January 2020

M4: Freddie Mac single loan dataset

Using machine learning in predicting credit performance on US mortgage bonds

Group	Jess Holstein Nielsen Lars Børty Nielsen
Advisor	Daniel Hain
Length	8400w / 35p
Deadline	Friday 10th of January

AALBORG UNIVERSITET
STUDENTERRAPPORT

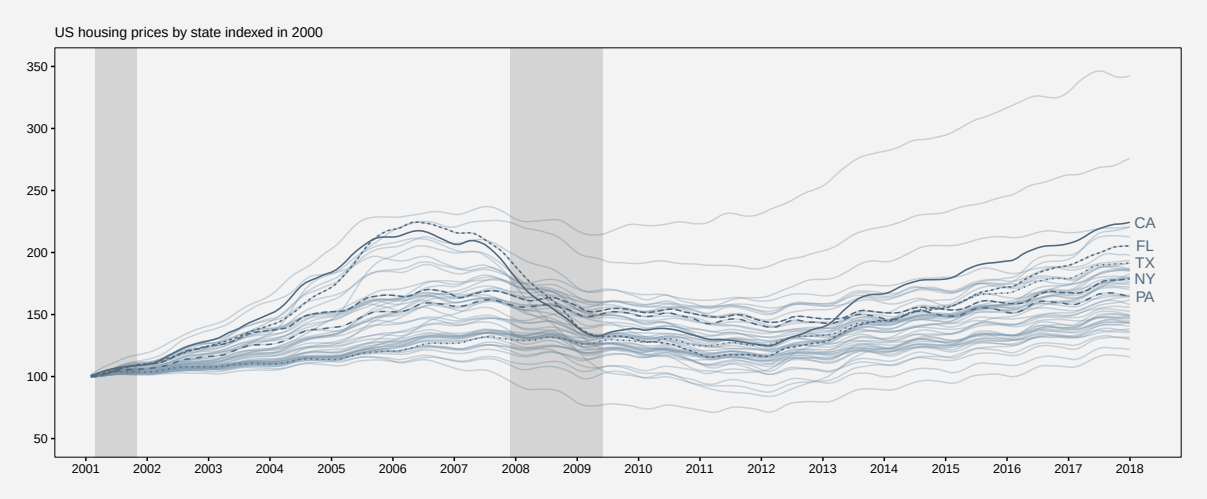
Contents

Chapter 1

Introduction

Housing is almost always central to financial crises. This is due to the fact that the house for most individuals is by far their most valuable asset and therefore fluctuations in the price of this asset influence aggregated demand. The demand for housing is driven not only by utility but also at times by speculation, and one sign that things are getting out of hand is if people stop paying installments on their mortgages, also known as delinquencies. In the run up to the financial crisis of 07-09 housing prices in the US increased dramatically as seen in figure 1.1

Figure 1.1 US housing



Note: Grey areas are official US recessions and the five most populous states are highlighted.

The easy access to credit that helped fuel the bubble ended abruptly in 2007 and left many unable to refinance expensive teaser rates and hereby going delinquent, which for many ended in foreclosure.

In finance it is critical to assess credit performance as in the chance of delinquency in individual mortgages. This is true for the credit issuing institutions, the Government Sponsored Enterprises (GSE) like Fannie Mae/Freddie Mac who securitize the mortgages, and for the private/institutional investors in Collateralized Debt Obligations (CDO's). A US family looking for a loan will go to a bank, e.g. Countrywide Financial. Countrywide then sells the loan to a

GSE, like Freddie Mac, who bundles thousands of individual loans into mortgage bonds which are resold to primarily institutional investors. In this process of securitization, it is important for all parts of the chain to be able to judge the credit risk.

The data the banks and GSE's utilize to do this assessment is inaccessible to us. But in 2014 Freddie Mac released a dataset of 22m US single family loans. This was by instruction from their regulator FHFA and done to build transparency and help investors build credit performance models. (FreddieMac, 2019). As of December 2019, this dataset contains mortgages issued between 1999 and 2018, and with monthly performance data up until March 2019.

With this data we're looking to predict delinquency status and other measures relevant for credit performance. We also want to visualize and explore aspects of this data since it is a large part of the US housing debt issued in the last two decades (around half with the rest held by Fannie Mae). Therefore uncovering the factors influencing this credit performance is fundamental to understanding the macroeconomic situation in US.

1.1 Problem Statement

This leads us to state the following problem:

Is it possible to utilize Machine Learning techniques to predict delinquencies in mortgages for US homeowners?

1.2 Methodology

To solve this problem, we initially explore the data, try to get a understanding of what variables are of interest and how we can enhance the data with additional data sources. This is a big part of the project and it's important not to underestimate the importance of getting a good feeling for the data.

The analysis will be split in two sections. First, we apply a set of ML models on a small subsample of the data to access how well they are at predicting the right outcome, but also at what cost this comes computationally (how long they take to run). This is because we're limited in time, so we want both a good model that is also fast.

Secondly, we will choose one or two of these models and compare them against a neural network and some baseline models to access how well the algorithms are in predicting the delinquency status of individuals.

The project includes a lot of plots in vector format and benefits from being read on a high-resolution screen.

All codes to replicate the analysis can be found on Github and the original 80+ GB data can be downloaded from FreddieMac. Notice that replicating the complete analysis is computational very heavy.

All codes are written in R (R Core Team, 2019) with version: 3.6.1. The primarily workhorse packages used in the analysis is:

- Datawrangeling is done with Tidyverse v. 1.3.0 (Wickham et al., 2019)
- ML models are fit with CARET 6.0-84 (Kuhn, 2019)
- Tensorflow is accessed by Keras 2.2.5 (Allaire and Chollet, 2019)

Chapter 2

Theory

In this section the theory of the applied methods is presented. Focus will especially be on the supervised algorithms and their hyperparameters. The chapter will also contain sections about the practical implementations of these algorithms and the CloudML framework used to handle large data sizes.

2.1 Supervised Learning

Supervised learning is focused on data problems where the data contains several explanatory variables *x* and an output variable *y*, also called labeled data. The goal is to create models that relates the variables to the output in order to make predictions of future observations. This topic can further be reduced to classification or regression problems depending on if the output variable is discrete or continuous. In this project the focus will be on classification, as the goal is to predict whether a person falls behind on loan payments or not, which is a binary outcome. (James et al., 2014, p.26-28)

Bias-Variance Trade-off

The expected error of a given problem can be decomposed into the three components on the right hand side of equation 2.1. The variance of $\hat{f}(x_0)$, the bias of $\hat{f}(x_0)$ and ϵ the variance of the error term. This is the *bias-variance trade-off*

$$E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\epsilon) \tag{2.1}$$

To reduce the test error the bias and the variance of $\hat{f}(x_0)$ have to be minimized. These are both positive values and for this reason the test error can never be reduced below the variance of ϵ. Flexible and more complex models tend to fit the data better, resulting in a low bias. However, they might become to flexible and prone to noise, which might result in overfitting and vastly different results if new data is feed to the model. Less flexible models will reduce the variance, but due to the simplicity will lead to higher bias. The goal in selecting the statistical learning methods, is to find models with both low bias and low variance.

Classification

The same approach can be used for classification settings, the main difference is that *y* is no longer numerical but instead a categorical variable. In 2.2 $I(y_i \neq \hat{y}_i)$ is an indicator function, it has the value 1 if the prediction is correctly classified and otherwise 0. The equation then produces a measure of incorrect classification. (James et al., 2014, p.34-37)

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i) \tag{2.2}$$

Resampling methods

The data contained in this project will randomly be divided into a train and a test set in ratio of $75\%/25\%$. The train set can further be divided into a train set and a validation set, where the models are fit using the train set and the model performance is evaluated on the validation set. A measure of loss is selected, and the best performing model is selected. This approach is very sensitive to the random selection of the data, as different divisions may yield vastly different results. To circumvent this problem k-fold cross validation is used instead. The data is divided into k sets and the model is fit using k-1 sets and the validated on the remaining set. This process is repeated for k different validation sets, afterwards the test error is calculated by averaging the loss for the cross validations. The optimal model is then selected and used to predict on the test set. In this project k=5 is used as this is not as computational demanding given the large data seize. (James et al., 2014, p.176-181)

The algorithms used in this project can be categorized into tree-based models, penalized models/ variable selection models and neural networks.

2.1.1 Tree-based Models

The simplest tree-based model is the CART (Classification And Regression Tree). The tree-based approach divides the feature space into a set of regions and fits models in each region. The splitting points are selected by the splits that leads to the largest increase in the fit. The largest increase in fit is found by selecting values of *s* and *j* that minimize 2.3, where *s* is the splitting point and *j* is the *j*-th variable. R_1 and R_2 are the Regions after a split, $\hat{y}_{R1}$ and $\hat{y}_{R2}$ are the means for each of the regions.

$$min_{j,s}[\sum_{i|x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i|x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2] \tag{2.3}$$

The process is repeated until some stopping criteria is met. The number of nodes selected for the model is important, since a small tree might be to biased and a large tree might overfit. To be able to deal with the bias-variance trade-off pruning is used. In pruning a large tree is grown, splits that do not improve the fit much is then removed. The sub tree *T* that minimizes 2.4 is selected.

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{i|x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T| \tag{2.4}$$

The tuning parameter α determines the penalty for growing a large tree. If $\alpha = 0$ the full tree is grown, higher values of α will result in a smaller tree. The optimal value of α is selected using k-fold cross validation. |T| is the number of terminal nodes of a given tree *T*. Since this project focuses on classification predictions, the classification error approach from 2.2 will be used. The observations in each region will be assigned to the most occurring class.

The CART algorithm however can be very sensitive to certain observations, this can effect what variables are used to divide the feature space. Instead more complex tree based models can be used to improve the predictive ability, such as bagging, random forest and boosting. (James et al., 2014, p.306-310)

Bagging

Bagging produces a large number of trees and take the average of theses to reduce the variance. Since only one dataset is available bootstrapping is used create B different bootstrapped training sets. Then B trees are grown, these trees are not pruned since the high variance of the large trees are reduced by averaging. By using bootstrapping random variations are generated in the dataset, then new datasets of size *n* are created from the original data. The data is random sampled from the original with replacements, the news sets have the same size as the original.

Since this project deals with a classification problem, a majority vote is used to determine the output. The output $\hat{f}^j(x)$ is predicted for all B trees and the most occurring output is then selected. (James et al., 2014, p.316-317)

$$\hat{f}_{avg} = argmax_k \sum_{j=1}^{B} I(\hat{f}^j(x) = k) \tag{2.5}$$

Random forest

Random Forrest improves on bagging by reducing the correlation between variables. If the data contains a dominant predictor, then most trees in bagging will appear similar. To solve this the random forest algorithm is used and data of size *n* is once again sampled from the original data with replacements. Then *m* random variables are selected from the total *p* variables without replacements. B trees are then grown without pruning and then averaged. (James et al., 2014, p.320)

Boosting

Boosting focuses on improving on weak learners, by giving miss-classified observations a higher weight. Instead of growing a lot of trees and averaging, the trees are grown in sequential order. The information from preciously grown trees is incorporated into the next tree. The full data set and all the variables are used in boosting. The predicted output becomes a combination of all the fitted values from the grown trees (James et al., 2014, p.321-323). The output of the boosting algorithm is given by 2.6.

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x) \tag{2.6}$$

λ defines the rate of how quickly the boosting algorithm learns. Extreme Gradient Boosting is the boosting method used in this project(XGB), it is a model with more focus on regularization to reduce overfitting.

2.1.2 Penalized Logistic Regression

A way to improve predictions, compared to simple logistic regression, is to reduce the variance by shrinking parameters. The models used for this is ridge regression, lasso and the combination of these called elastic net.

