Predicting Credit Card Default with XGBoost

By Zachary Kneupper

Abstract

Machine learning techniques can be an important tool for financial risk management. Performant and scalable machine learning models that are able to predict the probability of credit card default can bolster the risk management toolkits of credit scoring agencies (*e.g.*, Experian) and credit card originators (*e.g.*, Bank of America). This paper describes our application of gradient boosting tree models to predict consumer credit card default. Specifically, we used a high-performance implementation of the gradient boosting algorithm called XGBoost to predict credit card default. We trained and tested our gradient boosting model on the anonymized dataset of credit card holders used by Yeh & Lien (2009).[1] We found that the predictive power (as measured by the lift chart area ratio) of XGBoost outperformed that of the models tested by Yeh & Lien. XGBoost achieved an area ratio score of 0.56, whereas the best-performing model used by Yeh & Lien (a neural network model) achieved an area ratio score of 0.54. Additionally, our XGBoost model achieved a cross-validated ROC AUC score of 0.78.

Keywords: Risk management; Machine learning; Gradient boosting; XGBoost

Table of Contents

I. Introduction 2
II. Description of the Dataset 2
III. Data Wrangling and Exploratory Data Analysis 3
IV. Feature Engineering 7
V. Machine Learning Pipeline and Grid Search 9
VI. Results 9
Appendix A. Source Code and Jupyter Notebooks 11

[1] I-Cheng Yeh, and Che-Hui Lien. "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients." *Expert Systems with Applications* 36, no. 2 (2009): 2473-480. doi:10.1016/j.eswa.2007.12.020.

I. Introduction

Logistic regression has been a workhorse in the credit rating industry for decades. This tried-and-true method had proven effective given the data and computing resources available. In the recent past, computing power has increased considerably (per Moore's Law), and new machine learning methods have been developed. This paper focuses on one such machine learning technique—gradient boosting tree models—and its application to predicting credit card default. Specifically, we used a high-performance implementation of the gradient boosting algorithm called XGBoost. Ultimately, we found that the predictive power[2] of XGBoost outperformed that of logistic regression[3].

II. Description of the Dataset

We used an anonymized dataset of Taiwanese credit card holders from October 2005 used by Yeh & Lien (2009). This dataset was made available to the public and posted on the UC Irvine Machine Learning Repository website.[4] The dataset posted on the UC Irvine Machine Learning Repository website was in `.xls` format.

The dataset contains 30,000 observations. Each of these observations corresponds to an individual credit card holder.

A. Response Variable

The dataset contains a binary response variable y indicating whether or not an individual defaulted on their credit card payments in October 2005. For each observation, if an individual defaulted on their credit car payments in October 2005, then $y = 1$; Otherwise, $y = 0$. Among the total 30,000 individuals in the dataset, 6,636 (22.1%) defaulted on their credit card payments in October 2005.

B. Original Features

The dataset contains 23 features (explanatory variables). As per the data description given by Yeh & Lien (2009), these features include:

- Age (in years);
- Gender (where 1 = male and 2 = female);
- Marital status (where 1 = married, 2 = single, and 3 = other);

[2] Here, predictive power is measured by the area ratio score.

[3] As reported by Yeh & Lien (2009).

[4] See: http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients.

- Education (where 1 = graduate school, 2 = university, 3= high school, and 4 = other); and
- Credit limit denominated in New Taiwan Dollars (TWD).

The dataset also contains three sets of historical explanatory variables. Each of these three sets contains six features, one for each month from April 2005 to September 2005. These three sets are:

- Past payment status that month, where:[5]
  - -2 = no consumption;
  - -1 = paid in full;
  - 0 = the use of revolving credit;
  - 1 = payment delay for one month;
  - 2 = payment delay for two months;
  - …;
  - 8 = payment delay for eight months;
  - 9 = payment delay for nine months and above;
- Amount billed that month (in TWD); and
- Amount paid that month (in TWD).

III. Data Wrangling and Exploratory Data Analysis

A. Data Wrangling

1) Null Values

The dataset did not contain any null values.

