Capstone 1 : Final Report
Lending club loan data
Matthew Gudorf

1. Introduction

The financial loss associated with loans being "charged-off" is incredibly large. According to the FDIC the amount of loans and leases merely past due in 2018 is on the order of 95 billion dollars. This provides motivation to create better metrics and criteria for deciding whether a loan should be provided to potential borrowers. In addition to machine learning metrics, the amount of money lost to charged-off loans or delinquency provide a quantitative metric for future evaluation of the model. That is to say, the model can be validated by seeing a decrease in the amount of capital lost over time and the amount of delinquency. This is perhaps one of the most common types of analysis and applications of machine learning methods but as with most cases, deciding on specific algorithms and parameters to use is mostly black magic. This project provides a comparison between different models based on predictive power and computational accuracy. In addition, this project attempts to find a smarter and more efficient criterion than scanning through the parameter values to find the best model. The process of finding the best model is typically unique to each dataset, hence the lack of generalized results.

2. Data cleaning

The goal of this project is to create two models; the first predicts whether a loan will default, the second predicts the potential recoveries therefrom. The first step for any data science project is to clean and wrangle the dataset, which is what this document details.

Upon importation into a Jupyter notebook, a warning was displayed indicating that the data contained variables of mixed type (e.g. a combination of python lists and floats in one data feature). This is an issue that could be handled by interpreting the imported data as strings, for instance. There are a number of such properties that need to be taken into account before the prediction models can be created. Specifically, my belief is that the important data cleaning decisions to make are: how to handle missing values, redundant information (a feature column identically equal to a single value), multiple types of data. Care must be taken when manipulating the data in these ways, as to not ruin the generalizability of any models.

2.1 Handling missing values

Firstly, for the missing data values there are a number of options. For instance the missing values can be imputed by a specified method (replace the missing value with the mean of the data feature, replace with a constant value, etc.), the samples containing missing values can be dropped from the data set, or if the feature is a categorical variable, “missing” can become its own category. Imputation is not acceptable as it will ultimately alter the summary statistics of the data, excluding the statistic maintained by the replacement rule. That is to say, if the missing values are replaced with the mean then the mean will be maintained, but other summary statistics will not be maintained. An alternative would be to just drop all samples with missing values; unfortunately the missing values are distributed in such a manner such that if the samples with missing values are removed from the dataset then there are exactly zero samples left to work with.

While the distribution of missing values is not optimal, I developed a strategy to work around them. There are a number of features (columns) wherein the vast majority of values are missing. These are discarded if the percentage of missing values is above a threshold value. When placed in decreasing order, the distribution of these percentages in figure 1 has multiple plateaus. This leads to an interesting choice of where to put the threshold. While it is still arbitrary, the essential choice is whether to include the “plateau” that occurs around 40%. I have not decided upon the best choice but it is evident that these values correspond to a different time period than the features with a smaller proportion of missing values.

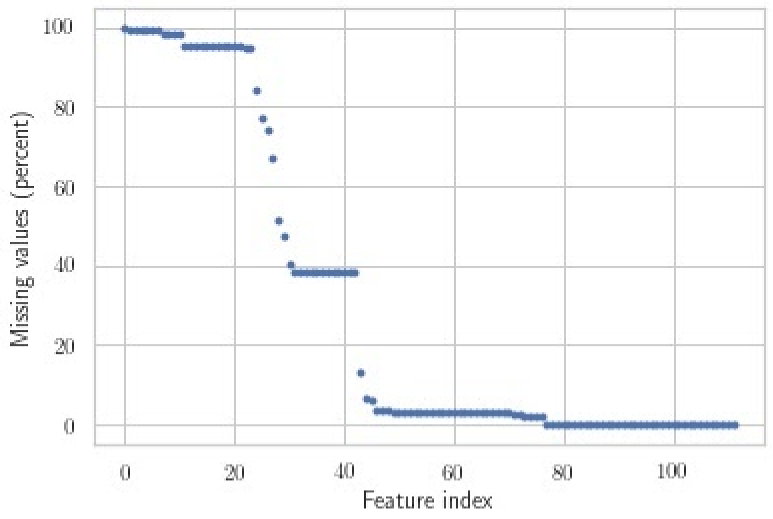


Figure 1. The percentage of missing values in each feature, in decreasing order.

The main strategy I employ for the current data used in the two-step modeling procedure is to first replace all missing values in categorical feature data with a new category labeled

“Missing”. Next, I remove all features with a proportion of missing values greater than 30%. Lastly, the remaining samples that have missing (numerical) values are removed from the dataset, as it is hard to motivate any particular imputation. The main trade-off in this process is that I keep more in exchange for fewer features.

2.2 Handling pathological features

Once these measures to account for missing values have been performed, I move to the other general measures that are performed to process the data to be ready for each stage of the modeling process. For example, there are some features which contain a single value. As it stands, any partition of the data into training and testing data quite obviously contains only one value. This is unlikely to be an example of sampling bias due to the number of samples; it is simply a systematic issue with how the data is recorded. For example, there is a feature named “policy_code” which only takes a value of 1. The lack of utility stems from the fact that the training domain being a single value has the implication that it won’t know how to handle differing values. There are also features which have a highly imbalanced distribution, namely, the vast majority of samples take one value. This case is not as well motivated as the prior case, but still I believe it is a valid measure to prune these features from the dataset.

One last case of hard to manage data features are categorical features which are idiosyncratic in nature. For example, one such feature would be the description for the reason for the loan provided by the borrower. Excluding certain generic categories such as “debt consolidation”, there are a very large number of unique responses and hence categories that would need to be encoded. One way of handling this would be to bin all unique responses into an “Other” category. The problem with this approach is that the cutoff would be arbitrary, essential defined by how many categories I deem important. I deemed this too dubious and so I opted for removing these types of categorical features from the dataset, such that they would not be used in the modeling procedure.

2.3 Dependent variable preparation

The last component of the data wrangling is processing the data to adhere to our modeling hypotheses. For instance, the loan outcome classification portion of this project wants to predict whether or not a loan will be charged off by its maturity date. Therefore, I need to first subset the loans which have matured, and then choose a way of handling the possible outcomes of each loan (recorded in “loan_status”). For the maturity subselection, the data contains information pertaining to loans of three year terms and five year terms, up to December 1st, 2018. Therefore, I took the subset of loans whose issuance date plus term was equal to or less than this date. It could be argued that there should be two models, one for each loan term, as they might represent distinct populations. While I did process the data using this belief, I did not end up creating separate models for the two populations.

3. Exploratory Data Analysis

3.1 General exploration and data visualization

The dataset containing loan information that I am using (the Lending Club loan dataset from (https://www.kaggle.com/wordsforthewise/lending-club), contains categorical variables in the form of strings, discrete numerical variables, and continuous numerical variables. The data can also be segmented into information describing the borrower (geographical location, income, etc.) and information about the loan (principal amount, interest rate, etc.). The goal of this document is to develop an understanding of the data. There are also features that are time dependent and can be naturally visualized as a time series. One example of a time dependent feature is the date when each loan was issued. By aggregating by month, we can see that there is an upwards trend in the number of issued loans. As the number of loans issued increases, it becomes more important to prevent loan defaults, as the amount of capital they represent also increases (assuming a constant proportion of issued loans default over time).

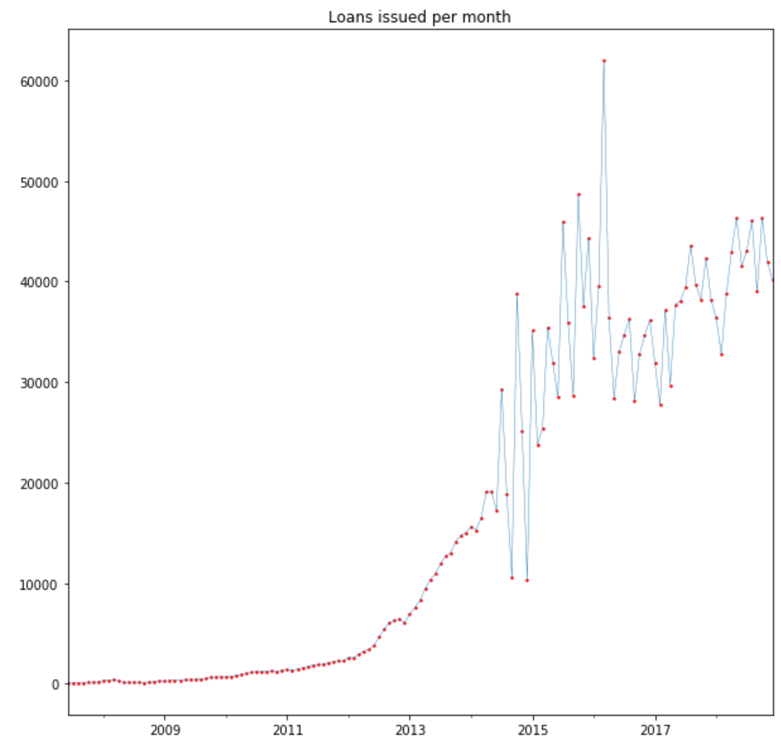


Figure 2. Loans issued per month over time.

This is only a figure to give a sense of how the loans have grown up to the present moment. Aggregating by the (top three) statuses of the loan shows the time distribution of when fully paid, current and charged off loans. All of the loans have terms of three or five years so it's understandable that the approximate maximum of the fully paid loans and charged off loans occurs in the same range, three to five years ago. (The most current date in the data set is December 1st 2018).

