Lending Club Project

Kevin Torres

**Task:** I will create a model to help predict people who showed a profile of having a high probability of paying investors back. I will try to classify and predict whether or not the borrower paid their loan back in full.

**Dataset:** I will be using publicly available lending data from 2007-2010. This data is from Lending Club which connects people who need money with investors.

Load the .csv file and save it as a data frame called loans

loans <- read.csv('loan\_data.csv')  
loans <- as.data.frame(loans)  
head(loans)

## credit.policy purpose int.rate installment log.annual.inc dti  
## 1 1 debt\_consolidation 0.1189 829.10 11.35041 19.48  
## 2 1 credit\_card 0.1071 228.22 11.08214 14.29  
## 3 1 debt\_consolidation 0.1357 366.86 10.37349 11.63  
## 4 1 debt\_consolidation 0.1008 162.34 11.35041 8.10  
## 5 1 credit\_card 0.1426 102.92 11.29973 14.97  
## 6 1 credit\_card 0.0788 125.13 11.90497 16.98  
## fico days.with.cr.line revol.bal revol.util inq.last.6mths delinq.2yrs  
## 1 737 5639.958 28854 52.1 0 0  
## 2 707 2760.000 33623 76.7 0 0  
## 3 682 4710.000 3511 25.6 1 0  
## 4 712 2699.958 33667 73.2 1 0  
## 5 667 4066.000 4740 39.5 0 1  
## 6 727 6120.042 50807 51.0 0 0  
## pub.rec not.fully.paid  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0

str(loans)

## 'data.frame': 9578 obs. of 14 variables:  
## $ credit.policy : int 1 1 1 1 1 1 1 1 1 1 ...  
## $ purpose : Factor w/ 7 levels "all\_other","credit\_card",..: 3 2 3 3 2 2 3 1 5 3 ...  
## $ int.rate : num 0.119 0.107 0.136 0.101 0.143 ...  
## $ installment : num 829 228 367 162 103 ...  
## $ log.annual.inc : num 11.4 11.1 10.4 11.4 11.3 ...  
## $ dti : num 19.5 14.3 11.6 8.1 15 ...  
## $ fico : int 737 707 682 712 667 727 667 722 682 707 ...  
## $ days.with.cr.line: num 5640 2760 4710 2700 4066 ...  
## $ revol.bal : int 28854 33623 3511 33667 4740 50807 3839 24220 69909 5630 ...  
## $ revol.util : num 52.1 76.7 25.6 73.2 39.5 51 76.8 68.6 51.1 23 ...  
## $ inq.last.6mths : int 0 0 1 1 0 0 0 0 1 1 ...  
## $ delinq.2yrs : int 0 0 0 0 1 0 0 0 0 0 ...  
## $ pub.rec : int 0 0 0 0 0 0 1 0 0 0 ...  
## $ not.fully.paid : int 0 0 0 0 0 0 1 1 0 0 ...

summary(loans)

