Practical Work 2 - Regularised Regression Methods

Adib Habbou - Alae Khidour

24-10-2022

Data Import

```
diabetes_data <- read.table(file = "diabetes.txt",header = TRUE)
dim(diabetes_data)
```

```
[1] 442  11
```

```
head(diabetes_data, 10)
```

```
   AGE SEX  BMI  BP  S1    S2 S3   S4     S5 S6   Y
1   59   2 32.1 101 157  93.2 38 4.00 4.8598 87 151
2   48   1 21.6  87 183 103.2 70 3.00 3.8918 69  75
3   72   2 30.5  93 156  93.6 41 4.00 4.6728 85 141
4   24   1 25.3  84 198 131.4 40 5.00 4.8903 89 206
5   50   1 23.0 101 192 125.4 52 4.00 4.2905 80 135
6   23   1 22.6  89 139  64.8 61 2.00 4.1897 68  97
7   36   2 22.0  90 160  99.6 50 3.00 3.9512 82 138
8   66   2 26.2 114 255 185.0 56 4.55 4.2485 92  63
9   60   2 32.1  83 179 119.4 42 4.00 4.4773 94 110
10  29   1 30.0  85 180  93.4 43 4.00 5.3845 88 310
```

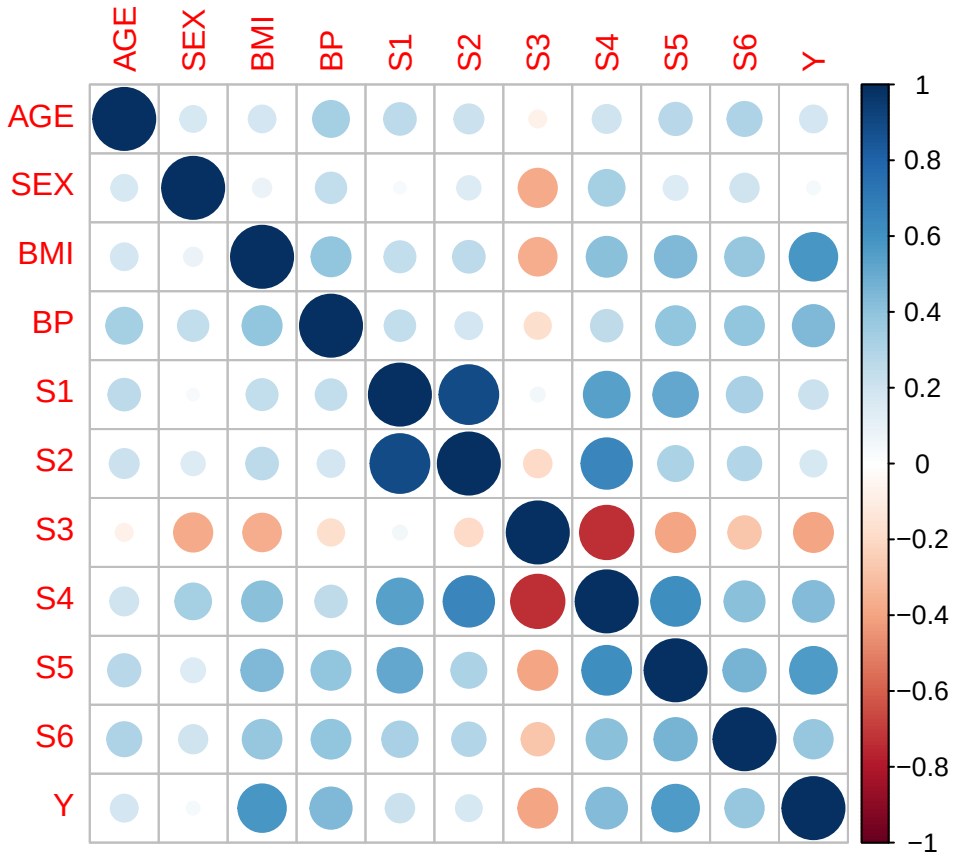
```
tail(diabetes_data, 10)
```

```
    AGE SEX  BMI     BP  S1    S2 S3   S4     S5  S6   Y
433  51   1 31.5  93.00 231 144.0 49 4.70 5.2523 117 173
434  41   1 20.8  86.00 223 128.2 83 3.00 4.0775  89  72
435  53   1 26.5  97.00 193 122.4 58 3.00 4.1431  99  49
436  45   1 24.2  83.00 177 118.4 45 4.00 4.2195  82  64
437  33   1 19.5  80.00 171  85.4 75 2.00 3.9703  80  48
438  60   2 28.2 112.00 185 113.8 42 4.00 4.9836  93 178
439  47   2 24.9  75.00 225 166.0 42 5.00 4.4427 102 104
440  60   2 24.9  99.67 162 106.6 43 3.77 4.1271  95 132
441  36   1 30.0  95.00 201 125.2 42 4.79 5.1299  85 220
442  36   1 19.6  71.00 250 133.2 97 3.00 4.5951  92  57
```

Study of correlation

First, let's look at the correlation between our variables by plotting the matrix:

```
corr <- cor(diabetes_data)
corrplot(corr, method = "circle")
```



The correlation plot provides us with a lot of information such as:

- S1 and S2 are highly positively correlated;
- S3 and S4 are highly negatively correlated.

We need to keep in mind the correlation between our variables.

The potential colinearity between variables can have an impact on the Standard Error.

More than that, it means that the co-variable signifacitivity test is useless.

Multiple Regression

```
diabetes_model <- lm(formula = Y ~ ., data = diabetes_data)
summary(diabetes_model)
```

```
Call:
lm(formula = Y ~ ., data = diabetes_data)

Residuals:
     Min       1Q   Median       3Q      Max
-155.827  -38.536   -0.228   37.806  151.353

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -334.56714   67.45462  -4.960 1.02e-06 ***
AGE           -0.03636    0.21704  -0.168 0.867031
SEX          -22.85965    5.83582  -3.917 0.000104 ***
BMI            5.60296    0.71711   7.813 4.30e-14 ***
BP             1.11681    0.22524   4.958 1.02e-06 ***
S1            -1.09000    0.57333  -1.901 0.057948 .
S2             0.74645    0.53083   1.406 0.160390
S3             0.37200    0.78246   0.475 0.634723
S4             6.53383    5.95864   1.097 0.273459
S5            68.48312   15.66972   4.370 1.56e-05 ***
S6             0.28012    0.27331   1.025 0.305990
---
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
```

As we have seen in the previous practical work, the multiple regression model doesn't give us satisfying results. To have a better a model we need to do some variable selection.

In order to do that we have two main ways:

- Greedy Methods such as forward, backward and stepwise;
- Maximization of penalized log-likelihood using AIC, BIC...

Let's try both of the methods and look at the results.

Greedy Methods

Backward Regerssion

```
diabetes_backward <- step(diabetes_model, direction = "backward")
```

```
Start:  AIC=3539.64
Y ~ AGE + SEX + BMI + BP + S1 + S2 + S3 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- AGE   1        82 1264068 3537.7
- S3    1       663 1264649 3537.9
- S6    1      3080 1267066 3538.7
- S4    1      3526 1267512 3538.9
<none>              1263986 3539.6
- S2    1      5799 1269785 3539.7
- S1    1     10600 1274586 3541.3
- SEX   1     44999 1308984 3553.1
- S5    1     56016 1320001 3556.8
- BP    1     72100 1336086 3562.2
- BMI   1    179033 1443019 3596.2

Step:  AIC=3537.67
Y ~ SEX + BMI + BP + S1 + S2 + S3 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- S3    1       646 1264715 3535.9
- S6    1      3001 1267069 3536.7
- S4    1      3543 1267611 3536.9
<none>              1264068 3537.7
- S2    1      5751 1269820 3537.7
- S1    1     10569 1274637 3539.4
- SEX   1     45830 1309898 3551.4
- S5    1     55964 1320032 3554.8
- BP    1     73847 1337915 3560.8
- BMI   1    179084 1443152 3594.2

Step:  AIC=3535.9
Y ~ SEX + BMI + BP + S1 + S2 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- S6    1      3093 1267808 3535.0
- S4    1      3247 1267961 3535.0
<none>              1264715 3535.9
- S2    1      7505 1272219 3536.5
- S1    1     26839 1291554 3543.2
- SEX   1     46381 1311096 3549.8
- BP    1     73533 1338248 3558.9
- S5    1     97508 1362223 3566.7
- BMI   1    178542 1443256 3592.3

Step:  AIC=3534.98
```

```
Y ~ SEX + BMI + BP + S1 + S2 + S4 + S5

       Df Sum of Sq     RSS    AIC
- S4    1      3686 1271494 3534.3
<none>              1267808 3535.0
- S2    1      7472 1275280 3535.6
- S1    1     26378 1294186 3542.1
- SEX   1     44684 1312492 3548.3
- BP    1     82152 1349960 3560.7
- S5    1    102520 1370328 3567.3
- BMI   1    189976 1457784 3594.7

Step:  AIC=3534.26
Y ~ SEX + BMI + BP + S1 + S2 + S5

       Df Sum of Sq     RSS    AIC
<none>              1271494 3534.3
- S2    1     39377 1310871 3545.7
- SEX   1     41856 1313350 3546.6
- S1    1     65236 1336730 3554.4
- BP    1     79625 1351119 3559.1
- BMI   1    190592 1462086 3594.0
- S5    1    294092 1565586 3624.2
```

