ALY 6015 M3 Report - Thota, Sunil Raj.R

# Intermediate Analytics  
# ALY 6015  
# Module 3 - Regularization Assignment  
# 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")  
  
# Loaded the below libraries into the work space

library(plyr)

library(dplyr)

library(tidyr)

require(datasets)

library(biglasso)

library(bigmemory)

library(ncvreg)

data(mtcars)  
attach(mtcars)  
View(mtcars)

head(mtcars)

## mpg cyl disp hp drat wt qsec vs am gear carb  
## Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4  
## Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4  
## Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1  
## Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1  
## Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2  
## Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

tail(mtcars)

## mpg cyl disp hp drat wt qsec vs am gear carb  
## Porsche 914-2 26.0 4 120.3 91 4.43 2.140 16.7 0 1 5 2  
## Lotus Europa 30.4 4 95.1 113 3.77 1.513 16.9 1 1 5 2  
## Ford Pantera L 15.8 8 351.0 264 4.22 3.170 14.5 0 1 5 4  
## Ferrari Dino 19.7 6 145.0 175 3.62 2.770 15.5 0 1 5 6  
## Maserati Bora 15.0 8 301.0 335 3.54 3.570 14.6 0 1 5 8  
## Volvo 142E 21.4 4 121.0 109 4.11 2.780 18.6 1 1 4 2

str(mtcars)

## 'data.frame': 32 obs. of 11 variables:  
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...  
## $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...  
## $ disp: num 160 160 108 258 360 ...  
## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...  
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...  
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...  
## $ qsec: num 16.5 17 18.6 19.4 17 ...  
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...  
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...  
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...  
## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...

summary(mtcars)

## mpg cyl disp hp   
## Min. :10.40 Min. :4.000 Min. : 71.1 Min. : 52.0   
## 1st Qu.:15.43 1st Qu.:4.000 1st Qu.:120.8 1st Qu.: 96.5   
## Median :19.20 Median :6.000 Median :196.3 Median :123.0   
## Mean :20.09 Mean :6.188 Mean :230.7 Mean :146.7   
## 3rd Qu.:22.80 3rd Qu.:8.000 3rd Qu.:326.0 3rd Qu.:180.0   
## Max. :33.90 Max. :8.000 Max. :472.0 Max. :335.0   
## drat wt qsec vs   
## Min. :2.760 Min. :1.513 Min. :14.50 Min. :0.0000   
## 1st Qu.:3.080 1st Qu.:2.581 1st Qu.:16.89 1st Qu.:0.0000   
## Median :3.695 Median :3.325 Median :17.71 Median :0.0000   
## Mean :3.597 Mean :3.217 Mean :17.85 Mean :0.4375   
## 3rd Qu.:3.920 3rd Qu.:3.610 3rd Qu.:18.90 3rd Qu.:1.0000   
## Max. :4.930 Max. :5.424 Max. :22.90 Max. :1.0000   
## am gear carb   
## Min. :0.0000 Min. :3.000 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:3.000 1st Qu.:2.000   
## Median :0.0000 Median :4.000 Median :2.000   
## Mean :0.4062 Mean :3.688 Mean :2.812   
## 3rd Qu.:1.0000 3rd Qu.:4.000 3rd Qu.:4.000   
## Max. :1.0000 Max. :5.000 Max. :8.000

# Let's perform some Regularization analysis and techniques using "mtcars" data set. This data set is readily available in the R Studio and can be loaded to the work space in R Studio. Or we can also install the packages by using install.packages("packagename") command. Once it is loaded we can use it in the code for further analysis and calculations.  
  
# Loaded the "mtcars" data into the work space. To reduce the repetitive usage of "mtcars" data set, "attach" is used to set it once throughout the work space. 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. Summary() Provides the Descriptive Stats of the x variable in diabetes Data set.

