Analytics and Modeling Capstone Project

Oscar Hernandez

Northwestern University

June 9th, 2019

Model #101: Credit Card Default Model

Model Development Guide

1. Introduction

This model development guide was focused on developing a credit card default model that can accurately classify a default payment occurring based on a variety of information pertaining to a client such as their age, marriage status and payment history. The development of the final model included data preparation, exploratory data analysis, feature engineering and evaluating several model classes for performance.

The motivation behind developing such a model is driven by the increasing need of creating effective and robust risk management solutions that financial institutions can utilize. Financial institutions play an important role in the well-being of the global economy. As such, their ability to accurately identify the risk associated with current and new clients is a top priority. Specifically, estimating the probability of a default (e.g. not paying back a private loan) given various attributes of a client (e.g. time with company, income level) has been the focus of substantial academic and industry research from various organizations.

This model development guide will provide the steps taken to understand which attributes of a client are important in determining if they will have a default payment and then using those attributes to develop a classification model.

2. The Data

The data used for this model development guide come from a Taiwanese bank that issues both credit and cash cards. Specifically, the data represents only the credit card clients of the bank. The data have 30,000 observations and 24 variables, which include the response variable *DEFAULT*. The dependent variable represents the existence (1 = Yes, 0 = No) of a default payment by the credit card client. As such, it is a binary response variable. Predictor variables include various descriptive information about the clients such as their age, education level and marriage status. Furthermore, there are several variables that include information about the client's payment history.

Table 1 provides a data dictionary of the default variables. Certain variables were condensed into one row since they represent various points in time, but all relate to the same activity such as making a payment. For example, there are multiple *PAY_X* variables such as *PAY_0*, *PAY_2*, *PAY_3* and so forth. Moreover, all the values of the default variables are integers. Also, the use of "Discrete" in the Data Type column in Table 1 denotes that the variable contains a limited amount of values (i.e. only three values for *MARRIAGE*). It does not mean it was provided as a categorical variable in the data set.

Table 1: Data Dictionary of Default Variables

Default Variable	Data Type	Data Value Description
LIMIT_BAL	Continuous	Amount of the given credit (NT dollar)
SEX	Binary	Gender (1 = Male, 2 = Female)
EDUCATION	Discrete	Education (1 = graduate school; 2 = university; 3 = high school; 4 = others)
MARRIAGE	Discrete	Marital Status (1 = married; 2 = single; 3 = others)
AGE	Continuous	Age (years old)
PAY_0 – PAY_6	Discrete	History of repayment status (-1 = pay duly; 1 = payment delay one month….9 = payment delay nine months)
BILL_AMT1 – BILL_AMT6	Continuous	Amount of bill statement (NT dollar)
PAY_AMT1 – PAY_AMT6	Continuous	Amount of previous payment (NT dollar)
DEFAULT	Binary	Default payment (1 = Yes, 0 = No)

The *PAY_X*, *BIL_AMTX* and *PAY_AMT6* variables represent six months of repayment status, amount on bill statement and amount of payment made, respectively. The date range for this data set is April 2005 through September 2005. For example, *PAY_AMT1* represents the amount of paid in September 2005 and *PAY_AMT6* represents the amount paid in April 2005.

Furthermore, the *PAY_AMTX* variable is a lagged variable which means that the time period denoted by its variable name doesn't match up directly with the time period in the *BILL_AMTX* name. For example, the values in *BILL_AMT6* represent the credit card bill amount in April 2005. The values in *PAY_AMT5* show the payment made in May 2005 which would demonstrate how much of the credit card bill amount from April 2005 a customer paid. The other variables in the data do not require any explanation.

2.1 Data Quality Check

The initial step in building the credit card default model is to complete a data quality check on all 24 variables and 30,000 observations. This step will identify any initial issues that need to be rectified or at least be aware of prior to any feature engineering, data analysis and model training.

The first error in the data set is that *PAY_0* needs to be changed into *PAY_1* so that it follows the same naming convention of the other billing and payment variables. There are no other errors with respect to the variable names in the data.

As displayed in Table 1, the data include a combination of discrete, binary and continuous variables. These variables will be used to feature engineer new ones which will subsequently be part of a pool of predictor variables available to train each of the credit card default models. Therefore, the next part of the data quality check is to review the actual values within each variable to determine if there are any errors. An error is classified as a value that shouldn't exist based on the correct range of values a variable should hold. (Please note that the term "value" is referencing the categories within a variable but will be referred to as "value(s)" initially since the variables have not been formally discretized yet).

Table 2 displays the summary statistics for the entire data set. The first takeaway is there are no missing observations which means the development guide will not cover a missing observations imputation strategy. Strictly looking at the range of values, another takeaway is that the *LIMIT_BAL, SEX, AGE* and *DEFAULT* variables appear to hold values that are correct. For example, there should not be any *AGE* values corresponding to children such as "15" or any *DEFAULT* values greater than "1." *EDUCATION* and *MARRIAGE* hold values that are inconsistent with the data dictionary. There should not be any "0" values for *MARRIAGE* or *EDUCATION.* Furthermore, there should not be any values greater than "4" for *EDUCATION*. The steps taken to correct these values will be discussed in the next section.

Reviewing the range of values does not necessarily mean there are no errors within the data. Creating tables of the discrete variables is necessary to see the counts of each value. For example, a table output of *EDUCATION* could show it contains values such as "3.2" which would be considered incorrect. However, just by looking at the range from the Table 2, this error would not be caught. Although not displayed in this development guide, this step was taken for all the discrete variables. The following is a summary of the results:

- *LIMIT_BAL, SEX, AGE, DEFAULT* don't have any errors
- *EDUCATION* has values of "0, 5 and 6" which are not part of the data dictionary
- *MARRIAGE* has values of "0" which is not part of data dictionary
- *PAY_0 – PAY_4* contain values of "-2" and "0" which are not part of data dictionary
- *PAY_5* and *PAY_6* has the same condition as the previous *PAY_X* variables and are missing "1" values

Furthermore, the *PAY_X* variables do not follow the data dictionary guidelines. The smallest value should be "-1" but Table 2 shows that the smallest value is "-2." The *BILL_AMTX* variables do not have any erroneous values. Although the minimum values for the *BILL_AMTX* variables appear idiosyncratic (generally speaking, credit card balances are not negative), they are not to be considered an error because a client could pay more than what was owed. Lastly, PAY_*AMTX* does not have any errors since the minimum values are not negative. Negative values are not allowed since that would represent a negative payment which is not possible.

Table 2: Summary Statistics for Default Variables

Statistic	N	Mean	St. Dev.	Min	P(25)	Median	P(75)	Max
LIMIT_BAL	30,000	167,484.30	129,747.70	10,000	50,000	140,000	240,000	1,000,000
SEX	30,000	1.60	0.49	1	1	2	2	2
EDUCATION	30,000	1.85	0.79	0	1	2	2	6
MARRIAGE	30,000	1.55	0.52	0	1	2	2	3
AGE	30,000	35.49	9.22	21	28	34	41	79
PAY_0	30,000	-0.02	1.12	-2	-1	0	0	8
PAY_2	30,000	-0.13	1.20	-2	-1	0	0	8
PAY_3	30,000	-0.17	1.20	-2	-1	0	0	8
PAY_4	30,000	-0.22	1.17	-2	-1	0	0	8
PAY_5	30,000	-0.27	1.13	-2	-1	0	0	8
PAY_6	30,000	-0.29	1.15	-2	-1	0	0	8
BILL_AMT1	30,000	51,223.33	73,635.86	-165,580	3,558.8	22,381.5	67,091	964,511
BILL_AMT2	30,000	49,179.08	71,173.77	-69,777	2,984.8	21,200	64,006.2	983,931
BILL_AMT3	30,000	47,013.15	69,349.39	-157,264	2,666.2	20,088.5	60,164.8	1,664,089
BILL_AMT4	30,000	43,262.95	64,332.86	-170,000	2,326.8	19,052	54,506	891,586
BILL_AMT5	30,000	40,311.40	60,797.16	-81,334	1,763	18,104.5	50,190.5	927,171
BILL_AMT6	30,000	38,871.76	59,554.11	-339,603	1,256	17,071	49,198.2	961,664
PAY_AMT1	30,000	5,663.58	16,563.28	0	1,000	2,100	5,006	873,552
PAY_AMT2	30,000	5,921.16	23,040.87	0	833	2,009	5,000	1,684,259
PAY_AMT3	30,000	5,225.68	17,606.96	0	390	1,800	4,505	896,040
PAY_AMT4	30,000	4,826.08	15,666.16	0	296	1,500	4,013.2	621,000
PAY_AMT5	30,000	4,799.39	15,278.31	0	252.5	1,500	4,031.5	426,529
PAY_AMT6	30,000	5,215.50	17,777.47	0	117.8	1,500	4,000	528,666
DEFAULT	30,000	0.22	0.42	0	0	0	0	1

After completing the data quality check, the next step is to fix the errors using a variety of methods. One final item to note prior to the next section is that out of the 30,000 observations, there are 23,364 non-default payment clients (77.88%) and 6,636 default payment clients (22.12%).

