Credit Risk Analysis - LendingClub Loan data

DATA 621: Business Analytics and Data Mining

Sreejaya, Vuthy and Suman

December 12, 2016

1 Abstract

LendingClub is an online lending platform for loans. Borrowers apply for a loan online, and if accepted, the loan gets listed in the marketplace. As an investor/lender, you can browse the loans in the marketplace, and choose to invest in individual loans at your discretion - basically purchase *notes* backed by payments based on loans. In this project, we will attempt to analyse and predict if a loan is risky or not so that we can avoid investment in high-risk notes.

2 Keywords:

peer-to-peer, dti (debt-to-income ratio), credit utilization ratio, logistic regression, random forest

3 Literature Review

We have reviewed few of the previous projects/analysis/kaggle competitions done in this area (Kaggle, 2011) (Pandey and Srinivasan, 2011) (Liang, 2011) (Chang, Simon, and Genki, 2011). Majority of these were applying machine learning to improve the loan default prediction. Some of these determined that Random Forest appeared to be better performing, and others logistic regression would be better. However, real-world records often behave differently from curated data in competitions like kaggle, so, we will try to apply different regression techniques including the logistic regression and random forest on the real loan data during the years 2012-13 to search for a better predictive model. We will also include additional features like dti, credit utilization etc. from the LendingClub dataset, to see if they influence our target (credit risk) variable.

The questions for logistic regression are, (1) what is the best way to describe the relationship between predictor and outcome, and (2) Is the relationship statistically significant - does knowing something about the predictor variable enable a better prediction of the outcome than knowing nothing about the predictor ?. The logistic regression equation takes the form:

$$log\ odds = \log\left(\frac{p}{1-p}\right) = b_0 + b_1 * (predictor\ value)$$

where b_1 is the amount of increase in the log odds of the outcome given by an increase in one unit of the predictor, and b_0 is the intercept of the model. So, the logistic regression does *not* assume a linear relationship between the dependent and independent variables (but between the logit of the outcome and the predictor values). There are several sophisticated alternatives to logistic regression techniques available, such as decision trees, random forests, neural networks etc. (Reed and Wu, 2013)

Random forest is a nonparametric machine learning strategy that can be used for building a risk prediction model in survival analysis. In survival settings, the predictor is an *ensemble* formed by combining the results of many survival trees. (Mogensen, Ishwaran, and Gerds, 2012)

4 Methodology

4.1 Data Exploration

The dataset we used for this analysis is from publicly available data from LendingClub.com, which is basically an online market place that connects borrowers and investors. As part of the data exploration, we will perform *Exploratory data analysis* to better understand the relationships in the given data including correlations, feature distributions and basic summary statistics. We will also identify the outliers, missing data and any look for invalid data.

4.2 Data Preparation

As part of the data preparation, we will try to fix the data issues we noticed in our exploratory analysis, which involves treating the outliers, missing data, invalid data etc. We will also identify the data classifications, and create dummy variables wherever required, and will convert the categorical data to numeric data which would help us later in model building.

4.3 Model Development

Our primary objective here is to *identify risky loans*, which is binary outcome, so we will be using *logistics regression, and random forest* modeling techniques to examine and predict the credit risk using a training subset (80%) of the original full data.

4.4 Model Validation

A validation subset (20%) was used to test how well our candidate models predict the target variable. AUC , the model Accuracy and mis-classification error will be used to select a better predicting model.

5 Experimentation and Results

5.1 Data Exploration and Preparation

5.1.1 Feature Selection:

There are 111 fields and 188K observations. However, not all fields are intuitively useful for our model building, such as the loan ID, member ID, last payment month etc., so we will be removing such fields. We will also be removing features with majority of NAs (80% NAs). In order to label the dataset, we will classify the loans that defaulted, charged off, or were late on payments as negative cases, and those that were fully paid or current was classified as positive loans.

5.1.2 Outliers:

Look for outliers and remove or transform the outliers that might affect the model. We are going to consider the loans with annual income of less than 250K for the model building.

5.1.3 Tidy Data:

Convert the date fields such as issue date to a proper date format, and remove % sign for interest rates, dti and convert those into numeric values. Furthermore, we will consider only matured loans. [issue date + term months < today], and remove variables where more than 25% of the observations are missing values.

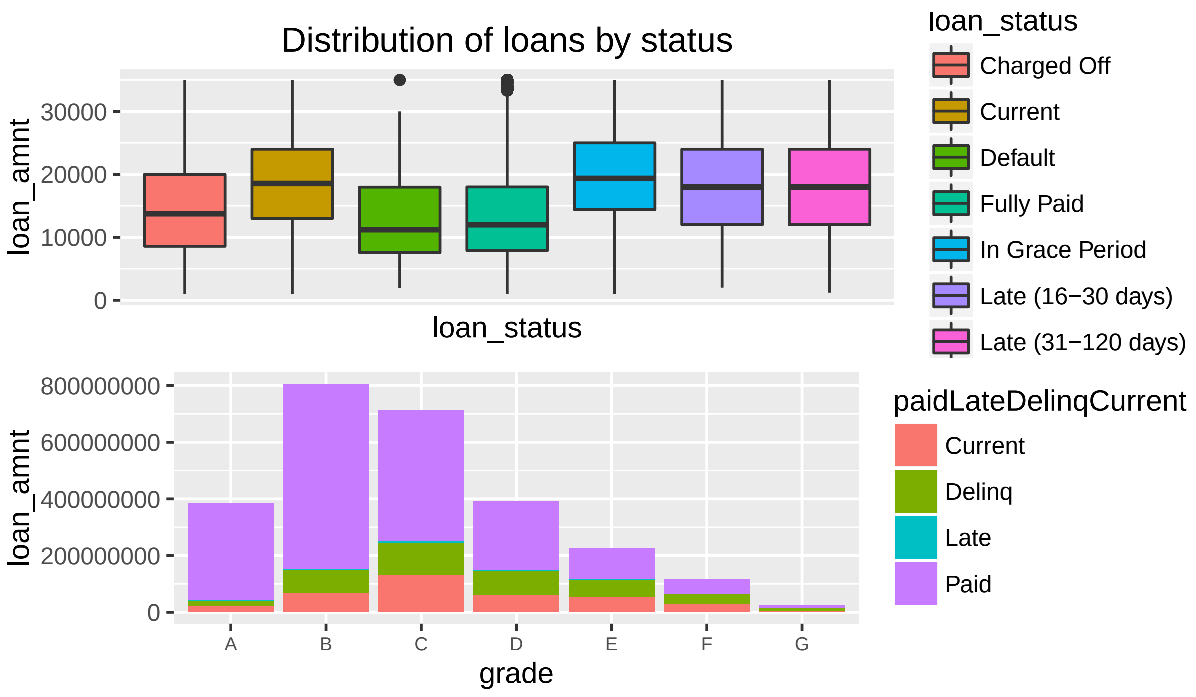


Figure 1: Distribution of loans by status and grade

Factorize the loan status levels with proper ordering ('Charged Off', 'Default', 'Late (31-120 days)', 'Late (16-30 days)', 'In Grace Period', 'Current', 'Fully Paid'). Also loans issued by Lending Club fall into three categories of verification: "income verified," "income source verified," and "not verified." The "home ownership" is another factor variable provided by the borrower during registration or obtained from the credit report. The values are: RENT, OWN, MORTGAGE, OTHER.

Create a factor label *bad.loan* which indicates a loan is bad if it is delinquent, or late consider as negative, otherwise positive. This would be our response variable which indicates a loan is bad or not.

5.1.4 Shortlist Numeric variables:

Identify the numeric variable, and compare how those distributions plot against the good, bad label, and pick a few variables with differences in the bad and good populations.(yhat, 2013)

Lets visualize the correlation graph of these numeric variable:

From our numerical feature list, it appears like the *total_pymnt*, and *total_rec_prncp* has high correlation with the *bad_loan* classification.

