Practical session - Modèles de régression régularisée

Piseth KHENG, Borachhun YOU

03 October 2022

IV. Medical data

Firstly, we load the data into data frames.

```
tab <- read.table("diabetes.txt", header=TRUE, sep="\t")
X <- tab[, 1:10]
Y <- tab[, 11]
head(tab)
```

```
##   AGE SEX  BMI  BP  S1    S2 S3 S4     S5 S6   Y
## 1  59   2 32.1 101 157  93.2 38  4 4.8598 87 151
## 2  48   1 21.6  87 183 103.2 70  3 3.8918 69  75
## 3  72   2 30.5  93 156  93.6 41  4 4.6728 85 141
## 4  24   1 25.3  84 198 131.4 40  5 4.8903 89 206
## 5  50   1 23.0 101 192 125.4 52  4 4.2905 80 135
## 6  23   1 22.6  89 139  64.8 61  2 4.1897 68  97
```

We now try computing for a multiple linear regression model with all the variables.

```
reg <- lm(Y~., data=tab)
summary(reg)
```

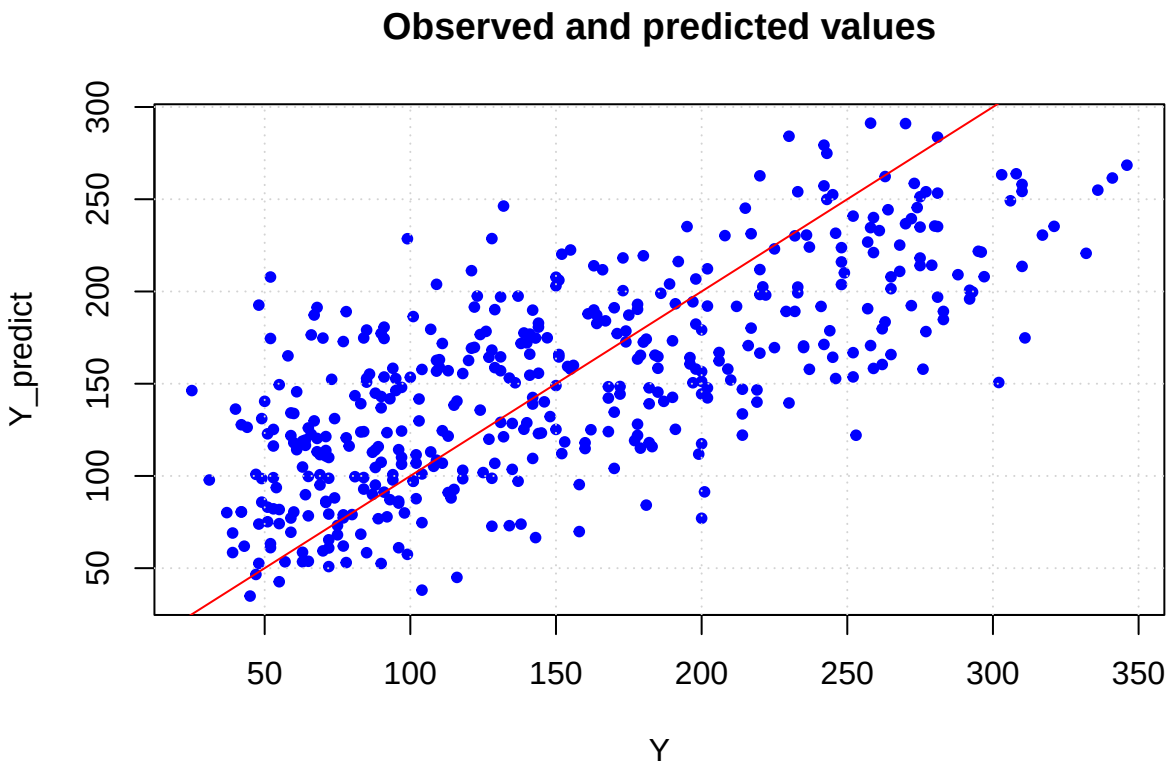
```
##
## Call:
## lm(formula = Y ~ ., data = tab)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -155.827  -38.536   -0.228   37.806  151.353
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -334.56714   67.45462  -4.960 1.02e-06 ***
## AGE           -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
```

Now, we predict the values of y using the obtained model above and compare them with the observed values in the data set.

```
Y_predict <- predict(reg, tab)

plot(Y, Y_predict, col="blue", pch=20,
     main="Observed and predicted values")
grid()
abline(a=0, b=1, col="red")
```



We can see in the $(y, \hat{y})$ graph above that the plotted points are scattered around the bisector, showing that the linear model is not good enough.