2.2 Data Quality Check – Error Imputations

EDUCATION has observations with “0, 5 and 6” as values which are not part of the data dictionary. Table 3 displays the frequency of those observations in the data. The decision was made to place those observations into the “4” category which according to the data dictionary represents “others.”

Table 3: Frequency Table of EDUCATION

Category	Frequency
0	14
1	10,585
2	14,030
3	4,917
4	123
5	280
6	51

MARRIAGE has observations with “0” values and the decision was made to place those observations into the “3” category which represents “others.” Table 4 displays the frequency of those observations in the data.

Table 4: Frequency Table of MARRIAGE

Category	Frequency
0	54
1	13,659
2	15,964
3	323

Most of the observations for the *PAY_X* variables have “0” as values. The data dictionary does not provide a definition for what that value means. An initial thought is that it represents a client is not behind on their payment which means it was incorrectly labelled as a “0” instead of a “-1.” However, after reviewing a sample of the data, it was determined that this is not the case because there were observations that included clients with “0” values across multiple *PAY_X* variables but were behind on payments. The decision was made to place those observations into the “1” category. Furthermore, the decision was made to place observations with “-2” values into the “-1” category.

There are no further imputations that are necessary for the data. The next part of the model development guide outlines the feature engineering that was completed. All the data will be utilized as part of this step. After completing the data quality check, the data was split into training, testing and validation data sets. Table 5 displays a breakdown of the training, testing and validation data sets. Only the training set will be used as part of the feature engineering step to minimize introducing data leakage issues into our models.

Table 5: Training, Testing and Validation Data Sets

Category	Frequency	% of All Data
Training	15,180	50.60%
Testing	7,323	24.41%
Validation	7,497	24.99%

3. Feature Engineering

Although the data provide multiple variables that can be used to train a credit card default model, it is not required that those variables are used. Feature engineering is a useful step of model development that is completed to extract new variables from the existing ones provided. These new variables can be used to increase the predictive accuracy of a model.

One common feature engineering method is the discretization of continuous variables. *AGE* was discretized using the Weight of Evidence (WOE) optimal binning algorithm. Specifically, the R “woe.tree.binning” function was used to discretize *AGE* into five bins. Table 6 displays the results of using that optimal binning algorithm. It shows the number of observations that are a part of the new *AGE* bins along with the proportion of those observations associated with either *DEFAULT* class as a percentage of the total observations. Moreover, *SEX*, *EDUCATION* and *MARRIAGE* were converted to categorical variables since they take on discrete values.

Table 6: AGE Binning Results

Category	Frequency	0 - Dist. %	1 - Dist. %	WOE
<= 24	1357	8.3%	11.1%	-28.6
(24, 25]	607	3.9%	4.4%	-11.1
(25, 34]	5965	40.6%	34.9%	15.2
(34, 48]	5665	37.2%	37.8%	-1.8
>= 49	1586	10%	11.9%	-17

Table 7 highlights the new variables that were created during the feature engineering step. However, there may be additional ones created as a result of the exploratory data analysis (EDA). There will be a table with a pool of predictor variables provided after the EDA is complete.

Table 7: Feature Engineered Variables

Variable	Definition
Age	Discretized version of AGE using WOE optimal binning algorithm
Avg_Bill_Amt	Represents the average of BILL_AMT1 through BILL_AMT6
Avg_Pmt_Amt	Represents the average of PAY_AMT1 through PAY_AMT6
Pmt_Ratio_2 – Pmt_Ratio_6	(PAY_AMTX/BILL_AMTX+1) *100; starting at X=1
Avg_Pmt_Ratio	Represents the average of Pmt_Ratio_2 through Pmt_Ratio_6
Util_1 – Util_6	(BILL_AMTX/LIMIT_BAL) *100; starting at X=1
Avg_Util	Represents the average of Util_1 through Util_6
Bal_Growth_6mo	Binary variable; difference between BILL_AMT1 and BILL_AMT6
Util_Growth_6mo	Represents difference between Util_1 and Util_6
Max_Bill_Amt	Max value of BILL_AMT1 through BILL_AMT6
Max_Pmt_Amt	Max value of PAY_AMT1 through PAY_AMT6
Max_DLQ	Max value of PAY_1 through PAY_6
Pmt_Ratio_Growth_6mo	Represents difference between Pmt_Ratio_2 and Pmt_Ratio_6
Pmt_Status_Change_6mo	Represents difference between PAY_1 and PAY_6

Pmt_Ratio_X represents the proportion of each bill a client has been paying. These variables were created using the definition provided in Table 7. Since payments are made for the prior month's bill, it was only possible to create five versions. There is no data available to create a version representing how much of the September 2005 bill was paid. After making the initial calculation of these variables, some of the resulting values needed to be adjusted so they can be utilized by each model more effectively. The following provides a more detailed explanation of how that was completed:

- If *Pmt_Ratio_X* and *PAY_AMTX* was equal to 0 and *BILL_AMTX* was negative, *Pmt_Ratio_X* was changed to "101"; "101" shows a client had a credit balance and didn't make a payment
- If *PAY_AMTX* was equal to 0 and *BILL_AMTX* was equal to 0, *Pmt_Ratio_X* was changed to "100"; "100" shows a client didn't have a balance and didn't make a payment
- If *Pmt_Ratio_X* was negative, it was changed to "101"; "101" shows a client had a credit balance and did make a payment
- If *PAY_AMTX* was positive and *BILL_AMTX* was equal to 0, *Pmt_Ratio_X* was changed to "101"; "101" means customer had a "0" balance and made a payment
- If *Pmt_Ratio_X* was greater than "101" it was changed to "101"; these values were considered outliers and thus were winsorized

Util_X represents the proportion of a client's credit card limit that was utilized each month. These variables were created using the definition provided in Table 7. After making the initial calculation of these variables, some of the resulting values needed to be adjusted so they could be utilized by each model more effectively. The following provides a more detailed explanation of how that was completed:

- If *Util_X* was negative it was changed to “0”; represents customer did not utilize any of their credit card limit
- If *Util_X* was greater than “100” it was changed to “100”; customer used more than allowed by their credit card limit; decision was made to winsorize these values to show entire credit card limit was used

Furthermore, *Bal_Growth_6mo* was created as a dummy variable where “1” represents that a client grew their balance from April 2005 through September 2005 for all possible scenarios. For example, if a client had a smaller credit balance in September 2005 than they did in April 2005, they were given a “1” because it shows they added debt even though they still hold a credit balance. All the other variables listed in Table 7 do not require additional explanation.

4. Traditional Exploratory Data Analysis

After completing the feature engineering step, it’s necessary to conduct an exploratory data analysis of the training set. Table 8 displays a statistical summary of the initial pool of continuous variables. The initial pool of categorical variables cannot be displayed in the same table.

Currently, there are 28 continuous and categorical predictor variables in the pool. These predictor variables include some of the ones that were provided with the original data set along with ones that were created through the feature engineering step. It’s important to note the variables in Table 8 are continuous because the initial data set only contained such data types (i.e. contained discrete types but were not in categorical form) and they were not discretized as part of the feature engineering step. However, that does not mean that the variables will stay as a continuous data type. EDA will be conducted to provide helpful insight to decide which variables may need to be discretized. The engineered features will also be candidates for subsequent feature engineering.

Therefore, Table 8 can be thought of as a data summary of the initial pool of continuous predictor variables that are subject to change prior to being used to train a model. The EDA will incorporate the continuous and/or categorical form of the predictor variables.