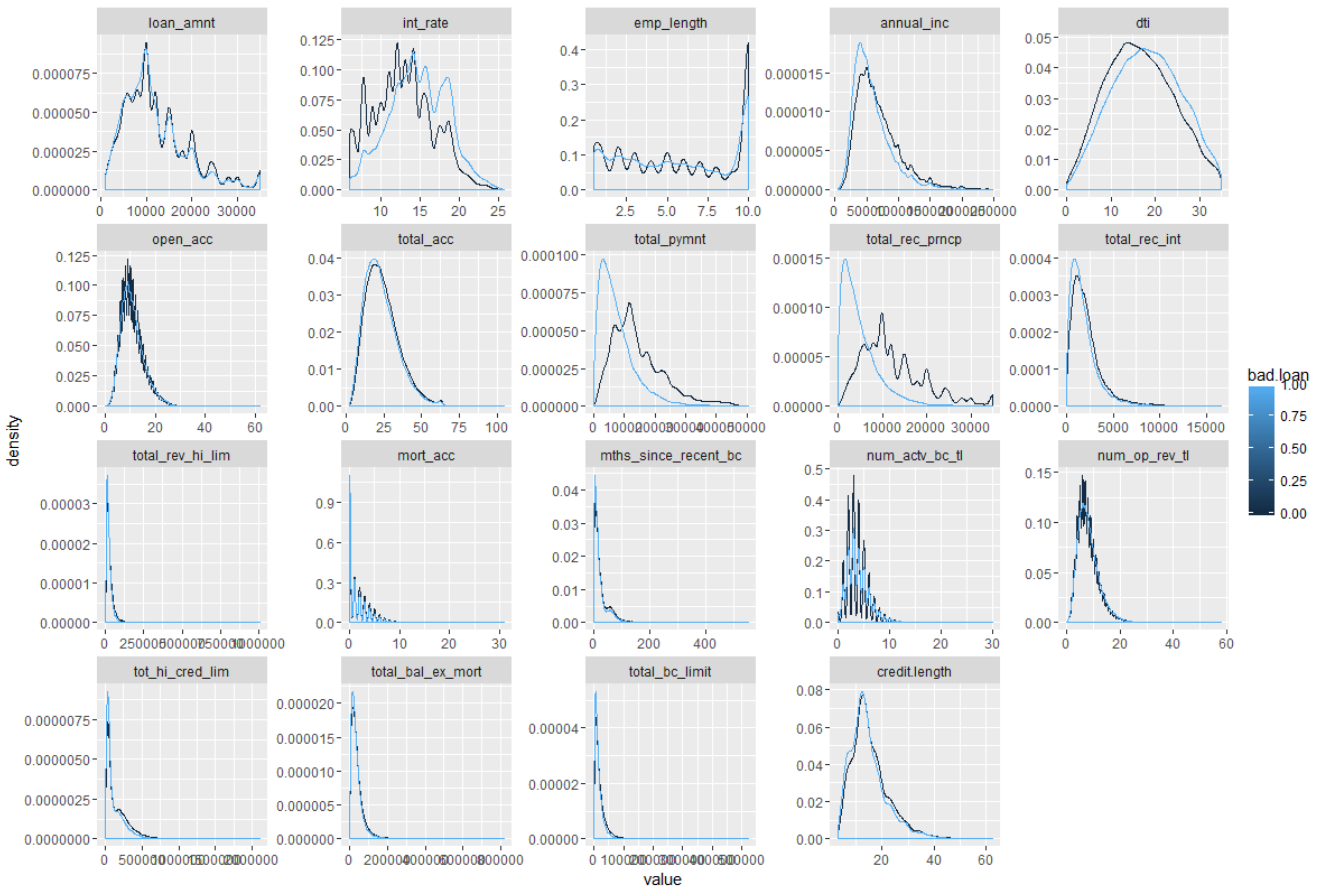


Figure 2: Data Distributions

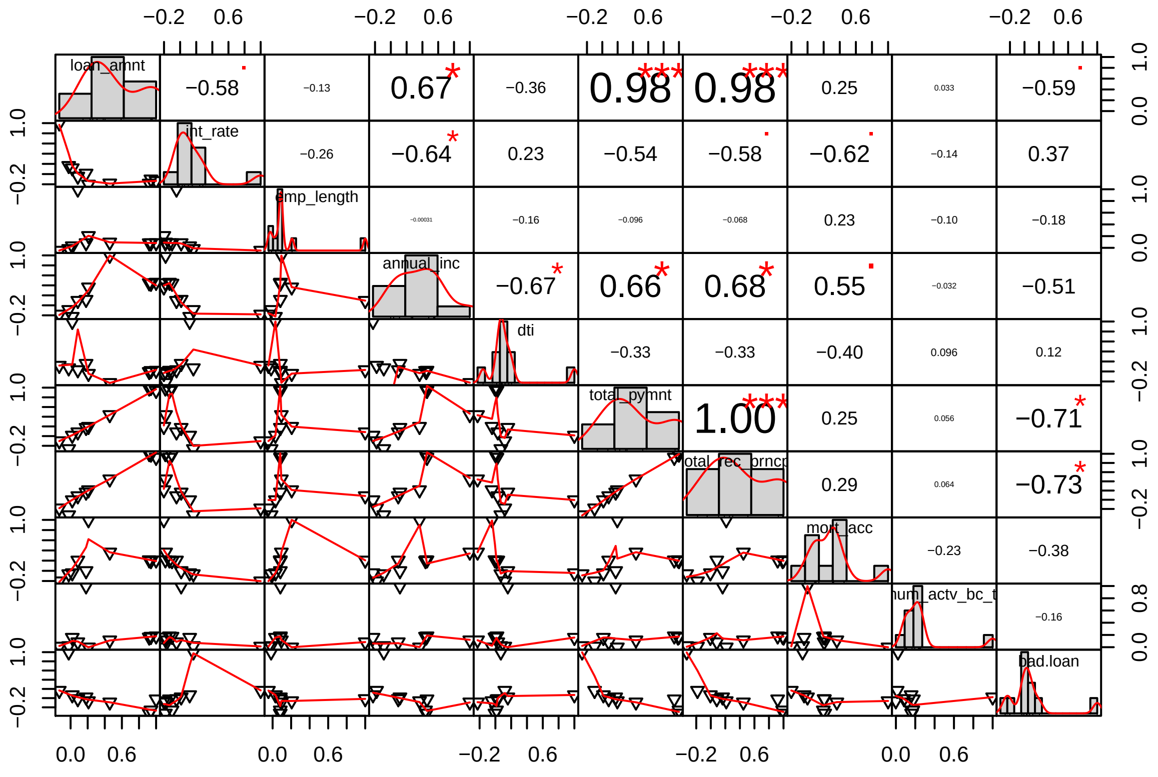


Figure 3: Correlations

5.1.5 Dummy variables:

Since R's glm function will take care of the dummy variables, we just need to make sure that the categorical variables are factorized.

5.1.6 Split the dataset into training and test:

We will randomly split our dataset into training (80%) and test (20%).

```
## Observations: 114,190
## Variables: 14
## $ loan_amnt           <int> 14400, 15000, 10000, 15000, 35000, 5600, 4...
## $ int_rate            <dbl> 12.12, 11.14, 10.74, 7.90, 14.30, 18.75, 2...
## $ grade               <ord> B, B, B, A, C, D, E, B, C, B, A, C, B, D, ...
## $ emp_length          <dbl> 8.0, 6.0, 2.0, 2.0, 2.0, 6.0, 2.0, 0.5, 10...
## $ home_ownership      <fctr> RENT, MORTGAGE, RENT, MORTGAGE, MORTGAGE,...
## $ annual_inc          <dbl> 45000, 78000, 170000, 104000, 170000, 6200...
## $ verification_status <fctr> Source Verified, Verified, Source Verifie...
## $ issue_d             <date> 2012-09-01, 2012-12-01, 2012-03-01, 2013-...
## $ dti                 <dbl> 22.32, 22.72, 4.59, 19.54, 9.00, 21.43, 3....
## $ total_pymnt         <dbl> 17247.966, 17194.223, 9498.950, 16299.200,...
## $ total_rec_prncp     <dbl> 14400.0, 15000.0, 7474.0, 15000.0, 35000.0...
## $ mort_acc            <dbl> 0, 2, 0, 6, 2, 1, 0, 0, 5, 5, 1, 0, 2, 2, ...
## $ num_actv_bc_tl      <dbl> 4, 4, 0, 2, 3, 3, 3, 1, 8, 2, 4, 6, 11, 4,...
## $ bad.loan            <fctr> 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,...
```

