Predicting Lending Club Loan Default

Apoorva Joshi

Hyunjae Lee

**PROBLEM DEFINITION**

Predicting loan defaults benefits both the borrowers and the investors (lenders). The borrowers can make better financial decisions while the investors can prevent any future financial distress. From the perspective of the borrower, defaulting a loan leads to lower credit scores. This not only prevents him / her from getting loans in the future but also affects the person’s ability to rent, get access to utilities, or even get a job. As for the investor, having multiple borrowers default can lead to the loss of both interest and principal. Predicting the probability of a default can help investors maximize their returns and help identify risky investments. Furthermore, classifying borrowers can help prevent the potential “defaulters” and such classification can be implemented through each borrower’s information (e.g., employment, home ownership status, annual income, etc.).

Machine learning algorithms were born out of pattern recognition. Today they are used in variety of sectors like health care, financial services, and marketing. The powerful predictive capacity of machine learning algorithms is an important reason for their fame. This project demonstrates one out of the many ways in which potential defaulters can be identified as risky before the banks invest in them.

Supervised learning is a branch of machine learning algorithms that utilizes labels obtained from the training set to predict the outcome of a test set. In a supervised classification task, the algorithm classifies the test data based on the test attributes (i.e., information on the borrower and loan) and the model (e.g., logistic regression model) learned from the training set.

Banks have large datasets with detailed information on the past loan applications. This information contains both the defaulted and successful loan applications that can be used as the label for the training set. Consequently, the model trained with past applications would be able to predict the outcome of incoming new applications.

There is a variety of classification algorithms each with its own set of advantages and disadvantages [1]. Some of the classification algorithms are Naïve-Bayes, Logistic Regression, Decision Trees, Random Forests, Support Vector Machine (SVM), etc. Choosing any one of these algorithms involves a process of identifying a trade-off between complexity and interpretability that suits the business case. For example, Logistic Regression models are more interpretable when compared to SVM, but SVM is generally more accurate and can work with data that is not linearly separable in the feature space. On the other hand, Decision Trees can handle feature interactions and are non-parametric, which relieves the burden of handling outliers. At last, Ensemble methods like Random Forests provide high accuracy, require less tuning and are more interpretable than SVM. This project will apply some of the algorithms mentioned above to predict the status of loan applications.

**General Terms**

Machine Learning, Supervised Classification, Support Vector Machines (SVM), Decision Trees, Logistic Regression, Radial Kernel, Random Forests, Specificity, Sensitivity.

# DATA DESCRIPTION

The dataset consists of all the loans issued by a company called Lending Club from 2012 to 2015 and is available from Kaggle.com. The point data has 75 features and approximately 880,000 instances, which comes in a single CSV file of size 420 MB. Some of the features that are intuitively important are:

term: Number of payments, in months, on the loan (36 or 60)

annualInc: Self-reported annual income

tot\_cur\_bal: Total current balance of all accounts

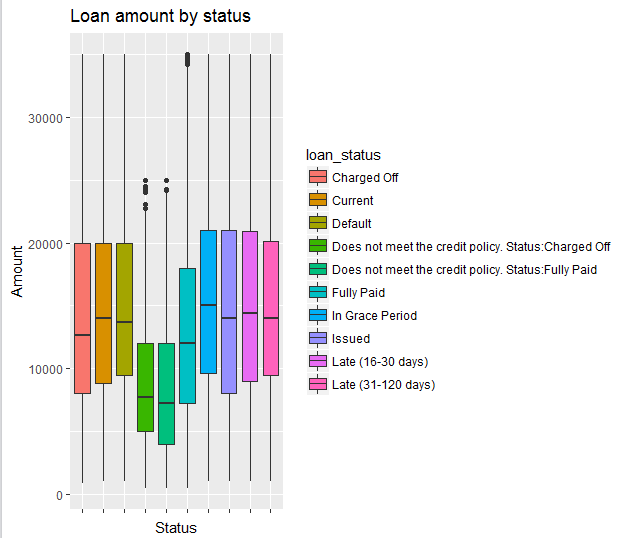
emp\_length: Employment length in years (0 ~ 10). 0 means less than one year and 10 means ten or more years.