Table 8: Summary Statistics for EDA Variables

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
LIMIT_BAL	15,180	168,064.70	130,547.60	10,000	50,000	140,000	240,000	800,000
AGE	15,180	35.48	9.21	21	28	34	41	75
Avg_Bill_Amt	15,180	44,934.91	63,150.01	-56,043	4,789.1	21,198.4	56,880.5	592,432
Avg_Pmt_Amt	15,180	5,255.11	10,150.14	0.00	1,111.75	2,389.42	5,554.42	627,344.30
Pmt_Ratio_2	15,180	40.24	44.03	0	4.5	10.1	100	101
Pmt_Ratio_3	15,180	41.09	44.40	0	4.5	10.3	100	101
Pmt_Ratio_4	15,180	40.16	44.79	0	3.8	8.4	100	101
Pmt_Ratio_5	15,180	40.14	45.06	0	3.6	7.6	100	101
Pmt_Ratio_6	15,180	42.74	45.52	0	3.8	8.9	100	101
Avg_Pmt_Ratio	15,180	40.87	40.10	0	4.8	18.6	82.6	101
Util_1	15,180	41.50	38.70	0	2.19	31.65	83.02	100
Util_2	15,180	40.46	38.42	0	1.86	29.68	81.10	100
Util_3	15,180	38.64	37.47	0	1.62	27.52	75.32	100
Util_4	15,180	35.54	35.49	0	1.5	24	66.5	100
Util_5	15,180	32.95	33.90	0	1.1	21.1	60.3	100
Util_6	15,180	31.62	33.61	0	0.8	18.5	58.2	100
Avg_Util	15,180	36.79	33.94	0	3.14	28.63	68.43	100
Util_Growth_6mo	15,180	9.87	27.36	-100.00	-2.79	0.52	17.50	100.00
Max_Bill_Amt	15,180	60,425.45	77,746.88	-2,900	10,050.8	31,587.5	79,119.5	823,540
Max_Pmt_Amt	15,180	15,620.73	35,279.24	0	2,195.8	5,000	12,200.8	1,215,471
Max_DLQ	15,180	1.00	1.13	-1	1	1	2	8
Pmt_Ratio_Growth_6mo	15,180	-2.51	36.12	-101	-1	0	1.8	101
Pmt_Status_Change_6mo	15,180	0.14	1.06	-7	0	0	0	6

The goal of the credit card default model is to accurately predict which clients will have a default payment. Therefore, it is necessary to understand which variables are useful in classifying a default payment. For example, it could be that knowing the age of the client is the only variable necessary to correctly classify a default payment. Also, an exploration of predictor variables against each other is not of real value to this analysis. The focus of this analysis will be to better understand the relationship with various predictor variables and *DEFAULT*.

After reviewing Table 8, the decision was made to apply the "woe.tree.binning" function to *LIMIT*_BAL in order to discretize the variable. This makes sense because *LIMIT_BAL* values appear to be categories rather than representative of a continuous variable. Table 9 displays the results of using that optimal binning algorithm. An initial thought is that knowing which bin of *LIMIT_BAL* a client is a part of will assist with accurately classifying if they will have a default payment.

Table 9: LIMIT_BAL Binning Results

Category	Frequency	0 - Dist. %	1 - Dist. %	WOE
<= 30,000	2,020	10.8%	22%	-71.1
(30,000, 140,000]	5,747	36.2%	43.5%	-18.2
>= 150,000	7,413	53%	34.6%	42.7

Furthermore, it's important to compare the continuous predictor variables against the response variable, *DEFAULT*. These plots will point out if there is a segment of values in the continuous predictor variables that are more strongly associated with either class of *DEFAULT*. Figure 1 plots the *Avg_Bill_Amt* variable against *DEFAULT* with outliers excluded. The plot does not show a discernable segment of *Avg_Bill_Amt* that is mainly or only associated with either class of *DEFAULT*. There is a large amount of overlap which provides preliminary evidence that this may not be a good predictor.

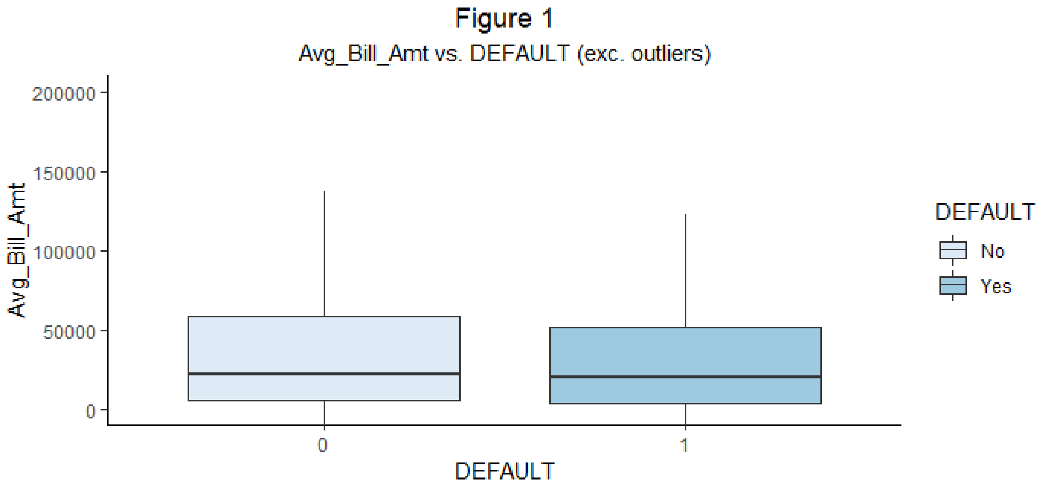


Figure 2 is a boxplot of *Avg_Util* and *DEFAULT* which resembles many of the same characteristics of Figure 1. There doesn't appear to be a discernable segment of *Avg_Util* that may be helpful in classifying a *DEFAULT*. However, there does appear to be a noticeable difference in the median values of *Avg_Util* across both classes. It seems that default payments are associated with clients that utilize more of their credit limit which seems valid.

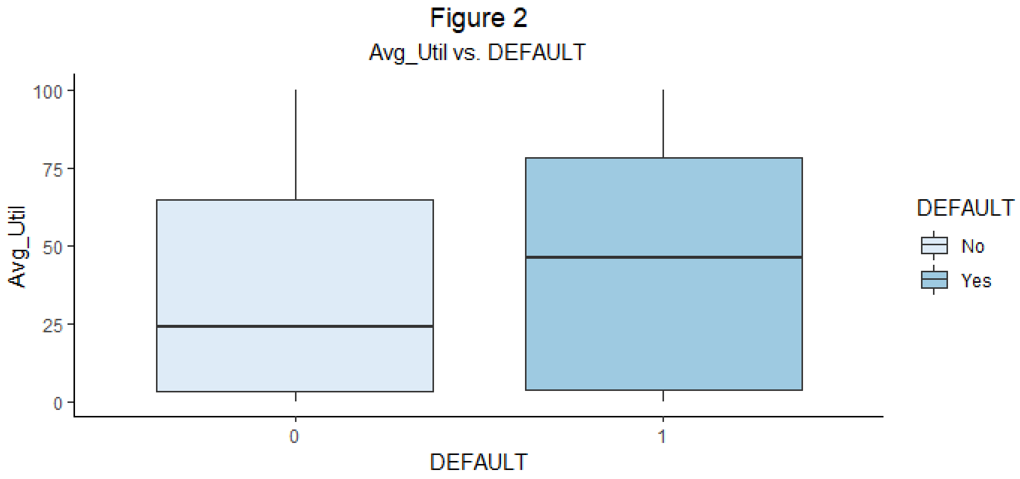


Figure 3 is a boxplot of *Avg_Pmt_Ratio* and *DEFAULT* which shows that the median average payment ratio for non-default payment class is noticeably higher compared to the default payment class. Also, the top quartile for the default payment class is approximately 65 which suggests that a client with an average payment ratio higher than 65 is more likely to belong to the non-default payment class.

