ASSIGNMENT 3

Vaishnav Mule
A20516627

1 Recitation Exercises

1.1 Chapter 6

Exercise: 1

a) In backward selection we can select any term to delete at any phase, But in forward selection we can observe only the immediate pair of terms at a given step. Hence, Backward selection is used to produce lowest training RSS.

b) It is hard to surely pick an option, the model that has the best subset approach will choose the model with k predictors that is best.

c)

1. True
2. True
3. False
4. False
5. True

Exercise: 2

a) the correct option is (iii), Lasso does a better job than least squares is because of the bias-variance trade-off. Also lasso performs variable selection, which is easier to understand than other techniques like ridge regression. Just at a moderate cost increase in bias lasso approach can reduce variance when the variance of the least squares estimates is sufficiently big.

b) option (iii) is correct. Ridge regression is less adaptable than least squares, as the variance falls, and the bias rises as the coefficient nears 0.

c) option (i) is correct, as non-linear regression models are comparatively more flexible.

Exercise: 3

a) option (i) is correct as beta variables have less constraint as s increases as they can now accept values form wider range. Model now fits the training data better.

b) option (ii) us correct as though the model can initially fit data well, after a point it tends to overfit and the model becomes very flexible.

c) option (iii) is correct as more betas are included, the model becomes flexible and there will be a steady increase in variance.

d) option (iv) is correct, it is self-explanatory.

e) option (v) is correct as error term has nothing to do with the model.

Exercise: 4

a) option (iii) is correct, an increase in lambda will minimize beta squared term. Hence, overall fit will get worse for training as lambda is increased.

b) option (i) is correct, the test error will initially decrease as variance drops but the test MSE will grow if lambda increases as a result of variance increasing.

c) option (iv) is correct, several beta values are very close to zero hence the complexity of the model will decrease on the whole.

d) option (iii) is correct as model gets simpler.

e) option (v) is correct as it can’t be controlled. Error term is not dependant on the model.

Exercise: 5

a)

Q5 a) $L = (y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \beta_1 x_{21} - \beta_2 x_{22})^2$
$+ \lambda(\hat{\beta}_1^2 + \beta_2^2)$

let $x_{11} = x_{12} = x_1$ & $x_{21} = x_{22} = x_2$

$L = (y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_1)^2 + (y_2 - (\hat{\beta}_1 + \hat{\beta}_2) x_2)^2$
$+ (\lambda(\hat{\beta}_1^2 + \hat{\beta}_2^2))$

b)

b) Differentiate ① with respt to $\hat{\beta}_1$ & $\hat{\beta}_2$

$\frac{\partial L}{\partial \hat{\beta}_1} = 2(y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_1)(-x_1) + 2(y_2 - (\hat{\beta}_1 + \hat{\beta}_2) x_2)$
$x(-x^2) + 2\lambda \hat{\beta}_1 = 0$

$\Rightarrow -x_1 y_1 + (\hat{\beta}_1 + \hat{\beta}_2) x_1^2 + x_2 y_2 + (\hat{\beta}_1 + \hat{\beta}_2) x_2^2 + \lambda \hat{\beta}_1 = 0$

$\Rightarrow (x_1^2 + x_2^2 + \lambda) \hat{\beta}_1 + (x_1^2 + x_2^2) \hat{\beta}_2 = x_1 y_1 + x_2 y_2$ — ②

$\frac{\partial L}{\partial \hat{\beta}_2} = 2(y_1 - (\hat{\beta}_1 + \hat{\beta}_2) x_1)(-x_1) + 2(y_2 - (\hat{\beta}_1 + \hat{\beta}_2) x_2(-x_2)) +$
$+ 2\lambda \hat{\beta}_2 = 0$

$= -x_1 y_1 + (\hat{\beta}_1 + \hat{\beta}_2) x_1^2 - x_2 y_2 + (\hat{\beta}_1 + \hat{\beta}_2) x_2^2 + \lambda \hat{\beta}_2 = 0$

$\Rightarrow \hat{\beta}_2 (x_1^2 + x_2^2) + \hat{\beta}_2 (x_1^2 + x_2^2 + \lambda) = x_1 y_1 + x_2 y_2$ — ③

Subtract ② from ③

$\Rightarrow \hat{\beta}_1 (x_1^2 + x_2^2) - \hat{\beta}_1 (x_1^2 + x_2^2 + \lambda) + \hat{\beta}_2 (x_1^2 + x_2^2 + \lambda) -$
$\hat{\beta}_2 (x_1^2 + x_2^2) = 0$

