Financial Loan Risk Classification

UNIVERSITY OF CALGARY

Data 606

Final Project

Youssef Abdelwahab, Maryam Ahmed, Golin Chen, Tasheika Wilson

Abstract

Creditworthiness assessments impact millions of individuals and businesses, shaping access to loans, mortgages, and financial opportunities. Traditional credit evaluation methods, while widely used, often rely on rigid criteria that may not fully capture the complexities of an individual's financial situation. This study investigates the relationship between an individual's revenue-generating capacity, financial circumstances, and creditworthiness, operationalized through two dependent variables: a categorical financial risk grade (assigned by lenders to denote loan delinquency likelihood) and a continuous FICO_range (reflects credit health and repayment likelihood).

Objective

The primary objective is to develop generalized predictive models that quantify the influence of financial, demographic, and loan-specific variables on these credit outcomes. The proposed models aim to (1) classify categorical loan risk grades and (2) evaluate the marginal effects of predictor variables on continuous credit scores (FICO_score).

Data Exploration

The analysis utilizes a real-world dataset from LendingClub.com (hosted on Kaggle) (SITE), comprising 1,048,576 accepted loan applications spanning 2007 to 2018. The dataset includes 111 variables capturing applicant demographics, income metrics, credit history, debt-to-income ratios, and loan-specific parameters. Key variables include the financial risk grade, which stratifies loans into discrete categories (A–F, where A denotes the lowest risk of default), and the FICO_score, a standardized creditworthiness metric ranging from 600 to 850, as the data set is only of approved loans.

Data Cleaning

To begin our analysis and gain a deeper understanding of the data, we first focused on data preparation. This involved cleaning the dataset by removing rows with more than 100,000 missing values in any column. The rationale behind this step was to eliminate variables that lacked sufficient data to provide meaningful insights into both our categorical and continuous dependent variables. Additionally, we removed columns that were irrelevant to the scope of our project, such as membership IDs, URLs, job titles (emp_title), and other variables like policy codes that contained over 100 unique combinations.

Once the data was cleaned, we turned our attention to preparing it for regression modeling. Specifically, we needed a single value between fico_high and fico_low to use as our predictive variable. To address this, we created a new variable called fico_avg, which we used in place of the credit score to ensure a more manageable and effective analysis moving forward.

Testing the Normality Assumption

Various statistical learning methods assume the normal distribution. Therefore, we must check if this assumption upholds within the various variables in our dataset to be able to apply the appropriate learning methods.

3.1 Testing Categorial Variables

Measurements can be expressed in qualitative or categorical terms. They can be expressed as counts of instances that fall into each of the classes of category. In the case of our dataset the variables, *[grade], [sub_grade], [home_ownership], [pymnt_plan], [initial_list_status], [application_type], and [hardship_flag]* are categorical variables.

The *goodness of fit test* uses the chi-square statistic to test whether sample data indicates that a specific model for a population distribution does not fit the data **[1]**, in our case it is the normal distribution. Large values obtained from our observed distribution indicate significant deviation from our expected distribution. Give us very small p-values [appendix] providing us with statistical evidence to reject our null hypothesis at the significance level of $\alpha = 0.05$, that our data follows the normal distribution. Figure 1. shows normalized (0-1) chi-square statistics for our categorical variables on a radar plot

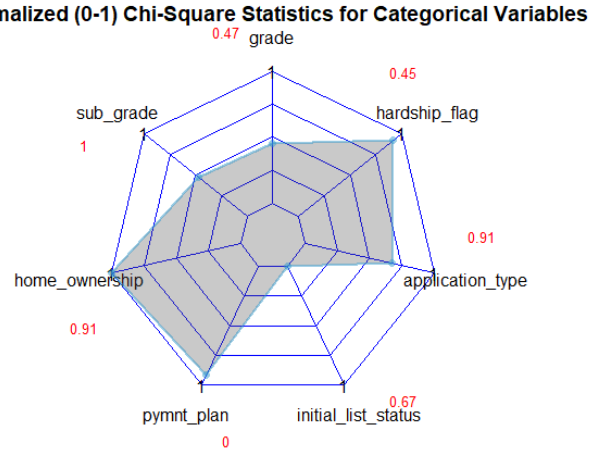
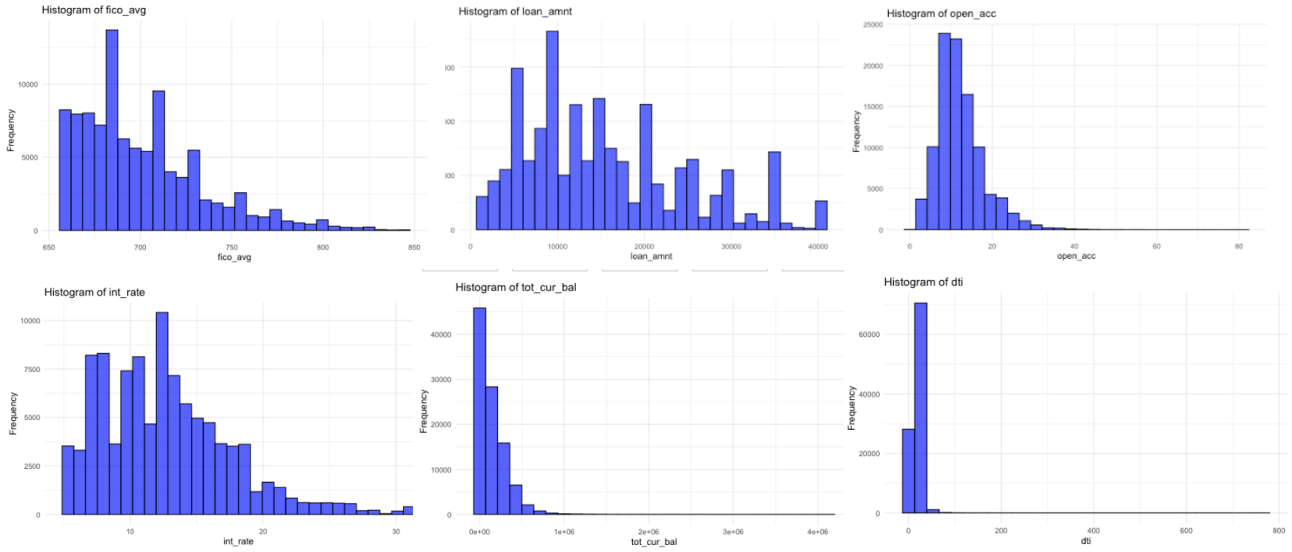