Figure 3

Avg_Pmt_Ratio vs. DEFAULT

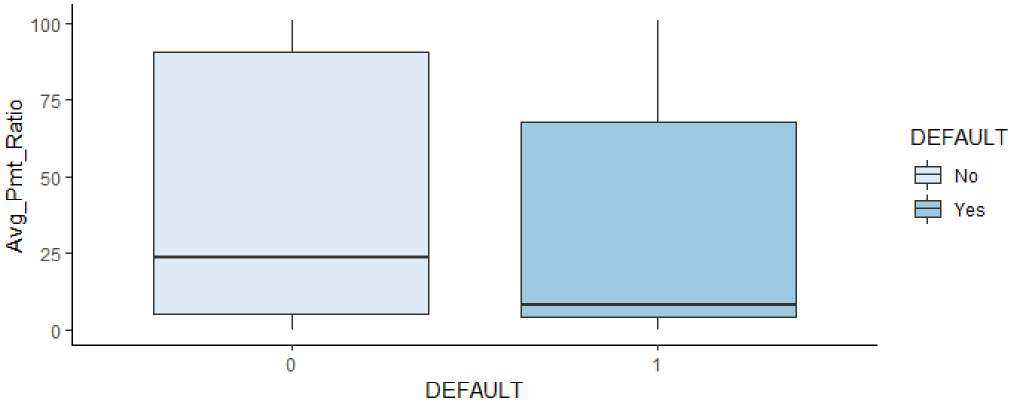


Figure 4 displays *Max_Bill_Amt* segmented by the *Age* variable. This multivariate plot provides additional information regarding the relationship between multiple variables including the response variable, *DEFAULT*. Much like the previous plots, there is some significant overlap in the distribution of the continuous variable being plotted. However, the plot suggests that clients who are over 25 years old and had a maximum bill amount greater than $50,000 are more likely to fall into the default payment class.

Figure 4

Max_Bill_Amt vs. Age

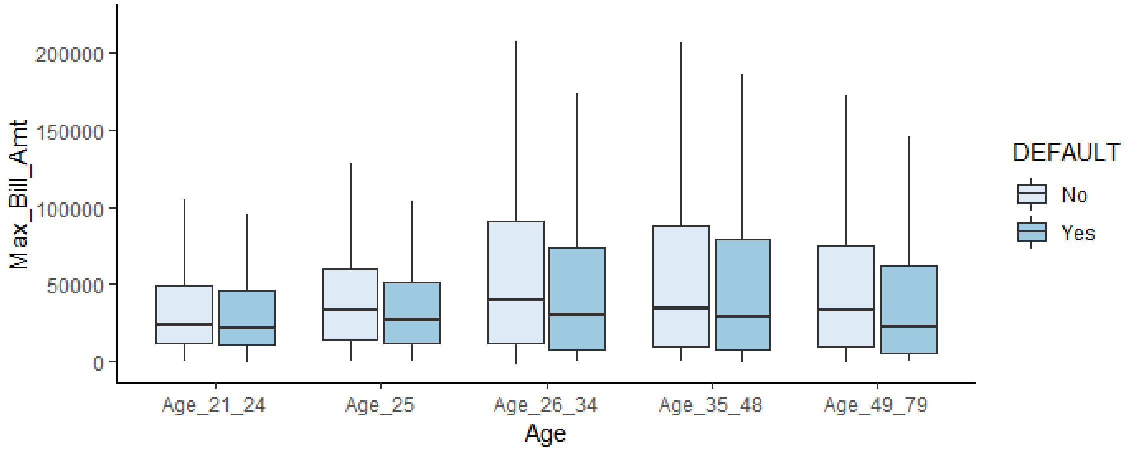
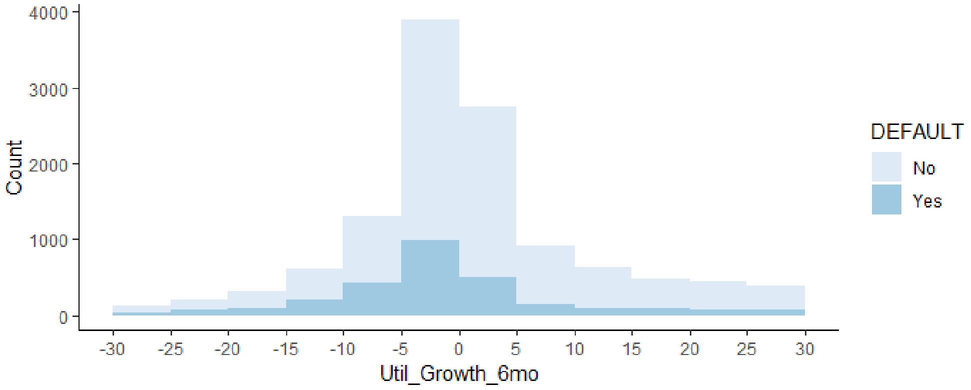


Figure 5 is a histogram of *Util_Growth_6mo* with respect to *DEFAULT*. The plot shows the distribution of *DEFAULT* across a cross-section of the utilization growth between April and September 2005. There is evidence that suggests that for clients that fell into the default payment class, their likely utilization growth for that six-month period is between -10% and 5%.

Figure 5

Util_Growth_6mo (Truncated)



Moreover, it’s also useful to study the relationship between only categorical variables and *DEFAULT*. Figure 6 shows the distribution of *MAX_DLQ* with respect to each of the default payment classes. The data shows that between April and September 2005, the most common payment status for clients was being one month behind. A significant portion of those clients that were one month behind fell into the non-default payment class. As such, the *MAX_DLQ* variable may not be a useful predictor of a default payment.

Figure 6

Max_DLQ

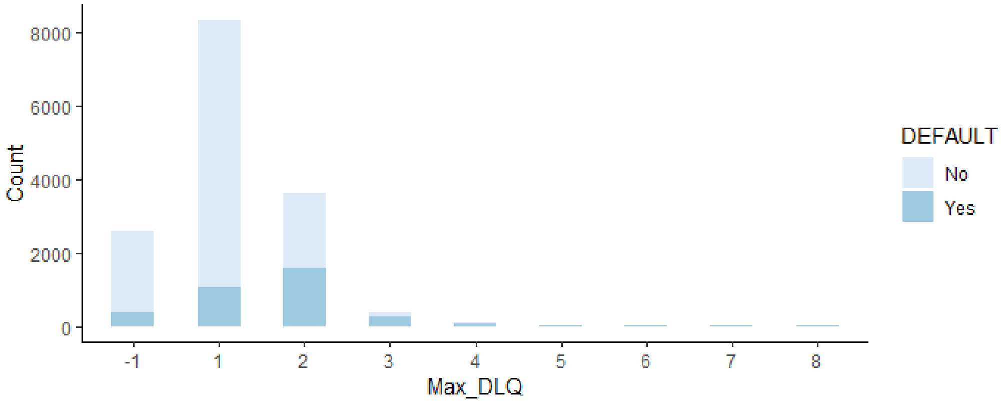
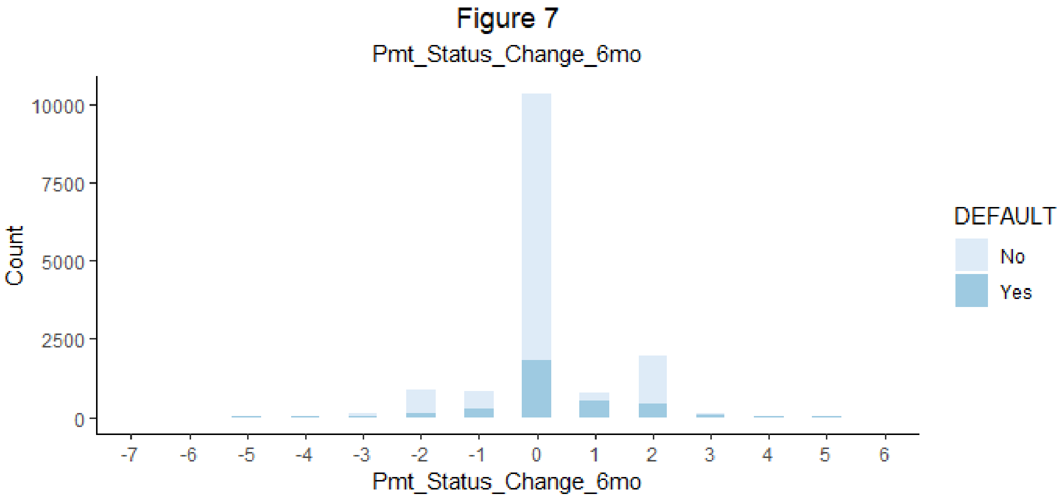


Figure 7 shows the distribution of *Pmt_Status_Change_6mo* with respect to *DEFAULT*. A clear takeaway is that most of the clients did not have a change in their payment status between April and September 2005. This variable does not seem that it will be useful in classifying a default payment client given the large number of non-default payment clients that fell into the "0" bin. This variable would have been beneficial if the default payment clients fell into one or more bins without having non-default payment clients in that same bin or bins.