Number of observations in *training* dataset is 114190

Number of observations in *test* dataset is 28548

5.2 Model Development

5.2.1 Model 1 - Logistic regression considering all predictors:

	GVIF	Df	GVIF^(1/(2*Df))
loan_amnt	210.211433	1	14.498670
int_rate	13.789034	1	3.713359
grade	13.158463	6	1.239559
emp_length	1.103136	1	1.050303
home_ownership	1.473040	4	1.049607
annual_inc	1.451972	1	1.204978
verification_status	1.160992	2	1.038024
issue_d	1.732473	1	1.316234
dti	1.127885	1	1.062019
total_pymnt	270.667965	1	16.451990
total_rec_prncp	330.533383	1	18.180577
mort_acc	1.518237	1	1.232168
num_actv_bc_tl	1.442362	1	1.200984

Noticed high multicollinearity in the predictors, so removing the high VIF variables - loan_amnt, int_rate, grade, total_pymnt and total_rec_prncp and try to fit again.

5.2.2 Model 2 - Logistic regression removing high multicollinear predictors:

.rownames	Estimate	Std..Error	z.value	Pr...z..
(Intercept)	5.2784	0.8105	6.5123	0.0000
emp_length	0.0044	0.0027	1.6328	0.1025
home_ownershipNONE	0.3928	0.4994	0.7866	0.4315
home_ownershipOTHER	0.0658	0.5442	0.1209	0.9038
home_ownershipOWN	0.0772	0.0342	2.2549	0.0241
home_ownershipRENT	0.1914	0.0224	8.5245	0.0000
annual_inc	0.0000	0.0000	-21.9819	0.0000
verification_statusSource Verified	0.1420	0.0243	5.8476	0.0000
verification_statusVerified	0.1324	0.0211	6.2676	0.0000
issue_d	-0.0005	0.0001	-8.9192	0.0000
dti	0.0171	0.0012	13.6957	0.0000
mort_acc	-0.0330	0.0058	-5.6788	0.0000
num_actv_bc_tl	0.0239	0.0043	5.6010	0.0000

5.2.3 Model 3 - Random Forest, considering all predictors :

Since the decision trees could be susceptible to noise in the training data, lets directly go with random forest, which would randomly select observations and variables, and build several trees and takes majority vote from all these trees for our prediction. With 500 trees, the random forest shows the below Gini index - a measure how each variable contribute to the homogeneity of the nodes and leaves in the resulting random forest. The *principal amount received to date*, *loan amount*, *payments received to date for total amount funded*, *debt to income ratio*, *interest rate*, and *annual income* are some of the high important variables here. Its also interesting to see that *issue_d* is also shown as one of the important variables.

```
##
## Call:
##  randomForest(formula = bad.loan ~ ., data = loans.ss.train, do.trace = F,      type = "class", importan
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 0.9%
## Confusion matrix:
##       0     1  class.error
## 0 99679    24 0.0002407149
## 1  1009 13478 0.0696486505
```

Here, Out Of Bag error estimate can be taken as estimate of performance on unseen data.

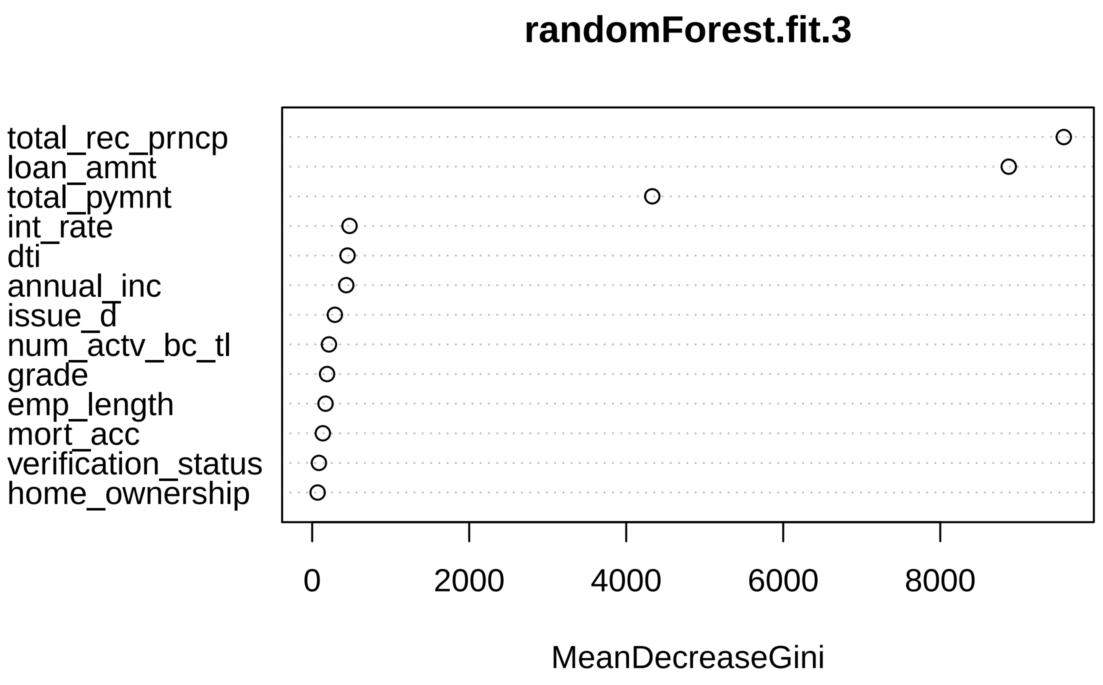


Figure 4: Random Forest - Variable Importance Plot

5.2.4 Model 4 - Random Forest, removing high multicollinear predictors :

Lets generate the *random forest* model by considering low VIF predictors only and review the important features:

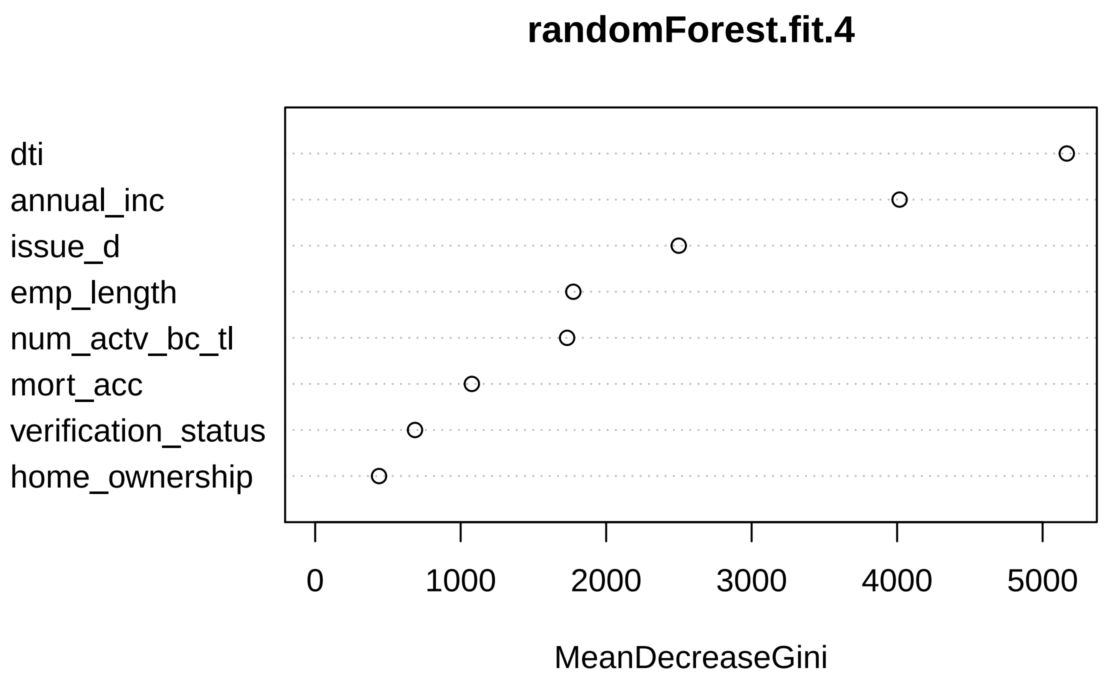


Figure 5: Random Forest - Variable Importance Plot

5.2.5 Model 5 - Logistic regression considering the features from "random forest model":

We thought, even if we can not use randorm forest as primary tool for some reason, we can still use it to select important variables from our data. We will consider the top 6 important features resulted from the *random forest* and build the logistic regression model.

