Linear Regression

Jieqi Tu (jt3098)

3/26/2019

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------------------- tidy
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.7
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts -------------------------------------------------------------------------- tidyverse_
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library("leaps")
```

#Load and tidy data

```
#read data
rawdata <- read.csv("kc_house_data.csv", header = TRUE)

#inspect the structure of data
str(rawdata)
```

```
## 'data.frame':    21613 obs. of  21 variables:
##  $ id           : num  7.13e+09 6.41e+09 5.63e+09 2.49e+09 1.95e+09 ...
##  $ date         : Factor w/ 372 levels "20140502T000000",..: 165 221 291 221 284 11 57 252 340 306 .
##  $ price        : num  221900 538000 180000 604000 510000 ...
##  $ bedrooms     : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms    : num  1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
##  $ sqft_living  : int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ sqft_lot     : int  5650 7242 10000 5000 8080 101930 6819 9711 7470 6560 ...
##  $ floors       : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ waterfront   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ view         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition    : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ sqft_above   : int  1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
##  $ sqft_basement: int  0 400 0 910 0 1530 0 0 730 0 ...
##  $ yr_built     : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
##  $ yr_renovated : int  0 1991 0 0 0 0 0 0 0 0 ...
##  $ zipcode      : int  98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
```

```
##  $ lat          : num  47.5 47.7 47.7 47.5 47.6 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
##  $ sqft_living15: int  1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
##  $ sqft_lot15   : int  5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
```

`id`, `date`, `zipcode`, `lat`, `long` can be removed from the dataframe.
The cleaned dataset to be used in this project include one response variable `price` and additional 15 variables. `view`, `sqft_basement` and `yr_renovated` are continuous or integer variables in the original dataset. For the purpose of easy interpretation in later modelling, we convert these variables to be binary. Then these variables indicate whether the house has been viewed by potential buyers, whether the house has basement or not and whether the house has been renovated or not, respectively.

```
#clean the rawdata; create a tidied dataset for analysis and modelling.
#subset data: only those with view >0.
housing =
  rawdata %>%
  select(-id, -date, -zipcode, -lat, -long) %>%
  mutate(price = (price^0.02-1)/0.02) %>%
  filter(view > 0 & bedrooms <30) %>%
  mutate(basement = ifelse(sqft_basement == 0, 0, 1),
         renovated = ifelse(yr_renovated == 0, 0, 1)) %>%
  select(-sqft_basement, -yr_renovated, -view, - sqft_living, -sqft_lot)
```

```
# vector of response
y <- housing$price
x <- model.matrix(price~.,housing)[,-1]
```

Model Building

Linear model

```
# build a least-square linear model
ctrl1 = trainControl(method = "repeatedcv", number = 10, repeats = 5)
set.seed(1)

# stepwise elimination to select variables
lm_fit = lm(price~., data = housing)
step(lm_fit, direction = 'backward')
```

```
## Start:  AIC=-3710.37
## price ~ bedrooms + bathrooms + floors + waterfront + condition +
##     grade + sqft_above + yr_built + sqft_living15 + sqft_lot15 +
##     basement + renovated
##
##                 Df Sum of Sq    RSS     AIC
## <none>                       365.74 -3710.4
## - bedrooms       1     0.369 366.11 -3710.2
## - renovated      1     1.272 367.02 -3705.0
## - floors         1     2.308 368.05 -3699.0
## - sqft_lot15     1     2.448 368.19 -3698.2
## - bathrooms      1     4.244 369.99 -3687.9
## - condition      1     6.808 372.55 -3673.2
## - sqft_above     1     7.140 372.88 -3671.3
## - basement       1    10.448 376.19 -3652.5
## - sqft_living15  1    20.555 386.30 -3596.2
```

```
## - yr_built       1    30.276 396.02 -3543.4
## - waterfront     1    46.622 412.37 -3457.5
## - grade          1    82.144 447.89 -3282.0
```