2) Duplicate Records

There were 35 rows that were possible duplicates. These potential duplicates account for only 0.1 percent of the data. Most of these potential duplicates have 0 bill amount and 0 pay amount for all six months. It is entirely possible that individuals of the same age, gender, marital status, etc. did not use their credit cards at all over these six months. This would result in identical row values for individual observations that were in fact independent. In light of this, we opted not to drop these 35 possibly duplicated rows.

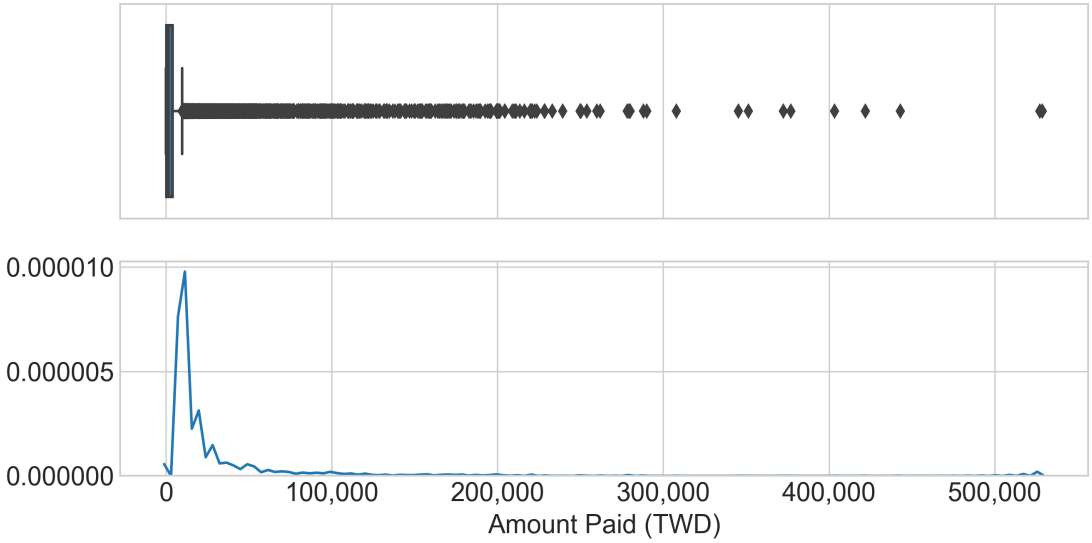
[5] This description of past payment status was taken from https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset/discussion/34608.

3) Outliers

Unlike linear classification models, which often require pre-processing of data to transform and remove large outliers, tree-based models (like XGBoost) do not require any special treatment for outliers. In our data wrangling process, the reason that we check for outliers is to identify values that might be erroneous measurements.

We found that all of the features that are measured in currency (namely, amounts billed, amounts paid, and credit limit) had large skew and kurtosis, having many large outliers. To illustrate, Exhibit III-1 displays the distribution of amounts paid in April 2005.

Exhibit III-1: Box Plot & Kernel Density Plot of Amounts Paid in April 2005



The amounts paid in April 2005 ranged from TWD 0 to TWD 528,666 (approximately USD 16,450).[6] However, for over 90% of observations, the amount paid in April 2005 was less than TWD 10,000 (approximately USD 310). The distribution was highly skewed to the right. We found that this sort of extreme right skew and large kurtosis was typical of all of the amounts billed, amounts paid, and credit limit. However, we did not identify any outliers that were unambiguously erroneous values.