$\Rightarrow \hat{\beta}_1 \lambda + \lambda \hat{\beta}_2 = 0$

$$\boxed{\hat{\beta}_1 = \hat{\beta}_2}$$

c) d)

c) $L = (y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_1)^2 + (y_2 - (\hat{\beta}_1 + \hat{\beta}_2)x_2)^2 + \lambda(|\hat{\beta}_1| + |\hat{\beta}_2|)$

d) consider optimization problem,

$(y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_1)^2 + (y_2 - (\hat{\beta}_1 + \hat{\beta}_2)x_2))^2$ subject to $|\hat{\beta}_1| + |\hat{\beta}_2| \leq s$

using given constraints in given question, we minimize

$$2(y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_1)^2 \geq 0$$

possible solution is $y_1 - (\hat{\beta}_1 + \hat{\beta}_2)x_1 = 0$

$$\Rightarrow \hat{\beta}_1 + \hat{\beta}_2 = y_1/x_1$$

we have a set of solutions:-

$\Rightarrow (\hat{\beta}_1, \hat{\beta}_2) = \bar{\beta}_1 + \bar{\beta}_2 = s$ with $\hat{\beta}_1, \hat{\beta}_2 \geq 0$

$\Rightarrow (\hat{\beta}_1 + \hat{\beta}_2) = -3$ with $\hat{\beta}_1, \hat{\beta}_2 < 0$

1.2 Chapter 7

Exercise: 2

a) Lambda approaching infinity indicates that the first term is not relevant. When m=0, g(0)=g and therefore g' = 0

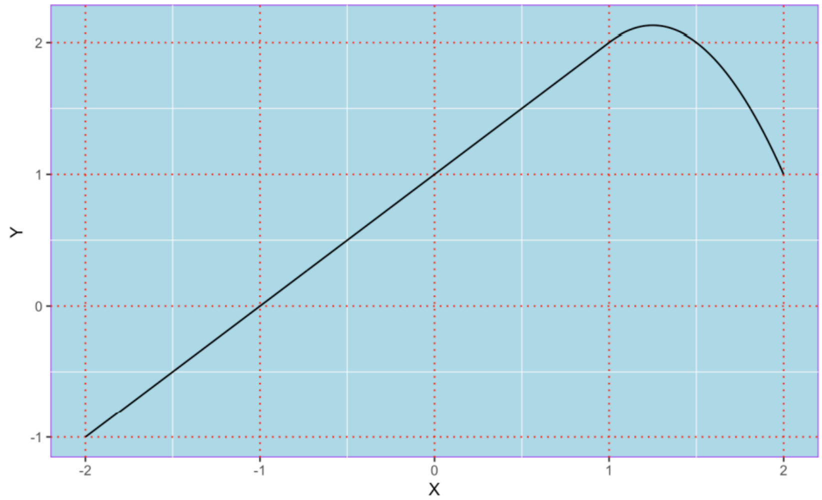
b) Lambda approaching infinity indicates that the first term will not be relevant. When m = 0, g(1) = g '. Horizontal line would denote function ĝ which minimizes first derivative.

c) Lambda approaching infinity indicates that the first term will not be relevant. When m = 0, g(2) = g''. A linear line would denote function ĝ which minuses the second derivative.

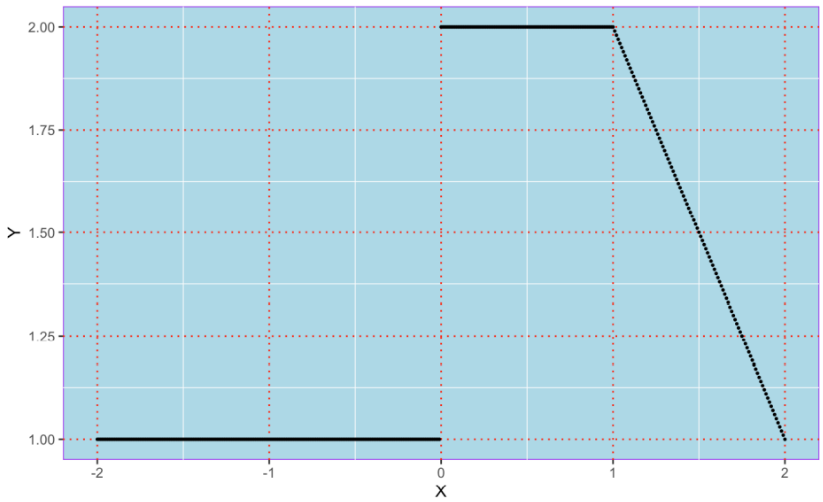
d) Lambda approaching infinity indicates that the first term will not be relevant. When m = 0, g(3) = g'''. The functions ĝ that minuses the second derivative is a quadratic.

e) Lambda = 0 indicates that the second term won't be relevant. Equation then becomes std least squares eq.

Exercise: 3



Exercise: 4



Exercise: 5

a) The g_hat_2 will have a smaller RSS as it will be more flexible, due to the order of penalty term.

b) the g_hat_1 will have smaller test RSS if g_hat_2 overfits because of the higher flexibility.

c) g_hat_1 and g_hat_2 will equal each other when lambda = 0.

