

Module 4: Classification

Abstract

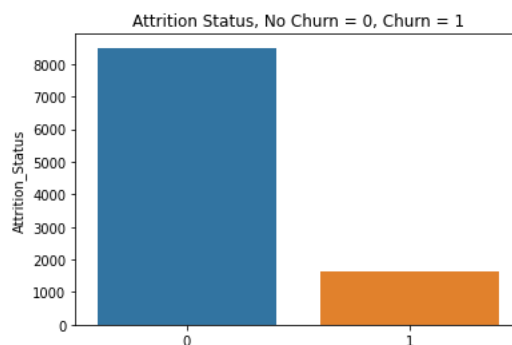
It is important for credit card companies to identify reasons for customer churn so they can understand the issue and take necessary steps to address customers who leave their company. In order to understand this behavior, many features are involved that could affect whether a user churns or not. Machine learning classification is ideally suited to address this problem.

Design

My original dataset consists of 10127 entries and 24 features. The features describe a customer ID, demographic information, and credit card behavior for each customer. Each customer also has a corresponding target that indicates whether that person is an Existing customer (not churned) or an Attrited customer (churned).

Categorical values were converted to numerical values, both by dummyming as well as assigning graded numbers. These categorical values were mostly demographic features.

The target was transformed to a binary (0: not churned, 1: churned). Our dataset is imbalanced, so we had 83.93% of 0, and 16.07% of 1.

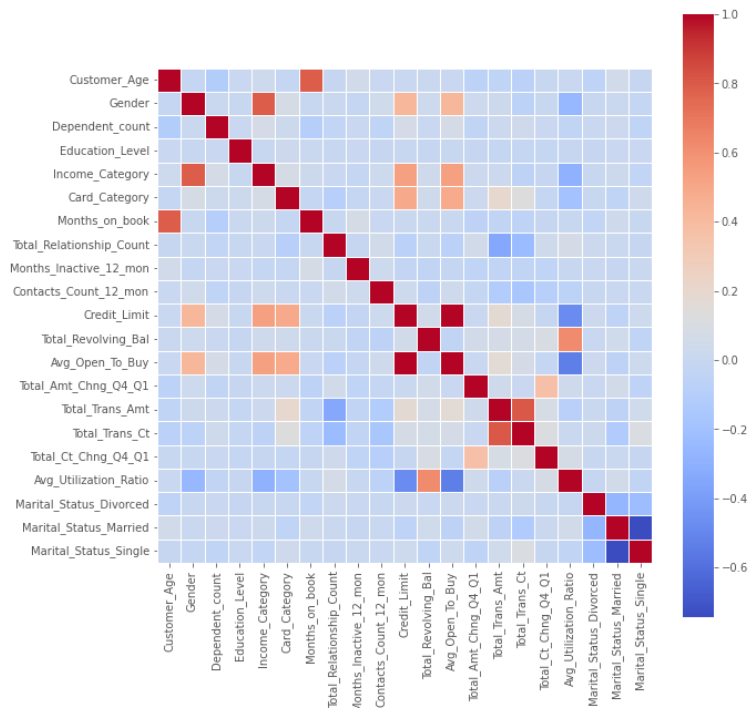


The final feature list can be shown as:

Customer_Age	int64
Gender	int8
Dependent_count	int64
Education_Level	float64
Income_Category	float64
Card_Category	int64
Months_on_book	int64
Total_Relationship_Count	int64