Figure 1. Chi-Statistic Radar Plot

3.2 Testing Numerical Variables

To assess the normality assumption, we first created histograms for key continuous variables, including [*FICO Avg], [Loan Amount], [Open Accounts], [Interest Rate], [Total Payment Involved], [Total Current Balance], [Debt-to-Income Ratio]* and [*Annual Income]*.

Upon reviewing the histograms, we observed that our dependent variable, *FICO Avg* (*first graph*



in figure 2), is heavily right-skewed, indicating that many approved applicants fall on the lower end of the FICO spectrum. Moreover, none of the variables appear to follow a normal distribution, with the possible exception of *Open Accounts*.

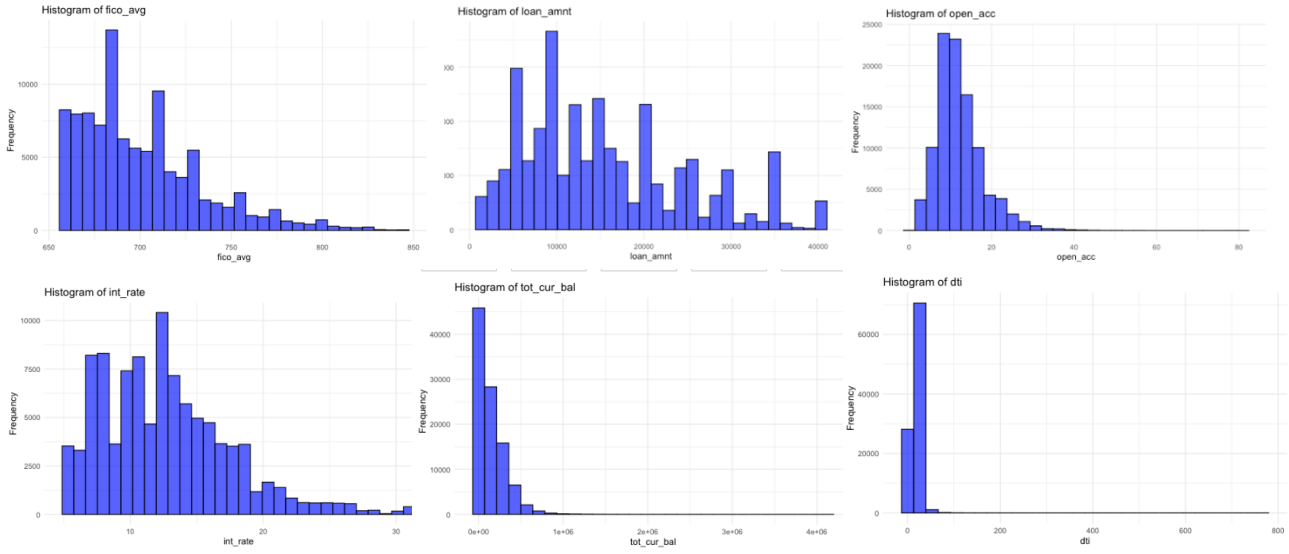


Figure 2: Histograms of Independent Variables

To validate our assumptions of the data's normality and better understand the underlying data, we employed the Kolmogorov-Smirnov (KS) test. This test is designed to assess whether a sample comes from a specific distribution, in this case, a normal distribution **[6].** Technically, the KS test compares the distribution of sample A to the distribution of sample B. Since the KS test is nonparametric **[7],** it does not assume any specific underlying distribution, but in this context, we are assuming that the distribution of A is normal, while B represents our variable of interest. The test follows a hypothesis framework:

- **Null Hypothesis**: The data follows the specified distribution (normal).
- **Alternative Hypothesis**: The data does not follow the specified distribution (normal).

After applying all the variables to the KS test, all the variables p-values where less than the α of 0.05 reaching values to 8e-71, hence we reject the null hypothesis in favor of the alternative, concluding that the variables do not follow a normal distribution.

Predictive Models

To predict both the categorical variable [*Grade]* and the continuous variable [*FICO* Score], we adopted a multifaceted approach. Specifically, we developed two classification models to predict Grade and two regression models to estimate the [*FICO Score*]. In the following section, we will describe each model in detail, discuss their limitations, and outline the various approach processes and iterations we explored before finalizing our modeling strategies.

