Loan Default Prediction and Identification of Interesting Relations between Attributes of Peer-to-Peer Loan Applications

Zakaria Alomari, Dmitriy Fingerman
Faculty of Engineering and Computer Science, Concordia University, Department of Computer Science and Software Engineering, Montreal, Quebec, Canada
{zakaria.alomari, dmitriy.fingerman}@gmail.com

Abstract. We present here a study for default prediction of Peer-to-Peer loans and for learning interesting associations between various attributes of the same loan applications which aim at helping the potential investors, all this using Data Mining techniques. In this study, we present background research into use of Data Mining techniques in the domain of loan default predictions. Subsequently, we propose a solution for the stated goals, which consists of exploration, preprocessing and mining of the Peer-to-Peer dataset. Subsequently, we describe experiments which we did following the proposed solution and finally we describe the results that were achieved.

Keywords. Mining of peer-to-peer loan data; Association rules extraction; Data classification.

1 Introduction

Data Mining techniques are frequently used for the problem of loan default prediction. There is a relatively new type of loans, called peer to peer lending, which is growing in popularity nowadays. This type of lending allows borrowers to borrow money from other private investors and not from banks as in traditional lending.

Lending Club is a peer-to-peer lending company located in San Francisco, California. It enables borrowers to create loan listings and investors to fund the listing according to their choice. The loan amounts range from $1,000 to $35,000 and loans are given for a period of 36 or 60 months.

Lending Club’s data is publicly available for download. It is organized into different files grouped by the loan issue years, where each file contains data for 2-4 years. The records in these files have 52 dimensions. Each record contains information about one loan, which includes data about the borrower from the time of loan application, such as state of residence, income and credit history score, and also information about current status of the loan (current, paid off, charged off, etc.), amount of principal and interest paid, next payment date, etc.

There are two goals for this work. The first goal is to find a classification solution that will be as accurate as possible at predicting whether a peer-to-peer loan will be paid off or default. This will help potential investors during evaluations of loan applications to decide whether to invest in a particular loan or not. The second goal of this work is to find and investigate
interesting relations and associations among the attributes of peer-to-peer lending applications to discover possible not obvious hidden knowledge that can be valuable to potential investors.

For finding classification model for default prediction we compared the performance of 7 classifiers with various tuning parameters: Random Forest (RF), Decision Tree (DT), Naïve Bayes (NB), k-Nearest-Neighbors, 1R, AdaBoost and Artificial Neural Network (ANN). For finding hidden associations between various attributes of loan applications we used Apriori algorithm which helped to find interesting and meaningful rules.

We encountered several challenges in this project. The original dataset had 52 attributes and it required a significant effort to explore all the attributes in order to prepare their statistical and visual representations. Next, it was challenging to do preprocessing of the data, especially dimensionality reduction due to the big number of attributes. In the Classification task it was challenging to run the big number of experiments, because at the beginning we were doing it manually and it was taking very long time until we started to use Weka Experimenter which allows designing and executing many experiments with different tuning parameters. Weka Experimenter allowed to work more systematically, but it still took time and effort to run the experiments and analyse the results. Also, there was a limitation on applying multi metrics (support, confidence and lift) in the same execution of Apriori algorithm in Weka because it allows using either support and confidence or support and lifts. Applying multi metrics on the same dataset at the same time would help us to get more accurate results in Association Rules task.

2 Related Work

2.1 Classification

As has been mentioned (Moin & Ahmed 2012), the software can learn to classify the data items into groups in classification method. Classification process includes two steps: first a learning step which is building the classification model and second a classification step itself which is using classifier for classification. In learning, the classification algorithms build the classifier. The classifier is built from the training set and their associated class labels. In classification, the classifier is used for classification. Here the test data is used to estimate the accuracy of classification rules. The classification rules can be applied to the new data tuples if the accuracy is considered acceptable. The classifier-training algorithm uses these pre-classified examples in order to determine all the parameters necessary for proper judgment. The algorithm then encodes these parameters into a model called a classifier. The paper does mention that classification models predict categorical class labels; for instance, we can build a classification model to categorise bank loan applications as either safe or risky and to distinguish borrowers who repay loans promptly from those who do not.

Data mining was proposed in Customer Relationship Management (CRM) (Raju, Bai & Chaitanya 2014). The supervised learning method, Decision Tree, implemented using Classification and Regression Trees (CART) algorithm is used for customer retention. One of the sectors which applied the data mining is Customer Retention in banking which also depends on two methods: Classification Methods and Value Prediction Methods. In Value Prediction Methods, there is another way to classify the application of new loans, it attempts to predict prospect conventional amounts for new loan applications Neural Network and
regression are used for this purpose, and the most common data mining used for customer profiling are: association rule discovery (descriptive) and sequential pattern discovery (predictive). Association plans to determine the relationship between attributes in the database on the emphasis to derive a multi-attribute correlation, satisfying support and trust of the threshold. On the other hand, sequential intended to identify relationships between items over time, this can be considered essentially as association discovery over a temporal database.

In a study (Sawant & Chawan 2013) a comparative analysis of five different classification algorithms was done on three customer loan datasets of small, medium and large sizes with the goal to select an algorithm which the most accurately predicts if a loan will be paid off or defaulted. The algorithms which were compared are: NB, DT, Boosting, Bagging and Random Forests. Last 3 are ensemble methods, which combine several learning algorithms to make better predictions. In Boosting, the ensemble is built incrementally with each model trained on tuples misclassified by the previous model. In Bagging, each model is trained on a random subset of training data and participates in equal-weight voting. Random Forest consists of multiple Decision Trees and outputs the mode of all the trees’ outputs. The paper does not mention which classifiers were used for constructing Boosting and Bagging ensembles. It was found that for a smaller dataset NB showed the best result with 85% of accuracy, while for a medium and large size datasets the Random Forest algorithm showed the highest accuracy of 86% and 72% respectively.