2 Practicum Problems

2.1 Problem 1

Vaishnavi

2023-10-05

```
install.packages("doMC", repos="http://R-Forge.R-project.org")

## Installing package into 'C:/Users/91951/AppData/Local/R/win-library/4.2
'
## (as 'lib' is unspecified)

## installing the source package 'doMC'
```

loading data :

```
mtcars_df = data.frame(mtcars)
head(mtcars_df)

##                    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
```

split data to train set and test set :

```
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

mtcars_df_split = createDataPartition(mtcars_df$mpg, p = 0.80, list=FALSE)
mtcars_train_set = mtcars_df[mtcars_df_split,]
mtcars_test_set = mtcars_df[-mtcars_df_split,]
```

fitting model :

```
linear_model_mtcars = lm(mpg ~ cyl + disp + hp + drat + wt + qsec + vs + a
m + gear + carb, data = mtcars_train_set)

summary(linear_model_mtcars)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + disp + hp + drat + wt + qsec + vs +
##     am + gear + carb, data = mtcars_train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0498 -1.3743  0.0014  0.8413  3.4257
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -35.424905  25.724916  -1.377   0.1864
## cyl           0.637367   1.050158   0.607   0.5519
## disp          0.030854   0.016979   1.817   0.0869 .
## hp           -0.007901   0.020962  -0.377   0.7109
## drat          2.751038   1.626190   1.692   0.1089
## wt           -5.089883   1.775482  -2.867   0.0107 *
## qsec          2.264852   1.029643   2.200   0.0420 *
## vs           -1.192677   2.226921  -0.536   0.5992
## am            1.244666   1.867636   0.666   0.5141
## gear          3.707569   1.687494   2.197   0.0422 *
## carb         -0.552617   0.875073  -0.632   0.5361
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.196 on 17 degrees of freedom
## Multiple R-squared:  0.9041, Adjusted R-squared:  0.8477
## F-statistic: 16.03 on 10 and 17 DF,  p-value: 9.257e-07
```

Based on the t-stat, wt seems relevant. Furthermore other features like qsec, hp, disp also have big t-stat value.

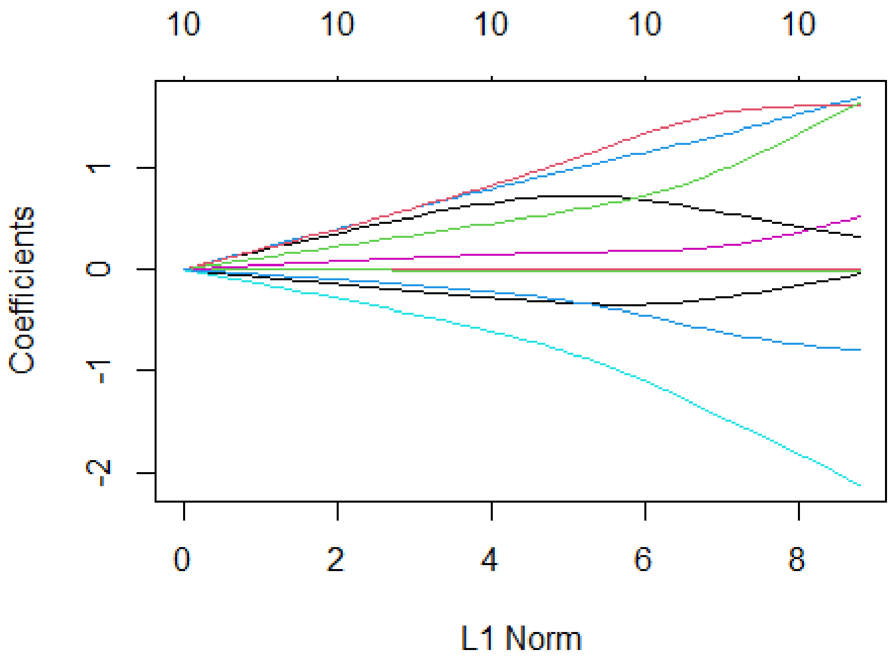
-> wt = -4.74715 -> qsec = 2.14918 -> hp = -0.01274 -> disp = 0.02338

```
coef(linear_model_mtcars)

##   (Intercept)           cyl          disp            hp          drat
## -35.424905463   0.637367233   0.030853587  -0.007901284   2.751037674
##            wt          qsec            vs            am          gear
##  -5.089882786   2.264852144  -1.192676997   1.244665660   3.707568754
##          carb
##  -0.552616915
```

performing ridge regression :

```
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-6

library(doMC)

## Loading required package: foreach

## Loading required package: iterators
```

```
## Loading required package: parallel

y = mtcars_train_set$mpg
x = data.matrix(mtcars_train_set[,c('cyl','disp','hp','drat','wt','qsec','
vs','am','gear','carb')])
glm = glmnet(x,y, alpha= 0)
summary(glm)

##           Length Class     Mode
## a0         100   -none-    numeric
## beta      1000   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         4   -none-    call
## nobs         1   -none-    numeric

plot(glm)
```