Ridge regression

The ridge regression seeks to maximize 2.7, the last term is the shrinkage penalty. If $\lambda = 0$ the coefficients are not penalized, and the model just becomes an ordinary regression. When the value of λ is increased the coefficients are shrunk towards 0. λ is selected using cross validation. Shrinkage is able to reduce problems caused by correlation between predictors.

$$\sum_{i=1}^{n}[y_i X_i \mathbb{B} - log(1 + e^{X_i \mathbb{B}})] + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{2.7}$$

Lasso

The lasso algorithm seeks to maximize 2.8 in the same way as ridge. The main difference is that lasso is able to set coefficients to 0, whereas ridge is only able to shrink them towards 0. By setting certain coefficients to 0 lasso is able to do variable selection. (James et al., 2014, p.215-225)

$$\sum_{i=1}^{n}[y_i X_i \mathbb{B} - log(1 + e^{X_i \mathbb{B}})] + \lambda \sum_{j=1}^{P} |\beta_j| \tag{2.8}$$

Elastic net

The elastic net seeks to maximize 2.9, α and λ er determined separately. The elastic net penalty is a weighted average of the penalties from the lasso and the ridge regression. The elastic net does variable selection like then lasso and shrinks the parameters of correlated variables like the ridge regression.

$$\lambda \sum_{j=1}^{P} (\alpha |\beta_j| + (1 - \alpha)\beta_j^2) \tag{2.9}$$

2.1.3 Neural Networks

Neural networks are made up of nodes, these nodes are units of computation. The networks are built layer wise, where the first layer is an input layer with a number of nodes equal to the dimensionality of the used data. This layer is then connected to the next layer in the neural network by weights. The Perceptron is the most basic network and contains only two layers, an input layer and an output layer with only one node. The input layers only receive and sends values to the next layer, they do not make any computations. A linear combination of the weights and inputs is passed on to the output node, where an activation function, often

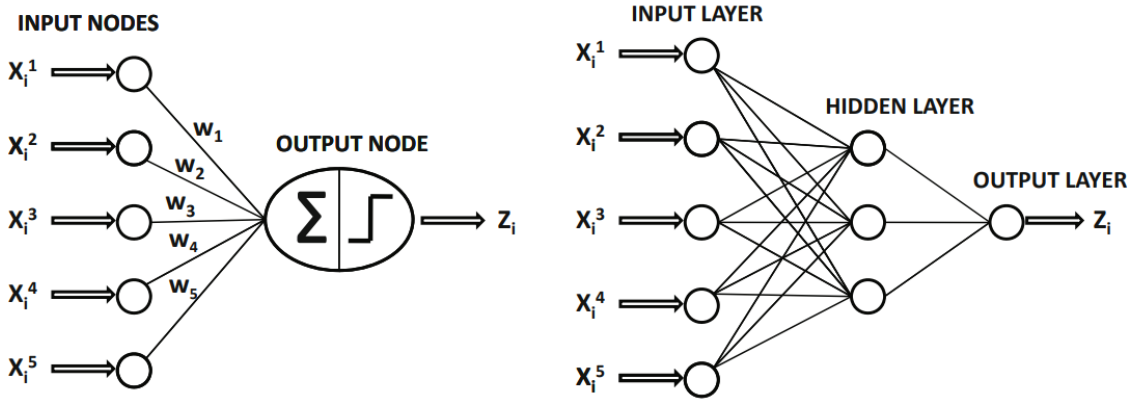
the sigmoid/logistic function, will determine the predicted output. This process is illustrated in **figur** 2.1 (a). Initial values of the weights are assigned randomly, the neural network is then trained by calculating the loss function and updating the weights, throughout the entire network, when the predictions are incorrect. These cycles through the training set are referred to as epochs, with each epoch consisting of a number of batches of data where in between the weights are updated. Multiple epochs are often required to train a neural network, but an excessively large number of epochs won't necessarily yield better predictions and might lead to over fitting. (Aggarwal, 2015, p.326-331)

The perceptron is a rather simple model since it only contains one activation function. More complex patterns might be found by using multi-layer neural network, these networks contain hidden layers in between the input and output layer. The nodes in the layers are connected to all other nodes in the next layer and each nod in the hidden layers contains an activation function. The multilayered neural network is illustrated in **figur** 2.1 (b).

The prediction of a m-layer neural network is given by 2.10. $g(\cdot)$ is the activation function, $a_i^{(m-1)}$ is an output from the previous layer and $w_{jm}^{(m)}$ is the weights connecting the two layers.

$$\hat{y} = g\left(w_{0i}^{(m)} + \sum_{j=1}^{k^{(m)}} w_{jm}^{(m)} a_i^{(m-1)}\right) \tag{2.10}$$

Figure 2.1 Neural Networks



(a) Perceptron (b) Multilayer

Illustration from: (Aggarwal, 2015, p.328)

Backpropagation is used to update the weights when prediction errors are made, it consists of a forward- and a backward phase. (Aggarwal, 2015, p.329-331)

- In the forward phase training data is feed into the neural network, computations are performed through the layers with the current weights and a prediction is made in the output layer. The prediction is then compared to the actual class.
- In the backward phase the weights are updated from the output layer all the way back through the network. The weights are updated by computing the error of each neuron

in the hidden layers. This error is found by a function of weights and errors from all the nodes in the layer in front of it. The error is then used in a gradient descent function to update the weights.

2.2 Performance Metrics

In dealing with a classification problem the standard performance measure is the confusion matrix. They can be hard to compare as they depend on the sample size so from this matrix some standard metrics are calculated.

Table 2.1 Confusion Matrix

		True class	
		FALSE	TRUE
Predicted class	FALSE	TN	FP
	TRUE	FN	TP

From the confusion matrix in table 2.1 the following can be calculated:

$$precision = \frac{TN}{TN + FP} \qquad recall = \frac{TN}{TN + FN}$$

$$spec = \frac{TP}{TP + FP} \qquad npv = \frac{TP}{TP + FN}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

From these formulas especially negative predictive value npv and specificity $spec$ are of interest. This is due to the fact the data has unbalanced classes. It is more interesting finding the ones experiencing delinquency even if it means getting some false negative. Accuracy is here not the best measure because one can get a good accuracy by just guessing on the dominating class, which isn't really useful.

To summarize, npv describes how good the predictions of the model are for the minority class, and $spec$ describes large a proportion of the minority class is predicted.

2.3 Implementations

2.3.1 CARET

In this project the CARET (Classification And REgression Training) package in R has been used to run all the algorithms except the neural network. This package is used since it streamlines the train and test process for classification and regression problems.

The most important functions used are:

- *createDataPartition* randomly splits the data into a train and test set. The ratio of train to test size is selected for this project at *p=.75*
- *trainControl* selects the resampling method, in this project cross validation is selected.
- *train* combines the above functions, by using the train set to evaluate model performance.

Each of these functions can be customized further by adding additional arguments. The metric argument is used to select the optimization criteria, for this project the metrics *Kappa* and *ROC* are selected. (Kuhn, 2019)

2.3.2 Keras with Tensorflow Back-end

Keras is used for computing neural networks in this project. It is an API with primary focus on being able to do fast experimentation, it is written i Python but it has been implemented in R as well. The Keras for R uses TensorFlow as its default backend engine.

The primary task of the user is to decide the architecture of the neural network. This is made easy by using Keras, as the package allows the user to specify the number of layers, the amounts of nodes and what activation functions to use. Additionally, Keras allows for the use of additional specifications to reduce overfitting, such as dropout layers used in this project (Allaire and Chollet, 2019).

The model structure used in this project consists of an input layer, an output layer, three hidden layers and two dropout layers. The hidden layers all contain 128 nodes.As the project works with a binary outcome, the output layer will only contain one node. A Sigmoid activation function will be used, as the probability of a observation belonging to a certain class is predicted. Sigmoid ranges from 0 to 1, predicted values above a certain threshold will be labeled as 1/TRUE or 0/FALSE if below.

Sigmoid activation

$$\phi(z) = \frac{1}{1 + e^{-z}} \tag{2.11}$$

ReLU

The activation function used in the hidden layers is Relu activation. f(z) is equal to zero when z is negative and f(z) is equal to z when equal or above 0.

$$R(z) = max(0, z) \tag{2.12}$$

2.3.3 Google CloudML

With the full data the modeling process gets computationally infeasible on single computers. Therefore, this project utilize the CloudML framework from Google. This framework allows us to send data and models to be trained in googles servers, and from there deploy the models. It is integrated with R through the *cloudml* package. When training models here we can specify exactly the specifications of the machine the model will be run on, so if we can get as many CPU's, GPU's or RAM as we please.

This is preferable to creating our own server on google cloud as it only uses precisely the resources needed for the job. The drawbacks are that we're can't train models not build with the Tensorflow back-end, and that we can only get up to 100 predictions per query, as predicting in batches to our knowledge isn't implemented in R. But 100 is enough to get an idea of precision of the predictions.

There is an implementation of other models in tensorflow through the package *tfestimators* but it seems this package is out of date and therefore if we want to run other models, like random forest or regularized regressions we have to go through python (Allaire, n.d.).

Chapter 3

Data Exploration

In the following chapter the data will be visualized. The features added to the dataset will be described, both from performance data and external data sources. The final part of the chapter will feature data exploration of all the variables together and identifications of interesting patterns.

3.1 Data Management

The data used in this project is freely available from Freddie Mac's webpage [1]. The full dataset contains approximately 22.5 million loans issued between 1999 and 2018 with performance data up until March 2019. The data is structured in two different data formats. One is loan level information, it contains details like *Loan to Value* or *Credit Score* on each individual loan, from this dataset we use all the variables. The other and much larger dataset contains information on the monthly performance of each loan. This means that in this dataset each loan can have up to 240 entries (Months between January 1999 and March 2019). This contains data on *Delinquency Status* and *Unpaid Balance*.

In the data management process additional variables are created. Some with data from external sources which will be reviewed in the feature engineering section. The rest are constructed based on the original data. Of these the most important variable is the binary delinquency variable that is the outcome variable for the prediction problem. A summary table of the variables in the constructed dataset can be found in figure A.1 in the appendix.

- **delic_binary** This variable is constructed so that it is 0 if a single loan never in it's existence has been behind on payments, and 1 if at one point in the performance dataset the loan has been delinquent. It is important to understand that this doesn't differentiate between someone who struggled one time, and someone who completely stops payments and eventually gets evicted.
- **delic_binary6** Same as above but just for 6 months behind on payments.
- **delic_date** The date of the first time a loan goes to 1 in delinquency.
- **delic_sum** The sum of total delinquency numbers. if this variable is very high it indicated that payment has stopped for a long time and is not being paid back.

[1] http://www.freddiemac.com/research/datasets/sf_loanlevel_dataset.page

- **surv_binary** If the debtor at one point experienced problems paying, but then recovers and goes back to paying.
- **surv** The number of times that the debtor has recovered from payment trouble.
- **survival** A counter that runs until maturity or event. (delic_binary)
- **recovered** A binary variable indicating whether the last observation in the data the debtor is paying the installments.
- **first_complete_stop** A binary variable of whether a debtor after first delinquency never recovers.

3.2 Feature Engineering

As the dataset includes information of the area, zipcode, metropolitan code and state, it is possible to enhance the dataset with statistics on these levels. It might be the case that there is a higher probability of delinquency in low income areas, the following features are added [2]:

- **pop** Population in each area.
- **median_income** The median income in 2018.
- **Ethnicity** The ethnic composition: White\Black\Native\...

3.3 Data Visualization

It is said that one of the most common mistakes within data science is not looking at the data. In this section some purely descriptive plots will be presented, that is there is nothing fancy other than plotting the variables. The next section will seek to take it a step further and try to extract insights. The following figures are standard summary plots with the delinquency status coded in. That is that the fraction of loans that at one time experiences problems repaying a loan is plotted with a lighter color. Numbers on the y-axis are in 1000's. All plots, unless stated, has been run on a sample of 1m mortgages.

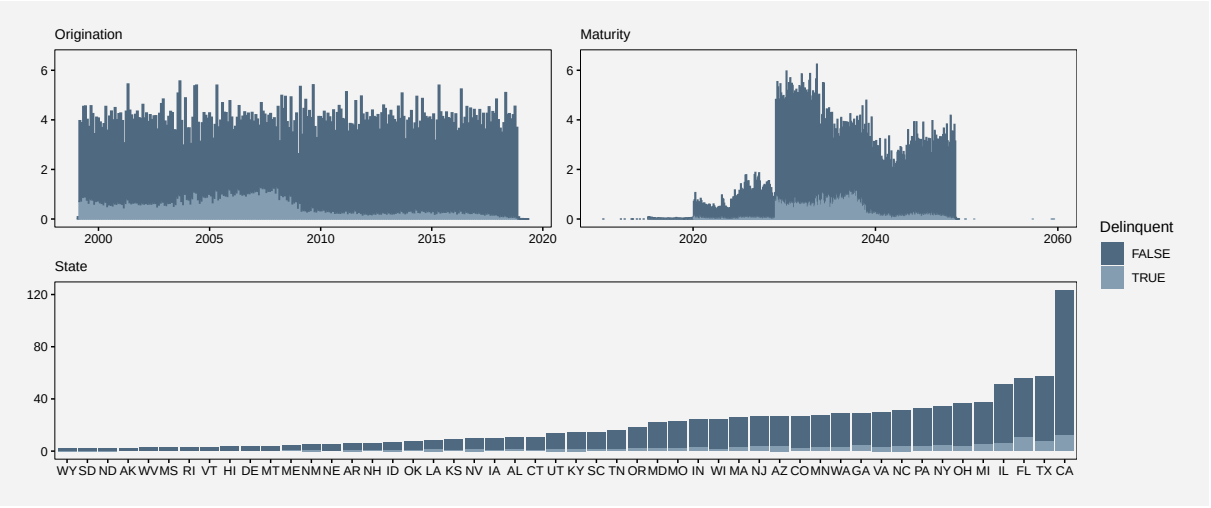
The first figure 3.1 is about origination, a lot of the issued mortgages in 2005 - 2008 experiences problems repaying, this fits well with the understanding of an overheated credit market with deteriorating credit assessment and increasing cheating in income statements.