B. Exploratory Data Analysis

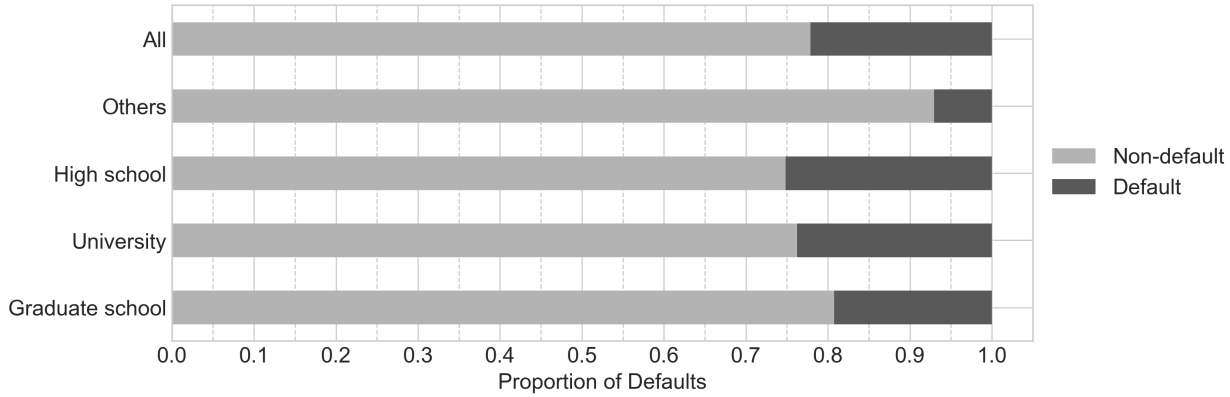
Exploratory Data Analysis ("EDA") is an important preliminary step in machine learning projects. Data visualization can help reveal patterns and relationships in data might otherwise be overlooked.

[6] Based on 2015 annual average exchange rate of TWD 32.131 = USD 1. Source: The Federal Reserve Bank of St. Louis (https://fred.stlouisfed.org/series/AEXTAUS).

We used histograms, box plots, and kernel density plots (like the one shown in Exhibit III-1) for univariate EDA on the continuous features in our dataset.

To explore how the proportion of defaults differed between groups based on categorical variables, such as level of education, we created graphs like the one shown below.

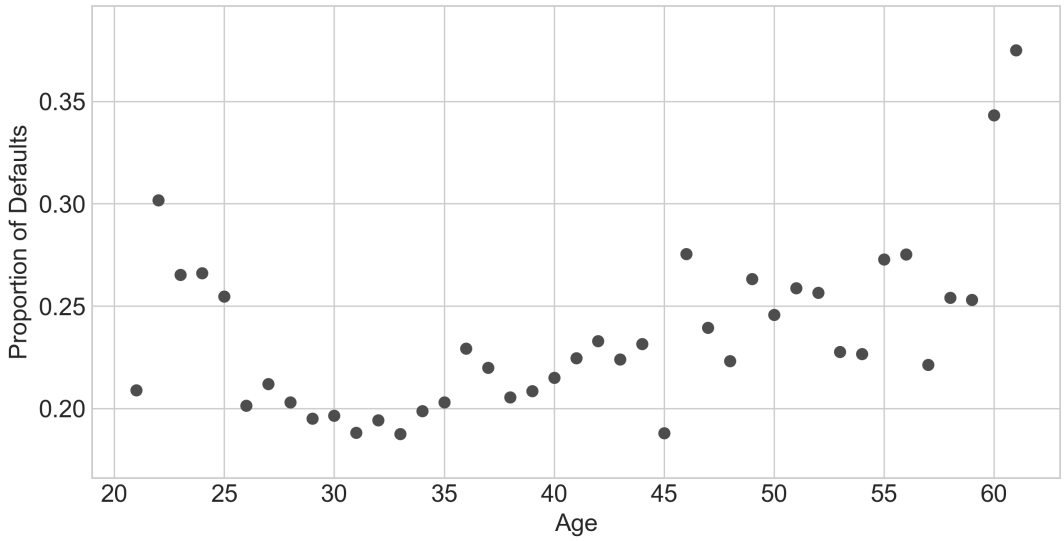
Exhibit III-2: Proportion of Defaults by Education



We also explored how the proportion of defaults changed across values of continuous features by binning and calculating proportions within each bin. When using this data visualization technique, we decided to drop bins with too few observations. If the number of observations in a given bin was below some threshold (say 50), we would not plot the proportion of defaults for that bin.

For example, we visualized how the proportion of defaults changed with age, where we by binning by age (in years).

