Thota, Sunil Raj - LASSO Regression in R Practice.R

```
# Intermediate Analytics
# ALY 6015
# Module 3 - LASSO Regression in R Practice
# 02/03/2021
# Sunil Raj Thota
# NUID: 001099670

# Get and set the working directories
getwd()

## [1] "G:/NEU/Coursework/2021 Q1 Winter/ALY 6015 IA/Discussions &
Assignments"

setwd('G:/NEU/Coursework/2021 Q1 Winter/ALY 6015 IA/Discussions &
Assignments')
getwd()

## [1] "G:/NEU/Coursework/2021 Q1 Winter/ALY 6015 IA/Discussions &
Assignments"

# Installed the above packages into the work space


install.packages("datasets")
install.packages("plyr")
install.packages("dplyr")
install.packages("tidyr")
install.packages("ncvreg")
install.packages("biglasso")
install.packages("bigmemory")
install.packages("glmnet")
install.packages("lars")

# Loaded the below libraries into the work space

library(plyr)
library(dplyr)
library(tidyr)
require(datasets)
library(biglasso)
library(bigmemory)
library(ncvreg)
```

```r
# Exercise 1

library(lars)

data(diabetes)
attach(diabetes)

View(diabetes)
str(diabetes)

## 'data.frame':    442 obs. of  3 variables:
##  $ x : 'AsIs' num [1:442, 1:10] 0.03808 -0.00188 0.0853 -0.08906 0.00538
...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:10] "age" "sex" "bmi" "map" ...
##  $ y : num  151 75 141 206 135 97 138 63 110 310 ...
##  $ x2: 'AsIs' num [1:442, 1:64] 0.03808 -0.00188 0.0853 -0.08906 0.00538
...
##   ..- attr(*, ".Names")= chr [1:28288] "age" "age" "age" "age" ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:442] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:64] "age" "sex" "bmi" "map" ...

head(diabetes)

##          x.age        x.sex        x.bmi        x.map         x.tc
x.ldl
## 1  0.038075906  0.050680119  0.061696207  0.021872355 -0.044223498 -
0.034820763
## 2 -0.001882017 -0.044641637 -0.051474061 -0.026327835 -0.008448724 -
0.019163340
## 3  0.085298906  0.050680119  0.044451213 -0.005670611 -0.045599451 -
0.034194466
## 4 -0.089062939 -0.044641637 -0.011595015 -0.036656447  0.012190569
0.024990593
## 5  0.005383060 -0.044641637 -0.036384692  0.021872355  0.003934852
0.015596140
## 6 -0.092695478 -0.044641637 -0.040695940 -0.019442093 -0.068990650 -
0.079287844
##          x.hdl        x.tch        x.ltg        x.glu   y        x2.age
## 1 -0.043400846 -0.002592262  0.019908421 -0.017646125 151  0.0380759064
## 2  0.074411564 -0.039493383 -0.068329744 -0.092204050  75 -0.0018820165
## 3 -0.032355932 -0.002592262  0.002863771 -0.025930339 141  0.0852989063
## 4 -0.036037570  0.034308859  0.022692023 -0.009361911 206 -0.0890629394
## 5  0.008142084 -0.002592262 -0.031991445 -0.046640874 135  0.0053830604
## 6  0.041276824 -0.076394504 -0.041180385 -0.096346157  97 -0.0926954778
##          x2.sex        x2.bmi        x2.map         x2.tc        x2.ldl
## 1  0.0506801187  0.0616962065  0.0218723550 -0.0442234984 -0.0348207628
## 2 -0.0446416365 -0.0514740612 -0.0263278347 -0.0084487241 -0.0191633397
## 3  0.0506801187  0.0444512133 -0.0056706106 -0.0455994513 -0.0341944659
```

```