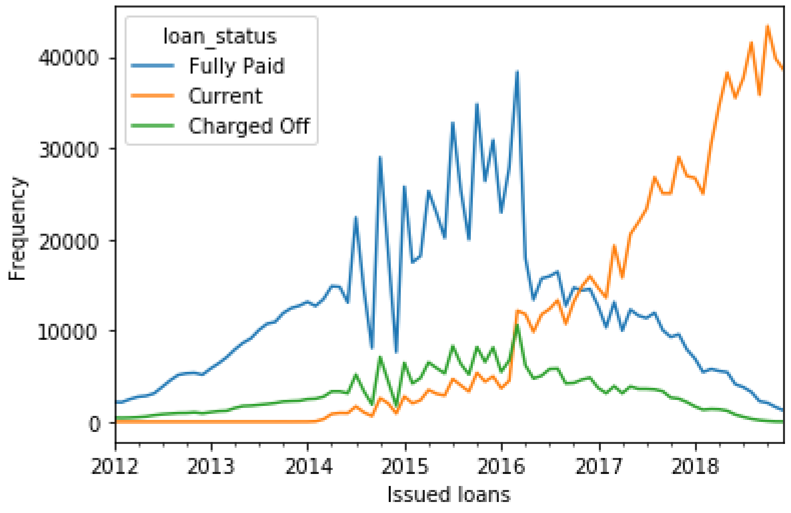


Figure 3. Loan status (three main categories) time series

Continuing with the data exploration, a unique and interesting quantity is the distribution of loan amounts; it really shows the affect of human psychology by how the distribution manifests.

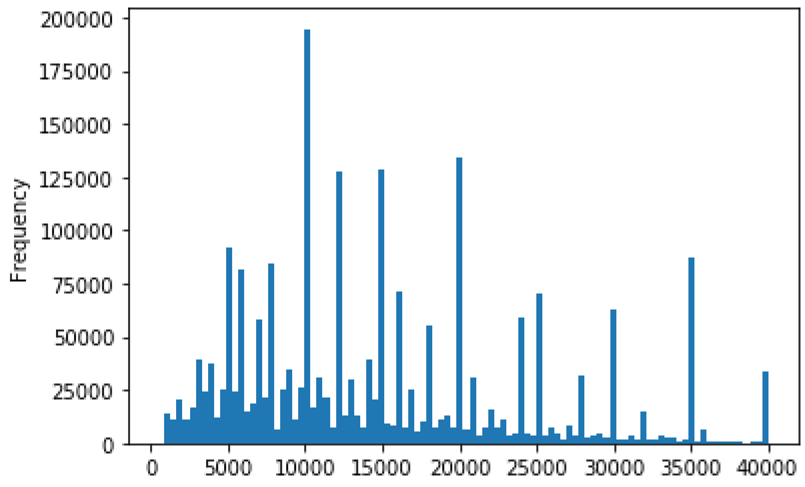


Figure 4. Distribution of loan principal amounts

Why is this distribution so strange? As can be seen, the distribution seems to be clustered around "pretty" numbers. This includes round numbers like multiples of five thousand and ten thousand. To show this explicitly, let's look at the top three most frequent loan values.

Amount	Count
10000	187236
20000	131006
15000	123226

Table 1 : Top three most common principal loan amounts

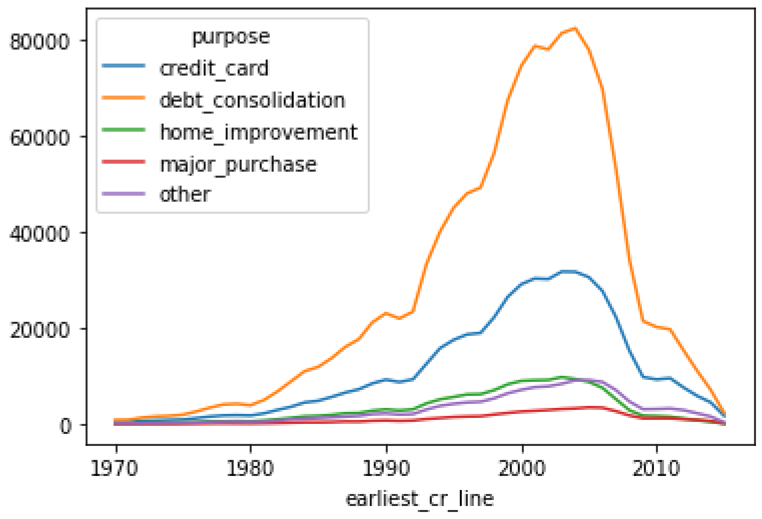


Figure 5. Loan purposes over time

There doesn't seem to be any distinctive behavior between loan purposes, this would have been evidenced by difference in the mean of each the time series. The goal of stratifying the earliest credit line by the purpose of the loan was to see if different age demographics use loans for different purposes. This would be measured by differences in the mean between these stratified distributions, as I believe it is a fair assumption that everyone gets their first credit line (earliest) around the same age.

For the categorical variables the frequency of each category can be displayed in a histogram. Other features were investigated such as the distribution of utilized credit, the distribution of grades and subgrades of loans, the geographical distribution by zip code of the borrower, the date of the earliest credit line of the borrower.

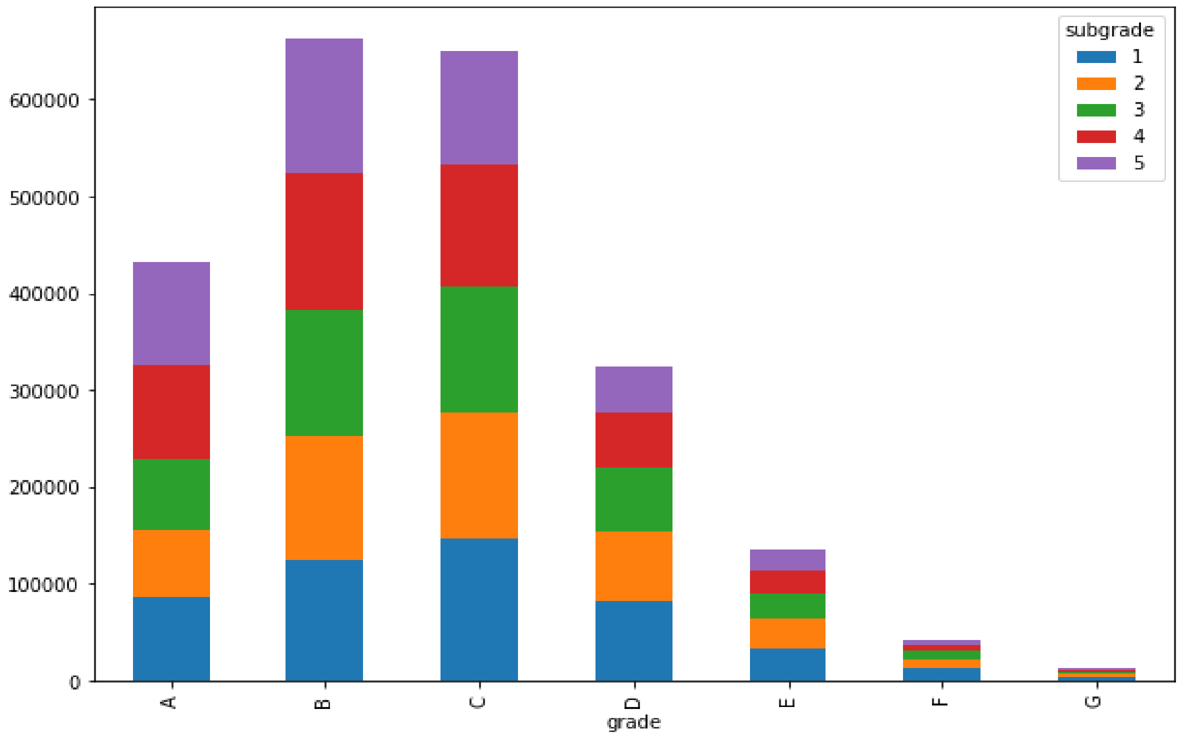


Figure 6. Distribution of loans by grade and subgrade.

3.2 Investigation of significant subsets of borrowers.

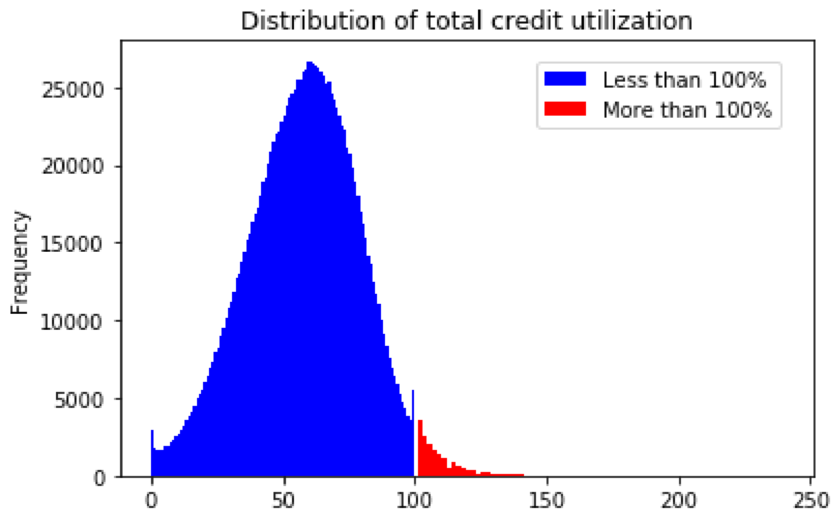


Figure 7. Total credit utilization distribution, values over 100% represent borrowers past their credit limits.