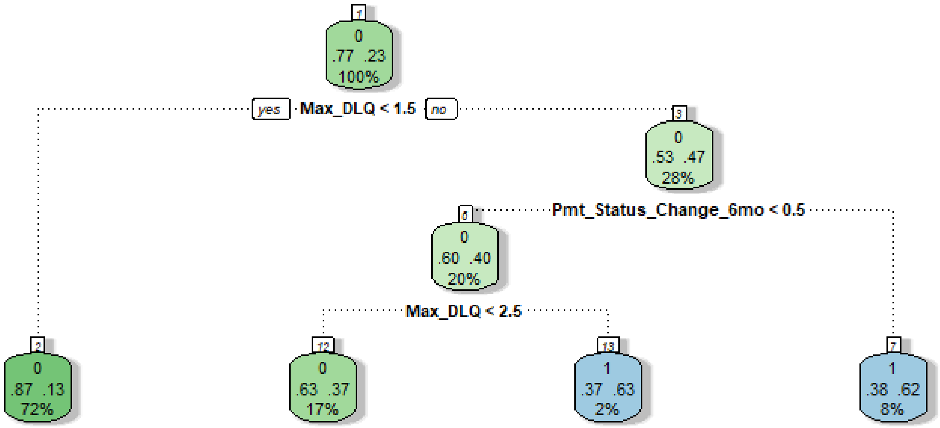
Overall, the traditional EDA approach yielded some useful insights into the relationship between various predictor variables and *DEFAULT*. However, as data sets become wider (e.g. more variables), there comes a point where using computational methods to explore the data makes the model development process more efficient. The subsequent section will cover a model-based approach to EDA.

4. Model-Based Exploratory Data Analysis

After completing traditional EDA, there are now 29 total predictor variables in the pool. There are five categorical variables – *SEX, EDUCATION, MARRIAGE, Age, Bal_Growth_6mo* and *LIMIT_BAL_Cat*. As with traditional EDA, the goal of model-based EDA is to determine if any variables create a separation of the *DEFAULT* classes. Using the R "rpart" library, a decision tree was fit on the entire data set and the results are displayed in Figure 8. The decision tree made the initial split on the *MAX_DLQ* variable and can be interrupted as follows:

- For all observations in the training set, when *MAX_DLQ* is less than 1.5, 87% of those observations fell into the non-default payment class and 13% fell into the default payment class. This rule applies to 72% of the total observations in the training set.

Figure 8: Decision Tree – MAX_DLQ Split



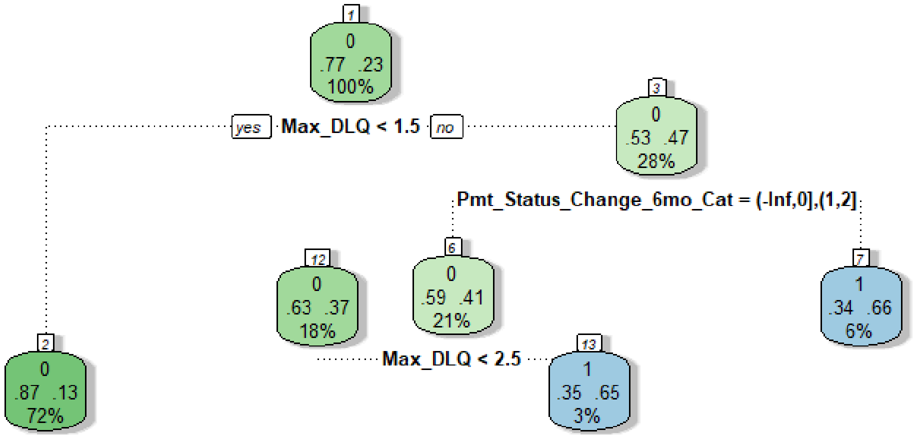
The decision tree keyed on the *MAX_DLQ* variable which was already analyzed during the traditional EDA for class separation. Given that the goal is to classify default payments, this variable doesn't necessarily aid that endeavor.

As a form of experimentation, the following variables will be discretized (using the same WOE optimal binning algorithm from previous sections) and used to fit a second decision tree:

- *Avg_Bill_Amt, Avg_Pmt_Amt, Avg_Pmt_Ratio, Avg_Util_Cat, Util_Growth_6mo, Max_Bill_Amt, Max_Pmt_Amt, Max_DLQ, Pmt_Ratio_Growth_6mo, Pmt_Status_Change_6mo*

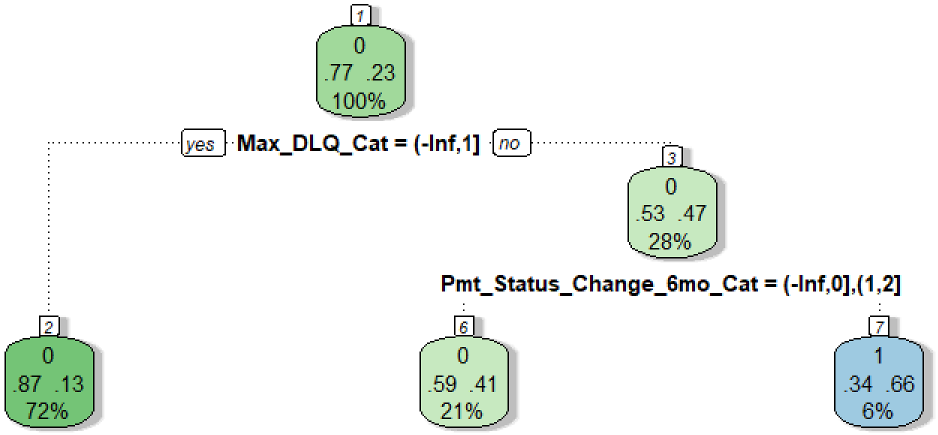
Figure 9 displays the results of fitting a second decision tree and can be interpreted in a similar manner as Figure 8. There isn't any new meaningful insight that was gained from adding discretized variables while also considering the previous variables used in the first decision tree.

Figure 9: Decision Tree – Add 10 Discretized Variables



The final decision tree fitted will only consider the categorical variables that are available. Figure 10 displays the results and once again the results resemble the previous two decision trees in the sense that the same variables were keyed on. The main difference is that the Figure 10 decision contains less splits.

Figure 10: Decision Tree – Only Categorical Variables



The previous results were gained from using the R “rpart” library, but further analysis was completed using the R “OneR” library. The R “OneR” library is based on a one rule classification algorithm which effectively builds 1-level decision trees across the predictor variables. The algorithm sorts the variables by highest accuracy.

Table 10 displays the accuracy results for the top five variables. A very good accuracy of 93.83% is achieved effortlessly when using the *Pmt_Status_Change_6mo* variable and will thus serve as a baseline for model performance when more sophisticated approaches are used. Accuracy is only one measurement of how well a classification model performs.

Table 10: OneR Accuracy Results

Variable	Accuracy %
Pmt_Status_Change_6mo	93.83%
Pmt_Ratio_Growth_6mo	82.56%
Avg_Util_Cat	81.53%
Max_Bill_Amt_Cat	81.53%
Avg_Bill_Amt_Cat	81.46%

Overall, the model-based EDA provides evidence to suggest that *MAX_DLQ* and *Pmt_Status_Change_6mo* (as well as their discretized versions) will be important predictor variables to classify a default payment. At the conclusion of the entire EDA process, there are 39 predictor variables available in the pool to train a credit card default model.

5. Predictive Modeling: Methods and Results

Three different modeling approaches – Random Forest, Gradient Boosting and Logistic Regression – were utilized to determine an optimal credit card default model. The Random Forest and Gradient Boosting models will help narrow down the current pool of predictor variables and the resulting set will be utilized to train the Logistic Regression model. In terms of consistency, the current pool of 39 predictor variables will be utilized to train the Random Forest and Gradient Boosting models. The difference in each of those models is with respect to their hyperparameters. The Logistic Regression model is the one that will be deployed into production after the entire model development process is complete.

Performance will be measured on both the training and test data set by comparing each model’s True Positive rate, False Positive rate and Accuracy. The test data set will be prepared in the same manner as the training set including applying the same bins to the test set from the training set.