Backward Regression Summary

```
summary(diabetes_backward)
```

```
Call:
lm(formula = Y ~ SEX + BMI + BP + S1 + S2 + S5, data = diabetes_data)

Residuals:
     Min       1Q   Median       3Q      Max
-158.275  -39.476   -2.065   37.219  148.690

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -313.7666    25.3848 -12.360  < 2e-16 ***
SEX          -21.5910     5.7056  -3.784 0.000176 ***
BMI            5.7111     0.7073   8.075 6.69e-15 ***
BP             1.1266     0.2158   5.219 2.79e-07 ***
S1            -1.0429     0.2208  -4.724 3.12e-06 ***
S2             0.8433     0.2298   3.670 0.000272 ***
S5            73.3065     7.3083  10.031  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.06 on 435 degrees of freedom
Multiple R-squared:  0.5149,    Adjusted R-squared:  0.5082
F-statistic: 76.95 on 6 and 435 DF,  p-value: < 2.2e-16
```

Forward Regression

```
diabetes_forward <- step(lm(Y ~ 1, data = diabetes_data),
                         list(upper = diabetes_model),
                         direction = "forward")
```

```
Start:  AIC=3841.99
Y ~ 1

       Df Sum of Sq     RSS    AIC
+ BMI   1    901427 1719582 3657.7
+ S5    1    839308 1781701 3673.4
+ BP    1    510851 2110158 3748.2
+ S4    1    485646 2135363 3753.4
+ S3    1    408507 2212502 3769.1
+ S6    1    383437 2237572 3774.1
+ S1    1    117824 2503186 3823.7
+ AGE   1     92527 2528482 3828.1
+ S2    1     79403 2541607 3830.4
<none>              2621009 3842.0
+ SEX   1      4860 2616149 3843.2

Step:  AIC=3657.7
Y ~ BMI

       Df Sum of Sq     RSS    AIC
+ S5    1    302888 1416694 3574.1
+ BP    1    136477 1583105 3623.1
+ S4    1    111511 1608071 3630.1
+ S3    1     97767 1621815 3633.8
+ S6    1     73738 1645844 3640.3
+ AGE   1     17087 1702495 3655.3
+ S1    1     12008 1707574 3656.6
<none>              1719582 3657.7
+ S2    1      1228 1718354 3659.4
+ SEX   1       197 1719385 3659.6

Step:  AIC=3574.06
Y ~ BMI + S5

       Df Sum of Sq     RSS    AIC
+ BP    1     53985 1362709 3558.9
+ S1    1     27624 1389070 3567.4
+ S3    1     26914 1389781 3567.6
+ S2    1      9256 1407438 3573.2
+ SEX   1      6881 1409813 3573.9
+ S6    1      6801 1409893 3573.9
<none>              1416694 3574.1
+ S4    1      2376 1414318 3575.3
+ AGE   1       176 1416518 3576.0

Step:  AIC=3558.88
Y ~ BMI + S5 + BP
```

```
       Df Sum of Sq     RSS    AIC
+ S1    1   31277.3 1331431 3550.6
+ S3    1   29921.2 1332787 3551.1
+ SEX   1   17532.1 1345177 3555.2
+ S2    1   10809.8 1351899 3557.4
<none>              1362709 3558.9
+ S4    1    3218.7 1359490 3559.8
+ AGE   1    2106.4 1360602 3560.2
+ S6    1    1240.1 1361469 3560.5

Step:  AIC=3550.62
Y ~ BMI + S5 + BP + S1

       Df Sum of Sq     RSS    AIC
+ SEX   1   20560.5 1310871 3545.7
+ S2    1   18080.9 1313350 3546.6
+ S4    1   15188.0 1316243 3547.6
+ S3    1   14360.4 1317071 3547.8
<none>              1331431 3550.6
+ S6    1    2898.8 1328533 3551.7
+ AGE   1     472.0 1330959 3552.5

Step:  AIC=3545.74
Y ~ BMI + S5 + BP + S1 + SEX

       Df Sum of Sq     RSS    AIC
+ S2    1     39377 1271494 3534.3
+ S4    1     35591 1275280 3535.6
+ S3    1     35001 1275870 3535.8
<none>              1310871 3545.7
+ S6    1      5288 1305583 3546.0
+ AGE   1        49 1310822 3547.7

Step:  AIC=3534.26
Y ~ BMI + S5 + BP + S1 + SEX + S2

       Df Sum of Sq     RSS    AIC
<none>              1271494 3534.3
+ S4    1    3686.2 1267808 3535.0
+ S6    1    3532.6 1267961 3535.0
+ S3    1     394.8 1271099 3536.1
+ AGE   1      10.9 1271483 3536.3
```

Forward Regerssion Summary

```
summary(diabetes_forward)
```

```
Call:
lm(formula = Y ~ BMI + S5 + BP + S1 + SEX + S2, data = diabetes_data)

Residuals:
     Min       1Q   Median       3Q      Max
-158.275  -39.476   -2.065   37.219  148.690

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -313.7666    25.3848 -12.360  < 2e-16 ***
BMI            5.7111     0.7073   8.075 6.69e-15 ***
S5            73.3065     7.3083  10.031  < 2e-16 ***
BP             1.1266     0.2158   5.219 2.79e-07 ***
S1            -1.0429     0.2208  -4.724 3.12e-06 ***
SEX          -21.5910     5.7056  -3.784 0.000176 ***
S2             0.8433     0.2298   3.670 0.000272 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.06 on 435 degrees of freedom
Multiple R-squared:  0.5149,    Adjusted R-squared:  0.5082
F-statistic: 76.95 on 6 and 435 DF,  p-value: < 2.2e-16
```

Stepwise Regression

```
diabetes_both <- step(diabetes_model, direction = "both")
```

```
Start:  AIC=3539.64
Y ~ AGE + SEX + BMI + BP + S1 + S2 + S3 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- AGE   1        82 1264068 3537.7
- S3    1       663 1264649 3537.9
- S6    1      3080 1267066 3538.7
- S4    1      3526 1267512 3538.9
<none>              1263986 3539.6
- S2    1      5799 1269785 3539.7
- S1    1     10600 1274586 3541.3
- SEX   1     44999 1308984 3553.1
- S5    1     56016 1320001 3556.8
- BP    1     72100 1336086 3562.2
- BMI   1    179033 1443019 3596.2

Step:  AIC=3537.67
Y ~ SEX + BMI + BP + S1 + S2 + S3 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- S3    1       646 1264715 3535.9
- S6    1      3001 1267069 3536.7
- S4    1      3543 1267611 3536.9
<none>              1264068 3537.7
- S2    1      5751 1269820 3537.7
- S1    1     10569 1274637 3539.4
+ AGE   1        82 1263986 3539.6
- SEX   1     45830 1309898 3551.4
- S5    1     55964 1320032 3554.8
- BP    1     73847 1337915 3560.8
- BMI   1    179084 1443152 3594.2

Step:  AIC=3535.9
Y ~ SEX + BMI + BP + S1 + S2 + S4 + S5 + S6

       Df Sum of Sq     RSS    AIC
- S6    1      3093 1267808 3535.0
- S4    1      3247 1267961 3535.0
<none>              1264715 3535.9
- S2    1      7505 1272219 3536.5
+ S3    1       646 1264068 3537.7
+ AGE   1        66 1264649 3537.9
- S1    1     26839 1291554 3543.2
- SEX   1     46381 1311096 3549.8
- BP    1     73533 1338248 3558.9
- S5    1     97508 1362223 3566.7
- BMI   1    178542 1443256 3592.3

Step:  AIC=3534.98
```

```
Y ~ SEX + BMI + BP + S1 + S2 + S4 + S5

       Df Sum of Sq     RSS    AIC
- S4    1      3686 1271494 3534.3
<none>              1267808 3535.0
- S2    1      7472 1275280 3535.6
+ S6    1      3093 1264715 3535.9
+ S3    1       738 1267069 3536.7
+ AGE   1         0 1267807 3537.0
- S1    1     26378 1294186 3542.1
- SEX   1     44684 1312492 3548.3
- BP    1     82152 1349960 3560.7
- S5    1    102520 1370328 3567.3
- BMI   1    189976 1457784 3594.7

Step:  AIC=3534.26
Y ~ SEX + BMI + BP + S1 + S2 + S5

       Df Sum of Sq     RSS    AIC
<none>              1271494 3534.3
+ S4    1      3686 1267808 3535.0
+ S6    1      3533 1267961 3535.0
+ S3    1       395 1271099 3536.1
+ AGE   1        11 1271483 3536.3
- S2    1     39377 1310871 3545.7
- SEX   1     41856 1313350 3546.6
- S1    1     65236 1336730 3554.4
- BP    1     79625 1351119 3559.1
- BMI   1    190592 1462086 3594.0
- S5    1    294092 1565586 3624.2
```

Stepwise Regression Summary

```
summary(diabetes_both)
```

```
Call:
lm(formula = Y ~ SEX + BMI + BP + S1 + S2 + S5, data = diabetes_data)

Residuals:
     Min       1Q   Median       3Q      Max
-158.275  -39.476   -2.065   37.219  148.690

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -313.7666    25.3848 -12.360  < 2e-16 ***
SEX          -21.5910     5.7056  -3.784 0.000176 ***
BMI            5.7111     0.7073   8.075 6.69e-15 ***
BP             1.1266     0.2158   5.219 2.79e-07 ***
S1            -1.0429     0.2208  -4.724 3.12e-06 ***
S2             0.8433     0.2298   3.670 0.000272 ***
S5            73.3065     7.3083  10.031  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.06 on 435 degrees of freedom
Multiple R-squared:  0.5149,    Adjusted R-squared:  0.5082
F-statistic: 76.95 on 6 and 435 DF,  p-value: < 2.2e-16
```

Interpretation of Greedy Methods

We can observe that the 3 methods give us the same model which contains the following variables:

- SEX;
- BMI;
- BP;
- S1;
- S2;
- S5.