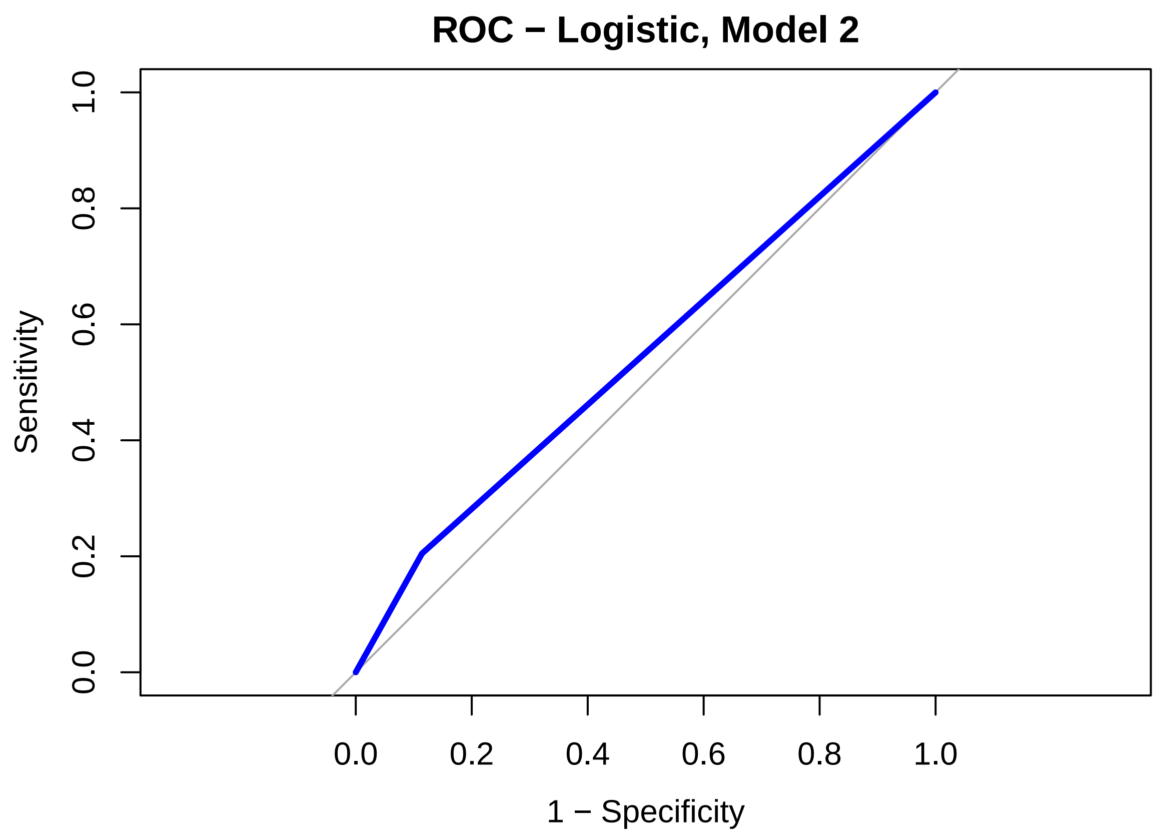
.rownames	Estimate	Std..Error	z.value	Pr...z..
(Intercept)	5.3463	0.8100	6.6005	0.0000
dti	0.0184	0.0012	15.1492	0.0000
annual_inc	0.0000	0.0000	-22.1280	0.0000
issue_d	-0.0005	0.0001	-8.7677	0.0000
num_actv_bc_tl	0.0241	0.0043	5.6462	0.0000
emp_length	0.0017	0.0027	0.6410	0.5215
mort_acc	-0.0521	0.0052	-9.9972	0.0000

5.3 Model Validation

Now that our models are trained, lets apply those to the test dataset and derive the performance measures.

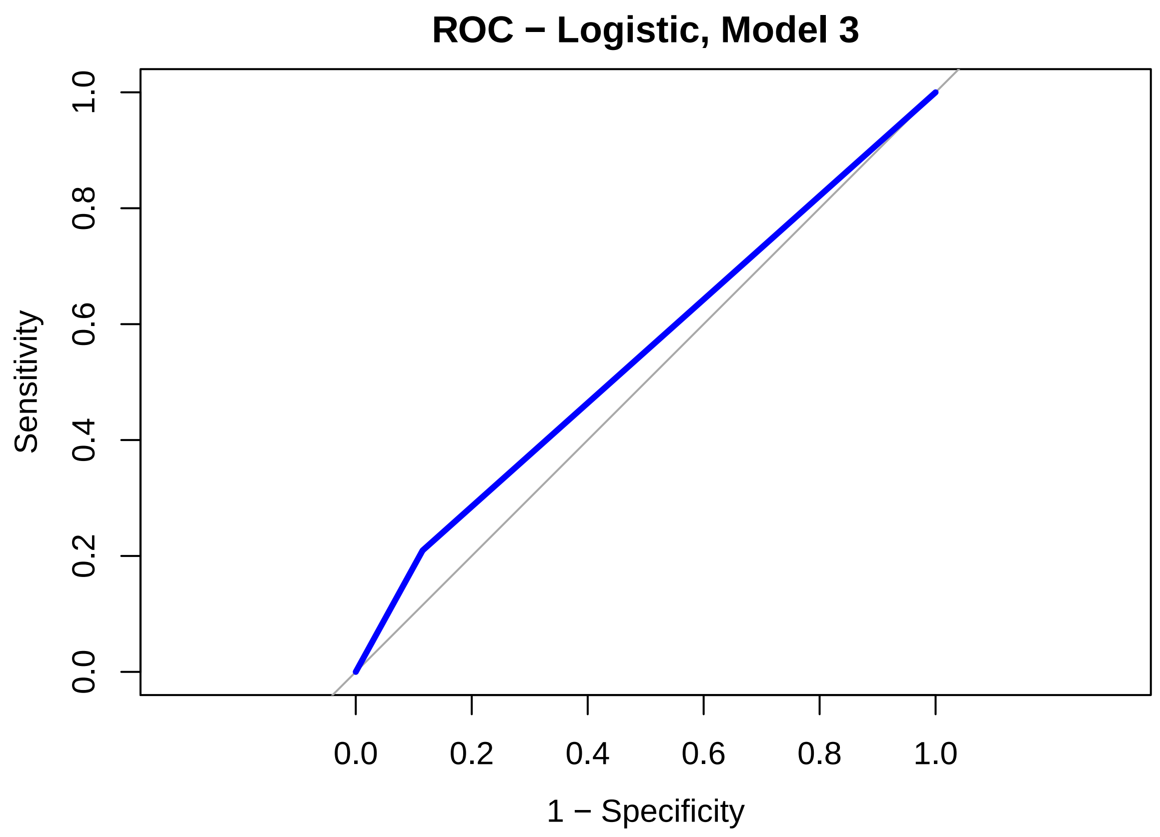
$$Sensitivity = \frac{TP}{TP + FN}$$
$$Specificity = \frac{TN}{TN + FP}$$
$$Accuracy = \frac{TP + TN}{(TP + FN) + (FP + TN)}$$

For an ideal model, the predictions will be perfect - meaning the 'accuracy, sensitivity and specificity' will all be 1 where as the mis-classification error will be zero. In practical scenarios we would like to have the sensitivity and spcificity as close to 1 as possible.