5.1 Random Forest

Random Forest is considered one of the most popular “out-of-the-box” machine learning algorithms. These types of models can be used for regression and classification problems. Also, they do not require an excessive amount of pre-processing. Furthermore, hyperparameter tuning is an important part of training a Random Forest model. Hyperparameter tuning involves determining what values to use for each hyperparameter. There are several hyperparameters that can be reviewed as part of this process but only the number of trees (*ntrees*), maximum number of nodes (*maxnodes*) and the number of variables to randomly sample as candidates at each split (*mtry*) will be considered as part of training the Random Forest mode. The R “caret” and “randomForest” library will be utilized to develop the Random Forest model.

The following is a summary of what will be completed for the Random Forest model development:

- Three Random Forest models will be trained sequentially using the Random Search method to find optimal *ntrees, maxnodes* and *mtry* values (other hyperparameters will use default values); the values placed in each grid for each hyperparameter that the

Random Search method will choose from will vary (e.g. hyperparameter value grid for *ntree* will include 400, 500, 600, 700, 1000 and 2000)

- k-fold cross-validation (k=5) will be utilized to measure accuracy for each model as part of determining optimal *ntrees*, *maxnodes* and *mtry* values
- Train final Random Forest model using optimal *ntrees*, *maxnodes* and *mtry* values
- Review True Positive rate, False Positive rate and Accuracy on training and test set
- Provide relevant output for model (e.g. variable importance plot)

After completing the hyperparameter tuning steps, the optimal values were calculated to be as follows:

- number of trees (*ntrees*) = 400
- maximum number of nodes (*maxnodes*) = 12
- number of variables to randomly sample as candidates at each split (*mtry*) = 12

After fitting the final Random Forest model, several outputs need to be reviewed to help evaluate model attributes and performance. Figure 11 is the variable importance plot which can be interpreted as showing which variables are the most useful in classifying *DEFAULT*. Specifically, the Figure 11 plot ranks the total decrease in node impurity that results from splits over that variable which is averaged over all trees (Mean Decrease Gini). The ability for a variable to decrease node impurity is a desired characteristic. *MAX_DLQ, MAX_DLQ_Cat* and

Pmt_Status_Change_6mo_Cat are the top three variables that decrease node impurity.

Figure 11: Random Forest Model Top 15 Features

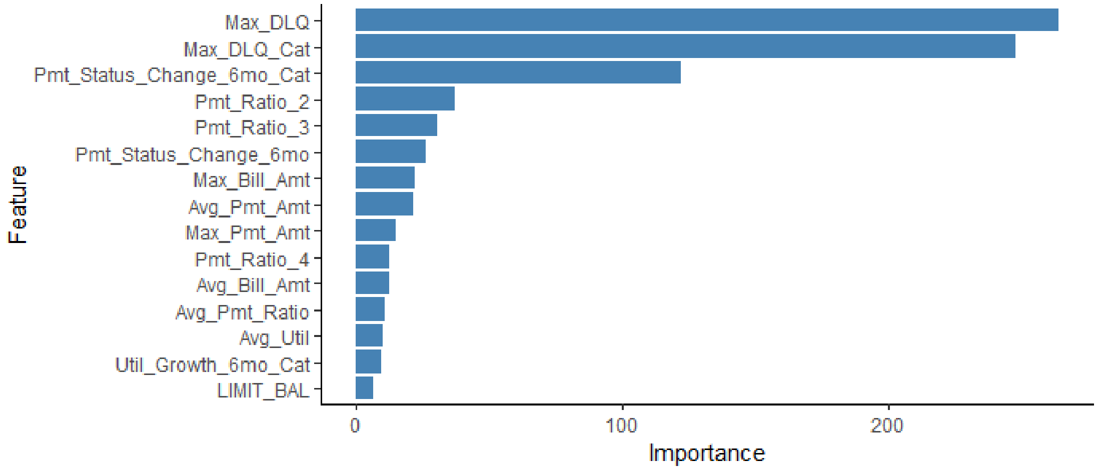


Table 11 displays the same data from Figure 11 but also adds a secondary description of variable importance – Mean Decrease Accuracy, which represents the mean decrease in accuracy in predictions on the out of bag samples when a variable is withheld. With respect to Mean Decrease Accuracy, *Pmt_Status_Change_6mo_Cat* is the most important variable.

Table 11: Random Forest Top 15 Features

Feature	**Mean Decrease Accuracy**	**Mean Decrease Gini**
Pmt_Status_Change_6mo_Cat	18.57	122.08
Max_DLQ	14.75	264.36
Max_DLQ_Cat	12.85	248.34
Pmt_Ratio_3	8.69	30.48
Avg_Pmt_Ratio	8.67	11.03
Pmt_Ratio_2	8.57	36.85
Pmt_Status_Change_6mo	8.51	26.19
Pmt_Ratio_4	5.34	12.63
Util_1	5.13	6.38
Max_Bill_Amt	4.85	21.83
Util_Growth_6mo	4.59	6.46
Util_Growth_6mo_Cat	4.36	9.61
Avg_Pmt_Ratio_Cat	4.30	4.66
Util_3	4.17	2.80
Max_Pmt_Amt	4.049	14.70

A review of the feature importance data indicates that the change in a client's payment status (e.g. how many months they are behind on their credit card payments) over the past six months is a strong predictor on whether they will have a default payment. Furthermore, it appears that the age of the client is not a strong predictor.

Another useful output from the Random Forest model is the confusion matrix. Figure 12 displays the confusion matrix (A=Actual; P=Predicted) which has been normalized to show the rates instead of the raw value counts. Based on the rates displayed, the Random Forest model does not seem to be able to predict a default payment accurately when it exists based on the 21% True Positive rate. Furthermore, it does seem to be able to predict a non-default payment when it exists based on the 97% True Negative rate. The high True Negative rate does not necessarily mean the model will be able to generalize well on new data. The Random Forest model could have been "lazy," predicted a non-default payment and still had a decent chance at being correct since most of the training set contained non-default payment clients.

Table 12: Random Forest Confusion Matrix

	Non-Default Payment (P)	Default Payment (P)
Non-Default Payment (A)	0.97	0.03
Default Payment (A)	0.79	0.21

The last step is to evaluate the Random Forest model by applying it to the test data set. Table 13 displays the True Positive rate, False Positive rate and Accuracy for both the training and test data sets. The model performed slightly better with respect to the True Positive rate and Accuracy metrics on the test data set. Although the results are not ideal, the performance on the test data set was not significantly lower than the training set which provides some level of comfort in the model’s classification ability (i.e. the model did not overfit the data). Overall, the model does not seem to do a good job at classifying a default payment when it exists.

Table 13: Random Forest Performance Results

Metric	Training Set	Test Set
True Positive Rate	0.211	0.212
False Positive Rate	0.028	0.029
Accuracy	0.799	0.809

Moreover, the performance metrics displayed were computed using the default rules from the R “randomForest” library and not from using an optimal cut-off value. An optimal cut-off value will be calculated as part of the Logistic Regression model evaluation. That cut-off value will then be utilized to recalculate the performance results for the Random Forest and Gradient Boosting model. This approach will standardize the performance results across all three models for a more accurate comparison.

5.2 Gradient Boosting

Gradient Boosting models are another popular machine learning algorithm. Unlike Random Forest which is an ensemble of individual decision trees, Gradient Boosting is an ensemble of shallow and weak successive models (in this case, decision trees) in which each model improves from the previous one. As such, Gradient Boosting can be described as an additive model. The R “caret” library will be used to develop the Gradient Boosting Model.

As with the Random Forest model, there are several hyperparameters that can be considered as part of the tuning process. Only certain ones will be considered for the Gradient Boosting model development. The following is a summary of the hyperparameter settings that will be used to train the final Gradient Boosting model:

- number of trees (*n.trees*) = 2,000

- depth of trees (*interaction.depth*) = 2
- learning rate (*shrinkage*) = 0.01
- minimum number of observations in node (*n.minobsinnode*) = 1

After fitting the Gradient Boosting model, there are several outputs that are reviewed to evaluate model attributes and performance. Table 14 is the variable importance plot which indicates which variables are the most useful in classifying *DEFAULT*. Specifically, the plot ranks the relative influence which is the average increase in accuracy that is attained when a variable is used at a split. *MAX_DLQ*, *MAX_DLQ_Cat* and *Avg_Pmt_Amt* are the most important variables in classifying *DEFAULT* according to the Gradient Boosting Model.