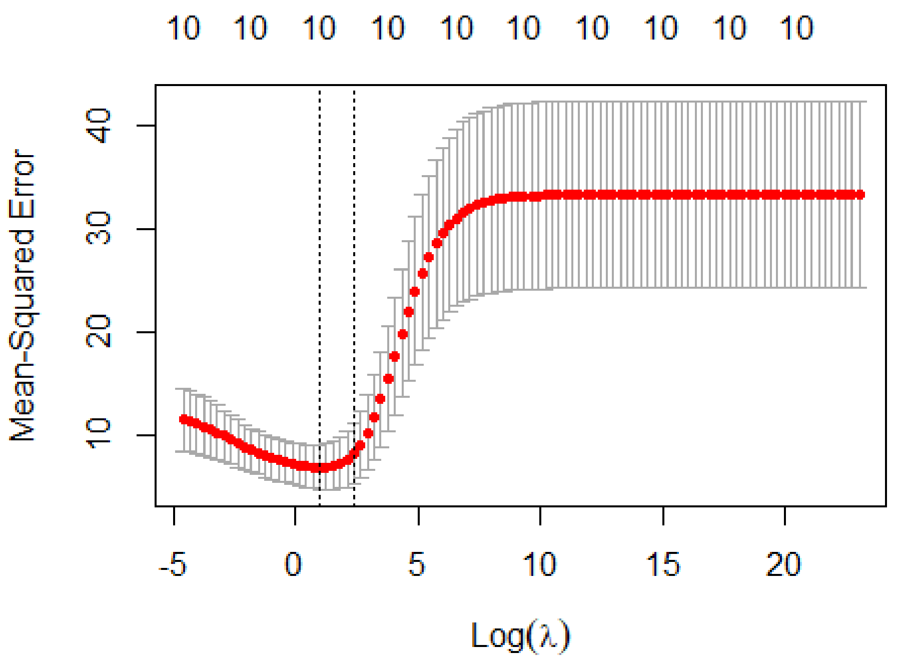

```
registerDoMC(core=2)
grid_lambda  = 10^seq(10,-2,length=100)
mtcars_cross_val_glm = cv.glmnet(x,y, alpha= 0, lambda = grid_lambda, para
llel = TRUE, grouped=FALSE)
summary(mtcars_cross_val_glm)

##            Length Class  Mode
## lambda     100    -none- numeric
```

```
## cvm         100    -none- numeric
## cvsd        100    -none- numeric
## cvup        100    -none- numeric
## cvlo        100    -none- numeric
## nzero       100    -none- numeric
## call          7    -none- call
## name          1    -none- character
## glmnet.fit   12    elnet  list
## lambda.min    1    -none- numeric
## lambda.1se    1    -none- numeric
## index         2    -none- numeric
```

```
best_lambda_val = mtcars_cross_val_glm$lambda.min
best_lambda_val
```

```
## [1] 2.656088
```

```
plot(mtcars_cross_val_glm)
```




```
mtcars_new_model = glmnet(x, y, alpha=0, lambda=best_lambda_val)
summary(mtcars_new_model)
```

```
##           Length Class     Mode
## a0         1     -none-    numeric
## beta      10     dgCMatrix S4
## df         1     -none-    numeric
## dim        2     -none-    numeric
## lambda     1     -none-    numeric
## dev.ratio  1     -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
```

```
y_test = mtcars_test_set$mpg
x_test = data.matrix(mtcars_test_set[,c('cyl','disp','hp','drat','wt','qsec','vs','am','gear','carb')])
y_mtcars_predictions = predict(mtcars_new_model, s=best_lambda_val, newx=x_test)
```

results Sample Test :

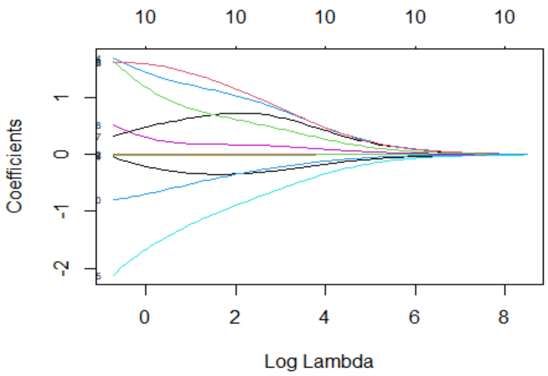
```
SST_mtcars = sum((y_test -mean(y_test))^2)
cat("SST : ",SST_mtcars)

## SST :  270.91

SSE_mtcars = sum((y_mtcars_predictions -mean(y_test))^2)
cat("\nSSE : ", SSE_mtcars)

##
## SSE :  109.7211

mtcars_R_sq = 1 - (SSE_mtcars/SST_mtcars)
cat("\nR -square : ",mtcars_R_sq)

##
## R -square :  0.5949908
```

Difference in coeffcient values of models can be seen below :

```
coef(mtcars_new_model)

## 11 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 18.109297624
## cyl         -0.326445530
## disp        -0.003237479
## hp          -0.010148915
## drat         1.227817848
## wt          -1.242302378
## qsec         0.192948571
## vs           0.636884201
## am           1.441677967
## gear         0.816995641
## carb        -0.525317638

coef(linear_model_mtcars)

##   (Intercept)          cyl         disp           hp         drat
## -35.424905463  0.637367233  0.030853587 -0.007901284  2.751037674
##            wt         qsec           vs           am         gear
##  -5.089882786  2.264852144 -1.192676997  1.244665660  3.707568754
##          carb
##  -0.552616915
```

It can be seen from new coefficent values that Ridge Regression did not do variable section (coff = 0). It performed ‘Shrinkage’.

```
plot(glm, xvar="lambda", label=TRUE)
```