We now then try to find a better predictive model using the variable selection method, Ridge regression and Lasso regression.

1. Variable selection

Here, we are using the **forward selection** method.

```
regforward <- step(lm(Y~1, data=tab), list(upper=reg), 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
```

With forward selection, we start with a model with no variable. Then at each step, we add a variable to the model that gives the best improvement. In order to determine which variable to add to the model, the algorithm calculates the AIC of the model for each case of the variables, and chooses the one with the lowest AIC value.

```
summary(regforward)
```

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

In the final model, we can see that the variable AGE, S3, S4 and S6 are removed.

In terms of the R-squared and the adjusted R-squared, there are very small differences between this model and the multiple linear regression model above, meaning that this model is still not good enough.

As for the **backward selection**, it follows the same principle, but in opposite direction (starting with a full model with all variables then removing a variable at each step).

2. Ridge regression

With Ridge regression, the estimate of the coefficients $\hat{\beta}$ of the model is given by:

$$\hat{\beta}_{RR} = (X^T X + \lambda I_p)^{-1} X^T Y$$

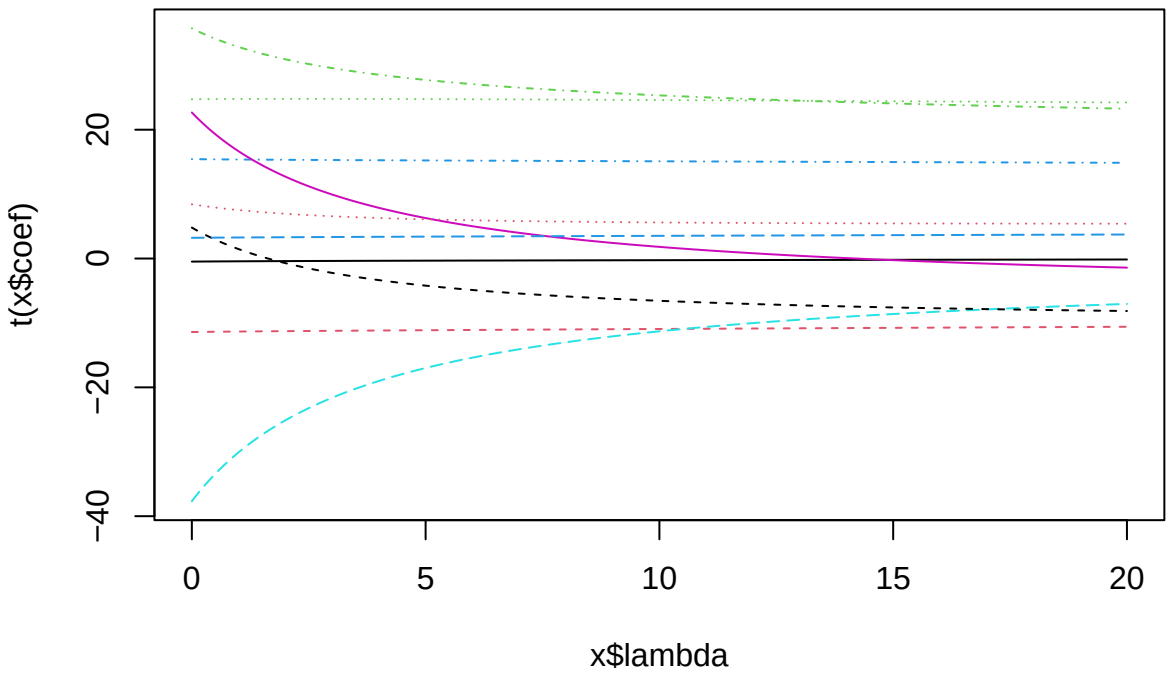
We then need to find the value of the penalization parameter λ.