Each time, we obtain the same value for the AIC criteria which equals 3534.26. We could choose another criteria for our model selection, such as BIC or C_p. It could influence our model because the formula we will minimize changes. This formula will depend differently on the number of co-variables p. So the choice of criteria will always depend on the business constraint behind. More than that, we can customize our criteria as we want by choosing the value of λ in the function *step*.

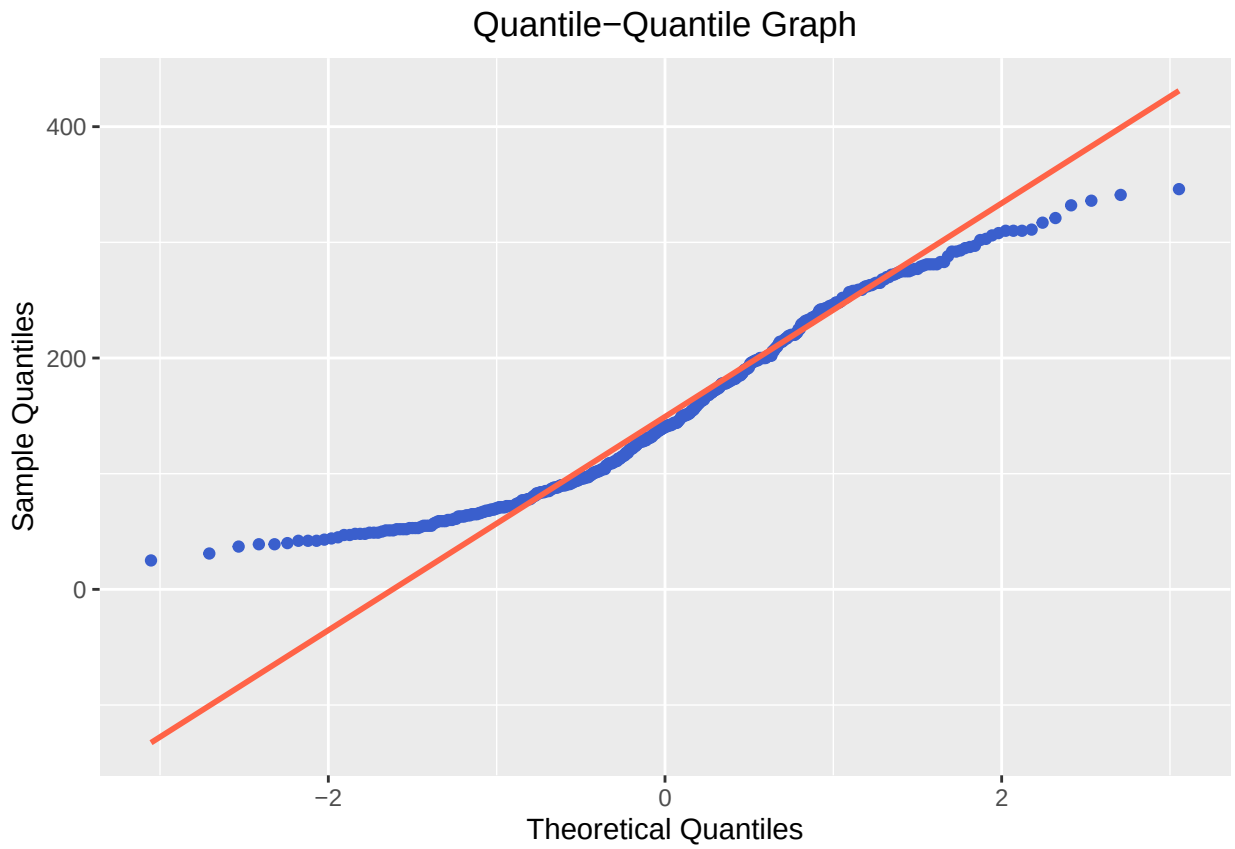
Our final model will be the following:

```
diabetes_reg <- lm(formula(diabetes_both), data = diabetes_data)
summary(diabetes_reg)
```

```
Call:
lm(formula = formula(diabetes_both), data = diabetes_data)

Residuals:
     Min       1Q   Median       3Q      Max
-158.275  -39.476   -2.065   37.219  148.690

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -313.7666    25.3848 -12.360  < 2e-16 ***
SEX          -21.5910     5.7056  -3.784 0.000176 ***
BMI            5.7111     0.7073   8.075 6.69e-15 ***
BP             1.1266     0.2158   5.219 2.79e-07 ***
S1            -1.0429     0.2208  -4.724 3.12e-06 ***
S2             0.8433     0.2298   3.670 0.000272 ***
S5            73.3065     7.3083  10.031  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.06 on 435 degrees of freedom
Multiple R-squared:  0.5149,    Adjusted R-squared:  0.5082
F-statistic: 76.95 on 6 and 435 DF,  p-value: < 2.2e-16
```

Study of Quantiles

```
ggplot(data = diabetes_data, mapping = aes(sample = Y)) +
  stat_qq(size = 1.5, color = "royalblue3") +
  stat_qq_line(size = 1, color = "tomato1") +
  labs(title = "Quantile-Quantile Graph",
       x = "Theoretical Quantiles", y = "Sample Quantiles") +
  theme(plot.title = element_text(hjust = 0.5))
```



The quantiles are values dividing a probability distribution into equal intervals, with every interval having the same fraction of the total population.

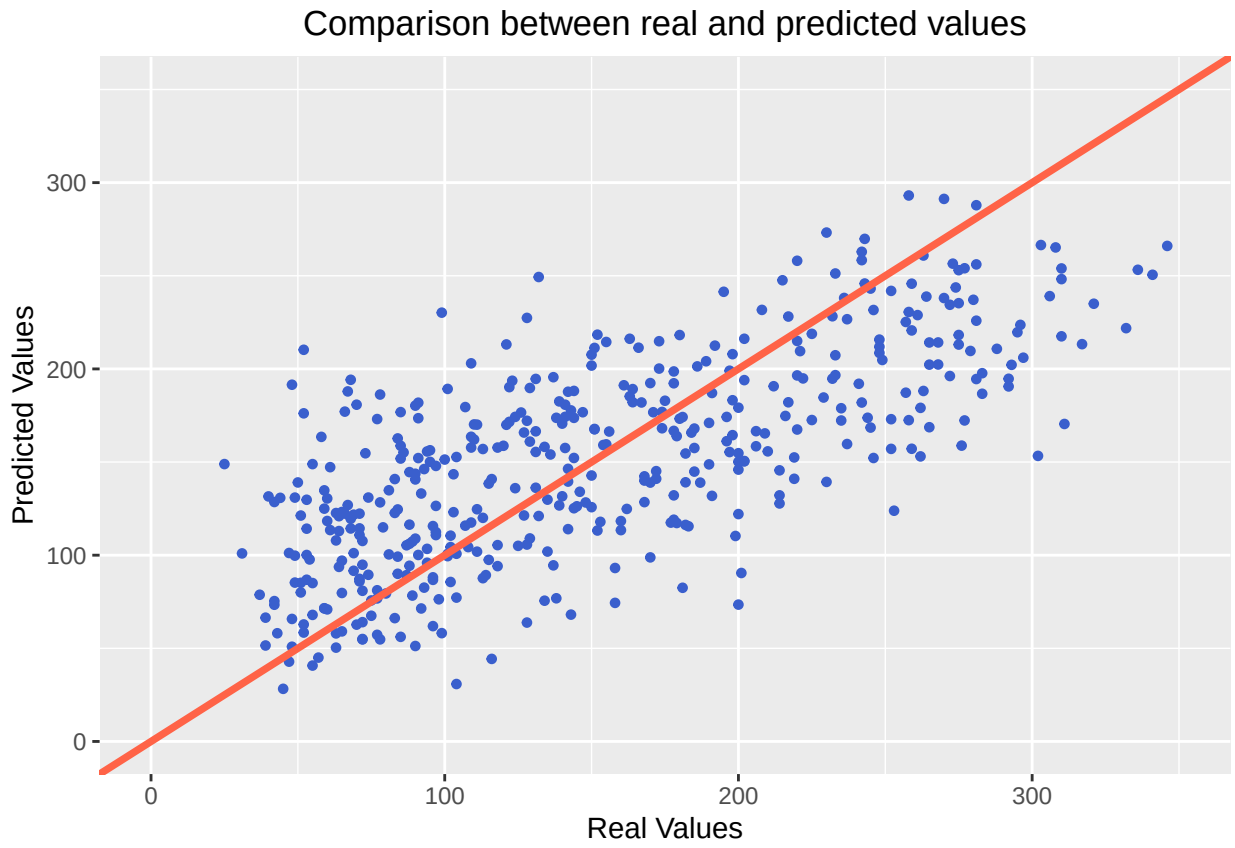
The purpose of the Quantile-Quantile plot is to show if two data sets come from the same distribution. Plotting the first data set's quantiles along the x-axis and plotting the second data set's quantiles along the y-axis is how the plot is constructed. When looking at the QQ plot, we see the points match up along a straight line which shows that the quantiles match or not.