## 4 -0.0446416365 -0.0115950145 -0.0366564468  0.0121905688  0.0249905934
## 5 -0.0446416365 -0.0363846922  0.0218723550  0.0039348516  0.0155961395
## 6 -0.0446416365 -0.0406959405 -0.0194420933 -0.0689906499 -0.0792878444
##          x2.hdl        x2.tch        x2.ltg        x2.glu      x2.age^2
## 1 -0.0434008457 -0.0025922620  0.0199084209 -0.0176461252 -0.0148551625
## 2  0.0744115641 -0.0394933829 -0.0683297436 -0.0922040496 -0.0412915429
## 3 -0.0323559322 -0.0025922620  0.0028637705 -0.0259303390  0.0916434391
## 4 -0.0360375700  0.0343088589  0.0226920226 -0.0093619113  0.1036403301
## 5  0.0081420836 -0.0025922620 -0.0319914449 -0.0466408736 -0.0408265979
## 6  0.0412768238 -0.0763945038 -0.0411803852 -0.0963461565  0.1157092549
##        x2.bmi^2      x2.map^2       x2.tc^2      x2.ldl^2      x2.hdl^2
## 1  0.0225045739 -0.0310446765 -0.0043311197 -0.0137399243 -0.0046314248
## 2  0.0056427733 -0.0273076609 -0.0309389016 -0.0248010319  0.0400365241
## 3 -0.0041764214 -0.0388099038 -0.0025859366 -0.0143055611 -0.0148614560
## 4 -0.0310170859 -0.0159874156 -0.0298483890 -0.0214340114 -0.0117828858
## 5 -0.0136807215 -0.0310446765 -0.0317282077 -0.0264236384 -0.0268506700
## 6 -0.0088370145 -0.0327918526  0.0352626533  0.0526602897 -0.0068304035
##        x2.tch^2      x2.ltg^2      x2.glu^2    x2.age:sex    x2.age:bmi
## 1 -0.0304484629 -0.0288162192 -0.0275255618  0.0328649758  0.0405716741
## 2 -0.0094854824  0.0371612444  0.0880219609 -0.0066099928 -0.0067648038
## 3 -0.0304484629 -0.0348099250 -0.0224326123  0.0840517453  0.0708891293
## 4 -0.0146503349 -0.0269850701 -0.0306820952  0.0766292449  0.0129034054
## 5 -0.0304484629 -0.0191324512 -0.0012284185 -0.0135465959 -0.0129173209
## 6  0.0482386067 -0.0087497144  0.0990402558  0.0800975464  0.0704832806
##      x2.age:map     x2.age:tc    x2.age:ldl    x2.age:hdl    x2.age:tch
## 1  0.0016606410 -0.0465532511 -0.0382447104 -0.0345115069 -0.0121122609
## 2 -0.0159342243 -0.0117287997 -0.0096555406  0.0006995477 -0.0083689963
## 3 -0.0279128687 -0.0917442739 -0.0716415163 -0.0602920920 -0.0147605279
## 4  0.0562904032 -0.0342989420 -0.0571356258  0.0786805432 -0.0760817036
## 5 -0.0144024121 -0.0116206046 -0.0086502423  0.0049801694 -0.0102788452
## 6  0.0234365273  0.1189685396  0.1438718189 -0.0851146968  0.1432199339
##      x2.age:ltg    x2.age:glu    x2.sex:bmi    x2.sex:map     x2.sex:tc
## 1  0.0030648916 -0.0302775066  0.0621030123  0.0122820371 -0.0485722318
## 2 -0.0102016795 -0.0113801440  0.0445181909  0.0137392710  0.0062226212
## 3 -0.0077635174 -0.0646990887  0.0435615292 -0.0181579597 -0.0500315236
## 4 -0.0555091413  0.0033785934  0.0067497817  0.0237941848 -0.0130586592
## 5 -0.0165418441 -0.0208710562  0.0302274415 -0.0331836602 -0.0053461470
## 6  0.0675437504  0.1843686827  0.0343105127  0.0070359951  0.0627810439
##      x2.sex:ldl    x2.sex:hdl    x2.sex:tch    x2.sex:ltg    x2.sex:glu
## 1 -0.0441240238 -0.0308042981 -0.0195479919  0.0142268228 -0.0293859648
## 2  0.0112617659 -0.0565675488  0.0224021267  0.0575880019  0.0784642525
## 3 -0.0434530875 -0.0179545693 -0.0195479919 -0.0041216840 -0.0384231554
## 4 -0.0304033769  0.0566194244 -0.0505546014 -0.0287218500 -0.0011399371
## 5 -0.0215384529  0.0113446351 -0.0140762373  0.0231308241  0.0346819482
## 6  0.0679972794 -0.0226114568  0.0588804908  0.0318440815  0.0824444620
##      x2.bmi:map     x2.bmi:tc    x2.bmi:ldl    x2.bmi:hdl    x2.bmi:tch
## 1  0.0090008988 -0.0717180778 -0.0603176794 -0.0413575386 -0.0240342004
## 2  0.0091148702 -0.0028355367  0.0087097314 -0.0671553344  0.0240456899
## 3 -0.0226918084 -0.0564432291 -0.0464818380 -0.0136167525 -0.0230540270
## 4 -0.0092925186 -0.0153834201 -0.0193921008  0.0279274131 -0.0292499537
```