int\_rate: Rate of Interest

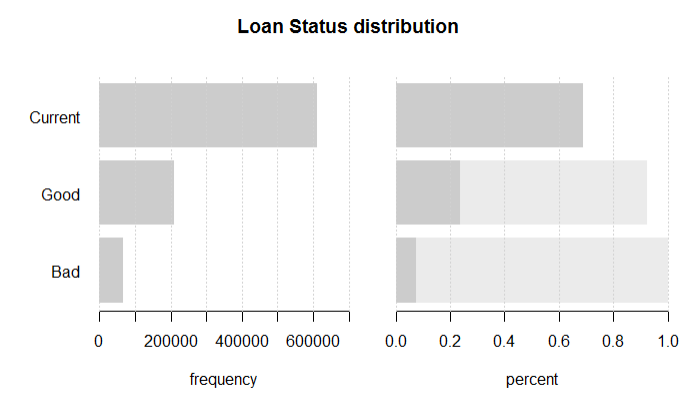
loan\_status: Status of the loan application

## EXPLORATORY DATA ANALYSIS

The outcome variable is loan\_status. It has multiple levels like “Fully Paid”, “Default”, “Current”, etc. After carefully observing factor levels in the outcome variable loan\_status, the existing ten levels are grouped into three levels for the sake of convenient classification: “Good”, “Bad”, and “Current”. “Current” includes all instances that are “Issued” or “Current”. “Good” includes all instances such as “Fully Paid”, “Charged Off”, etc. “Bad” includes all instances with values like “In Grace Period” and “Late”. Figure 1 shows the distribution of factor levels of loan\_status prior to the merge.

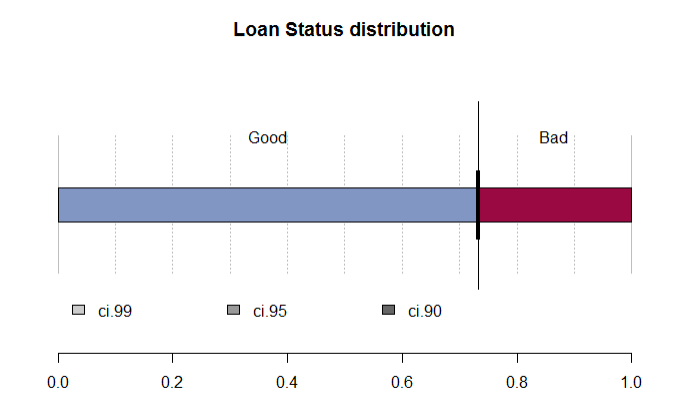


**Figure 1. Distribution of original loan\_status**

****

**Figure 2. Distribution of merged Loan Status**

Figure 2 addresses the next challenge in analysis, where the “Current” loans consist of an overwhelming majority even though they are literally in progress and cannot add value to the prediction. All “Current” loans are removed from the dataset. Distribution of loan\_status after removing the records is given below:



**Figure 3. Distribution of Loan Status with results**

Following are the features chosen intuitively to gain more insight into the data: loan amount, interest rate, and grade.