## credit.policy purpose int.rate installment   
## Min. :0.000 all\_other :2331 Min. :0.0600 Min. : 15.67   
## 1st Qu.:1.000 credit\_card :1262 1st Qu.:0.1039 1st Qu.:163.77   
## Median :1.000 debt\_consolidation:3957 Median :0.1221 Median :268.95   
## Mean :0.805 educational : 343 Mean :0.1226 Mean :319.09   
## 3rd Qu.:1.000 home\_improvement : 629 3rd Qu.:0.1407 3rd Qu.:432.76   
## Max. :1.000 major\_purchase : 437 Max. :0.2164 Max. :940.14   
## small\_business : 619   
## log.annual.inc dti fico days.with.cr.line  
## Min. : 7.548 Min. : 0.000 Min. :612.0 Min. : 179   
## 1st Qu.:10.558 1st Qu.: 7.213 1st Qu.:682.0 1st Qu.: 2820   
## Median :10.929 Median :12.665 Median :707.0 Median : 4140   
## Mean :10.932 Mean :12.607 Mean :710.8 Mean : 4561   
## 3rd Qu.:11.291 3rd Qu.:17.950 3rd Qu.:737.0 3rd Qu.: 5730   
## Max. :14.528 Max. :29.960 Max. :827.0 Max. :17640   
##   
## revol.bal revol.util inq.last.6mths delinq.2yrs   
## Min. : 0 Min. : 0.0 Min. : 0.000 Min. : 0.0000   
## 1st Qu.: 3187 1st Qu.: 22.6 1st Qu.: 0.000 1st Qu.: 0.0000   
## Median : 8596 Median : 46.3 Median : 1.000 Median : 0.0000   
## Mean : 16914 Mean : 46.8 Mean : 1.577 Mean : 0.1637   
## 3rd Qu.: 18250 3rd Qu.: 70.9 3rd Qu.: 2.000 3rd Qu.: 0.0000   
## Max. :1207359 Max. :119.0 Max. :33.000 Max. :13.0000   
##   
## pub.rec not.fully.paid   
## Min. :0.00000 Min. :0.0000   
## 1st Qu.:0.00000 1st Qu.:0.0000   
## Median :0.00000 Median :0.0000   
## Mean :0.06212 Mean :0.1601   
## 3rd Qu.:0.00000 3rd Qu.:0.0000   
## Max. :5.00000 Max. :1.0000   
##

Convert certain columns to categorical data (factor)

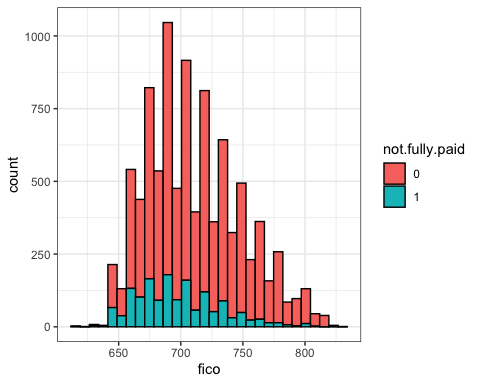
loans$inq.last.6mths <- factor(loans$inq.last.6mths)  
loans$delinq.2yrs <- factor(loans$delinq.2yrs)  
loans$pub.rec <- factor(loans$pub.rec)  
loans$not.fully.paid <- factor(loans$not.fully.paid)  
loans$credit.policy <- factor(loans$credit.policy)  
  
str(loans)

## 'data.frame': 9578 obs. of 14 variables:  
## $ credit.policy : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...  
## $ purpose : Factor w/ 7 levels "all\_other","credit\_card",..: 3 2 3 3 2 2 3 1 5 3 ...  
## $ int.rate : num 0.119 0.107 0.136 0.101 0.143 ...  
## $ installment : num 829 228 367 162 103 ...  
## $ log.annual.inc : num 11.4 11.1 10.4 11.4 11.3 ...  
## $ dti : num 19.5 14.3 11.6 8.1 15 ...  
## $ fico : int 737 707 682 712 667 727 667 722 682 707 ...  
## $ days.with.cr.line: num 5640 2760 4710 2700 4066 ...  
## $ revol.bal : int 28854 33623 3511 33667 4740 50807 3839 24220 69909 5630 ...  
## $ revol.util : num 52.1 76.7 25.6 73.2 39.5 51 76.8 68.6 51.1 23 ...  
## $ inq.last.6mths : Factor w/ 28 levels "0","1","2","3",..: 1 1 2 2 1 1 1 1 2 2 ...  
## $ delinq.2yrs : Factor w/ 11 levels "0","1","2","3",..: 1 1 1 1 2 1 1 1 1 1 ...  
## $ pub.rec : Factor w/ 6 levels "0","1","2","3",..: 1 1 1 1 1 1 2 1 1 1 ...  
## $ not.fully.paid : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 2 1 1 ...