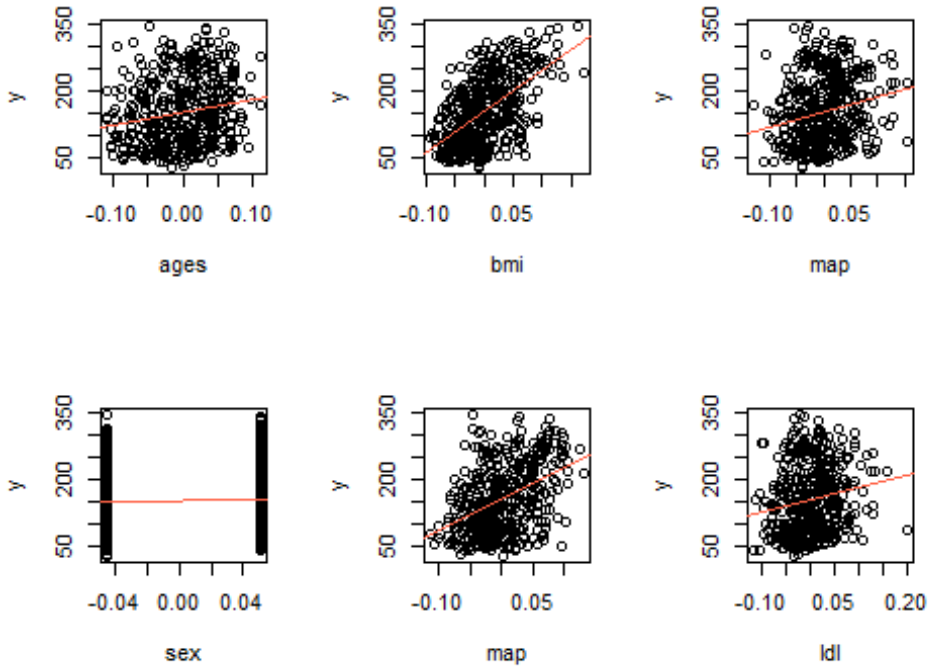
```
## 5 -0.0334523099 -0.0154230197 -0.0255070025  0.0119441348 -0.0184594644
## 6 -0.0020460266  0.0488321410  0.0580412284 -0.0190229456  0.0476394964
##      x2.bmi:ltg    x2.bmi:glu     x2.map:tc    x2.map:ldl    x2.map:hdl
## 1  0.0048261936 -0.0410278367 -0.0327209536 -0.0257474212 -0.0112074291
## 2  0.0552994440  0.0806093115 -0.0070399803  0.0018462404 -0.0319794277
## 3 -0.0194513972 -0.0423606977 -0.0062598621 -0.0049233843  0.0120934525
## 4 -0.0280603703 -0.0160690142 -0.0214874364 -0.0291135216  0.0354925517
## 5  0.0034088636  0.0170453175 -0.0099837004 -0.0017149265  0.0119825521
## 6  0.0146962189  0.0634061398  0.0171122204  0.0244459437 -0.0081883427
##      x2.map:tch    x2.map:ltg    x2.map:glu     x2.tc:ldl     x2.tc:hdl
## 1 -0.0138251131 -0.0101938361 -0.0257784633 -0.0068689647  0.0398230899
## 2  0.0098745200  0.0203696547  0.0313619644 -0.0262353123 -0.0164622948
## 3 -0.0122818754 -0.0203183065 -0.0149534838 -0.0065969775  0.0300168640
## 4 -0.0397827564 -0.0385992765 -0.0109701272 -0.0242291837 -0.0122792654
## 5 -0.0138251131 -0.0356386917 -0.0386583542 -0.0276482686 -0.0018670731
## 6  0.0195036026 -0.0020081382  0.0201031994  0.0483666130 -0.0654803859
##       x2.tc:tch     x2.tc:ltg     x2.tc:glu    x2.ldl:hdl    x2.ldl:tch
## 1 -0.0193030537 -0.0428915072  0.0009629700  0.0423549304 -0.0220378305
## 2 -0.0155012002 -0.0123430995  0.0009326831 -0.0212564243 -0.0115642516
## 3 -0.0192411418 -0.0271777643  0.0098716066  0.0335869646 -0.0220633407
## 4 -0.0140331587 -0.0186440420 -0.0188580962 -0.0098784247 -0.0099839514
## 5 -0.0214699727 -0.0270791696 -0.0203958711  0.0123759491 -0.0240914053
## 6  0.0701909475  0.0350969957  0.1309600895 -0.0612519836  0.0717191451
##      x2.ldl:ltg    x2.ldl:glu    x2.hdl:tch    x2.hdl:ltg    x2.hdl:glu
## 1 -0.0311245646 -0.0009221095  0.0334936252  0.0008521487  0.0311502576
## 2  0.0129733755  0.0237834359 -0.0238146613 -0.0945055990 -0.1403775894
## 3 -0.0180161691  0.0049134553  0.0329558784  0.0182807986  0.0327952439
## 4 -0.0033727555 -0.0191092832  0.0081586184  0.0018977323  0.0215138966
## 5 -0.0268464903 -0.0296874121  0.0309841399  0.0144891320  0.0053856590
## 6  0.0560369456  0.1496630223 -0.0278444433 -0.0180308997 -0.0755127518
##      x2.tch:ltg    x2.tch:glu    x2.ltg:glu
## 1 -0.0281911757 -0.0176581553 -0.0277936831
## 2  0.0252977155  0.0530335390  0.1040132768
## 3 -0.0273318271 -0.0172359590 -0.0223037368
## 4 -0.0120454909 -0.0248722040 -0.0250419367
## 5 -0.0255745144 -0.0161804685  0.0087351987
## 6  0.0339989574  0.1261465729  0.0577886517

# Let's perform some Regularization analysis and techniques using "diabetes"
data set. To get this data set we need to install the 'lars' package from the
packages tab which is right side to the work space in R Studio.

# Or we can also install the packages by using install.packages("package
name") command. Once it is loaded we can use it in the code for further
analysis and calculations.

# Loaded the "lars" library into the work space. Loaded the diabetes Data set
into the Environment. To reduce the repetitive usage of "x" in "diabetes"
data set, "attach" is used to set it once throughout the work space.
