Credit Card Default Prediction Using Machine Learning Techniques- DATA 1030 Final Report

Megan Sindhi
Brown University- Data Science Institute
December 7, 2021
Github Repository: https://github.com/megan-sindhi/data-1030-project

1. Introduction

The last thing a financial institution wants to see is a default on a credit card, as it is a loss on their part. They vet people before giving out cards, and enforce credit limits, but there are always some who still default. If these institutions could find a way to figure out who would default before they do, they could work with the clients to prevent default and minimize company losses.

Project Goals and Dataset

This project works to help solve this problem by creating a model to predict credit card default. This model will be a binary classification model, and the target variable is if the client defaults. The data set for this project comes from the UCI Machine Learning Repository and was published by Chung Hua University and Tamkang University in Taiwan. It is a dataset that contains information on 30,000 credit card clients in Taiwan. There are 25 features in this dataset: an ID column, 23 features that can be used to predict default, and the target variable, which is if the client defaults on their credit card payment. The features that can be used to predict default are the amount of given credit, age, marital status, gender, highest education degree, monthly repayment status, bill amount, and payment amount for April to September of 2005.

Dataset Literature Review

This dataset has been used extensively in publications on predicting credit card default. Yeh and Lien use this dataset to introduce a method on how to estimate the probability of default and then perform a regression using this as the target variable. They also create several classification models to compare and can get the error rate down to 0.18 using K-Nearest Neighbors with all the features [1]. Other articles using this dataset investigate the best tools for each step of the machine learning pipeline, such as feature selection. Laborda and Ryoo use this dataset to investigate the effectiveness of forward selection, backward selection, and correlation coefficients and chi-squared tests as feature selection methods for common classification models with this dataset. They created initial Logistic Regression, K-NN, SVM, and Random Forest models with a Mean Absolute Error of about 20-25% and were able to improve all of them by at least 18% using forward stepwise selection. All the models ended up with between 4-7 variables, and variables that showed up in all models included monthly repayment status variables and education level [2]. Both papers were successful in correctly classifying a high percentage of defaults, and their insights provide a good starting point, but there is still room for improvement. Every step closer to perfect classification means less money lost for financial institutions.

2. Exploratory Data Analysis

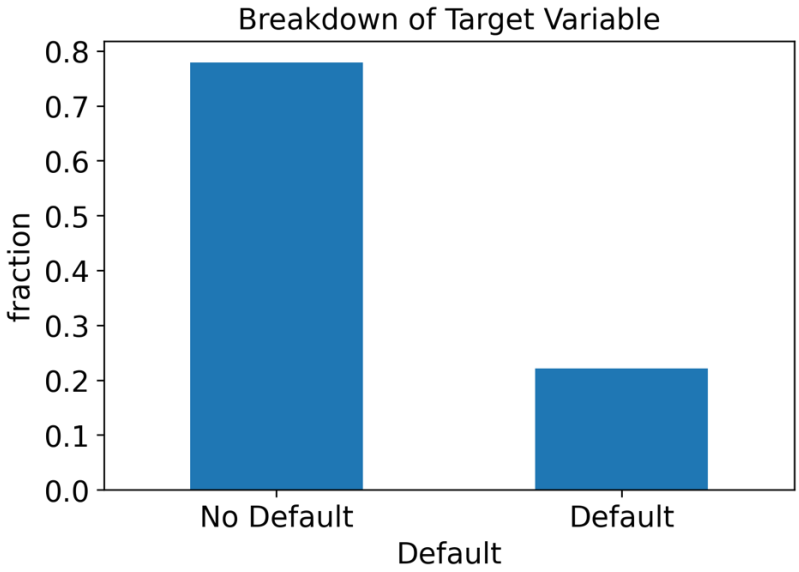

Figure 1: Breakdown of Target Variable

This figure shows us the breakdown of the target variable: if the person will default next month. It is important to look at this breakdown because the classes are imbalanced, and there significantly more people that don't default versus people that do. This is common in loan and credit card default problems, and something that must be accounted for, especially when splitting the data.