Furthermore, (Zhou & Wang 2012) propose improvements to Random Forest algorithm and shows its efficacy for loan default predicting for big and imbalanced loan datasets. The first modification is the usage of weights on decision trees and employment of voting during prediction process. Trees with lower prediction errors are assigned higher weights which improves the overall prediction accuracy of the ensemble. To handle the imbalanced datasets where classes are different in size, the paper discusses 2 approaches. One is to balance the training data and train the decision trees on it and the second one is to use wrapper sampling algorithms to create Random Forest. Original Random Forest algorithm is sequential, and to improve its performance for bigger datasets the other proposed modification was to make the execution parallel by dividing the forest into sub-forests, execute each sub forest in parallel and at the end to merge the result. The modified Random Forrest was compared to the original one and also to other algorithms (Support Vector Machines (SVM), kNN and DT C4.5) and it was found the most accurate. It improved over the original Random Forest by 0.08% in overall accuracy and by 25% - 50% in performance as 3 implementations of parallelisation where compared.

Another study (Salame 2011) compared the performance of 3 classification algorithms (Logistic Regression (LR), DT and ANN) that were applied for prediction of agricultural loan defaulting for loans which were refinanced and matured in 2006 – 2010. It was found that none of these methods outperformed others in all cases as for different groups (or clusters) of loans different models provided the best classification rate. In particular, the study showed the need for different prediction models for the data from 2006 and 2007-2010 periods because significant variables for the subset of data from 2006 are different from ones for 2007-2010, possibly because of increase of grain prices between these periods. Also, it was found that different models are required for the loan applications with full customer data versus the ones with only partial data. If we extrapolate this conclusion beyond the farming loans area it may mean that each specific loan dataset requires finding the specific model
which works best for it.

Paper (Ramakrishnan, Mirzaei & Bekri 2015) compares the performance of AdaBoost classifier with 4 different base learners with the performance of the single learners. The learners that were used are: LR, ANN, DT and SVM. AdaBoost is a type of boosting classifier (Sawant & Chawan 2013). It uses a baseline classifier K times where K is specified by the user and in each iteration the training is focused on a different subset of learning examples giving preference to the tuples misclassified by previous classifiers. This classifier works using weights of training examples. Initially, all examples are given same weight which is $1/n$ where n is the number of tuples in the training set. After that, in each iteration, correctly classified tuples are given less weight and incorrectly classified ones are given more weight to increase their chances of being chosen in the next training iteration. This setup helps to learn the “hard” examples. After completion of training iterations single classifiers obtained from the K iterations are combined together to form a highly accurate strong classifier. The strong classifier is formed as a linear combination of the K classifiers weighted by the error of each classifier from the corresponding iteration. Results show that AdaBoost in all cases outperformed the single classifiers. Also, with and without AdaBoost, SVM was the best in accuracy, LR was second, DT was third and ANN was the worst.

Study (Zhao & Hassan 2013) compared performance of 3 classification algorithms for predicting pollution in new Hong Kong rural area, namely ANN, kNN and AdaBoost M1. Although that study is not related to loan default prediction problems, we are interested in it because it reveals an interesting observation about AdaBoost M1 which we want to verify in our paper in order to improve our classification results. In that study, AdaBoost M1 classifier was found to be more accurate than ANN and kNN. Unfortunately, it is not explicitly stated which base classifier was boosted in those experiments. Also, that study researched to find the most optimal number of iterations in the range of 1-100 and found that the highest accuracy is achieved with 80 iterations.

2.2 Association Rules

Paper (Mak, Ho & Ting 2011) aims to obtain a better understanding and insights into investment behaviour which was achieved by using Financial Data Mining Model (FDMM) which in turn depends on clustering analysis and Association Rules (ARs). Cluster analysis is commonly used for identifying interesting distributions and patterns in the data before mining the ARs. The ARs aim to discover the relations between variables in large databases. More importantly, the FDMM is composed of three modules, specifically; the Data Selection and Preprocessing Module (DSPM), the Clustering Module (CM) and the Rules Discovery Module (RDM). The RDM stage aims to discover the relationships in a specific cluster. Therefore it can directly extract an input data set from the CM to generate useful rules. In contrast, the Apriori algorithm is applied in RDM module to find the frequent patterns, correlations and associations. After the generation of the rules, the rules allow management to make an evaluation. Then, the sales and marketing department can use such rules for decision making in regard to a specific cluster.

Furthermore, study (Martin, Manjula & Venkatesan 2011) develops an ontological model from financial information of an organisation by analysing the semantics of the financial statement of a business. The financial ontological model of the relationship between financial data is discovered using data mining algorithms. Consequently, a new business intelligence
(BI) model is developed to predict the bankruptcy, by combining financial domain ontological model with Association Rules mining algorithm and bankruptcy prediction model (Altman Z-score model). In addition, the author proposed Association Rule Mining Algorithm to extract the new knowledge from the relevant data obtained through financial ontology tree, to come up with the necessary decisions. To discover the new knowledge, data mining is applied in two steps: cluster analysis and Association Rules mining algorithm. As a result, the Association Rules mining Algorithm augments the efficiency of the proposed method by providing relevant results based on the association between the businesses’ financial statements. Thus BI model can be used effectively.

Moreover, study (Farajian & Mohammadi 2010) presents a new two-stage framework of customer behaviour analysis that integrated a K-means algorithm and Apriori Association Rules inducer. In other words, there is two stage approach for customer behaviour analysis of implicit knowledge using bank profile data of the customers and their debit cards transactions. In the first stage, K-means algorithm was used to divide customers into groups of customers based on customer behaviour and RFM (recency, frequency, monetary). In the second stage, Apriori was used to characterise the groups of customers by creating customer profiles; it is mainly used to find out the Association Rules and meaningful relationships between the huge number of items or features that occur synchronously in the database, so Apriori mechanism was used for finding relevant clustering rules. Therefore, rule candidates are considered useful and become Association Rules only if their support is larger than minimum support (minsupp) threshold and whose confidence is larger than minimum confidence (minconf) threshold. Accordingly, after briefly reviewing the customer profiles using the Association Rule inducer, the customers with a higher customer behaviour or RFM might be the target customer groups of precedence.

3 Problem Definition

A peer-to-peer lending platforms have few restrictions on borrower eligibility, which results in adverse selection problems and high borrower default rates. Furthermore, some investors view the lack of liquidity for these loans, most of which have a minimum three-year term, as undesirable.