Regression Trees

The Regression tree begins by filtering various loan-related features, such as loan amount, interest rate, annual income, and debt-to-income ratio. The goal of this analysis is to create a predictive model that can accurately estimate the FICO score range based on these features. The process includes feature selection, model training, cross-validation, pruning to prevent overfitting, and evaluation of the model's performance **[8].**

The dataset was then split into features (independent variables) and the target variable (fico_range_avg) which is the mean value between the fico_range_low and fico_range_high. To identify the most relevant features for predicting fico_range_avg, stepwise regression was employed. This method iteratively adds or removes features based on their statistical significance (p-values < 0.05) [Appendix 4] .

A regression tree was constructed using the training data. The tree works by recursively splitting the data into subsets based on the feature values, aiming to minimize the variance of the target variable within each subset. Each subset is considered as a node, like branch on the tree each split will result in a better accuracy of the model when going to the lower branches. However, we can encounter an issue of overfitting where unnecessary node is created increasing runtime and might result in poor performance. The model was trained to predict fico_range_avg using the selected features.

To assess the model's robustness and generalizability, cross-validation was performed. This technique divides the training data into 8 subsets and uses an 8-2 split to train our model. The process is repeated 8 times, and the average performance is calculated. The cross-validation results provided an estimate of the model's performance on unseen data, with the root mean squared error (RMSE) serving as the primary metric.

To prevent the regression tree from overfitting the training data, pruning was applied by limiting the maximum depth of the tree. Overfitting occurs when the model captures noise in the training

data, leading to poor performance on new data. The tree was simplified by setting a maximum depth of 5, ensuring it generalizes well to unseen data while maintaining predictive accuracy.

High Order Linear Model

A higher-order linear model is one that remains linear in its parameters but incorporates non-linear transformations such as polynomial terms of the predictor variables. This approach allows us to model complex, non-linear relationships while still leveraging the simplicity and interpretability of linear models **[8].**

Our process began with developing a baseline linear model to predict the [FICO score] using over 37 predictor variables. To identify the most influential factors affecting the average score, we initially employed a stepwise selection method with 10-fold cross-validation. Unfortunately, this approach led to severe overfitting on the training data and poor performance on the testing data.

To mitigate overfitting and improve variable selection, we turned to LASSO regression (Least Absolute Shrinkage and Selection Operator). LASSO adds an L1 regularization penalty proportional to the sum of the absolute values of the coefficients to the ordinary least squares objective **[9].** This penalty not only shrinks the coefficients, which reduces overfitting, but also forces some coefficients to become exactly zero, effectively selecting the most relevant variables from our large pool of predictors **[9].**

After refining our model using LASSO regression, we proceeded to enhance it by incorporating higher-order terms. We then evaluated whether these additional non-linear variables improved the model's performance and contributed significantly to predicting the [FICO score].

Multinomial Logistic Regression

Multinomial Logistic Regression (MLR) is an extension of Binary Logistic Regression used for classification problems where the dependent variable has more than two categories **[5]**. Compared to the standard logistic regression or other methods requiring pairwise or One-vs-Rest strategies, the MLR models the probability of all possible outcomes simultaneously.

This is done by estimating log-odds for each class relative to a reference class. This approach is computationally efficient and provides interpretable probabilities for all classes directly. For a response variable Y with K classes, the model computes the probability P(Y=K) for each class K using the softmax function **[4].** The softmax function converts the raw log-odds scores into probabilities. After computing probabilities for all classes, the model picks the class with the maximum probability as the final prediction. This means that the predicted class is simply the one with the highest probability after applying the softmax function.

Multinomial logistic regression assumes linear relationships between features and log-odds. While this restricts it to linear separability, it avoids the computational complexity of kernel methods and provides transparent coefficient interpretations. With MLR coefficients indicate the direction and magnitude of feature impacts on log-odds, aiding actionable insights.

Support Vector Machines

The support vector machine is an extension of the support vector classifier that is utilized when the boundary between classes is nonlinear **[3]**, which occurs when multiple classes exist to be classified *[grade]* such is in our case. The process of conducting multinomial classification is conducted obtaining the dot product of two observations after enlarging the feature space to address this non-linearity over all observations in the dataset. These processes are called kernel methods.

Powell et al. kernel introduced the radial basis function which is a kernel function whose output is equivalent to an inner product in a higher-dimensional space, computed implicitly via the Euclidean distance **[2]**. This makes it more computationally efficient than the polynomial kernel which requires expanding the features combinatorically explicitly, which is our motivation for utilizing it in our model.

We use One-vs-Rest multiclass classification for our problem. It involves training one binary classifier per class in our response variable where each classifier distinguishes "class *i*" from all other ones. At prediction time the input is passed through all the classifiers and the one that yields the highest accuracy score is chosen for the output.