2.2 Problem 2

Vaishnavi

2023-10-05

Load Dataset :

```
swiss_df = data.frame(swiss)
head(swiss)

##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice
```

```
swiss_df_split = createDataPartition(swiss_df$Fertility, p = 0.80, list=FA
LSE)
swiss_df_train_set = swiss_df[swiss_df_split,]
swiss_df_test_set =  swiss_df[-swiss_df_split,]
```

Fit a linear model :

```
swiss_linearmodel = lm(Fertility ~ Agriculture + Examination + Education +
Catholic + Infant.Mortality, data = swiss_df_train_set)
summary(swiss_linearmodel)

##
## Call:
## lm(formula = Fertility ~ Agriculture + Examination + Education +
##     Catholic + Infant.Mortality, data = swiss_df_train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.0815  -4.2016  -0.1039   4.5836  13.4483
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      72.54340   12.66230   5.729 2.14e-06 ***
## Agriculture      -0.16029    0.07854  -2.041  0.04934 *
## Examination      -0.42089    0.32496  -1.295  0.20425
## Education        -0.72175    0.23333  -3.093  0.00401 **
## Catholic          0.09250    0.04403   2.101  0.04340 *
## Infant.Mortality  0.85870    0.42425   2.024  0.05112 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.416 on 33 degrees of freedom
## Multiple R-squared:  0.6973, Adjusted R-squared:  0.6514
## F-statistic:  15.2 on 5 and 33 DF,  p-value: 9.292e-08
```

Based on t-values and p-values, probable relevant features are Infant. Mortality and education

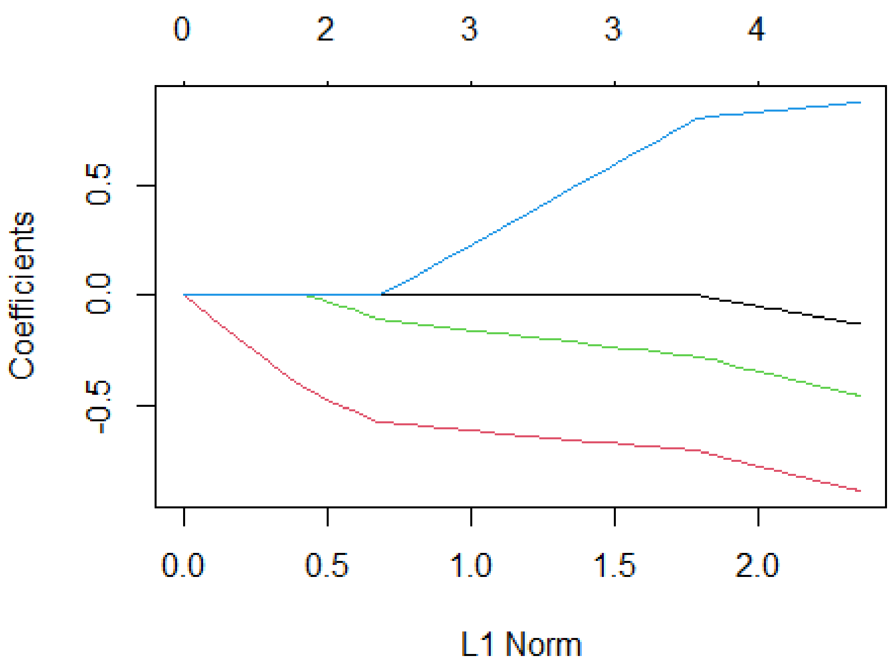
coefficient (education) = -0.83356522 coefficient (Infant.Mortality) = 1.20966304

```
coef(swiss_linearmodel)

##      (Intercept)      Agriculture      Examination        Education
##      72.54340033      -0.16028654      -0.42088540      -0.72174890
##         Catholic Infant.Mortality
##       0.09249552       0.85870229
```

Perform Lasso Regression

```
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-6

library(doMC)
```

```
## Loading required package: foreach

## Loading required package: iterators

## Loading required package: parallel

swiss_x_train = data.matrix(swiss_df_train_set[,c("Agriculture", "Examinat
ion", "Education", "Infant.Mortality")])
swiss_y_train = swiss_df_train_set$Fertility
swiss_lassomodel = glmnet(swiss_x_train, swiss_y_train, alpha=1)
summary(swiss_lassomodel)

##           Length Class     Mode
## a0         65    -none-    numeric
## beta      260    dgCMatrix S4
## df         65    -none-    numeric
## dim         2    -none-    numeric
## lambda     65    -none-    numeric
## dev.ratio  65    -none-    numeric
## nulldev     1    -none-    numeric
## npasses     1    -none-    numeric
## jerr        1    -none-    numeric
## offset      1    -none-    logical
## call        4    -none-    call
## nobs        1    -none-    numeric

plot(swiss_lassomodel)
```




```
registerDoMC(cores=2)
lambda_vec = 10^seq(10,-3,length=100)
swiss_crossval_model = cv.glmnet(swiss_x_train, swiss_y_train, alpha=1, la
```