Peer-to-peer loans have terms of 3-5 years, which means actual returns are unknown until the full portfolio of loans has matured and paid off or defaulted. Lending Club delays declaring a loan in default for months after the borrower has stopped paying it. On the other hand a new problem has appeared which affects the approval process by putting more pressure on it, that all was a result of pouring millions of dollars into peer-to-peer loans by investors.

For these reasons, deciding to invest into a peer-to-peer loan is a tough problem for investors and this study proposes solutions that aim at helping the potential investors.

4 Proposed Solution

In high level, our solution consists of the following stages:

- Data Exploration: learning of initial characteristics of the dataset using statistical and visual tools.
Data Preprocessing: this step is required to prepare the dataset for use in Classification and Association Rules Mining algorithms and it includes data cleaning, dimensionality and data reduction and attributes’ transformations.

Data Mining: Classification using 7 classification algorithms (Random Forest, DT, NB, kNN, 1R and ANN) and Association Rules Mining using Apriori algorithm to achieve the aims of this project. Following sections describe the major steps of the solution in detail.

4.1 Data Exploration

Descriptive statistics help to understand our data in terms of its distribution. To examine our data’s distribution, we will use the measures of central tendency and dispersion. Before we calculate a measure of central tendency and dispersion, let’s define what we mean by distribution; the ideal distribution of data is called the normal distribution. For a normal distribution, all measures of central tendency are the same, and there is an equal number of observed data points on either side of these measures of central tendency. A histogram is a great way of initially visualizing our data’s distribution because we can get a sense of the central tendency and dispersion of data around that centre before calculating any statistics.

A measure of central tendency: having results of statistical measurements that reflect the distribution of attributes allows comparing individual sample data points to the observed values which are represented by a measure of central tendency. The following are three measurements of central tendency: Mean, Median and Mode.

Measures of dispersion: measures of dispersion are important for describing the spread of the data, or its variation around a central value. For example, two distinct samples may have the same mean or median, but completely different levels of variability. A proper description of a set of data should include both of these characteristics. The following are measures of dispersion: range, variance, and standard deviation.

Visualizing our data: when we are presenting a measure of central tendency and dispersion, it is helpful to provide a visualisation of our data’s distribution in addition to the plain numbers. Histograms and box plots can help illustrating data’s distribution. Frequency histogram, which shows the frequency of attribute’s values over a series of intervals that covers the entire range of the attribute; a box plot also illustrates the distribution of the data. A box plot is made up of the following values derived from the dataset: median, minimum value, maximum value, quartile 1 value, and quartile 2 values. In other words, descriptive statistics, histograms and box plots together help describe and better understand the nature of our data.

4.2 Data Preprocessing

Cleaning of missing values: some of the attributes in our dataset had missing values. We analyzed these attributes and replaced the missing values with ones which make sense in the context of those attributes using Excel. The reason for doing it in Excel was following: the missing values were Numeric attributes, but we could not assign numeric values to where they were missing because it would be logically incorrect.

After getting rid of missing values we performed manual discretisation to those attributes also in Excel.
Reduction of irrelevant attributes: our original dataset contained 52 attributes. Some of them are from the time of loan application and others are related to its current operational status after funding. For our goals, we are interested only in the attributes from the time of loan application. Also, some of the attributes had the same value in all the tuples. For these reasons, we reduced the irrelevant attributes.

Reduction of highly correlated attributes: we identified pairs of a highly correlated attribute using the Visualization Tab in Weka Explorer and removed one from each pair.

Attribute Transformations: at this stage, we transformed some of our attributes into other data types, and replaced other attributes which were not useful for our Data Mining tasks with the derived attributes.

Removal of Outliers: It is known that outliers can cause problems in classification algorithms. We identified outliers using Weka’s Interquartile Range Filter and removed the tuples with outliers.

Discretization of numeric attributes: Besides the three columns for which we had to do manual discretisation in Excel, we had several Numeric columns in our dataset that needed to be discretised before using Classification and Association Rules Mining algorithms. For classification experiments we did not discretise these numeric columns at the Preprocessing stage, because during Data Mining stage we experimented with different discretisations (supervised, unsupervised equal-width and equal-frequency) that where done as a filter step just prior to running the classification algorithm using Weka’s FilteredClassifier. However, we discretised these numeric columns using unsupervised equal-frequency discretisation with 5 bins for the Association Rules Mining task. We selected this discretisation for Association Rules Mining task because in classification tasks we had more success with the equal-frequency binning and therefore believe that it is most optimal for our dataset.

Data Balancing: Classification algorithms need to work on balanced datasets. Therefore we balanced our dataset for the classification task using Weka’s supervised Resample filter. However, we used the original unbalanced dataset for the Association Rules Mining task because Association Rules Mining is not a supervised technique.

4.3 Data Mining – Classification

We conducted a long list of experimentations with 6 classification algorithms under a variety of different tuning parameters on our dataset and for each one analysed the results: Random Forest, NB, DT, kNN, 1R and ANN.

For each of the 6 classifiers, we selected the winning combination of tuning parameters which lead to the most accurate result and used AdaBoost 1M algorithm on those combinations in order to check if it will improve the results even more. The adaboost M1 algorithm uses several instances of another baseline classifier, whose output it combines into a weighted sum to produce the final output of the algorithm.

We designed all Classification experiments using Weka Experimenter. It allows configuring and executing a list of tasks with different parameters and with the desired number of repetitions for each task. After completion of all the tasks it can analyse the results and display accuracy of each task and the standard deviation of accuracy.

In all experiments, we used the setting for splitting the dataset to training and testing as follows: 66% training and 34% testing. This is the default setting in Weka and also it is
considered as valid splitting in Data Mining in general. For all Classifiers, we used different tuning parameters.

For all classifiers we experimented with different Feature Selection Methods: with all attributes (without feature selection), with Wrapper feature selection and with Filter feature selection.

