Homework 3

Jiaming Chen

1/2/2018

## loading data

getwd()

## [1] "/Users/jessie/Desktop/Bittiger/Month1"

setwd('/Users/jessie/Desktop/Bittiger/Month1')  
list.files()

## [1] "HW1.pdf" "HW1.Rmd" "HW2.pdf" "HW2.Rmd"   
## [5] "HW3.Rmd" "HW4\_script.R" "Loan.Rmd" "note12\_12.Rmd"

loan <- read.csv('../Data/loan.csv', stringsAsFactors = FALSE)

## Clean Data

First think about what features could be included in the model > i.e., what features would be available during model building. Work example. > e.g., loan payment features will not be available when deciding interest rate.

Second think about what features should be included in the model > i.e., Remove features using intuition, Remove features with unique value per row or no variance. > Remove redundant features > e.g., id, member.id

num.value <- sapply(loan, function(x){return(length(unique(x)))})  
which(num.value == 1)

## policy\_code   
## 52

which(num.value == dim(loan)[1])

## id member\_id url   
## 1 2 19

# Feature Engineering

loan <- loan[,-which(colnames(loan) %in% c('id', 'member\_id', 'url', 'policy\_code'))]  
loan$dti <- ifelse(!is.na(loan$dti\_joint), loan$dti\_joint, loan$dti)  
loan$annual\_inc <- ifelse(!is.na(loan$annual\_inc\_joint), loan$annual\_inc\_joint, loan$annual\_inc)  
loan$home\_ownership <- ifelse(loan$home\_ownership %in% c('ANY', 'NONE', 'OTHER'), 'OTHER',  
 loan$home\_ownership)  
int\_state <- by(loan, loan$addr\_state, function(x) {  
 return(mean(x$int\_rate))  
})  
loan$state\_mean\_int <-  
 ifelse(loan$addr\_state %in% names(int\_state)[which(int\_state <= quantile(int\_state, 0.25))],   
 'low', ifelse(loan$addr\_state %in% names(int\_state)[which(int\_state <= quantile(int\_state, 0.5))],  
 'lowmedium', ifelse(loan$addr\_state %in% names(int\_state)[which(int\_state <= quantile(int\_state, 0.75))],   
 'mediumhigh', 'high')))  
num.NA <- sort(sapply(loan, function(x) { sum(is.na(x))} ), decreasing = TRUE)  
remain.col <- names(num.NA)[which(num.NA <= 0.8 \* dim(loan)[1])]  
loan <- loan[, remain.col]

inputation

num.NA <- sort(sapply(loan, function(x) { sum(is.na(x))} ), decreasing = TRUE)  
for(col.i in names(num.NA)[which(num.NA > 0)]) {  
 loan[which(is.na(loan[,col.i])), col.i] <- median(loan[,col.i], na.rm = TRUE)  
}

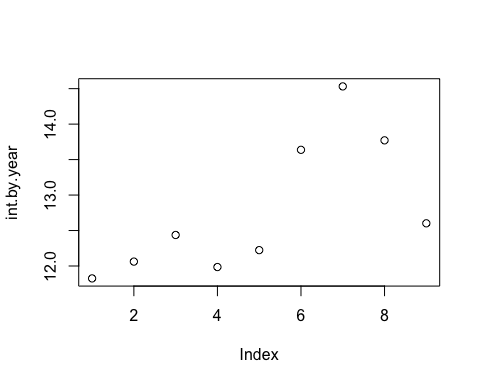
generate new features

library(zoo)

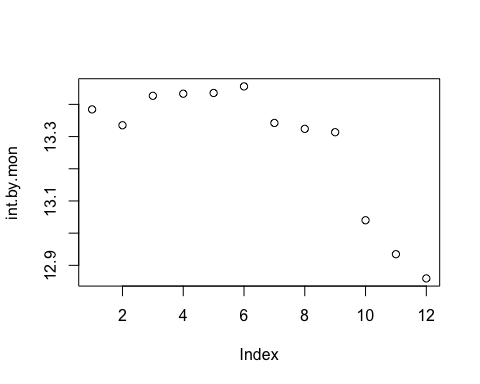
##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