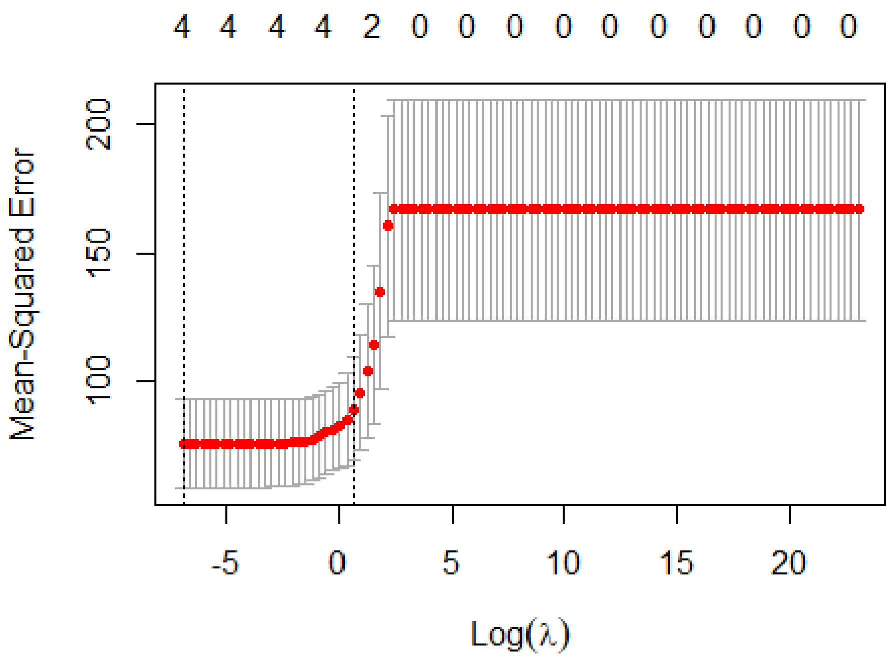
```
mbda=lambda_vec, parallel=TRUE, grouped=FALSE)
summary(swiss_crossval_model)
```

```
##             Length Class  Mode
## lambda      100    -none- numeric
## cvm         100    -none- numeric
## cvsd        100    -none- numeric
## cvup        100    -none- numeric
## cvlo        100    -none- numeric
## nzero       100    -none- numeric
## call          7    -none- call
## name          1    -none- character
## glmnet.fit   12    elnet  list
## lambda.min    1    -none- numeric
## lambda.1se    1    -none- numeric
## index         2    -none- numeric
```

```
swiss_bestlambda_val = swiss_crossval_model$lambda.min
swiss_bestlambda_val
```

```
## [1] 0.001
```

```
plot(swiss_crossval_model)
```




```
swiss_new_model = glmnet(swiss_x_train, swiss_y_train, alpha=1, lambda=swi
ss_bestlambda_val)
summary(swiss_new_model)
```

```
##           Length Class     Mode
## a0        1      -none-    numeric
## beta      4      dgCMatrix S4
## df        1      -none-    numeric
```

```
## dim        2       -none-     numeric
## lambda     1       -none-     numeric
## dev.ratio 1       -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
```

```
coef(swiss_new_model)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                          s0
## (Intercept)      79.5714706
## Agriculture      -0.1336910
## Examination      -0.8928483
## Education        -0.4621702
## Infant.Mortality  0.8808018
```

```
coef(swiss_linearmodel)
```

```
##      (Intercept)      Agriculture      Examination        Education
##      72.54340033      -0.16028654      -0.42088540      -0.72174890
##         Catholic Infant.Mortality
##       0.09249552       0.85870229
```

```
swiss_x_test = data.matrix(swiss_df_test_set[,c("Agriculture", "Examinatio
n", "Education", "Infant.Mortality")])
swiss_y_test = swiss_df_test_set$Fertility
swiss_y_predictions = predict(swiss_new_model, newx=swiss_x_test, lambda=s
wiss_bestlambda_val)
```

```
SST_swiss = sum((swiss_y_test -mean(swiss_y_test))^2)
cat("SST : ",SST_swiss)
```

```
## SST :  1175.359
```

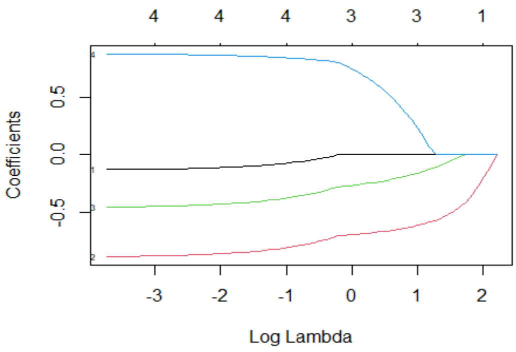
```
SSE_swiss = sum((swiss_y_predictions -mean(swiss_y_test))^2)
cat("\nSSE : ", SSE_swiss)
```

```
##
## SSE :  282.8698
```

```
swiss_R_sq = 1 - (SSE_swiss/SST_swiss)
cat("\nR -square : ",swiss_R_sq)
```

```
##
## R -square :  0.7593332
```

```
plot(swiss_lassomodel, xvar="lambda", label=TRUE)
```