y <- mtcars$hp  
y

## [1] 110 110 93 110 175 105 245 62 95 123 123 180 180 180 205 215 230 66 52  
## [20] 65 97 150 150 245 175 66 91 113 264 175 335 109

x <- data.matrix(mtcars[, c('mpg', 'wt', 'drat', 'qsec')])  
x

## mpg wt drat qsec  
## Mazda RX4 21.0 2.620 3.90 16.46  
## Mazda RX4 Wag 21.0 2.875 3.90 17.02  
## Datsun 710 22.8 2.320 3.85 18.61  
## Hornet 4 Drive 21.4 3.215 3.08 19.44  
## Hornet Sportabout 18.7 3.440 3.15 17.02  
## Valiant 18.1 3.460 2.76 20.22  
## Duster 360 14.3 3.570 3.21 15.84  
## Merc 240D 24.4 3.190 3.69 20.00  
## Merc 230 22.8 3.150 3.92 22.90  
## Merc 280 19.2 3.440 3.92 18.30  
## Merc 280C 17.8 3.440 3.92 18.90  
## Merc 450SE 16.4 4.070 3.07 17.40  
## Merc 450SL 17.3 3.730 3.07 17.60  
## Merc 450SLC 15.2 3.780 3.07 18.00  
## Cadillac Fleetwood 10.4 5.250 2.93 17.98  
## Lincoln Continental 10.4 5.424 3.00 17.82  
## Chrysler Imperial 14.7 5.345 3.23 17.42  
## Fiat 128 32.4 2.200 4.08 19.47  
## Honda Civic 30.4 1.615 4.93 18.52  
## Toyota Corolla 33.9 1.835 4.22 19.90  
## Toyota Corona 21.5 2.465 3.70 20.01  
## Dodge Challenger 15.5 3.520 2.76 16.87  
## AMC Javelin 15.2 3.435 3.15 17.30  
## Camaro Z28 13.3 3.840 3.73 15.41  
## Pontiac Firebird 19.2 3.845 3.08 17.05  
## Fiat X1-9 27.3 1.935 4.08 18.90  
## Porsche 914-2 26.0 2.140 4.43 16.70  
## Lotus Europa 30.4 1.513 3.77 16.90  
## Ford Pantera L 15.8 3.170 4.22 14.50  
## Ferrari Dino 19.7 2.770 3.62 15.50  
## Maserati Bora 15.0 3.570 3.54 14.60  
## Volvo 142E 21.4 2.780 4.11 18.60

linReglOLS <- lm(y ~ x)  
linReglOLS

##   
## Call:  
## lm(formula = y ~ x)  
##   
## Coefficients:  
## (Intercept) xmpg xwt xdrat xqsec   
## 473.779 -2.877 26.037 4.819 -20.751

summary(linReglOLS)

##   
## Call:  
## lm(formula = y ~ x)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -48.801 -16.007 -5.482 11.614 97.338   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 473.779 105.213 4.503 0.000116 \*\*\*  
## xmpg -2.877 2.381 -1.209 0.237319   
## xwt 26.037 13.514 1.927 0.064600 .   
## xdrat 4.819 15.952 0.302 0.764910   
## xqsec -20.751 3.993 -5.197 1.79e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 32.25 on 27 degrees of freedom  
## Multiple R-squared: 0.8073, Adjusted R-squared: 0.7787   
## F-statistic: 28.27 on 4 and 27 DF, p-value: 2.647e-09

# Here "y" variable is taken as the response variable. Here "x" is assigned with a matrix of predictor variables  
  
# 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

library(glmnet)

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

## Loaded glmnet 4.1

lambdaSeq <- 10 ^ seq(2, -2, by = -.1)  
lambdaSeq

## [1] 100.00000000 79.43282347 63.09573445 50.11872336 39.81071706  
## [6] 31.62277660 25.11886432 19.95262315 15.84893192 12.58925412  
## [11] 10.00000000 7.94328235 6.30957344 5.01187234 3.98107171  
## [16] 3.16227766 2.51188643 1.99526231 1.58489319 1.25892541  
## [21] 1.00000000 0.79432823 0.63095734 0.50118723 0.39810717  
## [26] 0.31622777 0.25118864 0.19952623 0.15848932 0.12589254  
## [31] 0.10000000 0.07943282 0.06309573 0.05011872 0.03981072  
## [36] 0.03162278 0.02511886 0.01995262 0.01584893 0.01258925  
## [41] 0.01000000