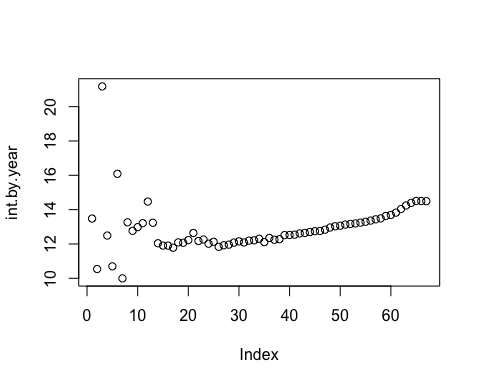
loan$issue\_d\_1 <- as.Date(as.yearmon(loan$issue\_d, "%b-%Y"))  
loan$issue\_year <- as.numeric(format(loan$issue\_d\_1, "%Y"))  
loan$issue\_mon <- as.numeric(format(loan$issue\_d\_1, "%m"))  
int.by.year <- by(loan, loan$issue\_year, function(x){return(mean(x$int\_rate))})  
plot(int.by.year)



int.by.mon <- by(loan, loan$issue\_mon, function(x){return(mean(x$int\_rate))})  
plot(int.by.mon)



loan$earliest\_cr\_line\_date <- as.Date(as.yearmon(loan$earliest\_cr\_line, "%b-%Y"))  
loan$earliest\_cr\_line\_year <- as.numeric(format(loan$earliest\_cr\_line\_date, "%Y"))  
int.by.year <- by(loan, loan$earliest\_cr\_line\_year, function(x){return(mean(x$int\_rate))})  
plot(int.by.year)



# earliest\_cr\_line\_year looks like a good predictor

## Splitting Training and Test set

# split data into train and test for model performance  
set.seed(1)  
train.ind <- sample(1:dim(loan)[1], 0.7 \* dim(loan)[1])  
train <- loan[train.ind, ]  
test <- loan[-train.ind, ]

# feature selection

categ\_names <- c('purpose', 'home\_ownership', 'state\_mean\_int', 'term', 'verification\_status')  
numer\_names <- c('int\_rate', 'annual\_inc', 'dti', 'loan\_amnt', 'total\_acc', 'tot\_cur\_bal', 'open\_acc',  
 'issue\_year', 'earliest\_cr\_line\_year')  
train.sub <- train[, c(numer\_names,categ\_names)]  
test.sub <- test[, c(numer\_names,categ\_names)]

# Standardize

train.sub.scale <- train.sub  
train.sub.scale[, c(2:9)] <- scale(train.sub.scale[, c(2:9)])  
  
test.sub.scale <- test.sub  
test.sub.scale[, c(2:9)] <- scale(test.sub.scale[, c(2:9)])

# Simple Linear Model

results <- NULL  
mod2 <- lm(int\_rate ~ ., data = train.sub.scale)  
summary(mod2)