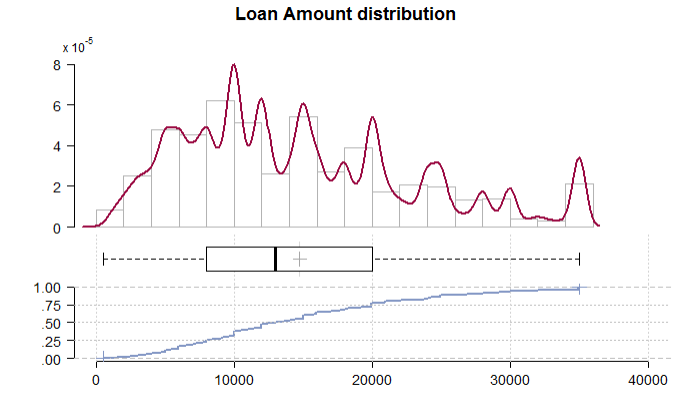


Figure 4. Distribution of Loan Amount

The listed loan amount shows a fine distribution with a slight skew to the right. The median amount lies somewhere near $12,000 while few loans applied for large sums as much as $35,000.

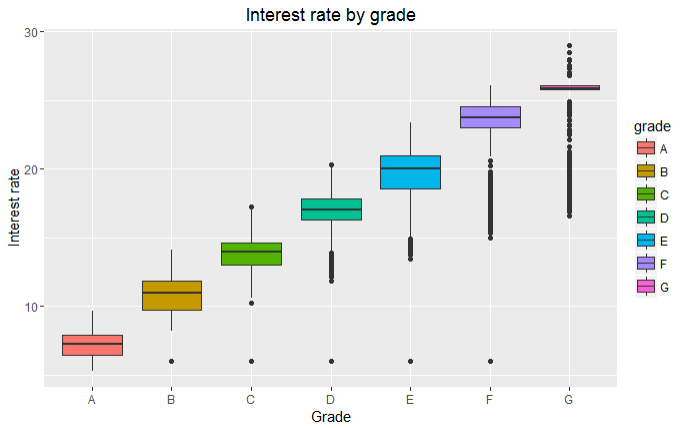


Figure 5. Interest Rate by Grade

Figure 5 shows how the interest rate varies according to the grade assigned by the company to the loan application. The interest rate increases as we move from grade A to grade G.

# DATA PREPARARTION

The data requires a significant amount of preprocessing before feature selection. The dataset was imbalanced by nature and required undersampling of the majority class. Some of the columns had more than 90% of its values missing while some had as less as 7%. Some of the algorithms required scaling of the variables to reduce the influence of variables with large values. Below is a systematic description of the data cleaning process.

## Data Quality

Features like loan\_status, verification\_status, and home\_ownership are factor variables. To ensure consistent factor levels from the factor variables, it is important checking case sensitivity, spelling, and special characters. Dates were converted into numeric values, as difference in months from an arbitrarily chosen date (i.e., January 1, 2010).

Another aspect of factors is that sometimes the attribute is represented by too many levels of factor (e.g., emp\_title). Because such attributes cannot be translated into other data types, they are neglected in the analysis.

## Accuracy of the data and Outlier analysis

There are two aspects of accuracy: semantic and syntactic. Fortunately, the majority of attributes like interest rates and total revolving balances had numeric values.

Numeric attributes are then checked for outliers using distribution graphs; outliers indicate either exceptional values or wrong recordings. Since the dataset is large (277,140 records after merging and dropping “Current”) and the outliers are only a small portion of the set, they are negligible.

## Handling missing values

Some of the variables like open\_il\_12m, total\_bal\_il and inq\_fi have more than 50% of the values missing. This could be because either that particular piece of information was not collected in the application process or those columns did not apply to majority of the instances (i.e., Missing Completely at Random). In total, there are 21 columns with that many values missing and thus dropped.

Other variables like annual\_inc, pub\_rec (Number of derogatory public records), and open\_acc (The number of open credit lines in the borrower's credit file) are Missing at Random. Missing values in such features are filled with the mean of remaining values.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Conference’10*, Month 1–2, 2010, City, State, Country.

Copyright 2010 ACM 1-58113-000-0/00/0010 …$15.00.

## Data Normalization

It is essential to factorize the numeric variables and transform the categorical variables into dummy variables when employing SVM as a classification tool. Some of the features like annual income and total revolving balance have very high absolute values as compared to other numeric features like interest rate.

