_Relationship_Count`. The least important features are `Marital_Status`, `Months_on_book`, `Card_Category`, and `Education_Level` as shown in Figure 11.

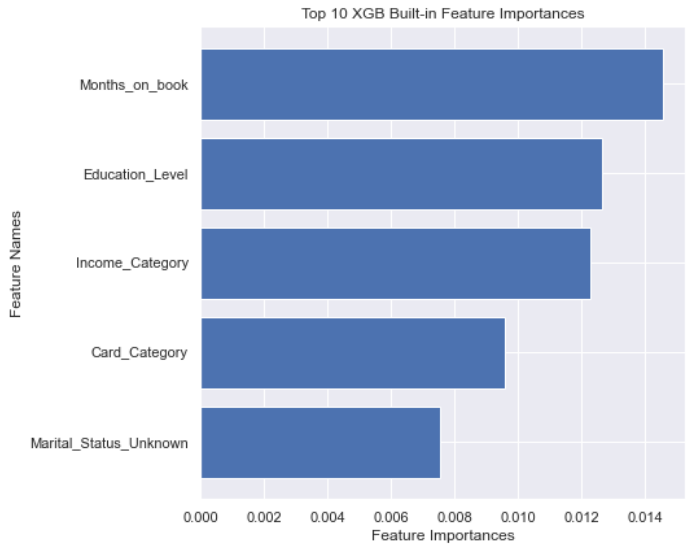


**Figure 12** Least Important Features

The most important features are quite consistent with what I expected, but there are some least important features that I find counterintuitive and interesting to know that they may play less important roles in deciding whether the client is likely to churn. For example, I expect `Months_on_book` to be pretty decisive in determining whether the client is likely to churn or not, but on the contrary, it is the least determining feature.

Besides the global feature importance discussed above, local feature importance of several features is also looked up by visualizing their SHAP values via the first 1000 data. For example, the SHAP value for `Customer_Age` is shown in Figure 12 on the y-axis. Besides, the color suggests that there is not really interaction effect between `Customer_Age` and `Total_Trans_Ct` since there is no distinct vertical pattern of coloring.

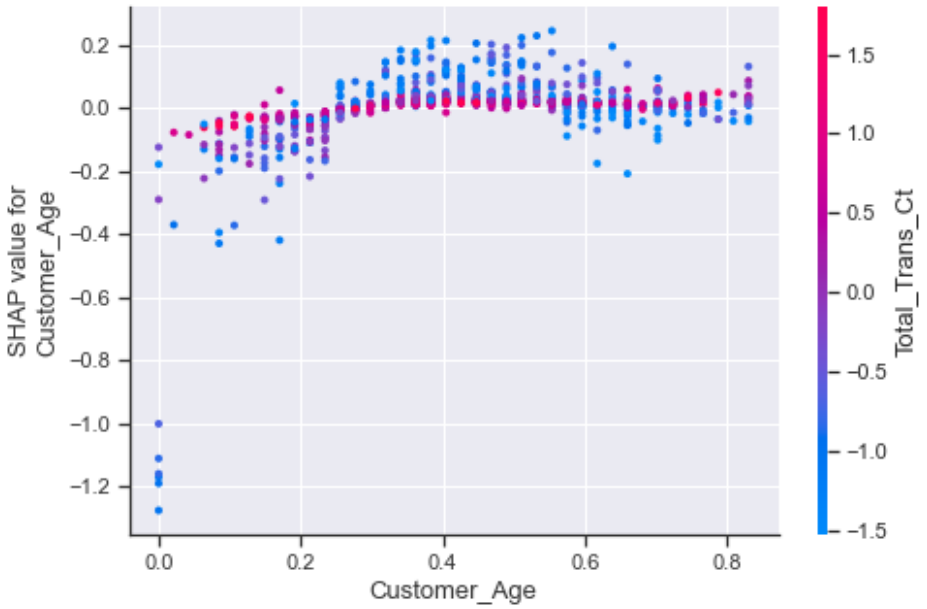


**Figure 12** SHAP for `Customer_Age`

# 5 Outlook

Given that there is a slight class imbalance of my dataset, I would like to leverage methods like oversampling and SMOTE to adjust the class distribution of my data in order to mitigate the potential overfitting issue of the model. What's more, given the class imbalance property of the dataset, the weak spots of using the accuracy score as the performance evaluation metrics since it has troubles in distinguishing between the numbers of correctly classified data of different classes. So, using this metric may lead to some churning clients undetected by the model. Therefore, an improvement could be taking the recall score and f1 score into consideration as well when choosing the most powerful model. So, if more data of customers who churned can be collected in the future, the model performance will be boosted by having more information of the minority class.

# Reference

[1]: Chetboun, Luc. Data Exploration, Model Evaluation on BankChurners.
https://www.kaggle.com/chetbounl/data-exploration-model-evaluation-on-bankchurners

[2]: Konstantin, Thomas. Bank Churn Data Exploration And Churn Prediction.
https://www.kaggle.com/thomaskonstantin/bank-churn-data-exploration-and-churn-prediction/notebook