##   
## Call:  
## lm(formula = int\_rate ~ ., data = train.sub.scale)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -15.3358 -2.6407 -0.2135 2.3299 21.3495   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 10.317886 0.048396 213.199 <2e-16  
## annual\_inc -0.127585 0.005319 -23.987 <2e-16  
## dti 0.593933 0.005081 116.902 <2e-16  
## loan\_amnt 0.106658 0.005788 18.427 <2e-16  
## total\_acc -0.216208 0.006748 -32.042 <2e-16  
## tot\_cur\_bal -0.285198 0.005870 -48.582 <2e-16  
## open\_acc -0.008480 0.006608 -1.283 0.1994  
## issue\_year -0.438486 0.004839 -90.610 <2e-16  
## earliest\_cr\_line\_year 0.467602 0.004911 95.212 <2e-16  
## purposecredit\_card -0.471251 0.047089 -10.008 <2e-16  
## purposedebt\_consolidation 0.975087 0.046497 20.971 <2e-16  
## purposeeducational -0.364965 0.214637 -1.700 0.0891  
## purposehome\_improvement 1.067375 0.049861 21.407 <2e-16  
## purposehouse 3.400715 0.084533 40.229 <2e-16  
## purposemajor\_purchase 0.563053 0.056423 9.979 <2e-16  
## purposemedical 2.633215 0.065579 40.153 <2e-16  
## purposemoving 3.642312 0.074599 48.825 <2e-16  
## purposeother 2.861996 0.050435 56.746 <2e-16  
## purposerenewable\_energy 3.089915 0.188224 16.416 <2e-16  
## purposesmall\_business 3.501489 0.062594 55.940 <2e-16  
## purposevacation 2.657462 0.077819 34.149 <2e-16  
## purposewedding 1.445427 0.101439 14.249 <2e-16  
## home\_ownershipOTHER 0.396981 0.289819 1.370 0.1708  
## home\_ownershipOWN 0.272713 0.016432 16.597 <2e-16  
## home\_ownershipRENT 0.409984 0.011507 35.628 <2e-16  
## state\_mean\_intlow -0.380246 0.020159 -18.862 <2e-16  
## state\_mean\_intlowmedium -0.192930 0.014381 -13.416 <2e-16  
## state\_mean\_intmediumhigh -0.131830 0.015626 -8.437 <2e-16  
## term 60 months 3.917348 0.011169 350.730 <2e-16  
## verification\_statusSource Verified 0.812503 0.011677 69.582 <2e-16  
## verification\_statusVerified 1.910823 0.012196 156.673 <2e-16  
##   
## (Intercept) \*\*\*  
## annual\_inc \*\*\*  
## dti \*\*\*  
## loan\_amnt \*\*\*  
## total\_acc \*\*\*  
## tot\_cur\_bal \*\*\*  
## open\_acc   
## issue\_year \*\*\*  
## earliest\_cr\_line\_year \*\*\*  
## purposecredit\_card \*\*\*  
## purposedebt\_consolidation \*\*\*  
## purposeeducational .   
## purposehome\_improvement \*\*\*  
## purposehouse \*\*\*  
## purposemajor\_purchase \*\*\*  
## purposemedical \*\*\*  
## purposemoving \*\*\*  
## purposeother \*\*\*  
## purposerenewable\_energy \*\*\*  
## purposesmall\_business \*\*\*  
## purposevacation \*\*\*  
## purposewedding \*\*\*  
## home\_ownershipOTHER   
## home\_ownershipOWN \*\*\*  
## home\_ownershipRENT \*\*\*  
## state\_mean\_intlow \*\*\*  
## state\_mean\_intlowmedium \*\*\*  
## state\_mean\_intmediumhigh \*\*\*  
## term 60 months \*\*\*  
## verification\_statusSource Verified \*\*\*  
## verification\_statusVerified \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.614 on 621116 degrees of freedom  
## (18 observations deleted due to missingness)  
## Multiple R-squared: 0.3201, Adjusted R-squared: 0.32   
## F-statistic: 9747 on 30 and 621116 DF, p-value: < 2.2e-16

yhat\_lm <- predict(mod2, newdata = test.sub.scale)  
# results <- rbind(results, data.frame(RMSE=mean( (yhat\_lm - test.sub.scale$int\_rate)^2, na.rm = T ),  
# type='linear'))  
# which(is.na(yhat\_lm)) --- I don't understand why there's NA in the predicted values

## hypothesis test

train.sub.matrix <- model.matrix( ~., train.sub.scale)  
#head(train.sub.matrix)  
x <- train.sub.matrix[, -2]  
y <- train.sub.matrix[, 2]

## F test

works the same as ANOVA, test if the model works or not

sst = sum((y - mean(y))^2) # sum of square total, df = n - 1 = 572060  
ssr = sum(mod2$res^2) # sum of square residual, df = n-1-p = 572047  
ssm = sum((y - mean(y))^2) - sum(mod2$res^2) # sum of square model, df = 13  
Fstats = (ssm)/(13) / (ssr / (dim(train.sub)[1] - 13 -1))  
1 - pf(Fstats, 13, (dim(train.sub)[1] - 13 - 1)) # def = p and n-1-p

## [1] 0

## testing residual normality

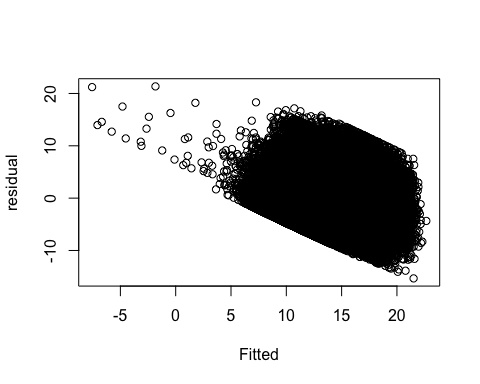
# residual = observed - fitted  
# head(sort(mod2$res))  
mod2$res[which.min(mod2$res)]