ridgeFit <- glmnet(x, y, alpha = 0, lambda = lambdaSeq)  
ridgeFit

##   
## Call: glmnet(x = x, y = y, alpha = 0, lambda = lambdaSeq)   
##   
## Df %Dev Lambda  
## 1 4 64.58 100.000  
## 2 4 68.02 79.430  
## 3 4 70.91 63.100  
## 4 4 73.28 50.120  
## 5 4 75.18 39.810  
## 6 4 76.65 31.620  
## 7 4 77.78 25.120  
## 8 4 78.62 19.950  
## 9 4 79.24 15.850  
## 10 4 79.68 12.590  
## 11 4 80.00 10.000  
## 12 4 80.23 7.943  
## 13 4 80.38 6.310  
## 14 4 80.49 5.012  
## 15 4 80.57 3.981  
## 16 4 80.62 3.162  
## 17 4 80.65 2.512  
## 18 4 80.68 1.995  
## 19 4 80.69 1.585  
## 20 4 80.70 1.259  
## 21 4 80.71 1.000  
## 22 4 80.72 0.794  
## 23 4 80.72 0.631  
## 24 4 80.72 0.501  
## 25 4 80.72 0.398  
## 26 4 80.72 0.316  
## 27 4 80.72 0.251  
## 28 4 80.72 0.200  
## 29 4 80.73 0.158  
## 30 4 80.73 0.126  
## 31 4 80.73 0.100  
## 32 4 80.73 0.079  
## 33 4 80.73 0.063  
## 34 4 80.73 0.050  
## 35 4 80.73 0.040  
## 36 4 80.73 0.032  
## 37 4 80.73 0.025  
## 38 4 80.73 0.020  
## 39 4 80.73 0.016  
## 40 4 80.73 0.013  
## 41 4 80.73 0.010

summary(ridgeFit)

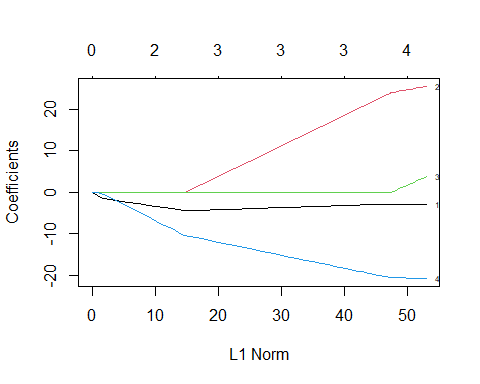
## Length Class Mode   
## a0 41 -none- numeric  
## beta 164 dgCMatrix S4   
## df 41 -none- numeric  
## dim 2 -none- numeric  
## lambda 41 -none- numeric  
## dev.ratio 41 -none- numeric  
## nulldev 1 -none- numeric  
## npasses 1 -none- numeric  
## jerr 1 -none- numeric  
## offset 1 -none- logical  
## call 5 -none- call   
## nobs 1 -none- numeric

# Setting the range of lambda values and Using glmnet() method to build the ridge regression in R. Checking the model using the Summary()

modelLASSO <- glmnet(x, y, alpha = 1)  
modelLASSO

##   
## Call: glmnet(x = x, y = y, alpha = 1)   
##   
## Df %Dev Lambda  
## 1 0 0.00 52.380  
## 2 1 10.23 47.730  
## 3 2 19.52 43.490  
## 4 2 29.45 39.620  
## 5 2 37.71 36.100  
## 6 2 44.56 32.900  
## 7 2 50.24 29.970  
## 8 2 54.97 27.310  
## 9 2 58.89 24.880  
## 10 2 62.14 22.670  
## 11 2 64.84 20.660  
## 12 2 67.09 18.820  
## 13 3 69.32 17.150  
## 14 3 71.25 15.630  
## 15 3 72.84 14.240  
## 16 3 74.17 12.970  
## 17 3 75.27 11.820  
## 18 3 76.19 10.770  
## 19 3 76.95 9.815  
## 20 3 77.58 8.943  
## 21 3 78.10 8.148  
## 22 3 78.53 7.424  
## 23 3 78.90 6.765  
## 24 3 79.19 6.164  
## 25 3 79.44 5.616  
## 26 3 79.65 5.117  
## 27 3 79.82 4.663  
## 28 3 79.96 4.249  
## 29 3 80.08 3.871  
## 30 3 80.18 3.527  
## 31 3 80.26 3.214  
## 32 3 80.33 2.928  
## 33 3 80.39 2.668  
## 34 3 80.43 2.431  
## 35 3 80.47 2.215  
## 36 3 80.50 2.018  
## 37 3 80.53 1.839  
## 38 3 80.55 1.676  
## 39 3 80.57 1.527  
## 40 3 80.59 1.391  
## 41 3 80.60 1.268  
## 42 3 80.61 1.155  
## 43 3 80.62 1.052  
## 44 3 80.62 0.959  
## 45 3 80.63 0.874  
## 46 3 80.64 0.796  
## 47 3 80.64 0.725  
## 48 4 80.65 0.661  
## 49 4 80.67 0.602  
## 50 4 80.68 0.549  
## 51 4 80.68 0.500  
## 52 4 80.69 0.456  
## 53 4 80.70 0.415  
## 54 4 80.70 0.378  
## 55 4 80.71 0.345  
## 56 4 80.71 0.314  
## 57 4 80.71 0.286  
## 58 4 80.71 0.261  
## 59 4 80.72 0.238  
## 60 4 80.72 0.216  
## 61 4 80.72 0.197  
## 62 4 80.72 0.180  
## 63 4 80.72 0.164  
## 64 4 80.72 0.149