In our case, we are comparing two probability distributions by plotting their quantiles against each other. They fit the y = x line so we can assume that the two distributions are for values between 1 and -1. So, the linear model is the right model to use in this situation.

However, for extreme values the points are a little bit far from the first bisector which means that the distributions may not be as similar as we assume, especially for the extreme values.

Study of Predictions

```
bivariate_data <- as.data.frame(cbind(diabetes_data$Y, predict(diabetes_backward)))
ggplot(data = bivariate_data, mapping = aes(x = V1, y = V2)) +
  geom_point(size = 1.2, color = "royalblue3") +
  geom_abline(size = 1.3, color = "tomato1") +
  labs(title = "Comparison between real and predicted values",
       x = "Real Values", y = "Predicted Values") +
  xlim(0, 350) + ylim(0, 350) +
  theme(plot.title = element_text(hjust = 0.5))
```

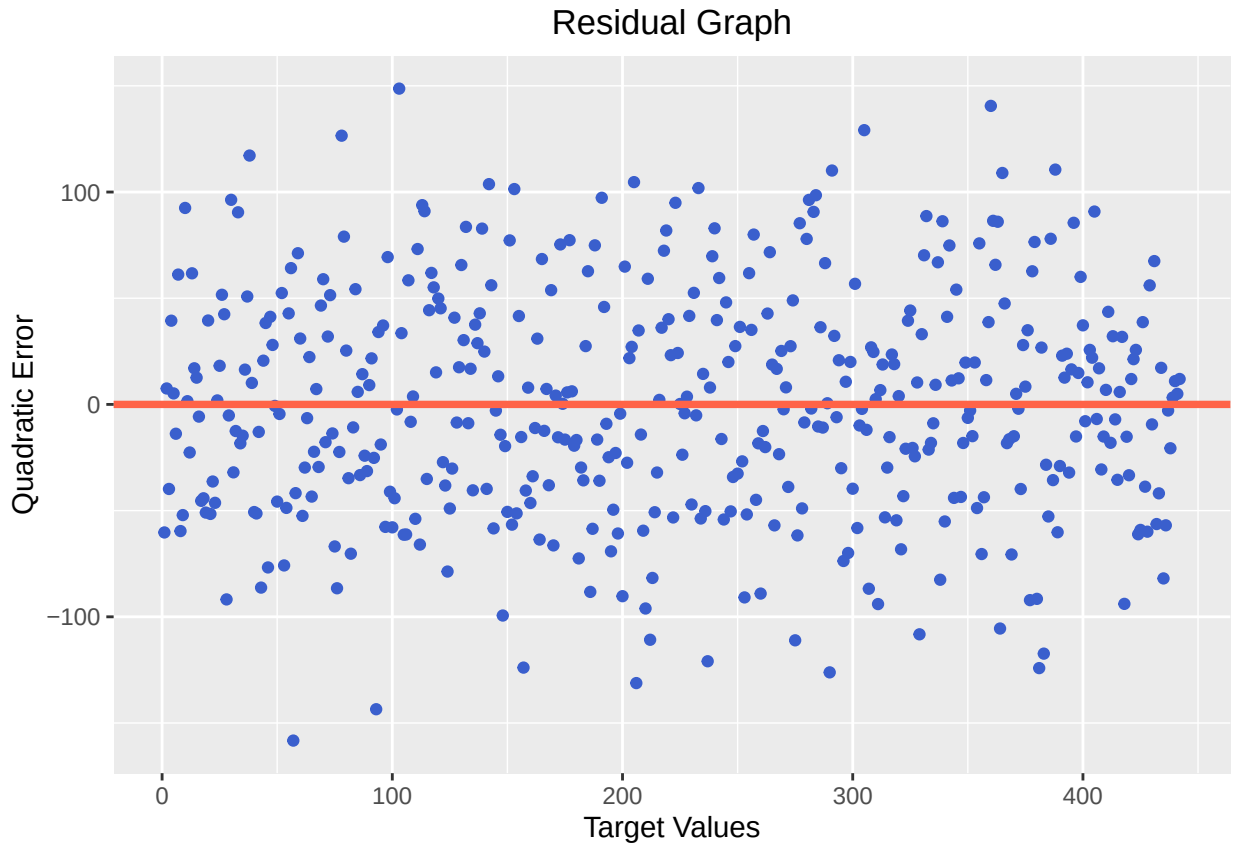


Plotting the values predicted by our model against the real values of our data set provides us information about how good are the predictions of our model.

By looking at the above figure, we can tell that the predicted values are close to the real ones because they are all stacked around the red line within a certain surface which corresponds to the error.

Study of Residuals

```
residuals <- as.data.frame(diabetes_reg$residuals)
ggplot(data = residuals,
       mapping = aes(x = seq(1, nrow(residuals)), y = diabetes_reg$residuals)) +
  geom_point(size = 1.5, color = "royalblue3") +
  geom_hline(yintercept = 0, size = 1.3, color = "tomato1") +
  labs(title = "Residual Graph", x = "Target Values", y = "Quadratic Error") +
  theme(plot.title = element_text(hjust = 0.5))
```



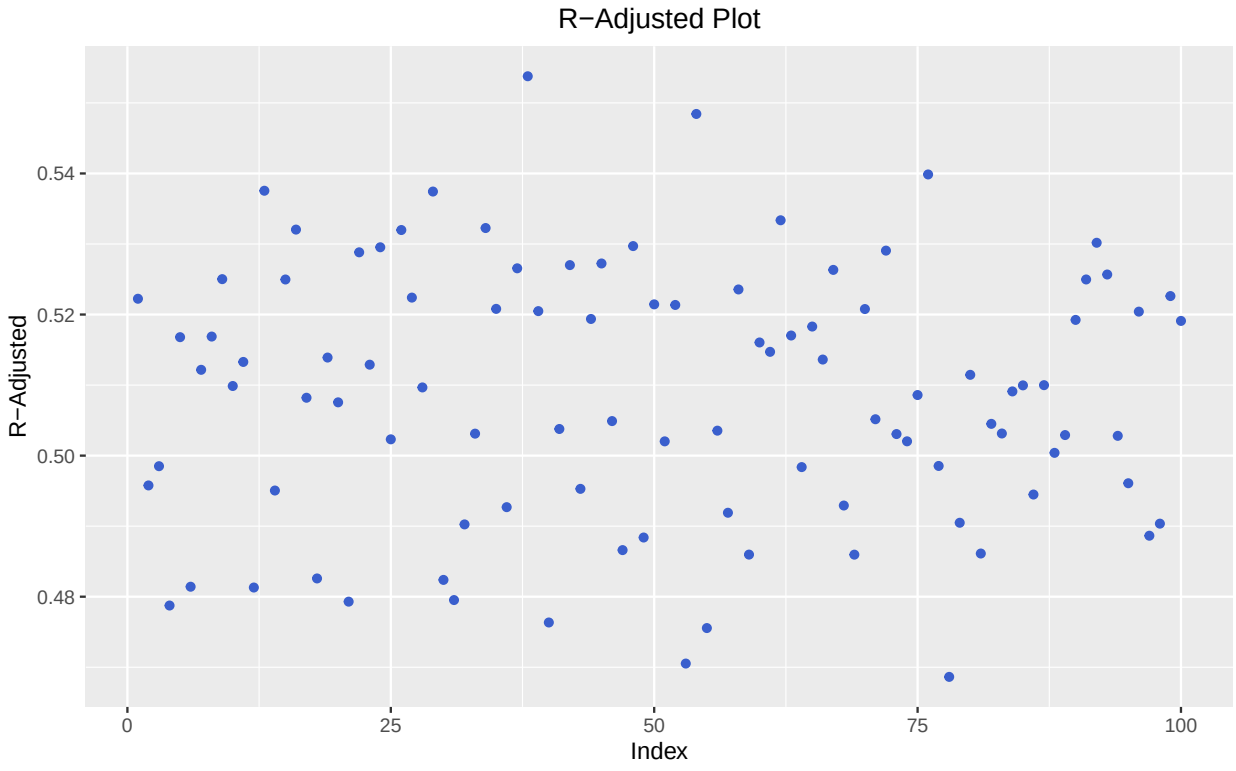
If the residuals are randomly scattered around the residual = 0, it means that a linear model approximates the data points well without favoring certain inputs. In such a case, we conclude that a linear model is appropriate. By looking at the plot above, we can tell that the distribution of points is random, so there is no more information to capture from the residuals.