Hence, the features are normalized by mean and variance to adjust its influence on the model.

## Handling unbalanced data

When the dataset is unbalanced, as shown in Figure 3, classifiers are biased towards the majority class because their loss functions optimize the error rate. They do not take the distribution of the data into consideration and so the algorithm generates a trivial classifier that classifies all instances into the majority class, which in this case would be “Good”.

Given a large enough dataset, the majority class is under-sampled to balance out the distribution of the data. Distribution of the outcome variable loan\_status after the adjustment is given below:

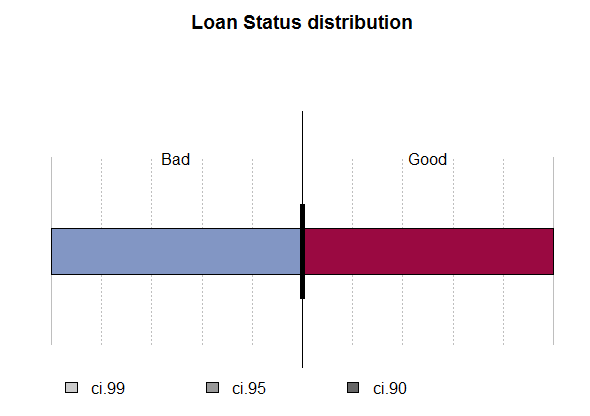


Figure 6. Distribution of Loan Status after under-sampling

# METHODOLOGY

## Feature Selection

Choosing relevant features plays an important role in the machine learning algorithm. Feature Selection is a process of dropping irrelevant features that do not contribute to the accuracy of the model and even hinder the model. It makes the model simpler and easier to understand.

First, based on the context, features like url, loanID and memberID do not provide any valuable information about the loan. In addition, the attributes ending with “\_inv” are redundant amounts set for investors. These features are dropped from the dataset before the selection process.

There are many ways to select relevant features like filter methods, embedded methods, information gain, etc. Information Gain of an attribute tells how much information it provides about the target classification variable and is thus ideal for a classification problem. Also, techniques like forward step and backward step were consuming more than 8 hours since the full model had the outcome variable regressed on 42 predictor variables. Hence we chose information gain as a method to select the most relevant variables. After the data cleaning process, there are 42 variables and the top ten attributes are selected among them. The table below shows the top five attributes with its information gain:

|  |  |
| --- | --- |
| **Attribute** | **Information gain** |
| total\_rec\_prncp | 0.62314941 |
| last\_pymnt\_amnt | 0.38712412 |
| total\_pymnt | 0.16988318 |
| recoveries | 0.14750181 |
| collection\_recovery\_fee | 0.13936011 |

**Table 1. Information Gain**

The task is a supervised classification task for which we are going to employ Logistic Regression, Decision Trees and SVM to classify the test instances. We have chosen to start with Logistic Regression as our baseline model.

## Logistic Regression

Logistic Regression is a highly interpretable model since it follows a probabilistic framework. It does not assume linearity because it applies non-linear log transformation to the predicted odds ratio.

Since the Logistic Regression assumes that P(Y=1) is the probability of the event occurring, it requires the response variable in a binary form. A separate indicator variable has been hard-coded to reflect the binary nature under the name “loan\_status\_ind” where 1 = “Good”.

To ensure accurate prediction, the sample dataset is kept at large with 1:1 split between training and test set with about 67,000 records in each set. Binary Logistic Regression assumes that there is no high-multicollinearity between the variables. The maximum correlation coefficient threshold value is set at 0.90 and the correlation coefficient among the independent variables is less than our defined threshold.

The outcome variable loan\_status\_ind was regressed against the ten features with the highest information gain. A logistic regression model was developed with following performance metrics:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Accuracy** | **Specificity** | **Sensitivity** | **AUC** | **Time taken** |
| 0.90522 | 0.94414 | 0.87215 | 0.90546 | 1.14 s |

Table 2. Logistic Regression Performance

This model is our baseline model. The reason behind choosing the evaluation criteria specified above has been discussed in the Evaluation and Results section.