```

```r
# To View the diabetes Data set we use View() command, To observe the
structure of the Data set we use str() command, and head () and tail() shows
first and last few rows in the Data set

# Exercise 2

summary(x)
```

```
##       age                 sex                 bmi                 map
##  Min.   :-0.107226   Min.   :-0.04464   Min.   :-0.090275   Min.   :-
0.112400
##  1st Qu.:-0.037299   1st Qu.:-0.04464   1st Qu.:-0.034229   1st Qu.:-
0.036656
##  Median : 0.005383   Median :-0.04464   Median :-0.007284   Median :-
0.005671
##  Mean   : 0.000000   Mean   : 0.00000   Mean   : 0.000000   Mean   :
0.000000
##  3rd Qu.: 0.038076   3rd Qu.: 0.05068   3rd Qu.: 0.031248   3rd Qu.:
0.035644
##  Max.   : 0.110727   Max.   : 0.05068   Max.   : 0.170555   Max.   :
0.132044
##        tc                  ldl                  hdl
##  Min.   :-0.126781   Min.   :-0.115613   Min.   :-0.102307
##  1st Qu.:-0.034248   1st Qu.:-0.030358   1st Qu.:-0.035117
##  Median :-0.004321   Median :-0.003819   Median :-0.006584
##  Mean   : 0.000000   Mean   : 0.000000   Mean   : 0.000000
##  3rd Qu.: 0.028358   3rd Qu.: 0.029844   3rd Qu.: 0.029312
##  Max.   : 0.153914   Max.   : 0.198788   Max.   : 0.181179
##       tch                 ltg                  glu
##  Min.   :-0.076395   Min.   :-0.126097   Min.   :-0.137767
##  1st Qu.:-0.039493   1st Qu.:-0.033249   1st Qu.:-0.033179
##  Median :-0.002592   Median :-0.001948   Median :-0.001078
##  Mean   : 0.000000   Mean   : 0.000000   Mean   : 0.000000
##  3rd Qu.: 0.034309   3rd Qu.: 0.032433   3rd Qu.: 0.027917
##  Max.   : 0.185234   Max.   : 0.133599   Max.   : 0.135612
```

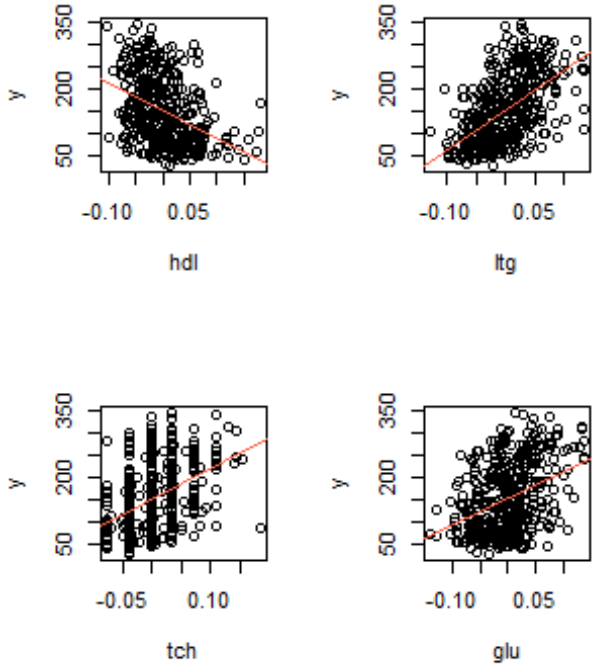
```r
par(mfcol = c(2, 3))
for (idx in 1:10)
{
  if (idx == 1) {
    xLabel = "ages"
  }
  if (idx == 2) {
    xLabel = "sex"
  }
  if (idx == 3) {
    xLabel = "bmi"
  }
  if (idx == 4) {
    xLabel = "map"
```

```
  }
  if (idx == 5) {
    xlab = "tc"
  }
  if (idx == 6) {
    xLabel = "ldl"
  }
  if (idx == 7) {
    xLabel = "hdl"
  }
  if (idx == 8) {
    xLabel = "tch"
  }
  if (idx == 9) {
    xLabel = "ltg"
  }
  if (idx == 10) {
    xLabel = "glu"
  }
  plot(x[, idx], y,  xlab = xLabel)
  abline(lm(y ~ x[, idx]), col = "tomato")
}
```




```
# Summary() Provides the Descriptive Stats of the x variable in diabetes Data set. Once this is done, let's find out the list of elements which are present in "x" using looping concept (for loop).