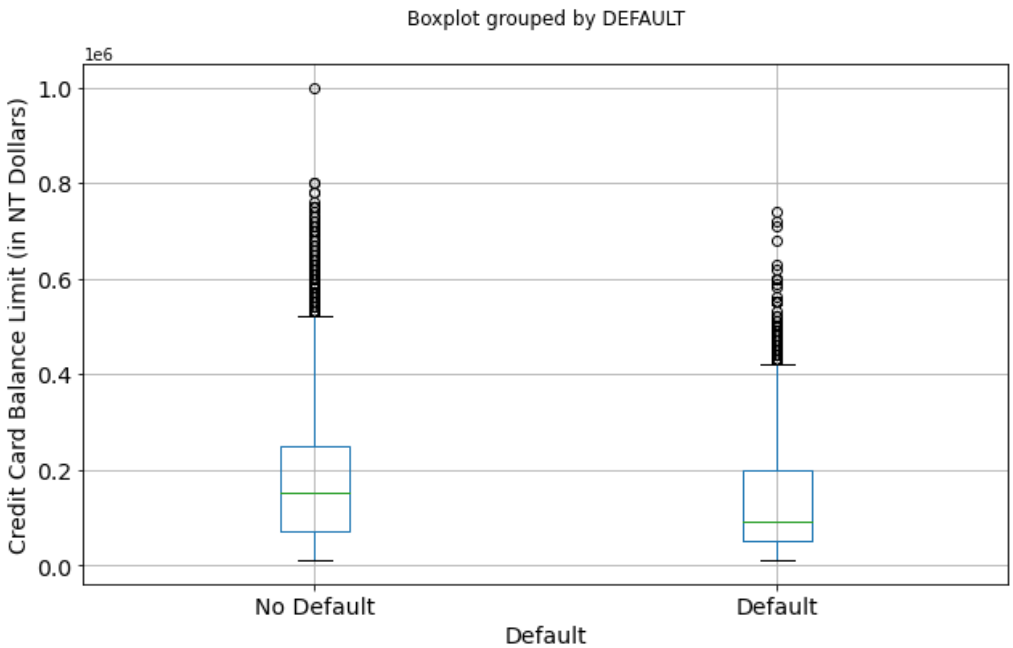

Figure 2: Boxplot of Credit Card Balance Limit Grouped by Default

The above figure displays a boxplot of range of credit card balance limits grouped by if the client defaults or not. The distributions are similar, but the upper end of the range is higher for the non-default group compared to the default group. The median is also 60,000 NT (New Taiwan) dollars larger for the non-default group. There seem to be some differences in the distributions of credit card balance limits for defaulters and non-defaulters.

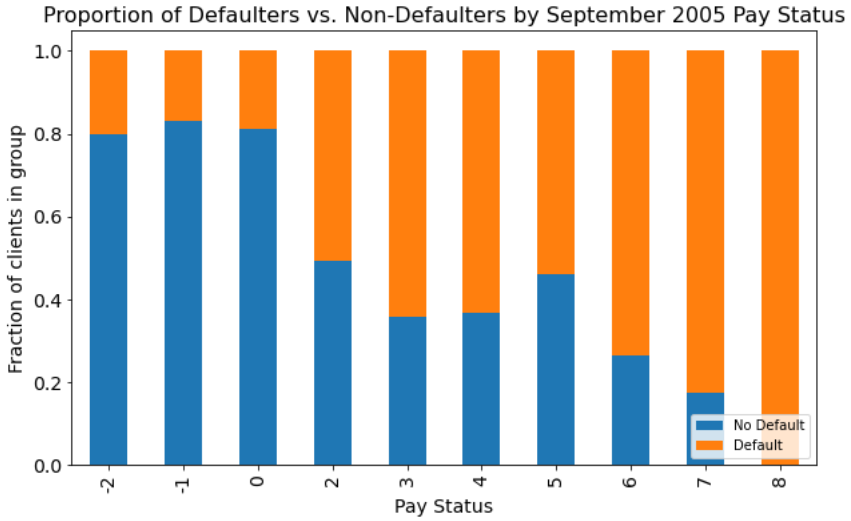

Figure 3: Stacked Bar plot of Proportion of Defaulters by Most Recent Payment Status (where -2= No balance, -1= Paid in full, 0= Made minimum payment, and from 1 onward the number denotes how many months delayed the payment is 1= 1 month payment delay, 2= 2-month payment delay …)

This figure displays the proportion of people in each pay status group for September 2005 who have defaulted or not. The no balance, paid in full, and minimum payment groups have only a 20% rate of default. This percentage generally increases with each additional month in payment delay. At 3 and 4 months, over 60% of the people in the groups have defaulted. At 8 months, everyone in the group has defaulted. This and the 5 other payment status variables have potential to be important features in the predictive models as there seems to be a relationship between default and pay status.