Model Assessment

Regression Tree

Using the stepwise selection the model is left with 7 variable nodes which include, Loan amount [*loan_amnt*], Interest rate [*int_rate*], funded amount [*funded_amnt*], annual income [*annual_inc*], Revolving balance [*revol_bal*], total collection amount [*tot_coll_amt*], total current balance [*tot_cur_bal*]. [appendix 4]

These features were chosen because they demonstrated a strong relationship (p-value) with the target variable and contributed significantly to the model's predictive power. After building the tree for 8 different datasets and apply pruning techniques for overfitting. The pruned regression tree was evaluated on the test dataset using several metrics RSME, MSE MAE and R^2, across 8 different set here is the result:

Train	data1	data2	data3	data4	data5	data6	data7	data8	Avg
RMSE	24.886	24.854	24.896	24.908	25.029	25.039	24.991	24.960	**24.941**
MSE	619.347	617.752	619.812	620.427	626.470	626.969	624.576	623.038	**622.361**
MAE	0.430	0.435	0.434	0.431	0.429	0.432	0.429	0.430	**0.431**
R^2	19.497	19.456	19.516	19.467	19.614	19.612	19.591	19.527	**19.535**

Test	data1	data2	data3	data4	data5	data6	data7	data8	Avg
RMSE	25.023	25.022	25.006	24.918	25.062	24.813	25.135	24.883	**25.008**

MSE	626.181	626.110	625.335	620.913	628.142	615.686	631.749	619.160	**625.155**
MAE	0.431	0.428	0.425	0.422	0.421	0.427	0.431	0.425	**0.426**
R^2	19.567	19.690	19.570	19.562	19.621	19.443	19.701	19.537	**19.589**

- The RMSE values are very close, 24.941 and 25.008 indicating that the model performs similarly on both the training and testing datasets. This suggests that the model is generalizing well and not overfitting.
- Similar result in the MSE values, they are also very close, further confirming that the model's performance is consistent across both datasets.
- It is nearly identical for the MAE values of 0.431 and 0.426, indicating that the model's average prediction error is consistent on both the training and testing datasets.
- Lastly the R^2 value suggesting that the model explains a similar proportion of variance in both the training and testing datasets

The model's performance on the testing data is very close to its performance on the training data. This indicates that the model is not overfitting and is generalizing well to unseen data. The small differences in RMSE, MSE, MAE, and R^2 between the training and testing datasets suggest that the model is consistent in its predictions across both datasets. However, considering the R^2 values of (19.535% and 19.589%) indicate that model explains only a small portion of the variance, which might indicate that the model may not capture all the underlying patterns of the data, leaving room for improvement.

The regression tree was visualized to provide insights into the decision-making process. The tree diagram shows how the model splits the data based on feature values, with each node representing a decision point and each leaf node representing a predicted value for fico_range_avg. This visualization helps interpret the model and understand the importance of each feature in predicting the target variable.

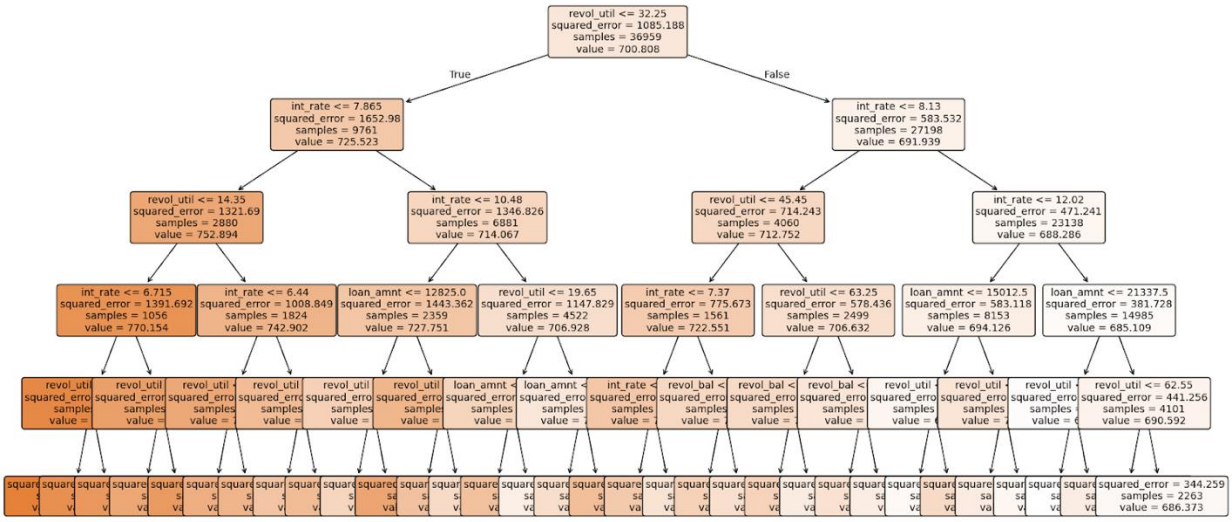


Figure 3. Regression Tree

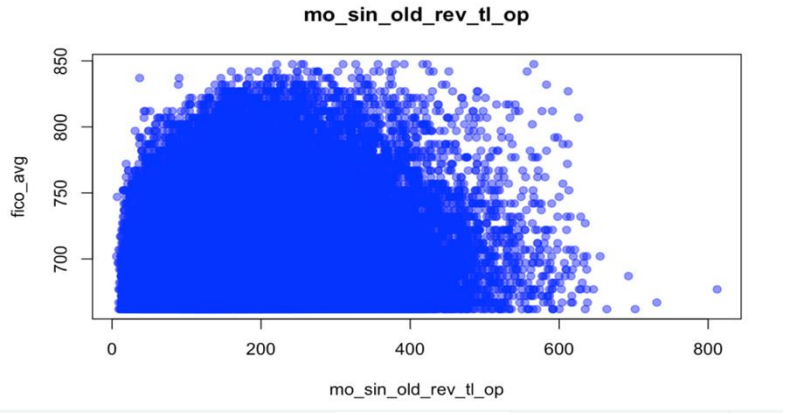


Figure 3:

Higher Order Model

Before developing our initial model, we performed additional data preparation to improve its performance. First, we narrowed down the dataset, from 1 million rows to a sample size of 100,000 rows, simply to successfully run the model on our computers. To eliminate multicollinearity in the variables, we removed highly correlated variables using the Variance Inflation Factor (VIF), eliminating those with a VIF of 5 or higher. This step ensured that each predictor contributed unique information to the model, leading to clearer insights and a more robust analysis. Next, because the dataset was unstandardized and contained significant variance within variables, we applied the Box-Cox transformation to stabilize the variance and normalize the data distribution. With these preprocessing steps completed, we split the data into 80% training and 20% testing to test the accuracy of the model, next we proceeded to build the higher order regression model.