Model selection not performed by Lasso as none of the coefficients are = 0

2.2 Problem 3

Vaishnavi

2023-10-05

Load the Dataset :

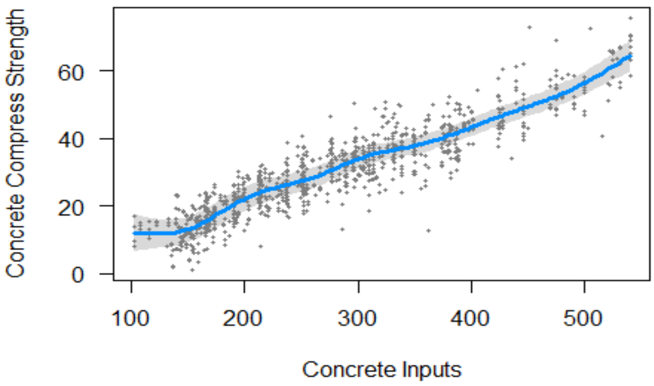
```
library(readxl)
concrete_df = read_excel('Concrete_Data.xls')
names(concrete_df) = c('cement','blastfurnaceslag','fly_ash','water','supe
rplasticizer','coarse_aggregate', 'fine_aggregate', 'age', 'concretecompre
ssstrength')
head(concrete_df)

## # A tibble: 6 × 9
##   cement blastfurnaceslag fly_ash water superpla…¹ coars…² fine_…³   ag
e concr…⁴
##    <dbl>            <dbl>   <dbl> <dbl>      <dbl>   <dbl>   <dbl> <dbl
>   <dbl>
## 1   540                0       0   162        2.5    1040     676     2
8    80.0
## 2   540                0       0   162        2.5    1055     676     2
8    61.9
## 3   332.             142.      0   228        0       932     594    27
```

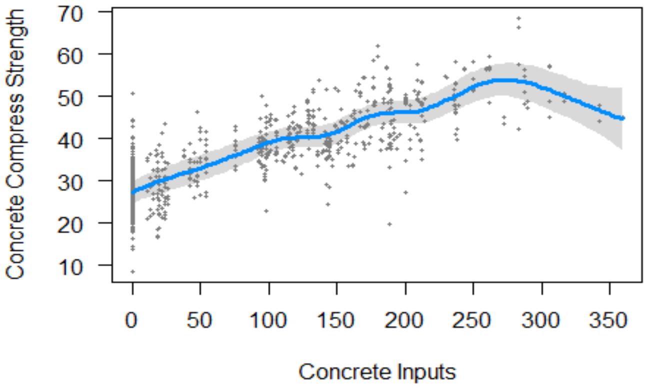
```
0    40.3
## 4   332.              142.        0    228         0       932      594     36
5    41.1
## 5   199.              132.        0    192         0       978.     826.    36
0    44.3
## 6   266               114         0    228         0       932      670      9
0    47.0
## # … with abbreviated variable names ¹superplasticizer, ²coarse_aggregat
e,
## #   ³fine_aggregate, ⁴concretecompressstrength
```

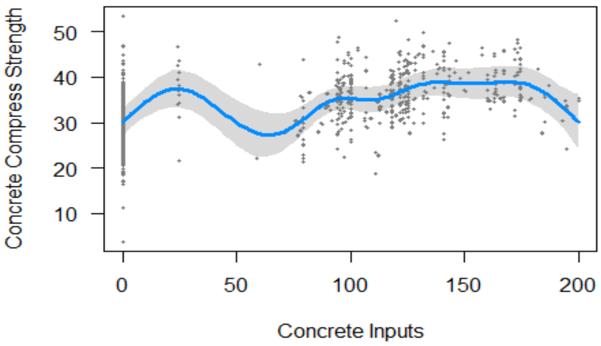
Create GAM model :

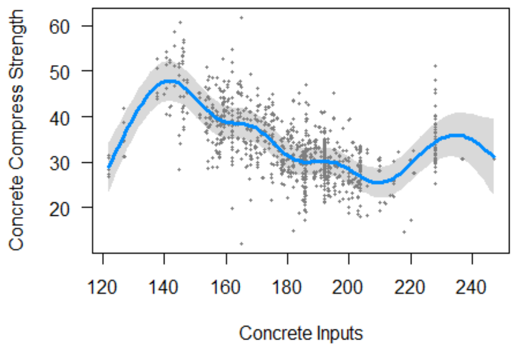
```
library(mgcv)

## Loading required package: nlme

## This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.

library(nlme)
concrete_gam_model = gam(concretecompressstrength ~ cement + blastfurnaces
lag + fly_ash + water + superplasticizer + coarse_aggregate + fine_aggrega
te + age, data= concrete_df)
summary(concrete_gam_model)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## concretecompressstrength ~ cement + blastfurnaceslag + fly_ash +
##     water + superplasticizer + coarse_aggregate + fine_aggregate +
##     age
##
## Parametric coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -23.163756  26.588421  -0.871 0.383851
## cement             0.119785   0.008489  14.110  < 2e-16 ***
## blastfurnaceslag   0.103847   0.010136  10.245  < 2e-16 ***
## fly_ash            0.087943   0.012585   6.988 5.03e-12 ***
## water             -0.150298   0.040179  -3.741 0.000194 ***
## superplasticizer   0.290687   0.093460   3.110 0.001921 **
## coarse_aggregate   0.018030   0.009394   1.919 0.055227 .
## fine_aggregate     0.020154   0.010703   1.883 0.059968 .
## age                0.114226   0.005427  21.046  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.612   Deviance explained = 61.5%
## GCV = 109.11  Scale est. = 108.16    n = 1030

concrete_gam_model = gam(concretecompressstrength ~ s(cement) + s(blastfur
naceslag) + s(fly_ash) + s(water) + s(superplasticizer) + s(coarse_aggrega
```