```

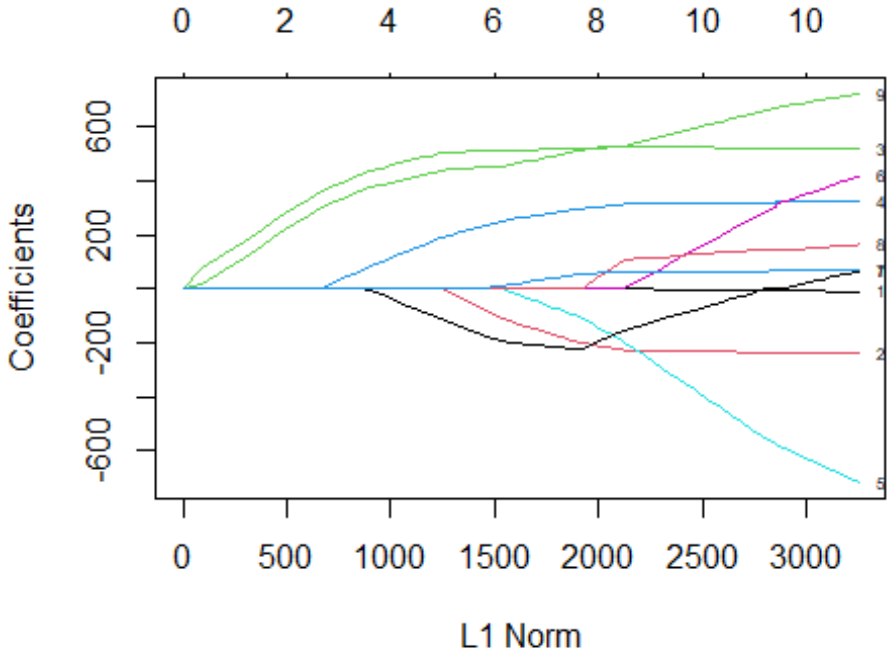
```
# We noticed 10 variables from the statistics given in the summary. Now,
let's plot Scatter plots for these variables and stack them in 2 rows and 5
columns

# To generate separate Scatter Plots with the line of best fit for all the
predictors in "x" as horizontal axis vs "y" as  vertical axis.

# In this for loop I used 'idx' as index value for "x" variable and increased
the value of 'idx' by iterating the loop 10 times.

# I have included conditional statements to allocate individual x-labels to
their respective plots. abline() is used to plot a linear regression line on
these plots.
```



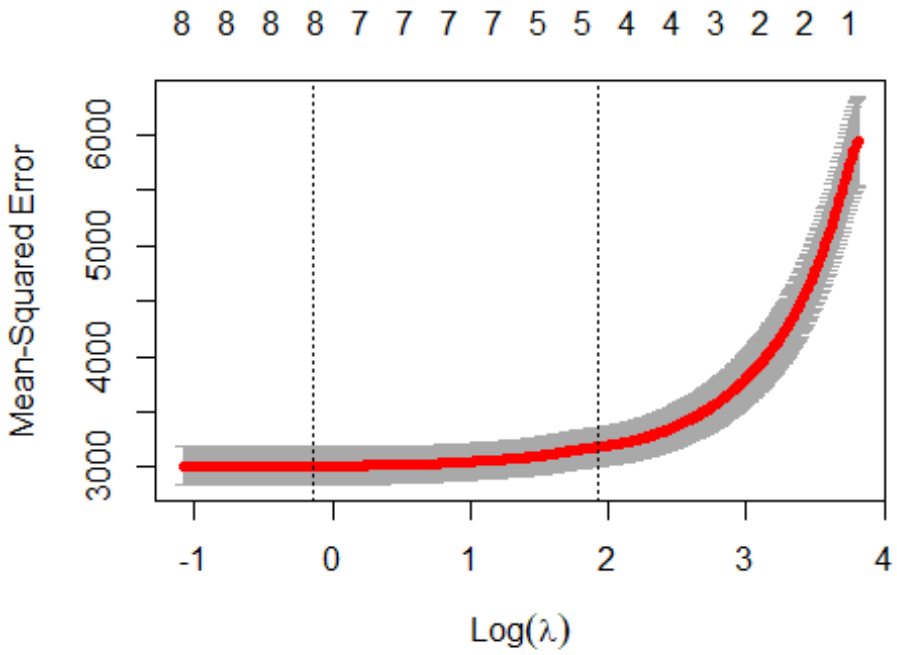

```
# Exercise 3

linReglOLS <- lm(y ~ x)
linReglOLS

##
## Call:
## lm(formula = y ~ x)
##
## Coefficients:
## (Intercept)         xage         xsex         xbmi         xmap
xtc
##      152.13       -10.01      -239.82       519.84       324.39      -
```

```
792.18
##        xldl         xhdl         xtch         xltg         xglu
##      476.75       101.04       177.06       751.28        67.63

summary(linReglOLS)

##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -155.829  -38.534   -0.227   37.806  151.355
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  152.133      2.576  59.061  < 2e-16 ***
