1 Data Description

The dataset we work with comes from a bank, and our goal is to predict whether a person will experience 90 days past due delinquency or worse.

The predictors that are available are the following:

Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits.

Age of borrower.

Monthly debt payments, alimony, living costs divided by monthly gross income.

Monthly income.

Number of open loans and lines of credit.

Number of mortgage and real estate loans including home equity lines of credit.

Number of times borrower has been 30-59 days past due but no worse in the last 2 years.

Number of times borrower has been 60-89 days past due but no worse in the last 2 years.

Number of times borrower has been 90 days or more past due.

Number of dependents in family excluding themselves (spouse, children etc.).

2 First view of the data - Data preparation

In order to get a notion of the data we have in hand, we first created a summary of the data. We noticed that there were N/A's in two of the columns, NumberOfDependents and MonthlyIncome, spread in 1018 rows. Since we have 5000 samples for only 10 variables, we decided that omitting these samples would not make much of a difference in our results. Figure 1 shows the missing values in our dataset.

The next step was to split our dataset into the training and test datasets. Since we were left with 3982 samples, we arbitrarily decided that the training set will consist of 3000 randomly chosen samples, and the rest 982 samples will be used as the test data.

Hereafter, all the figures and algorithms refer to the train data. The test data have been used only for computing the accuracy of the algorithms.

Figure 2 shows the scatter plot of all pairs of the variables. However, since this kind of plot is more suitable for linear correlation and continuous variables, it does not give us much information about the data.

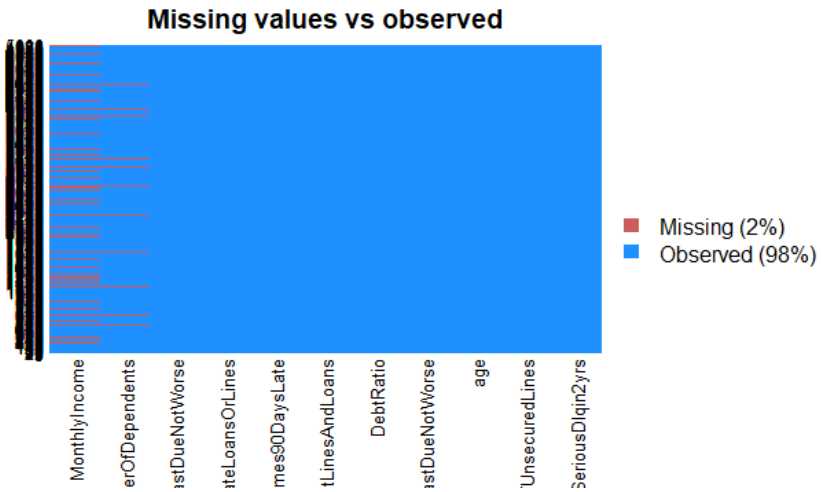

Figure 1: Missing values in our data.

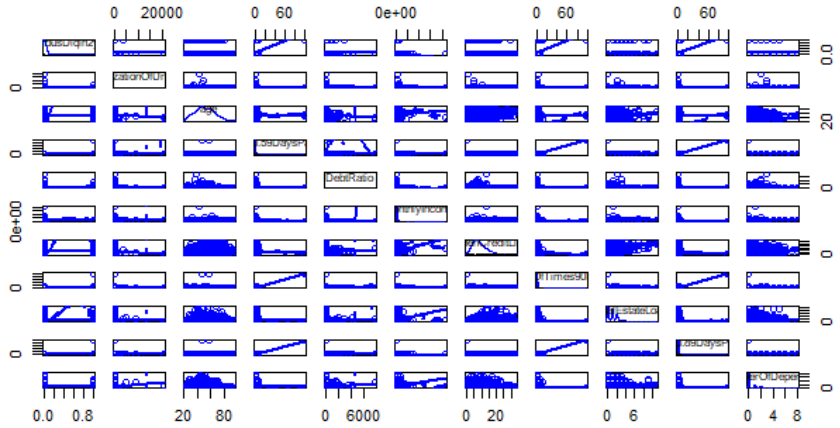

Figure 2: Scatter plot of all the variables.

3 Feature Selection

In order to help our algorithms perform better, we performed two feature selection methods. We used a stepwise algorithm which scores the variables using the Akaike Information Criterion (AIC). We tried forward, backward and both-ways selection. Forward and both-ways selection output the same set of variables, while the backward selection did not choose any of the variables. Forward selection chose six out of the ten predictors.