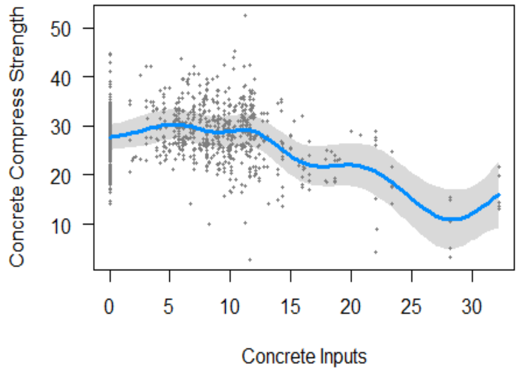
```
te) + s(fine_aggregate) + s(age), data= concrete_df)
summary(concrete_gam_model)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## concretecompressstrength ~ s(cement) + s(blastfurnaceslag) +
##     s(fly_ash) + s(water) + s(superplasticizer) + s(coarse_aggregate) +
##     s(fine_aggregate) + s(age)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.8178     0.1675   213.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                        edf Ref.df       F  p-value
## s(cement)            8.223  8.830  48.690  < 2e-16 ***
## s(blastfurnaceslag) 8.114  8.757  25.041  < 2e-16 ***
## s(fly_ash)           8.256  8.817   9.354  < 2e-16 ***
## s(water)             8.742  8.973  26.116  < 2e-16 ***
## s(superplasticizer) 8.039  8.743  10.845  < 2e-16 ***
## s(coarse_aggregate) 7.904  8.673   3.403 0.000593 ***
## s(fine_aggregate)   8.618  8.951  18.435  < 2e-16 ***
## s(age)               8.559  8.900 365.119  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.896   Deviance explained = 90.3%
## GCV = 30.914  Scale est. = 28.89     n = 1030

library(visreg)
visreg(concrete_gam_model, xlab="Concrete Inputs", ylab = "Concrete Compre
ss Strength")
```

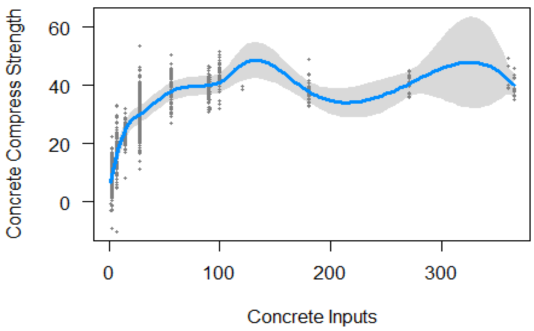

Concrete Compress Strength
0
20
40
60
100
200
300
400
500
Concrete Inputs

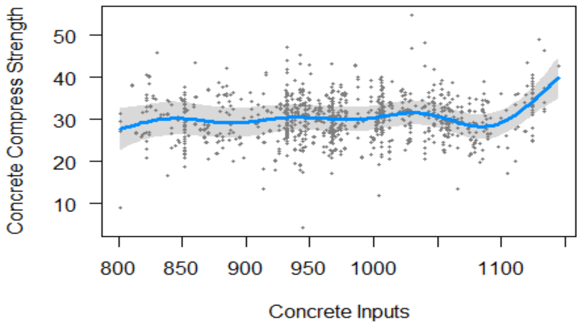

Concrete Compress Strength
10
20
30
40
50
60
70
0
50
100
150
200
250
300
350
Concrete Inputs

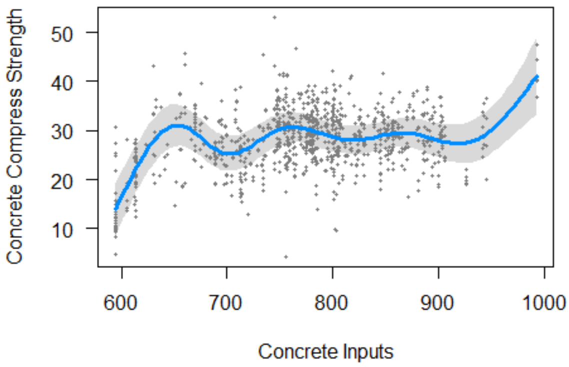

Concrete Compress Strength
10
20
30
40
50
0
50
100
150
200
Concrete Inputs











Although there are certain data points that are not very close to our confidence interval, it is seen that there are predictors that are linearly associated and some non-linearly linked is also quite well fitted.