When running the LASSO model, we ran the model with 10 cross folds for the data to keep preforming on unseen data. This means the dataset is split into 10 parts, and the model is trained and validated 10 times, each time using a different fold as the validation set. This helps prevent overfitting and provides a reliable estimate of the model's performance on unseen data, which we struggled a lot with when performing linear regression. The best variables were then spit out, and we tested the variables to see the performance on the test dataset. Which rendered us the RMSE of 22.71, telling us on average, the model's predictions deviate from the actual credit scores by about 22.71 points. Given that the scores range from 662 to 847.5, a span of 186 points, the error represents roughly 12% of the total range [**appendix 9**].

Since the data did not meet the linearity assumptions after plotting the residuals, we needed to adjust our approach for a better fit. To address this, we began by plotting each predictor against the dependent variable to visually assess any underlying trends. An example of this is the plot in figure 3 to the right. The plot shows relationship between *Months since oldest revolving account opened* and *FICO_Avg*, you can see that there is a potential probabilistic or any other relationship besides linear, so exploring polynomial or other non-linear fits could be justified.

Once we placed each variable to a polynomial order, we ran the regression model and evaluated each variable at a significance level of 0.01, iteratively adding higher order terms until the variables were insignificant. After refining the model through this process, our final model [**appendix 10**] the non-linear significant terms are as follows:

- *poly(last_pymnt_amnt,4):* The amount of the last payment is modeled with a 4th-degree polynomial, capturing a potential non-linear effect on FICO score.
- *poly(tot_coll_amt,6):* Total collection amount is modeled with a 6th-degree polynomial
- *poly(bc_open_to_buy,4):* Models the open-to-buy credit on a 4th-degree polynomial scale.
- *poly(mo_sin_old_il_acct,4) & poly(mo_sin_old_rev_tl_op,4) & poly(mo_sin_rcnt_rev_tl_op,5):* These terms represent the months since opening various types of accounts, with polynomial degrees chosen to capture nuances in how these durations impact credit scores.
- poly(mths_since_recent_bc,2): Models the months since the most recent bankruptcy or similar event with a quadratic term.
- poly(pct_tl_nvr_dlq,14): A very high-degree polynomial for the percentage of accounts that have never been delinquent, suggesting the possibility of a highly complex relationship.
- *poly(percent_bc_gt_75,2):* Models the percentage of something (perhaps credit utilization or another metric) with a quadratic term.

Using the best model's outcome on the training set, which resulted in an RMSE of 21.45 telling us that on average, the model's predictions deviate from the actual credit scores by about 21.45 points, above or below. While the Adj R^2 is 58.15%, indicating that the model tells us about 58% of the variability of the average credit score. However, predicting on the testing dataset, it predicted similarity with a RMSE of 21.406 and Adj R^2 is 58.55%.

In conclusion, our model has an average prediction error of about 21.406 points on credit scores, approximately 12% relative to the score range. This performance suggests that while the model captures some key trends, its accuracy is likely limited by missing influential factors. Since the dataset focuses exclusively on approved loan applicants, it may lack broader variability and omit critical factors of creditworthiness. Theoretically, variables such as age and years of credit history are important predictors of credit risk because they reflect financial maturity and stability. Including these factors potentially could lead to a more robust and accurate model. Future research might therefore benefit from incorporating these additional predictors to further enhance model performance.

Multinomial Logistic Regression

Before training the model, all continuous features were standardized to ensure uniformity and prevent dominance by features with larger magnitudes numeric features were also adjusted so that their average value is 0 and their spread (standard deviation) is 1. This ensures all features contribute equally to the model and helps the training process work smoothly. The categorical variables were dummy coded to allow for proper inclusion in the model. To address class imbalance, the SMOTE technique was applied to the training dataset, which generated synthetic samples for underrepresented classes to improve model generalization. The model was then trained using the K-fold cross validation. It helps to evaluate the performance of a model by splitting the dataset into K subsets (folds). The number of folds used in this model is 3 (K=3). The more common or popular number of folds for K-fold cross-validation is k=5 and k=10. However, the larger the K value the greater the computation time. For the model, K=3 was

selected for this reason along with the computation power of the machine being used. The model has an accuracy rate of 99.58% [**Appendix 5].** This indicates that the model correctly classifies approximately 99.58% of all test cases. The model has high sensitivity values (ranging from 0.9742 for Class G to 0.9992 for Class A) which indicates that the model is good at identifying true positives for each class. It also has a high specificity value (ranging from 0.9881 for Class E to 0.99 for Class F) showing that the model is also good at identifying true negatives. The precision values are high across all classes, indicating that when the model predicts a class, it is usually correct. The F1 score for the model is 0.985 or 98.5%. All F1 scores are in the 90s range signifying that the model performs well across all the classes.