For the percentage of utilized credit it can be seen that there is a small portion of the dataset which exceeds the theoretical one-hundred percent threshold. The reason for these values is unknown but it made me ask the question, can more targeted models be made based on different subsets of the borrower population? To investigate, I first looked towards the difference between those above and below 100% credit utilization. For example, the loan amounts and average current balances for each group were:

The average **balance** for borrowers utilizing **more than 100%** of their credit: $182575.92
The average **balance** for borrowers utilizing **less than 100%** of their credit: $143809.02

The average **loan amount** for borrowers utilizing **more than 100%** of their credit: $11729.86
The average **loan amount** for borrowers utilizing **less than 100%** of their credit: $15274.88

The average current balance is actually greater for people utilizing less of their credit, This is not indicative of not paying off loans; in fact, people that use less credit on average have loans with higher principal amounts. To get a sense as to whether the separation by credit utilization is useful for partitioning the set of borrowers, we can look at the differences in summary statistics between the two populations. The quantity that stood out to me is the total balance excluding mortgage. The averages for each sample population were computed to be approximately 40000 dollars; but it seems that this value nearly doubles when excluding mortgage balances. It could be that perhaps there is a difference in home ownership status between the two groups.

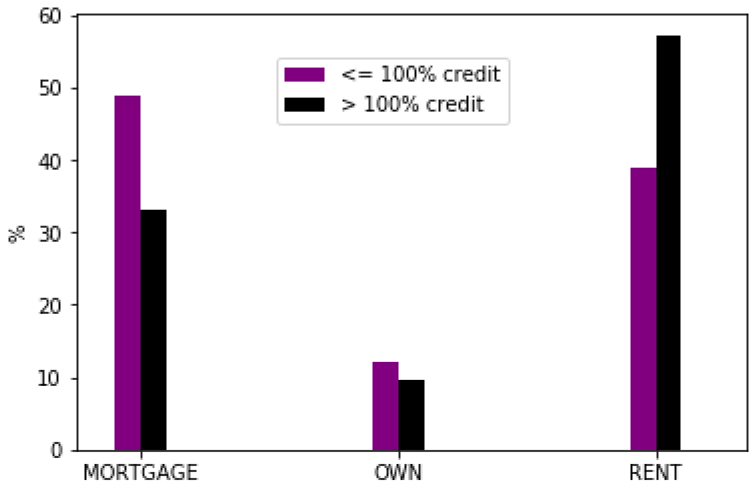


Figure 8. Home ownership status broken up by credit utilization percentage.

While this bar plot does not prove anything it does seem to imply that there is a difference in housing status between the two groups. It seems that a majority of the borrowers over their credit limits are renters while the other borrowers are more likely to either own or mortgage a home. The question I wanted to answer was this: can home ownership status be used as a distinguishing factor in any other manner? To explore this line of thinking, let's look at the total current balances of all borrowers.

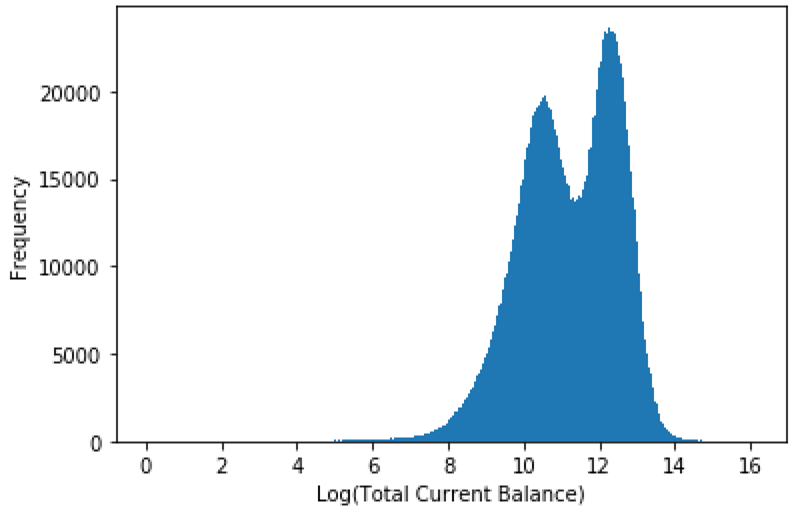


Figure 9. Distribution of the logarithm of total current balances. Note the bi-modal structure.

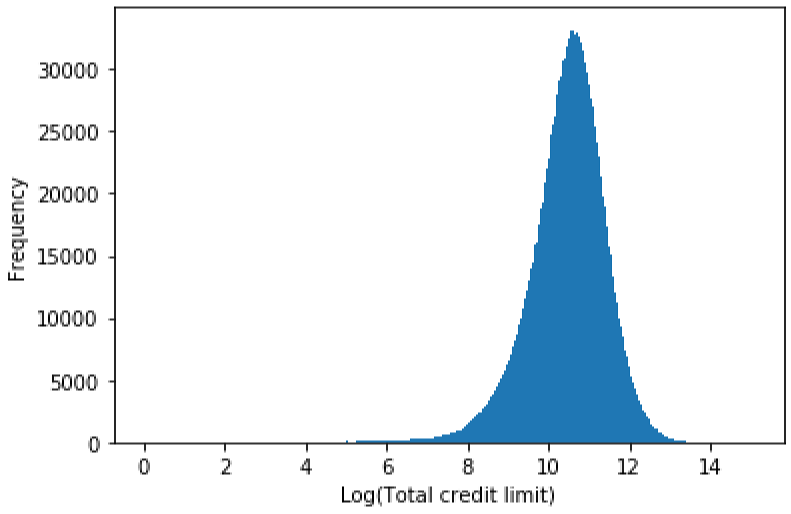


Figure 10. Distribution of the logarithm of total current balance, excluding mortgages.

I would have guessed that the total current balance of all accounts would have a normal distribution. Plotting the histogram, however, shows another story as can be seen in figure 8. My goal is to explore where these two peaks are originating from I then plot the total balance excluding mortgages.

The effect of home ownership status seems immediate as the two distributions are entirely different. To investigate the reason for the bimodal distribution I split the total current balance data in half (via the median value) and then create a bar plot for the home ownership status for each half. The stratification is much more drastic than was the case with credit utilization. distribution. For an example on how to interpret: mortgages account for nearly 80% of the accounts above the median total current balance, as indicated by the green bar reaching a value ~80%.

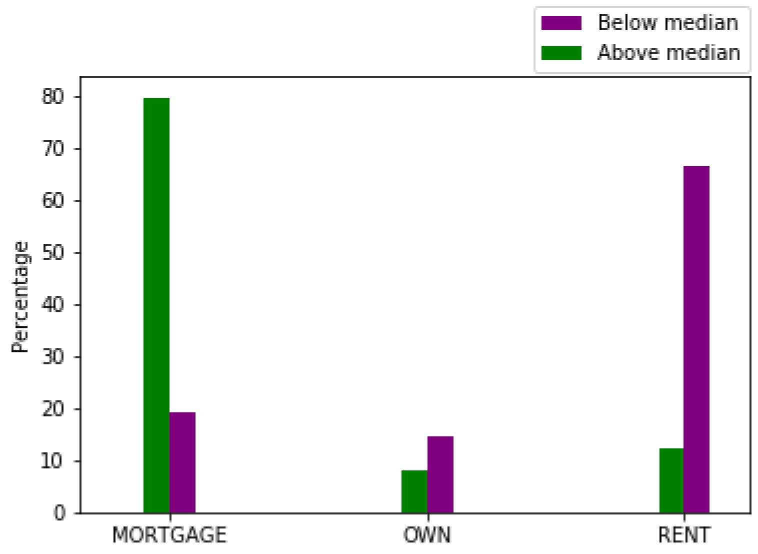


Figure 11. Percentage of each home ownership category for each half of the total

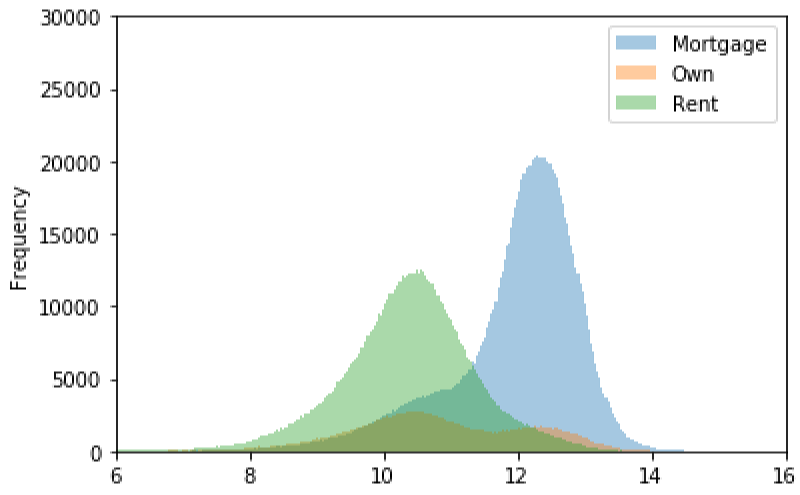


Figure 12. Total current balance distribution stratified by home ownership category.

This is visualized in figure 11 by plotting the distributions of (log) total current balance with respect to each category. Own and mortgage and perhaps the distribution for those who have mortgages still look like they're bimodal, but the total current balance distribution for renters seems to be unimodal. This almost seems like it's hinting at the possibility that there are hidden populations of customers. The idea I have in my head at least is that perhaps there is a useful way of subdividing or partitioning the data such that these subsets could each be treated as their own population; that is to say, each would have a separate model instead of there being single models for loan outcome prediction and recovered capital regression. The problem is that the next step forward, breaking down the mortgage and own categories is not obvious, even after repeating the same steps. So, in summary, it might be useful to further investigate or subdivide the borrowers into distinct populations, but up until now I have not found an obvious means of doing so.

3.3 Exploration of loan outcomes

To continue, the main goal of this project is to produce a model which accurately predicts the outcome of a loan. A bad outcome is defined as when either a borrower is late on payments or the borrower has charged off the loan. A good outcome is when the loan is paid in full. This “good” and “bad” dichotomy is an artificial construction resulting from aggregation by loan status. This frames our problem as a binary classification problem which can be modeled by logistic regression and random forest classification. With these methods and other considerations, a model for loan status prediction can be formulated.