For every classifier we experimented with different types of Discretizations. To achieve this we used FilteredClassifier because it allows both are selecting the desired discretisation method and the classification algorithm to be executed after discretization. In our experiments we used 3 types of Discretizations: supervised entropy-based discretization, and 2 types of unsupervised discretisation methods, equal-frequency and equal-width binning. For supervised discretization it was very important to use FilteredClassifier because it discretizes separately training and testing data, otherwise it would be a pre-processing cheating.

Tasks that didn’t include feature selection were designed using FilteredClassifier with the desired discretisation method and classification algorithm.

Tasks that included Wrapper Feature Selection were configured as follows: 1. FilteredClassifier with the desired discretisation method and with AttributeSelectedClassifier as the classification method. 2. In AttributeSelectedClassifier we used the desired classification algorithm and in the feature evaluator we used ClassifierSubsetEval (Wrapper Feature Selection) with the same classification algorithm.

Tasks that included Filter Feature Selection were configured same as tasks without feature selection, but with a dataset which already had only N best attributes according to Filter feature selection method. Such dataset was produced prior to the experiments using Weka Experimenter, using FilteredAttributeEval method from within Weka Explorer.

4.4 Data Mining – Association Rules

The goal of mining association rules is to discover all association rules that have support and confidence greater than or equal to the user-specified minimum support (minsup) and minimum confidence (minconf).

Association rule mining is a data mining task that discovers relationships among items in a transactional database. Following key parameters are used to generate valuable rules:

- **Support(s)** of an association rule is defined as the percentage of records that contain X ∪ Y to the total number of records in the database. The count for each item is increased by one every time the item is encountered in different transactions in database during the scanning process. Support(s) is calculated by the following formula:

  \[
  \text{support} (X \Rightarrow Y) = \frac{\text{Prob} \{X \cup Y\}}{\text{Prob} \{X\}} \quad (1)
  \]

- **Confidence** of an association rule is defined as the percentage of the number of transactions that contain X ∪ Y to the total number of records that contain X. Confidence is calculated by the following formula:

  \[
  \text{confidence} (X \Rightarrow Y) = \frac{\text{Prob} \{X|Y\}}{\text{Prob} \{X\}} = \frac{\text{support} (X \cup Y)}{\text{support}(X)} \quad (2)
  \]
• **Lift** is a measure of the improvement in the occurrence of the Y given the X: it is the ratio of the conditional probability of the Y given the X, divided by the unconditional probability of the Y. Lift is calculated by the following formula:

\[
\text{lift}(X \implies Y) = \frac{\text{confidence}(X \implies Y)}{\text{support}(Y)} = \frac{\text{support}(X \cup Y)}{\text{support}(Y) \cdot \text{support}(X)}
\]

(3)

Using Weka for association rule mining uses a single minsup and a single minconf in the mining process. This is not appropriate for our dataset, the reason behind that is the class distribution of our data is often extremely imbalanced. Using a single minsup cause the following issue; if the minsup is set too high such as 95%, we may not find those rules that involve the minority class, which is the class that we are interested in, which would mean that our dataset doesn't have any rules that meet the minimum support of 95%. So as to find rules that involve the minority class, we have to set the minsup very low such as 5%. This may cause a combinatorial explosion because the majority class may have too many rules and most of them are overfitted with many conditions and covering very few data cases. These rules have little predictive value and they also cause increased execution time.

Using a single minconf also cause the following issue; if we set the minconf at 96%, we may not be able to find any rule because it is unlikely that the database contains reliable rules with such a high confidence. If we set a lower confidence, say 50%, we will find many rules that have the confidence between 50%-95% and such rules are meaningless. We can solve these issues by using different minsups and minconfs for rules of different classes.

Furthermore, Weka allows the use of different metrics such as lift. Lift is one more parameter of interest in the association analysis, and it is nothing but the ratio of Confidence to Expected Confidence. Hence Lift is a value that gives us information about the increase in probability of the (consequent) given the (antecedent) part. A lift ratio larger than 1 implies that the relationship between the antecedent and the consequent is more significant than would be expected if the two sets were independent. The larger the lift ratio, the more significant the association.

In this project, when evaluating the association rule interestingness, various measures can be used to help finding the rules that give maximally useful information to us. Some of the proposed association rule interestingness measures are confidence and lift.

5 **Experiments**

For our experiments we used the Lending Club data which is publicly available. Since the usual loan timeframe is 36 months, we used the dataset for 2012-2013 because it contains more loans with a determined final status (either paid off or defaulted) comparing to the datasets from 2013-2014 or 2015. Such records are good for the 2 aims of our work. This dataset has 188,124 records and 52 attributes, and the file size is 104 MB.

After balancing the original highly imbalanced dataset we left 22% of the original amount of tuples with 20,003 records with Class=N and 20,133 records with Class=Y which makes our dataset ready for classification algorithms. We used this balanced dataset only for classification task and the unbalanced version for Association Rules Discovery, because for Associations Rules Discovery the dataset is not needed to be balanced.
Machines that we used for experiments were regular consumer-grade laptops with I7 CPU, RAM 8-32 GB and Windows 7 OS.

5.1 Preprocessing

5.1.1 Cleaning of missing values

Three attributes in the original dataset had missing values in many tuples. We replaced them with the value ‘Never’ which means that the client never had the corresponding conditions. These attributes were following: mths_since_last_delinq, mths_since_last_major_derog and mths_since_last_record.

5.1.2 Reduction of irrelevant attributes

We removed 8 attributes which were irrelevant for our tasks either because of those where ID attributes which had a different value in each tuple or because the attributes had an absolute majority of values same in all tuples and the rest did not correlate prediction classes. Also, for the goals of our work we are interested only in the attributes which are known at the time of loan application using which we can find data mining solutions for helping investors in determining whether they should invest in a given loan based on how likely it is that the loan will default and whether the loan’s borrower is likely to be a good borrow. Therefore, we removed 20 attributes which have meaning only after a loan has been funded by investors.

5.1.3 Reduction of highly correlated attributes

In total, 4 attributes were removed because they had a strong correlation to another attribute. The correlations were identified using Weka’s Visualization Tab in Weka Explorer.

5.1.4 Reduction of highly correlated attributes

All attribute transformations were done in Excel due to Weka’s limited functionality in performing attribute transformation in Weka Explorer.