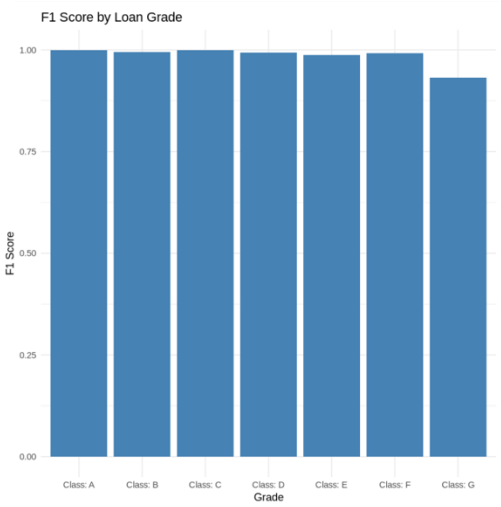


Figure 4. F1 Score by loan Grade

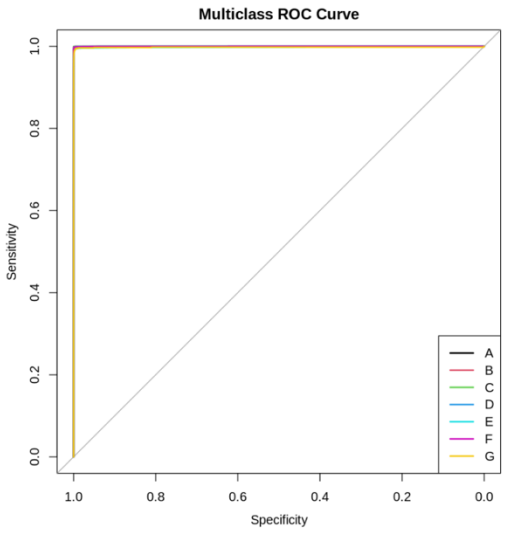


Figure 5. Multiclass ROC Curve

Support Vector Machines

The model was trained using cross-validation of 10 folds, increasing the hyperparameter C which sets a misclassification threshold for improved model generalization. Our model performed best with c = 1.0 both times. To handle the class imbalance, we used synthetic minority over-sampling technique on the smaller classes. When applying principal component analysis selecting the top 40 features from our dataset it achieved an overall accuracy of 84.82% with a F1-score of 69.63%. On the other hand, keeping all features, we obtained an accuracy score of 89.57% and an F1-score of 78.82% [Appendix 7,8]. We notice a performance gain without PCA which suggests that reducing the dimensionality leads to information loss, which in result leads to better generalization. We see an increase of approximately ~ 10% in the ability to distinguish between classes given the higher F1 score, especially in our imbalanced dataset.

However, we do observe that misclassification is more prone to certain classes. It displays overfitting or in other words bias to majority classes where larger classes (0,1,2) exhibit better performance than smaller classes (5,6) [Figure 7].

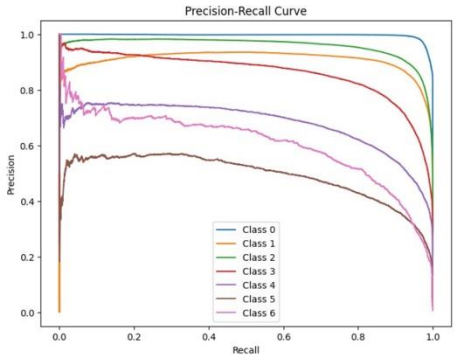


Figure 6. PR Cure with PCA

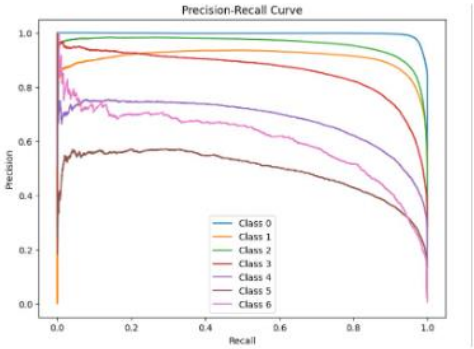


Figure 7. PR Curve (All)

Final Model Performance Metrics

The table below represents the prediction performance on test splits:

*Regression Models**

Model	RMSE	Adj-R^2
Regression Tree	25.008	19.56%
Higher Order Linear Regression	21.406	58.55%

*Classification Models**

	Accuracy (PCA)	Accuracy (All)
Logistic Regression	-	99.58%
SVM	84.82%	89.52%

Conclusion

Looking at our results we see a start performance difference between our regression methods and their classification counterparts. Various reasons can account for this such as, 1. the loan application *[grade]* is evaluated directly by LendingClub.com the financial vendor by their inhouse evaluation metrics, while loan applicant [FICO_Score] is evaluated by external financial institutions such as banks etc. that have a different set of parameters than the ones in our dataset. 2. The regression models utilized in our report might not capture the entire relationship between the [FICO_Score]. To account for this ensemble methods might prove to be very useful in combining different models to boost the generalization ability and reduce the mean error.

The higher-order linear regression with LASSO regularization achieved good performance, yielding an RMSE of 21.406 and explaining 58.55% of the variance (Adj. R^2). The regression tree, while interpretable, had higher error (RMSE: 25.008) and lower explanatory power (Adj. R^2: 19.59%), indicating limitations in capturing complex relationships. Multinomial logistic regression demonstrated exceptional accuracy (99.58%), aided by SMOTE for class balancing and standardized feature engineering. SVM, while less accurate (89.57% without PCA), highlighted trade-offs between dimensionality reduction and information loss.