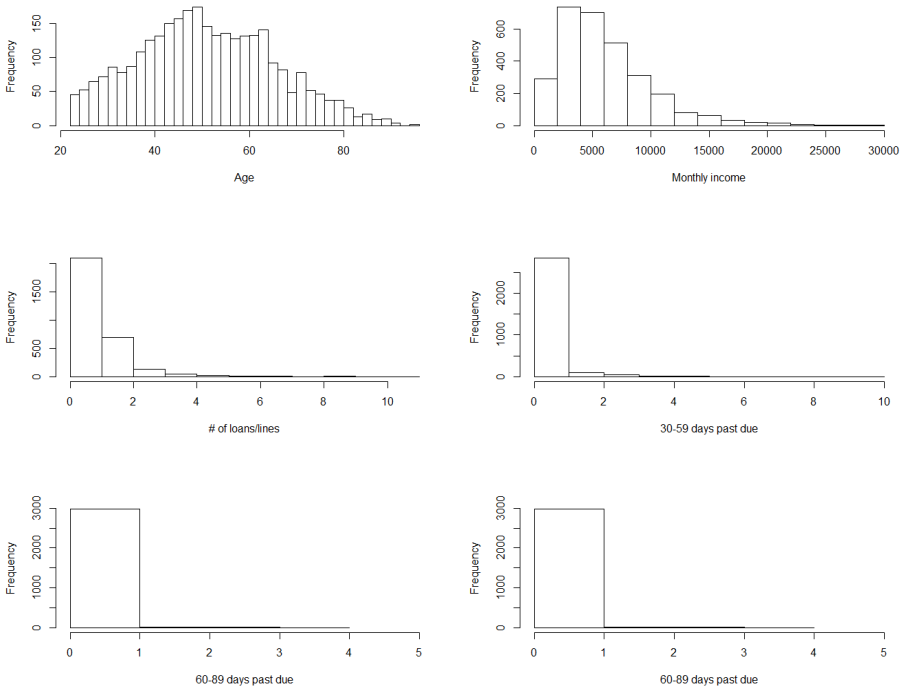

Figure 3: Histograms of frequencies for each selected variable in train data.

Figure 3 shows the distributions of all the selected variables in the training set.

Notice that the predictors include the person's monthly income, but not the debt ratio. Including only one of these variables is expected, as the debt ratio takes into account a person's monthly income.

4 Algorithms used

Since the target variable has two possible values (0/1), we have been requested to work with a classification problem. All the algorithms we used are appropriate for this type of problems. A brief presentation and discussion on each of them follows.
Also, in the end of each description, we depict the accuracy of the algorithm on the first run.

4.1 Logistic regression

Logistic regression is a method for fitting a regression curve. It is an easy to understand and interpret method. We fitted a logistic regression model given the variables we selected before.

The prediction function outputs the probability of a sample existing in the "1" class. Thus, we have to pick a threshold according to which we split the test samples in the two classes. Normally, we should perform an analysis on choosing the threshold so that we pick the one with the highest accuracy, and without over-fitting our data. However, we used 0.5 as a threshold, without performing further analysis.
Accuracy: 0.9297352

4.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is used to find a linear combination of continuous features that separates the two classes we are trying to predict. LDA computes the discriminants that will represent the axes that that maximize the separation between classes.

LDA makes some simplifying assumptions about your data: a)That our data is Gaussian, that each variable is is shaped like a bell curve when plotted.
b)That each attribute has the same variance, that values of each variable vary around the mean by the same amount on average.
Accuracy: 0.9287169

4.3 Naive Bayes

The Naive Bayes classifier is based on the Naive Bayes theorem. It assumes that all the features used to predict the target variables are independent of each other, given the target variable.

Naive Bayesian models are easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite their simplicity, Naive Bayesian classifiers often outperform more sophisticated classification methods.
Accuracy: 0.913442

4.4 Random Forest

Random Forests are one more family of algorithms used fro classification and regression problems. They operate by constructing multiple decision trees and outputting the class that is the mode of the classes, in the case of classification. They are preferred over decision trees, as they are not bound to overfit.
Accuracy: 0.9154786

4.5 Support Vector Machine

Support Vector Machines are non-probabilistic binary linear classifiers. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In our case, we used a radial kernel to predict non-linear relationships among the predictors and target variable by implicitly mapping the inputs into high-dimensional feature spaces.We tried various combinations of the "cost" and "gamma" parameters. The best model is the one with cost=1 and gamma=0.5, and this is the combination we used in further runs.
Accuracy: 0.9327902

4.6 K-nearest neighbors

KNN is a non-parametric algorithm. The input consists of the k closest training examples in the feature space. The output, in the case of classification, is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor.
Accuracy: 0.9002037

5 Bootstrapping

Bootstrapping relies on sampling from a dataset with replacement.This can be done assuming that the observations are independent and identically distributed. We use bootstrapping as a way to ensure the stability of the results we presented before.
Thus, we created 100 datasets from the original one, and trained all the models presented in the previous section.

5.1 Results

In this section we summarize the results when using bootstrapping.

Algorithm	Accuracy	AUC
Logistic Regression	0.9315141	0.73915
LDA	0.9284666	0.69811
Naive Bayes	0.907844	0.4355
Random Forest	0.9196657	0.73702
SVM	0.9317596	0.51111
KNN	0.8884008	0.54107

Table 5.1 shows the accuracy and AUC of all the algorithms described above.

