Predicting Loan Default In Peer-To-Peer Lending

François Lemoine February 27th, 2017

I. Definition

Project Overview

Loan default prediction is a common problem for various financial companies and a well defined and known problem in data science. This is the type of problem banks (e.g Goldman Sachs), credit card companies (e.g. Visa), micro-credit banks (e.g. Grameen Bank) or FinTech companies (e.g Lending Club) face whenever customers ask for a loan. The data from Lending Club about lenders is available on their website and we will be using it to make predictions about loan default and whether or not we should lend to a customer. We will be using the data from 2007-2011 because most of the loans have been repaid or defaulted on.

Lending Club is the platform or rather the marketplace where investors and borrowers meet virtually. Lending Club processes the application with their own data science methodes but on the side of the investor there is still due diligence to be performed in order to insure the creditworthiness of the borrower and the level of risk involved in any given case. Applying machine learning to loan default predictions showcase a useful application of this branch of artificial intelligence to solve real-world and business problems. We will try to build this model with the most transparency possible as to mirror the conditions in which financial institution must disclose this process. Even though we are building this model with a conservative investor in mind, the model could be later tweaked as to be used in a financial institution or a Lending Club investor.

Problem Statement

If a model is able to identify credit-worthy customers that were not recognized by traditional credit scores while minimizing their risk of default on the loans, this can be a lucrative niche market or micro-market. Pushing higher the profit margin of the financial institution or investor. Although, the prospect of more customers seems positive, we need to be careful as to not lend to people that will default on the loan, this would cause a drop in the earlier stated objective. Thus a conservative approach and rigorous evaluation metrics will be kept in mind throughout the project. The loan default prediction is a problem of binary classification (do we lend or not). We will at first need to understand the data and we will achieve our goal of better understanding by analyzing the data with the data dictionary. This will give us a sense of what is relevant or which columns could be reworked to build

features if necessary. We also expect to clean the dataset because it is common to find data with a sizable amount of missing values or trivial rows and columns. One recurring problem in binary classification is the possibility of class imbalance, or outcome imbalance. This generally occurs when we are trying to classify two states, but that one state is much more common. In our case, we can safely anticipate that loans are repaid, otherwise Lending Club would not be a profitable company. Whenever we are talking about class imbalance, we can talk about precision. And to improve the precision of our prediction, we will be using logistic regression and random forests which are models that have been demonstrated to be good classifiers in these instances.

Metrics

To measure the success rate of our model, the clearest metric is the precise prediction of loan default. The decision that the model takes has consequences as to the profitability of the investor or the financial institution putting the money on the line. This is why work will have to be done with the error types (false positive, false negative, true positive and true negative) which we will explain later as this will help guide us to a conservative evaluation of the loan default rate. The best metric to evaluate the model is the precision of our algorithm to predict whether or not a customer is going to repay the loan. We achieve this by training our model on the training dataset and then trying to predict – based on the columns value – the good customers from the bad customers. We can as a consequence, measure the precision, practicality and realism of the model. The precision of our model is measured by the sum of true positives divided by the sum of true positives with false negatives:

$$Precision = TP / (TP + FP)$$

We calculate the precision of our prediction at every steps of the implementation and this helps guide us towards a working model. But we need to have a high precision rate at the expanse of accuracy. This is explained by our objective to keep a steady return on our investment. In order to increase our confidence that the loan will be repaid, we follow the strict requirements developed by the model. This implies that a lot of loans are not funded by the conservative investor following our model, but it makes sure that it is repaid. The return on investment is small but statistically constant. As we saw in the Jupyter Notebook attached to this project, the accuracy of our predictions decreased as we improved the precision of the model. This means that we pass on many opportunities but we have a very high confidence level that the loan will be repaid with interests when we lend to borrowers. Some

models try to balance the accuracy with the precision (Tsai, Ramiah, and Singh, 2014), but we feel this is not the lowest risk strategy in this context.