## 24093   
## -15.33584

mod2$res[which.max(mod2$res)]

## 325186   
## 21.3495

plot(mod2$fit, mod2$res, xlab = 'Fitted', ylab = 'residual')



# plot(mod2)

## adjustment

It’s impossible for the interest rate to be a negative value, so we need to adjust the model.

mod2\_1 <- lm(log(int\_rate) ~. ,data = train.sub.scale)  
summary(mod2\_1)

##   
## Call:  
## lm(formula = log(int\_rate) ~ ., data = train.sub.scale)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.3461 -0.1938 0.0225 0.2041 1.8317   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 2.2844343 0.0038414 594.680 < 2e-16  
## annual\_inc -0.0108399 0.0004222 -25.675 < 2e-16  
## dti 0.0505736 0.0004033 125.407 < 2e-16  
## loan\_amnt 0.0036759 0.0004594 8.001 1.24e-15  
## total\_acc -0.0173276 0.0005356 -32.351 < 2e-16  
## tot\_cur\_bal -0.0229789 0.0004660 -49.314 < 2e-16  
## open\_acc -0.0030647 0.0005246 -5.842 5.15e-09  
## issue\_year -0.0370430 0.0003841 -96.435 < 2e-16  
## earliest\_cr\_line\_year 0.0376846 0.0003898 96.669 < 2e-16  
## purposecredit\_card -0.0274032 0.0037378 -7.331 2.28e-13  
## purposedebt\_consolidation 0.0903330 0.0036908 24.475 < 2e-16  
## purposeeducational -0.0005089 0.0170371 -0.030 0.9762  
## purposehome\_improvement 0.0916043 0.0039578 23.145 < 2e-16  
## purposehouse 0.2486863 0.0067099 37.062 < 2e-16  
## purposemajor\_purchase 0.0457570 0.0044786 10.217 < 2e-16  
## purposemedical 0.2120578 0.0052054 40.738 < 2e-16  
## purposemoving 0.2817811 0.0059213 47.587 < 2e-16  
## purposeother 0.2259560 0.0040033 56.442 < 2e-16  
## purposerenewable\_energy 0.2361995 0.0149404 15.809 < 2e-16  
## purposesmall\_business 0.2625794 0.0049685 52.849 < 2e-16  
## purposevacation 0.2172501 0.0061770 35.171 < 2e-16  
## purposewedding 0.1080417 0.0080518 13.418 < 2e-16  
## home\_ownershipOTHER 0.0439434 0.0230047 1.910 0.0561  
## home\_ownershipOWN 0.0204807 0.0013043 15.703 < 2e-16  
## home\_ownershipRENT 0.0337669 0.0009134 36.968 < 2e-16  
## state\_mean\_intlow -0.0321359 0.0016001 -20.083 < 2e-16  
## state\_mean\_intlowmedium -0.0153511 0.0011415 -13.448 < 2e-16  
## state\_mean\_intmediumhigh -0.0101266 0.0012403 -8.165 3.23e-16  
## term 60 months 0.3002345 0.0008866 338.651 < 2e-16  
## verification\_statusSource Verified 0.0697458 0.0009269 75.249 < 2e-16  
## verification\_statusVerified 0.1529407 0.0009681 157.982 < 2e-16  
##   
## (Intercept) \*\*\*  
## annual\_inc \*\*\*  
## dti \*\*\*  
## loan\_amnt \*\*\*  
## total\_acc \*\*\*  
## tot\_cur\_bal \*\*\*  
## open\_acc \*\*\*  
## issue\_year \*\*\*  
## earliest\_cr\_line\_year \*\*\*  
## purposecredit\_card \*\*\*  
## purposedebt\_consolidation \*\*\*  
## purposeeducational   
## purposehome\_improvement \*\*\*  
## purposehouse \*\*\*  
## purposemajor\_purchase \*\*\*  
## purposemedical \*\*\*  
## purposemoving \*\*\*  
## purposeother \*\*\*  
## purposerenewable\_energy \*\*\*  
## purposesmall\_business \*\*\*  
## purposevacation \*\*\*  
## purposewedding \*\*\*  
## home\_ownershipOTHER .   
## home\_ownershipOWN \*\*\*  
## home\_ownershipRENT \*\*\*  
## state\_mean\_intlow \*\*\*  
## state\_mean\_intlowmedium \*\*\*  
## state\_mean\_intmediumhigh \*\*\*  
## term 60 months \*\*\*  
## verification\_statusSource Verified \*\*\*  
## verification\_statusVerified \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2869 on 621116 degrees of freedom  
## (18 observations deleted due to missingness)  
## Multiple R-squared: 0.3123, Adjusted R-squared: 0.3122   
## F-statistic: 9400 on 30 and 621116 DF, p-value: < 2.2e-16