```
##
## Call:
## lm(formula = price ~ bedrooms + bathrooms + floors + waterfront +
##     condition + grade + sqft_above + yr_built + sqft_living15 +
##     sqft_lot15 + basement + renovated, data = housing)
##
## Coefficients:
##   (Intercept)       bedrooms      bathrooms         floors     waterfront
##     2.294e+01     -1.685e-02      8.722e-02      7.812e-02      5.711e-01
##     condition          grade     sqft_above       yr_built  sqft_living15
##     9.126e-02      2.568e-01      1.194e-04     -5.599e-03      1.799e-04
##    sqft_lot15       basement      renovated
##    -8.558e-07      1.857e-01      9.137e-02
```

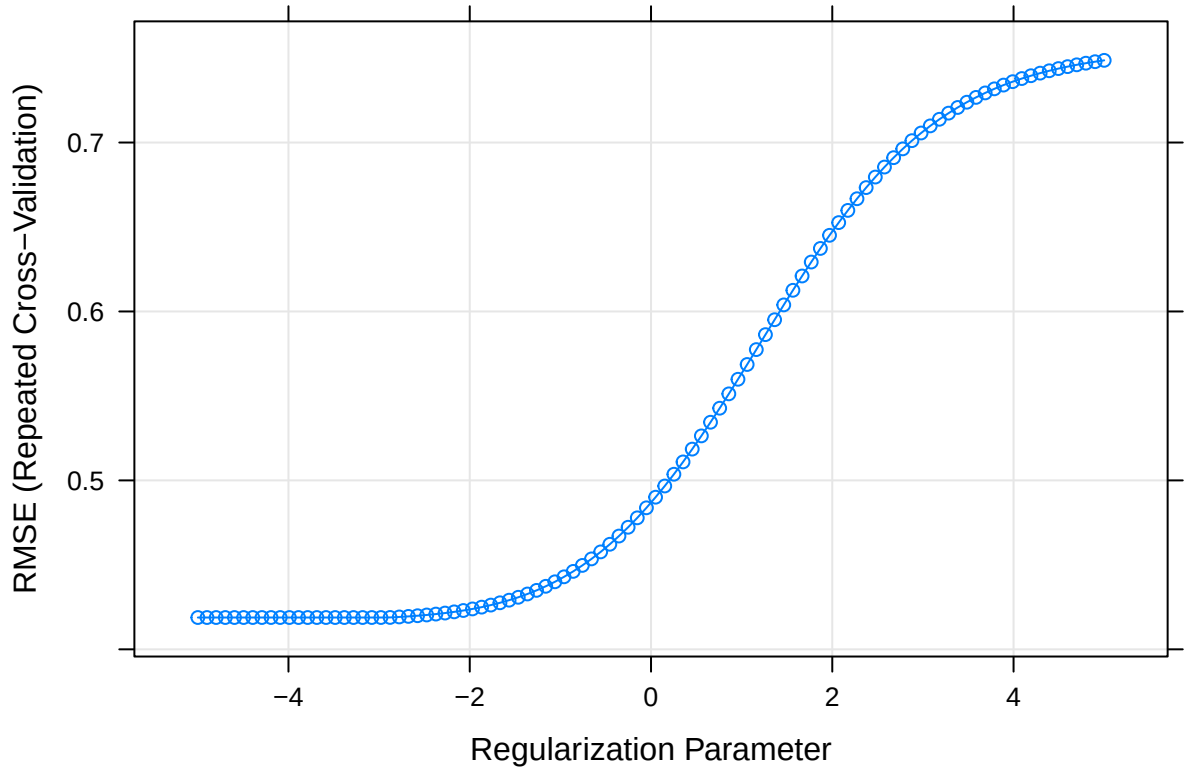
```
model_ls = train(x, y,
                 method = "lm",
                 preProcess = c("center", "scale"),
                 trControl = ctrl1)

#obtain coefficients
coef_ls = model_ls$finalModel$coefficients %>% as.data.frame(); coef_ls
```

```
##                          .
## (Intercept)   15.60147966
## bedrooms      -0.01676496
## bathrooms      0.08023556
## floors         0.04287175
## waterfront     0.15206428
## condition      0.06310154
## grade          0.36608136
## sqft_above     0.12260930
## yr_built      -0.16198233
## sqft_living15  0.14088610
## sqft_lot15    -0.03538383
## basement       0.08660144
## renovated      0.02745167
```

```
set.seed(1)
# fit a ridge model using caret package
ridge.fit = train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 0,
                                         lambda = exp(seq(-5, 5, length = 100))),
                  preProcess = c("center", "scale"),
                  trControl = ctrl1)