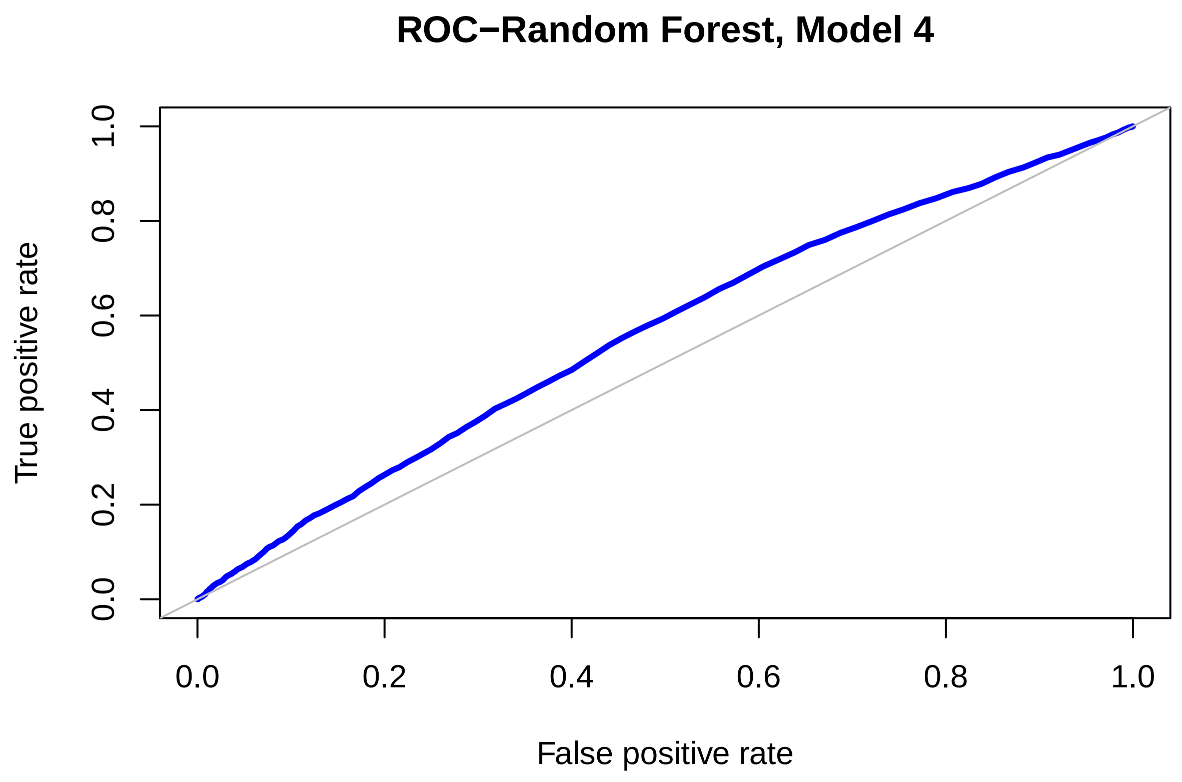
```
##
## Call:
## roc.formula(formula = factor(predicted) ~ as.numeric(bad.loan),     data = loans.ss.test, plot = FALSE,
##
## Data: as.numeric(bad.loan) in 25842 controls (factor(predicted) 0) < 2706 cases (factor(predicted) 1).
## Area under the curve: 0.5453
## 95% CI: 0.5374-0.5531 (DeLong)
```

The Area under the curve for the above plot is 0.54, and the accuracy is 0.82.

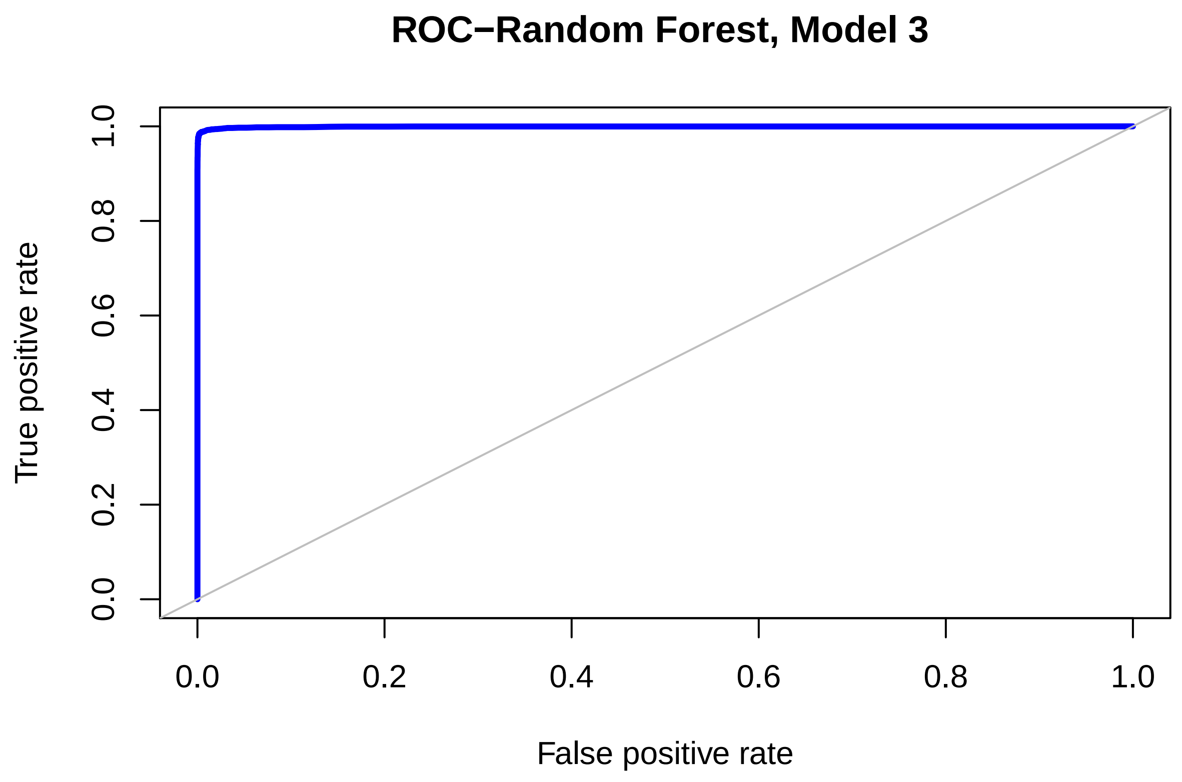


```
##
## Call:
## roc.formula(formula = factor(predicted) ~ as.numeric(bad.loan),     data = loans.ss.test, plot = FALSE,
##
## Data: as.numeric(bad.loan) in 26271 controls (factor(predicted) 0) < 2277 cases (factor(predicted) 1).
## Area under the curve: 0.5471
## 95% CI: 0.5385-0.5557 (DeLong)
```

The Area under the curve for the above plot is 0.55, and the accuracy is 0.83, which is slightly better than the previous model.



Random forest model (with selective variables) resulted in AUC of 0.56 and the accuracy of 0.88, whih is slightly better than the logistic models. Lets see the random forest model with all variables included.



This resulted is pretty good accurancy and auc measures ! , both are about 99% !

Lets summarize the results of our selected logistic and random forest models:

Model	Accuracy	AUC
Logistic, Model 2	0.8212	0.5453
Logistic, Model 5	0.8309	0.5471
Random Forest, Model 4	0.8772	0.5636
Random Forest, Model 3	0.9915	0.9993

From the above, it is evident that *Random Forest, Model 3* performed well here. The AUC [P(predicted TRUE|actual TRUE) Vs P(FALSE|FALSE)] and the accuracy [P(TRUE|TRUE).P(actual TRUE) + P(FALSE|FALSE).P(actual FALSE)], are both higher compared to the other models.

6 Conclusion

Depending on the features we have chosen for this study, the *random forest* model provided the best outcome out of the few we analysed. So, the limitation here is our selection of the predictors (14 out of 111) and possibly the overall dataset. The logistic regression models interstingly showed high accuracy and low AUC, this could be due the cutoff point we have chosen for the accuracy, and AUC here indicates that the probability of classifying a loan is risky is roughly around 55%.