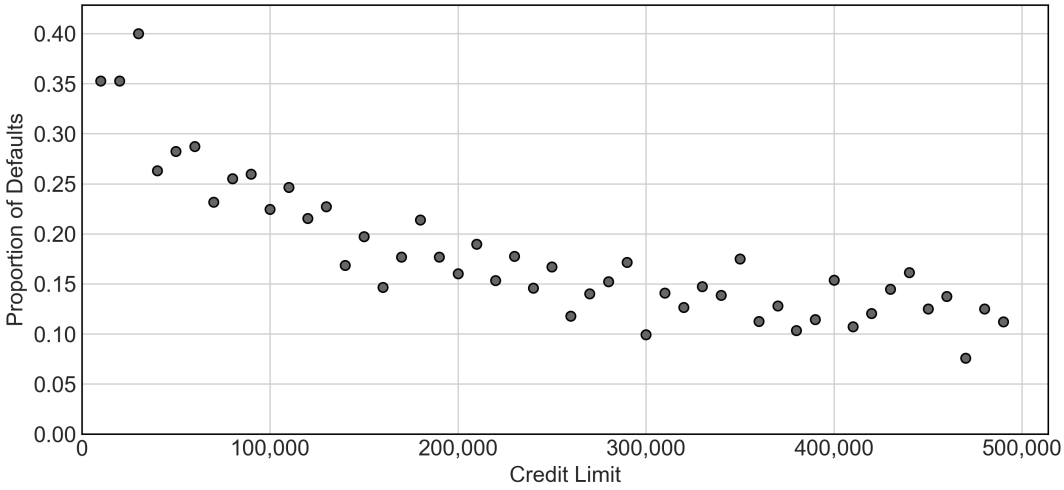
Exhibit III-3: Proportion of Defaults by Age



From Exhibit III-3, it appears that the probability of default changes with age, but it does not change *linearly* with age. Nonlinear relationships like this may not be easily learned by machine learning models like logistic regression, but tree-based models like XGBoost can handle these sorts of nonlinearities.

In Exhibit III-4, we visualized how the proportion of defaults changed with credit limit. When binning credit limit, we used a bin size of TWD 10,000 (approximately USD 310).