II. Analysis

Data Exploration

The dataset has 42538 rows and 111 columns. The columns represent different information gathered as part of the first inquiry by Lending Club. The data dictionary file provided with the dataset, indicates that columns are information about the borrower and the outcome of their loan repayment. Part of why we picked the data from 2007-2011 is because we are almost certain that the loans have been repaid or defaulted on by this time. This means that we are trying to model future outcomes based on past data. This reduces the uncertainty of more recent data and it is understood that recent data could be used after the model is tested to produce actionable predictions.

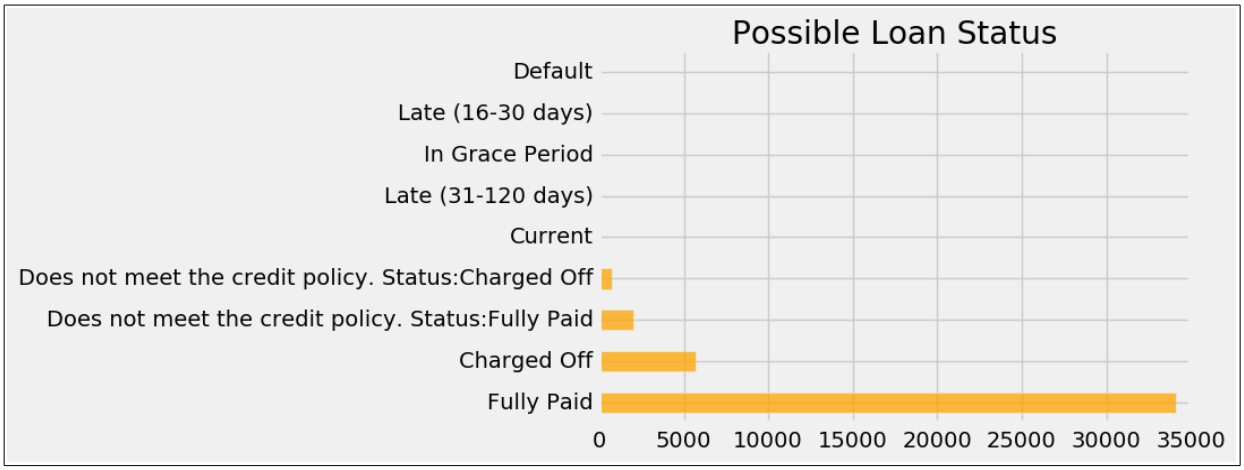
At first, we wanted to know if the columns were filled with useful informations or were mostly empty. Our data exploration has uncovered many empty or almost empty columns and we have removed these columns from the dataset because it would prove a difficult task to go back and try to answer for each data point that did not seem necessary at the time of the loan application. Columns linking to the user's profile (with a URL) and a description (given by the customer) of the demand were removed because they were mostly filled with text data.

We also decided to remove columns that had more than 50% of missing values. This will free up space and make the processing faster. We believe that we could not provide reasonable answers to missing values. Although, we can never be sure about what information might or might not be helpful in retrospect, we must accept that practicality must overcome these possibilities. Moving forward with what we have at hand, we noticed a candidate for the target of our algorithm (what we want to predict) which is the *loan_status* column. The loan status is what we are most interested in because this indicates whether the lender repays the loan in full or not.

Exploratory Visualization

As mentioned in the data exploration section, an important part of the dataset and binary classification in general is the possibility of class imbalance. As a matter of fact, only one seventh of the data is about defaulters, which means that the model has a good chance of getting it right by predicting all good outcomes (right 85% of the time). To focus the attention on what we need to predict we removed the other 7 possibilities (grace period, late 16-30 days, etc.) and by doing so we