Appendix

Appendix 1

Testing Normality (Chi-Square Test)

```
------------------------------
Chi-Square Goodness-of-Fit Test for: grade

        Chi-squared test for given probabilities

data:  counts
X-squared = 544640, df = 6, p-value < 2.2e-16


------------------------------
Chi-Square Goodness-of-Fit Test for: sub_grade

        Chi-squared test for given probabilities

data:  counts
X-squared = 558851, df = 34, p-value < 2.2e-16


------------------------------
Chi-Square Goodness-of-Fit Test for: home_ownership

        Chi-squared test for given probabilities

data:  counts
X-squared = 949583, df = 4, p-value < 2.2e-16
```

Appendix 2

```
------------------------------
Chi-Square Goodness-of-Fit Test for: pymnt_plan

        Chi-squared test for given probabilities

data:  counts
X-squared = 885768, df = 1, p-value < 2.2e-16


------------------------------
Chi-Square Goodness-of-Fit Test for: initial_list_status

        Chi-squared test for given probabilities

data:  counts
X-squared = 210759, df = 1, p-value < 2.2e-16


------------------------------
Chi-Square Goodness-of-Fit Test for: application_type

        Chi-squared test for given probabilities

data:  counts
X-squared = 708357, df = 1, p-value < 2.2e-16
```

Appendix 3

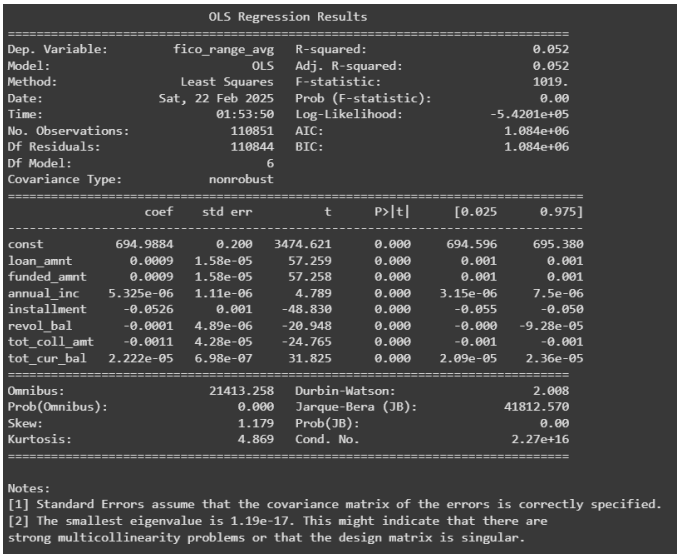
```
------------------------------
Chi-Square Goodness-of-Fit Test for: hardship_flag

        Chi-squared test for given probabilities

data:  counts
X-squared = 885353, df = 1, p-value < 2.2e-16
```

Regression Trees

Appendix 4



OLS Regression Results

Dep. Variable:	fico_range_avg	R-squared:	0.052
Model:	OLS	Adj. R-squared:	0.052
Method:	Least Squares	F-statistic:	1019.
Date:	Sat, 22 Feb 2025	Prob (F-statistic):	0.00
Time:	01:53:50	Log-Likelihood:	-5.4201e+05
No. Observations:	110851	AIC:	1.084e+06
Df Residuals:	110844	BIC:	1.084e+06
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	694.9884	0.200	3474.621	0.000	694.596	695.380
loan_amnt	0.0009	1.58e-05	57.259	0.000	0.001	0.001
funded_amnt	0.0009	1.58e-05	57.258	0.000	0.001	0.001
annual_inc	5.325e-06	1.11e-06	4.789	0.000	3.15e-06	7.5e-06
installment	-0.0526	0.001	-48.830	0.000	-0.055	-0.050
revol_bal	-0.0001	4.89e-06	-20.948	0.000	-0.000	-9.28e-05
tot_coll_amt	-0.0011	4.28e-05	-24.765	0.000	-0.001	-0.001
tot_cur_bal	2.222e-05	6.98e-07	31.825	0.000	2.09e-05	2.36e-05

Omnibus:	21413.258	Durbin-Watson:	2.008
Prob(Omnibus):	0.000	Jarque-Bera (JB):	41812.570
Skew:	1.179	Prob(JB):	0.00
Kurtosis:	4.869	Cond. No.	2.27e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.19e-17. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

Appendix 5

```python
# Print training performance metrics
print("\nTraining Performance:")
print(f"RMSE: {train_rmse:.4f}")
print(f"MSE: {train_mse:.4f}")
print(f"R²: {train_r2:.4f}")
print(f"MAE: {train_mae:.4f}")

# Print testing performance metrics
print("\nTesting Performance:")
print(f"RMSE: {test_rmse:.4f}")
print(f"MSE: {test_mse:.4f}")
print(f"R²: {test_r2:.4f}")
print(f"MAE: {test_mae:.4f}")
```

```
Cross-Validation RMSE: 34.02627572704292

Training Performance:
RMSE: 24.9084
MSE: 620.4272
R²: 0.4316
MAE: 19.4673

Testing Performance:
RMSE: 24.9181
MSE: 620.9139
R²: 0.4225
MAE: 19.5622
```

Multinomial:

Appendix 6

```
]: testing$grade <- as.factor(testing$grade)
   testing$grade<-relevel(testing$grade,ref = "A")
   test_pred <- predict(model,newdata=testing)
   conf_matrix=confusionMatrix(test_pred,testing$grade)
   conf_matrix
```

```
Confusion Matrix and Statistics

          Reference
Prediction     A     B     C     D     E     F     G
         A 37022     1     5    29    39     3     1
         B     1 50580    31    54   129    18    19
         C     8    13 51164     1     1     0     0
         D     9   171    17 23405    36     9     3
         E     1     0     2     0 10365    12     4
         F     0     0     0     0     1  3065     0
         G    11    37     8    13    45     8  1018

Overall Statistics

               Accuracy : 0.9958
                 95% CI : (0.9955, 0.9961)
    No Information Rate : 0.2888
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9946

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E Class: F
Sensitivity            0.9992   0.9956   0.9988   0.9959  0.97636  0.98395
Specificity            0.9994   0.9980   0.9998   0.9984  0.99989  0.99999
Pos Pred Value         0.9979   0.9950   0.9996   0.9896  0.99817  0.99967
Neg Pred Value         0.9998   0.9982   0.9995   0.9994  0.99850  0.99971
Prevalence             0.2089   0.2864   0.2888   0.1325  0.05986  0.01756
Detection Rate         0.2087   0.2852   0.2885   0.1320  0.05844  0.01728
Detection Prevalence   0.2092   0.2866   0.2886   0.1333  0.05855  0.01729
Balanced Accuracy      0.9993   0.9968   0.9993   0.9971  0.98812  0.99197
                     Class: G
Sensitivity          0.974163
Specificity          0.999308
Pos Pred Value       0.892982
Neg Pred Value       0.999847
Prevalence           0.005892
Detection Rate       0.005740
Detection Prevalence 0.006428
Balanced Accuracy    0.986735
```

SVM:

Appendix 7

	precision	recall	f1-score	support
0	0.9444	0.9339	0.9391	37960
1	0.8829	0.8828	0.8829	50821
2	0.8953	0.8301	0.8615	54424
3	0.6999	0.7379	0.7184	21114
4	0.5346	0.6241	0.5759	8823
5	0.3370	0.4318	0.3785	3078
6	0.4031	0.7227	0.5175	1143
accuracy			0.8386	177363
macro avg	0.6710	0.7376	0.6963	177363
weighted avg	0.8482	0.8386	0.8424	177363

Appendix 8

	precision	recall	f1-score	support
0	0.9665	0.9468	0.9566	36931
1	0.9182	0.9310	0.9246	50749
2	0.9152	0.9226	0.9189	51325
3	0.8336	0.8198	0.8266	23455
4	0.7104	0.7052	0.7078	10729
5	0.5806	0.5921	0.5863	3133
6	0.5888	0.6052	0.5969	1041
accuracy			0.8956	177363
macro avg	0.7876	0.7890	0.7882	177363
weighted avg	0.8957	0.8956	0.8956	177363

Higher Order Linear Model:

Appendix 9

```
Residual standard error: 21.48 on 79931 degrees of freedom
Multiple R-squared:  0.5815,    Adjusted R-squared:  0.5811
F-statistic:  1633 on 68 and 79931 DF,  p-value: < 2.2e-16
```

Appendix 10

```
fico_avg ~ grade + home_ownership + delinq_2yrs + inq_last_6mths +
+ initial_list_status + poly(last_pymnt_amnt,4) + collections_12_mths_ex_med +
+     application_type + poly(tot_coll_amt,6) + acc_open_past_24mths +
+     poly(bc_open_to_buy,4) + chargeoff_within_12_mths  + delinq_2yrs +
+     poly(mo_sin_old_il_acct,4) + poly(mo_sin_old_rev_tl_op,4) + poly(mo_sin_rcnt_rev_tl_op,5) +
+     mo_sin_rcnt_tl + mort_acc + poly(mths_since_recent_bc,2) + num_accts_ever_120_pd +
+     num_tl_120dpd_2m + num_tl_90g_dpd_24m + num_tl_op_past_12m +
+     poly(pct_tl_nvr_dlq,14) + poly(percent_bc_gt_75,2) + pub_rec_bankruptcies
```

References

[1] Wackerly, Dennis D., et al. “Analysis of Categorical Data.” Cengage Learning, Belmont, CA, 2002, pp. 713–718.

[2] Powell, Michael J. D. “Radial Basis Functions for Multivariable Interpolation: A Review.” *Algorithms for Approximation*, Clarendon Press, 1987, pp. 143–167.

[3] James, Gareth, et al. “9.3 Support Vector Machines.” Springer, pp. 349–356.

[4] James, Gareth, et al. “4.3.5 Multinomial Logistic Regression s.” Springer, pp. 140–141.

[5] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley.

[6] National Institute of Standards and Technology. (n.d.). Graphical techniques. In NIST/SEMATECH e-Handbook of Statistical Methods. Retrieved February 20, 2025, from https://www.itl.nist.gov/div898/handbook/eda/section3/eda35g.htm

[7] Arize. (n.d.). Kolmogorov-Smirnov Test. Retrieved February 20, 2025, from https://arize.com/blog-course/kolmogorov-smirnov-test/

[8] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: With Applications in R. New York, NY: Springer.

[9] GeeksforGeeks. (n.d.). What is Lasso Regression? Retrieved February 20, 2025, from https://www.geeksforgeeks.org/what-is-lasso-regression/