3. Methods

Splitting and Preprocessing

This data is iid and does not have a group structure. Each point is a different client and contains all information on that client. Since the data is imbalanced as seen in the EDA, a stratified split is used to keep the proportions the same as the initial data breakdown of about 22.1% defaulters and 77.9% non-defaulters. 60% of the data was put in training, 20% in validation, and 20% in testing, so that we can tune model parameters, train the model, and test the generalizability of the model on unseen data. K fold cross validation was not used because the data is on the larger side. There is no missing data, so preprocessing the data consisted of transforming the columns based on their datatypes. The min max scaler was applied on age because it is reasonably bounded, with a range of 21 to 79. The standard scaler was applied to all the bill amount and payment amount variables as well as the credit card balance limit because

they were not reasonably bounded. These variables all had a large percentage of their values within 0-100,000 but there were outliers up to 500,000 or even 1 million in some cases. Payment status was already encoded as an ordinal feature. However, there is not a definitive answer of what is better or worse between no balance and paid in full, so this feature was reencoded using one hot encoding. Additionally, the one hot encoder was used on sex, marital status, and education level. Sex and marital status cannot be ranked, and education level had another column which is also difficult to rank. With this preprocessing there are 88 features.

Model Selection

Six different machine learning models were trained: a logistic model with L1 regularization, a logistic model with L2 regularization, a logistic model with ElasticNet regularization, Random Forest classifier, K-Nearest Neighbors classifier, and XGboost classifier. Because of the importance of correctly predicting defaults (positives), the f1 score was used to evaluate the models. Each model was hyperparameter tuned with a grid search to find the best parameters for each model. Below are the parameters tried for each model.

Model	Parameters
L1 Logistic Regression	**C**: 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3, 1e4
L2 Logistic Regression	**C**: 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3, 1e4
ElasticNet Logistic Regression	**C**: 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 1e3, 1e4; **l1_ratio**: 0.01, 0.1, 0.25, 0.5, 0.75, 0.9, 0.99
Random Forest	**min_samples_leaf**: 1, 2, 5, 10; **max_depth**: 1, 3, 10, 30, 100; **max_features**: 0.5, 0.75, 1.0
K Nearest Neighbors	**n_neighbors**: 3, 10, 30, 100; **weights**: uniform, distance
XGBoost	**reg_alpha**: 0, 1e-2, 1e-1, 1e0, 1e1, 1e2; **reg_lambda**: 0, 1e-2, 1e-1, 1e0, 1e1, 1e2; **max_depth**: 1,3,10,30,100; **colsample_bytree**: 0.5, 0.7, 0.9; **subsample**: 0.5, 0.66, 0.75, 1

Figure 4: Table of Possible Values for Hyperparameter Tuning

After tuning, the best parameters for each model were taken and used on a test set to compare across models. The mean f1 score across 5 random states (to account for randomness in splitting and in Random Forest and XGBoost) for each model was calculated and used to select the final model. The final model was trained 100 times using the best hyperparameters from the tuning to once again account for randomness in splitting and the model. For each split, the dataset was once again split into 60% training, 20% validation, and 20% test since XGBoost was the final model chosen and its early stopping requires a validation set. For each random state the model and test score were saved.

4. Results

The figure below shows the mean and standard deviation of the test score over the five random states:

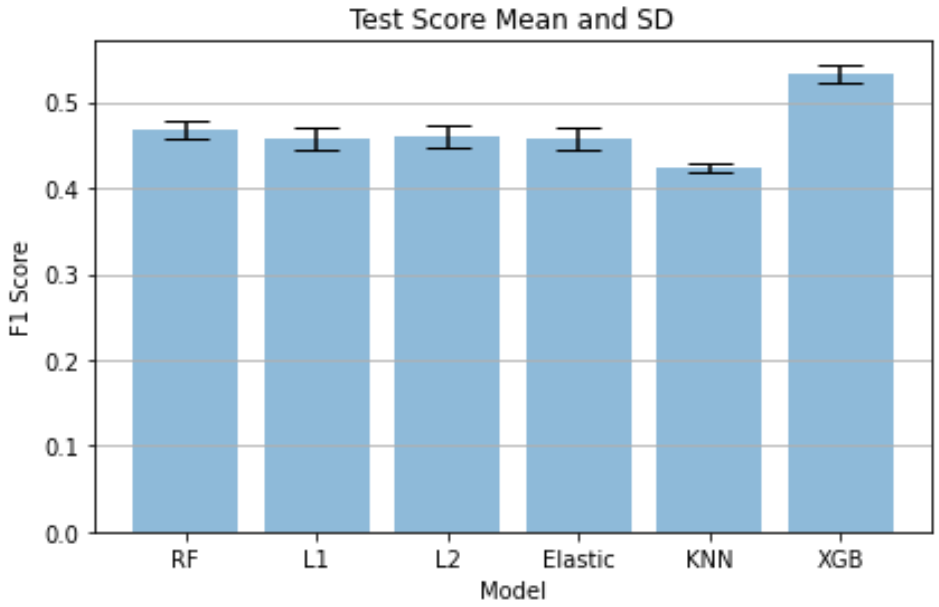

Figure 5: Comparison of Different Model Test Scores

XGBoost was clearly the highest performing model on the test set, so it was chosen as the final model. Based on the parameter turning, an alpha value of 1, lambda of 1, max depth of 10, col sample by tree value of 0.7 and subsample value of 0.66 were used in the final model. Over the 100 random states, the models returned an average f1 score of 0.537 and a standard deviation of 0.011. The baseline f1 score (calculated by predicting all points as positive) was 0.362, which is about 16 standard deviations below the mean, so there was a significant improvement.

Interpretation

Global feature importance was calculated using the built in XGboost feature importance. Three different importances were calculated: weight, gain and cover. Additionally, local feature importance was calculated using SHAP values. The graphs showing the 10 most important features according to each metric below:

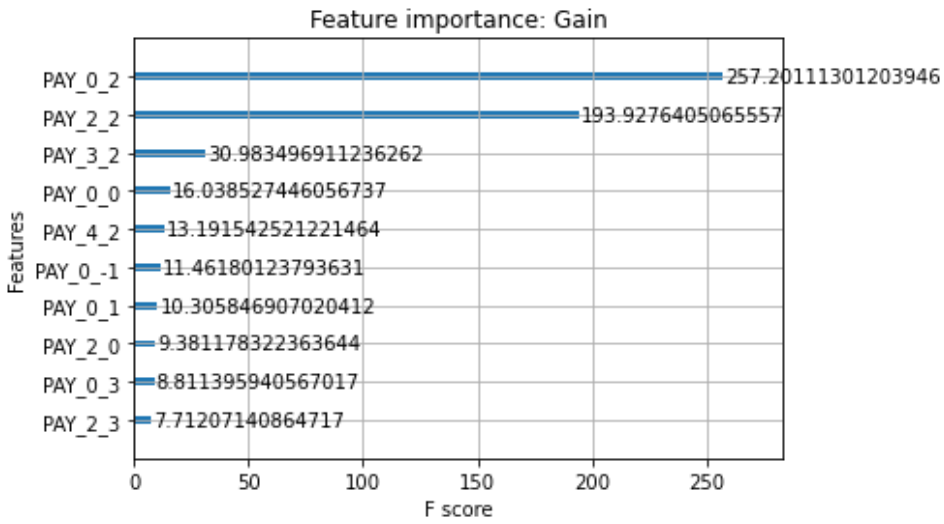

Figure 6: Top 10 Most Important Features: XGBoost Gain

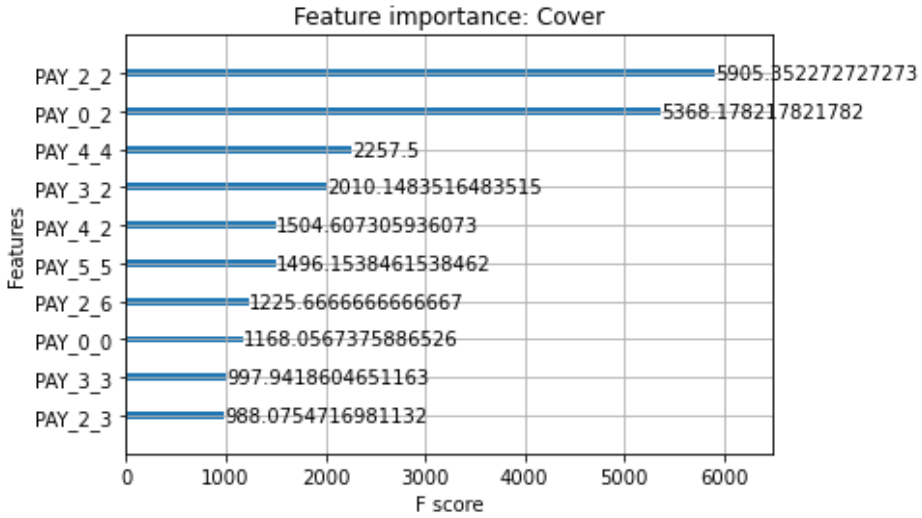

Figure 7: Top 10 Most Important Features: XGBoost Cover

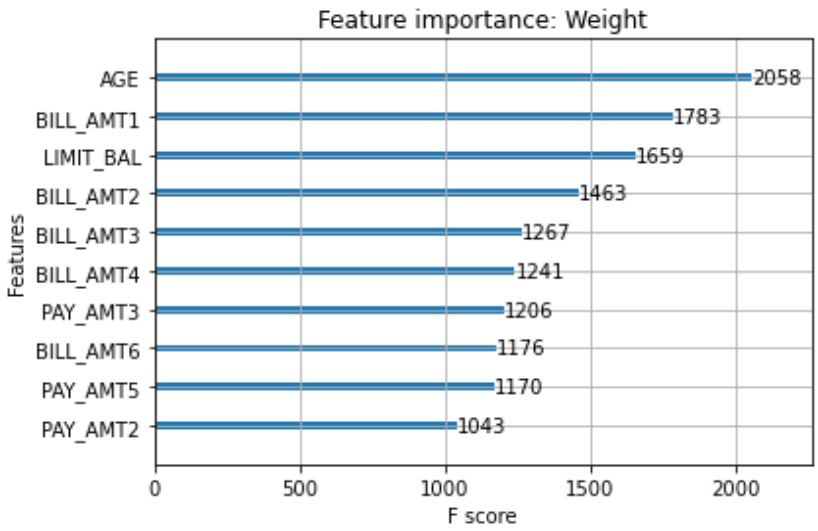

Figure 8: Top 10 Most Important Features: XGBoost Weight

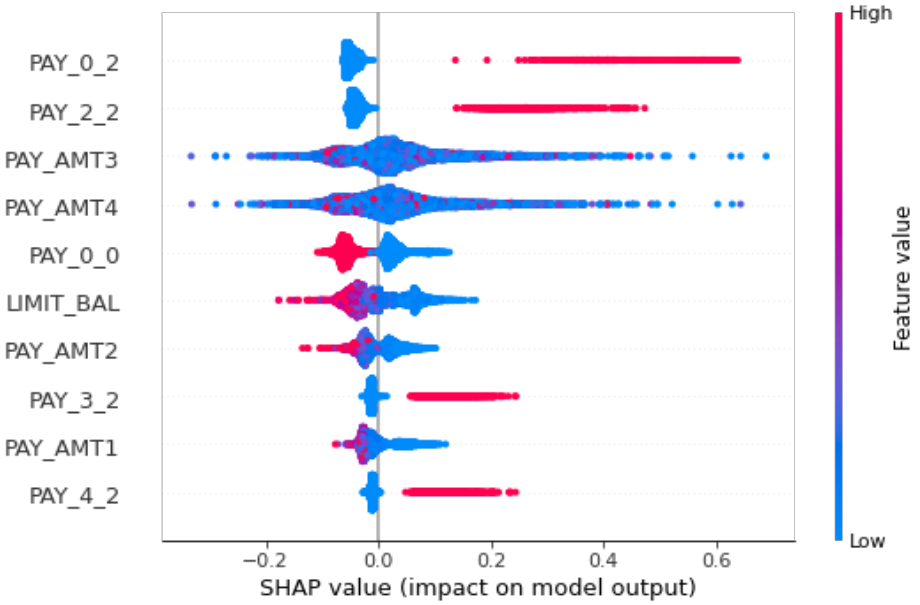

Figure 9: Top 10 Features: SHAP Local Feature Importance

As we can see in all importance metrics, the pay status variables are some of the most important, and in gain and cover, these make up all the top 10 most important features. This tracks with the EDA and with general theory that if you've been behind in the past, it is more likely you'll default. These were also variables that were selected in models in previous literature. One other thing of interest when comparing this model to the previous literature was that unlike the previous models, education status did not seem to be important in this model. While not as prominent, it is also worth noting that the balance limit variable is important according to the weight metric and the SHAP local feature importance. Overall, the feature importances seem to be robust with a good amount of overlap between measures of importance. They also fit with

general ideas of what factors are important for predicting default, which is relevant in banking when companies must be able to explain their models and have them vetted by auditors.

5. Outlook

While the final test score was significantly higher than the baseline, there is still room for improvement. Two things that could possibly improve the model are working on the imbalance and adding more data. I attempted to use the class weight parameter for all the models when parameter tuning, but it did not seem to improve the models. Some sampling or maybe SMOTE could improve it in future iterations. However, it is more likely that gathering additional data will yield the biggest improvement. Most default prediction models at banks have external datasets about other customer accounts that are helpful, such as FICO score information.

References:

[1] Yeh, I. C., & Lien, C. H. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Systems with Applications, 36(2), 2473-2480.
[2] Laborda, Juan & Ryoo, Seyong. (2021). Feature Selection in a Credit Scoring Model (Mathematics ISSN 2227-7390). 9 (7). 10.3390/math9070746.
[3] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.