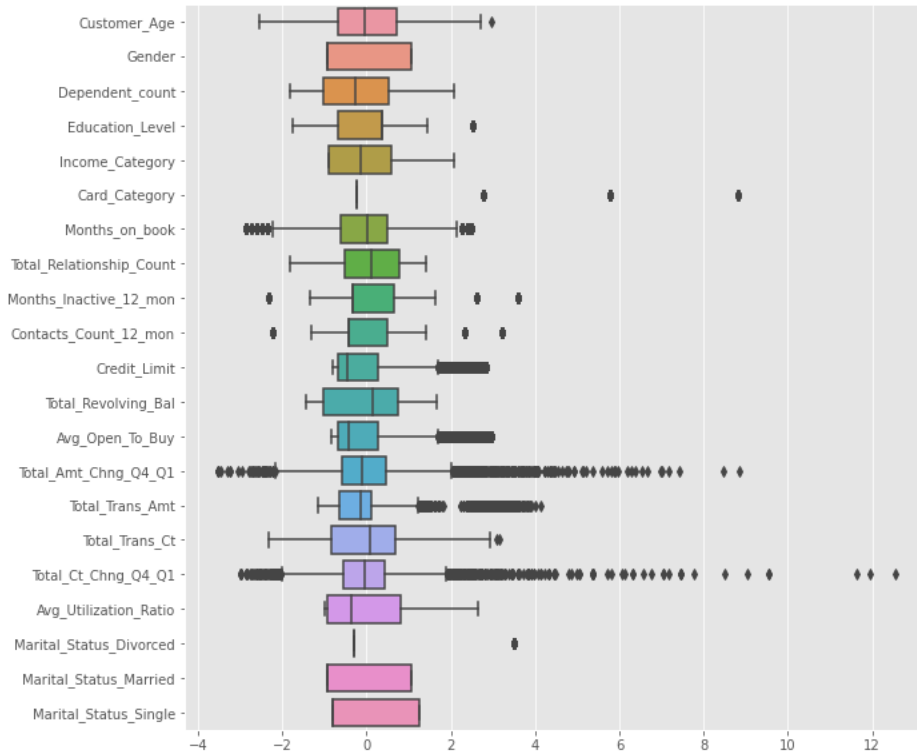
Months_Inactive_12_mon	int64
Contacts_Count_12_mon	int64
Credit_Limit	float64
Total_Revolving_Bal	int64
Avg_Open_To_Buy	float64
Total_Amt_Chng_Q4_Q1	float64
Total_Trans_Amt	int64
Total_Trans_Ct	int64
Total_Ct_Chng_Q4_Q1	float64
Avg_Utilization_Ratio	float64
Marital_Status_Divorced	uint8
Marital_Status_Married	uint8
Marital_Status_Single	uint8

Some of the features have strong correlation with each other, but most have weak correlation. As seen by the heatmap below:



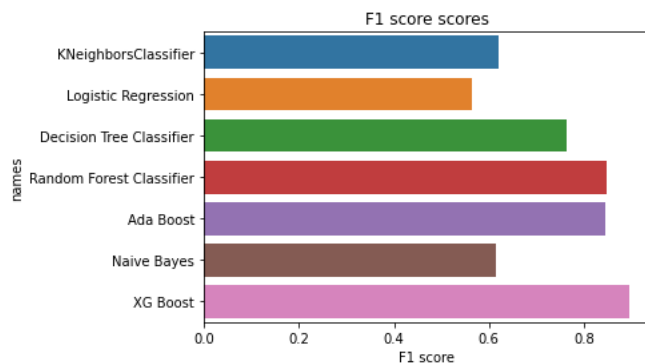
The data was split into Training and Test portions for the modelling portion. The Train and Test portions.

We also standardized the features, which is important for certain classification models. The final distributions of each feature can be seen below.



Data and Results

The first step was baselining. In order to get an idea of what models were compatible with the data, we tried out 7 models. The results are the models are shown below:



Full Data can be seen as:

names	Jaccard Scores	Accuracy	Precision	Recall	F1 score
KNeighborsClassifier	45.10%	88.94%	69.43%	56.27%	62.16%
Logistic Regression	39.21%	88.60%	73.76%	45.57%	56.33%
Decision Tree Classifier	61.90%	90.92%	65.71%	91.44%	76.47%

Random Forest Classifier	73.49%	95.46%	92.73%	77.98%	84.72%
Ada Boost	73.00%	95.16%	88.04%	81.04%	84.39%
Naive Bayes	44.29%	88.45%	66.67%	56.88%	61.39%
XG Boost	81.07%	96.69%	91.40%	87.77%	89.55%

After baselining, we have identified XG Boost as the most promising model. From here, the goal is to improve the XGBoost model by optimizing the hyperparameters of the model. I briefly tested the impact of different hyperparameters on the model, and the ones that seemed to make the most impact were:

N_estimators: number of estimators

Learning_rate: step size shrinkage

Reg_alpha: regularization term

Scale_pos_weight: controls the balance of positive and negative weights.

We used a grid of values that netted 480 unique hyperparameter combinations that we created models for. After, this the top model was one where:

N_estimators = 50

Learning_rate = 0.6

Reg_alpha = 0.7

Scale_pos_weight = 1

When comparing this model with the model before optimization, we see a very large improvement in the Recall with a small sacrifice in Precision. Since we care about the balance of these two metrics as evaluated by the F-1 score, we are happy with these results.