Random Partitionning

```
R_Squared_Adj <- RMSD_Train <- RMSD_Test <- vector(length = 100)
```

```
for (i in 1:100)
{
  # train and test split
  sample <- sample(c(TRUE, FALSE), nrow(diabetes_data), replace = TRUE, prob = c(0.75, 0.25))
  Train <- diabetes_data[sample, ]
  Test <- diabetes_data[!sample, ]
  # model training
  model <- lm(formula(diabetes_both), data = Train)
  # compute R Adjusted
  R_Squared_Adj[i] <- summary(model)["adj.r.squared"]$adj.r.squared
  # compute RMSD Train
  Sum_Of_Square_Train <- sum((Train$Y - predict(model, newdata = Train))^2)
  RMSD_Train[i] <- sqrt(Sum_Of_Square_Train / length(Train$Y))
  # compute RMSD Test
  Sum_Of_Square_Test <- sum((Test$Y - predict(model, newdata = Test))^2)
  RMSD_Test[i] <- sqrt(Sum_Of_Square_Test / length(Test$Y))
}
```

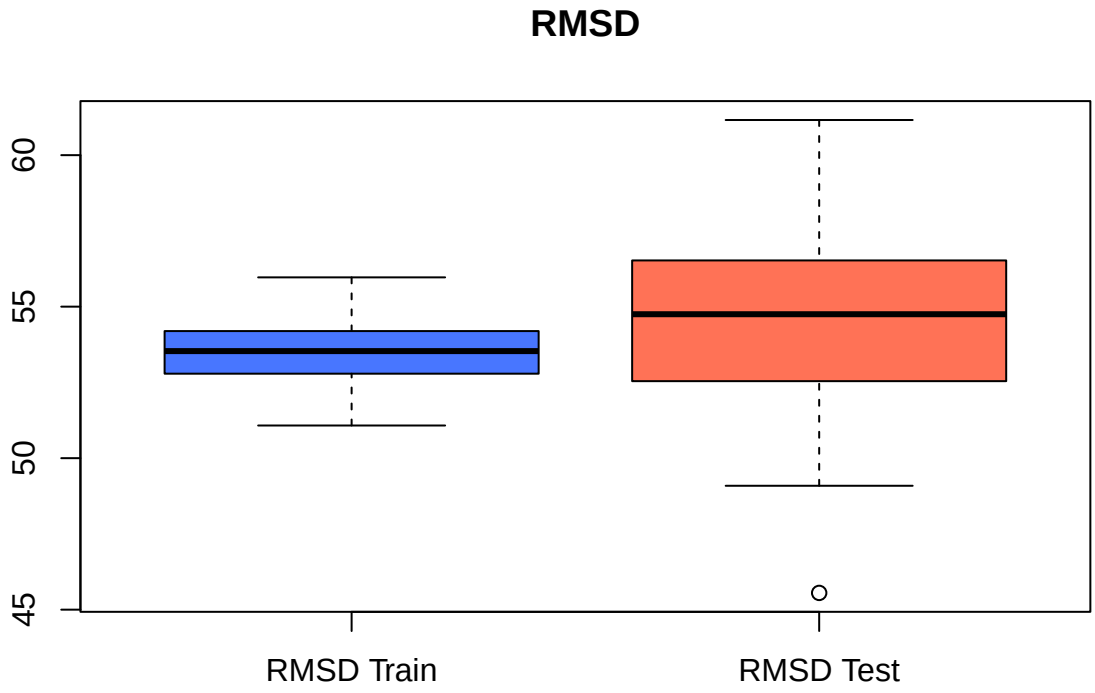
```
ggplot(data.frame(x = 1:length(R_Squared_Adj), y = R_Squared_Adj), aes(x, y)) +
  geom_point(size = 1.5, color = "royalblue3") +
  labs(title = "R-Adjusted Plot", x = "Index", y = "R-Adjusted") +
  theme(plot.title = element_text(hjust = 0.5))
```



After plotting the R-Adjusted we observe that all the values are in between 0.47 and 0.57.

Using boxplots we can compare the RSMD on train data set with the RMSD on the test data set:

```
boxplot(RMSD_Train, RMSD_Test,
        main = "RMSD", names = c("RMSD Train", "RMSD Test"),
        col = c("royalblue1", "coral1"))
```



We notice that the interquatile of the RMSD computed on the test data set is larger than the interquantile of the RMSD on the train data set. We can explain that by the fact that the coefficients are estimated based on the train data set which means they perfectly fit those data compared to the test data. We also observe that the min on the test data set is lower than the min on the train data set, same for the max of the test data set which are larger than the max of the train data set. Even though, we can see that the two medians are quite similar. This plot help us to identify if our model is overfitting the train data set.

Ridge Regeression

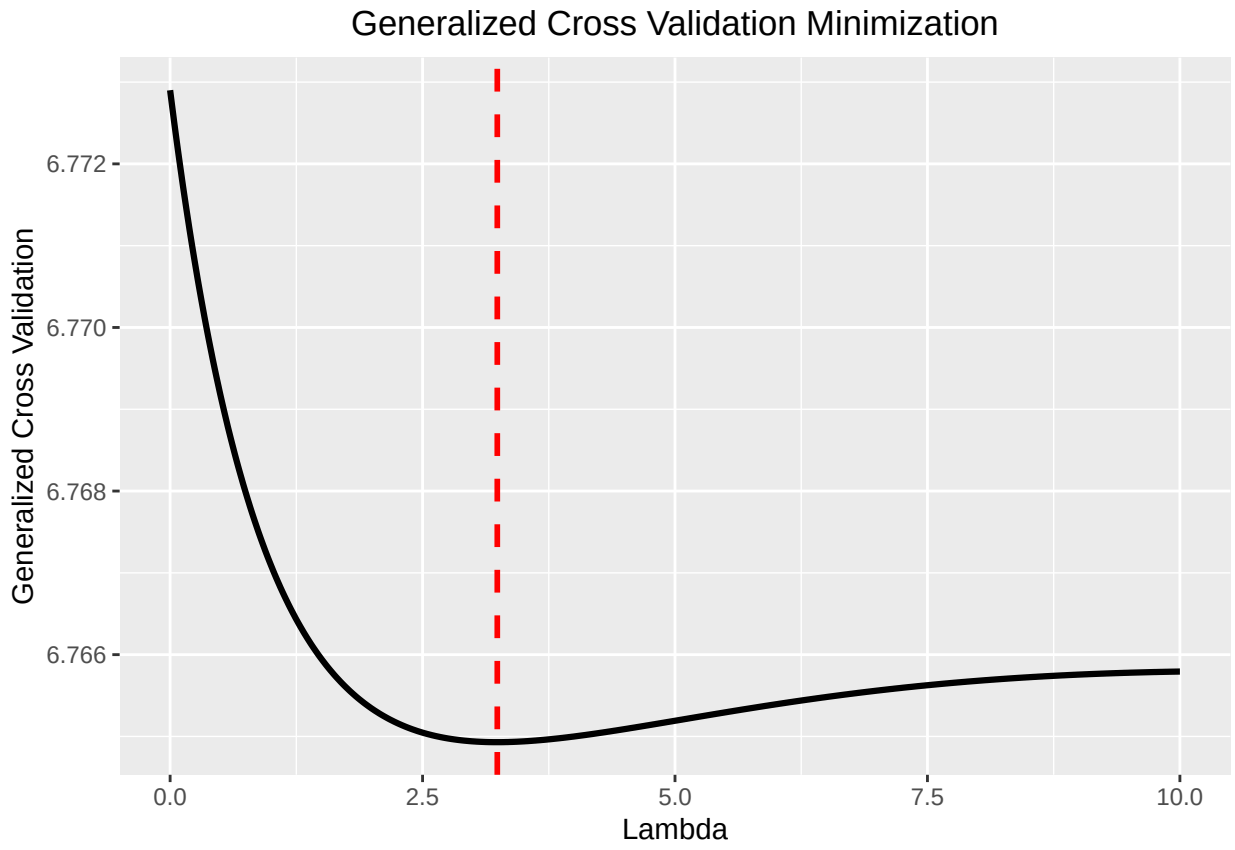
```
diabetes_ridge <- lm.ridge(formula = Y ~ ., data = diabetes_data, lambda = seq(0,10,0.01))
```

For better looking plots and easier results manipulation we will use the library broom:

- Tidy constructs a tibble that summarizes the model's statistical findings;
- Glance construct a concise one-row summary of the model.