Exhibit III-4: Proportion of Defaults vs. Credit Limit

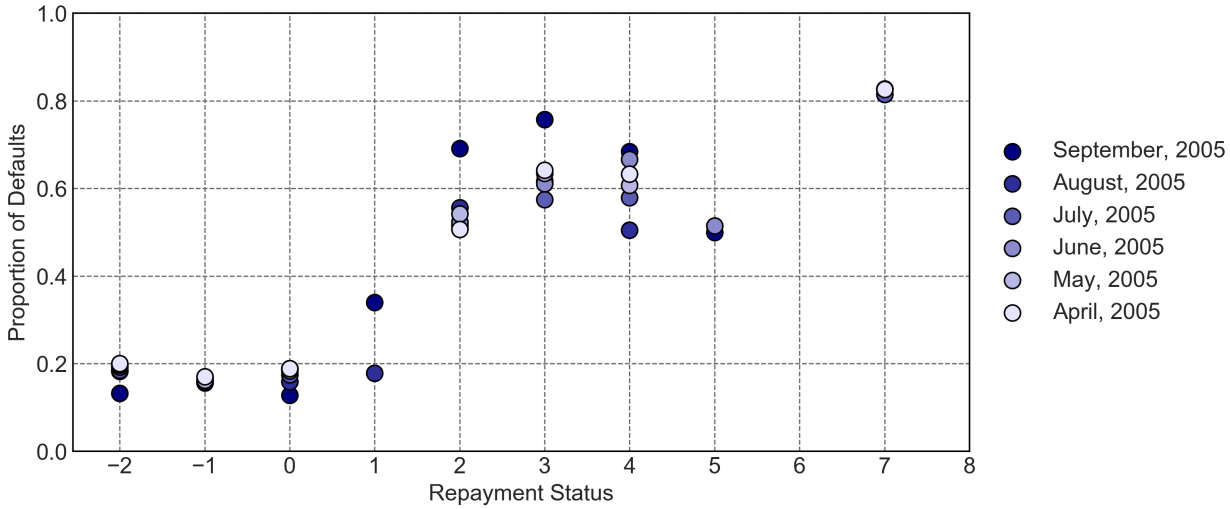


The probability of default is negatively correlated with credit limit. Generally, customers who are deemed to be less credit-worthy (less likely to pay back their debts) are not eligible for large credit limits, while customers deemed more credit-worthy are granted larger credit limits. Credit card companies appear to be doing a good job at due diligence.

We created similar data visualizations for the other continuous features.

In Exhibit III-5, we plot proportion of defaults by repayment status for each month. You may recall, the data contain a repayment status feature for each of the six months from April to September 2005.

Exhibit III-5: Proportion of Defaults vs. Repayment Status



Repayment status category key: -2 = no consumption; -1 = paid in full; 0 = the use of revolving credit; 1 = payment delay for one month; 2 = payment delay for two months; … ; 8 = payment delay for eight months; 9 = payment delay for nine months and above.

It appears that if there is no delay in repayment, then the probability of default is relatively low. If repayment is delay for two months or more, the probability of default is much higher.

IV. Feature Engineering

We performed two types of feature engineering processes: (1) Calculating meaningful ratios of certain continuous features; and (2) One-hot encoding categorical features. We describe each of these processes below.

A. Feature Engineering: Ratios

As mentioned before, the dataset contains amounts billed and amounts paid for each month from April to September, 2005. We use these features, along with credit limit, to calculate two sets of ratios. The first is the ratio of $\left(\frac{Amount\ Billed}{Credit\ Limit}\right)$. The second is the ratio of $\left(\frac{(Amount\ Billed) - (Amount\ Paid)}{Credit\ Limit}\right)$.

Our economic intuition led us to believe that the probability of default would increase as either of these two ratios increases, all else equal. This intuition seems to be born out by the data. In the following two exhibits, we visualized how the proportion of defaults changes with each ratio.

Exhibit IV-1: Proportion of Defaults vs.
the Ratio of (Amount Billed / Credit Limit)