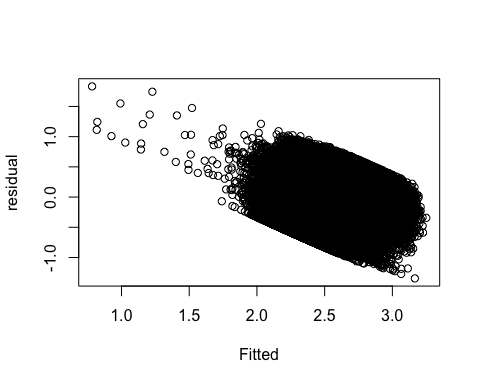
mod2\_1$res[which.min(mod2\_1$res)]

## 24093   
## -1.346142

mod2\_1$res[which.max(mod2\_1$res)]

## 866974   
## 1.831657

plot(mod2\_1$fit, mod2\_1$res, xlab = 'Fitted', ylab = 'residual')



# still large residuals for some data points. Check the reason.  
cbind(train.sub[which(mod2\_1$fitted <= 1.5), ],  
 pred = round(exp(predict(mod2\_1, train.sub[which(mod2\_1$fitted <= 1.5), ])), 2))

## int\_rate annual\_inc dti loan\_amnt total\_acc tot\_cur\_bal open\_acc  
## 506741 6.89 9000000 0.08 11000 29 324692 15  
## 552198 12.69 8253000 0.14 30000 26 130006 18  
## 15920 7.49 32400 3.41 5000 7 80559 2  
## 672226 18.25 65000 21.94 20000 38 104507 6  
## 329123 16.29 41000 21.02 16000 26 78255 9  
## 532769 8.18 111000 16.70 20350 48 426134 20  
## 125136 7.90 36000 7.67 8000 15 76124 9  
## 857304 15.99 55000 26.99 18000 22 148589 12  
## 272152 13.35 60000 3.72 12000 17 4828 10  
## 259349 13.66 95000 27.92 26000 30 98131 9  
## 209621 8.90 80000 18.84 9000 46 80559 16  
## 378458 12.99 73000 6.10 14675 16 46308 7  
## 289093 16.99 63000 24.42 12000 36 40519 17  
## 229997 7.90 32000 9.00 7200 14 80559 11  
## 296173 16.29 62000 9.95 18000 10 151430 7  
## 120165 6.62 47375 11.37 9500 20 175896 8  
## 579104 12.69 50000 20.88 8000 7 72891 6  
## issue\_year earliest\_cr\_line\_year purpose home\_ownership  
## 506741 2015 2004 debt\_consolidation MORTGAGE  
## 552198 2015 2002 debt\_consolidation OWN  
## 15920 2011 1999 car MORTGAGE  
## 672226 2015 2004 debt\_consolidation MORTGAGE  
## 329123 2014 1995 credit\_card MORTGAGE  
## 532769 2015 1990 debt\_consolidation RENT  
## 125136 2013 2002 credit\_card MORTGAGE  
## 857304 2015 1998 debt\_consolidation MORTGAGE  
## 272152 2014 2004 other RENT  
## 259349 2014 2001 debt\_consolidation RENT  
## 209621 2012 1995 debt\_consolidation RENT  
## 378458 2014 1971 debt\_consolidation MORTGAGE  
## 289093 2014 1991 debt\_consolidation RENT  
## 229997 2012 2004 credit\_card RENT  
## 296173 2014 2005 credit\_card MORTGAGE  
## 120165 2013 1999 debt\_consolidation MORTGAGE  
## 579104 2015 2008 other MORTGAGE  
## state\_mean\_int term verification\_status pred  
## 506741 lowmedium 36 months Source Verified 0  
## 552198 lowmedium 36 months Source Verified 0  
## 15920 mediumhigh 36 months Not Verified 0  
## 672226 mediumhigh 60 months Source Verified 0  
## 329123 high 60 months Verified 0  
## 532769 low 36 months Source Verified 0  
## 125136 lowmedium 36 months Not Verified 0  
## 857304 lowmedium 60 months Not Verified 0  
## 272152 mediumhigh 36 months Verified 0  
## 259349 lowmedium 60 months Source Verified 0  
## 209621 lowmedium 36 months Not Verified 0  
## 378458 lowmedium 36 months Source Verified 0  
## 289093 lowmedium 60 months Verified 0  
## 229997 lowmedium 36 months Not Verified 0  
## 296173 high 60 months Verified 0  
## 120165 lowmedium 36 months Verified 0  
## 579104 high 36 months Source Verified 0

