LoanRiskAnalysis_InterestRate

Liang Tan

Read data

```
loan <- read.csv("loan.csv", stringsAsFactors = FALSE)
loanT <- loan
num.NA <- sort(sapply(loan, function(x) sum(is.na(x))), decreasing = TRUE)
remain.col = names(num.NA)[(num.NA < 0.8 * dim(loan)[1])]
delete.col = names(num.NA)[(num.NA >= 0.8 * dim(loan)[1])]
delete.col
```

```
##  [1] "dti_joint"                 "annual_inc_joint"
##  [3] "il_util"                   "mths_since_rcnt_il"
##  [5] "open_acc_6m"               "open_il_6m"
##  [7] "open_il_12m"               "open_il_24m"
##  [9] "total_bal_il"              "open_rv_12m"
## [11] "open_rv_24m"               "max_bal_bc"
## [13] "all_util"                  "inq_fi"
## [15] "total_cu_tl"               "inq_last_12m"
## [17] "mths_since_last_record"
```

Feature engineering and selection

User feature selection
addr_state, emp_title, member_id, zip_code is removed
emp_length, home_ownership is reserved

```
# encode home_ownership
loan$home_ownership <- ifelse(loan$home_ownership %in% c("ANY", "NONE", "OTHER"),
    "OTHER", loan$home_ownership)
# encode state information with the help of int_rate
int_state <- by(loan, loan$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_sta
    quantile(int_state, 0.75))], "mediumhigh", "high")))
select.features_1 <- c("home_ownership", "state_mean_int")
```

Financial feature selection
combine annual_inc and annual_inc_joint, dti and dti_joint, verification_status and verification_status_joint based on joint condition

```
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$annual_inc[which(is.na(loan$annual_inc))] <- median(loan$annual_inc, na.rm = T)
```

```
loan$verification_status <- ifelse(loan$application_type == "JOINT", loan$verification_status_joint,
    loan$verification_status)
select.features_2 <- c("dti", "annual_inc", "verification_status")
```

Credit scores feature selection
inq_fi, inq_last_12m is removed for over 80% NA values.
The earliest_cr_line and last_credit_pull_d are reserved

```
select.features_3 <- c("earliest_cr_line", "last_credit_pull_d")
```

credit lines feature selection
all_util, open_acc_6m, total_cu_tl, open_il_6m, open_il_12m, open_il_24m, open_rv_12m, open_rv_24m, max_bal_bc, mths_since_last_record, il_util, mths_since_rcnt_il, total_bal_il, max_bal_bc are removed for over 80% NA values
policy_code and url are removed for irrelavance
total_acc, tot_cur_bal, open_acc, acc_now_delinq, delinq_2yrs, mths_since_last_delinq, collections_12_mths_ex_med, tot_coll_amt, pub_rec, mths_since_last_major_derog, revol_util, total_rev_hi_lim are reserved

```
# mean and median are similar so I use mean for na
loan$total_acc[which(is.na(loan$total_acc))] <- mean(loan$total_acc, na.rm = T)
# mean of tot_cur_bal is more influenced by large value so I use median
loan$tot_cur_bal[which(is.na(loan$tot_cur_bal))] <- median(loan$tot_cur_bal,
    na.rm = T)
# mean and median are similar so I use mean for na
loan$open_acc[which(is.na(loan$open_acc))] <- mean(loan$open_acc, na.rm = T)
# acc_now_delinq is int number, so I use median for na
loan$acc_now_delinq[which(is.na(loan$acc_now_delinq))] <- median(loan$acc_now_delinq,
    na.rm = T)
# delinq_2yrs is int number, so I use median for na
loan$delinq_2yrs[which(is.na(loan$delinq_2yrs))] <- median(loan$delinq_2yrs,
    na.rm = T)
# mths_since_last_delinq is int number, so I use median for na
loan$mths_since_last_delinq[which(is.na(loan$mths_since_last_delinq))] <- median(loan$mths_since_last_de
    na.rm = T)
# collections_12_mths_ex_med is int number, so I use median for na
loan$collections_12_mths_ex_med[which(is.na(loan$collections_12_mths_ex_med))] <- median(loan$collectio
    na.rm = T)
# tot_coll_amt is int number, so I use median for na
loan$tot_coll_amt[which(is.na(loan$tot_coll_amt))] <- median(loan$tot_coll_amt,
    na.rm = T)
# pub_rec is int number, so I use median for na
loan$pub_rec[which(is.na(loan$pub_rec))] <- median(loan$pub_rec, na.rm = T)
# mths_since_last_major_derog is int number, so I use median for na
loan$mths_since_last_major_derog[which(is.na(loan$mths_since_last_major_derog))] <- median(loan$mths_si
    na.rm = T)
# mean and median is similar so I use mean for revol_util na values
loan$revol_util[which(is.na(loan$revol_util))] <- mean(loan$revol_util, na.rm = T)
# total_rev_hi_lim is int number, so I use median for na
loan$total_rev_hi_lim[which(is.na(loan$total_rev_hi_lim))] <- median(loan$total_rev_hi_lim,
    na.rm = T)

select.features_4 <- c("total_acc", "tot_cur_bal", "open_acc", "acc_now_delinq",
    "delinq_2yrs", "mths_since_last_delinq", "collections_12_mths_ex_med", "tot_coll_amt",
    "pub_rec", "mths_since_last_major_derog", "revol_util", "total_rev_hi_lim")
```