From the figure it also seems that most loans with 15/20-year maturity don't experience problems in the same way as the standard 30-year. This should be useful for the models. A quick look at the states reveals that Florida almost have the same number of mortgages that experience delinquencies as California even though they have less than half their number of issued mortgages.

Next is figure 3.2 where more information on the mortgages is displayed. In this plot some noticeable patterns appear regarding some of the variables, like the insurance percentage. Most have 000, but other is 12, 25 and 30. To understand this requires knowledge of the American real-estate system, but it is probably due to some regulation or standards in the industry. Another thing to notice on these plots is the number of borrowers, there are more loans with 2 borrowers, but still more mortgages with one borrower experience problems, this makes sense as one debtor

[2] This additional data is found on https://www.census.gov/data/tables.html

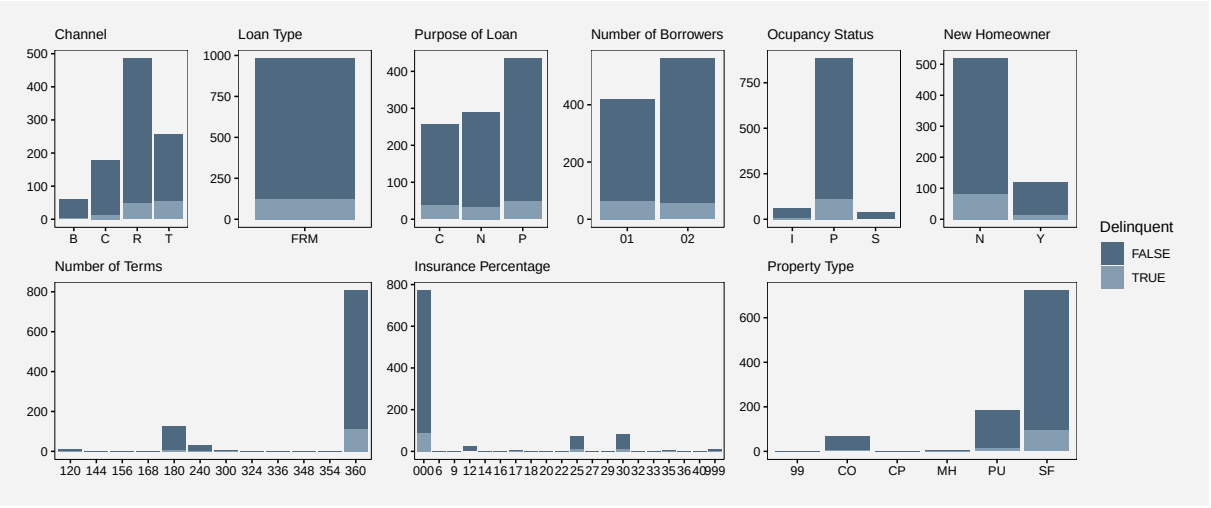
Figure 3.1 Dates and location



Note: Some of the states are over represented due to a higher population. This will be adjusted in the data wrangling section to see if there is any states that are over represented in the dataset.

is more exposed to losing a job which could lead to getting behind on payments. Information on the specific variables can be found in (FreddieMac, 2019).

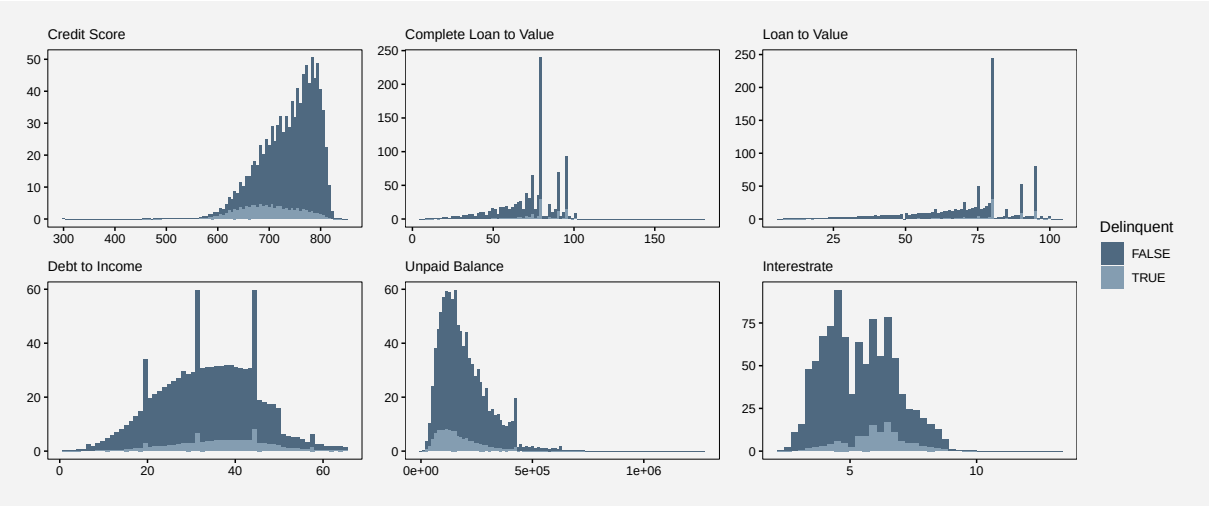
Figure 3.2 Info on the mortgages



Note: The figures have been sorted for NA and obs < 50.

In plot 3.3 continuous variables are plotted as histograms, not to be confused with the discrete bar-plots above. Credit score as intuitively expected appears to be important in predicting delinquencies, since a large proportion of borrowers with low credit score at one-point falls behind on payments. Also, interest rate seems to be important, although there might be some correlation as individuals with a low credit score might be offered worse rates. Again some patterns appear in the data, in debt to income there are spikes around 20, 30 and 45. This could be a result of people taking loans up to a limit set by their income, or it could be a sign that they stretch things to the limit as to meet some threshold. But it looks suspicious as both income and the exact amount people need to borrow would be expected to follow a more smooth distribution.

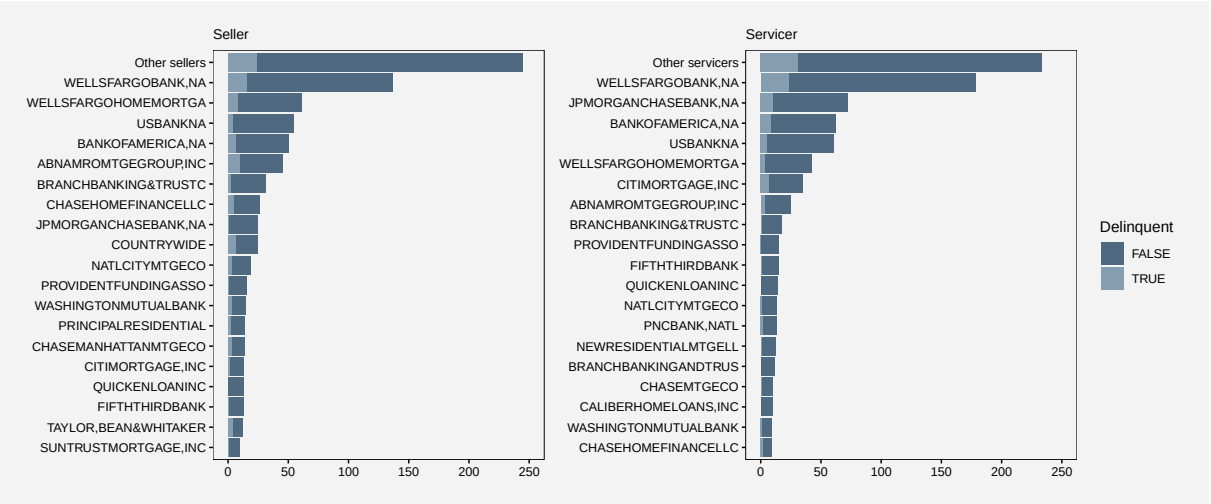
Figure 3.3 Histograms of variables



Note: Bin size is 100, except for debt and interest-rate that is 60

Another interesting part in the data is that it contains information regarding seller of the loans and who are servicing them, this information is plotted in figure 3.4. There are some infamous names in there connected with the meltdown of the subprime real estate market. Countrywide, WaMu and TBW, these will be examined more in the next section.

Figure 3.4 Top 20 Sellers and servicers



Note: The name of the seller and servicer don't always exactly match for the same company. This might be due to legal separation between subsidiaries.

3.4 Data Wrangling

Tabular data contains two dimensions, columns and rows. But often it is of interest to analyze a third dimension, groupings[3]. In the previous section the options in illustrating these two dimensions were more or less exhausted but adding the third dimensions allows for analysis of summary statistics within different groupings. With this third dimension the possibilities expand

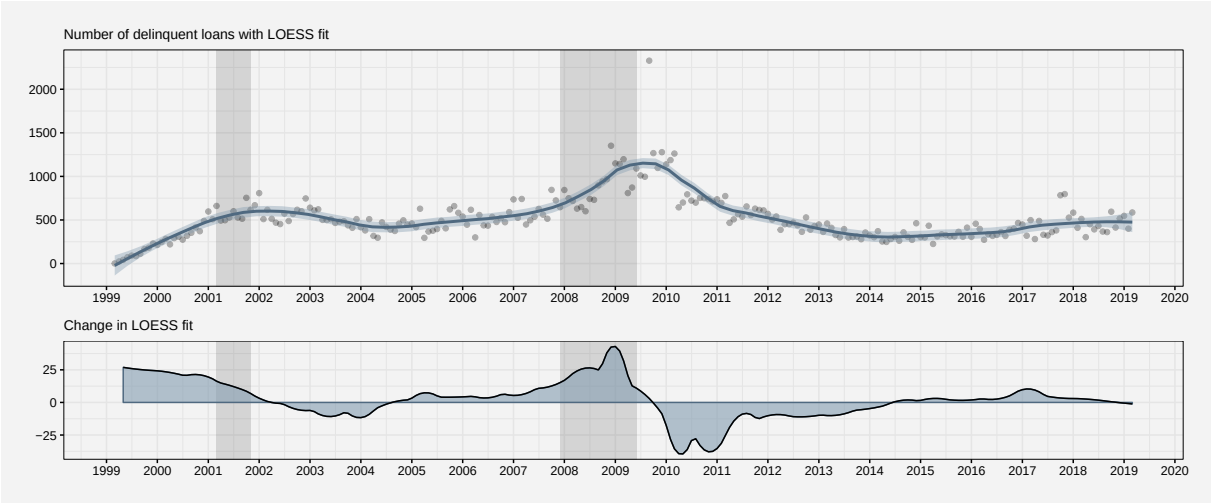
[3]Technically groupings were used in the first section when the rate of delinquency within each variable was illustrated

greatly, therefore the following section contains a subjective selection of interesting plots, which is just a tiny fraction of possible relationships to illustrate.

3.4.1 Basic Analysis

Firstly, one of the most interesting features of the data is examined. How the number of delinquencies evolve over time. This is done by the *delic_date* variable by counting the number each month. On this a LOESS model with a span width of 0.15 is fitted. In plot 3.5 a big increase appears around 2006-08, which tops around the end of the recession. The LOESS function clearly shows that the number of people experiencing problems starts rising around 2005, that is still well before prices topped compared with figure 1.1.