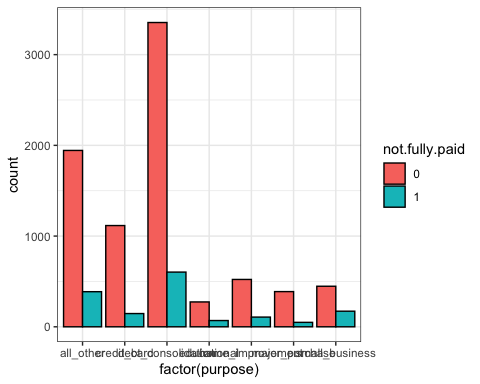
Exploratory Data Analysis

library(ggplot2)  
  
# Histogram of fico scores colored by whether the loan was fully paid or not  
ggplot(loans, aes(fico, fill = not.fully.paid)) + geom\_histogram(color = 'black') + theme\_bw()

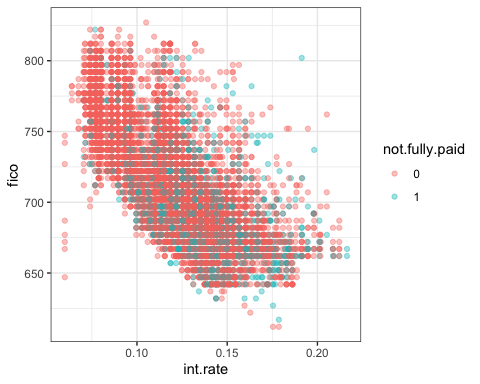
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



# Barplot of the purpose of loan counts, colored by whether the loan was fully paid or not  
ggplot(loans, aes(x = factor(purpose), fill = not.fully.paid)) + geom\_bar(color ='black', position = position\_dodge()) + theme\_bw()



# Scatterplot showing fico score versus interest rate   
ggplot(loans, aes(int.rate,fico, color = not.fully.paid)) + geom\_point(alpha = .4) + theme\_bw()



Split data into train and test sets

library(caTools)  
  
sample <- sample.split(loans$not.fully.paid, SplitRatio = .7)  
train <- subset(loans, sample == T)  
test <- subset(loans, sample == F)

Build and train a model using the svm function

library(e1071)  
  
model <- svm(not.fully.paid ~ ., train)  
summary(model)

##   
## Call:  
## svm(formula = not.fully.paid ~ ., data = train)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 2867  
##   
## ( 1794 1073 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

Only pass in test data that does not have a label

predicted.values <- predict(model, test[1:13])  
table(predicted.values, test$not.fully.paid)

##   
## predicted.values 0 1  
## 0 2413 460  
## 1 0 0

# These are bad results, model is predicting that everyone paid off their loan  
# We need to use the tune function to try out different cost and gamma values

Use the tune function to try out different cost and gamma values

tuned.results <- tune(svm, train.x = not.fully.paid ~ ., data = train, kernel = 'radial', ranges = list(cost = c(100,125,150,200), gamma= c(.075,.1)))  
summary(tuned.results)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 100 0.075  
##   
## - best performance: 0.1955248   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 100 0.075 0.1955248 0.01552347  
## 2 125 0.075 0.2002967 0.01855394  
## 3 150 0.075 0.2005955 0.01976197  
## 4 200 0.075 0.2073052 0.01749579  
## 5 100 0.100 0.2079020 0.01733323  
## 6 125 0.100 0.2116307 0.01789986  
## 7 150 0.100 0.2140192 0.01740708  
## 8 200 0.100 0.2204297 0.01703563

# We find our best parameters are 100 and .075  
tuned.model <- svm(not.fully.paid ~ ., data = train, cost = 100, gamma = .075)

Make predictions based off this tuned model

tuned.predictions <- predict(tuned.model, test[1:13])  
table(tuned.predictions, test$not.fully.paid)

##   
## tuned.predictions 0 1  
## 0 2230 382  
## 1 183 78

# Now we are no longer classifying everyone as paying off their loan

This model still is not too great but we could add more values to the ranges for cost and gamma but it would take time