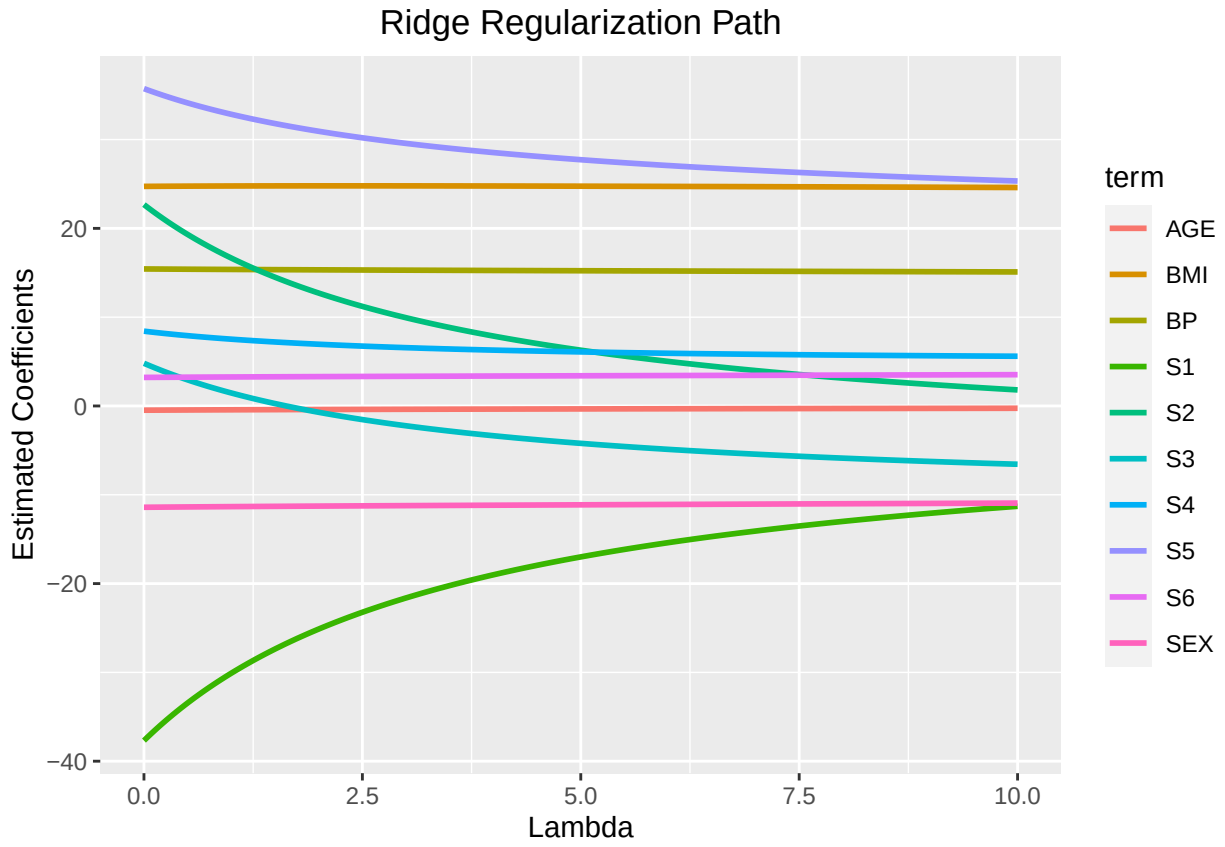
```
tidy_ridge <- tidy(diabetes_ridge)
glance_ridge <- glance(diabetes_ridge)
```

```
ggplot(tidy_ridge, aes(lambda, GCV)) +
  geom_line(lwd = 1.1) +
  geom_vline(xintercept = glance_ridge$lambdaGCV, col = "red", lty = 2, lwd = 1) +
  labs(title = "Generalized Cross Validation Minimization",
       x = "Lambda", y = "Generalized Cross Validation") +
  theme(plot.title = element_text(hjust = 0.5))
```



We want to choose the best lambda for our model, in order to that we need to minimize the Generalized Cross Validation. In the plot below, we can see the GCV for different values of lambda between 0 and 10. The red line shows us the value of lambda which minimizes the GCV.

```
ggplot(tidy_ridge, aes(lambda, estimate, color = term)) +
  geom_line(lwd = 1) +
  labs(title = "Ridge Regularization Path",
       x = "Lambda", y = "Estimated Coefficients") +
  theme(plot.title = element_text(hjust = 0.5))
```



In the plot above, we can see the values of all the evolution of the coefficient values depending on lambda. Knowing that we are doing a ridge regression, we can tell that they will all converge to 0 for the same value of lambda. Here we just found the value of lambda that minimizes GCV, using this value we found the corresponding coefficients:

```
min_lambda_index <- which.min(diabetes_ridge$GCV)
coef_ridge <- coef(lm.ridge(formula = Y ~ .,
                            data = diabetes_data,
                            lambda = diabetes_ridge$GCV[min_lambda_index]))
coef_ridge
```

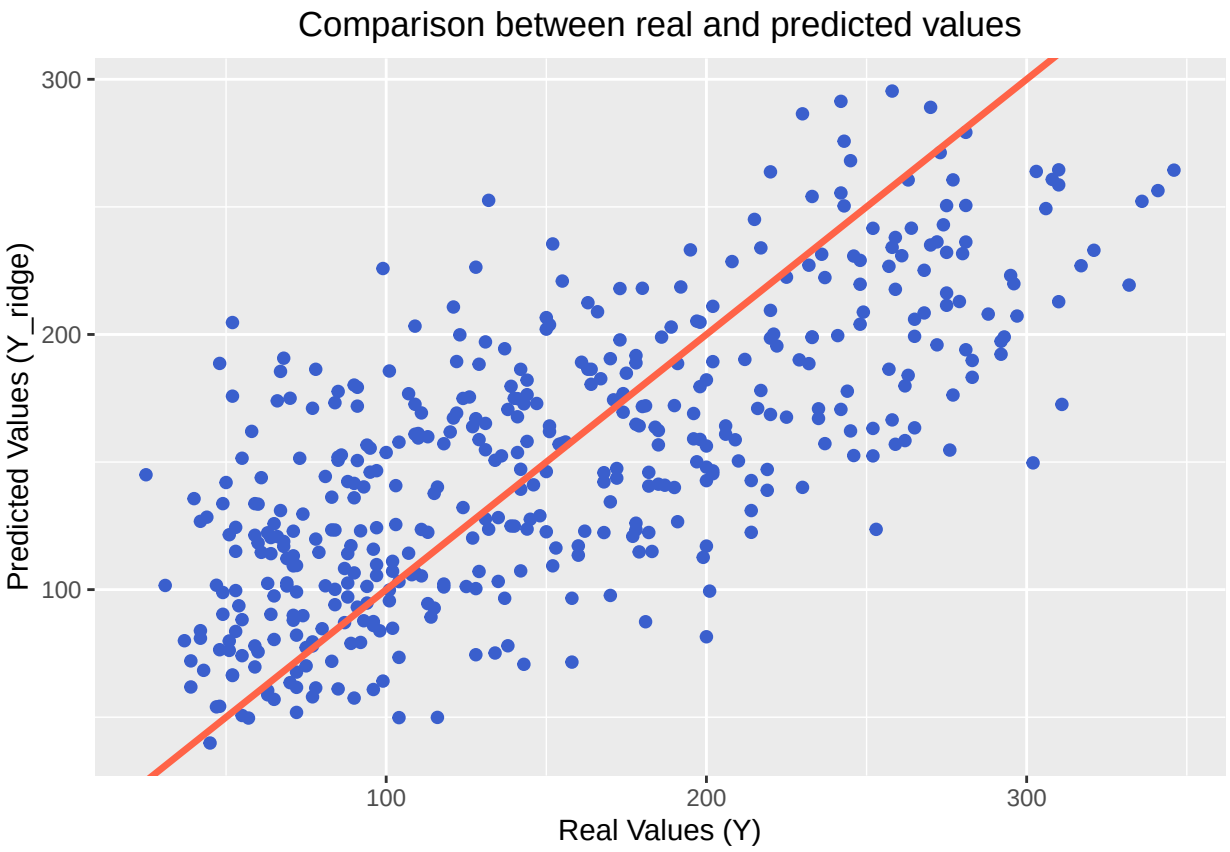
```
                    AGE           SEX           BMI           BP
-265.66427416 -0.02310917 -22.16232322  5.59778994  1.09860806
           S1           S2           S3           S4          S5
 -0.41568275  0.13869969 -0.41100359  4.53402482 51.07659116
           S6
  0.30009284
```

```
X <- cbind(rep(1, nrow(diabetes_data)), diabetes_data[c(1:10)])
colnames(X)[1] <- "1"
Y <- diabetes_data$Y
Y_ridge <- as.matrix(X) %*% as.vector(coef_ridge)
sqrt(sum((Y - Y_ridge) ^ 2) / length(Y_ridge))
```

```
[1] 53.5653
```

The value of the mean quadratic error is 53 which is quite big if we look at the order of magnitude of the values of our target variable Y.

```
comparaison_data <- as.data.frame(cbind(Y, Y_ridge))
ggplot(data = comparaison_data, mapping = aes(x = Y, y = Y_ridge)) +
  geom_point(size = 1.7, color = "royalblue3") +
  geom_abline(size = 1.2, color = "tomato1") +
  labs(title = "Comparison between real and predicted values",
       x = "Real Values (Y)", y = "Predicted Values (Y_ridge)") +
  theme(plot.title = element_text(hjust = 0.5))
```