Figure 3.5 Monthly first delinquency



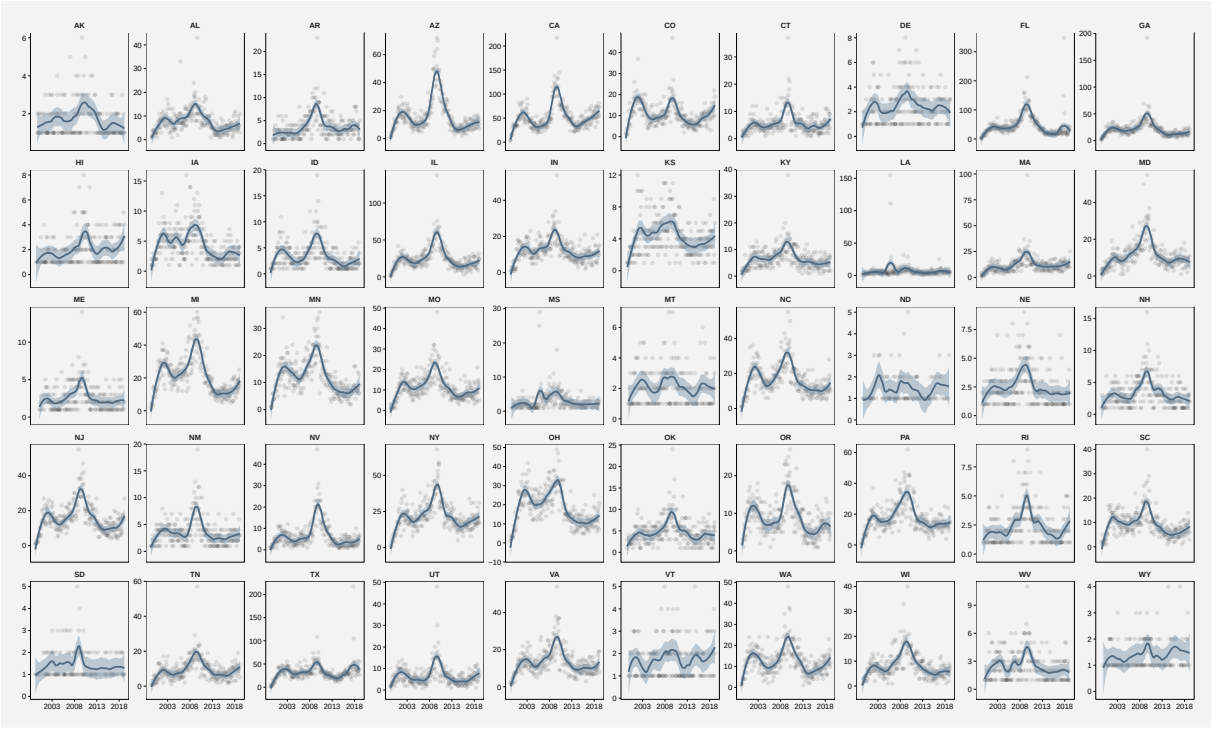
Note: There is no adjustment for the fact that there are fewer active loans in the beginning of the period compared to the end, and therefore by the end of the series there is a larger pool of loans that can experience delinquency.

Compared to the house price indexes from figure 1.1 the fit seems lagged but is easy a question of the direction of causality. Does falling prices cause borrowers to get behind on payments or is it borrowers getting behind on payments that makes prices fall. Or maybe there is an outside factor, that influences both, like rising unemployment. Another approach is to compare how the curves evolves in the different stats in the US. This is one way to compare different regional situations and if the crisis started one place in specific. The different states are plotted in figure 3.6.

One challenge aside from there being a lot of states is that some have few observations, and maybe too few to draw any conclusions. In 3.7 the five biggest states by population have been isolated. The fits are also plotted against each other so it's easy to compare. Pennsylvania seems to have started experiencing problems first. It should be noted that Texas and California are starting to rise a bit again towards the end of 2018.

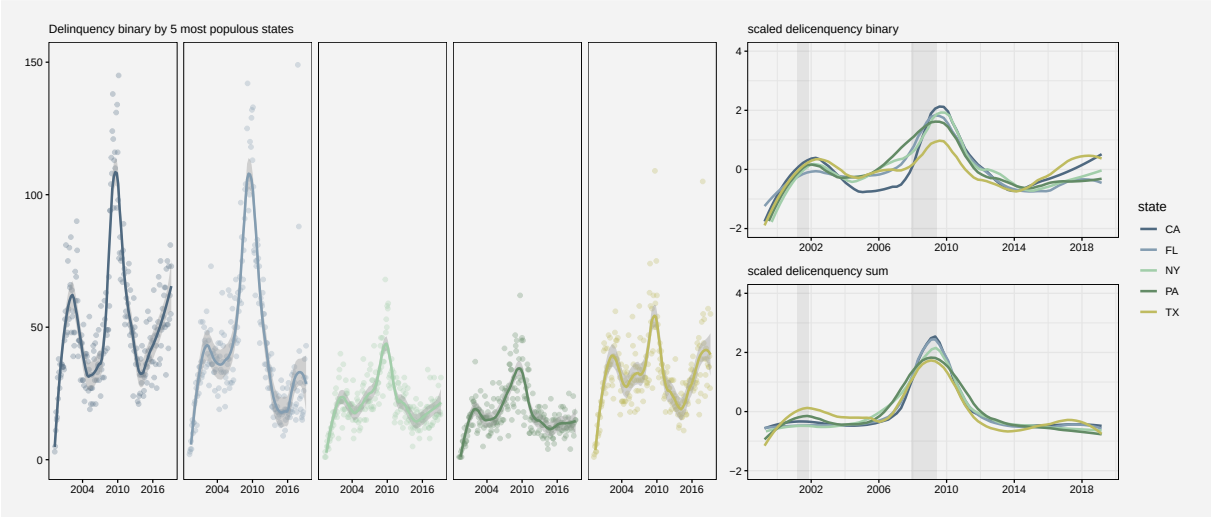
In the last section we saw that there might be differences between sellers and states. Below we've plotted the rate of delinquency within each of those. We see that there are big differences. This might be that the banks target different groups. We see that Louisiana, Florida and Alabama takes top 3, these are states prone to hurricane damage, which we will see in the spatial analysis later. But aside from them we still see some differences. This might be due to some housing

Figure 3.6 Monthly first delinquency by state



Note: The effect that higher credit score gives lower rates, might not be significant. The plot is done on a sample of the data (50.000 obs.) to avoid excessively large file size.

Figure 3.7 Top 5 states with scaled fits

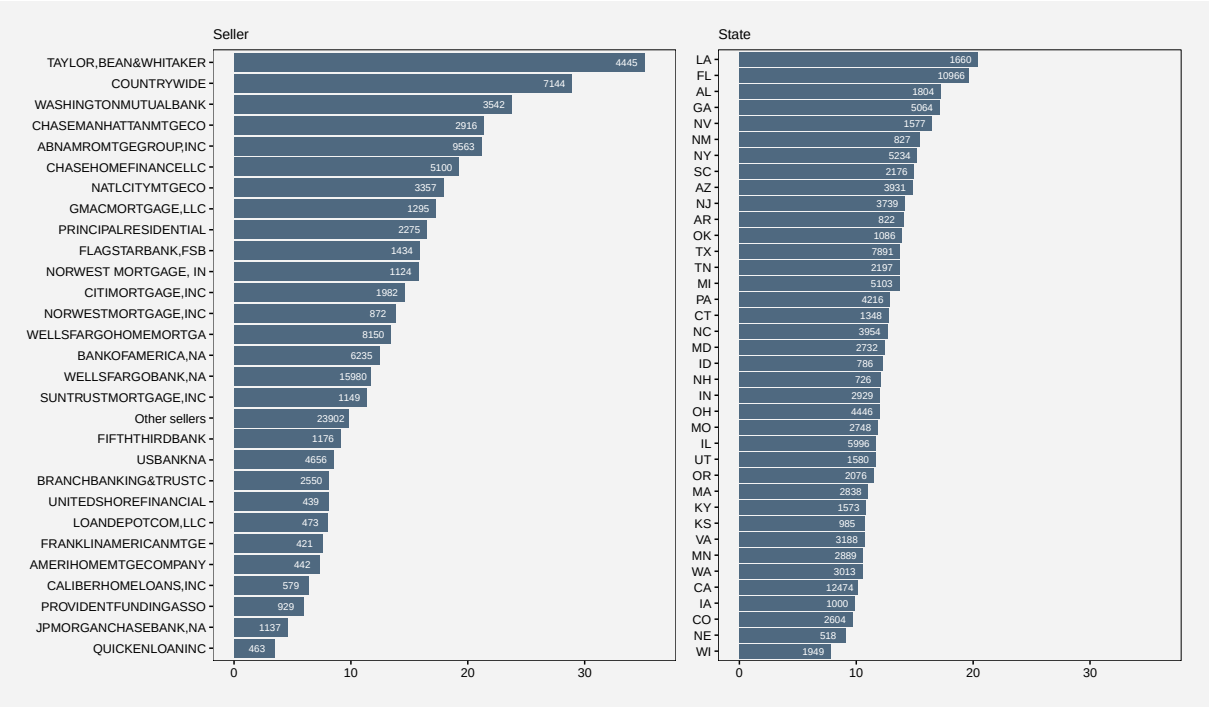


The sum delinquency tries to adjust for how much trouble is in each point by summing all the future problems. Here the crisis stands out even clearer. This might be because the problems of early 00's disappeared quickly with rising housing prices.

markets being more heated than others. We see New York in the top but California towards the bottom even though they also seems to experienced a very heated housing market as seen in figure 1.1.

In the previous section interest rate and credit score seemed to be of some significance to determine if people fall behind on payments. In 3.9 a simple linear model of the relationship is plotted, some differences are visible within the two groups. It is especially clear when comparing

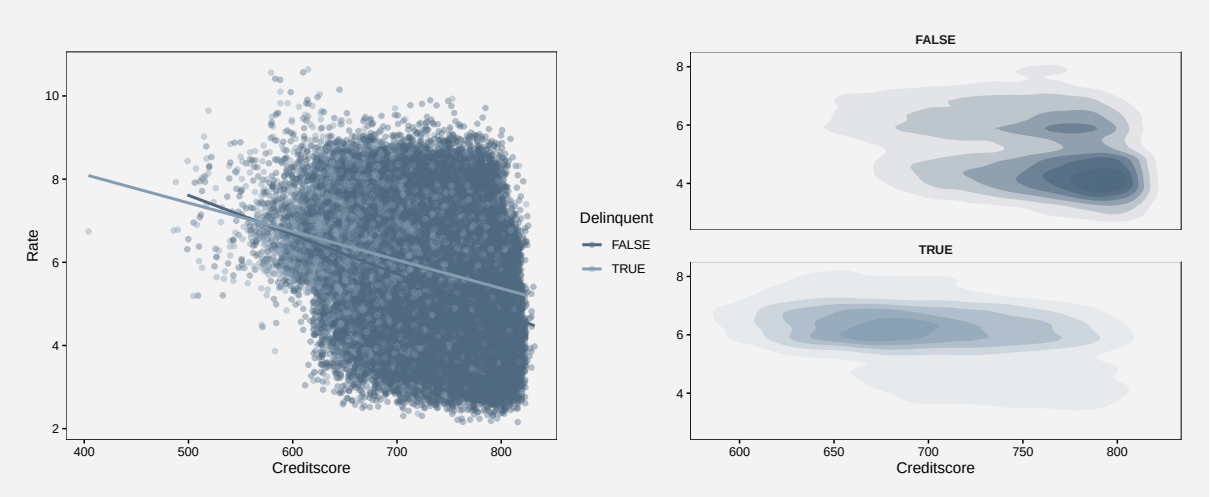
Figure 3.8 Sellers and states ranked by rate of loans experiencing delinquency



Note: The top 3 mortgage brokers with the highest delinquency rates, (TB&W, Countrywide and WaMu), were all wiped out in the financial crisis. This gives some bias as they haven't sold loans in the following years where delinquency rates were lower, this is not a problem with states.

the densities of borrowers, those who don't experience problems center around 800/4 and problematic loans center around 675/6. This makes sense given that the creditors give a higher rate to people with a bad credit rate, and these borrowers more often experience problems.

Figure 3.9 Relationship between credit score and rate

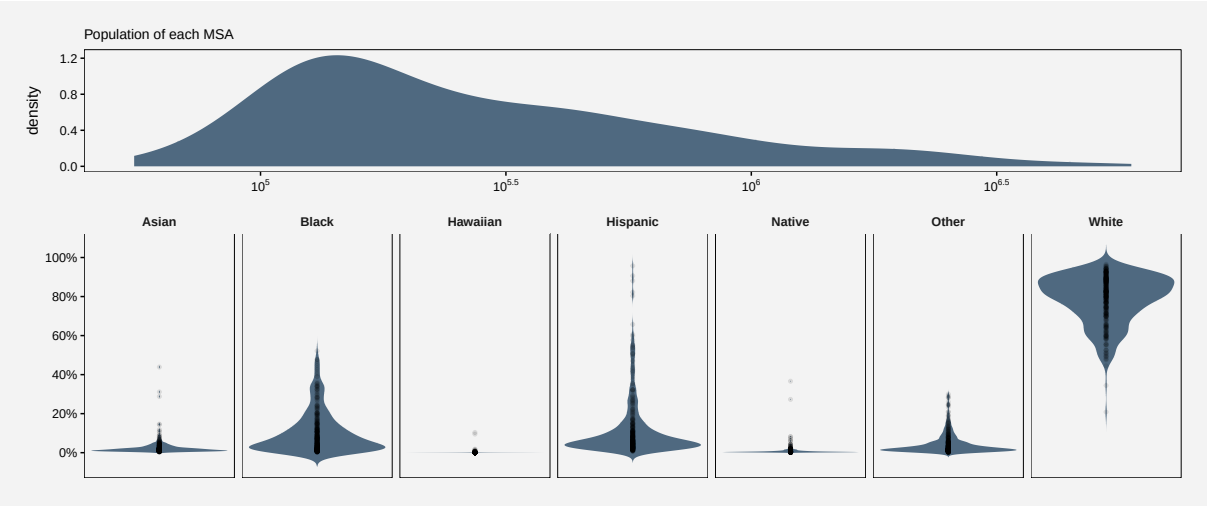


Note: there is a small effect that higher credit score gives lower rates, might not be significant. Also, the plot is a subset of 50.000 obs. this is to avoid excessive file size.