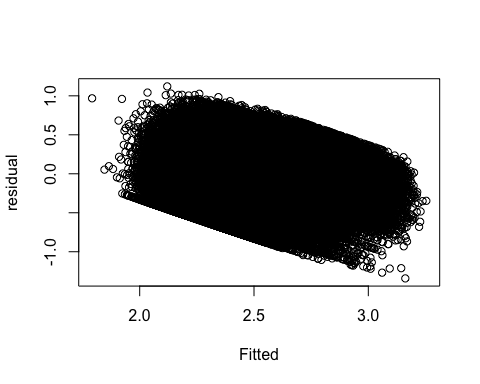
train.sub.scale$tanh\_annual\_inc <- tanh(train.sub.scale$annual\_inc)  
mod2\_2 <- lm(log(int\_rate) ~. ,data = train.sub.scale[,-which(colnames(train.sub.scale) == 'annual\_inc')])  
mod2\_2$res[which.min(mod2\_1$res)]

## 24093   
## -1.342117

mod2\_2$res[which.max(mod2\_1$res)]

## 866974   
## 0.5046654

plot(mod2\_2$fit, mod2\_2$res, xlab = 'Fitted', ylab = 'residual')

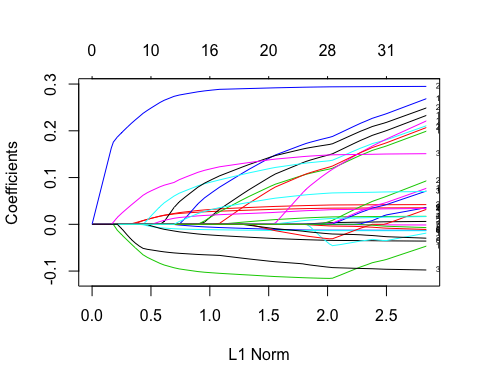


# much better  
  
# test on test set  
test.sub.scale$tanh\_annual\_inc <- tanh(test.sub.scale$annual\_inc)  
yhat\_lm\_1 <- predict(mod2\_2, newdata = test.sub.scale[,-which(colnames(test.sub.scale) == 'annual\_inc')])  
results <- rbind(results, data.frame(RMSE=mean( (yhat\_lm\_1 - log(test.sub.scale$int\_rate))^2, na.rm = T ),  
 type='linear\_1'))  
  
# adjust the RMSE for linear model, otherwise the results are not comparable  
results <- rbind(results, data.frame(RMSE=mean( (log(yhat\_lm) - log(test.sub.scale$int\_rate))^2, na.rm = T ),  
 type='linear'))

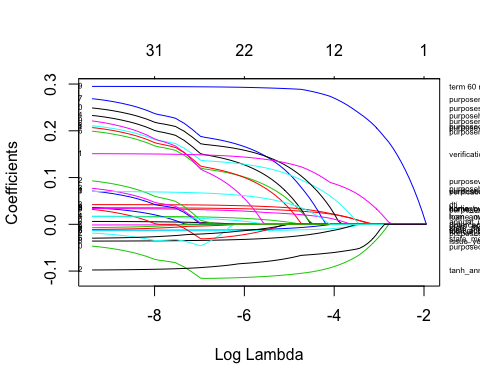
## Warning in log(yhat\_lm): NaNs produced