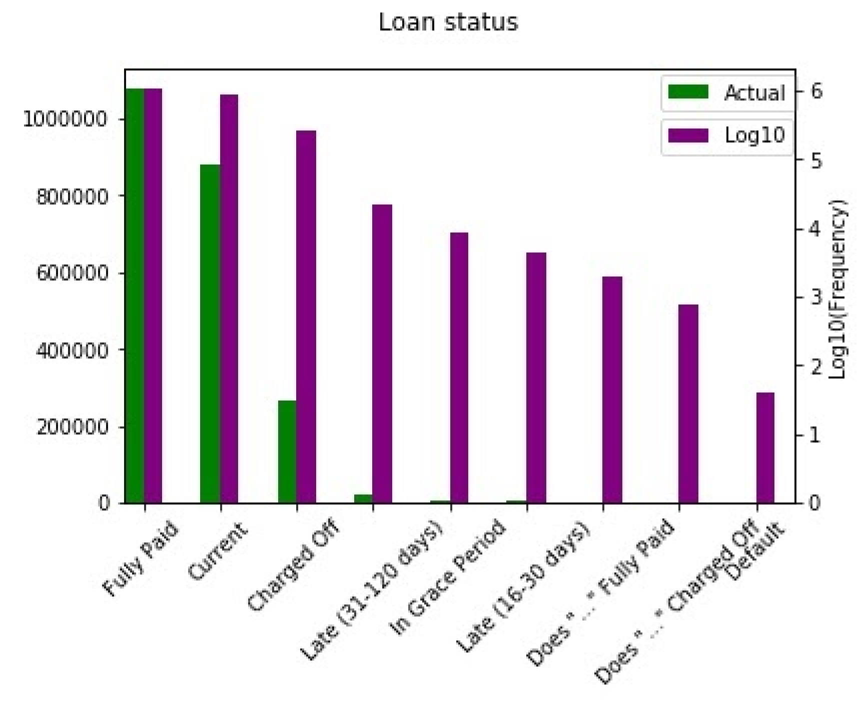


Figure 13. Loan status distribution (and logarithm for scale purposes)

As we can see by this figure 12, the vast majority of loans fall into three categories: "Fully paid", "Current", and "Charged Off". Because of the time dependence of the problem, the quantity being modeled is modified to respect this distribution. The modification is to predict the outcome of loans by their maturity date. Therefore we can prune the loans which have not matured yet. In addition, to make the problem a relatively balanced, binary classification problem the only categories that are retained are "Fully paid" and "Charged Off". The binary distribution plot shows the proportion of loans with bad outcomes.

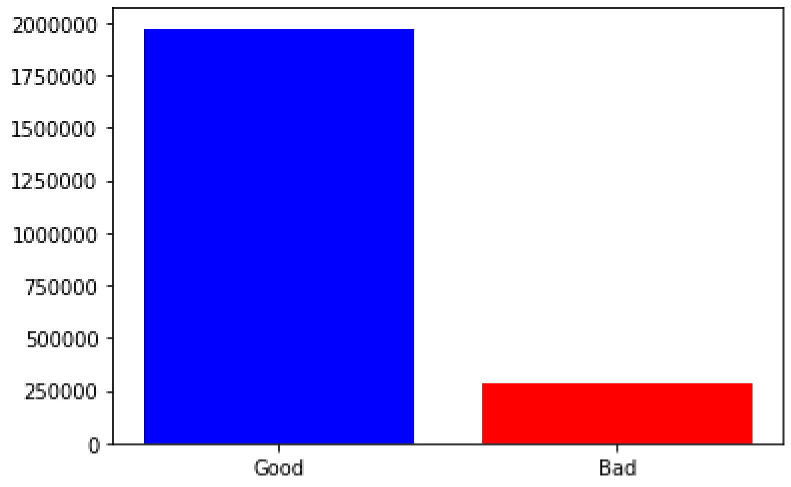


Figure 14. Loan statuses binned as either “good” or “bad”.

3.4 Exploration of loan recoveries

What can be done with loans of bad standing? Instead of the money being lost to the ether, financial institutions will naturally attempt to recover capital of charged off loans. This amount of capital recovered can be modeled as a continuous variable via regression techniques such as linear regression and many others. This variable is important because it is closely associated with loans of bad status, and provides a recommendation or course of action for loans that have become charged off or delinquent. Unsurprisingly the recovery amount is correlated to loan status but this is a misleading quantity as loans of good status do not need recovery to begin with. Because of this, I postulate that it may be wise to completely filter out loans of good status before performing any modeling. The main concern is the order of these time dependent variables in regards to training and testing data sets for the models as well as for cross validation. If future data is accidentally used in the training then the predictions are essentially worthless. This goes down all the way to the level of normalization; that is, if normalizing the data then one should be sure to only normalize with respect to the training set.

How should these be accounted for in the modeling process? For the loan classification process, all time dependent quantities need to be removed from the model training process for the loan classification problem, as the model is attempting to predict a decision made at a certain point in time. The model for the recovery amounts is agnostic of time as it merely wants to predict the amount that can be recovered going forward; it is not a time sensitive decision; therefore, no considerations need to be made for the second stage of the capital recovery model.

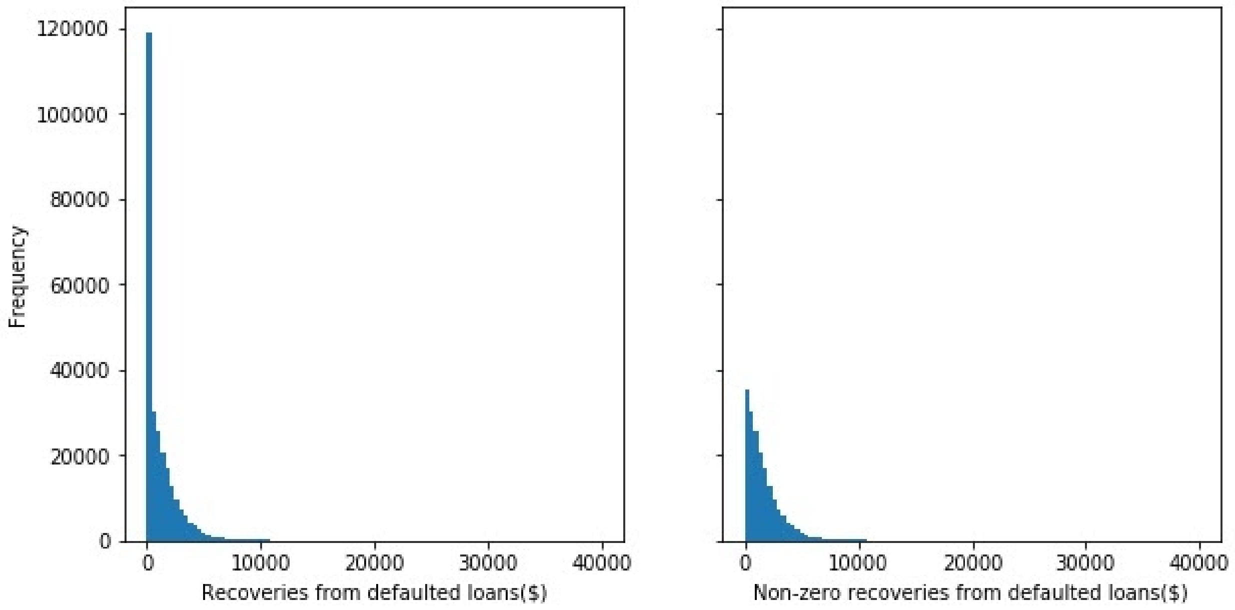


Figure 15. Distribution of recoveries from defaulted loans, with and without zero recoveries.

First, look at the distribution of the would-be dependent variable. It seems that there are still a large number of charged-off loans that have had no money recovered from them; part of the motivation for this stage of the project. Nearly a third of all defaulted loans have had zero capital recovered. In the modeling stage of this project, I explore what happens when both including and excluding these 0 value from the models.

4. Statistical Analysis

4.1 Kolmogorov-Smirnov testing numerical features

The data types in this data set are heterogeneous; even within the subset of features which are defined on a continuous domain (approximately continuous, as most variables monetary in nature), they can have dramatically different distributions. I investigate these distributions, as I believe that these continuous variables will have more of an effect in the modeling process, mainly because I think they are better positioned to capture the individualistic nature of loan borrowers, as they take on more unique values than other variables. My idea is to investigate the numerical features to know how to handle the data in the modeling process, i.e. how to rescale, if a transformation is warranted. The variables I investigate are:

1. The annual income (without some very extreme outliers)
2. Total payments made to date
3. The percentage of credit being utilized
4. Number of open accounts
5. Debt to income ratio

Income inequality is a relatively common topic in pop culture such that it is fairly well known that the distribution of annual income does not follow a normal distribution. It is not clear if this will remain the case for the distribution of annual incomes of loan borrowers. It is questions like these that motivated me to investigate the distributions of the various quantities. Specifically, using the Kolmogorov-Smirnov test and the corresponding p-values I attempt to classify the different distributions that occur in the data set. This could have an effect on my choice for rescaling and engineering of the various features. Because the sample size is so large and I am relatively new to financial data, I would have assumed most features would follow a normal distribution; it turns out I was quite wrong in this regard. The next handful of figures represent histograms for the values of each feature; the red line corresponds to the kernel density estimate produced by the "kdeplot" plot function from the Seaborn package. The y-axis labels on the left correspond to the number of borrowers in each histogram bin, while the right side labels are the density. In this stage, the goal is to find the distribution which matches the histogram so that the sample distribution and hypothesized distribution can be compared via the Kolmogorov-Smirnov test. First off is the distribution of annual income.