- int_rate: initially this attribute was in the format: xx.xx% Where x is a digit. We removed the percent symbol in Excel and converted the attribute into the Numeric format.
- Class: this attribute replaces the loan_status attribute to be the prediction classes attribute. Loan_status has the following values: Current, Fully Paid, In Grace period, Late (16-31), Late (31-120), Default and Charged Off. We assumed that loans in status Current, Fully Paid and In Grace Period would be eventually paid off and loans in statuses Late (16-31), Late (31-120), Default and Charged Off will default and the value of the new attribute was assigned to either Y or N based on this split.
- desc_length: this attribute replaces the desc attribute. desc attribute is the description of the loan as provided by the borrower. We could not use this attribute because it has too many different values, but the length of the description can be used for classification.
- Years_since_earliest_cr_line: this attribute replaces the attribute earliest_cr_line which had the following format: MMM-YY. We converted these values to 2013 – YY which means the number of years since earliest credit line prior to 2013 because we are working with the
loans from years 2012-2013.

5.1.5 Removal of outliers

Table 1 lists all the attributes which had outliers or extreme values that were found in Weka using the Interquartile Range filter prior to discretisation. We removed all the tuples which had outliers because of annual_inc, desc_len, years_since_earliest_cr_line or open_acc attributes. We also removed delinq_2y and pub_rec attributes from the dataset because in those attributes all the none-zero values were found as extreme values and since there were found too many instances of them we decided that it is better to reduce these attributes instead of removing thousands of tuples. Also, the vast majority (84% and 93% consequently) of values were zeros. Furthermore, none-zero values did not have a correlation to prediction classes.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th># of Outliers</th>
<th># of Extreme Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>annual_inc</td>
<td>1,853</td>
<td>529</td>
</tr>
<tr>
<td>desc_len</td>
<td>2,292</td>
<td>330</td>
</tr>
<tr>
<td>delinq_2y</td>
<td>-</td>
<td>29,721</td>
</tr>
<tr>
<td>years_since_earliest_cr_line</td>
<td>526</td>
<td>-</td>
</tr>
<tr>
<td>open_acc</td>
<td>196</td>
<td>5</td>
</tr>
<tr>
<td>pub_rec</td>
<td>-</td>
<td>17,463</td>
</tr>
</tbody>
</table>

5.1.6 Discretization of numeric attributes

For attributes loan_amnt, int_rate, annual_inc, desc_length, dti, delinq_2yrs, years_since_earliest_cr_line, open_acc, pub_rec for classification experiments we applied different discretisations, including unsupervised equal frequency and equal width discretisation with a different number of bins and also supervised discretisation. We were doing supervised discretisation using Weka’s FilteredClassifier which splits the data into training and testing sets and performs supervised discretisation separately on both to ensure no cheating. For our Association Rules Discovery task we used equal-frequency 5 bins discretisation.

For attributes mths_since_last_major_derog and mths_since_last_records we applied manual discretization using Excel into 6 equal-length bins with the following values: Never, 0-33, 34-66, 67-99, 100-133 and 134+. This could not be done via Weka Explorer because the missing values could not be converted into zeros and fall into the range of (0-N) because zeros here have totally different meaning. So, first zeros were converted to value ‘Never’ and discretised the attributes manually as explained above. In Excel, it was hard to do equal-frequency binning that is why we used equal-width binning.

5.1.7 Data Balancing

We also balanced out dataset before using it in classification algorithms, because it is
important to have the data balanced on the prediction classes for classification algorithms. Our Table 2 displays the attributes left after dimensionality reduction.

**Table 2. Attributes after Dimensionality Reductions**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Original[y] Transformed [N]</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>loan_amnt</td>
<td>Y</td>
<td>The listed amount of the loan applied for by the borrower.</td>
<td>Numeric</td>
</tr>
<tr>
<td>term</td>
<td>Y</td>
<td>The number of payments on the loan. Values are in months and can be either 36 or 60.</td>
<td>Nominal, 2 values</td>
</tr>
<tr>
<td>int_rate</td>
<td>Y</td>
<td>Interest Rate on the loan.</td>
<td>Numeric</td>
</tr>
<tr>
<td>grade</td>
<td>Y</td>
<td>LC assigned loan grade. Values from A to G.</td>
<td>Nominal, 7 values</td>
</tr>
<tr>
<td>emp_length</td>
<td>Y</td>
<td>Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years.</td>
<td>Nominal, 12 values</td>
</tr>
<tr>
<td>home_ownership</td>
<td>Y</td>
<td>The homeownership status provided by the borrower during registration. Values are: RENT, OWN, MORTGAGE, OTHER.</td>
<td>Nominal, 5 values</td>
</tr>
<tr>
<td>annual_inc</td>
<td>Y</td>
<td>The annual income provided by the borrower during registration.</td>
<td>Numeric</td>
</tr>
<tr>
<td>verification_status</td>
<td>Y</td>
<td>Indicates if income was verified by LC, not verified, or if the income source was verified.</td>
<td>Nominal</td>
</tr>
<tr>
<td>desc_length</td>
<td>N</td>
<td>The length of loan description as provided by the borrower.</td>
<td>Numeric</td>
</tr>
<tr>
<td>purpose</td>
<td>Y</td>
<td>A category provided by the borrower for the loan request.</td>
<td>Nominal, 12 values</td>
</tr>
<tr>
<td>addr_state</td>
<td>Y</td>
<td>The state provided by the borrower in the loan application.</td>
<td>Nominal, 49 values</td>
</tr>
<tr>
<td>dti</td>
<td>Y</td>
<td>A ratio calculated using the borrower’s total monthly</td>
<td>Numeric</td>
</tr>
<tr>
<td>Feature</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>years_since_earliest_t_cr_line</td>
<td>Numeric</td>
<td>A number of years since the borrower's earliest reported credit line was opened, prior to 2013.</td>
<td></td>
</tr>
<tr>
<td>mths_since_last_record</td>
<td>Numeric</td>
<td>The number of months since the last public record.</td>
<td></td>
</tr>
<tr>
<td>open_acc</td>
<td>Numeric</td>
<td>The number of open credit lines in the borrower's credit file.</td>
<td></td>
</tr>
<tr>
<td>initial_list_status</td>
<td>Nominal</td>
<td>The initial listing status of the loan.</td>
<td></td>
</tr>
<tr>
<td>mths_since_last_major_dereg</td>
<td>Nominal</td>
<td>Months since most recent 90-day or worse rating.</td>
<td></td>
</tr>
<tr>
<td>class</td>
<td>Nominal</td>
<td>Classification whether a loan will default (Y/N).</td>
<td></td>
</tr>
</tbody>
</table>

### 5.2 Classification

The goal of these experiments was to find the most accurate classifier for the loan default prediction for this dataset using the proposed solution.