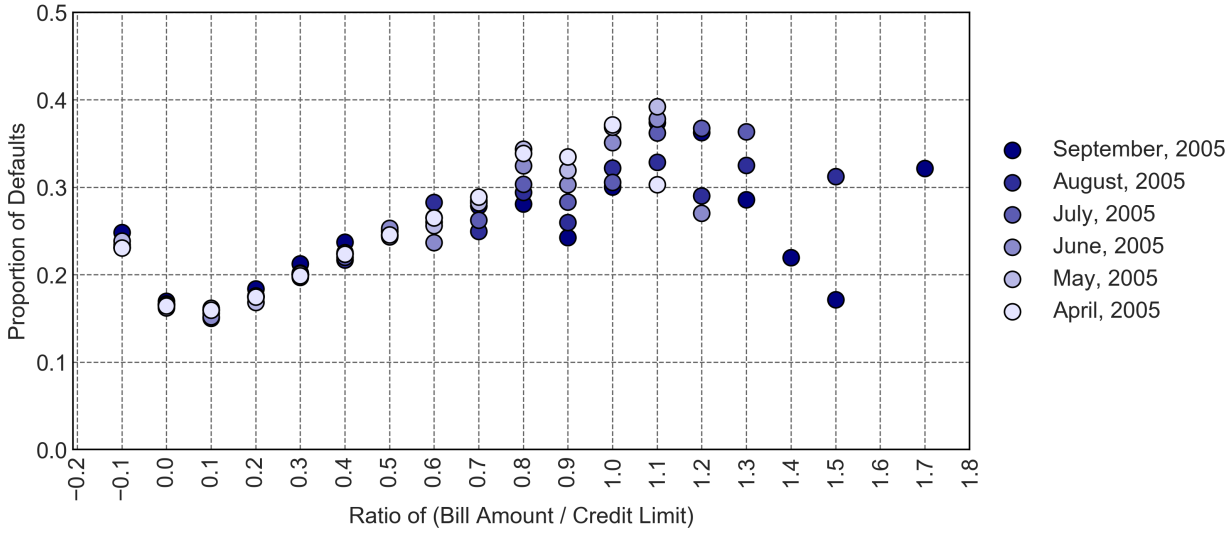
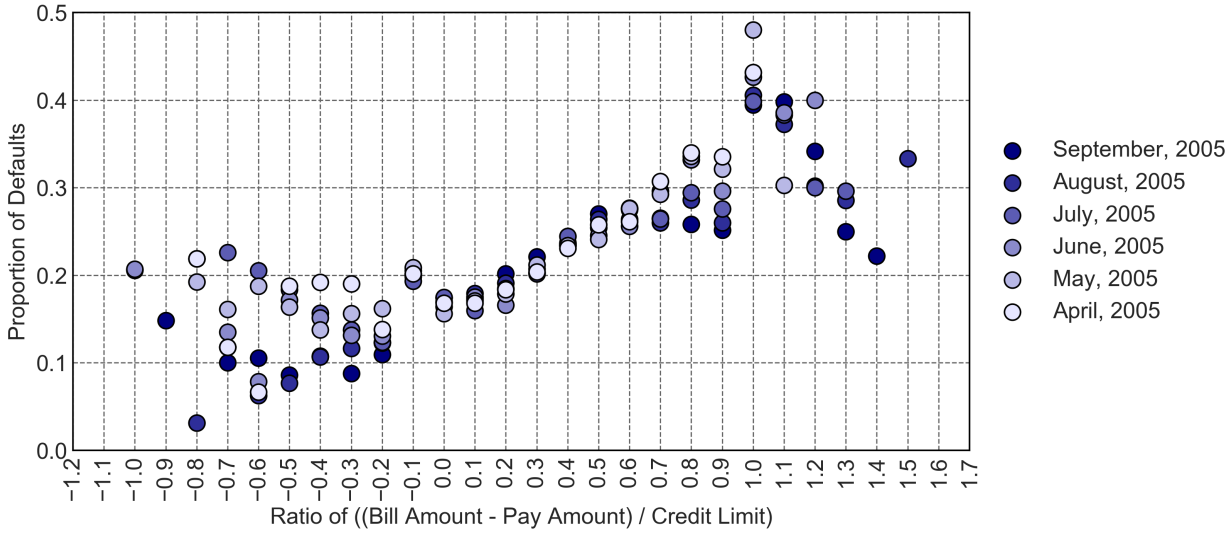


Exhibit IV-2: Proportion of Defaults vs.
the Ratio of ((Amount Billed – Amount Paid) / Credit Limit)



B. Feature Engineering: One-Hot Encoding

The dataset contains nine categorical features: gender, marital status, education, and six features for past payment status (one for each month from April 2005 to September 2005). These categorical features had been encoded with integer values in the original dataset. We transformed these nine categorical feature columns into 65 one-hot encoded columns.

Using one-hot encoding expanded the total number of features from 36 to 92. In the preprocessing phase of our machine learning pipeline, we used a simple dimensionality reduction technique to reign in number of features.

V. Machine Learning Pipeline and Grid Search

We used cross-validated grid search to identify the optimal preprocessing techniques and optimal hyperparameters. The following exhibit shows the hyperparameter space that we searched.

Exhibit IV-1: XGBoost Hyperparameters Grid Search

Parameters	Values
n_estimators	1000
learning_rate	0.01, 0.1
gamma	0.01, 0.1
max_depth	3, 4
min_child_weight	1, 3
subsample	0.8
reg_lambda	0.1, 1.0