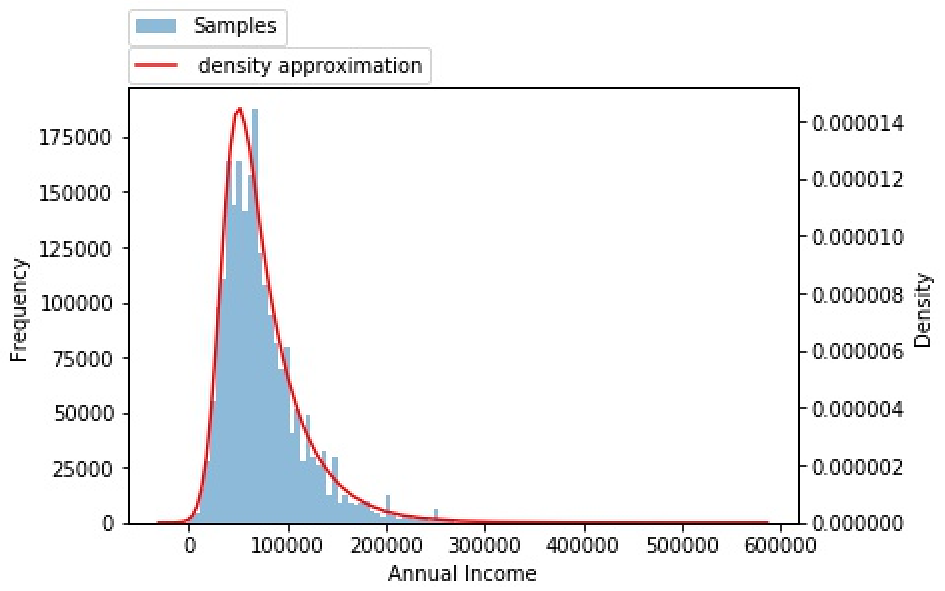


Figure 16. Annual income with exponential-normal kernel density estimate.

My methodology is as follows: look at specific quantities such as skew and kurtosis as well as the crude shape produced by the histogram. For annual income I hypothesized that it followed an exponential normal distribution.

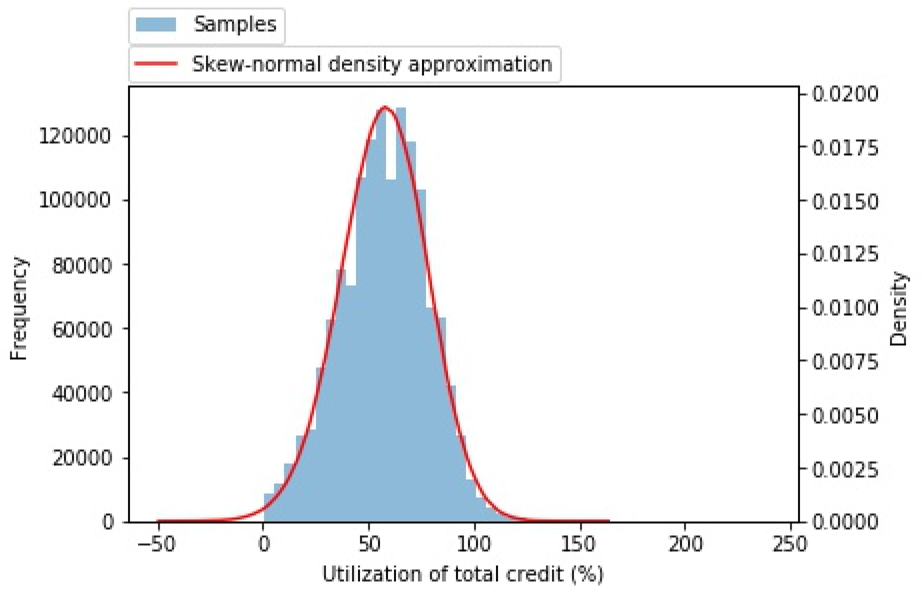


Figure 17. Total credit utilization distribution with skew normal kernel density estimation.

Another quantity previously reported was the utilization of credit; I believed (before visualization) that it followed a normal distribution but it appears upon further analysis that my claim is that it follows a log-normal distribution.

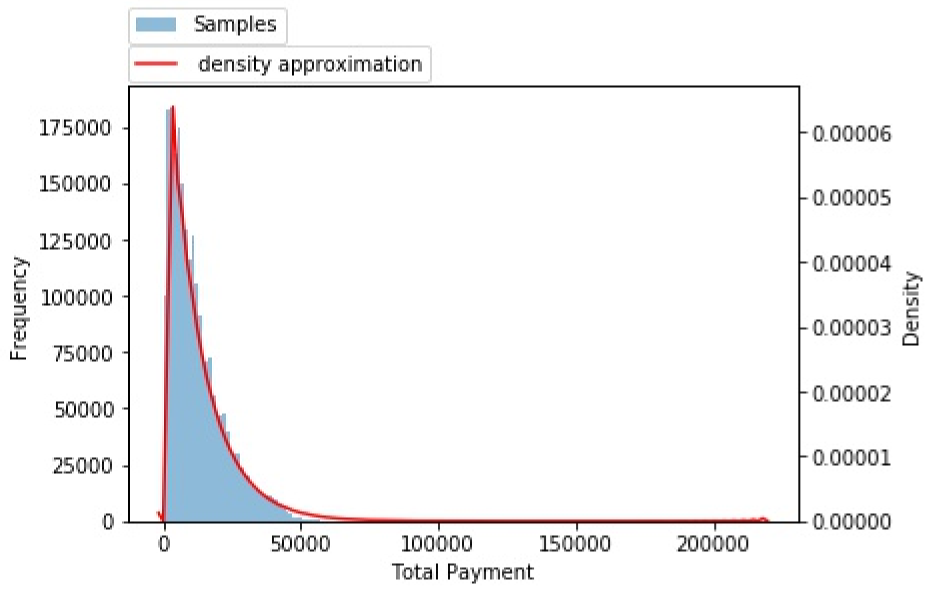


Figure 18. Total payment with exponential kernel density estimate.

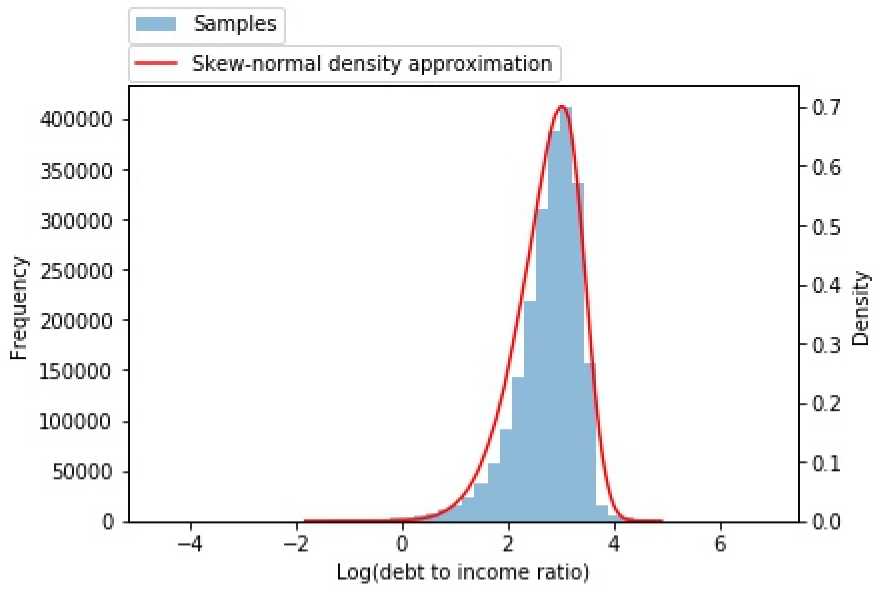


Figure 19. Logarithm of debt to income ratio with skew normal kernel density estimate.

The debt to income ratio is yet another quantity which is close to exponentially distributed has a skewness that says otherwise. Therefore, instead I took the logarithm of the distribution such as to use a skew-normal model for the logarithm.An alternative model for an exponentially distributed random variable Y is to take the log and model it with either a skew-normal or normal distribution. This is likely a better choice because the skewness of the variates is not what would be expected from an exponential distribution, where it is a constant value independent of the commonly used exponential distribution parameter. In order to apply the logarithm, however, the values equal to zero have to be removed to avoid transformed values of negative infinity; this does not discard too many data samples and so I find it a viable option.