Next the added features are explored. The features are joined by the MSA codes(Metropolitan Statistical Areas). In figure 3.10 the distributions of the variables are visualized. Most of the areas have populations of around 100k - 300k inhabitants. The ethnic composition is mostly

white, but with large minorities of Blacks and Hispanics.

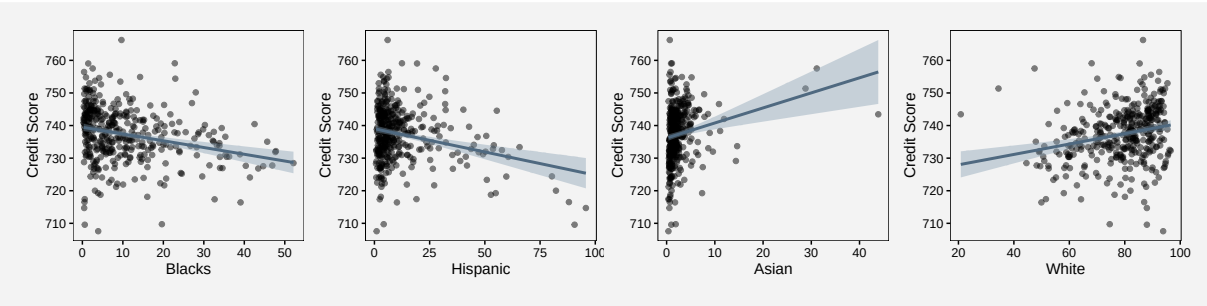
Figure 3.10 Density plots of demographics in MSA



Note: from looking at the Hispanic variable we can see that some of the US territories must be part of the groupings.

To give an idea if the new variables are of any use to the classification, they are regressed on the credit score. The result can be seen in figure 3.11. It appears that areas with large minority populations have lower credit scores. The positive fit regarding Asian minorities seems to be mostly influenced by a few extreme observations.

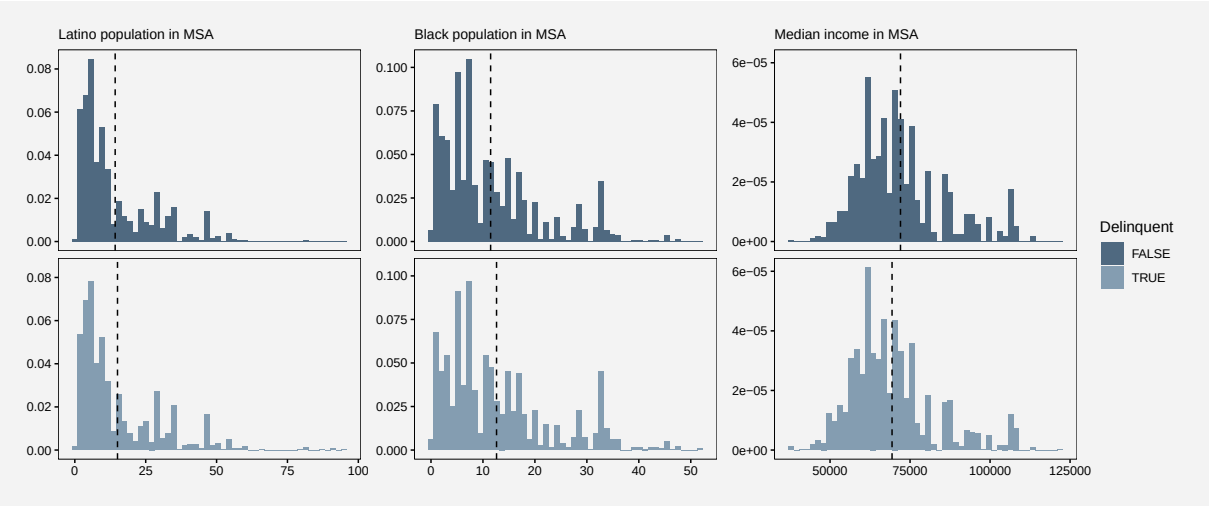
Figure 3.11 Density plots of demographics in MSA segmented by delic status



Note: The fact that Puerto Rico is in the data will probably give some bias in the Hispanic plot

In figure 3.12 the density of some of the demographics are plotted, as well as the means of both delinquency TRUE and FALSE. From the lower part of the figure it appears that delinquent loans have a higher mean of Latin/Blacks and a lower median income. The variables seem to contain at least some information related to the classification.

Figure 3.12 Density plots of demographics in MSA segmented by delinquency status

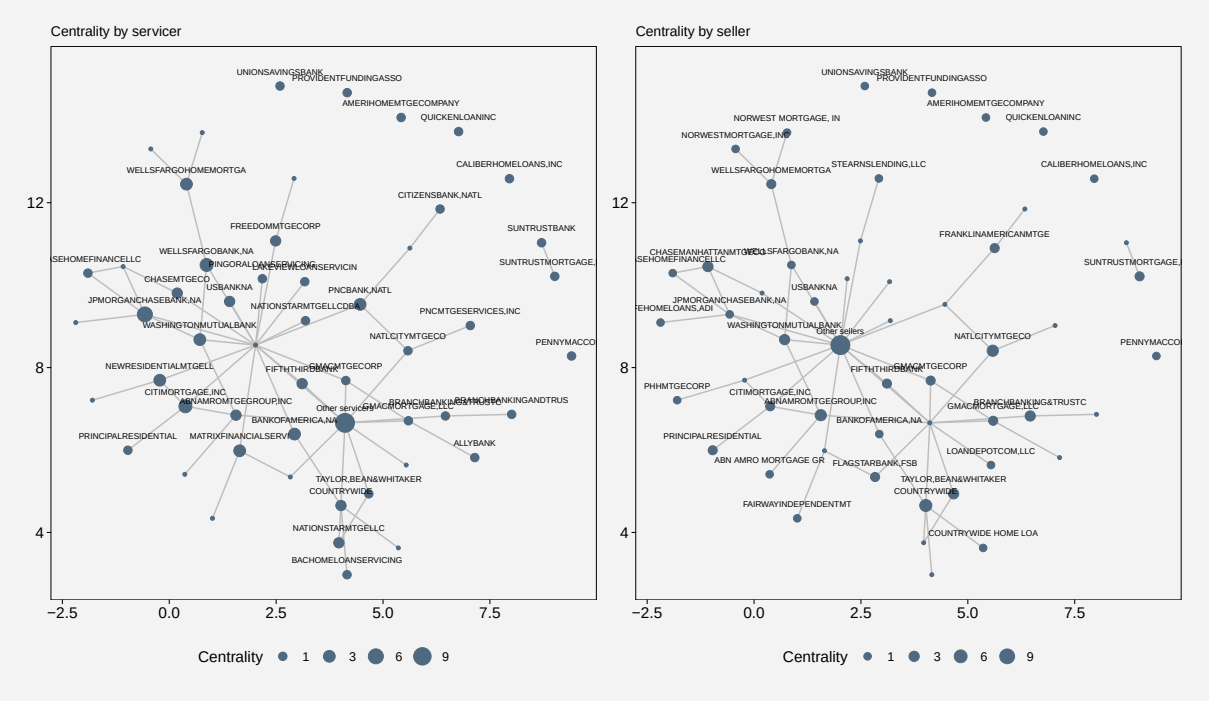


Note: The top row displays the density plots for delic_binary = 0 and the bottom row displays delic_binary=1.

3.4.2 Network Analysis

As shown before each loan has a stated servicer and seller. This can be used to see what banks work together in the process of issuing loans. We start by finding all unique combinations of buyers and sellers. And then calculate a centrality measure both on servicers and sellers. This can be seen in figure 3.13

Figure 3.13 Sellers and servicers network by direction

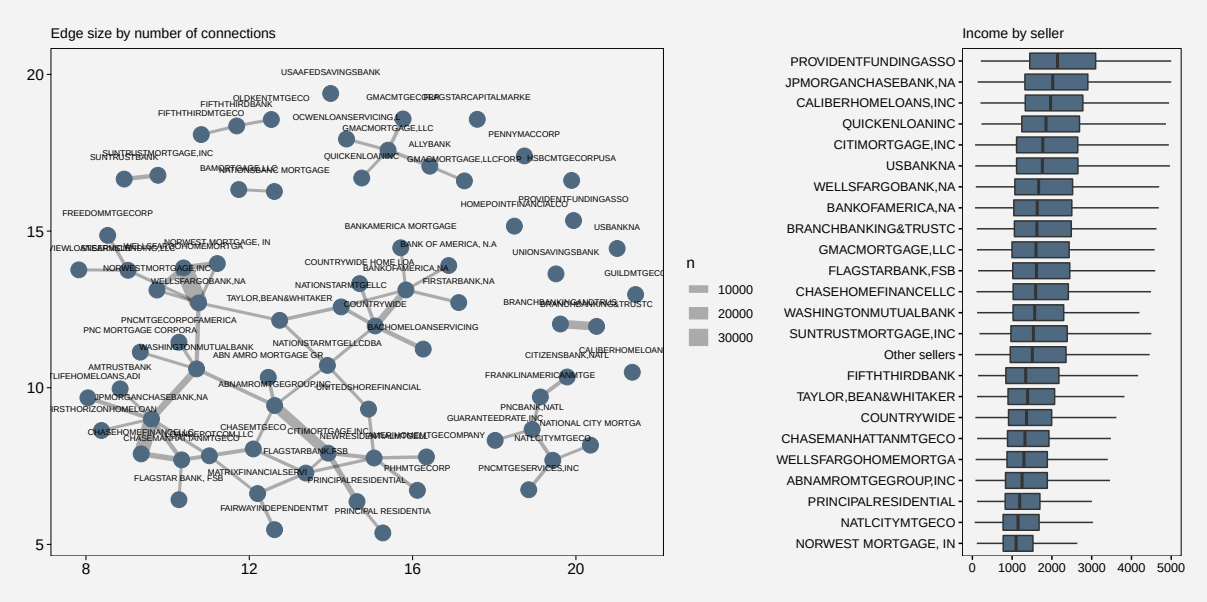


Note: Only id's with more than 1 connection is labeled, and it counts as two if it uses itself, and only connections with more than 1500 cases are shown.

From this it is noted that Other servicers and Other sellers are most central. However, some of the big financial institutions are central as well, like WaMU, Countrywide and Wells Fargo.

This plot doesn't take into consideration the number of connections. Therefore in plot 3.14 the connections are counted and the edge size is scaled with this number.

Figure 3.14 Network by number of connections



Note: We filter for connections below 500 and also remove other sellers and servicers.

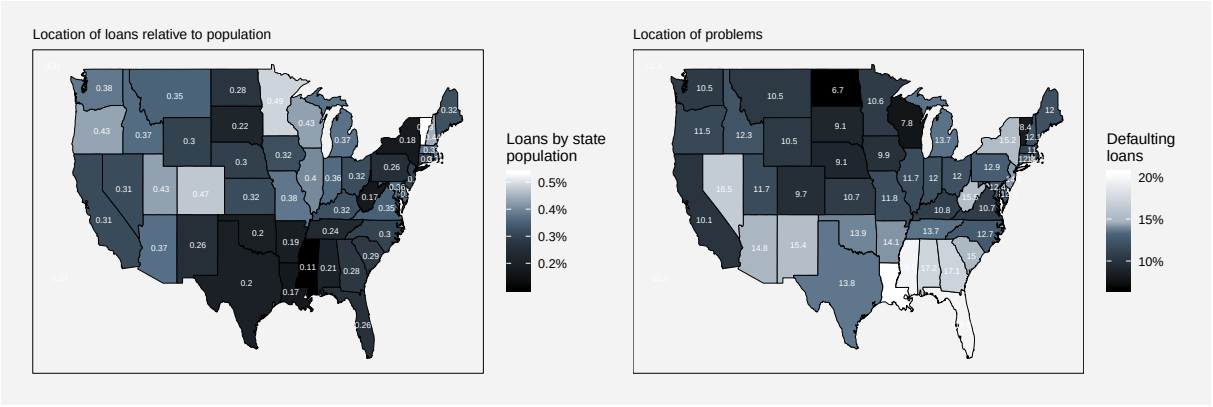
Boxplots of income of debtors are added to the plot as this could explain some of the clusters. Some banks may primarily service the upper middle class and work with other similar banks. However, no clear pattern appears. It can be seen that some banks clusters together in their connections.