loan feature selection
desc, id, title, issue_d, are removed
loan_amnt, application_type, purpose, term and initial_list_status are reserved

```
select.features_5 <- c("loan_amnt", "application_type", "purpose", "term", "initial_list_status")
```

loan payment feature selection
last_pymnt_amnt, last_pymnt_d, next_pymnt_d, total_pymnt, total_pymnt_inv, total_rec_int, total_rec_late_fee, total_rec_prncp are inrrelative here
installment, funded_amnt, funded_amnt_inv, pymnt_plan, recoveries collection_recovery_fee, out_prncp, out_prncp_inv are reserved

```
select.features_6 <- c("installment", "funded_amnt", "funded_amnt_inv", "pymnt_plan",
    "recoveries", "collection_recovery_fee", "out_prncp", "out_prncp_inv")
```

grade and int_rate are used as well

```
select.features <- c(select.features_1, select.features_2, select.features_3,
    select.features_4, select.features_5, select.features_6, "int_rate")
loan <- loan[select.features]
```

scale all numeric variables

```
select.features.num <- names(loan[, sapply(loan[, 1:32], is.numeric)])
loan.scale <- loan
loan.scale[, select.features.num] <- scale(loan.scale[, select.features.num])
```

check the level of all category variables

```
select.features.cate <- names(loan.scale[, sapply(loan.scale, is.character)])
n_levels <- sort(sapply(loan.scale[select.features.cate], function(x) {
    nlevels(as.factor(x))
}), decreasing = TRUE)
print(n_levels)
```

```
##    earliest_cr_line  last_credit_pull_d             purpose
##                 698                 104                  14
##      home_ownership      state_mean_int verification_status
##                   4                   4                   3
##    application_type                term initial_list_status
##                   2                   2                   2
##          pymnt_plan
##                   2
```

The level number of 'earliest_cr_line' and 'last_credit_pull_d' is too large. Further treatment needs applying.

```
anova_test <- aov(int_rate ~ earliest_cr_line, data = loan.scale)
summary(anova_test)
```

```
##                      Df   Sum Sq Mean Sq F value Pr(>F)
## earliest_cr_line    697   224605   322.2   16.99 <2e-16 ***
## Residuals        886681 16813728    19.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA test shows this feature is important so I can't delete it. Therefore, I will transfer it into years only.

```r
library("zoo")
```

```
##
## Attaching package: 'zoo'

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

```r
loan.scale$earliest_cr_line <- format(as.Date(as.yearmon(loan.scale$earliest_cr_line,
    "%B-%Y")), "%Y")
```

```
## Warning in strptime(x, format, tz = "GMT"): unknown timezone 'default/
## America/Los_Angeles'
```

```r
length(unique(loan.scale$earliest_cr_line))
```

```
## [1] 68
```

Now the levels of earliest_cr_line are reduced to 68.

```r
anova_test <- aov(int_rate ~ last_credit_pull_d, data = loan.scale)
summary(anova_test)
```

```
##                         Df   Sum Sq Mean Sq F value Pr(>F)
## last_credit_pull_d     103    82321   799.2   41.82 <2e-16 ***
## Residuals           887275 16956013    19.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ANOVA test shows this feature is important so I can't delete it. Therefore, I will transfer it into years only.