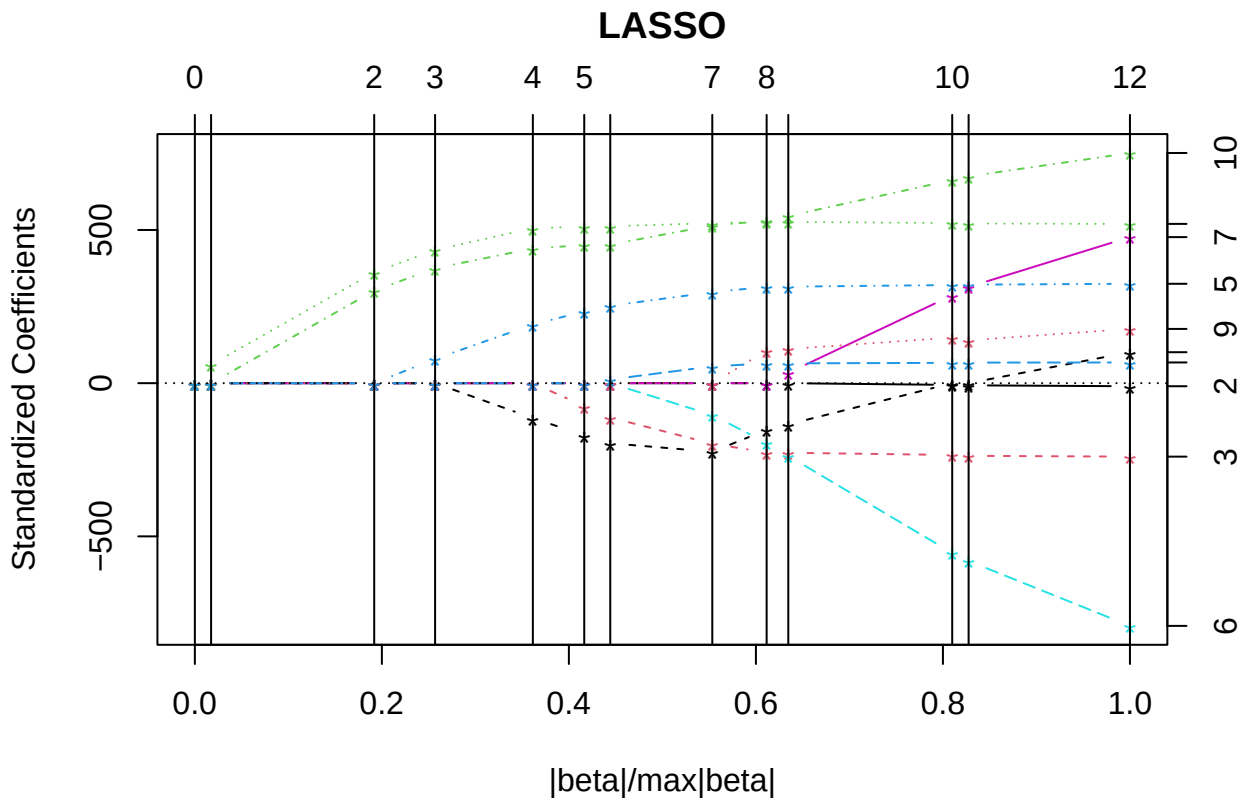


By plotting the real values versus the ridge prediction values computed we can see that the points are sometimes really far from the first bisector, which means the model doesn't quite good predictions.

Lasso Regression

```
diabetes_lasso <- lars(as.matrix(X), Y, type = "lasso")
```

```
plot(diabetes_lasso)
```



In the plot above we can observe the evolution of the variable coefficients. So when λ is equal to zero, there is no penalization, and you have the OLS solution which corresponds to the $max|\beta| = max \sum \beta_i$.

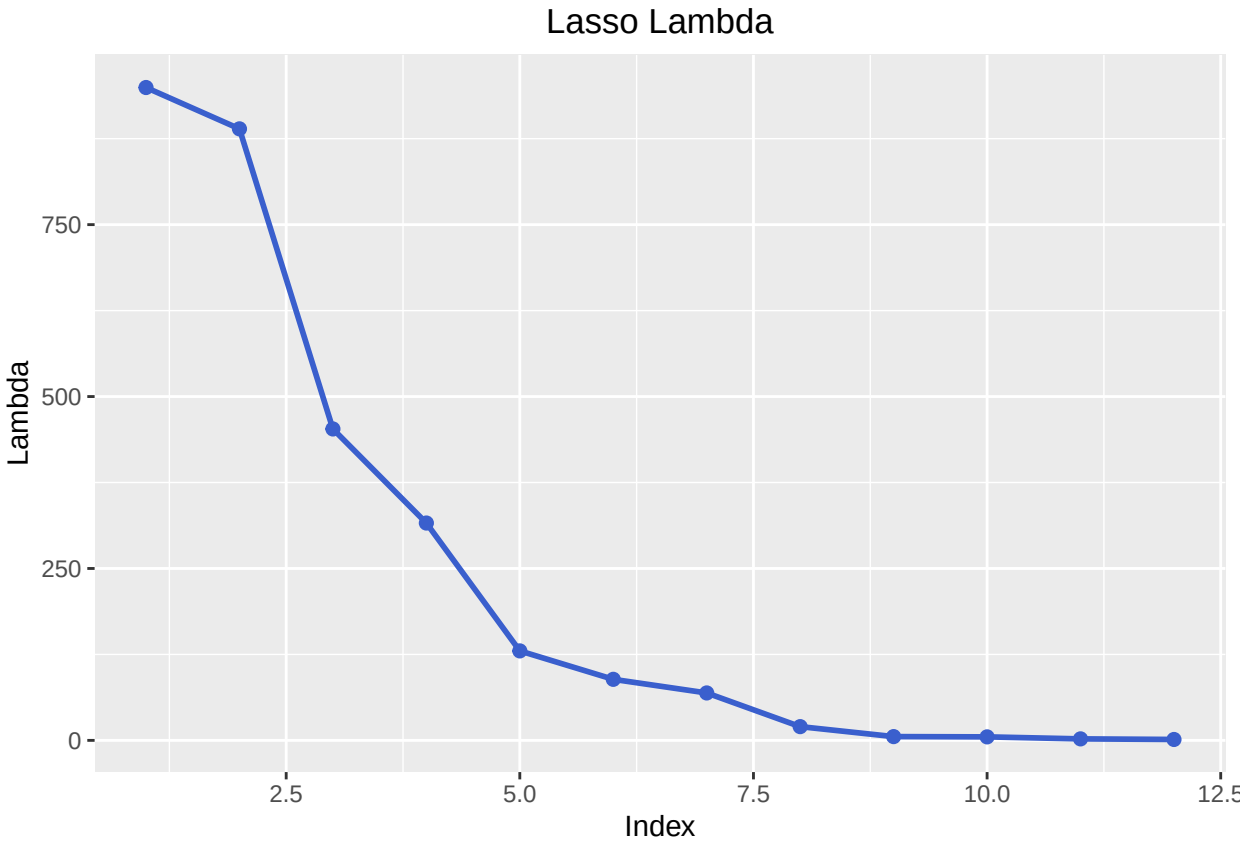
As the penalization λ increases, $\sum \beta_i$ is pulled towards zero, with the less important variables being pulled to zero earlier. At some level of λ, all the β_i have been pulled to zero.

The x-axis of the graph instead of presenting it as high λ on the left decreasing to zero when moving right, it presents it as the ratio of the sum of the absolute current estimate over the sum of the absolute OLS estimates.

The vertical bars indicate when a variable has been pulled to zero and is labeled with the number of variables remaining in the model.

The y-axis being standardized coefficients, generally when running LASSO, we need to standardize our variables so that the penalization occurs equally over the variables. If they were measured on different scales, the penalization would be uneven.

```r
ggplot(data = as.data.frame(diabetes_lasso$lambda),
       mapping = aes(x = seq(1,12), y = diabetes_lasso$lambda)) +
  geom_point(size = 2, color = "royalblue3") +
  geom_line(size = 1, color = "royalblue3") +
  labs(title = "Lasso Lambda", x = "Index", y = "Lambda") +
  theme(plot.title = element_text(hjust = 0.5))
```



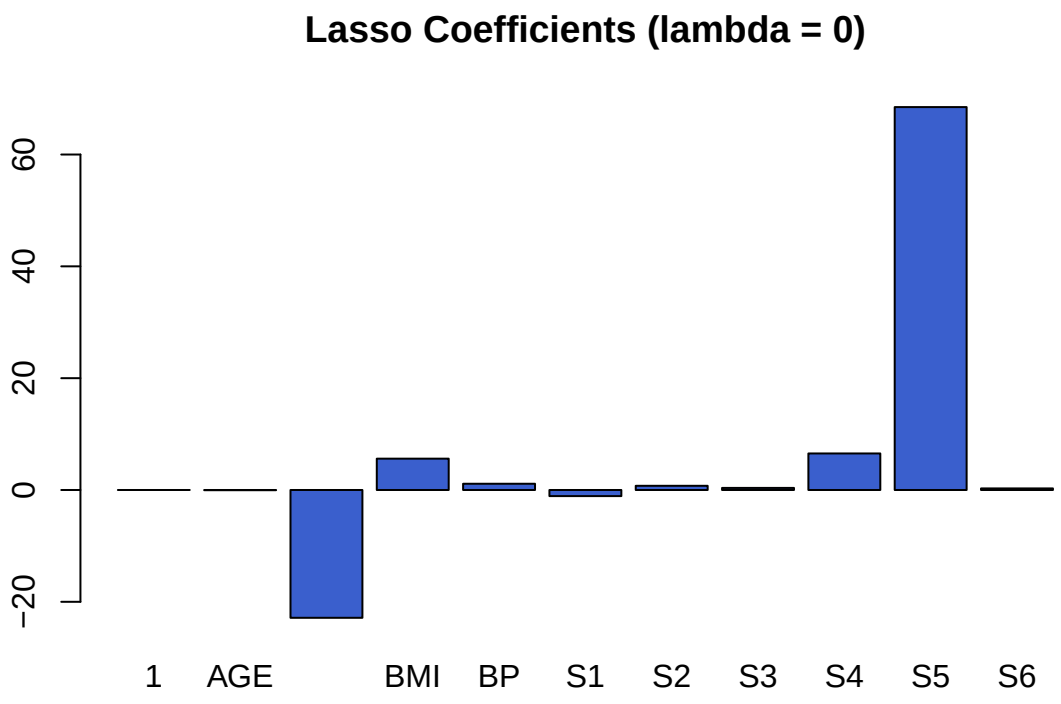
The above plot shows that the values of λ are decreasing by the index.