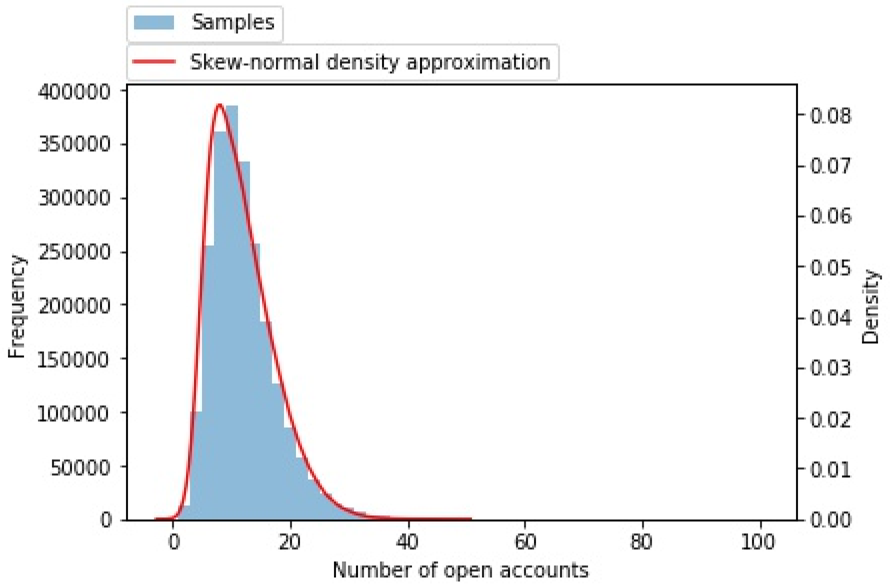
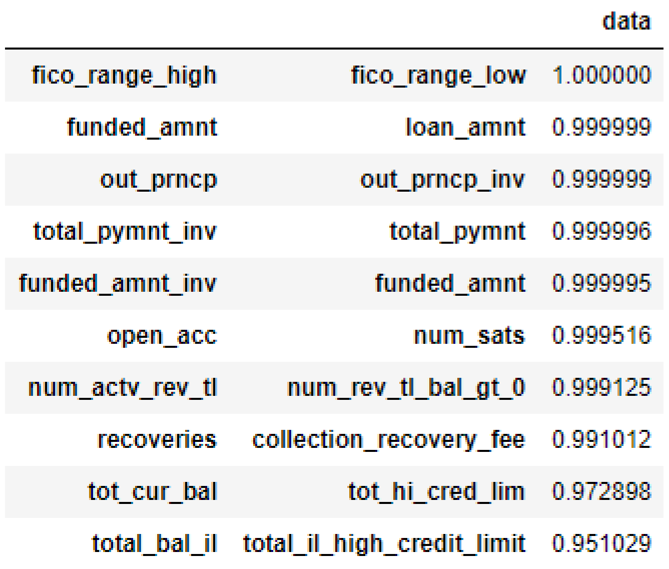


Figure 20. Number of open accounts (discrete) approximated by skew normal kernel density estimate.

The number of open, a discrete valued quantity, nearly looks continuous when plotted as a histogram with bin width greater than or equal to one. This presents an interesting test; given a discrete variable how well can it be modeled with a continuous distribution, taking only the integer part when sampling?

The Kolmogorov-Smirnov test compares cumulative distribution functions of sample and reference distribution (one-sample) or compares the distributions of two-samples. To claim that the approximated kernel density estimates for the distributions are accurate depictions of our sampling distributions, statistical testing is required. The way I set this up is that the null hypothesis is that the distribution is the specified one (with parameters included) and the alternative is that it is not. Therefore, if the KDE plots are accurate, the null hypothesis should be accepted. Unfortunately, this does not happen for any of the supposed distributions, as the p-values were all essentially 0. This means that we reject the null-hypothesis that the distributions were as I had claimed. It seems that the kernel density estimate plots are much more misleading than I thought.

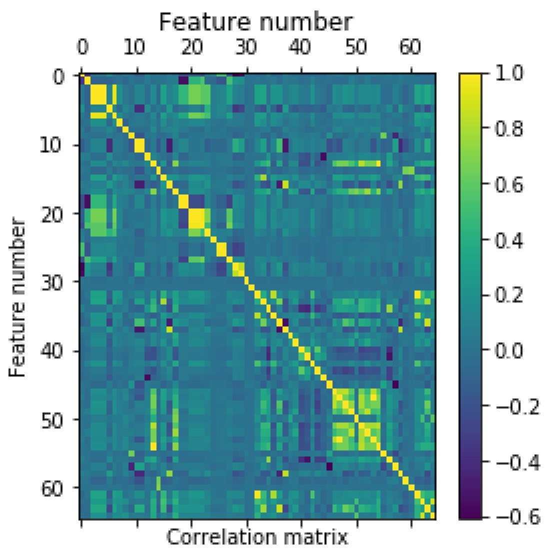
4.2 In-depth look into correlations between features



		data
fico_range_high	fico_range_low	1.000000
funded_amnt	loan_amnt	0.999999
out_prncp	out_prncp_inv	0.999999
total_pymnt_inv	total_pymnt	0.999996
funded_amnt_inv	funded_amnt	0.999995
open_acc	num_sats	0.999516
num_actv_rev_tl	num_rev_tl_bal_gt_0	0.999125
recoveries	collection_recovery_fee	0.991012
tot_cur_bal	tot_hi_cred_lim	0.972898
total_bal_il	total_il_high_credit_limit	0.951029

Table 2. The top correlations between features in descending order.

There are a number of pairs of features with pearson correlation scores greater than 0.999 for a specific and relatively obvious reason. Specifically, some features are essentially identical; an example being: the funded amount of a loan and funded amount of a loan from investors. If investors represent the overwhelming majority of loan funding then these features are nearly identical which seems to be the case upon inspection. I decided to not include the very highly correlated features as they should contribute little to the modeling process other than computation time. The correlations in their totality can be visualized by plotting the correlation matrix as a color coded image.



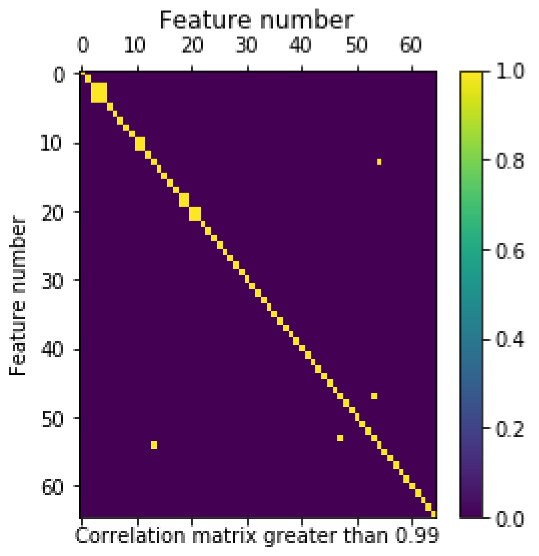


Figure 21. Plotting the color coded correlation matrix, as well as the masked version showing correlations with magnitudes exceeding 0.99.

Likewise, by filtering out all values less than a threshold, chosen here to be a value of 0.99, the highest correlation pairs are demonstrated. Note that the matrix is symmetric so there is redundant information. Because the main targets are the loan status and the recovery amounts, it is prudent to investigate the correlations with these variables. Because the loan status is a categorical variable, it would have to be encoded in order for this to be well defined. Pandas has one-hot encoding via a function "get_dummies" which allows for one-hot encoding. Using this to convert the categorical data to numerical, I could then compute the correlations of this with the other numerical features. As can be seen, other than the autocorrelations, the remainder of the features have correlation scores between -0.6 and 0.6, approximately. This is sufficient (to me) to not have to drop any of the other numerical features before the modeling process.

Likewise, the same can be computed for the recovered amount of capital. The largest correlation is between the recovered amount and the collection recovery fee; that is, the money that is recovered the more must be paid for its recovery. Therefore, in order to avoid a biased model, I drop the collection recovery fee data from the feature data.With this knowledge I believed that I was ready to begin the modeling process.

5. Modeling and analysis

5.1 Feature selection and engineering

For the loan outcome prediction portion of this project, only the loans which have both matured are considered. Additionally, in this subset of loans there are a number of outcomes but by far the most highly populated are loans which have been fully paid or charged off. This can be formulated as a binary classification problem: the loans that are charged-off
will be denoted as "successes" or by the integer 1. Likewise, loans which have been fully paid off will be denoted as "failure" by assigning these the integer 0.

The first main hurdle that we need to overcome is the time dependent nature of the problem. If not handled carefully, we could accidentally perform what is known as "data snooping", which is the contamination of the model by inclusion of information from the future. The two main effects that this time dependency has are Multiple time dependent variables and time ordered cross validation folds. To account for the first of these issues, all variables that are recorded at times more recent than the date that the loan was issued are removed. I could tell which features these were because their descriptions typically contained descriptive statements such as "in the past twelve months". Some of the features are ambiguous, however, and so I lean on the side of caution by removing these features as well.

By including only loans which have matured the predictive model is then only predicting whether or not a loan will default by its maturity date. The maturity date depends on the term of the loan; it might be wise to separate the loans by term but currently this is not done.

At this point I needed to account for the two different data types in the feature data, numerical (float) and categorical (object, could be cast as string). The categorical variables will be encoded and represented by binary numerical columns via one-hot encoding. Some of the categorical variables have many categories, of the order of a thousand, which complicates things.The most diverse categorical variables will be handled in one of two ways: we shall argue that they are either unnecessary or reduce the number of categories via binning.

The following are the choices made for these variables: originally the choice was made to remove all but the first two digits of the zip_code (there are only three recorded). This in fact lowers the resolution to a point where the state of residence is more precise; therefore, I believe that the zip_code can be dropped as it isn't informative. For the categorical variables which may be cast as datetime variables, the number of categories will be reduces by application of a binning strategy. The binning for the issuance date is naturally handled when the cross-validation folds are being produced, as this is the determining time dependent quantity. More specifically, we drop the issuance date from the calculation because based on the folds, the training and testing data will never intersect by definition; which I believe implies that the model will not know what to do. For the earliest_credit_line dates it is less obvious as to how to proceed. The earliest credit line is thought of to be a proxy for age groups. Depending on the context, demographic age groups can be quite large in terms of the age spread. Therefore, I believe that only creating a few bins for this variable is sufficient. Applying the same argument comparing zip_code and addr_state, I remove the 'grade' of the loan in favor of keeping the subgrade. The grades take the values such as 'A', 'B', 'C' while the sub-grade is more specific 'A1', 'A2', ... 'E4', 'E5'.

