**Predicting Churning Customers**

Data 1030 Midterm Report —— Yingfei Hong

October 12, 2021

1. **Introduction**

A manager at the bank is disturbed by more and more customers leaving their credit card services. The problem I am trying to solve is to predict the "churned customers" for the bank managers based on the given data so they can proactively go to the customer to provide them better services and turn customers' decisions in the opposite direction. Predicting churn is very important especially when clear customer feedback is absent. Retaining existing customers and thereby increasing their lifetime value is something everyone acknowledges as being important, however, there is little the bank managers can do about customer churn if they don't see it coming in the first place. This is where predicting churn has its value. Early and accurate churn prediction empowers CRM and customer experience teams to be creative and proactive in their engagement with the customer. In fact, by simply reaching out to the customer early enough, 11% of the churn can be avoided [1].

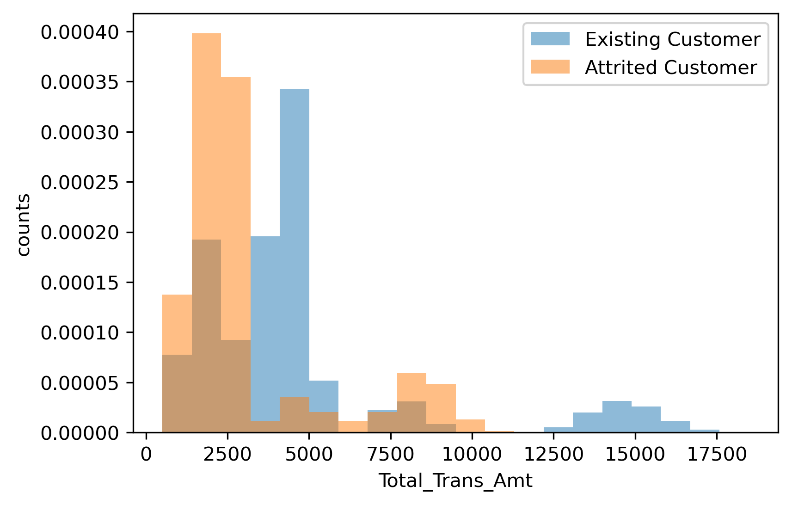
This dataset comes from LEAPS [2] and contains 21 columns and 10127 data points with ‘Attrition\_Flag’ as its target variable in which “Attrited customer” means that this customer is the churned customer we are looking for.

**Figure 1** The distribution of ‘Attrition\_Flag’

It is an imbalanced dataset since the proportion of the ‘Attrited customer’ is about 16.07%. Besides, through the distribution of the target variable, this task is certainly a typical classification problem.

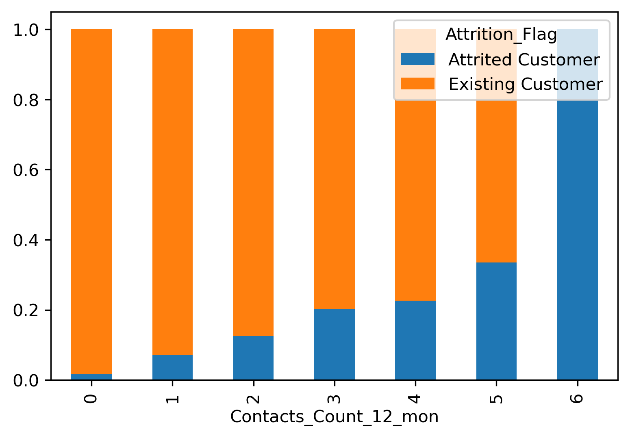
****The rest 20 features are described in the following table which contains their feature name, type, and meaning.

**Table 1** Description for each feature

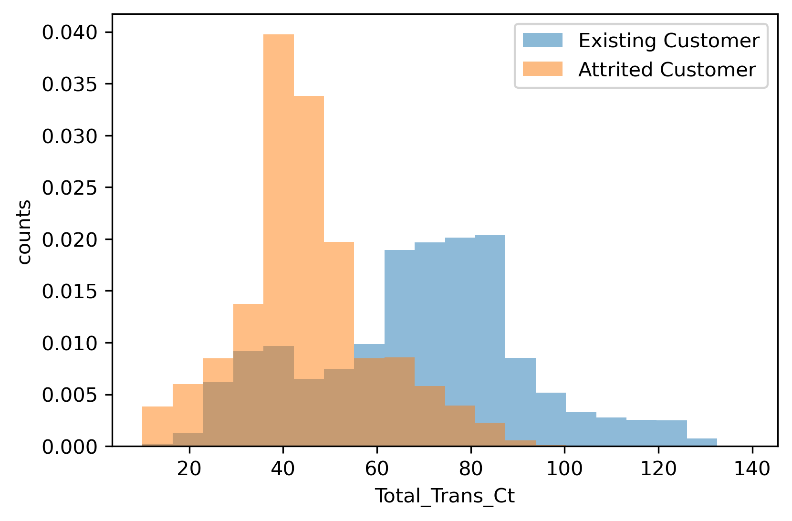
For this data, there are two main tasks, one is to improve the performance of predicting churned customers while the other is to find the most influential factors that make the customers "churn". However, finding the factors that matter most is a common strategy when enhancing the model performance so I rather treat them as a same task. Several projects have already been done to solve this problem and achieve good results.

Thomas[3] used SMOTE which is an approach to address imbalanced datasets by oversampling the minority class and found great improvement when processing the data generated by this strategy. The result showed that compared to the raw data, this new data could improve the F1 score from average 0.6 to average 0.9. Andi[4] looked into the details about the raw data and found some interesting relationships between the features and the target variable. The EDA showed that the likelihood of the customers' leaving is related to the money they spend annually, the months of inactivity in their bank account and their credit limit. Joseph[5] used Random Forest and LightGBM to predict with 97% recall and 95% accuracy and plotted the importance of these features which showed that the transaction feature ranked top in both models, so we need to look these features thoroughly when doing exploratory data analysis.