3.4.3 Spatial analysis

Given the available data on the location of the loans a spatial analysis is possible. The data contains several location variables, but the MSA is not optimal since it does not cover the entirety of the US geographical area. Zip codes could be used, but a zip code map is not available in R, the only option left is the state variable.

The initial data exploration in figure 3.1 showed that the majority of loans is very centered around CA, TX, FL and IL, but by adjusting for population in the left half of figure 3.15 the data becomes more homogeneous. There is a higher concentration of loan per capita in the West and Midwest. The second half of the plot depicts the proportion of the loans in each state that experiences delinquency. A large concentration of high ratios is situated in the south, however it is not apparent why this is the case and will be examined shortly.

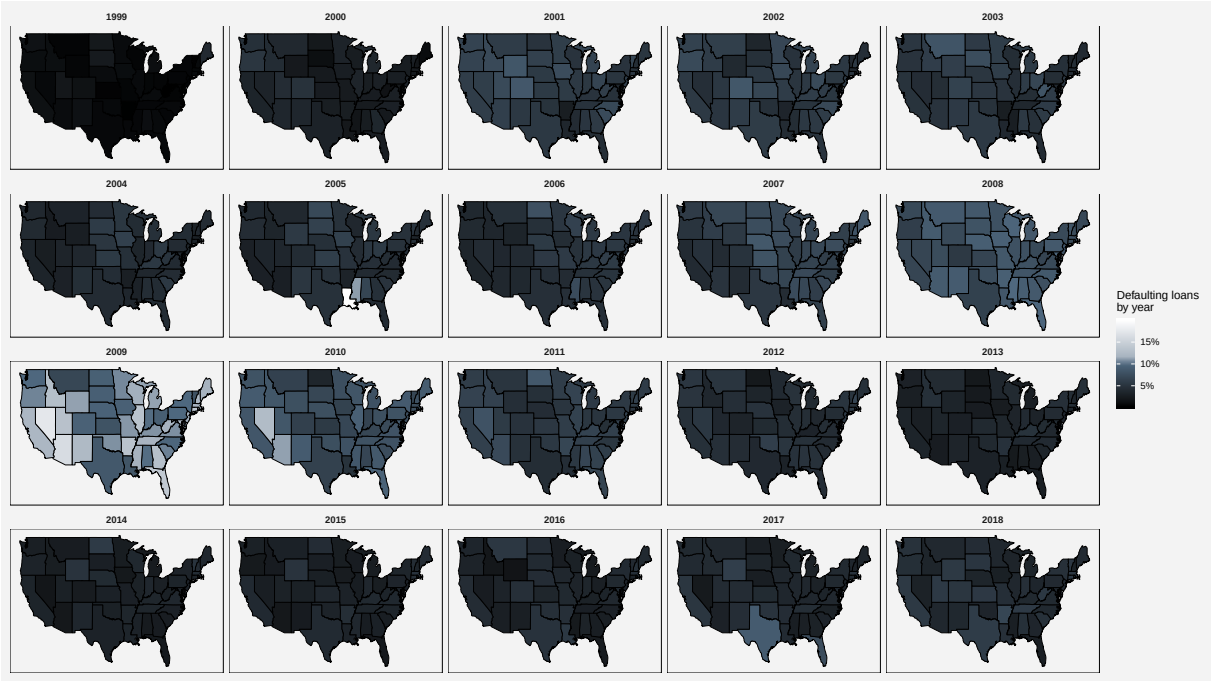
Figure 3.15 US spatial analysis



Note: Alaska, Hawaii and Puerto Rico are missing from the map

The delinquency variable will now be visualized over time. This was initially done in figure 3.5, the visualization over time will give an insight into how the states are located relative to each other in figure 3.16. The first thing that stands out is 2009 and how delinquencies topped that year, especially in Nevada where it hit extremely hard. Secondly in 2005 there are a few southern states that light up, this corresponds to the hurricane Katrina that hit August 2005. Similarly, a spike appears in 2017 in Texas which fits with the hurricane Harvey.

Figure 3.16 US defaulting status over time

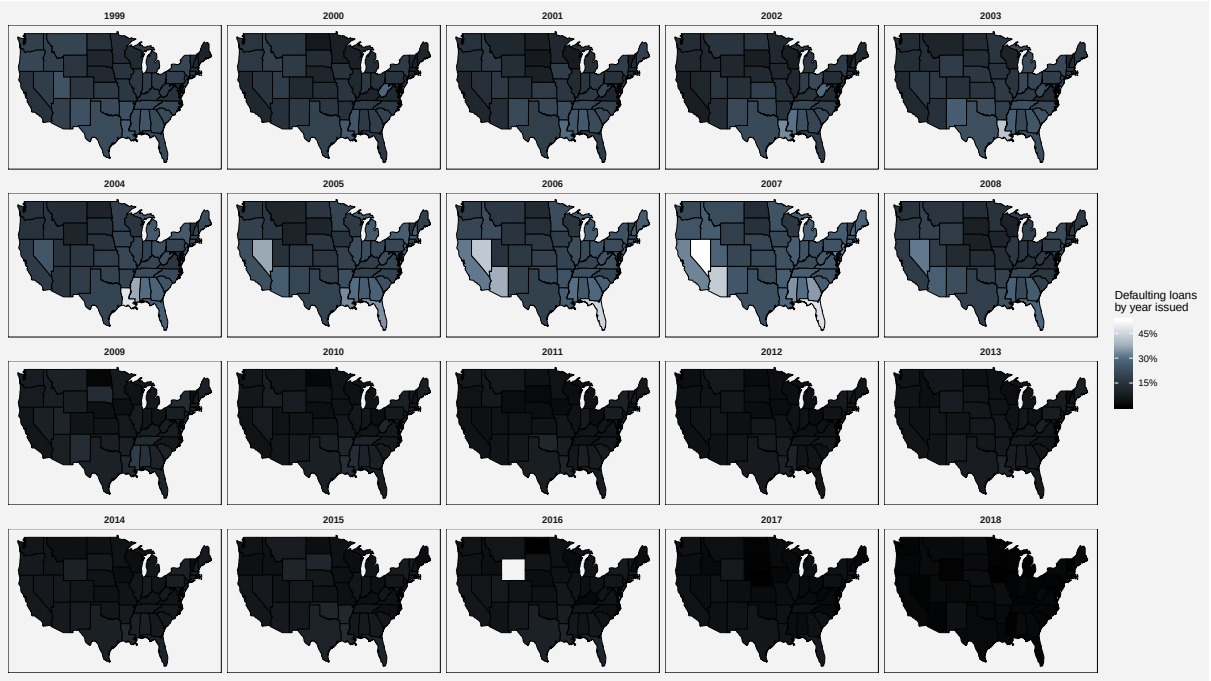


Note: The percentage shown is in relation to each state, this means that for each state all the years sum to 100.

Next up it is explored not when the delinquencies happen, but when the loans that experience problems originated. A simple hypothesis could be that in a very heated market standards were lowered, so when things turned especially bad these loans would experience delinquencies. This is somewhat confirmed in figure 3.17, it is noted how 2006 and 2007, right before the crisis were the years where most loans given turned bad. Nevada is especially bad where around half of the

loans issued in 2007 at one point went delinquent.

Figure 3.17 US defaulting status by loan origination over time

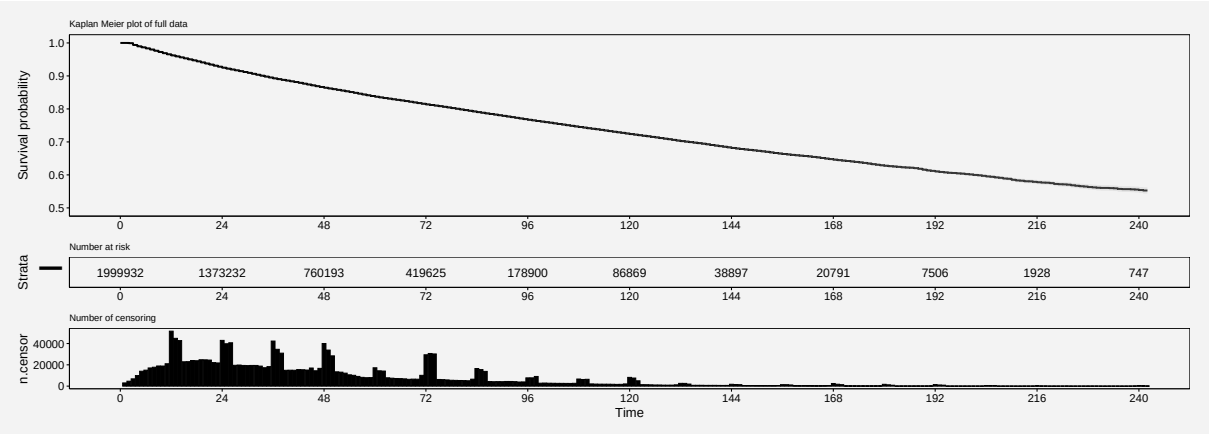


Note: As Wyoming only has few observations there was a problem of 100% in 2016, so it has been censored.

3.4.4 Survival Analysis

The data has a time dimension and lots of observations are censored, as the loans have not matured yet. Therefore survival analysis can be applied to help explore the data. As described a survival variable has been constructed that is defined by time to censoring or event. With this data a Kaplan Meier curve is constructed in figure 3.18.

Figure 3.18 KM for all data



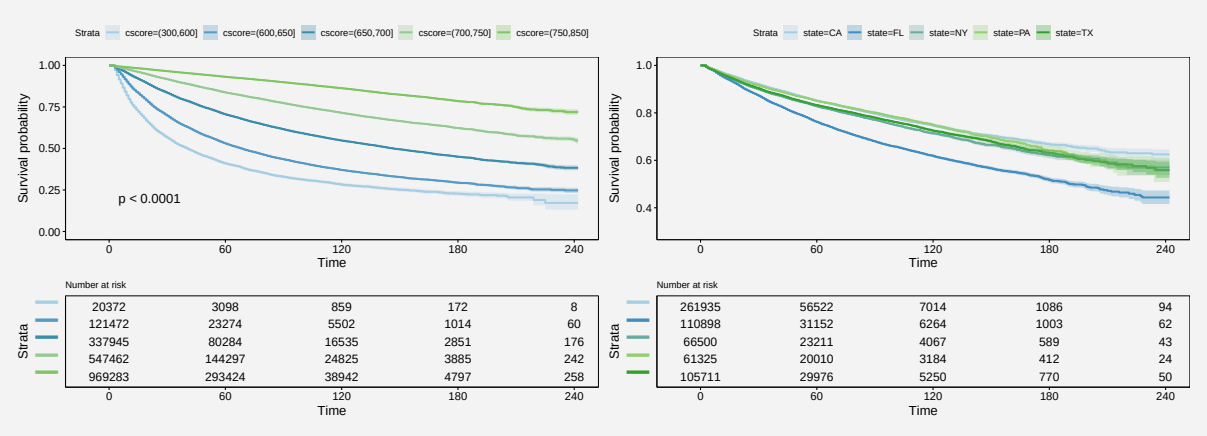
Note: The line is an estimate that's taking account for censored observations, but the 95% confidence interval is so small it's hard to see.

From this plot we see that by 10 years, still around 80% of the loans haven't experienced any problems, and by that time there are still 80.000 loans at risk. There seems to be a pattern in when the mortgages are censored. The pattern is just 12,24,36,... and a few months after. This

might be due to some regulation or refinancing that can only happen at a set time after being issued.

In figure 3.19 two plot shows the survival of different credit scores and states. As expected people with a low credit score have a higher hazard rate. The states looks similar but with Florida being the exception. This might be due to hurricane challenges, or particularly heated housing market.

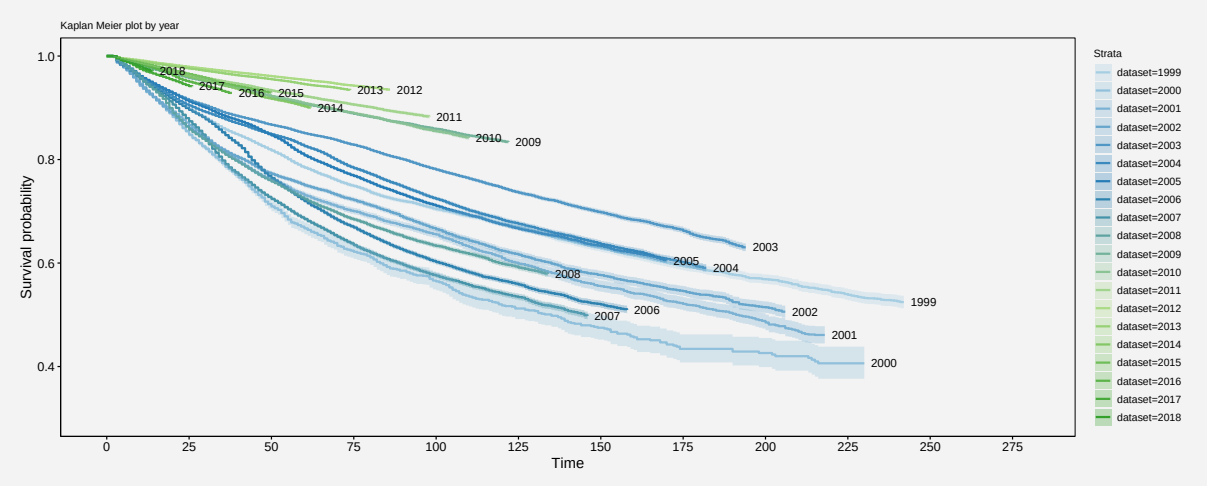
Figure 3.19 KM by credit score and state



Note: censored credit scores were removed

Finally in figure 3.20 The survival is divided by year of issuance. A clear pattern appears, that loans issued after the crisis in 08/09 are doing considerably better than ones before. Especially surprising might be the year 2000, this is due to being just before the dotcom crisis and following short recession.