	Precision	Recall	F-1 Score
Before Optimization	91.40%	87.77%	89.55%
After Optimization	91.05%	90.21%	90.63%

Conclusion

- XGBoost was the most promising model when evaluated over F1 score from our list of 7 baselining models.
- By tuning our XGBoost model, we obtained a final F1 score of: **90.63%**

Next Steps

- Determine if the current churn rate is acceptable and if not, devise mitigation strategies targeted at customers with high probabilities of churning
- Continue to improve the current model with more granular hyperparameter optimizing, gathering different data from customers, and updating the model as more users are added to the database.
- Try further ensembling methods such as Stacking or Voting ensembling

Tools

Pandas – Data preprocessing and feature engineering

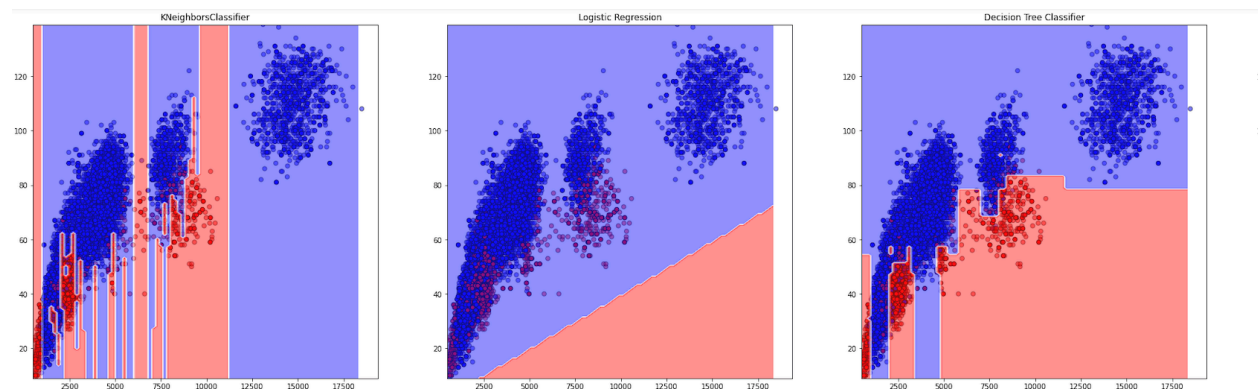
Scikit-Learn – The larger majority of models and model utilities

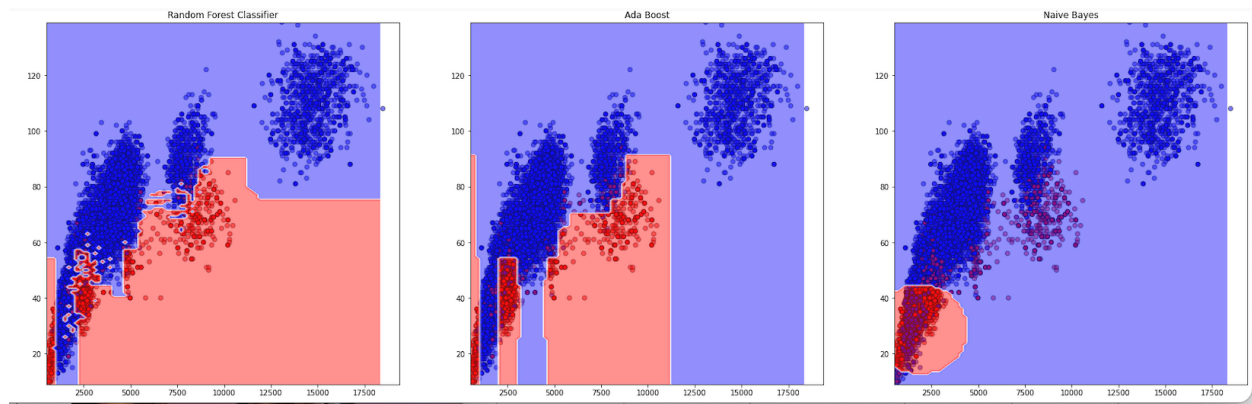
XGBoost – XGBClassifier model

Matplotlib and Seaborn – Graphing and visualization

Extra:

I also worked on 2D decision trees to visualize how different models created decision boundaries with two selected features. The ensembling methods seem to be more precise towards fitting the data. However, it is clear that the higher dimensional models are more effective.





Communication

For additional information, please contact kenhua15@gmail.com. The project will also be posted on my github found here: <https://github.com/kenhua15/Metis-Projects>