Future work - we will try including more predictors, and possibly loading data before the year 2012. We will also plan on including the *Naive Bayes* classification analysis and multinomial logistic regression techniques to the model suite.

7 References

Chang, S., Simon and Genki. Predicting Default Risk of Lending Club Loans. 2011. URL: http://cs229.stanford.edu /proj2015/199_report.pdf.

Kaggle. GiveMeSomeCredit. 2011. URL: https://www.kaggle.com/c/GiveMeSomeCredit.

Liang, J. Predicting Borrowers Chance Of Defaulting On Credit Loans. Kaggle Submission. 2011. URL: http://cs229.stanford.edu/proj2011/JunjieLiang-PredictingBorrowersChanceOfDefaultingOnCreditLoans.pdf.

Mogensen, U., H. Ishwaran and T. Gerds. "Evaluating Random Forests for Survival Analysis". In: Journal of Statistical Software (2012). URL: https://www.jstatsoft.org/article/view/v050i11.

Pandey, J. and M. Srinivasan. Predicting Probability of Loan Default. 2011. URL: http://cs229.stanford.edu/proj2011/ PandeySrinivasan-PredictingProbabilityOfLoanDefault.pdf.

Reed, P. and Y. Wu. "Logistic regression for risk factor modeling in stuttering research". In: Journal of Fluency Disorders (2013). URL: https://www.ucl.ac.uk/speech-research-group/publications/publications-files/reed_2013_logistic_regression.pdf.

yhat. Machine Learning for Predicting Bad Loans. 2013. URL: http://blog.yhat.com/posts/machine-learning-for-predicting-bad-loans.html.