```r
loan.scale$last_credit_pull_d <- format(as.Date(as.yearmon(loan.scale$last_credit_pull_d,
    "%B-%Y")), "%Y")
length(unique(loan.scale$last_credit_pull_d))
```

```
## [1] 11
```

Now the levels of last_credit_pull_d are reduced to 11.

Build model to predict the loan interest_rate

train, test data set selection

```r
set.seed(1)
train.ind <- sample(1:dim(loan.scale)[1], 0.8 * dim(loan)[1])
train <- loan.scale[train.ind, ]
test <- loan.scale[-train.ind, ]
```

build regression model

```r
mod <- lm(int_rate ~ ., data = train)
print(summary(mod))
```

```
##
## Call:
## lm(formula = int_rate ~ ., data = train)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.138  -1.962  -0.279   1.762  74.384
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          7.308618   2.011227   3.634 0.000279
## home_ownershipOTHER                  1.057133   0.208537   5.069 3.99e-07
## home_ownershipOWN                    0.322133   0.011806  27.287  < 2e-16
## home_ownershipRENT                   0.302525   0.008297  36.464  < 2e-16
## state_mean_intlow                   -0.273814   0.014445 -18.955  < 2e-16
## state_mean_intlowmedium             -0.108981   0.010300 -10.581  < 2e-16
## state_mean_intmediumhigh            -0.101530   0.011190  -9.074  < 2e-16
## dti                                  0.255680   0.003768  67.862  < 2e-16
## annual_inc                          -0.132142   0.003890 -33.971  < 2e-16
## verification_statusSource Verified   0.230975   0.008399  27.500  < 2e-16
## verification_statusVerified          1.030125   0.008876 116.051  < 2e-16
## earliest_cr_line1946                 1.318547   2.767774   0.476 0.633795
## earliest_cr_line1949                 0.314696   3.389799   0.093 0.926034
## earliest_cr_line1950                 0.004764   2.259932   0.002 0.998318
## earliest_cr_line1951                 4.739900   2.526626   1.876 0.060658
## earliest_cr_line1952                 2.137829   2.526613   0.846 0.397484
## earliest_cr_line1953                -0.231490   2.396940  -0.097 0.923062
## earliest_cr_line1954                 2.078404   2.396997   0.867 0.385895
## earliest_cr_line1955                 1.984297   2.113931   0.939 0.347897
## earliest_cr_line1956                 1.492778   2.092228   0.713 0.475545
## earliest_cr_line1957                 3.061605   2.092219   1.463 0.143378
## earliest_cr_line1958                 1.795678   2.057519   0.873 0.382806
## earliest_cr_line1959                 1.254077   2.012247   0.623 0.533139
## earliest_cr_line1960                 1.783477   1.993008   0.895 0.370859
## earliest_cr_line1961                 0.894775   1.995108   0.448 0.653804
## earliest_cr_line1962                 0.985832   1.985260   0.497 0.619489
## earliest_cr_line1963                 1.232905   1.972816   0.625 0.532006
## earliest_cr_line1964                 1.397428   1.969846   0.709 0.478070
## earliest_cr_line1965                 1.769353   1.965587   0.900 0.368033
## earliest_cr_line1966                 1.514710   1.964580   0.771 0.440701
## earliest_cr_line1967                 1.375398   1.962378   0.701 0.483376
## earliest_cr_line1968                 1.468830   1.962069   0.749 0.454091
## earliest_cr_line1969                 1.107192   1.960703   0.565 0.572284
## earliest_cr_line1970                 1.346445   1.960200   0.687 0.492151
## earliest_cr_line1971                 1.106779   1.960189   0.565 0.572327
## earliest_cr_line1972                 1.202393   1.959104   0.614 0.539383
## earliest_cr_line1973                 1.196592   1.958860   0.611 0.541292
## earliest_cr_line1974                 1.363708   1.958664   0.696 0.486276
## earliest_cr_line1975                 1.374496   1.958470   0.702 0.482791
## earliest_cr_line1976                 1.416585   1.958127   0.723 0.469411
## earliest_cr_line1977                 1.350826   1.957907   0.690 0.490236
## earliest_cr_line1978                 1.332256   1.957786   0.680 0.496194
## earliest_cr_line1979                 1.355165   1.957757   0.692 0.488810
## earliest_cr_line1980                 1.389232   1.957795   0.710 0.477959
## earliest_cr_line1981                 1.428219   1.957649   0.730 0.465660
## earliest_cr_line1982                 1.365158   1.957523   0.697 0.485559
## earliest_cr_line1983                 1.414262   1.957416   0.723 0.469978
## earliest_cr_line1984                 1.489322   1.957357   0.761 0.446727
```