## Decision Trees

Logistic Regression models search for a linear decision boundary in the feature space. In contrary, decision tree model partitions the feature space into half spaces using axis-aligned linear decision boundaries. This is a distinct advantage of Decision Trees over Logistic Regression. Decision trees also perform feature selection implicitly based on the amount of information an attribute provides and do not require any standardization of the input variables. They are also insensitive to outliers since the splitting happens based on the proportion of samples between the split ranges and not on absolute values. Due to these reasons, we have decided to build a Decision Tree for this classification task. We built three Decision Tree models with the first one being baseline model for the other two. The Baseline model grows without limitation and the other two models are pre-pruned for optimal building time & accuracy.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Specificity** | **Sensitivity** | **AUC** | **Time taken** |
| Baseline | 0.98234 | 0.98268 | 0.98199 | 0.98233 | 6.12 s |
| Pruned by minimum per node | 0.95889 | 0.96880 | 0.94928 | 0.95894 | 6.29 s |
| Pruned by depth | 0.90293 | 0.96276 | 0.85623 | 0.90329 | 3.53 s |

**Table 3. Decision Tree Performance**

A major disadvantage of Decision Trees is that they tend to overfit the data. But we have handled this by pruning the decision tree to its optimal size. The Complexity Parameter (CP) can be used for pruning the decision tree. If the cost of adding another value to the decision tree is greater than the value of CP, then the tree stops growing. The desired size of the tree would be the one that minimizes the cross-validation error; the value of cp corresponding to the minimum cross validation error is chosen for pruning the tree.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Specificity** | **Sensitivity** | **AUC** | **Time taken** |
| Baseline | 0.98416 | 0.98263 | 0.98572 | 0.98415 | 0.13 s |
| Pruned by minimum per node | 0.95889 | 0.96880 | 0.94928 | 0.95894 | 0.02 s |
| Pruned by depth | 0.90293 | 0.96276 | 0.85623 | 0.90329 | 0.02 s |

**Table 4. Decision Tree Performance, after pruning by CP with minimum error**

The base model shows a slight improvement with extra 0.13 seconds spent on pruning the decision model. Pruning by CP did not change much of the two latter models.

## Support Vector Machine

SVM is a classifier that is highly accurate but requires significant amount of parameter tuning and computational power. Even though they are not as interpretable as Decision Trees or Logistic Regression, they give good performance in comparison. A huge advantage of using kernel functions is that it allows the kernels to operate in a high-dimensional feature space by computing the inner products between the images of all the pairs of data in the feature space. This operation is cheaper than the process of computing the coordinates explicitly.

There are few aspects that need to be handled to ensure that the SVM is effective. It requires that the features should be normalized to reduce the effects of features with large absolute values. Also, results of an SVM are highly sensitive to the hyper-parameters and using k-fold cross-validation along with a hyper-parameter grid search is an important step before running the SVM model. Choosing the appropriate kernel is also an important part of applying SVM effectively. SVM has an advantage over other methods as it allows the algorithm to be fully expressed in terms of kernels without having to specify the feature space information.

Linear, Polynomial, and Radial SVMs are chosen and tuned. We will evaluate and choose the model based on the performance metrics given in the table.

### Linear SVM

The first SVM model that we start with uses a Linear Kernel. It runs much faster as compared to an SVM with a Polynomial or Radial function.