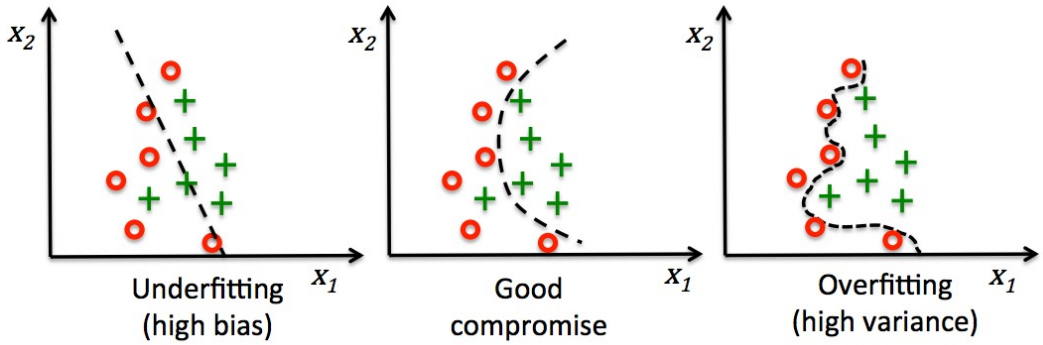
contextualize the difference between the two types of loan status. The class imbalance is a major concern when trying to classify the outcome of various loans and we will address the issue in the implementation section.



Algorithms and Techniques

The model we built uses logistic regression as the main classifier. We also ventured a little bit with random forest to try to improve our predictions, but did not pursue this model as this would mean many parameters fine-tuning.

Logistic regression is one of the most used classification algorithm in industry, due to the fact that it is predicated on simple mathematics and principle. Despite having the word regression in its name, logistic regression is a classification algorithm. As seen on the left side graph, we draw a line trying to classify data points. This would be an example of linear regression and would miss 3 or 4



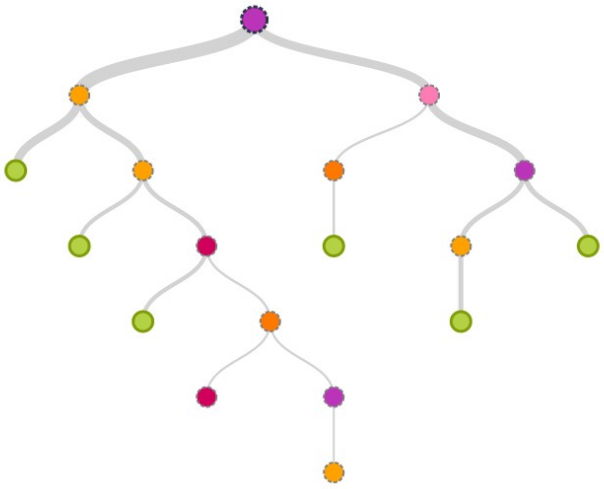
Taken from Python Machine Learning by Sebastian Raschka

points. The model is a linear equation that cannot take into consideration other variables and is used most notably to find the relation between independent and dependent variables (through a linear equation such as y = ax + b, where a and b are real numbers and y and x are the variables). The right graph is a logistic regression using too much variables to over fit the data. This is not a good model in this case because it will most likely fail to generalize to new data. The graph in the middle on the other

hand, uses the sigmoid function which gives a probability of likelihood of the possible binary classification. This helps the model to be more flexible to new data, even though we miss one datum in this case. This is a balance between high bias and high variance.

The reason we started with a logistic regression classifier is because it is fairly simple to implement, make rapid iterations and understand. We did some adjustments and found a good balance between true positives and false positives. The commonly agreed reasons why logistic regression is simple and powerful are ease of implementation, rapid “prototyping”, no parameter tuning, but conversely it can over fit when there is a class imbalance such as our case.

The other model we try on the dataset is build with decision trees. Decision trees are the building blocks of random forests and build complex models by dividing features with boundaries. They are good for general interpretability of their internal processes. By visualizing the splits and depth we can trace back decisions made based on their probability.