```
## earliest_cr_line1985            1.469763   1.957340   0.751 0.452714
## earliest_cr_line1986            1.517023   1.957311   0.775 0.438308
## earliest_cr_line1987            1.534787   1.957278   0.784 0.432956
## earliest_cr_line1988            1.543214   1.957258   0.788 0.430430
## earliest_cr_line1989            1.549040   1.957231   0.791 0.428685
## earliest_cr_line1990            1.554636   1.957219   0.794 0.427016
## earliest_cr_line1991            1.589613   1.957225   0.812 0.416690
## earliest_cr_line1992            1.601032   1.957217   0.818 0.413349
## earliest_cr_line1993            1.627125   1.957181   0.831 0.405770
## earliest_cr_line1994            1.643300   1.957165   0.840 0.401114
## earliest_cr_line1995            1.671327   1.957158   0.854 0.393129
## earliest_cr_line1996            1.699569   1.957154   0.868 0.385182
## earliest_cr_line1997            1.710917   1.957153   0.874 0.382017
## earliest_cr_line1998            1.726896   1.957144   0.882 0.377585
## earliest_cr_line1999            1.755628   1.957137   0.897 0.369699
## earliest_cr_line2000            1.769144   1.957132   0.904 0.366024
## earliest_cr_line2001            1.829731   1.957132   0.935 0.349838
## earliest_cr_line2002            1.886259   1.957136   0.964 0.335154
## earliest_cr_line2003            1.916286   1.957140   0.979 0.327518
## earliest_cr_line2004            2.036281   1.957146   1.040 0.298139
## earliest_cr_line2005            2.141803   1.957159   1.094 0.273805
## earliest_cr_line2006            2.315136   1.957174   1.183 0.236850
## earliest_cr_line2007            2.542659   1.957196   1.299 0.193899
## earliest_cr_line2008            2.883637   1.957254   1.473 0.140669
## earliest_cr_line2009            3.128877   1.957360   1.599 0.109928
## earliest_cr_line2010            3.333768   1.957406   1.703 0.088539
## earliest_cr_line2011            3.450617   1.957499   1.763 0.077940
## earliest_cr_line2012            3.726572   1.958351   1.903 0.057052
## last_credit_pull_d2008          0.971470   0.605256   1.605 0.108482
## last_credit_pull_d2009          0.607618   0.487981   1.245 0.213071
## last_credit_pull_d2010          0.384834   0.470405   0.818 0.413305
## last_credit_pull_d2011         -0.177715   0.465650  -0.382 0.702721
## last_credit_pull_d2012         -0.379274   0.464138  -0.817 0.413839
## last_credit_pull_d2013          0.041658   0.463205   0.090 0.928340
## last_credit_pull_d2014          0.523211   0.462749   1.131 0.258199
## last_credit_pull_d2015          0.651862   0.462538   1.409 0.158742
## last_credit_pull_d2016          0.193406   0.462464   0.418 0.675795
## total_acc                      -0.137579   0.005034 -27.329  < 2e-16
## tot_cur_bal                    -0.113345   0.004420 -25.643  < 2e-16
## open_acc                        0.193421   0.004931  39.222  < 2e-16
## acc_now_delinq                  0.068590   0.003299  20.790  < 2e-16
## delinq_2yrs                     0.238213   0.003875  61.482  < 2e-16
## mths_since_last_delinq          0.033740   0.004087   8.255  < 2e-16
## collections_12_mths_ex_med      0.050668   0.003328  15.224  < 2e-16
## tot_coll_amt                    0.239234   0.017302  13.827  < 2e-16
## pub_rec                         0.228828   0.003354  68.224  < 2e-16
## mths_since_last_major_derog     0.041425   0.003773  10.979  < 2e-16
## revol_util                      0.722650   0.003667 197.042  < 2e-16
## total_rev_hi_lim               -0.260701   0.004049 -64.391  < 2e-16
## loan_amnt                      -2.277094   0.086610 -26.291  < 2e-16
## application_typeJOINT           0.932187   0.134313   6.940 3.91e-12
## purposecredit_card             -0.527380   0.033839 -15.585  < 2e-16
## purposedebt_consolidation       0.419007   0.033376  12.554  < 2e-16
## purposeeducational              1.381881   0.154746   8.930  < 2e-16
```