**Random Forest:** We have done several trials with Random Forest with different parameters: executions with supervised and unsupervised discretisations (equal-frequency and equal-width), with all attributes and with top 10 according to Filter attribute selection. We were not able to use the Wrapper Feature Selection method because Weka was not able to complete such tasks. Also, we experimented with 100 and 200 trees and with different seeds (1 and 2).

In the experiments without attribute selection the best result was 71.75% and it was achieved with unsupervised equal-frequency 5 bins discretisation with 200 trees and seed equal to 2. The results were worst for the runs with supervised discretisation. For all of them the accuracy was below 70% and for all the executions with unsupervised discretization the results were all above 70% of accuracy. Runs with 200 trees always performed better than same configurations, but with 100 trees.

With 10 best features according to Filter attribute selection method the results are less good then with all the attributes. The best accuracy with Filter attribute selection was 69.80%. The best result was achieved with 7 equal-frequency unsupervised discretisations with 200 trees and seed equal to 1. Overall, with Random Forest the best result was achieved without using feature subset selection.

**Naïve Bayes:** With NB we experimented with all attributes, as well as with Wrapper and Filter attribute selection methods. The best result was achieved with unsupervised equal-
width 5 bins discretisation and it was 61.44% accurate. The worst results were achieved with unsupervised equal-frequency discretisations and it was 60.74% accurate.

**Decision Tree:** we experimented with J48 Decision Tree classifier which is an implementation of C4.5 Decision Tree. With this classifier, the lower the confidence factor, the more pruning is done. We used different confidence factors and saw that with higher confidence factor the accuracy is higher. With confidence factor of 0.15 the best accuracy is 62.12% and with a confidence factor of 0.25 it is 63.39%. It means that when less pruning is done the accuracy improves. However, it is known that the less pruning is done, the more overfitted becomes the model and it can cause bad results during classification. Probably, in our case the accuracy improves when pruning is decreased (=when model becomes more overfitted) because our training data is very close in characteristics to our testing data, there is no something unexpected that will show worse performance because of overfitting.

After that we experimented without binary splits and with without pruning. Runs without pruning showed significantly better result than with pruning and the best accuracy for this classifier, which is 65.39%, which further supports our understanding that overfitting improves the results because our training and testing data have close characteristics which prevent from the showing of typical problems related to overfitting. We also experimented with binary splits and with a pruned tree with a confidence factor of 0.25. Binary splits improve the accuracy of J48 algorithm which almost reaches the accuracy achieved without pruning: 65.38% vs. 65.39%. However, we noticed that with binary splits the algorithm was running for much longer than without binary splits.

**K-Nearest-Neighbors:** We run experiments with 5, 10 and 15 neighbours using IBk classifier which implements K-Nearest Neighbors Lazy Learning classifier. For this classifier, we used only all attributes and Filter feature selection methods because IBk algorithm was running for a very long time with Wrapper feature selection method.

The difference of the best result with 5, 10 and 15 nearest neighbours was very small: 61.18%, 61.05% and 61.08% respectively. However, it is interesting that the best result of 61.18% was achieved with the lowest number of nearest neighbours which was 5. Also, it is interesting that for all 3 numbers of nearest neighbours, the best result was always achieved with supervised discretization and the worst result was always achieved with unsupervised 7 bins equal frequency discretisation. Also, there seems to be a direct relation between the worst result and the number of nearest neighbors (NN): the bigger number of nearest neighbors, the better was the worst result.

**1R:** Experiments with 1R classifier showed that with any type of attribute selection (All attributes, Wrapper, Filter and 1R Filter) the accuracy was exactly the same. This can be explained because 1R classifier uses only 1 feature out of all that describes the class the best. In our case this feature is int_rate and it was selected in the list of the best features by all the feature selection methods that we used here. The best result was 59.99% of accuracy.

**Artificial Neural Network:** we tried ANN with different tuning parameters: number of epochs (500 and 1000), learning rate (0.2 and 0.3), momentum (0.2 and 0.3) and number of hidden layers ((# of attributes + # of classes) / 2 and # of attributes + # of classes). Since ANN take very long time to train, we run our experiments on a 10% balanced sample of the dataset on which we were running other classifiers, with around 2,000 instances of each of the two classes. With 500 epochs, the best result that we achieved was 60.55% of accuracy and with
1,000 epochs the best result that we achieved was 60.48%. In both cases, the accuracy of ANN could outperform only the results of 1R classifier.

**AdaBoost:** In these experiments, we wanted to check if AdaBoost can boost the winning configurations of classifiers found in previous experiments in order to push the best accuracy higher. Table 3 summarises the best configurations of baseline classifiers and of their use in AdaBoost. AdaBoost with Random Forest showed the less good result (71.61%) comparing to Random Forest by itself (71.75), although very close. Naïve Bayes and K-Nearest-Neighbors showed exactly same results with AdaBoost as by itself (61.44%). Decision Tree J48 showed big improvement with AdaBoost (68.85%) comparing to J48 by itself (65.38%). OneR showed a little improvement with AdaBoost (60.98%) comparing to OneR by itself (59.99%).

Overall, AdaBoost did not improve the best result which was achieved with Random Forest, but it did improve the performance of some classifiers, most notably of Decision Tree J48.

To summarise, the best result was achieved with Random Forest (71.75%) with following parameters: unsupervised equal frequency discretisation with 7 bins, without using attribute subset selection with 200 trees. The ranking of the accuracy of the 6 baseline classifiers is following: 1. Random Forest. 2. Decision Tree J48. 3. Naïve Bayes. 4. K-Nearest-Neighbors. 5. Artificial Neural Network. 6. OneR.