As we know that:

- the biggest value of λ will give us a model with no variables left
- the lowest value of λ will give us a model with all the variables.

So, the plot is consistent with the previous one.

```
coef_lasso <- predict.lars(diabetes_lasso, X, type ="coefficients", mode = "lambda", s = 0)
barplot(coef_lasso$coefficients, main = "Lasso Coefficients (lambda = 0)", col = "royalblue3")
```

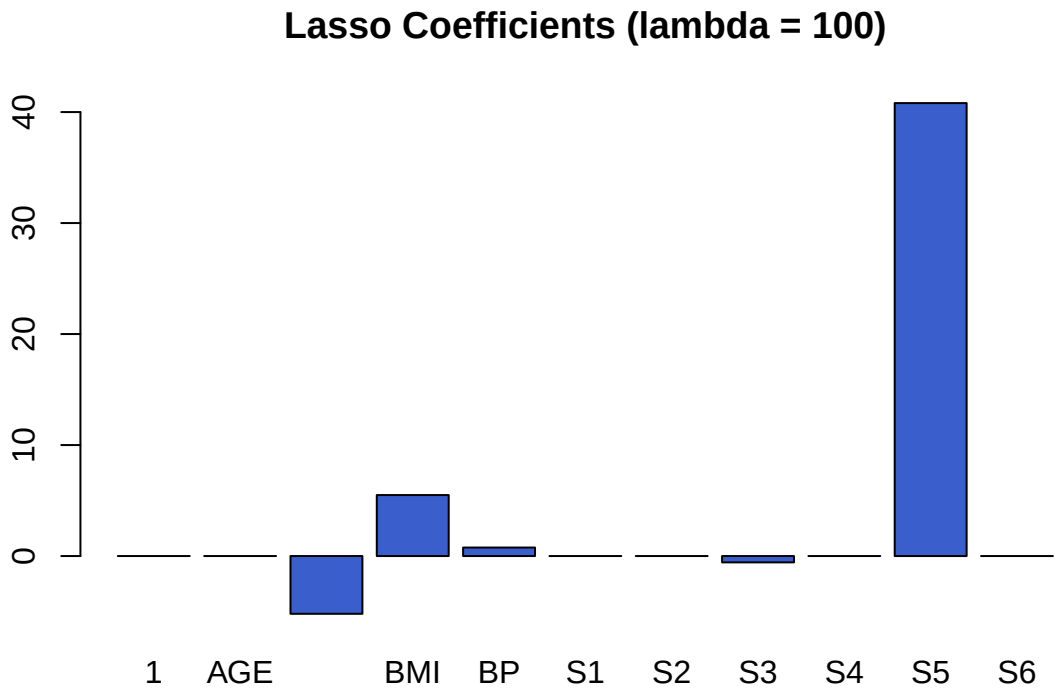


We can see that the predicted coefficients for $\lambda = 0$ gives us a model with:

- SEX;
- BMI;
- BP;
- S1;
- S2;
- S3;
- S4;
- S5;
- S6.

Let's now try to do the same plot with a larger value of λ, therefore we are going to recompute the lasso prediction using Lars again and plot the new coefficients obtained.

```r
coef_lasso <- predict.lars(diabetes_lasso, X, type ="coefficients", mode = "lambda", s = 100)
barplot(coef_lasso$coefficients, main = "Lasso Coefficients (lambda = 100)",col = "royalblue3")
```



We can see that the predicted coefficients for $\lambda = 100$ gives us a model with:

- SEX;
- BMI;
- BP;
- S3;
- S5.

We are going to choose the λ with the lowest mean quadratic error.

```r
MSE_lasso <- vector(length = length(diabetes_lasso$lambda) - 1)
for (k in 1:length(diabetes_lasso$lambda))
{
  Y_lasso <- predict.lars(diabetes_lasso, X, type = "fit",
                          mode = "lambda", s = diabetes_lasso$lambda[k])$fit
  MSE_lasso[k-1] <- sqrt(sum((Y - Y_lasso) ^ 2) / length(Y_lasso))
}
min_lambda <- diabetes_lasso$lambda[which.min(MSE_lasso)]
min_lambda
```

```
[1] 2.182267
```

The value of λ which minimizes the mean quadratic error is 2.182267.

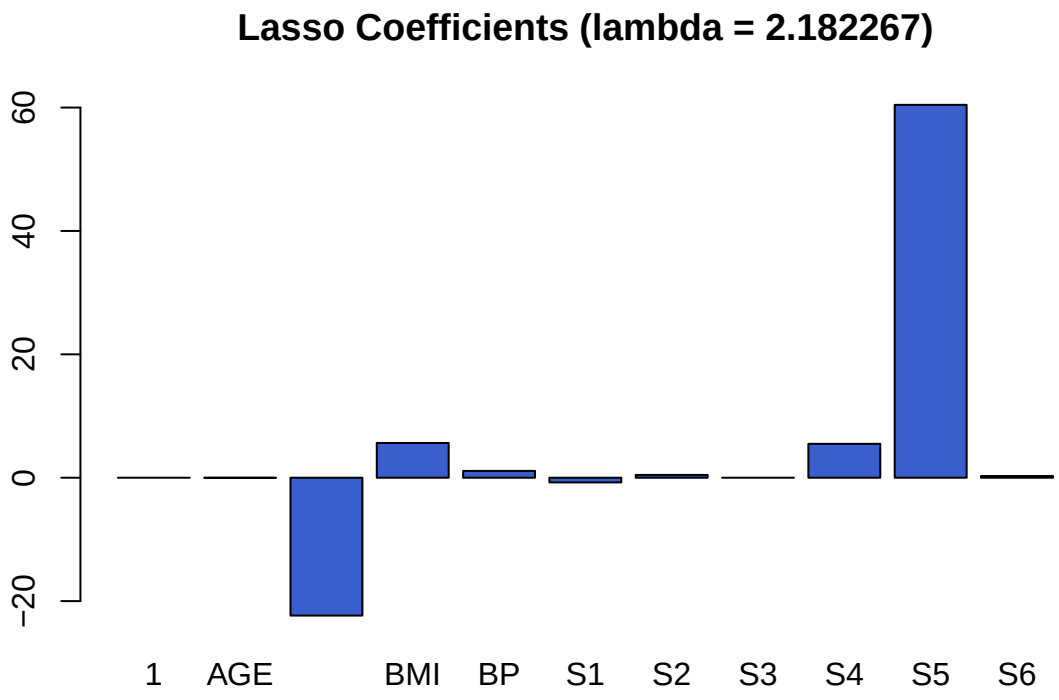
The minimum of mean quadratic error is 53.49715.

The coefficients corresponding to this value of λ are the following:

```
Y_lasso <- predict.lars(diabetes_lasso, X, type = "coefficients",
                        mode = "lambda", s = min_lambda)
data.frame(Y_lasso$coefficients)
```

```
    Y_lasso.coefficients
1             0.00000000
AGE          -0.02076645
SEX         -22.34287157
BMI           5.63323457
BP            1.10287047
S1           -0.76263741
S2            0.44894937
S3            0.00000000
S4            5.49456045
S5           60.43913023
S6            0.27475479
```

```
barplot(Y_lasso$coefficients, main = "Lasso Coefficients (lambda = 2.182267)",col = "royalblue3")
```



Here we can see the variables remaining in the model with their corresponding coefficients for the best value of λ that we have selected by minimizing the mean squared error.

Conclusion

In the classical regression, we just try to minimize the log-likelihood:

$$\beta_{OLS} = argmin_\beta \sum_{i=0}^{n} (y_i - \beta.x_i)^2$$

We can easily see that Ridge Regression encourages all coefficients to become small, meanwhile Lasso Regression encourages many coefficients to become zero, and a few non-zero. Both of them will reduce the accuracy on the training set, but improve prediction in some way by avoiding the overfitting.

$$\beta_{LASSO} = argmin_\beta \sum_{i=0}^{n} (y_i - \beta.x_i)^2 + \lambda.||\beta||_1$$

$$\beta_{RIDGE} = argmin_\beta \sum_{i=0}^{n} (y_i - \beta.x_i)^2 + \lambda.||\beta||_2^2$$

There is another method of penalization that can be very helpful if we want to combine Lasso and Ridge Regression by penalizing the likelihood with an L1-term and an L2-term.

$$\beta_{ELASTIC-NET} = argmin_\beta \sum_{i=0}^{n} (y_i - \beta.x_i)^2 + \lambda_1.||\beta||_1 + \lambda_2.||\beta||_2^2$$

Elastic Net Regression can be really helpful in order to have both advantages of Lasso and Ridge Regression. Another very useful penalization method is Group-Lasso which multiplies the β values by the square root number of elements in the group i.

$$\beta_{GROUP-LASSO} = argmin_\beta \sum_{i=0}^{n} (y_i - \beta.x_i)^2 + \lambda \sum_{i=0}^{n} \sqrt{p_i}.||\beta_i||_1$$

In the specific case of linear regression where there are not only continuous but also categorical variables (factors), the Lasso solution is not satisfactory as it only selects individual variables instead of whole factors. Moreover, the Lasso solution depends on how the variables are encoded. Choosing different contrasts for a categorical predictor will produce different solutions in general. Intuitively speaking, the Group Lasso can be preferred to the Lasso since it provides a means for us to incorporate a certain type of additional information into our estimate for the true coefficient.