```
## purposehome_improvement              0.730999   0.035748  20.448  < 2e-16
## purposehouse                         2.408781   0.060696  39.686  < 2e-16
## purposemajor_purchase                0.607569   0.040434  15.026  < 2e-16
## purposemedical                       2.176207   0.047028  46.274  < 2e-16
## purposemoving                        3.094704   0.053561  57.779  < 2e-16
## purposeother                         2.182518   0.036169  60.342  < 2e-16
## purposerenewable_energy              2.433864   0.134302  18.122  < 2e-16
## purposesmall_business                2.316455   0.045012  51.463  < 2e-16
## purposevacation                      2.591219   0.055834  46.409  < 2e-16
## purposewedding                       1.478811   0.072512  20.394  < 2e-16
## term 60 months                      11.058876   0.014833 745.547  < 2e-16
## initial_list_statusw                -0.695551   0.007107 -97.868  < 2e-16
## installment                         10.951038   0.019438 563.381  < 2e-16
## funded_amnt                         -9.422854   0.108169 -87.112  < 2e-16
## funded_amnt_inv                      0.111427   0.057270   1.946 0.051698
## pymnt_plany                          0.747394   0.978604   0.764 0.445025
## recoveries                           0.139759   0.005665  24.672  < 2e-16
## collection_recovery_fee             -0.039297   0.005528  -7.108 1.18e-12
## out_prncp                           12.939415   1.414022   9.151  < 2e-16
## out_prncp_inv                      -12.968174   1.414214  -9.170  < 2e-16
##
## (Intercept)                        ***
## home_ownershipOTHER                ***
## home_ownershipOWN                  ***
## home_ownershipRENT                 ***
## state_mean_intlow                  ***
## state_mean_intlowmedium            ***
## state_mean_intmediumhigh           ***
## dti                                ***
## annual_inc                         ***
## verification_statusSource Verified ***
## verification_statusVerified        ***
## earliest_cr_line1946
## earliest_cr_line1949
## earliest_cr_line1950
## earliest_cr_line1951               .
## earliest_cr_line1952
## earliest_cr_line1953
## earliest_cr_line1954
## earliest_cr_line1955
## earliest_cr_line1956
## earliest_cr_line1957
## earliest_cr_line1958
## earliest_cr_line1959
## earliest_cr_line1960
## earliest_cr_line1961
## earliest_cr_line1962
## earliest_cr_line1963
## earliest_cr_line1964
## earliest_cr_line1965
## earliest_cr_line1966
## earliest_cr_line1967
## earliest_cr_line1968
## earliest_cr_line1969
```

```
## earliest_cr_line1970
## earliest_cr_line1971
## earliest_cr_line1972
## earliest_cr_line1973
## earliest_cr_line1974
## earliest_cr_line1975
## earliest_cr_line1976
## earliest_cr_line1977
## earliest_cr_line1978
## earliest_cr_line1979
## earliest_cr_line1980
## earliest_cr_line1981
## earliest_cr_line1982
## earliest_cr_line1983
## earliest_cr_line1984
## earliest_cr_line1985
## earliest_cr_line1986
## earliest_cr_line1987
## earliest_cr_line1988
## earliest_cr_line1989
## earliest_cr_line1990
## earliest_cr_line1991
## earliest_cr_line1992
## earliest_cr_line1993
## earliest_cr_line1994
## earliest_cr_line1995
## earliest_cr_line1996
## earliest_cr_line1997
## earliest_cr_line1998
## earliest_cr_line1999
## earliest_cr_line2000
## earliest_cr_line2001
## earliest_cr_line2002
## earliest_cr_line2003
## earliest_cr_line2004
## earliest_cr_line2005
## earliest_cr_line2006
## earliest_cr_line2007
## earliest_cr_line2008
## earliest_cr_line2009
## earliest_cr_line2010               .
## earliest_cr_line2011               .
## earliest_cr_line2012               .
## last_credit_pull_d2008
## last_credit_pull_d2009
## last_credit_pull_d2010
## last_credit_pull_d2011
## last_credit_pull_d2012
## last_credit_pull_d2013
## last_credit_pull_d2014
## last_credit_pull_d2015
## last_credit_pull_d2016
## total_acc                          ***
## tot_cur_bal                        ***
```