## xage         -10.012     59.749  -0.168 0.867000
## xsex        -239.819     61.222  -3.917 0.000104 ***
## xbmi         519.840     66.534   7.813 4.30e-14 ***
## xmap         324.390     65.422   4.958 1.02e-06 ***
## xtc         -792.184    416.684  -1.901 0.057947 .
## xldl         476.746    339.035   1.406 0.160389
## xhdl         101.045    212.533   0.475 0.634721
## xtch         177.064    161.476   1.097 0.273456
## xltg         751.279    171.902   4.370 1.56e-05 ***
## xglu          67.625     65.984   1.025 0.305998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 54.15 on 431 degrees of freedom
## Multiple R-squared:  0.5177, Adjusted R-squared:  0.5066
## F-statistic: 46.27 on 10 and 431 DF,  p-value: < 2.2e-16

# In this, we need to regress "y" on the predictors in "x" using Ordinary
Least Squares(OLS). The regression model was taken between "y" and "x"

# Summary() gives us the descriptive stats and hypothesis testing values like
Standard Error, p-Value, t-Value, r-squared value, f-Statistic, Degrees of
Freedom, and etc.,

# This model is used as a baseline model to collate with the next upcoming
models

# Exercise 4

library(glmnet)

## Warning: package 'glmnet' was built under R version 4.0.3

## Loaded glmnet 4.1
```

```
modelLASSO <- glmnet(x, y, alpha = 1)
plot(modelLASSO,
     xvar = "norm",
     label = TRUE)
```




```
# LASSO regression is performed and for that to happen we use "glmnet"
package from the packages tab to install or simply use
install.packages("glmnet") command

# Now, let's load the "glmnet" in our work space to regularize the model
using LASSO and plot it using plot(). This plot indicates at which stage each
coefficients shrinks to 0. and the lines depicts the values used by various
other coefficients

# Exercise 5

crossValidFit <- cv.glmnet(
  x = x,
  y = y,
  alpha = 1,
  nlambda = 1000
)
plot(crossValidFit)
```




```
minLambda <- crossValidFit$lambda.min
minLambda
```

```
## [1] 0.8730528
```

```
# Here, Cross Validation is used to get the best value of lambda and plot the
curve using plot(). It is possible with cv.glmnet() method. nlambda signifies
the number of lambda values in sequence. In general, nlambda values must be
above 100.

# From the plot we can depict that the value of lambda increased when the
number of selected variables narrows down. This tells that higher the value
of lambda, more shrink the selection is. Now, we find the min. value of
lambda to get the best fit

# Exercise 6

estBetaMatFit <- glmnet(
  x = x,
  y = y,
  alpha = 1,
  lambda = minLambda
)
estBetaMatFit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##               s0
```

```
## age    .
## sex -201.065217
## bmi  522.602018
## map  299.095320
## tc  -111.562032
## ldl    .
## hdl -219.228019
## tch    8.497895
## ltg  516.016160
## glu   55.828353
```

```
estBetaMatFit$lambda
```

```
## [1] 0.8730528
```

```
# Here, we use the minimum lambda value again in glmnet() function to get the best fit

# There are 3 coefficients namely age, ldl, tch whose values have become 0. It's clear that these variables are not so necessary to determine the value of "y".

# Exercise 7

# Now we use a higher value of lambda that is within one standard error of the minimum to check its effect on shrinkage.

lambdaWithOneSE <- crossValidFit$lambda.1se
lambdaWithOneSE
```

```
## [1] 6.885531
```

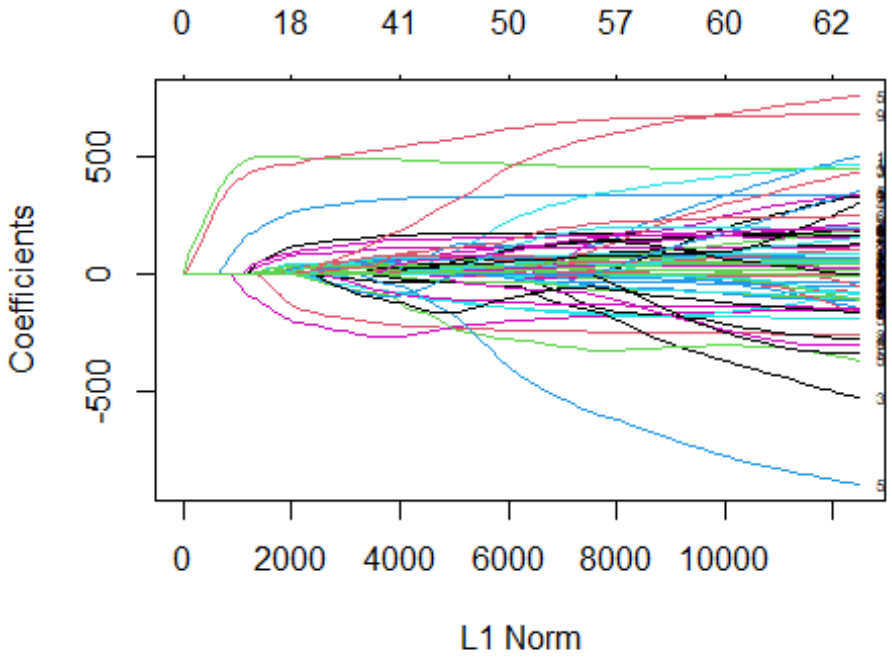
```
latestFit <- glmnet(
  x = x,
  y = y,
  alpha = 1,
  lambda = lambdaWithOneSE
)

latestFit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##            s0
## age    .
## sex    .
## bmi  500.1103
## map  182.4389
## tc     .
## ldl    .
## hdl -105.0984
## tch    .
```