Taken from https://blog.bigml.com/2013/04/19/a-new-way-to-visualize-decision-trees/

The most intuitive explanation of random forest is to consider the algorithm as an ensemble of decision trees. This way, we build a strong learning model with a multitude of weak learners (individual decision tree). The model “grows” different decision trees and then uses majority vote weighting to aggregate the prediction of each tree in order to assign the value of the binary classification. The more trees in the model the better the predictions, but also the higher the computational cost. This is often a trade off to be made while using this ensemble model.

Using random forest seemed like a good idea. But once we tried to fit the data it was over fitting and we decided to abandon the random forest models altogether. Random forests with proper depth and an appropriate number of trees can become a very good and predictive model. On the other hand, this can lead to complex parameters tuning and time consuming trial and error to find the right balance. But we had, at this point, a working model with logistic regression and decided to push its limits.

Benchmark

In the introduction we talked about the need of a conservative evaluation of the default rate. We must keep in mind that there is a strong imbalance with the target category of loan repayment in the dataset, because about 6 out of 7 loans are repaid. Meaning that we could lend money all the time (always predicting that the borrower would repay) and be right about 85.95% of the time that the loan would be repaid, but that would mean that the model would not be profitable.

Say we lend $1000 at 10% interest, we would expect a return of $100 on each loan. But if we run the experiment 7 times, we would earn on average $600 (6 x $100) and lose $1000 (the defaulter), we are left with a $400 loss. Hardly a profitable enterprise. The benchmark needs to encompass the weight of the defaulter and the optimization between the true positive rate (good borrowers) and the false positive rate (bad borrowers).

This implies that we need to ensure a viable machine learning model and predict a higher percentage of potential defaulters in order to avoid lending to them. The benchmark must beat the 85.95% average loan repayment. Although “money is left on the table”, a conservative investor would rather prefer a steady return on her investment than suffer the 1 in 7 chance of losing her capital.

III. Methodology

Data Preprocessing

As we said in the initial data exploration, we removed columns that were more than 50% empty. These values cannot be filled in any meaningful ways with only Lending Club data. Some interesting work has been done with the zip code of borrowers and census data (Chang, Dae-oong Kim, and Kondo, Autumn 2015-2016), but for our needs we will refrain going this route. We analyzed the 52 remaining columns with the data dictionary provided by Lending Club. From this, we worked in 4 steps, by analyzing 13 features at each step and trying to figure out whether they were valuable information about the customer.

We also had to deal with a common problem in data science which is “leakage from the future” by making sure that we did not use information that could not be known in advance while taking the decision to lend or not to a customer. This type of information about the future can lead to over fitting and would decrease the ability of our model to make accurate predictions.A noticeable quality of our dataset is the imbalance between the ‘Fully Paid’ and ‘Charged Off’, this implies that the model might

predict a loan outcome based on what is more prevalent in the set, this would lead to an over fitting model and we will need to work around this in the processing phase of our project.

Another part of the preprocessing is the removal of rows with missing values, we had to remove a little more than 700 rows because the informations of the application were incomplete. As we explored the dataset, we saw different data types, and we had to convert – for example – the object data type because it cannot be processed by the machine learning algorithms of Scikit-Learn. We changed values in the employment length to numerical values instead of text. Finally, we encoded the variables once the filtering and cleaning was done.

Implementation

But before we dive in the results of the machine learning algorithms on the dataset, we had to explore, clean and rework the dataset. The reason we had to do this was simply because the data was using a lot of memory and that empty columns slowed the training process of our model. Some other big data tools could have been used, but attempts at lowering compute cost are always welcome in machine learning.

We used a logistic regression algorithm from the library Scikit-Learn as the first algorithm that we fit on the filtered data. We did not put in place a weight on errors, this way, we tried the model with the class imbalance that we noticed previously. By implementing this algorithm on the dataset, we were able to validate our hypothesis about the imbalance and the over fitting that would ensue. If we want to increase the performance of our result, by getting the lowest acceptable level possible to take a decision about lending, we need to add a weight to errors that we want to minimize.