7.1 Appendix : R code

```r
library(knitcitations)
library(RefManageR)
knitr::opts_chunk$set(warning = FALSE, message = FALSE, echo = FALSE)
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 80), tidy = TRUE)
options(scipen = 999)
if (!require("devtools", character.only = TRUE)) (install.packages("devtools", dep = TRUE))
if (!require("yaml", character.only = TRUE)) (install.packages("yaml", dep = TRUE))
if (!require("rprojroot", character.only = TRUE)) (install.packages("rprojroot",
    dep = TRUE))
if (!require("papaja", character.only = TRUE)) (install.packages("papaja", dep = TRUE))
if (!require("ggplot2", character.only = TRUE)) (install.packages("ggplot2", dep = TRUE))
if (!require("randomForest", character.only = TRUE)) (install.packages("randomForest",
    dep = TRUE))
if (!require("broom", character.only = TRUE)) (install.packages("broom", dep = TRUE))
if (!require("ROCR", character.only = TRUE)) (install.packages("ROCR", dep = TRUE))
if (!require("knitcitations", character.only = TRUE)) (install.packages("knitcitations",
    dep = TRUE))
if (!require("RefManageR", character.only = TRUE)) (install.packages("RefManageR",
    dep = TRUE))

library(knitcitations)
library(RefManageR)
library(devtools)
library(yaml)
library(papaja)
library(rprojroot)
library(ggplot2)
library(gridExtra)
library(knitr)
library(car)
library(dplyr)
library(reshape2)
library(PerformanceAnalytics)
library(lubridate)
library(randomForest)
library(broom)
library(caret)
library(pROC)
library(ROCR)


Kaggle <- bibentry(bibtype = "Misc", publisher = "Kaggle", author = personList(person(family = "Kaggle",
    given = "")), title = "GiveMeSomeCredit", year = 2011, url = "https://www.kaggle.com/c/GiveMeSomeCredit


Jitendra <- bibentry(bibtype = "Misc", author = personList(person(family = "Pandey",
    given = "Jitendra"), person(family = "Srinivasan", given = "Maheshwaran")), title = "Predicting Probabi
    year = 2011, url = "http://cs229.stanford.edu/proj2011/PandeySrinivasan-PredictingProbabilityOfLoanDefa

Junjie <- bibentry(bibtype = "Misc", author = personList(person(family = "Liang",
    given = "Junjie")), title = "Predicting Borrowers Chance Of Defaulting On Credit Loans",
    year = 2011, note = "Kaggle Submission", url = "http://cs229.stanford.edu/proj2011/JunjieLiang-Predicti


Shunpo <- bibentry(bibtype = "Misc", author = personList(person(family = "Chang",
    given = "Shunpo"), person(family = "Simon", given = ""), person(family = "Genki",
    given = "")), title = "Predicting Default Risk of Lending Club Loans", year = 2011,
    url = "http://cs229.stanford.edu
                 /proj2015/199_report.pdf")

yhat <- bibentry(bibtype = "Misc", author = personList(person(family = "yhat", given = "")),
    title = "Machine Learning for Predicting Bad Loans", year = 2013, url = "http://blog.yhat.com/posts/mac


RForest <- bibentry(bibtype = "Misc", author = personList(person(family = "Coffey",
    given = "Kimberly")), title = "Random Forest for Loan Performance Prediction",
    year = 2016, url = "http://www.kimberlycoffey.com/blog/2016/3/19/random-forest")

PhilReed <- bibentry(bibtype = "Article", author = personList(person(family = "Reed",
    given = "Phil"), person(family = "Wu", given = "Yaqionq")), title = "Logistic regression for risk facto
    year = 2013, journal = "Journal of Fluency Disorders", url = "https://www.ucl.ac.uk/speech-research-gro

Ulla <- bibentry(bibtype = "Article", author = personList(person(family = "Mogensen",
    given = "Ulla"), person(family = "Ishwaran", given = "Hemant"), person(family = "Gerds",
    given = "Thomas")), title = "Evaluating Random Forests for Survival Analysis",
    year = 2012, journal = "Journal of Statistical Software", url = "https://www.jstatsoft.org/article/view
```

```r
# If file not found, then Load the file directly from lendingclub, unzip.
filename <- "LoanStats3b.csv"
if (!file.exists(filename)) {
    url <- "https://resources.lendingclub.com/LoanStats3b.csv.zip"
    download.file(url, destfile = "LendingClubLoans2012-13.csv.zip")
    unzip("LendingClubLoans2012-13.csv.zip", overwrite = TRUE)

}

# for practice only loading 10K rows
loans <- read.csv(filename, skip = 1, header = TRUE)

# 1. find the fields with majority of NA get rid of fields that are mainly NA
majority_na <- sapply(loans, function(x) {
    coverage <- 1 - sum(is.na(x))/length(x)
    coverage < 0.8
})

loans <- loans[, majority_na == FALSE]

# 2. Just capture the important features
shortlisted.var = c("loan_amnt", "term", "int_rate", "grade", "sub_grade", "emp_title",
    "emp_length", "home_ownership", "annual_inc", "verification_status", "issue_d",
    "loan_status", "purpose", "addr_state", "dti", "earliest_cr_line", "open_acc",
    "total_acc", "total_pymnt", "total_rec_prncp", "total_rec_int", "open_il_6m",
    "open_il_12m", "open_il_24m", "mths_since_rcnt_il", "total_bal_il", "il_util",
    "all_util", "total_rev_hi_lim", "mort_acc", "mths_since_recent_bc", "mths_since_recent_bc_dlq",
    "num_actv_bc_tl", "num_op_rev_tl", "tot_hi_cred_lim", "total_bal_ex_mort", "total_bc_limit")

loans.ss <- loans %>% select(which(names(loans) %in% shortlisted.var))

# 3. Looks for outliers.

# income
g.income <- ggplot(loans.ss, aes(x = "annual_inc", y = annual_inc)) + geom_boxplot()

# Data Transfromation -- Takes the base-10 logarithm of annual_inc
# loans.ss$log_annual_inc <- log10(loans.ss$annual_inc +1 )
g.loanAmt <- ggplot(loans.ss, aes(x = "loan_amnt", y = log_annual_inc)) + geom_boxplot()

# Lets consider loans with annual income < 250000
loans.ss <- loans.ss[which(loans.ss$annual_inc < 250000), ]

# loan amount , looks like this is not bad.
g.loanAmt <- ggplot(loans.ss, aes(x = "loan_amnt", y = loan_amnt)) + geom_boxplot()

# grid.arrange(g.income, g.loanAmt, nrow=1, top = 'Outliers')


# the int_rate variable is a factor with a percentage sign, so we need to derivce
# a numeric
loans.ss$int_rate <- (as.numeric(gsub("%", "", loans.ss$int_rate)))

g1 <- ggplot(loans.ss, aes(loan_amnt, col = grade)) + geom_histogram(bins = 50) +
    facet_grid(grade ~ .)
```

```
g2 <- ggplot(loans.ss, aes(int_rate, fill = grade)) + geom_density() + facet_grid(grade ~
    .)

# grid.arrange(g1, g2, nrow=1, top = 'Loan Grades 'Vs' Loan Amt, Int Rate')

g3 <- ggplot(loans.ss, aes(loan_status, loan_amnt, fill = loan_status)) + geom_boxplot() +
    scale_x_discrete(breaks = NULL) + ggtitle("Distribution of loans by status")

loans.ss$paidLateDelinqCurrent <- "Current"
loans.ss$paidLateDelinqCurrent[which(loans.ss$loan_status == "Fully Paid")] <- "Paid"
loans.ss$paidLateDelinqCurrent[which(loans.ss$loan_status == "Charged Off" | loans.ss$loan_status ==
    "Default")] <- "Delinq"
loans.ss$paidLateDelinqCurrent[which(loans.ss$loan_status == "Late (16-30 days)" |
    loans.ss$loan_status == "Late (31-120 days)")] <- "Late"

loans.ss$paidLateDelinqCurrent <- factor(loans.ss$paidLateDelinqCurrent)

# g1 <- ggplot(loans.ss, aes(paidLateDelinqCurrent, loan_amnt, fill =
# #paidLateDelinqCurrent)) + geom_bar(stat = 'identity') +
# theme(legend.position='none')

loan_by_grade <- aggregate(loan_amnt ~ grade + paidLateDelinqCurrent, data = loans.ss,
    sum)
gbar <- ggplot(loan_by_grade, aes(grade, loan_amnt, fill = paidLateDelinqCurrent))
g4 <- gbar + geom_bar(stat = "identity") + theme(axis.text.x = element_text(size = 7))

# grid.arrange(g1, g2, g3, g4, nrow=2)
grid.arrange(g3, g4, nrow = 2)
loans.paidLateDelinqCurrent <- NULL

# set up the issue date and the earliest credit line to Date types
loans.ss$issue_d <- as.Date(paste(paste("01", substr(loans.ss$issue_d, 1, 3), sep = ""),
    substr(loans.ss$issue_d, 5, 9), sep = ""), "%d%B%Y")

loans.ss$FirstCreditDate <- as.Date(paste(paste("01", substr(loans.ss$earliest_cr_line,
    1, 3), sep = ""), substr(loans.ss$earliest_cr_line, 5, 9), sep = ""), "%d%B%Y")

loans.ss$credit.length <- as.numeric((loans.ss$issue_d - loans.ss$FirstCreditDate)/365)


# Remove column with xx month to numeric variables
loans.ss$dti <- (as.numeric(gsub("%", "", loans.ss$dti)))

# Employment Length
loans.ss$emp_length <- gsub(" years", "", loans.ss$emp_length)
loans.ss$emp_length <- gsub(" year", "", loans.ss$emp_length)
loans.ss$emp_length <- ifelse(loans.ss$emp_length == "10+", 10, loans.ss$emp_length)
loans.ss$emp_length <- ifelse(loans.ss$emp_length == "< 1", 0.5, loans.ss$emp_length)
loans.ss$emp_length <- suppressWarnings(as.numeric(loans.ss$emp_length))

# Convert character to ordinal variable
loans.ss$grade[loans.ss$grade == ""] <- NA
loans.ss$grade <- ordered(loans.ss$grade)

# Identify loans that have already come to term
loans.ss <- loans.ss %>% mutate(term.months = as.numeric(ifelse(term == " 36 months",
```

```r
    36, 60))) %>% select(-term)


loans.ss$maturity.date <- loans.ss$issue_d + months(loans.ss$term.months)
today <- Sys.Date()
loans.ss$mature <- ifelse(loans.ss$maturity.date < today, 1, 0)
loans.ss$maturity.date <- NULL
remove(today)

# subset data to select only mature loans
loans.ss <- subset(loans.ss, mature == 1)

# Remove variables where more than 25% of the observations are missing values
loans.ss <- loans.ss[, colMeans(is.na(loans.ss)) <= 0.25]

loans.ss$FirstCreditDate <- NULL