Figure 3.20 KM by year included



Note: More recent year has shorter max survival as data runs to March 2019.

Chapter 4

Predictive Modeling

In this section several models and their performance are evaluated. Initially six different models along with two simple reference models are computed on a subset of data. The results are evaluated and from there two models are selected to run on a larger sample of data. These results will be compared to a neural network

4.1 Features

The models used in this chapter only contains a select number of the predictors/features mentioned in the preceding chapters. The MSA and postal codes are excluded due to computational restraints, though these are of great interest. These variables are treated as factors and would be converted into dummies in the dataframe. This would result in approximately 1300 additional columns, and when dealing with up to 22m rows this is not possible due to memory restraints. Instead some of the characteristics of the MSA's are incorporated into the dataset by the feature engineering, such as demographics and general income data. Below all the used variables are presented.

- *Delic binary*

- *The credit score*
- *First time homeowners*
- *State*
- *Channel*
- *Purpose of the loan*
- *Debt to income*
- *Unpaid balance*
- *Property type*
- *Seller*
- *Interest rate*
- *Units*
- *Occupation status*
- *Loan to value*
- *Number of borrowers*
- *% white population*
- *Median income in 2018*

4.2 Predictive models

The models mentioned in chapter 2 are initially run on 100k observations to determine what models have the best performance, and at what time cost [1]. The results are graphed in 4.1. The primary focus is on identifying borrowers who fall behind on their payments, for this reason, as mentioned earlier, the npv and $spec$ values are of special interest. The npv can be viewed as how qualified the predictions of $delic\_binary = TRUE$ are and the *spec* measures the percentage of the overall amount of $delic\_binary = TRUE$ the model is able to capture. It is important to notice that the best model depends on the problem at hand. One model might be better given that a financial company wants to avert as many borrowers who fall behind as possible. Another may be more interested in a more precise model and for this reason only will turn down customers that are classified by a pickier model.

The computational run time of the models must be taken into consideration and can be seen in table 4.1, there is quite a large time difference between the different algorithms. The time varies a great deal with hyperparameter tuning, metric was selected to yield reasonable results without being to computational expensive. As an example, in the elastic net a grid search is done of 3 alpha and 3 lambda values. Random forest is especially demanding, but no superior performance is observed.

Table 4.1 Runtime in minutes with a subset of 100k observations

Algorithm	Logistic	CART	Elastic net	Random forest	XGB	Bagging	Total
Run time	1.74	2.26	2.7	27.25	17.34	4.88	56.16

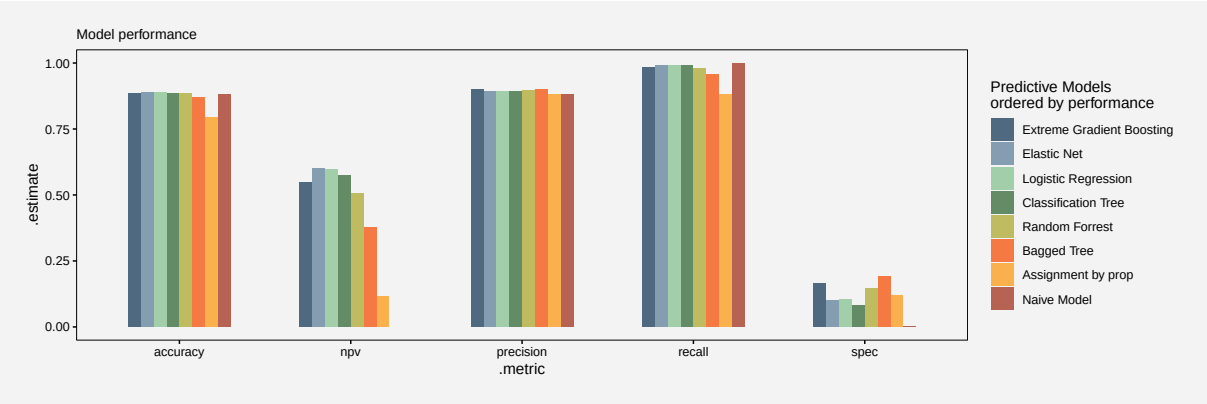
When comparing the npv values three different models are of certain interest: CART, elastic net and the classic logistic regression. The XGB algorithm is almost able to achieve similar results compared to these three models. However, when the $spec$ of the models are compared XGB is able to capture a larger proportion of the class. In general tree-based models appear to make more predictions, though bagging and random Forrest make more wrong predictions compared to the three above mentioned models. All the models have quite similar accuracy, which is a bit above the naive model. The XGB algorithm might appear to be one of the best if not the overall best performing on this subsample, however it is more computationally intensive compared to the elastic net with a runtime difference of almost 15 minutes. The assignment by probability and the naive model are included, this is only for comparison since it would make little sense for lenders to randomly select mortgages. The naive model only guesses the FALSE and for this reason gets a recall score of 100%, its accuracy is however a bit lower than most of the other algorithms. The random assignment is able get a high $spec$, but it uses many guesses to do that and therefore has the lowest accuracy. It may be concluded that the ML algorithms are picking up patterns in the data as they are able to outperform the reference methods.

Table 4.2 Confusion matrices 100k sample - Kappa

Logistic		CART		Elastic net		Random For		XGB		Bagging	
21860	2624	21891	2691	21873	2634	21644	2498	21668	2444	21138	2371
209	307	178	240	196	297	425	433	401	487	931	560

[1] 100k observations was selected due to the disproportion between the binary outcome. Fewer observations often lead to higher variation in the results.

Figure 4.1 Kappa metric: 100.000 observations



Note: The colour for each model change between the plots in this chapter, since they are ranked according to the sum of the metrics

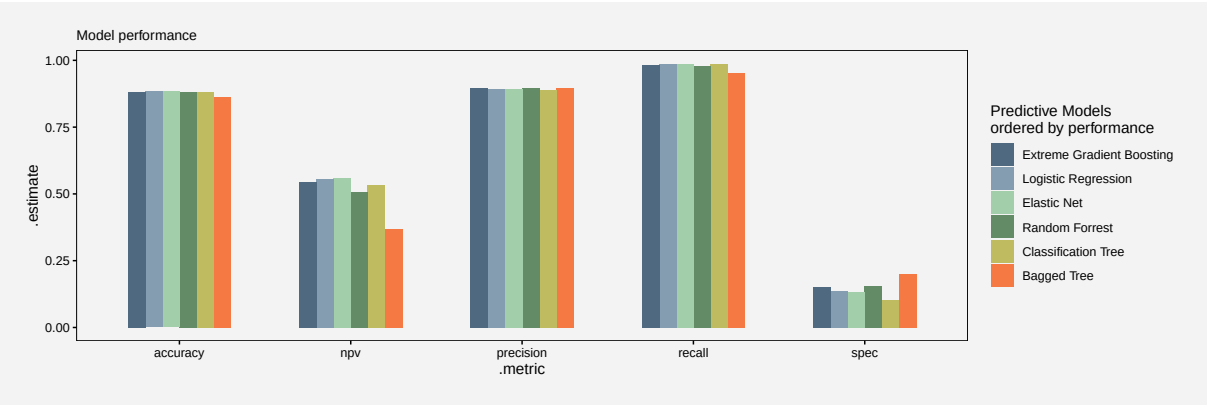
ROC - Receiver Operating characteristic

In the above section the used metric was Cohens Kappa, in this section the metric ROC will be used instead. This is done to hopefully be able to correctly predict a larger amount of delinquencies. From 4.2 one noticeable change appears, the spec have increased for elastic net and logistic regression. The ability of the algorithms to predict correctly however remains relatively similar to predictions made with the Kappa metric.

Table 4.3 Confusion matrices 100k sample - ROC

Logistic		CART		Elastic net		Random For		XGB		Bagging	
21677	2593	21735	2694	21690	2604	21553	2537	21620	2542	20982	2399
324	405	266	304	311	394	448	461	381	456	1019	599

Figure 4.2 ROC metric: 100.000 observations



Note: The reference models are not included

Elastic net and XGB is selected for the next phase of the analysis. It must be noted that the models have been tested on several randomly selected subsamples of 100k observations, some models' performance tend to vary but the overall conclusions remain the same. Elastic net almost always yields the highest *npv* value but with rather varying amounts. The elastic net is able to shrink the correlated predictors or remove irrelevant predictors, if all the variables are of importance there will be no penalty and it will become identical to the logistic regression.

Therefore, elastic net is selected instead of logistic regression. The XGB algorithm on the contrary have a very stable performance and seems to have very little variation. This may be caused by elastic net modeling the noise to well and the additional regularization in XGB to prevent overfitting.

Additional data

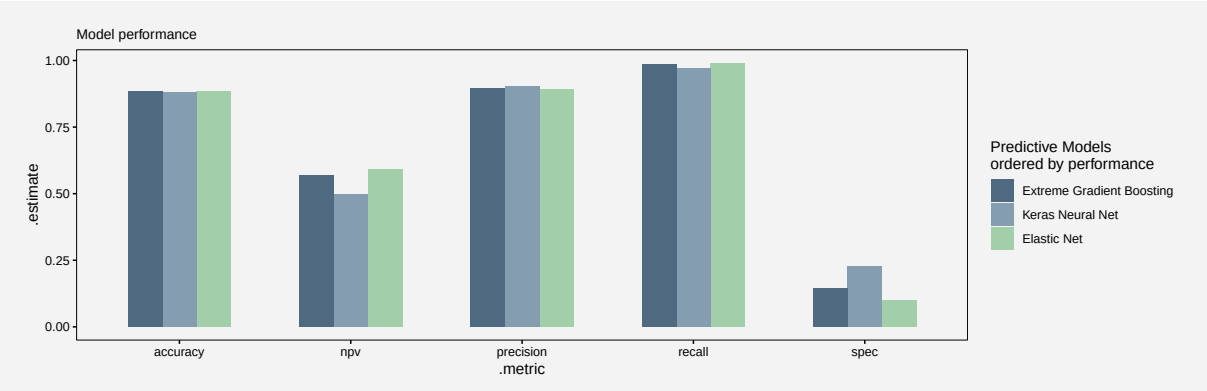
The selected models will now be computed on a larger dataset to determine if this leads to an increase in performance. The ROC metric will be used. The results are presented in figure 4.3, the results follow the same general patterns as the above results. Elastic net still has a higher *npv* score but looking at the *spec* the XGB algorithm is still able to achieve a higher overall amount of true predictions. Additional data does not seem to have any impact on the conclusion from the smaller subsamples.

IN this section a Keras neural network is included for comparison, the networks achieves the highest *spec* value at the cost of the lowest *npv* score. However, it should be noted that the *accuracy* metric is used, since *ROC* and *Kappa* are not available as default in Keras. The result of the neural network is not to surprising, since neural networks in general rarely beats or outperforms the more classic machine learning algorithms such as trees when used on tabular data. The complex nature of the neural network is often better suited for tasks such as computer vision or language processing. However, this does not mean that the neural network is without advantages in the context of this project. It is faster compared to the other used algorithms and it is easier to handle big datasets. A lot of external platforms are available for neural networks, such as Google CloudML. In figure 4.4 the delinquencies are displayed for varying values of the probability threshold. This plot depicts the trade-off between a precise model and a model capable of capturing larger proportions of the delinquencies. Additional model tuning might only have resulted in minor gains in performance.