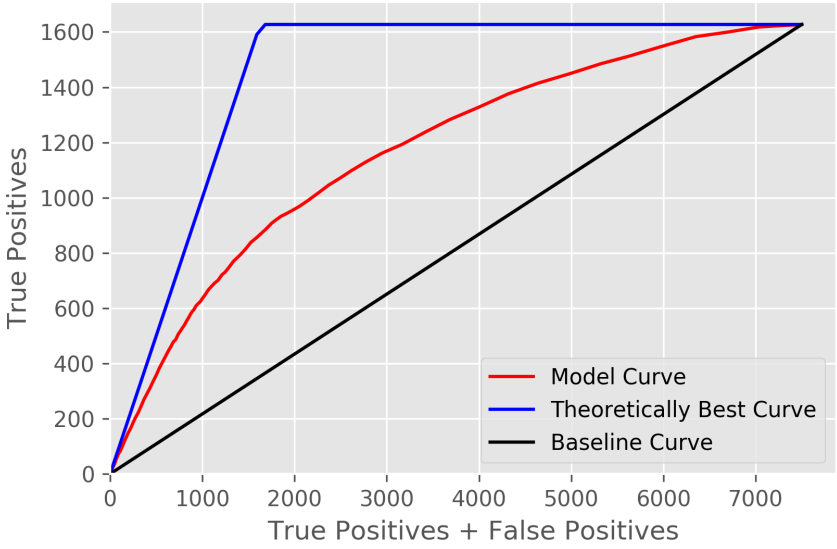
VI. Results

Area under the receiver operating characteristic curve ("ROC AUC") is often used to quantifying the predictive power of machine learning models for binary classification. The ROC AUC score can take on values from 0 to 1. An ROC AUC score of 0.5 implies that a model is not predictive. An ROC AUC score of 1 implies that a model is very predictive. Our XGBoost model achieved a cross-validated ROC AUC score of 0.78.

Yeh & Lien (2009) used the lift chart area ratio ("area ratio score"), rather than the ROC AUC score, to quantifying the predictive power of the models used in their analysis. We calculated the area ratio score for our XGBoost model so as to compare it against the scores achieved by Yeh & Lien (2009).

Exhibit VI-1 illustrates the lift chart for our XGBoost model.

Exhibit VI-1: Lift Chart – Test Set



In any lift chart, the x-axis represents the sum of true positives and false positives, and the y-axis represents the number of true positives. The area ratio score is calculated as follows:

$$\text{Area Ratio} = \frac{(\text{Area between model curve and baseline curve})}{(\text{Area between theoretically best curve and baseline curve})}$$

Our XGBoost model achieved an area ratio score of 0.56 on the test set. Exhibit VI-2 compares the performance of our XGBoost model against the performance of models analyzed in Yeh & Lien (2009).

Exhibit VI-2: Comparison of XGBoost against Yeh & Lien's Models

Method	Error Rate		Lift Curve Area Ratio	
	Training	Validation	Training	Validation
Results of Yeh & Lien (2009)				
K-nearest neighbor	0.18	0.16	0.68	0.45
Logistic regression	0.20	0.18	0.41	0.44
Discriminant analysis	0.29	0.26	0.40	0.43
Naive Bayesian	0.21	0.21	0.47	0.53
Neural networks	0.19	0.17	0.55	0.54
Classification trees	0.18	0.17	0.48	0.536
My Results				
XGBoost	0.17	0.18	0.659	0.560

Our XGBoost achieved a higher area ratio score on the test-set than that achieved by the models analyzed in Yeh & Lien (2009). XGBoost achieved an area ratio score of 0.56, whereas the best-performing model used by Yeh & Lien (a neural network model) achieved an area ratio score of 0.54.

These results indicate that gradient boosting models might outperform logistic regression when it comes to predicting credit card default. Despite that, gradient boosting models are less parsimonious, less interpretable, and less computational efficient. If a business (*e.g.*, a credit scoring agency) is required to explain and defend their model before regulators, the business may not be in a position to trade greater predictive power for less interpretability. If interpretability is needed, logistic regression may be preferable. Otherwise, XGBoost may be worth considering. Model choice must be informed by one's business goals, regulatory constraints, and computational resources.

Appendix A. Source Code and Jupyter Notebooks

All of the Python code and Jupyter notebooks used in this project can be found on GitHub: https://github.com/zkneupper/Default-Prediction-Capstone.