loans.ss$earliest_cr_line <- NULL

# Set the proper order for the loan_status factor
loans.ss$loan_status <- factor(loans.ss$loan_status, levels = c("Charged Off", "Default",
    "Late (31-120 days)", "Late (16-30 days)", "In Grace Period", "Current", "Fully Paid"))


# Remove variables where all values are the same
loans.ss <- loans.ss[sapply(loans.ss, function(x) length(levels(factor(x))) > 1)]

neg_ind <- c("Default", "Late (31-120 days) Charged Off", "Late (16-30 days)", "Charged Off")
loans.ss <- loans.ss %>% mutate(bad.loan = ifelse(loan_status %in% neg_ind, 1, ifelse(loan_status ==
    "", NA, 0)))

# figure out which columns are numeirc (and hence we can look at the
# distribution)
numeric_cols <- sapply(loans.ss, is.numeric)

# plot the distribution for bads and goods for each variable
loans.long <- melt(loans.ss[, numeric_cols], id = "bad.loan")

p <- ggplot(aes(x = value, group = bad.loan, colour = bad.loan), data = loans.long)

# relationship of variables with good/bad label. since we do not have enough
# space , we are not showing this.
p + geom_density() + facet_wrap(~variable, scales = "free")

remove.numeric.vars <- c("open_acc", "total_acc", "total_rec_int", "total_rev_hi_lim",
    "mths_since_recent_bc", "num_op_rev_tl", "tot_hi_cred_lim", "total_bal_ex_mort",
    "total_bc_limit", "credit.length", "CreditLength")

loans.ss <- loans.ss %>% select((which(!(names(loans.ss) %in% remove.numeric.vars))))
# glimpse(loans.ss)

numeric_cols <- sapply(loans.ss, is.numeric)
cor.matrix <- cor(loans.ss[, numeric_cols], use = "complete.obs")
chart.Correlation(cor.matrix, histogram = TRUE, pch = 25)

# Remove variables that provide additional outcome data about the loan
loans.ss$emp_title <- NULL
```

```r
# loans.ss$issue_d <- NULL
loans.ss$addr_state <- NULL
loans.ss$paidLateDelinqCurrent <- NULL
loans.ss$sub_grade <- NULL
loans.ss$loan_status <- NULL
loans.ss$purpose <- NULL

# Remove factor vars with too many levels
too.many.levels <- function(x) {
    is.factor(x) == TRUE & length(levels(x)) > 32
}
delete <- lapply(loans.ss, too.many.levels)
loan <- loans.ss[, delete == FALSE]
remove(too.many.levels, delete)

loans.ss$bad.loan <- as.factor(loans.ss$bad.loan)


medianVal <- median(loans.ss$emp_length, na.rm = TRUE)
loans.ss$emp_length <- ifelse(is.na(loans.ss$emp_length) == TRUE, medianVal, loans.ss$emp_length)

loans.ss$mort_acc <- ifelse(is.na(loans.ss$mort_acc) == TRUE, 0, loans.ss$mort_acc)

loans.ss$num_actv_bc_tl <- ifelse(is.na(loans.ss$num_actv_bc_tl) == TRUE, 0, loans.ss$num_actv_bc_tl)


set.seed(18)

s0 = sample(1:nrow(loans.ss), 0.8 * nrow(loans.ss))
loans.ss.train = loans.ss[s0, ]
loans.ss.test = loans.ss[-s0, ]

glimpse(loans.ss.train)

logit.fit.1 <- glm(bad.loan ~ ., data = loans.ss.train, family = binomial(link = "logit"),
    control = list(maxit = 50))

# kable(tidy(round(summary(logit.fit.1)$coef,4)))

# Lets check the Multicollinearity
vif.logit.fit1 <- vif(logit.fit.1)
knitr::kable(vif.logit.fit1)

logit.fit.2 <- glm(bad.loan ~ . - loan_amnt - int_rate - grade - total_pymnt - total_rec_prncp,
    data = loans.ss.train, family = binomial(link = "logit"), control = list(maxit = 50))

# vif looks great here. vif.logit.fit2 <- vif(logit.fit.2)
# knitr::kable(vif.logit.fit2)

# round(summary(logit.fit.2)$coef,4)
kable(tidy(round(summary(logit.fit.2)$coef, 4)))

randomForest.fit.3 <- randomForest(bad.loan ~ ., data = loans.ss.train, do.trace = F,
    type = "class", importance = TRUE, na.action = na.omit)
randomForest.fit.3
varImpPlot(randomForest.fit.3, type = 2)
```

```
randomForest.fit.4 <- randomForest(bad.loan ~ . - loan_amnt - int_rate - grade -
    total_pymnt - total_rec_prncp, data = loans.ss.train, do.trace = F, type = "class",
    importance = TRUE, na.action = na.omit)
varImpPlot(randomForest.fit.4, type = 2)

logit.fit.5 <- glm(bad.loan ~ +dti + annual_inc + issue_d + num_actv_bc_tl + emp_length +
    mort_acc, data = loans.ss.train, family = binomial(link = "logit"), control = list(maxit = 50))

# vif looks great here. vif.logit.fit5 <- vif(logit.fit.5)
# knitr::kable(vif.logit.fit5)

# round(summary(logit.fit.5)$coef,4)
kable(tidy(round(summary(logit.fit.5)$coef, 4)))

performance.results <- vector()

loans.ss.test$score <- predict(logit.fit.2, newdata = loans.ss.test, type = "response")
# ggplot(loans.ss.test, aes(y=bad.loan, x=score, color=bad.loan)) + geom_point()
# + geom_jitter()

cutoff = 0.18
loans.ss.test$predicted = as.numeric(loans.ss.test$score > cutoff)
TP = sum(loans.ss.test$predicted == 1 & loans.ss.test$bad.loan == 1)
FP = sum(loans.ss.test$predicted == 1 & loans.ss.test$bad.loan == 0)
FN = sum(loans.ss.test$predicted == 0 & loans.ss.test$bad.loan == 1)
TN = sum(loans.ss.test$predicted == 0 & loans.ss.test$bad.loan == 0)

# lets also calculate total number of real positives and negatives in the data
P = TP + FN
N = TN + FP
total = P + N

# confusionMatrix(factor(loans.ss.test$predicted),
# factor(loans.ss.test$bad.loan), positive = '1')
sensitivity <- round(sensitivity(factor(loans.ss.test$predicted), loans.ss.test$bad.loan,
    positive = "1"), 4)
specificity <- round(specificity(factor(loans.ss.test$predicted), loans.ss.test$bad.loan,
    negative = "0"), 4)
accuracy.logit.model2 <- round(((TP + TN)/(P + N)), 4)
cnfMtx <- confusionMatrix(loans.ss.test$predicted, loans.ss.test$bad.loan, positive = "1")

roc <- roc(factor(predicted) ~ as.numeric(bad.loan), data = loans.ss.test, plot = FALSE,
    ci = TRUE)
graphics::plot(roc, legacy.axes = TRUE, col = "blue", lwd = 3, main = "ROC - Logistic, Model 2")
auc.logit.model2 <- round(auc(factor(predicted) ~ as.numeric(bad.loan), loans.ss.test),
    4)

performance.results <- rbind(performance.results, c("Logistic, Model 2", accuracy.logit.model2,
    auc.logit.model2))
loans.ss.test$score <- predict(logit.fit.5, newdata = loans.ss.test, type = "response")
# ggplot(loans.ss.test, aes(y=bad.loan, x=score, color=bad.loan)) + geom_point()
# + geom_jitter()

cutoff = 0.18
loans.ss.test$predicted = as.numeric(loans.ss.test$score > cutoff)
TP = sum(loans.ss.test$predicted == 1 & loans.ss.test$bad.loan == 1)
```

```r
FP = sum(loans.ss.test$predicted == 1 & loans.ss.test$bad.loan == 0)
FN = sum(loans.ss.test$predicted == 0 & loans.ss.test$bad.loan == 1)
TN = sum(loans.ss.test$predicted == 0 & loans.ss.test$bad.loan == 0)

# lets also calculate total number of real positives and negatives in the data
P = TP + FN
N = TN + FP
total = P + N

# confusionMatrix(factor(loans.ss.test$predicted),
# factor(loans.ss.test$bad.loan), positive = '1')
sensitivity <- round(sensitivity(factor(loans.ss.test$predicted), loans.ss.test$bad.loan,
    positive = "1"), 4)
specificity <- round(specificity(factor(loans.ss.test$predicted), loans.ss.test$bad.loan,
    negative = "0"), 4)
accuracy.logit.model3 <- round(((TP + TN)/(P + N)), 4)
cnfMtx <- confusionMatrix(loans.ss.test$predicted, loans.ss.test$bad.loan, positive = "1")

roc <- roc(factor(predicted) ~ as.numeric(bad.loan), data = loans.ss.test, plot = FALSE,
    ci = TRUE)
graphics::plot(roc, legacy.axes = TRUE, col = "blue", lwd = 3, main = "ROC - Logistic, Model 3")
auc.logit.model3 <- round(auc(factor(predicted) ~ as.numeric(bad.loan), loans.ss.test),
    4)

performance.results <- rbind(performance.results, c("Logistic, Model 5", accuracy.logit.model3,
    auc.logit.model3))
loans.ss.test$prob <- predict(randomForest.fit.4, newdata = loans.ss.test, type = "prob")[,
    2]
roc_pred <- prediction(loans.ss.test$prob, loans.ss.test$bad.loan)

# http://stackoverflow.com/questions/20587978/calculate-accuracy-using-rocr-package-in-r
perf <- performance(roc_pred, "acc")
acc.rf.selectiveVar <- round(perf@y.values[[1]][max(which(perf@x.values[[1]] >= 0.5))],
    4)


auc.rf.selectiveVar <- round(as.numeric(performance(roc_pred, "auc")@y.values), 4)

# http://stackoverflow.com/questions/32292905/how-to-plot-a-roc-curve-from-classification-tree-probabilitie
perf <- performance(roc_pred, "tpr", "fpr")
plot(perf, col = "blue", lwd = 3, main = "ROC-Random Forest, Model 4")
abline(0, 1, col = "gray")

performance.results <- rbind(performance.results, c("Random Forest, Model 4", acc.rf.selectiveVar,
    auc.rf.selectiveVar))
loans.ss.test$prob <- predict(randomForest.fit.3, newdata = loans.ss.test, type = "prob")[,
    2]
roc_pred <- prediction(loans.ss.test$prob, loans.ss.test$bad.loan)

# http://stackoverflow.com/questions/20587978/calculate-accuracy-using-rocr-package-in-r
perf <- performance(roc_pred, "acc")
acc.rf.allVar <- round(perf@y.values[[1]][max(which(perf@x.values[[1]] >= 0.5))],
    4)

auc.rf.allVar <- round(as.numeric(performance(roc_pred, "auc")@y.values), 4)
```

```r
# http://stackoverflow.com/questions/32292905/how-to-plot-a-roc-curve-from-classification-tree-probabilitie
perf <- performance(roc_pred, "tpr", "fpr")
plot(perf, col = "blue", lwd = 3, main = "ROC-Random Forest, Model 3 ")
abline(0, 1, col = "gray")

performance.results <- rbind(performance.results, c("Random Forest, Model 3", acc.rf.allVar,
    auc.rf.allVar))
results <- as.data.frame(performance.results)
colnames(results) <- c("Model", "Accuracy", "AUC")
kable(results)
BibOptions(style = "html", bib.style = "authortitle")
bibliography()
```