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

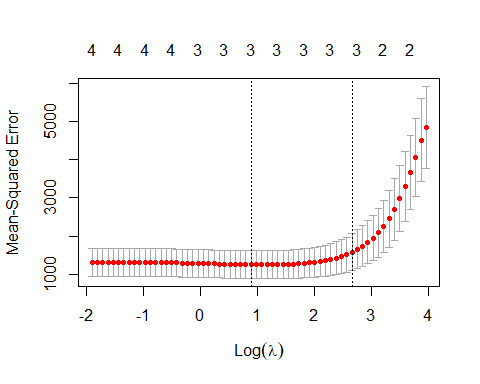
cvModel <- cv.glmnet(x, y, alpha = 1)  
cvModel

##   
## Call: cv.glmnet(x = x, y = y, alpha = 1)   
##   
## Measure: Mean-Squared Error   
##   
## Lambda Index Measure SE Nonzero  
## min 2.431 34 1250 362.4 3  
## 1se 14.239 15 1573 485.3 3

bestLambda <- cvModel$lambda.min  
bestLambda

## [1] 2.431182

plot(cvModel)



# 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.  
  
# Let's find optimal lambda value that minimizes test MSE and perform K-Fold Cross validation to find optimal lambda value. At last, let's produce the plot of test MSE by lambda value.  
  
# 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

lambdaWithOneSE <- cvModel$lambda.1se  
lambdaWithOneSE

## [1] 14.23948

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

## 4 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## mpg -4.024702  
## wt 5.766737  
## drat .   
## qsec -12.842480

# Here, we use the minimum lambda value again in glmnet() function to get the best latest fit. Now we use a higher value of lambda that is within one standard error of the minimum to check its effect on shrinkage.  
  
# There are 1 coefficients namely "drat" whose values have become 0. It's clear that this variable is not so necessary to determine the value of "y". LASSO tells that only 3 variables are necessary on which y depends. Thus the shrinkage increases.

bestModel <- glmnet(x, y, alpha = 1, lambda = bestLambda)  
coef(bestModel)

## 5 x 1 sparse Matrix of class "dgCMatrix"  
## s0  
## (Intercept) 485.152675  
## mpg -2.936266  
## wt 21.698919  
## drat .   
## qsec -19.569135

newObs <- matrix(c(21, 2.1, 3.6, 18.0), nrow = 1, ncol = 4)  
newObs

## [,1] [,2] [,3] [,4]  
## [1,] 21 2.1 3.6 18

predict(bestModel, s = bestLambda, newx = newObs)

## 1  
## [1,] 116.8144

yPred <- predict(bestModel, s = bestLambda, newx = x)  
  
sstValue <- sum((y - mean(y)) ^ 2)  
sseValue <- sum((yPred - y) ^ 2)  
  
rSquaredVal <- 1 - sseValue / sstValue  
rSquaredVal

## [1] 0.8043193

# To find the coefficients of best model, let's define a new observation and use LASSO regression model to predict response value. Use fitted best model to make predictions. Let's find SST, SSE, and R-Squared values for the new observation