We first compute the Ridge regression with λ having values from 0 to 20 with an increment of 0.01.

```
library(MASS)
resridge <- lm.ridge(Y~., data=tab, lambda=seq(0,20,0.01))
```

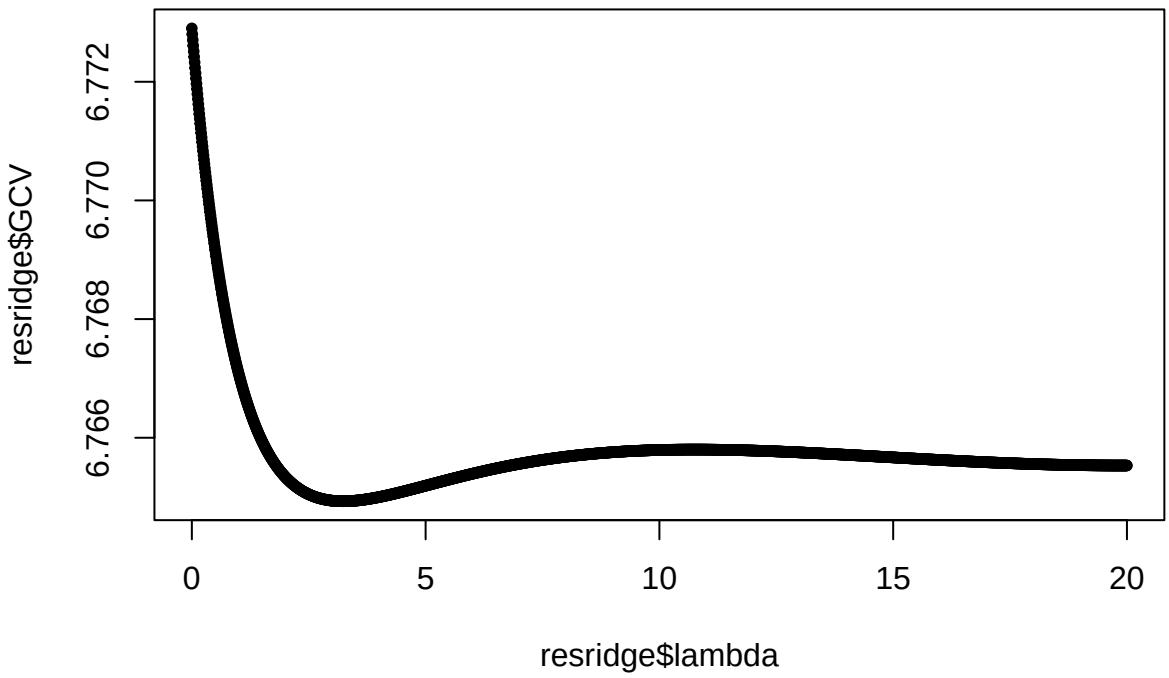
The graph below shows the evolution of the value of the coefficients with respect to λ from 0 to 20.

```
plot(resridge)
```



We now find the best λ by using the GCV (Generalized Cross Validation) values. The graph below is a plot of GCV against λ.

```
plot(resridge$lambda, resridge$GCV, pch=20)
```




```
best_ridge_lambda <- as.numeric(names(which.min(resridge$GCV)))
print(paste("Best lambda:", best_ridge_lambda))
```

```
## [1] "Best lambda: 3.24"
```

We then compute the regression with the best value of λ.

```
best_resridge <- lm.ridge(Y~., data=tab, lambda=best_ridge_lambda)
```

The values of the coefficients in the "rescaling" framework:

```
best_resridge$coef
```

```
##         AGE         SEX         BMI          BP          S1          S2
##  -0.3668366 -11.2105831  24.7803505  15.2836637 -20.9220865   9.3873897
##          S3          S4          S5          S6
##  -2.5302838   6.4908025  29.2938550   3.3482156
```

The values of the coefficients in the initial framework:

```
best_coefridge <- coef(best_resridge)
```

We now calculate the value of R-squared of the best Ridge regression model. The value of R-squared is given by:

$$R^2 = 1 - \frac{RSS}{TSS}$$

where RSS is the residual sum squared and TSS is the total sum squared.

```
Y_ridge <- cbind(matrix(1, nrow=dim(X)[1], ncol=1), as.matrix(X)) %*% as.vector(best_coefridge)
TSS <- sum((Y-mean(Y))^2)
```

```
RSS <- sum((Y-Y_predict)^2)
R_squared_ridge <- 1-(RSS/TSS)
print(paste("R-squared of Ridge regression model:", R_squared_ridge))
```

```
## [1] "R-squared of Ridge regression model: 0.51774842222035"
```

We can see that the value of R-squared of the Ridge regression model is almost the same as the value of R-squared of the multiple linear regression model, indicating that the resulting model is not improving.