The first model was built without any parameter tuning. The value of the hyperparameter cost was chosen as 10. This parameter corresponds to the soft margin cost function which controls the influence of each support vector. Large values for C implies that we are penalizing the SVM significantly for having data points within the margin. A lower value of C gives higher error on the training set, but finds a larger margin that might be more robust. SVM model performs its best when the value of the hyperparameter C is chosen using cross-validation. Using 10-fold cross validation and range of values for cost falling in the range [2 -4, 2 -3…24], we obtained slightly better accuracy than the default model. The value for cost chosen from the cross-validation process was 4.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Specificity** | **Sensitivity** | **AUC** |
| Without parameter tuning | 0.969835 | 0.958723 | 0.951370 | 0.9528 |
| With parameter tuning | 0.9780777 | 0.9748033 | 0.9813864 | 0.9781 |

**Table 5. Linear SVM Performance**

The next kernel that could be tried is the Polynomial kernel. The advantage of using a polynomial kernel over the linear SVM is that it separates non-linear regions after transforming to squared kernel space.

### Polynomial SVM

For considering the interaction between the features, we decided to use a polynomial kernel. We performed 10 fold cross-validation to choose the best value for cost. The range of possible values given to the function was [2 -4, 2 -3…24]. We chose the degree to be 2 and got the below performance statistics.

|  |  |  |  |
| --- | --- | --- | --- |
| **Accuracy** | **Specificity** | **Sensitivity** | **AUC** |
| 0.9655912 | 0.9537001 | 0.9776066 | 0.9657 |

**Table 6. Polynomial SVM Performance**

The performance of Linear SVM after parameter tuning was better than the performance of a SVM with a polynomial kernel. A disadvantage of using SVM with a polynomial kernel is that it tends to overfit the data which may lead to poor generalization. SVM with a Radial Kernel can generalize better than a polynomial kernel.

### SVM with a Radial Kernel

This kernel has two hyperparameters namely γ and C. γ is technically not a SVM parameter but the parameter of the kernel. This parameter controls the error due to bias and variance in the model. If the value of γ is too large, the model can overfit and is more prone to low bias and high variance. If the value of γ is too small, then it can lead to high bias and low variance in the model.

The best combination of C and γ is chosen by a hyper-parameter grid search using cross-validation. We used 10-fold cross with the range of value for C being [2 -4, 2 -3…24] and γ being [2 -3, 2 -2…23]. The final chosen value for C was 2 and was 0.5 for γ. The performance extracted from this model is given in the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Accuracy** | **Specificity** | **Sensitivity** | **AUC** |
| 0.979071 | 0.980762 | 0.977357 | 0.9791 |

**Table 7. SVM with a Radial Kernel Performance**

The amount of computational time and resources required for SVMs with complex kernels especially for selection of hyperparameters using cross-validation is significant. This was a major disadvantage of using this technique.

# EVALUATION AND RESULTS

There are many performance metrics available for comparing the performance of different models. Some of them are accuracy, sensitivity, specificity, precision, recall, F1 score, ROC curve etc. Accuracy simply measures the number of individuals who were correctly classified as True Positives and True Negatives. This is a very useful measure, but it is important to be more specific and check whether we are getting a True Positive or True Negative correct. We will use the two measures Specificity and Sensitivity and establish a trade-off between the two.

Sensitivity = true positives/ (true positives + false negatives)

Specificity = true negatives/ (true negatives + false positives)

Sensitivity will answer questions like :

Given that a person is going to default, how often will the test be positive (true positive rate)? If a test is highly sensitive and the test result is negative, we can be certain that the loan borrower will not default.

Sensitivity will answer questions like :

Given that a person is not going to default, how often will the test be negative (true negative rate)? If the test result for a highly specific test is positive, we can be certain that the loan borrower will default [5].

Ideally we would like to maximize these quantities, but usually there is a trade-off. We need to choose a threshold probability that will turn the probability model into a classification model. As we decrease the threshold probability, we tend to increase the sensitivity of the model while the specificity decreases. The threshold that we chose was 0.6.

When the true positive rate is plotted against the true negative rate, the resulting curve is called as the ROC curve. Another metric that we have considered is the AUC which is the area under the ROC curve. The advantage of using the AUC curve is that it considers all possible thresholds. Various thresholds result in various true positive and true negative rates.