# plot the RMSE by log(lambda)
plot(ridge.fit, xTrans = function(x) log(x))
```




```
# find the optimal lambda
ridge.fit$bestTune
```

```
##    alpha    lambda
## 21     0 0.0508031
```

```
# obtain the coefficients of ridge model
coef_ridge = coef(ridge.fit$finalModel, ridge.fit$bestTune$lambda); coef_ridge
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                            1
## (Intercept)   15.60147966
## bedrooms      -0.01076302
## bathrooms      0.08428439
## floors         0.04513391
## waterfront     0.14361055
## condition      0.06325932
## grade          0.31504291
## sqft_above     0.13065375
## yr_built      -0.13555821
## sqft_living15  0.14701303
## sqft_lot15    -0.03535608
## basement       0.08591822
## renovated      0.03326277
```

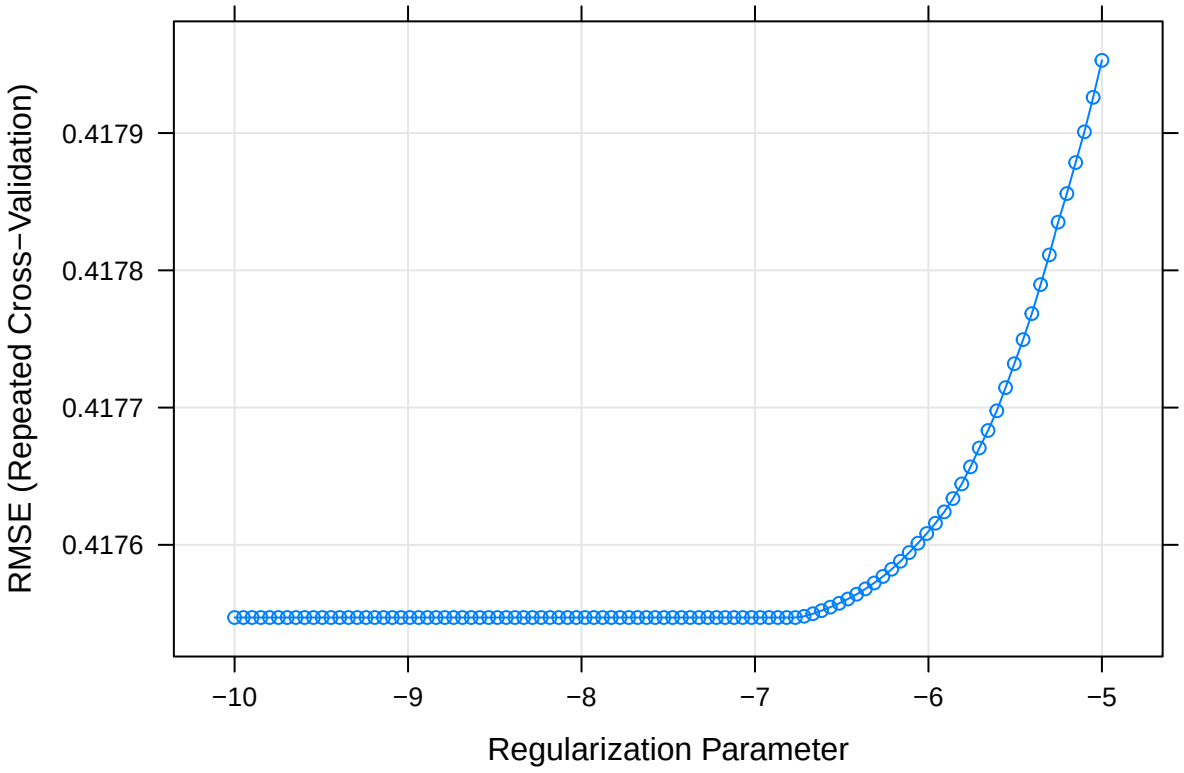
```
summary(ridge.fit)
```

```
##             Length Class      Mode
```

```
## a0           100   -none-     numeric
## beta        1200   dgCMatrix  S4
## df           100   -none-     numeric
## dim            2   -none-     numeric
## lambda       100   -none-     numeric
## dev.ratio    100   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        12   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      1   -none-     logical
## param          0   -none-     list
```

```
# fit a lasso model using caret
set.seed(1)
lasso.fit = train(x, y,
                  method = "glmnet",
                  tuneGrid = expand.grid(alpha = 1,
                                         lambda = exp(seq(-10, -5, length = 100))),
                  preProcess = c("center", "scale"),
                  trControl = ctrl1)

# plot the RMSE by log(lambda)
plot(lasso.fit, xTrans = function(x) log(x))
```




```
# obtain the optimal lambda
lasso.fit$bestTune
```

```
##    alpha      lambda
## 65     1 0.001150364
```

```
# check the coefficients for each predictors
coef(lasso.fit$finalModel, lasso.fit$bestTune$lambda)
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)   15.60147966
## bedrooms      -0.01333004
## bathrooms      0.07880923
## floors         0.04199924
## waterfront     0.15148964
## condition      0.06213604
## grade          0.36669154
## sqft_above     0.12027649
## yr_built      -0.16031943
## sqft_living15  0.13995017
## sqft_lot15    -0.03413374
## basement       0.08480128
## renovated      0.02709483
```

```
set.seed(1)
resamp = resamples(list(lasso = lasso.fit, ridge = ridge.fit, lm = model_ls))
summary(resamp)
```

```
##
## Call:
## summary.resamples(object = resamp)
##
## Models: lasso, ridge, lm
## Number of resamples: 50
##
## MAE
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.2921951 0.3160311 0.3273776 0.3276575 0.3410717 0.3631973    0
## ridge 0.2920736 0.3175003 0.3302950 0.3293157 0.3427772 0.3634749    0
## lm    0.2920987 0.3158639 0.3271664 0.3276489 0.3413040 0.3637798    0
##
## RMSE
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.3678710 0.4039861 0.4194591 0.4175471 0.4309753 0.4600296    0
## ridge 0.3662436 0.4042609 0.4200913 0.4188261 0.4319464 0.4569470    0
## lm    0.3683838 0.4040081 0.4192321 0.4175325 0.4312641 0.4604370    0
##
## Rsquared
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## lasso 0.6032333 0.6812459 0.6933172 0.6964344 0.7187618 0.7669841    0
## ridge 0.6069565 0.6792029 0.6928985 0.6954973 0.7179654 0.7671813    0
## lm    0.6030038 0.6809135 0.6933454 0.6964481 0.7192195 0.7673424    0
```

```r
bwplot(resamp, metric = "RMSE")
```