Table 14: Gradient Boosting Top 15 Features

Feature	**Relative Influence**
Max_DLQ	27.10
Max_DLQ_Cat	19.47
Avg_Pmt_Amt	5.40
Pmt_Status_Change_6mo_Cat	5.21
Pmt_Status_Change_6mo	4.96
Max_Bill_Amt	4.19
Util_Growth_6mo	3.37
Avg_Pmt_Ratio	2.73
Avg_Bill_Amt	2.59
Max_Pmt_Amt	2.34
LIMIT_BAL	1.67
Pmt_Ratio_6	1.66
Pmt_Ratio_4	1.56
Avg_Util	1.50
Avg_Util_Cat	1.42

Another useful output from the Gradient Boosting model is the confusion matrix. Table 15 displays the confusion matrix which has been normalized to show the rates instead of the raw value counts. Based on the 32% True Positive rate displayed, the Gradient Boosting model does not seem to be able to predict a default payment accurately. However, this result was higher than the Random Forest's True Positive rate. Furthermore, the Gradient Boosting model does seem to be able to predict a non-default payment when it exists based on the 95% True Negative rate.

Table 15: Gradient Boosting Confusion Matrix

	Non-Default Payment (P)	Default Payment (P)
Non-Default Payment (A)	0.95	0.05
Default Payment (A)	0.68	0.32

The last step is to evaluate the Gradient Boosting model by applying it to the test data set. Table 16 displays the True Positive rate, False Positive rate and Accuracy for both the training and test data sets. As was the case for the Random Forest model, the performance metrics displayed for the Gradient Boosting model were computed using the default rules from the R library ("caret") and not from using an optimal cut-off value. The model performed slightly better on the Accuracy metric but performed worse on both the True Positive and False positive rates with respect to the test data set. Although the results are not ideal, the performance on the test data set was not significantly lower than the training set. Overall, the Gradient Boosting model does not seem to do a good job at classifying a default payment when it exists which was the same takeaway after building Random Forest model.

Table 16: Gradient Boosting Performance Results

Metric	Training Set	Test Set
True Positive Rate	0.324	0.320
False Positive Rate	0.046	0.051
Accuracy	0.812	0.816

5.3 Logistic Regression with Variable Selection

Logistic Regression models calculate the probability that an observation falls into a certain class. In this case, the Logistic Regression model will use the available predictor variables to calculate the probability of a default payment. Table 17 displays the pool of predictor variables that will be considered by the variable selection algorithm as part of the Logistic Regression model development. This pool of predictor variables is less than the original 39 predictor variables that were available when training the Random Forest and Gradient Boosting models. The development of those models helped identify the new pool of predictor variables for the Logistic Regression model.

The top 15 important features that contributed to accuracy in the Random Forest and Gradient Boosting models were compared. Both models had nine variables in common which were added to the pool. The top two remaining variables from each respective model were then added which resulted in a pool of 13 predictor variables.

Table 17: Predictor Variables for Logistic Regression

Variable	Definition
Max_DLQ_Cat	Discretized version of Max_DLQ
Avg_Bill_Amt	Represents the average of BILL_AMT1 through BILL_AMT6
Avg_Pmt_Amt	Represents the average of PAY_AMT1 through PAY_AMT6
Avg_Pmt_Ratio	Represents the average of Pmt_Ratio_2 through Pmt_Ratio_6
Pmt_Ratio_4	Represents the % of June 2005 bill paid; (PAY_AMT3/BILL_AMT4)*100
Pmt_Status_Change_6mo_Cat	Discretized version of Pmt_Status_Change_6mo
Pmt_Ratio_3	Represents the % of July 2005 bill paid; (PAY_AMT2/BILL_AMT3)*100
Util_Growth_6mo	Represents difference between Util_1 and Util_6
Max_Bill_Amt	Max value of BILL_AMT1 through BILL_AMT6
Max_Pmt_Amt	Max value of PAY_AMT1 through PAY_AMT6
Max_DLQ	Max value of PAY_1 through PAY_6
Pmt_Ratio_2	Represents the % of August 2005 bill paid; (PAY_AMT1/BILL_AMT2)*100
Pmt_Status_Change_6mo	Represents difference between PAY_1 and PAY_6

Unlike the Random Forest and Gradient Boosting models, there wasn't any hyperparameter tuning necessary for the Logistic Regression model. The only decision that had to be made was choosing which variable selection algorithm would be used to fit the model. The stepwise algorithm was chosen. It is a semi-automated process that builds the Logistic Regression by model by successively removing or adding variables from the predictor pool based on the t-statistics of their estimated coefficients. The R "glm" library was used for this process. Table 18 is a summary of the model which lists all the variables (as well as the intercept), their estimated coefficients, standard error and statistical significance (denoted by asterisks).

Table 18: Logistic Regression Model Summary

Variable	**Coefficient Estimate**	**Std. Error**
Max_DLQ_Cat (>= 1)	1.29***	0.07
Pmt_Status_Change_6mo (0,1]	0.99***	0.07
Pmt_Status_Change_6mo (1,2]	0.02	0.10
Pmt_Status_Change_6mo (>= 2)	0.29	0.13
Avg_Pmt_Amt	-0.0001***	0.23
Max_Pmt_Amt	0.00002***	0.00001
Avg_Bill_Amt	0.00001***	0.000003
Util_Growth_6mo	-0.002	0.000002
Avg_Pmt_Ratio	0.005***	0.0009
Max_DLQ	0.16***	0.001
Pmt_Status_Change_6mo	0.10**	0.04
Max_Bill_Amt	-0.000002	0.000001
Constant	-1.99	
Observations	15,180	
Log Likelihood	-6,992.21	
Akaike Inf. Crit.	14,010.43	
Note:	*p<0.1; **p<0.05; ***p<0.01	

The results from Table 18 show that not all variables from the predictor pool were included in the model when using the stepwise variable selection algorithm. For example, the

Pmt_Ratio_X variables were left out but the *Avg_Pmt_Ratio* variable was included. Furthermore, the results show that the *Max_DLQ_Cat* variable has the largest impact (as measured by its estimated coefficient) on the probability of a default payment occurring. Specifically, this can be interpreted to mean that the odds of a default payment occurring versus not increases by a factor of 3.36 (result of exponentiating 1.29) if a client gets behind on a payment by at least one month. Overall, most of the variables in the Logistic Regression model are statistically significant.

One of the biggest differences between the results of the Logistic Regression model and the previous two models developed is that it predicts the probability of a client belonging to the default payment class rather than classifying a client belonging to a specific *DEFAULT* class. Therefore, it is necessary to create an optimal cutoff value (i.e. optimal probability) that will allow the probabilities to be converted into classification results.

Using the Logistic Regression model's fitted values, a receiver operating characteristic curve (ROC) is created using the R "pROC" library. For all points on the ROC curve, the optimal cutoff value is recognized as the one that maximizes Youden's J Index which is defined as follows:

$$\mathrm{J} = \mathrm{Sensitivity} + \mathrm{Specificity} - 1$$

A J Index with a value of "1" means that there are no false positives or false negatives. Table 19 displays the results of using the Logistic Regression model to calculate the optimal cutoff values. Therefore, any client with a predicted probability greater than 23.7% will be classified as a default payment client.

Table 19: Logistic Regression Optimal Cutoff Value Results

Metric	Value
Optimal Cutoff Value	0.237
Specificity	0.793
Sensitivity	0.616

After determining the optimal cutoff value, the Logistic Regression model's results are displayed in Table 20 using a confusion matrix. These results are significantly better than the initial results from the Random Forest and Gradient Boosting models. However, the results from the prior models need to be standardized so that a more valid comparison can be done.

Table 20: Logistic Regression Confusion Matrix

	Non-Default Payment (P)	Default Payment (P)
Non-Default Payment (A)	0.79	0.21
Default Payment (A)	0.38	0.62

The last step is to evaluate the Logistic Regression model by applying it to the test data set. Table 21 displays the True Positive rate, False Positive rate and Accuracy for both the training and test data sets. The model performed worse on the False Positive rate and Accuracy metrics but performed better on the True Positive rate with respect to the test data set. Although the results are not ideal, the performance on the test data set was not significantly lower than the training set which provides some level of comfort in the model's classification ability (i.e. the model did not overfit the data). Overall, the model does seem to do an average job at classifying a default payment when it exists.

Table 21: Logistic Regression Performance Results

Metric	Training Set	Test Set
True Positive Rate	0.616	0.626
False Positive Rate	0.208	0.217
Accuracy	0.753	0.749

6. Comparison of Results

In order to make an accurate comparison of model performance, the Random Forest and Gradient Boosting model's Accuracy, True Positive (TP) and False Positive (FP) metrics were recalculated using the optimal cutoff value determined from the Logistic Regression Model. It was not necessary to recalculate the area under the curve (AUC).