```
## ltg  434.5747
## glu    .

# Here, we use the minimum lambda value again in glmnet() function to get the
best latest fit

# There are 6 coefficients namely age, sx, tc, ldl, tch, and glu whose values
have become 0. It's clear that these variables are not so necessary to
determine the value of "y". LASSO tells that only 4 variables are necessary
on which y depends. Thus the shrinkage increases

# Exercise 8

linReglOLS2 <- lm(y ~ x2)
summary(linReglOLS2)

##
## Call:
## lm(formula = y ~ x2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -158.216  -30.809   -3.857   31.348  153.946
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   152.133      2.532  60.086  < 2e-16 ***
## x2age          50.721     65.513   0.774   0.4393
## x2sex        -267.344     65.270  -4.096 5.15e-05 ***
## x2bmi         460.721     84.601   5.446 9.32e-08 ***
## x2map         342.933     72.447   4.734 3.13e-06 ***
## x2tc        -3599.542  60575.187  -0.059   0.9526
## x2ldl        3028.281  53238.699   0.057   0.9547
## x2hdl        1103.047  22636.179   0.049   0.9612
## x2tch          74.937    275.807   0.272   0.7860
## x2ltg        1828.210  19914.504   0.092   0.9269
## x2glu          62.754     70.398   0.891   0.3733
## x2age^2        67.691     69.470   0.974   0.3305
## x2bmi^2        45.849     83.288   0.550   0.5823
## x2map^2        -8.460     71.652  -0.118   0.9061
## x2tc^2       6668.449   7059.159   0.945   0.3454
## x2ldl^2      3583.174   5326.148   0.673   0.5015
## x2hdl^2      1731.821   1590.574   1.089   0.2769
## x2tch^2       773.374    606.967   1.274   0.2034
## x2ltg^2      1451.581   1730.103   0.839   0.4020
## x2glu^2       114.149     94.122   1.213   0.2260
## x2age:sex     148.678     73.407   2.025   0.0435 *
## x2age:bmi     -18.052     79.620  -0.227   0.8208
## x2age:map      18.534     76.303   0.243   0.8082
## x2age:tc     -158.891    617.109  -0.257   0.7970
```