<table>
<thead>
<tr>
<th>Subset Feature Selection</th>
<th>Discretization (Supervised / Unsupervised)</th>
<th>Discretization (equal-freq. / equal-width)</th>
<th># of bins</th>
<th>Seed</th>
<th>Weak Learner</th>
<th>Weak Acc. %</th>
<th>Weak stdv</th>
<th>AdaBoost Acc. %</th>
<th>AdaBoost stdv</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>Unsuper.</td>
<td>Equal-frequency</td>
<td>7</td>
<td>1</td>
<td>Random Forest, 200 trees, seed 2</td>
<td>71.75</td>
<td>0.32</td>
<td>71.62</td>
<td>0.14</td>
</tr>
<tr>
<td>-</td>
<td>Unsuper.</td>
<td>Equal-frequency</td>
<td>7</td>
<td>2</td>
<td>Naïve Bayes</td>
<td>61.44</td>
<td>0.33</td>
<td>61.44</td>
<td>0.33</td>
</tr>
<tr>
<td>Wrapper</td>
<td>Unsuper.</td>
<td>Equal-width</td>
<td>5</td>
<td>1</td>
<td>Decision Tree J48, pruned, conf. level 0.25, binary splits</td>
<td>65.38</td>
<td>0.27</td>
<td>68.85</td>
<td>0.53</td>
</tr>
<tr>
<td>Wrapper</td>
<td>Unsuper.</td>
<td>Equal-width</td>
<td>5</td>
<td>2</td>
<td>K-Nearest Neighbors, k=5</td>
<td>61.18</td>
<td>0.47</td>
<td>61.18</td>
<td>0.47</td>
</tr>
</tbody>
</table>
5.3 Association Rules

Lending Club analysis often produces very large numbers of rules, especially when many items are involved. On the other hand there are constraints (Minimum Support, Minimum Confidence and Lift) are used to constrain the potential number of rules generated by screening out those that don't meet minimum benchmarks, as well these constraints can help manage useful rules generation, but even then there can be hundreds or thousands of rules that meet user specified screening criteria [8,10]. Once the rules are generated and an analyst is faced with gleaning essential and actionable information, there is no single best way to decide which rules are the "interesting" rules.

Our experiments are executed with Apriori algorithm on a dataset to finding interesting rules using low support and high confidence, also low support and high lift.

5.3.1 Interesting rules found with low support and high confidence

In order to generate interesting rules, (Farajian & Mohammadi 2010) we conducted three experiments setting the minimum support threshold to 5%, 10% and 15% based on three high confidence thresholds 95%, 97% and 100%, then compared the resulting association rules and their interesting-ness. Table 4 displays the result of generated rules with our experiments.

<table>
<thead>
<tr>
<th>Support %</th>
<th>Confidence %</th>
<th>Number of Rules Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>95%</td>
<td>2010</td>
</tr>
<tr>
<td></td>
<td>97%</td>
<td>1064</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>0</td>
</tr>
<tr>
<td>10%</td>
<td>95%</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>97%</td>
<td>193</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>0</td>
</tr>
<tr>
<td>15%</td>
<td>95%</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>97%</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>0</td>
</tr>
</tbody>
</table>

From Table 4, we can see that when the minimum support are going up (5%, 10% and 15%), the number of rules are reduced. It is because that the higher support threshold prunes more items in the procedure of association rule generating generated rules are reduced as expected. In our experiments, we found that support values higher than (15%) generate too few rules, leaving many potentially interesting rules out of the rule set. As well we found that in all our experiments the rules whose confidence was lower than 97% were extremely rarely truly interesting according to our goal (Farajian & Mohammadi 2010).

In our paper, due to the size of the dataset and the scattered nature of the data, the minimum...
support threshold was set low (15%) to detect rare patterns, while the confidence parameter was set high (97%) to ensure accuracy, then there are 36 strong association rules generated from dataset, respectively. The more interesting twelve rules were extracted:

**Rule 1:** Loans that didn’t default and were taken for 36 months with interest rate from 10.4 to 13.08, were assigned risk grade B by Lending Club at time of loan application.

**Rule 2:** The Lending Club assigns a loan to be with risk grade B, if the client has a loan with interest rate from 10.4 to 13.08 and he never had 90-day or worse rating.

**Rule 3:** Loans with interest rate from 10.4 to 13.08 with status not defaulted and when client never had a public record, were assigned risk grade B by Lending Club at time of loan application.

**Rule 4:** The loan is assigned to be with risk grade B, in case the client took a loan for 36 months with interest rate from 10.4 to 13.08 and he never had a public record.

**Rule 5:** If the initial listing status of the loan is a fractional investment with interest rate from 10.4 to 13.08, then the Lending Club would definitely assign the loan to be with risk grade B.

**Rule 6:** The loan duration is 36 months when the client’s home ownership status is rent and his income is not verified.

**Rule 7:** The loan duration is 36 months if the client’s loan status not defaulted and for the purpose of debt consolidation also his income is not verified.

**Rule 8:** If client’s income is not verified and his loan does not default, and also no loan description is provided, then the number of payments on the loan is 36 month.

**Rule 9:** Loan period is 36 months for payment in case of: the client’s income is not verified and the loan purpose of debt consolidation, add to that he never had a public record.

**Rule 10:** When the client’s income is not verified and he never had 90-days or worse rating, also his loan not defaulted with the initial listing status of the loan is a fractional investment, then the number of payments on the loan is 36 month.

**Rule 11:** The number of payments on the loan is 36 month, when the client’s income is not verified and he never had 90-day or worse rating and even never had a public record, also his loan not defaulted with the initial listing status of the loan is fractional investment.

**Rule 12:** When the loan not defaulted and its amount listed from 1,000 to 7,012.5 for the client who never had a public record, then the number of payments of his loan is 36 month.

Table 5 shows support and confidence for the 12 rules. As shown in Table 5, the 12 rules have a confidence level greater than of 97%. It implies that such rules are highly reliable in indicating the success rate to find interesting relations and associations among the attributes of peer-to-peer lending applications.