```
## open_acc                         ***
## acc_now_delinq                   ***
## delinq_2yrs                      ***
## mths_since_last_delinq           ***
## collections_12_mths_ex_med       ***
## tot_coll_amt                     ***
## pub_rec                          ***
## mths_since_last_major_derog      ***
## revol_util                       ***
## total_rev_hi_lim                 ***
## loan_amnt                        ***
## application_typeJOINT            ***
## purposecredit_card               ***
## purposedebt_consolidation        ***
## purposeeducational               ***
## purposehome_improvement          ***
## purposehouse                     ***
## purposemajor_purchase            ***
## purposemedical                   ***
## purposemoving                    ***
## purposeother                     ***
## purposerenewable_energy          ***
## purposesmall_business            ***
## purposevacation                  ***
## purposewedding                   ***
## term 60 months                   ***
## initial_list_statusw             ***
## installment                      ***
## funded_amnt                      ***
## funded_amnt_inv                  .
## pymnt_plany
## recoveries                       ***
## collection_recovery_fee          ***
## out_prncp                        ***
## out_prncp_inv                    ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.768 on 709719 degrees of freedom
##   (62 observations deleted due to missingness)
## Multiple R-squared:  0.601,  Adjusted R-squared:  0.6009
## F-statistic:  8834 on 121 and 709719 DF,  p-value: < 2.2e-16
```

Based on the summary information, I notice some features are not significant in building linear regression. So I decided to add Lasso regularization to penalize them.

```r
library(glmnet)
```

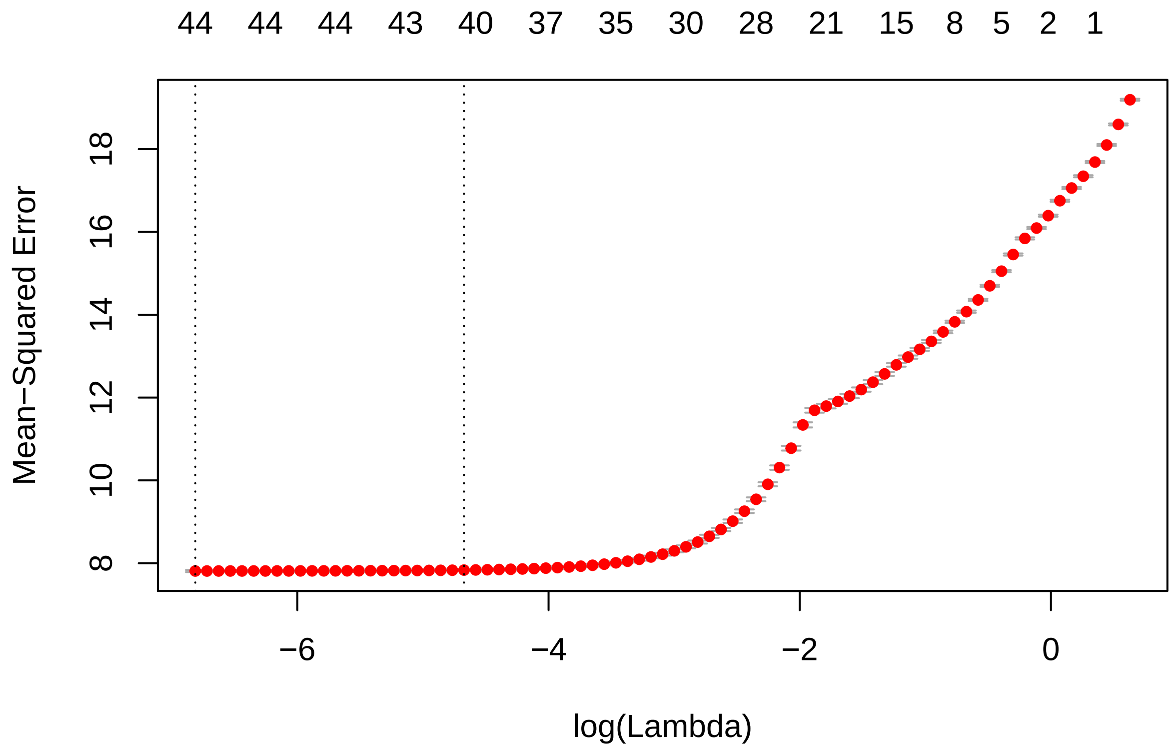
```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-13
```