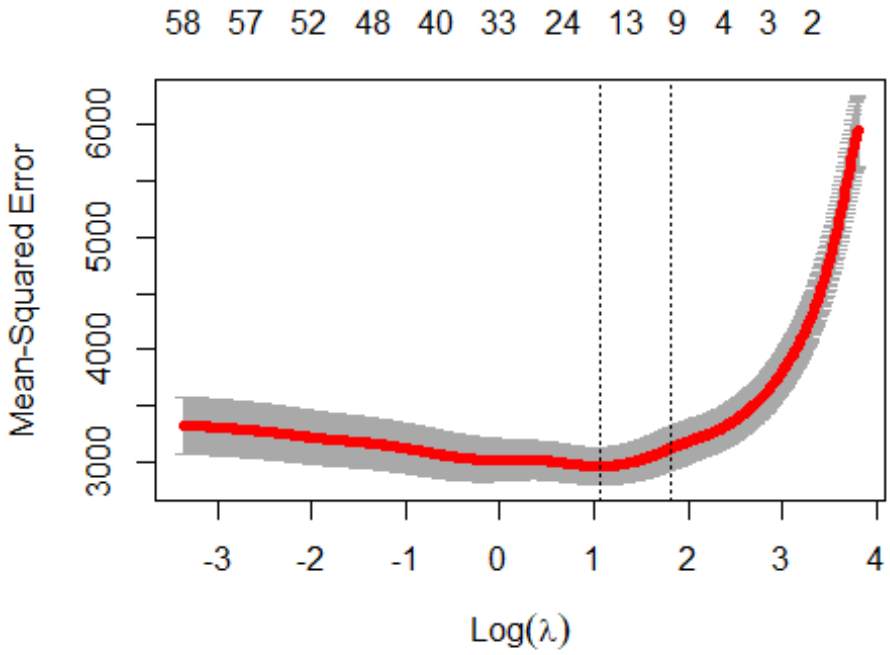
```
## x2age:ldl     -67.285    494.527  -0.136   0.8918
## x2age:hdl     209.245    280.614   0.746   0.4563
## x2age:tch     184.960    210.330   0.879   0.3798
## x2age:ltg     124.667    223.765   0.557   0.5778
## x2age:glu      62.575     80.377   0.779   0.4367
## x2sex:bmi      64.612     77.902   0.829   0.4074
## x2sex:map      88.472     74.744   1.184   0.2373
## x2sex:tc      433.598    590.709   0.734   0.4634
## x2sex:ldl    -352.823    468.951  -0.752   0.4523
## x2sex:hdl    -124.731    273.870  -0.455   0.6491
## x2sex:tch    -131.223    199.714  -0.657   0.5115
## x2sex:ltg    -118.995    226.493  -0.525   0.5996
## x2sex:glu      45.758     73.650   0.621   0.5348
## x2bmi:map     154.720     86.340   1.792   0.0739 .
## x2bmi:tc     -302.045    667.930  -0.452   0.6514
## x2bmi:ldl     241.540    561.026   0.431   0.6671
## x2bmi:hdl     121.942    329.884   0.370   0.7118
## x2bmi:tch     -33.445    230.836  -0.145   0.8849
## x2bmi:ltg     114.673    255.987   0.448   0.6544
## x2bmi:glu      23.377     91.037   0.257   0.7975
## x2map:tc      478.303    682.264   0.701   0.4837
## x2map:ldl    -326.740    574.317  -0.569   0.5697
## x2map:hdl    -187.305    309.589  -0.605   0.5455
## x2map:tch     -58.294    198.601  -0.294   0.7693
## x2map:ltg    -154.795    271.966  -0.569   0.5696
## x2map:glu    -133.476     91.314  -1.462   0.1447
## x2tc:ldl    -9313.775  11771.220  -0.791   0.4293
## x2tc:hdl    -3932.025   3816.572  -1.030   0.3036
## x2tc:tch    -2205.910   1761.843  -1.252   0.2113
## x2tc:ltg    -3801.442  13166.091  -0.289   0.7729
## x2tc:glu     -176.295    595.459  -0.296   0.7673
## x2ldl:hdl    2642.645   3165.926   0.835   0.4044
## x2ldl:tch    1206.822   1470.512   0.821   0.4123
## x2ldl:ltg    2773.697  10960.214   0.253   0.8004
## x2ldl:glu      85.626    505.102   0.170   0.8655
## x2hdl:tch    1188.406   1002.242   1.186   0.2365
## x2hdl:ltg    1467.845   4609.793   0.318   0.7503
## x2hdl:glu     217.541    296.749   0.733   0.4640
## x2tch:ltg     389.805    624.671   0.624   0.5330
## x2tch:glu     235.693    235.064   1.003   0.3167
## x2ltg:glu      83.525    264.726   0.316   0.7525
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53.23 on 377 degrees of freedom
## Multiple R-squared:  0.5924, Adjusted R-squared:  0.5233
## F-statistic: 8.563 on 64 and 377 DF,  p-value: < 2.2e-16

# In this, we need to regress "y" on the predictors in "x2" using Ordinary
Least Squares(OLS). The regression model was taken between "y" and "x"2
```

```r
# Summary() gives us the descriptive stats and hypothesis testing values like Standard Error, p-Value, t-Value, r-squared value, f-Statistic, Degrees of Freedom, and etc., From this we can see that there are more number of predictors in x2 than x.

# Exercise 9

modelLASSO2 <- glmnet(x2, y, alpha = 1)
plot(modelLASSO2, xvar = "norm", label = TRUE)
```




```r
# LASSO regression is performed and for that to happen we use "glmnet" package from the packages tab to install or simply use install.packages("glmnet") command

# Now, let's load the "glmnet" in our work space to regularize the model using LASSO and plot it using plot(). This plot are complex and the lines depicts the values used by various other coefficients

# Exercise 10

crossValidFit2 <- cv.glmnet(
  x = x2,
  y = y,
  alpha = 1,
  nlambda = 1000
)
```

```
plot(crossValidFit2)
```




```
minLambda2 <- crossValidFit2$lambda.min

estBetaMatFit2 <- glmnet(
  x = x2,
  y = y,
  alpha = 1,
  lambda = minLambda2
)
estBetaMatFit2$beta
```

```
## 64 x 1 sparse Matrix of class "dgCMatrix"
##                   s0
## age         .
## sex      -116.457385
## bmi       501.478323
## map       254.146483
## tc          .
## ldl         .
## hdl      -190.527089
## tch         .
## ltg       468.227778
## glu        19.670815
## age^2       9.718022
## bmi^2      39.571614
```

```
## map^2      .
## tc^2       .
## ldl^2      .
## hdl^2      .
## tch^2      .
## ltg^2      .
## glu^2     71.215436
## age:sex  109.359288
## age:bmi    .
## age:map   30.185352
## age:tc     .
## age:ldl    .
## age:hdl    .
## age:tch    .
## age:ltg    9.451642
## age:glu   11.123118
## sex:bmi    .
## sex:map    1.629466
## sex:tc     .
## sex:ldl    .
## sex:hdl    .
## sex:tch    .
## sex:ltg    .
## sex:glu    .
## bmi:map   86.745375
## bmi:tc     .
## bmi:ldl    .
## bmi:hdl    .
## bmi:tch    .
## bmi:ltg    .
## bmi:glu    .
## map:tc     .
## map:ldl    .
## map:hdl    .
## map:tch    .
## map:ltg    .
## map:glu    .
## tc:ldl     .
## tc:hdl     .
## tc:tch     .
## tc:ltg     .
## tc:glu     .
## ldl:hdl    .
## ldl:tch    .
## ldl:ltg    .
## ldl:glu    .
## hdl:tch    .
## hdl:ltg    .
## hdl:glu    .
## tch:ltg    .
```

```
## tch:glu    .
## ltg:glu    .
```

```
estBetaMatFit2$lambda
```

```
## [1] 2.921225
```

```
# Here, Cross Validation is used to get the best value of lambda and plot the
curve using plot(). It is possible with cv.glmnet() method. nlambda signifies
the number of lambda values in sequence. In general, n lambda values must be
above 100.

# From the plot we can depict that the value of lambda increased when the
number of selected variables narrows down. This tells that higher the value
of lambda, more shrink the selection is. Now, we find the min. value of
lambda to get the best fit

# Here, we use the minimum lambda value again in glmnet() function to get the
best fit. There are 50 coefficients whose values have become 0. It's clear
that these variables are not so necessary to determine the value of "y". With
this it shrinkage's the variables and it regularizes the model.
```