# Lasso

temp <- model.matrix( ~., train.sub.scale)  
x <- temp[, -2]  
y <- temp[, 2]  
  
temp <- model.matrix( ~., test.sub.scale)  
x\_test <- temp[, -2]  
y\_test <- temp[, 2]  
  
mod\_lasso <- glmnet(x=x, y=log(y)) # default is alpha = 1, lasso  
plot(mod\_lasso, label = T)



plot(mod\_lasso, xvar = "lambda", label = T)  
  
vnat=coef(mod\_lasso)  
vnat=vnat[-1,ncol(vnat)] # remove the intercept, and get the coefficients at the end of the path  
axis(4, at=vnat,line=-.5,label=colnames(x),las=1,tick=FALSE, cex.axis=0.5)

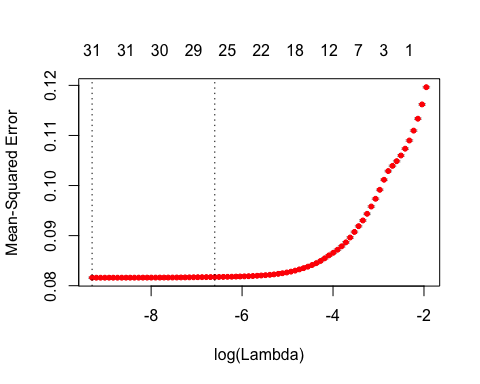


print(mod\_lasso)

##   
## Call: glmnet(x = x, y = log(y))   
##   
## Df %Dev Lambda  
## [1,] 0 0.00000 1.425e-01  
## [2,] 1 0.02880 1.298e-01  
## [3,] 1 0.05272 1.183e-01  
## [4,] 1 0.07257 1.078e-01  
## [5,] 1 0.08905 9.820e-02  
## [6,] 1 0.10270 8.948e-02  
## [7,] 1 0.11410 8.153e-02  
## [8,] 1 0.12350 7.428e-02  
## [9,] 1 0.13140 6.768e-02  
## [10,] 3 0.14000 6.167e-02  
## [11,] 3 0.15460 5.619e-02  
## [12,] 4 0.17130 5.120e-02  
## [13,] 4 0.18630 4.665e-02  
## [14,] 5 0.19920 4.251e-02  
## [15,] 6 0.21140 3.873e-02  
## [16,] 6 0.22250 3.529e-02  
## [17,] 7 0.23200 3.216e-02  
## [18,] 9 0.24160 2.930e-02  
## [19,] 10 0.25090 2.670e-02  
## [20,] 10 0.25900 2.432e-02  
## [21,] 10 0.26560 2.216e-02  
## [22,] 11 0.27140 2.019e-02  
## [23,] 11 0.27650 1.840e-02  
## [24,] 12 0.28070 1.677e-02  
## [25,] 13 0.28560 1.528e-02  
## [26,] 14 0.29010 1.392e-02  
## [27,] 14 0.29380 1.268e-02  
## [28,] 14 0.29690 1.156e-02  
## [29,] 16 0.29960 1.053e-02  
## [30,] 16 0.30210 9.594e-03  
## [31,] 18 0.30420 8.742e-03  
## [32,] 18 0.30620 7.965e-03  
## [33,] 18 0.30790 7.258e-03  
## [34,] 19 0.30930 6.613e-03  
## [35,] 19 0.31060 6.025e-03  
## [36,] 20 0.31160 5.490e-03  
## [37,] 20 0.31250 5.002e-03  
## [38,] 20 0.31320 4.558e-03  
## [39,] 20 0.31380 4.153e-03  
## [40,] 22 0.31440 3.784e-03  
## [41,] 22 0.31480 3.448e-03  
## [42,] 22 0.31520 3.142e-03  
## [43,] 22 0.31550 2.863e-03  
## [44,] 22 0.31580 2.608e-03  
## [45,] 22 0.31600 2.377e-03  
## [46,] 24 0.31620 2.165e-03  
## [47,] 24 0.31640 1.973e-03  
## [48,] 25 0.31660 1.798e-03  
## [49,] 26 0.31670 1.638e-03  
## [50,] 26 0.31680 1.493e-03  
## [51,] 27 0.31690 1.360e-03  
## [52,] 27 0.31700 1.239e-03  
## [53,] 28 0.31710 1.129e-03  
## [54,] 28 0.31720 1.029e-03  
## [55,] 29 0.31720 9.374e-04  
## [56,] 29 0.31740 8.541e-04  
## [57,] 29 0.31750 7.782e-04  
## [58,] 29 0.31760 7.091e-04  
## [59,] 31 0.31770 6.461e-04  
## [60,] 31 0.31770 5.887e-04  
## [61,] 31 0.31780 5.364e-04  
## [62,] 30 0.31780 4.887e-04  
## [63,] 30 0.31790 4.453e-04  
## [64,] 30 0.31790 4.058e-04  
## [65,] 30 0.31790 3.697e-04  
## [66,] 31 0.31790 3.369e-04  
## [67,] 31 0.31790 3.069e-04  
## [68,] 31 0.31800 2.797e-04  
## [69,] 31 0.31800 2.548e-04  
## [70,] 31 0.31800 2.322e-04  
## [71,] 31 0.31800 2.116e-04  
## [72,] 31 0.31800 1.928e-04  
## [73,] 31 0.31810 1.756e-04  
## [74,] 31 0.31810 1.600e-04  
## [75,] 31 0.31810 1.458e-04  
## [76,] 31 0.31810 1.329e-04  
## [77,] 31 0.31810 1.211e-04  
## [78,] 31 0.31810 1.103e-04  
## [79,] 31 0.31810 1.005e-04  
## [80,] 31 0.31810 9.158e-05  
## [81,] 31 0.31810 8.344e-05