We also had the option to use balanced accuracy as a measure of performance, but since we had balanced both the training and test datasets, we did not use it. If we had trained our model on a balanced dataset and tested them on an imbalanced dataset, then we would have used balanced accuracy [4].

After assessing the performances of all the models that were tried out, the baseline model (cp pruned) that we had built using the Decision Tree algorithm gave maximum values for accuracy, sensitivity and specificity. The performance metrics for this model are as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Specificity** | **Sensitivity** | **AUC** | **Time taken** |
| Baseline | 0.98416 | 0.98263 | 0.98572 | 0.98415 | 0.13 s |

Even though we had employed more complex models like Radial SVM, a simple Decision Tree algorithm seemed to work very well for this dataset. The reason behind this was that we were working with point data that is characterized by a set of attributes with values for each attribute. Also, the outcome variable had a finite set of values. This is important because when we use Decision trees, we want to arrive at some leaf in the tree while making the decision. Having finite number of outcome values makes it easy to label the leaf as a value.

In the near future, there are more complex algorithms that could be employed to solve the same problem. Random Forest would be the next choice to employ. It adds an additional layer of randomness to bagging. Unlike standard trees, in Random Forests each node is split using the best among a subset of predictors randomly chosen at that node. It is known to be more powerful than SVM and requires only two parameters towards which it is not that sensitive.

# FUTURE WORK

The classification models provide accurate analysis on investment outcomes without any compromise on the computation complexity or data quality.

More information on the financial decisions and the implications in the ten original levels of loan\_status can enrich the classification model. While the current model in this project merged all cases into two macro categories, (i.e., “Good” and “Bad”) for the classification convenience, it certainly misses the information provided by the variety of responses.

Another possibility for the predictive model is to utilize other sets of information that are available publicly. For example, the original data includes the first three digits of the zip code where the borrower resides. This bit of information can be integrated with the annual report from the Internal Revenue Service (IRS) that contains income and tax corresponding to each geographical units. Knowing more about the borrower (e.g., Is the borrower’s income level below or above the average income in his / her neighborhood?) can be used to further push up the model’s accuracy.

In the future when data is collected under a wider time window, the dataset can be separated into annual subgroups, where a year’s data can be used to predict the outcomes of loans in the following year. Building models annually would allow capturing the slowly changing trend.

# REFERENCES

1. Kevin Tsai, Sivagami Ramiah, Sudhanshu Singh, Peer Lending Risk Predictor
2. Shunpo Chang, Simon Dae-oong, Genki Kondo ,Predicting Default Risk of Lending Club Loans
3. Bhatnagar Pujun, Chow Nick, Lai Max, Demystifying the workings of Lending Club
4. Kay H. Brodersen, Cheng Soon Ong , Klaas E. Stephan and Joachim M. Buhmann, The balanced accuracy and its posterior distribution
5. Understanding and using sensitivity, specificity and predictive values, [Rajul Parikh](https://www.ncbi.nlm.nih.gov/pubmed/?term=Parikh%20R%5BAuthor%5D&cauthor=true&cauthor_uid=18158403), [Annie Mathai](https://www.ncbi.nlm.nih.gov/pubmed/?term=Mathai%20A%5BAuthor%5D&cauthor=true&cauthor_uid=18158403), [Shefali Parikh](https://www.ncbi.nlm.nih.gov/pubmed/?term=Parikh%20S%5BAuthor%5D&cauthor=true&cauthor_uid=18158403), [G Chandra Sekhar](https://www.ncbi.nlm.nih.gov/pubmed/?term=Chandra%20Sekhar%20G%5BAuthor%5D&cauthor=true&cauthor_uid=18158403) and [Ravi Thomas](https://www.ncbi.nlm.nih.gov/pubmed/?term=Thomas%20R%5BAuthor%5D&cauthor=true&cauthor_uid=18158403)
6. Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel, S. Sathiya Keerthi, Chih-Jen Lin