The first implementation gave us an over fitting model with true positive and false negative rates approaching 100%. This is in no way good predictions, because we are lending to all defaulters and losing money. To increase our success rate (measured by a low false positive rate), we added a weight to the errors. This is a simple function of the logistic regression module that can refine the results of the algorithm. As refinement the first weight added was the algorithm’s option of balancing the target’s error.

Refinement

The balanced weight of Scikit-Learn is generally 1 to the number of the opposite occurrences. In this case this would be about 7, but the result that we got after this “normal” re-balancing was not satisfactory for a “production” level model. The true and false positive rates were still too close. We decided to manually add a weight to the error (false positive) to decrease the false positive rate. We did so by trial and error and found that a weight of 10 to 1 for the error gave the best result with the

features chosen previously. Further refinement could be performed, but we left ideas in the improvement section of this report.

IV. Results

Model Evaluation and Validation

The first prediction that we did was predicting that every loans would be paid off. This meant that every true positives were found, but it also meant that we got all the false positive as well. We were effectively getting 100% true and false positive rates. This was not useful predictions and the precision of the model was 85,78% the same as the accuracy. This was predictable. We started by fitting a simple logistic regression classifier out of the box on the dataset and got the same result.

Next, we added cross-validation to our model to try to increase the difference between the two rates. By using a simple 3 folds cross-validation and applying a balanced weight on the errors, we were able to get a 66,73% true positive rate and 39,85% false positive rate. This means we were able to increase the difference between the two rates. We are no longer predicting that all loans will be paid off and thus we increase the likelihood of profitability in our lending. This gave us a precision of 90,99% and an accuracy of 65,80%.

Finally, we increased manually the weight for errors and this gave our model a false positive rate of 7,85% and a true positive rate of 22,66%. This leave many unfunded loans, but reduce the risk a great deal on the investor’s capital. This means that we are wrong about 8% which is a significant drop from the 15% of defaulting loans if we had funded all loans. We now had a precision of 94,56% and an accuracy of 32,54%. We ventured with random forests at the end, but found that the rates did not have a large enough difference to pursue further exploration, but with parameter tuning we could get a working model.

Justification

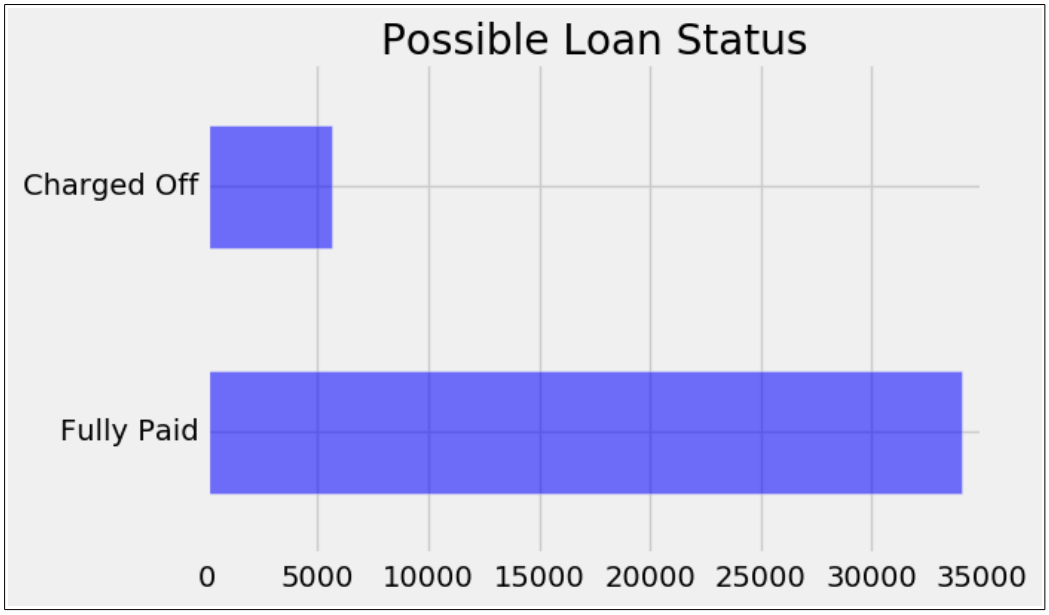
From the conservative investor’s standpoint our last result is fairly good. This is reassuring to find a false positive rate lower than the true positive rate, this means that as long as the pool of potential borrowers is big enough and the interests on the loan are high enough to offset the potential loss, lending is a viable way to invest one’s money. Although, we knew from the start that by refining our criteria to lend money we would lose some good borrowers to prevent lending to defaulters, we have a working model for new loan applications. By following the method, we are investing in only