Therefore before the modeling process the following steps shall be taken:

1. Remove target variable from feature data
2. Remove more coarse categorical variables
3. Encode datetime-like categorical variables with KBinsDiscretizer (OneHotEncoding strategy).
4. Encode remaining categorical variables with OneHotEncoder (scitkit-learn)

Note: To avoid collinearity in the one-hot encoding process, the category with the lowest frequency is dropped (for each feature). The first test of these methods was to develop an intuition as to what to include in the cross-validation process. To model, the data of course needed to be processed as previously described. There are a number of choices for the method with which to scale the numerical data. The tests are performed by using the entirety of the training data (named traintest because the testing folds come from this subset) to fit the model. The model is then tested using this data as well as a set of "holdout" data which the model does not know about. The holdout data, by virtue of the cross-validation folds, corresponds to the most recent loan data. The preliminary tests that were run were naive modeling using only the default hyperparameters of the two classifiers under investigation.

5.2 Cross validation setup

The cross validation process as well as the folds will be customly made, instead of using scikit-learn's TimeSeriesSplit() for instance. This to be absolutely sure that the time ordering is respected. Additionally, there are some considerations regarding encoding and preprocessing that are best handled by a custom cross-validation procedure.

I made the folds such that the number of training samples is cumulative over time. This attempts to reflect the collection of data over time. I also ensure that I created a "hold-out" set of data that will be used for final predictions and analysis after all cross-validation and model learning has been accomplished. In order to split the loan data into meaningful folds The loan issuance dates are aggregated by month, but from the metadata we know that the data is reported quarterly. Using this as motivation, the hold-out data was originally to be the most recent quarter. Because the number of loans has grown over time, the last quarter represents one seventh of all of the data, after processing and handling pathological values. Therefore I deemed that this quarterly fold creation is too imbalanced to be practical. With these cross validation folds, the next step in the modeling process is to create a pre-processing pipeline which handles and caresses the data such that it is properly positioned to be modeled with.

5.3 Loan outcome prediction models

Use only training data to see model performance as a preliminary to the full modeling process. For me there are two main takeaways from these tests. Firstly, the class weights need to be balanced to represent the frequency at which defaults occur (approximately 1/7 of the total samples). Second, it is easy to overtrain the random forest classifier, as indicated by the tremendous difference in performance between training and testing data. The main goal is to reduce capital loss not maximize profits. Therefore, we value prediction of when a loan will be charged off more than fully paid. We can account for this by changing the class weights in the classification process.

This model will reject loans that would have been fully paid in order to avoid loans that will become charged off. In other words, the goal is to maximize the number of true positives, where "positive" in this case is equivalent to a loan being charged off. It's clear that the class weights need to be balanced because otherwise the models are modeling the null information rate. Therefore, for everything that proceeds I shall weight the binary classes by their inverse frequency.

The choices for the hyper-parameters aren't obvious so the cross-validation process attempts to cast a broad (but coarse) net on the hyper-parameter space. For the random forest classification, the hyperparameters are not very intuitive to me, so I varied most of them including: the number of trees (estimators), the maximum depth of each tree, the minimum number of samples required to split, the minimum samples required in a leaf. The best results were given by the following set of parameters

```
RandomForestClassifier(bootstrap=False, ccp_alpha=0.0, class_weight='balanced',
                    criterion='gini', max_depth=27, max_features='auto',
                    max_leaf_nodes=None, max_samples=None,
                    min_impurity_decrease=0.0, min_impurity_split=None,
                    min_samples_leaf=200, min_samples_split=2,
                    min_weight_fraction_leaf=0.0, n_estimators=25,
                    n_jobs=None, oob_score=False, random_state=None,
                     verbose=0, warm_start=False)
```

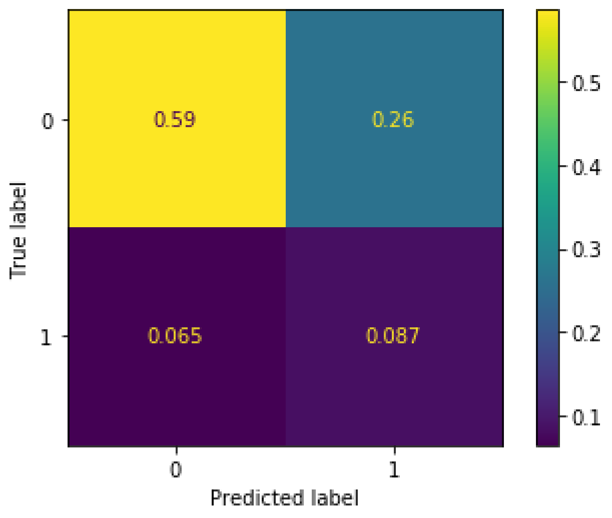


Figure 22. Confusion matrix using cross-validated random forest model using holdout-data set

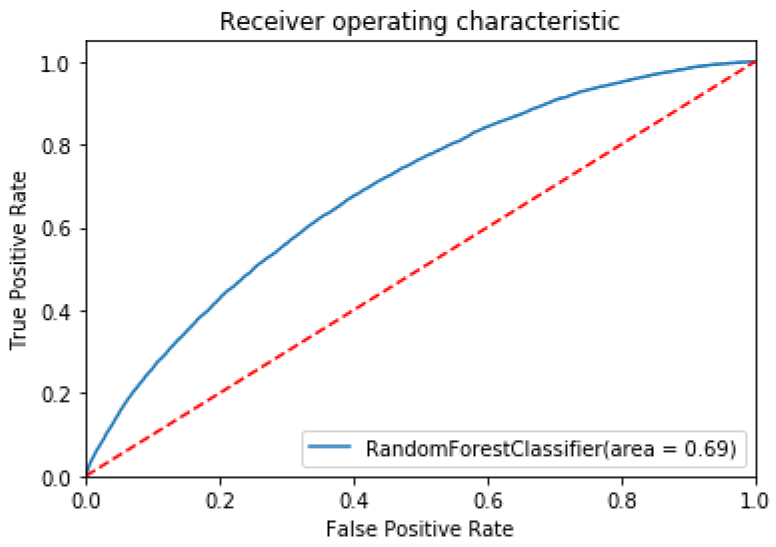


Figure 23. ROC-AUC using cross-validated random forest model using holdout-data set