1. **EDA**

I’ve plotted the relationship between every feature and the target variable and found some relations that are worth notice.

**Figure 2** This graph displays how the number of contacts is distributed across two different customers. It seems that churning customers have had more contacts in the last 12 months with the bank managers.

**Figure 3** This graph shows that the total transaction amount of attrited customers is smaller than that of existing customers which explains why this feature ranks top in both models of Joseph’s projects.

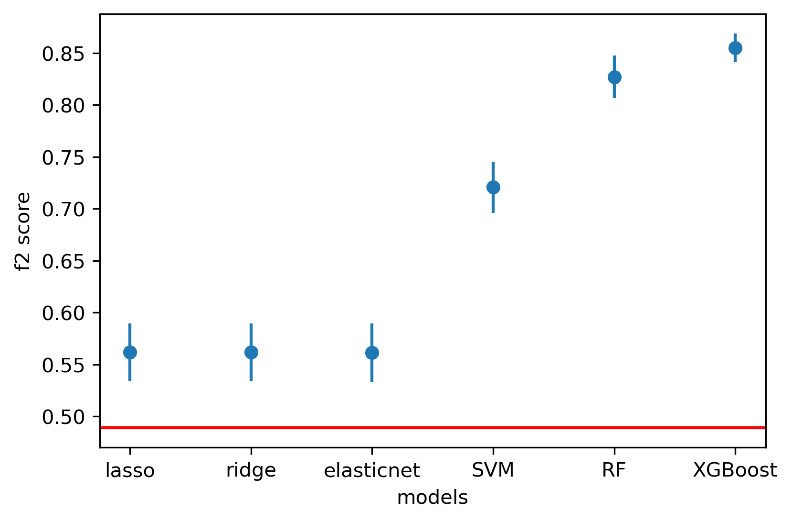
**Figure 4** This graph has a similar pattern as figure 3 which shows that total transaction count is also a important feature when we try to classify attrited customer from the existing customer.

1. **Method**

**3.1 Preprocessing**

This dataset is not IID since all features are not identically distributed. It does not have a group structure, nor does it have time-series data. But it is imbalanced, so it is better to use stratify method when splitting. The train size is set as 0.6 and the validation size as well as the test size both as 0.2 because it is how people normally split their data when they have lots of data points.

I treated discrete numerical features (dis\_fea) as ordinal features and put them directly into the model since they are already in numerical form. Categorical features (cat\_fea) need to be treated with OneHotEncoder because it is not sensible if we put gender and marital status in order. For ordinal features(ord\_fea), OrdinalEncoder is the best fit because there is ordinal information contained in educational level, income category, and card category. For continuous numerical feature(con\_fea), I chose StandardScaler. Since in these columns, 'Customer\_Age', 'Monts\_on\_book', and 'Total\_Trans\_Ct' are nearly normally distributed though some are skewed while other features have long tails which are not suitable to use MinMaxScaler.

In this case, I also transform the target varible by setting all "Existing Customer" into 1 and all "Attrited Customer" into 0 to make it machine-comprehensible because it is a binary classification problem and the type of the values in the target variable is ‘string'.

There are some missing values in some demographic variables like educational level, income category, and marital status and we can treat them as one special category since they are all categorical features.

**3.2 Parameter tuning**

The ML pipiline that I use is quite simple. For each random state, I first split and preprocess the data as the description above and then run all parameters combinations on the training data and validation data to find the best model in each random state, and calculate the test score based on the best model. In the end, we will have 10 test scores with 10 random states for each model.

In this project, I've tried six different models including three logistic models with different regularization methods and SVM, RandomForest, and XGBoost. The parameters of each model are as the following table.

**Figure 5** Parameters used for tuning

The metric that I use is f2 score because the loss of treating an attrited customer as the existing customer is far greater than that of predicting an existing customer as an attrited customer. Therefore, we need to pay more attention to recall, and thus I choose the f beta score as metric with more emphasis on the recall score.

1. **Results**

**4.1 Model evaluation**

From the errorbar plot below, we can see that all six models achieve better f2 scores than the baseline score. Here the baseline is the model whose predictions are all 1s ("Attrited Customer"). Among these models, XGBoost has the best performance with the best average test score and the lowest variance.

**Figure 6** F2-score for the best models over 10 random states

1. **Reference**

[1] Why customers leave &amp; what can banks do? Tiger Analytics. (2020, September 16). Retrieved October 12, 2021, from <https://www.tigeranalytics.com/blog/addressing-customer-churn-in-banking/>.

[2] Predict Customer Attrition Using Naïve Bayes Classification. ATH Leaps. Retrieved October 12, 2021, from <https://leapsapp.analyttica.com/cases/11>.

[3] Konstantin, T. (2021, May 1). Bank churn data exploration and churn prediction. Kaggle. Retrieved October 12, 2021, from <https://www.kaggle.com/thomaskonstantin/bank-churn-data-exploration-and-churn-prediction>.

[4] IDW, A. (2021, January 31). Customer churn - EDA, 95% ACC and 85% recall. Kaggle. Retrieved October 12, 2021, from <https://www.kaggle.com/paotografi/customer-churn-eda-95-acc-and-85-recall>.

[5] Chan, J. (2021, January 13). Bank Churners Classifier (Recall: 97% accuracy: 95%). Kaggle. Retrieved October 12, 2021, from <https://www.kaggle.com/josephchan524/bankchurnersclassifier-recall-97-accuracy-95>.

1. **Github repository**

https://github.com/YingfeiHong01/Data1030-FinalProject