```r
drops <- c("last_credit_pull_d", "earliest_cr_line", "funded_amnt_inv", "pymnt_plan",
    "int_rate")
ind <- train[, !(names(train) %in% drops)]
```

```r
ind <- model.matrix(~., ind)
dep <- train[, "int_rate"]
# Use cross validation to tune parameters
linear.cvfit <- cv.glmnet(ind, dep, family = "gaussian", alpha = 1)
plot(linear.cvfit)
```



Choose optimus parameters for this linear regression model.

```r
print(paste("The optimus lambda for model is", round(linear.cvfit$lambda.1se,
    5)))
```

```
## [1] "The optimus lambda for model is 0.00934"
```

```r
print(coef(linear.cvfit, s = "lambda.1se"))
```

```
## 47 x 1 sparse Matrix of class "dgCMatrix"
##                                            1
## (Intercept)                      9.834346613
## (Intercept)                      .
## home_ownershipOTHER              0.229581885
## home_ownershipOWN                0.305355320
## home_ownershipRENT               0.393131375
## state_mean_intlow               -0.173377826
## state_mean_intlowmedium         -0.014037715
## state_mean_intmediumhigh        -0.006463066
## dti                              0.265344320
## annual_inc                      -0.143185329
## verification_statusSource Verified  0.238613364
## verification_statusVerified      1.018611085
## total_acc                       -0.261678813
## tot_cur_bal                     -0.106412312
## open_acc                         0.234646512
```

```
## acc_now_delinq                    0.057446949
## delinq_2yrs                       0.194199527
## mths_since_last_delinq            .
## collections_12_mths_ex_med        0.040846361
## tot_coll_amt                      0.161642597
## pub_rec                           0.193772046
## mths_since_last_major_derog       0.031351450
## revol_util                        0.695764474
## total_rev_hi_lim                 -0.318058798
## loan_amnt                        -3.474334907
## application_typeJOINT             0.675893946
## purposecredit_card               -0.965759973
## purposedebt_consolidation         .
## purposeeducational                0.156677244
## purposehome_improvement           0.252616071
## purposehouse                      1.919354534
## purposemajor_purchase             0.135821497
## purposemedical                    1.628808633
## purposemoving                     2.608224099
## purposeother                      1.731291571
## purposerenewable_energy           1.688822941
## purposesmall_business             1.849671796
## purposevacation                   2.049574723
## purposewedding                    0.825003477
## term 60 months                   10.684587364
## initial_list_statusw             -0.687818705
## installment                      10.390850847
## funded_amnt                      -7.486506220
## recoveries                        0.117497811
## collection_recovery_fee           .
## out_prncp                         .
## out_prncp_inv                    -0.106171748
```

make predictions for test data set

```r
library(hydroGOF)
ind <- test[, !(names(test) %in% drops)]
ind <- model.matrix(~., ind)
cv.pred <- predict(linear.cvfit, s = linear.cvfit$lambda.1se, newx = ind)
print(paste0("The mean square error is: ", round(mse(cv.pred[, 1], test$int_rate),
    4), "%"))
```

```
## [1] "The mean square error is: 8.0005%"
```

```r
print(paste0("The mean absolute error is: ", round(mae(cv.pred[, 1], test$int_rate),
    4), "%"))
```

```
## [1] "The mean absolute error is: 2.2299%"
```