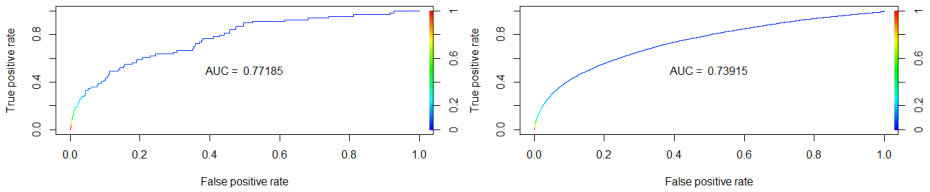

Figure 4: ROC curves for logistic regression (left: sample run, right: 100 runs).

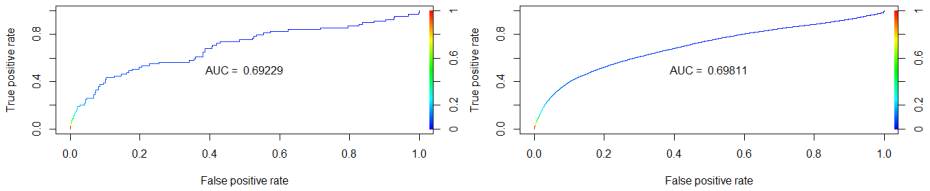

Figure 5: ROC curves for linear discriminant analysis (left: sample run, right: 100 runs).

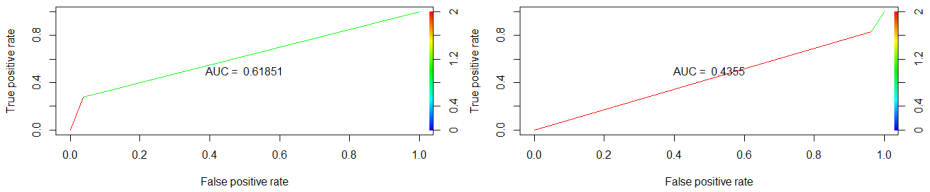

Figure 6: ROC curves for naive Bayes (left: sample run, right: 100 runs).

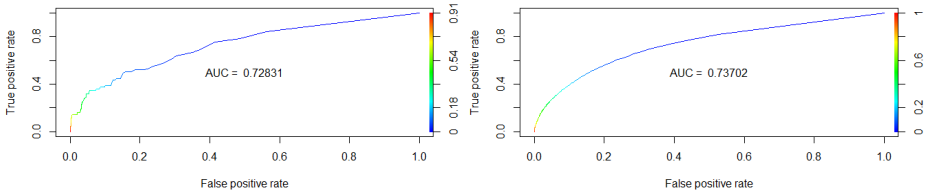

Figure 7: ROC curves for random forests (left: sample run, right: 100 runs).

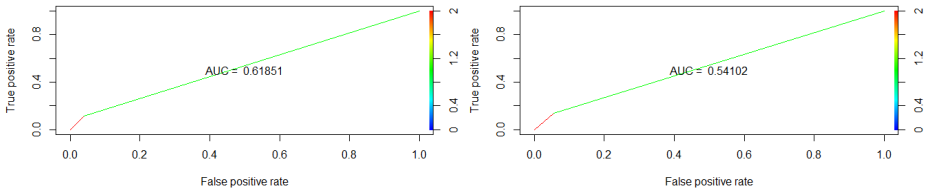

Figure 8: ROC curves for K-nearest neighbors (left: sample run, right: 100 runs).

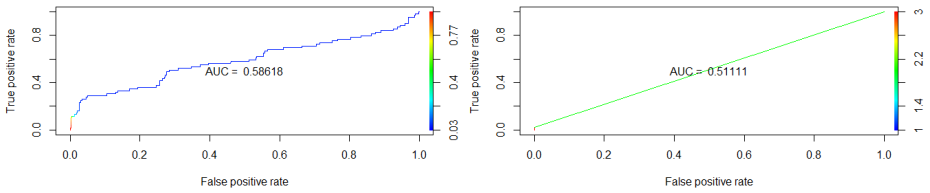

Figure 9: ROC curves for support vector machines (left: sample run, right: 100 runs).

Figures 4 to 9 also show the difference in the results between the first random run and the results of bootstrapping.

We can see that all algorithms have accuracy of about 90%. this is expected as the prior probability of 0's in our dataset is about 90%. Thus, if we constructed an algorithm that always chooses to output 0, then again the accuracy would be 90%.

The AUC is a more accurate measure. We see that, in our case, Naive Bayes, KNN and SVMs are close, some even worse, than a random choice among 0 and 1.

6 Suggestions

In order to create better models for our data, there are a couple of things we could try:

1. Transform some of the numerical variables into categorical variables (e.g. number of times past due). 2. Use imputation on the NULL values, so that we do not exclude more than 20% of our observations.