grade A loans, with very low risk of default and low return on investment (around 6-8%). This can seem low compared to grade G or H loans with interest around 30%, but it can be a good alternative to low yield bonds.

V. Conclusion

Free-Form Visualization

The most important feature of the dataset is the imbalance in outcomes and how we can deal with that. The following visualization is a different way to interpret the first graph showing all the types of outcomes. It is the outcomes reduced to only 2 options and it is telling of the challenge that we faced. The class imbalance – here we can approximate the 1 to 7 difference between the two possible loan status – was the reason why we had to tweak the machine learning algorithms and balance errors differently.



Reflection

The end-to-end process of building a machine learning model is a rigorous process of data analysis, data cleaning, data processing and machine learning algorithms fitting. One interesting aspect of the project was that even with a multitude of columns and data, the information is not always relevant to the problem that we are trying to solve. It obviously showcases the limits of data gathering. Big data does not necessary mean good insights into a problem and a lot of data engineering and processing is needed in order to extract what is relevant. Also, the fine line between information that can be asked of the potential borrower and information that could be discriminatory is not trivial in the

financial industry. The part that was the most enjoyable was the machine learning fitting, because it shows how far we have come from a messy .csv file.

Improvement

Different classification algorithms such as XGBoost, or even different machine learning architecture such as neural networks could provide better results. In at least one known case, it has be used by a FinTech company such as Underwrite.ai (Underwrite, 2016). Deep learning can be another way to discover and push the boundaries of what is possible in predicting the outcome of loans. Along with different algorithms, we could create more features, as the team at Stanford did with the zip code and U.S. census data. We could as well ensemble models (Raschka, 2015) to make better predictions.

On a side note, relaxing the strict requirements could help capturing the false negative market and that could lead to better profitability to the bank that finds a way. This is on the business side and not on the machine learning side, but nonetheless, business problems are a driver in the implementation and deployment of machine learning models. Being able to serve this under-served niche would open opportunities to those customers and potentially increase the profit margin as it had been shown by the Grameen Bank. Machine learning can, in many ways, help us discover hidden value in the world.

References

Lending club statistics. Available at: *https://www.lendingclub.com/info/download-data.action* (Accessed: 15 February 2017).

Tsai, K., Ramiah, S. and Singh, S. (2014) *Peer Lending Risk Predictor*. Available at: *http://cs229.stanford.edu/proj2014/Kevin%20Tsai,Sivagami%20Ramiah,Sudhanshu%20Singh,Peer%20Lending%20Risk%20Predictor.pdf* (Accessed: 23 February 2017).

Raschka, S. (2015) *Machine learning with R - Second edition*. Available at: *https://www.packtpub.com/big-data-and-business-intelligence/python-machine-learning* (Accessed: 15 February 2017).

Chang, S., Dae-oong Kim, S. and Kondo, G. (Autumn 2015-2016) Available at: *http://cs229.stanford.edu/proj2015/199_report.pdf* (Accessed: 13 February 2017).

Underwrite (2016) *Machine learning + big data for credit underwriting*. Available at: *https://www.underwrite.ai/* (Accessed: 15 February 2017).