3. Lasso regression

Here, we use `glmnet()` function of the `glmnet` library to compute Lasso regression.

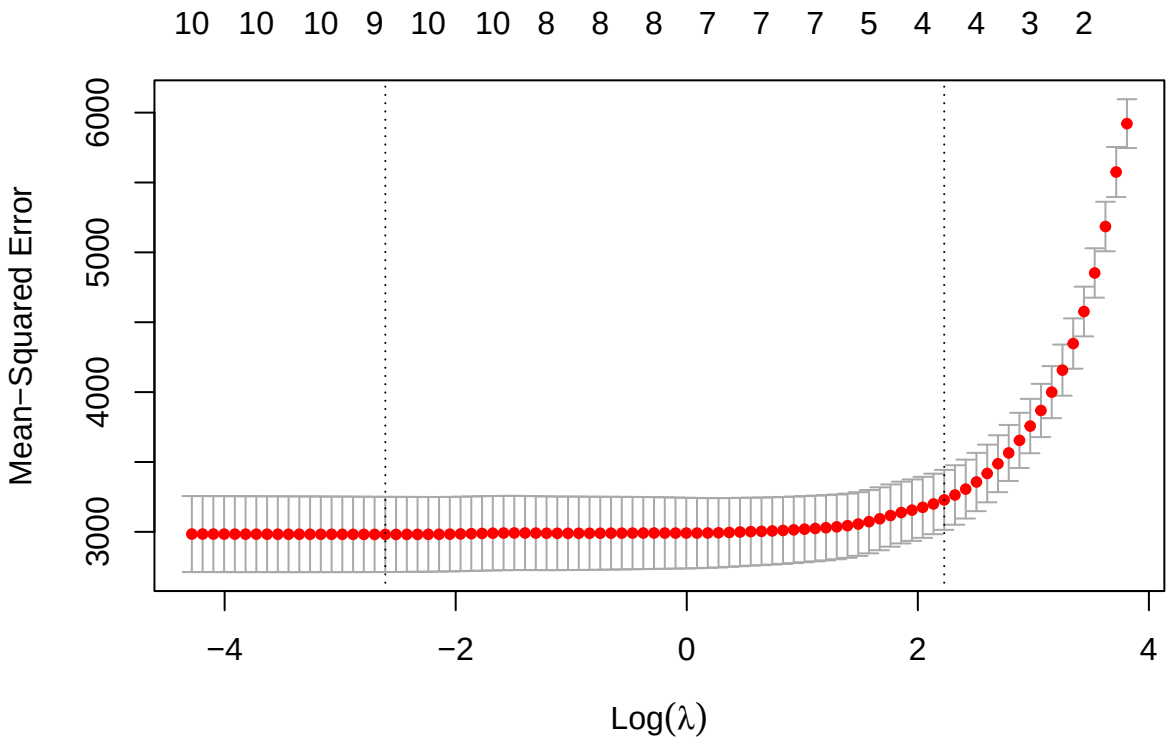
Similar to Ridge regression, we have to find the value of the penalization parameter λ. We therefore perform k-fold cross-validation in order to find the best value of λ.

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-4
```

```
cv_lasso <- cv.glmnet(as.matrix(X), Y, alpha=1)     # 10 folds default
plot(cv_lasso)
```




```
best_lasso_lambda <- cv_lasso$lambda.min
print(paste("Best lambda:", best_lasso_lambda))
```

```
## [1] "Best lambda: 0.0735995966173429"
```

We then compute the regression with the best value of λ.

```
best_reslasso <- glmnet(as.matrix(X), Y, alpha=1, lambda=best_lasso_lambda)
```

The values of the coefficients:

```
coef(best_reslasso)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept) -303.24112709
## AGE           -0.02432023
## SEX          -22.49809376
## BMI            5.62627039
## BP             1.10591890
## S1            -0.77818737
## S2             0.46707557
## S3             .
## S4             5.37361797
## S5            60.89521710
## S6             0.27696570
```

The value of R-squared:

```
best_reslasso$dev.ratio
```

```
## [1] 0.5174113
```

Once again, we see that the value of R-squared of the Lasso regression model is approximately the same as the value of R-squared of the multiple linear regression model, and thus the obtained model is not any better than the one of the multiple linear regression.