**Table 5. Summary of Interesting rules - low support and high confidence**

<table>
<thead>
<tr>
<th>Rule #</th>
<th>Conf. %</th>
<th>Support %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100%</td>
<td>16%</td>
</tr>
<tr>
<td>2</td>
<td>100%</td>
<td>16%</td>
</tr>
<tr>
<td>3</td>
<td>100%</td>
<td>16%</td>
</tr>
<tr>
<td>4</td>
<td>100%</td>
<td>16%</td>
</tr>
</tbody>
</table>
5.3.2 Interesting rules found with low support and high lift

In order to generate interesting rules, (Mak, Ho & Ting 2011) we conducted three experiments setting the minimum support threshold to 5%, 10% and 15% based on three high lift thresholds 3, 4 and 5 then compared the resulting association rules and their interestingness. Table 6 displays the result of generated rules with our experiments.

Table 6. Results of Comparison

<table>
<thead>
<tr>
<th>Support %</th>
<th>Lift</th>
<th>Number of Rules Produced</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>3</td>
<td>5032</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3022</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>670</td>
</tr>
<tr>
<td>10%</td>
<td>3</td>
<td>454</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>326</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>68</td>
</tr>
<tr>
<td>15%</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

From Table 6, we can see that when the minimum support are going up (5%, 10% and 15%), the number of rules are reduced. The reason behind it is a high support threshold keeps away from generating more number of rules, but at the cost of losing interesting rules of low support. In our experiments, we found that support values higher than (15%) generate too few rules, leaving many potentially interesting rules out of the rule set. On the other hand, we found that in all our experiments the rules whose lift was lower than 5 were extremely rarely truly interesting according to our goal (Mak, Ho & Ting 2011).

In our project, due to the size of the dataset and the scattered nature of the data, the minimum support threshold was set low (10%) to detect rare patterns, while the lift parameter was set high (5) to ensure accuracy, then there are 68 strong association rules generated from dataset, respectively. The more interesting eight rules were extracted:

**Rule 1:** If the Lending Club assigns the loan to be with risk grade A, then the number of payments on the loan is 36 month and the Interest Rate on the loan from 0 to 10.4 and the client never had a public record and the client never had 90-day or worse rating and the loan
is not defaulted.

Rule 2: The number of payments on the loan is 36 month and if the Interest Rate on the loan from 0 to 10.4 and if the client never had a public record and if the client never had 90-days or worse rating and if the loan does not default, then the Lending Club assigns the loan to be with risk grade A.

Rule 3: If the Lending Club assigned a loan to be with risk grade A, then the number of payments on the loan is 36 month and the Interest Rate on the loan from 0 to 10.4 and the client never had a public record and the client never had 90-day or worse rating.

Rule 4: If the Interest Rate on the loan is from 0 to 10.4 and if the client never had a public record and if the client never had 90-day or worse rating, then Lending Club assigns a loan to be with risk grade A and the initial listing status of the loan is fractional investment.

Rule 5: If the Lending Club assigned loan grade to A and if the loan not defaulted, then the number of payments on the loan is 36 month and the Interest Rate on loan from 0 to 10.4 and the client never had a public record and never had 90-day or worse rating.

Rule 6: If the Lending Club assigned a loan to be with risk grade A and if the loan does not default, then the number of payments on the loan is 36 month and the Interest Rate on the loan from 0 to 10.4 and the client never had a public record and never had 90-day or worse rating.

Rule 7: If the Interest Rate on loan is from 0 to 10.4 and if the client never had a public record and if the client never had 90-days or worse rating and if the loan not defaulted, then the Lending Club assigns a loan to be with risk grade A.

Rule 8: If the number of payments on the loan is 36 month and if the Lending Club assigned a loan to be with risk grade A, then the Interest Rate on the loan is from 0 to 10.4 and the client never had a public record and never had 90-day or worse rating and the loan is not defaulted.

Table 7 shows support and lift for the 8 rules. As shown in Table 7, eight rules have a lift ratios are greater than 5. It provides an insight into the prediction so as to increase the probability of the result and the condition parts. It also indicates that all items in the generated rules are positively correlated with other items.

<table>
<thead>
<tr>
<th>Rule #</th>
<th>Conf. %</th>
<th>Support</th>
<th>Lift Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>82%</td>
<td>12%</td>
<td>5.14</td>
</tr>
<tr>
<td>2</td>
<td>0.77</td>
<td>12%</td>
<td>5.14</td>
</tr>
<tr>
<td>3</td>
<td>0.86</td>
<td>13%</td>
<td>5.11</td>
</tr>
<tr>
<td>4</td>
<td>0.58</td>
<td>10%</td>
<td>5.11</td>
</tr>
<tr>
<td>5</td>
<td>0.86</td>
<td>12%</td>
<td>5.1</td>
</tr>
<tr>
<td>6</td>
<td>0.58</td>
<td>10%</td>
<td>5.08</td>
</tr>
<tr>
<td>7</td>
<td>0.76</td>
<td>13%</td>
<td>5.07</td>
</tr>
<tr>
<td>8</td>
<td>0.85</td>
<td>12%</td>
<td>5.06</td>
</tr>
</tbody>
</table>
6 Conclusion

This paper proposes a solution for predicting whether a peer-to-peer lending application at Lending Club will be paid off or defaulted. The methodology for finding the solution consisted of the following main stages: Data Exploration, which is learning the properties of the dataset, Data Preprocessing, which is preparing the data for analysis, and Classification which included a long list of experiments with various classification algorithms and various tuning parameters in those algorithms in order to find the most effective classification model. The most effective classification model was achieved using Random Forest and its accuracy is 71.75%.

This paper also proposes a solution for discovering interesting not immediately evident relations between attributes of the Lending Club loan applications. This is done using Association Rules mining algorithm Apriori. This paper presents a list of selected interesting associations that were discovered as part of this task.

7 References


Zhao, Y., & Hassan, Y. (2013). Comparison of three classification algorithms for predicting PM2.5 in Hong Kong rural area. Journal of Asian Scientific Research. 3(7), 715-728