Likewise, for the logistic regression model I only tested the regularization constant (‘C’, smaller means stronger regularization) and the tolerance. This resulted in the following model:

```
LogisticRegression(C=0.25, class_weight='balanced', dual=False,
                   fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                   max_iter=500, multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

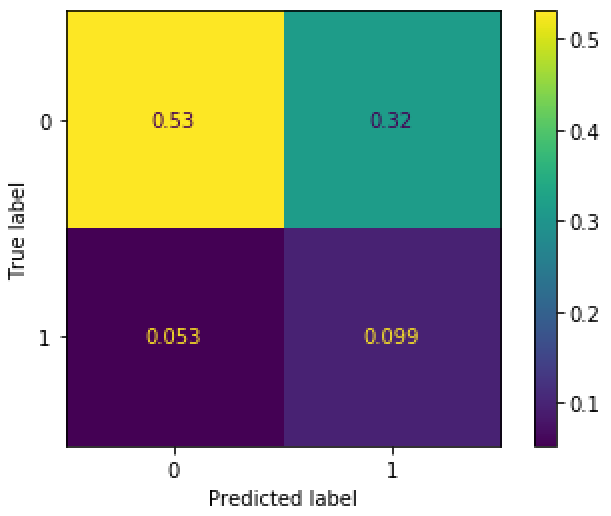


Figure 24. Confusion matrix using cross-validated logistic regression model using holdout-data set

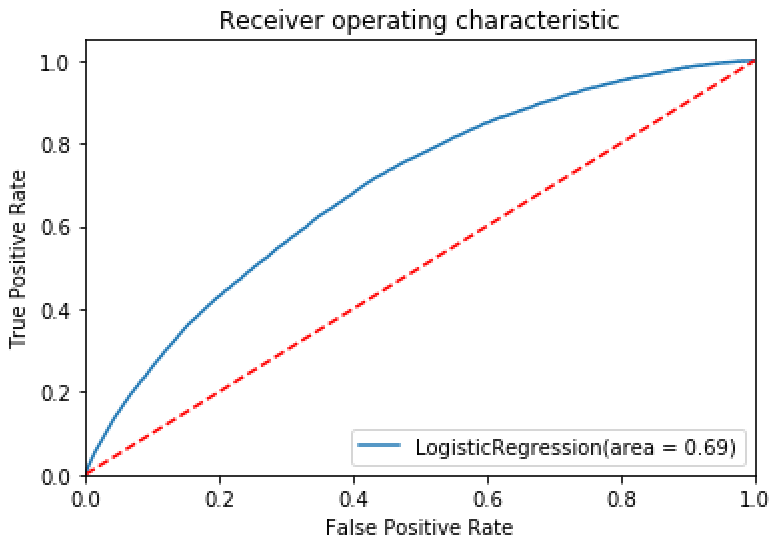


Figure 25. ROC-AUC using cross-validated logistic regression model using holdout-data set

The logistic regression model has decent performance in regards to the main goal which is attempting to identify true positives, which it accomplished about 66% of the time. Because there are so many false positives, however, I do not believe that this is sufficiently accurate to determine whether or not to provide loans, as it would reject far too many applicants. Therefore the best that this model can do is to flag accounts which
are at risk of defaulting. This is still a proactive measure but it would only be correct around 25% of the time and so it would take an additional investigation to see whether or not this is worth the effort. This is supported by the increase in both recall and precision of the logistic regression.

5.4 Loan recovery prediction models

The deliverable model that is about to be produced is one that predicts the amount of recoverable money from a "charged off" loan. This is performed via two different regression methods, Ridge regression and stochastic gradient descent. There is no time dependent component of the hypothesis in this case; that is, no special time order is required. To produce these two models all data needs to be in a numerical form, which requires the encoding of categorical variables. The procedure for how we first produce these models is described by the following enumerated list.

1. Take the subset of the data which corresponds to charged-off loans (already done).
2. Perform preprocessing operations (Categorical variable encoding and numerical variable rescaling).
3. Cross-validate a Ridge regression model
4. Cross-validate a StochasticGradientDescent (SGD**)** model
5. Analyze the effectiveness of each model with various metrics.
6. Decide on a final deliverable model.

There are many categorical type variables which need to be encoded in order to be used for regression. There are a number of categorical variables with a large (100+) number of categories. In order to better handle this fact there are a number of features which we prune from our analysis.

The prediction is occuring in the present moment. All of the data corresponding to charged-off loans can be used, and it does not have to be treated as a time series, as we are not trying to predict the future. As can be seen, the variables with the second to eight largest number of categories are related to dates; These features, from the cleaning process, are a mixture of dates (month-year) and the category 'Missing'. These variables will be encoded using binning, specifically KBinsDiscretizer using the One-hot encoding strategy with a small number of bins per feature vector. The manner with which we bin is to represent the dates by their year, and then use a uniform bin-width strategy with three bins.

Simultaneously, the "Missing" values are one-hot-encoded separately, such that the result is three bins for the date's years and an additional bin for the 'Missing' Category. The largest number of categories belongs to the zip code variable; which is reported as the first three digits (followed by 'xx'). We can reduce the number of categories by aggregating by the first *two* digits; but if we do this it is almost nearly
the same as identifying by the state; In fact, rounding to two digits is actually less specific than identifying by the state. We already have the state in 'addr_state' feature data so there is no motivation to keep the zip codes if reducing them to two digits. Additionally, the grade of each loan is less precise that the subgrade of each loan such that we dispose of this as well. I left these next calculations here for posterity, but it turns out that handling the datetime-like variables in this binning manner introduced a lot of noise into the models.

For each model we make a preliminary test on the training data; this is similar to the no-information rate analysis in classification problems. That is, we train and test with the same data set. If the model is inaccurate in this case then there is something seriously wrong with the data or the model. While there are an incredible number of options for fine tuning, my main investigation is into the efficacy of the numerical feature transformers. I used scikit-learn's SGDRegression (stochastic gradient descent) and Ridge regression (regularized linear least squares). The performance on the training set was perhaps too good; an indication of overfitting, but alas we press on. I investigate this peculiarity by a crude yet effective method of performing the same training and testing routine, but adding one feature at a time.

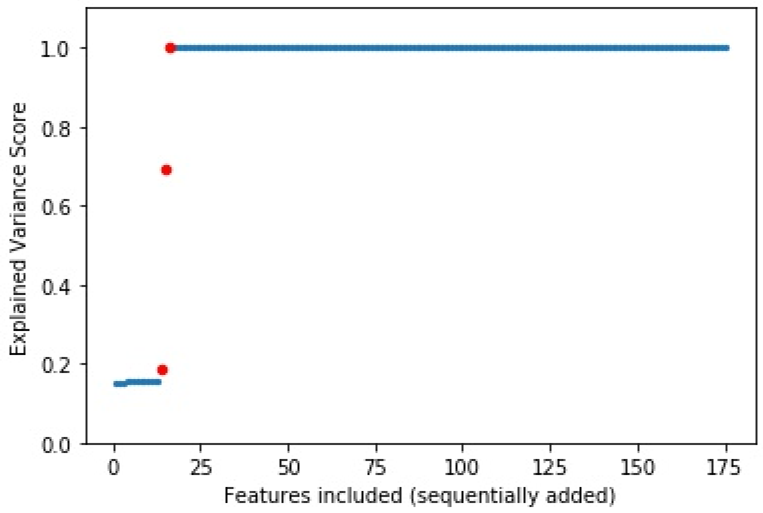


Figure 26. Dependence of explained variance on the number of included features (not randomized).

There is an incredible jump in explained variance that happens as three specific features (the first jump is small in relative terms but significant, represented by red dots). As if to indicate a collinearity between these columns and the target variable. I do not believe that there is any contamination between these features but at the same time they seem to perform too well as predictive variables.

According to the metadata, the definitions of the three features are:

1. total_pymnt : Payments received to date for total amount funded
2. total_rec_prncp : Principal received to date
3. total_rec_int : Interest received to date

And the target variable:

recoveries : post charge off gross recovery

Therefore my recommendation would be to uncover a more detailed explanation of the relation between these features and the target variable. Until then, a cautious application of this model is recommended. Unfortunately the performance in the absence of these three features is pretty terrible, as can be seen by the models without these three features. The explained variance drops from near unity to values approximately equal to 0.25.

Note that this isn't a time dependent problem so a possible is that total payments perhaps include post charge off gross recovery. and there does not seem to be any glaring relations between these three variables. It may be that these features are simply very good descriptors. With no way to know or follow up, I continue towards the real modeling procedure.
One way of improving the models that exclude the suspicious features was to subset the data even further, only including charged off loans which have non-zero recovery amounts. This idea originated from the fact that the distribution of recovery values has a mode equal to zero. By including this measure, the explained variance score increased to around 0.47.

The main cross validation process tested sets of parameters as well as different column transformers that acted on the numerical feature data. This is of course in addition to the two different models being used as mentioned earlier. In summary the feature data column transformers tested included: QuantileTransformer(), MinMaxScaler(), and StandardScaler(). The models tested were SGDRegressor() and Ridge() from scikit learn, the hyperparameters tested included the tolerance and regularization constants in both models. In every case the Ridge regression outperforms the stochastic gradient descent. The final ridge regression model had parameters

Ridge(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=None, normalize=False, random_state=None, solver='auto', tol=0.1)

And the final results on the different numerical transformers were

Transformer	Mean squared error	ExplainedVariance
QuantileTransformer	2278262.7349281064	0.4270508478802073
MinMaxScaler	2205599.158832141	0.44532419977748505
StandardScaler	2381178.413802254	0.40166608575310014

Such that for the final models, with and without the three dubious features, using MinMaxScaler() provided results given by the two following figures.

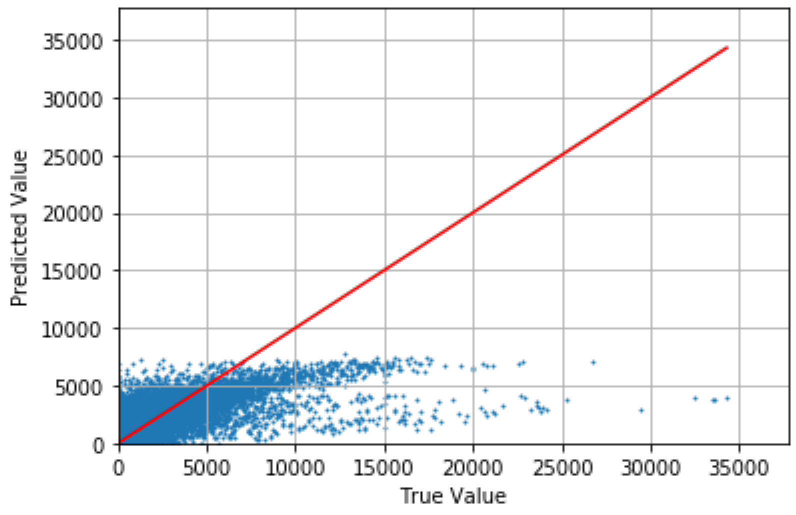


Figure 18. Final ridge regression model (minus 3 features): Explained variance score 0.46042358045918

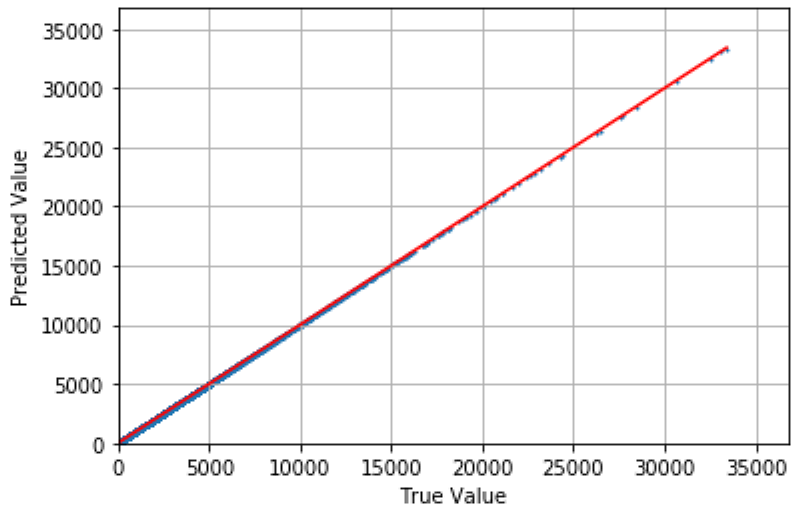


Figure 19. Final ridge regression model (including three "dubious" features) : Explained variance score 0.99999738611

The above plots represent results from scaling with MinMaxScaler on the feature data. It's quite clear that the inclusion of all of the features performs dramatically better, therefore the course of action would be to discover whether or not the inclusion of these features is valid. In summary, Ridge regression with small regularization (larger value = smaller regularization per scikit-learn's docs) but a relatively strict tolerance seems to perform the best; even though the model appears to have high variance (overfitting).

6. Future work

This project attempted to create a two-stage recommendation system which included proactive and reactive measures for capital recovery from defaulted loans. The performance of the first stage was subpar and the performance of the second stage was highly dependent on a connection between data features that could not be known with the current level of information. In light of this analysis, I would not recommend using these models as I had originally intended. The first stage could at best be used as a system to flag loans which may be at risk of defaulting so that proactive measures could be taken. The second stage could be used if it turns out that the data is not contaminated as the unrealistic performance seems to indicate.