yhat\_lasso <- predict(mod\_lasso, newx = x\_test)  
results <- rbind(results, data.frame(RMSE=mean( (yhat\_lasso - log(y\_test))^2, na.rm = T ),  
 type='lasso'))

# use cross validation to get optimal value of lambda,   
cvmod\_lasso <- cv.glmnet(x, log(y))  
plot(cvmod\_lasso)



# Two selected lambdas are shown,   
cvmod\_lasso$lambda.min # value of lambda gives minimal mean cross validated error

## [1] 9.158051e-05

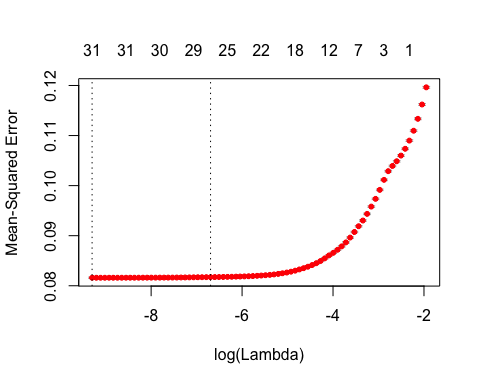
cvmod\_lasso$lambda.1se # most regularized model such that error is within one std err of the minimum

## [1] 0.001359941

yhat\_1se <- predict(cvmod\_lasso,newx=x\_test,s='lambda.1se')  
results <- rbind(results, data.frame(RMSE=mean( (yhat\_1se - log(y\_test))^2, na.rm = T ),  
 type='lasso\_1se'))

# Ridge Regression

cvmod\_ridge <- cv.glmnet(x, log(y))  
plot(cvmod\_ridge)



yhat\_ridge <- predict(cvmod\_ridge,newx=x\_test,s='lambda.min')  
results <- rbind(results, data.frame(RMSE=mean( (yhat\_ridge - log(y\_test))^2, na.rm = T ),  
 type='ridge\_min'))  
  
yhat\_ridge\_1se <- predict(cvmod\_ridge,newx=x\_test,s='lambda.1se')  
results <- rbind(results, data.frame(RMSE=mean( (yhat\_ridge\_1se - log(y\_test))^2, na.rm = T ),  
 type='ridge\_1se'))

# analyze the result

results

## RMSE type  
## 1 0.08166953 linear\_1  
## 2 0.08380158 linear  
## 3 0.08721365 lasso  
## 4 0.08176998 lasso\_1se  
## 5 0.08164000 ridge\_min  
## 6 0.08176004 ridge\_1se

We can see from the RMSE from test set that ridge regression with 1SE lambda performs the best for this training test set split. Usually lasso regression would performs better than ridge regression, we can try another training test split and compare which model gives the most stable performance.

From my point of view, lasso regression with one standard error lambda would be the best model, since it uses the smallest subset of available feature and explains the variance well.