Table 4.4 Confusion matrices 450k observations - ROC

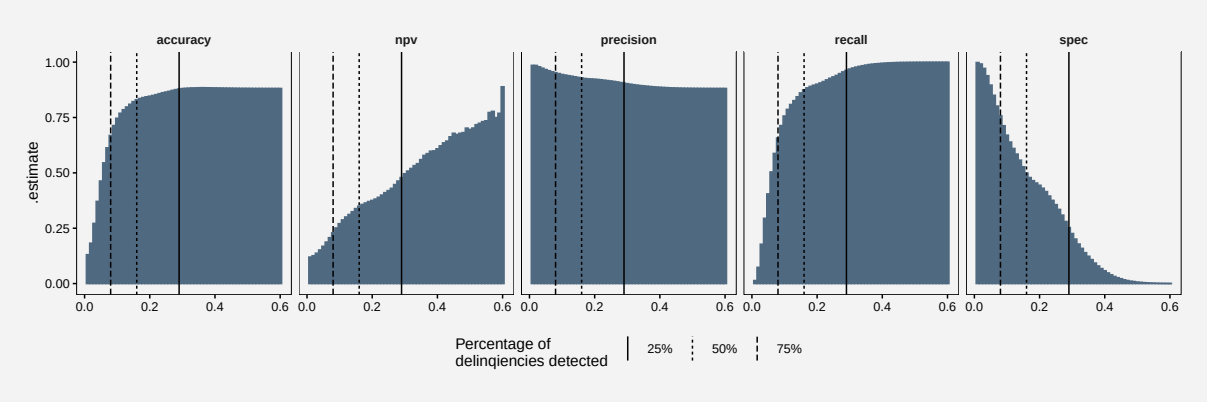
XGB		Elastic Net		Neural Net	
102460	12006	103030	12624	96081	10329
1524	2002	954	1384	3068	3021

Figure 4.3 ROC metric: 450.000 observations



Note: The neural net is set to a cutoff of 0.5

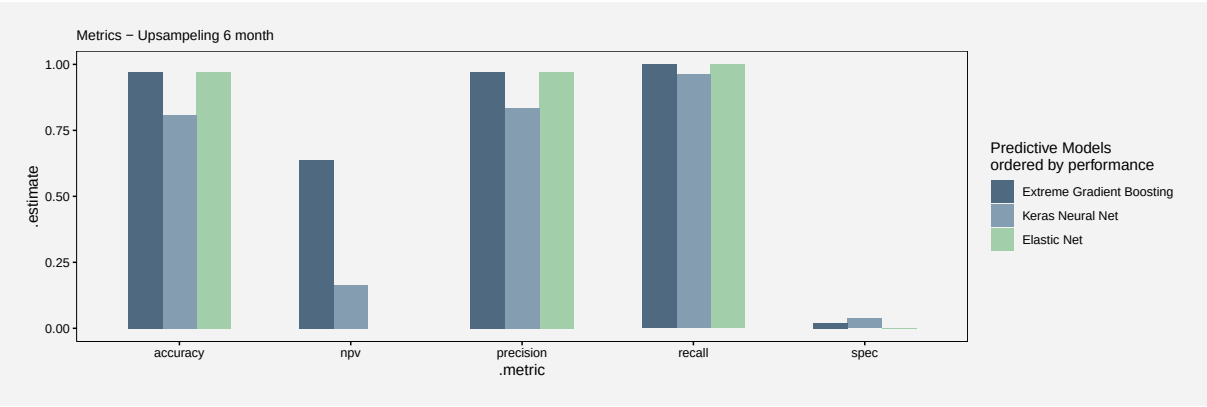
Figure 4.4 Adjusting cutoff rate on Keras Neural Net



New dependent variable

The introduced models only made a limited amount of *delic binary*=TRUE predictions, this may be caused by the fact that this project does not differentiate between a borrower who only falls behind on payment once and those who fall behind on several payments. The reason borrowers miss payments once or twice might be caused by idiosyncratic factors, compared to borrowers who does so many times. In this section the dependent variable will be changed to be borrowers who are 6 months of payments behind. The results are presented in 4.5. It is easy to observe that the results are quite unsatisfactory. XGB is able to predict the delinquencies rather precise, but the models almost exclusively predict the majority class.

Figure 4.5 Delinquencies 6 months



The data used in these models is even more disproportional than the data used for the previous sections. This causes many models to just predict the majority class, to solve this problem techniques such as down sampling can be used. This process was tested on the data, but it primarily resulted in low accuracy since the amount of delinquencies predictions was quite disproportional compared to the true amount of observations. Table 4.5 contains the predictions of the elastic net algorithm on a test set, using a down-sampled training set.

Table 4.5 Down-sampling 500k

Elastic Net	
16029	237
8260	474

It appears that the data and data management does not yield enough information to properly identify the "bad" borrowers. The relative success of the predictions of only one month of delinquencies might be associated to not being uncommon for individuals with low creditscore to get a little behind, but the ones that really fail isn't as easy to distinguish from the rest. The primary focus has been on predictions. Since the algorithms applied are considered to be "black boxes" it is hard to establish causality and determine what is the main contributors to the limited prediction abilities. One could go for more classical models and do inference but that is complicated by the large amount of categorical variables.

4.3 Considerations

Regarding the results in this analysis, a few things should be taken into consideration of how accuracy could have been improved.

- Additional information on the individuals, gender, age education, etc. held by Freddie Mac and the banks could very likely have increased the predictive power of the algorithms.
- The algorithms used in this project have only been tuned a limited amount, additional tuning of certain parameters or more iterations might have led to improvements in performance. This was omitted due to time constraints.
- The incorporated information regarding demographics and income was only selected for one year and assumed that the ratios had remained constant through the years. This might be a faulty assumption since some areas might become popular(unpopular) for certain ethnicities or become richer(poorer), especially over a span of two decades.

In the end it's important to understand it might not be possible to predict a large portion of the delinquencies, since they might be caused by a multitude of other unmeasurable reasons, there might be a big irreducible error the in prediction.

One of the important components of the models is the credit score. This score is a aggregation of a wide range of variables (FICO, 2019), from payment history to what type of credit cards the person holds. In the survival analysis it was shown that the credit score is very well at characterizing the hazard function of risk of delinquency. So, the lender can know beforehand the risk and set the rate accordingly. But the other plot, figure 3.20 showed that this might not be a good description of reality as one would have expected the years to look somewhat similar, but it appears that the loans given in the pre-crisis years are in general performing worse.

Another important consideration is regarding whether credit assessment is a prediction or forecasting problem. The models used in the project did not include any time variables. This is relevant as the data stretches almost two decades and runs through two recessions, and therefore data on the origination of the loans would most likely improve predictions. Even though the time aspect weren't included directly, it might influence indirectly like through banks going bankrupt and therefore not being a class through the whole dataset. If a time variable had been included it would create a forecasting problem that would complicate model evaluation considerably. With the recent history in mind it makes sense to consider the position in the business cycle when lending money on a 30-year horizon.

Chapter 5

Ethical Considerations

The rapid development and implementation of artificial intelligence and the ever-increasing amount of available data has enabled the ability to find patterns and make predictions far beyond the capabilities of the human mind. However, this gives rise to a series of ethical questions. Can we expect the models to be unbiased towards certain ethnic groups or reduce the preexisting face-to-face bias? Bartlett et al. (2019) find that traditional mortgage lenders charge 7,9 basis points higher rates, for Latin or African-American borrowers after adjusting for educational and social background. The article likewise concludes that financial institutions that have replaced/complemented face-to-face and instead implemented algorithms still discriminate against these ethnic groups, however at a one-third lower rate. This indicate that a data driven approach might lead to a reduction in bias towards minorities, though without removing it all together. The primary goal of Bartlett et al. (2019) was to find if decisions based on algorithms lead to an increase or a decrease in discrimination, and not what might be the cause of this. They however present a few ideas of possible reasons. Minority borrowers might be less prone to examine a large selection of loans, giving lenders a position of statistics on individuals to determine creditworthiness has been utilized for decades. With standard statistical models we can check the inference of the models and control if the relationships make sense. This strength. The bias might also be determined by geographic location, since some areas might be less price competitive. This prices loans higher and if the area have large minority populations this might lead to the discrimination (Bartlett et al., 2019).

The issue of machines replacing humans might be less of importance in the mortgage business, since a lot of customer service still is required. The machine-driven approach might also only be used as a supplement, since certain borrowers might have some idiosyncratic properties which can't be measured and thus requires a more classic approach. One use could also be in not having to explain certain relationships as they can be hidden within blackbox algorithms.

Another aspect that might increase bias could be a reinforcing effect, people won't move to a neighborhood or will avoid certain characteristics to please the models. But this is not a problem isolated only to Machine learning algorithms.

Chapter 6

Conclusion

The objective of this project has been to evaluate the ability of machine learning techniques to predict delinquencies of US homeowners. The algorithms have been able to capture and learn from parts of the data and in general been able to properly predict 20% of the delinquencies with a success rate of 50% - 60%. Is is possible to predict a larger proportion of the delinquencies, this however comes at the cost of a reduced success rate. The Keras neural network was able to predict 50% with a success rate of 35% when the probability threshold was adjusted, both cases clearly beating the reference models. Since a wide variety of models, ranging in complexity, have been used to predict, it is fair to assume additional hyperparameter tuning would only have led to minor increases in performance. In regards to the bias-variance trade-off it seems that a large amount of test errors are related to the irreducible error $Var(\epsilon)$ in equation 2.1. Insufficient data or omitted variables of high importance are presumably a large cause of the lacking prediction results. The primary focus in this project have been on practical implementation, a more theoretical approach might have allowed for identification of additional important factors.

Determining the best performing algorithm is a matter of perspective. Most of the top performing models have both pros and cons. The XGB can capture a larger portion of the delinquencies but at the cost of being more computational demanding. The elastic net model has high predictive quality but is pickier and captures less of the overall amount of delinquencies. Keras neural networks requires additional tuning since they must be constructed manually in R, the other methods are quite a lot easier to implement. If this approach where to be implemented in an institution related to mortgages, the selected models would depend on their willingness to accept certain types of classification errors.

The last part of the project tried to predict borrowers with a delinquency status of six months or more. In this section the models only made limited amount of delinquency predictions of lower quality. The used data and models are insufficient for predictions of borrowers who fall far behind on mortgage payments.

The data exploration revealed several interesting patterns and characteristics of the data. Of especial interest was the fact that the delinquency rates spiked at different years for certain states. Some of these spikes appear to be caused by external factors not contained in the data.

A large amount of additional data and variables where available from Freddie Mac database, these where often omitted due to computational restraints. However, it is not unlikely this data contained additional predictive information. Many avenues are still left to be explored in the data.

Bibliography

Aggarwal, C. C. (2015), *Data Mining: The Textbook*, Springer International Publishing.

Allaire, J. (n.d.), 'R interface to google cloudml'.
URL: *https://tensorflow.rstudio.com/tools/cloudml/getting*$_s tarted/$

Allaire, J. and Chollet, F. (2019), *keras: R Interface to 'Keras'*. R package version 2.2.5.0.
URL: *https://CRAN.R-project.org/package=keras*

Bartlett, R., Morse, A., Stanton, R. and Wallace, N. (2019), 'Consumer-lending discrimination in the fintech era'.

FICO (2019), 'What's in my fico® scores?'.
URL: *https://www.myfico.com/credit-education/whats-in-your-credit-score*

FreddieMac (2019), *Single Family Loan-Level Dataset - General User Guide*, http://www.freddiemac.com/research/datasets/sf$_{l}oanlevel_{d}ataset.html.$

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2014), *An Introduction to Statistical Learning: With Applications in R*, Springer Publishing Company, Incorporated.

Kuhn, M. (2019), *caret: Classification and Regression Training*. R package version 6.0-84.
URL: *https://CRAN.R-project.org/package=caret*

R Core Team (2019), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *https://www.R-project.org/*

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K. and Yutani, H. (2019), 'Welcome to the tidyverse', *Journal of Open Source Software* 4(43), 1686.

Appendix A

Appendix

Table A.1

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
credit score	987.500	754,1	385,9	300	703	782	9.999
area	853.758	NA	NA	NA	NA	NA	NA
units	987.500	1,0	0,3	1	1	1	99
com loan to value	987.500	73,1	18,2	6	64	81	999
loan to value	987.500	72,1	17,9	6	63	80	999
debt to income	987.500	45,2	104,7	1	26	42	999
upb	987.500	192.723	108.258	6.000	112.000	250.000	1.275.000
rate	987.500	5,4	1,4	2,2	4,2	6,4	13,2
term	987.500	330,4	66,0	60	360	360	604
delic mean	987.469	0,1	0,9	0,0	0,0	0,0	59,3
delic binary	987.469	0,1	0,3	0,0	0,0	0,0	1,0
surv binary	986.396	0,1	0,3	0,0	0,0	0,0	1,0
surv	987.469	0,3	1,5	0,0	0,0	0,0	66,0
max unpaid	987.469	192.303	108.294	0,0	111.000	250.000	1.272.000
recovered	974.594	0,1	0,3	0,0	0,0	0,0	1,0
first complete stop	987.469	0,02	0,1	0,0	0,0	0,0	1,0
income	975.838	66.459	45.658	0,0	33.120	88.200	508.800