In terms of Accuracy, the Random Forest performed the best on both the training and test data set. It was the only model that had a higher Accuracy metric on the test data set than the training set. As such, its accuracy was higher than the average of all the models. The Random Forest Model fell short with respect to the True Positive rate. It had the lowest rate on the training and test data sets. Since the purpose is to develop a credit card default model that will accurately predict a default payment, the Random Forest model cannot be considered the best model even though it has the highest accuracy.

Table 22: Model Performance Comparison

	Training Data Set				**Test Data Set**			
Model	**AUC**	**Accuracy**	**TP**	**FP**	**AUC**	**Accuracy**	**TP**	**FP**
Random Forest	0.764	0.805	0.279	0.042	0.747	0.812	0.280	0.045
Gradient Boosting	0.796	0.752	0.669	0.224	0.777	0.741	0.656	0.236
Logistic Regression	0.749	0.753	0.616	.208	0.757	0.749	0.626	0.217
Average	0.77	0.77	0.521	0.158	0.76	0.767	0.521	0.166

The Gradient Boosting model had the highest AUC score on both data sets which is important because it provides evidence that the model is better than a random guess of the *DEFAULT* classes. Furthermore, it had the highest True Positive rate across both data sets which is a desired characteristic because classifying a default payment correctly is what’s most important. However, its high True Positive rate comes in tandem with a large False Positive rate across both sets. In the case of predicting a default payment, this is not necessarily a large concern unless there is a significant cost associated with incorrectly predicting a client will have a default payment.

There isn’t a noticeable difference with the Logistic Regression model’s performance. Its performance was relatively “in between” both the other models. The one key difference is that it had the lowest AUC score on both data sets. Also, its True Positive rate was significantly higher than the Random Forest model but slightly lower than the Gradient Boosting model.

Overall, each model succeeded in the sense that their relative performance on the test data set did not drastically differ from the training set. In other words, none of the models overfit or underfit the data. However, it can be argued that each model did underfit the data due to the accuracy and AUC results. Based on the overall performance metrics, the Gradient Boosting model is considered the best performing one. The goal of the credit card default model is to predict if a client will have a default payment. The Gradient Boosting model excelled in that area while remaining competitive in the other areas of performance.

7. Conclusions

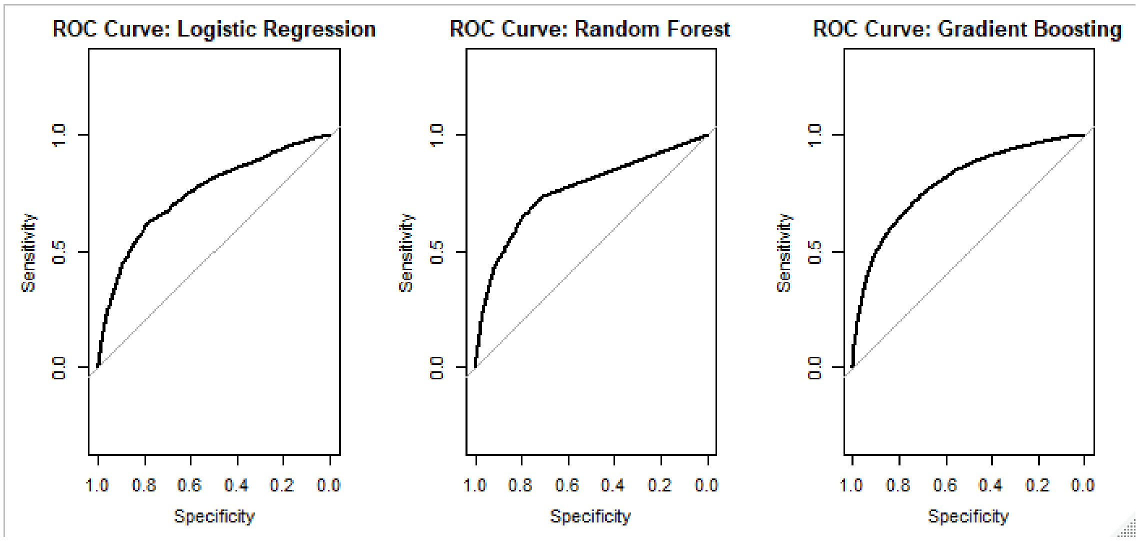
Financial institutions deal with a significant amount of risk that comes from competitive, regulatory and macroeconomic pressures. Given the abundance of data available to them, financial institutions have historically made use of this data by building models for various areas of their business. The advancements in technology have only aided in the development of more sophisticated and accurate models. Specifically, one of the areas that many financial institutions focus on is being able to determine if their current and new clients will be able to pay back money that is owed. The money that is owed can take on many product forms such as a private loan, mortgage or credit card. Being able to correctly classify an individual’s ability to pay back what’s owed is part of necessary risk management processes.

This model development guide was focused on developing a credit card default model that can accurately classify a default payment occurring based on a variety of information pertaining to a client such as their age, education level and past payments. The development of the final model included data preparation, exploratory data analysis, feature engineering and evaluating several model classes for performance. The exploratory data analysis suggests that a client's average credit card balance limit utilization, the change in a client's payment status over the past six months and if a client ever gets behind one month in making payments were going to be important predictors of a default payment. It was surprising to not see any class separation when considering a client's age or education level. The most important predictors turned out to be variables that considered a client's payment status. Overall, the different modeling approaches yielded similar results. There wasn't a model that had the best performance across every metric considered. The average accuracy of the models was 76.5% across both the training and test data sets.

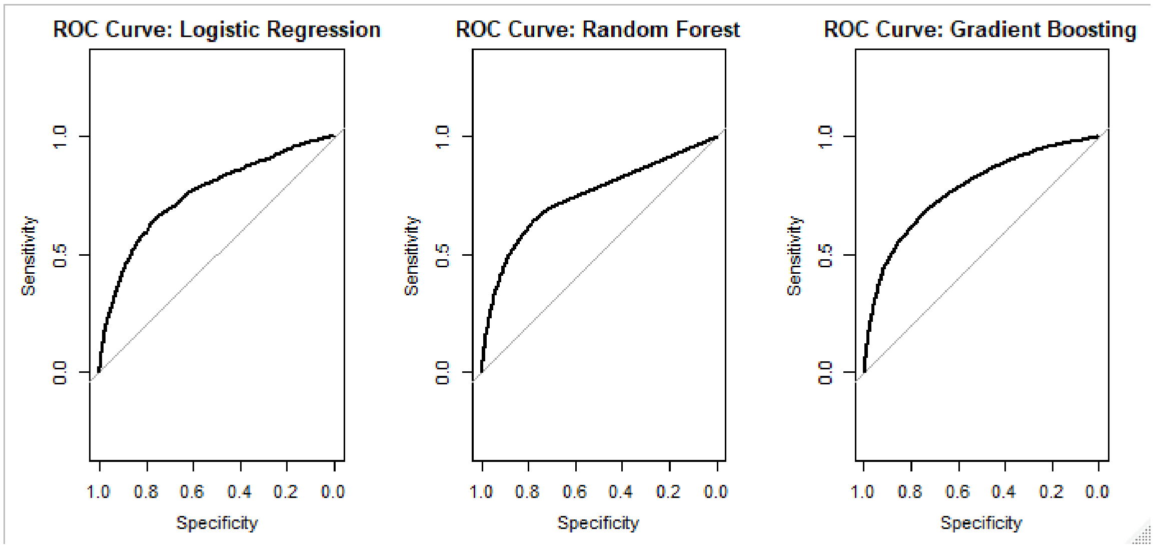
Although there was considerable care taken in the development of the final model, there still exists various other approaches that can be explored to improve model performance. Extra feature engineering steps could have been taken such as creating variables that account for percentage growth rather than absolute change within a certain time period. Furthermore, other model classes could have been considered such as Neural Networks and Naïve Bayes to determine if an improvement in model performance is attainable. Overall, the results from this model development guide provide an indication of which variables are important when trying to predict a client default payment. Further research should involve the expansion of the data set to include more than six months' worth of data and other variables that are not related to payment history.

9. Appendices

Appendix A: ROC Curves – Training Set



Appendix B: ROC Curves – Test Set


ROC Curve: Logistic Regression
ROC Curve: Random Forest
ROC Curve: Gradient Boosting
Sensitivity
Specificity