Exercises regression

Jan-Philipp Kolb

01 Juni, 2019

Exercise: regression Ames housing data

1) Install the package `AmesHousing` and create a **processed version** of the Ames housing data with the variables `Sale_Price`, `Gr_Liv_Area` and `TotRms_AbvGrd`
2) Create a regression model with `Sale_Price` as dependent and `Gr_Liv_Area` and `TotRms_AbvGrd` as independent variables. Then create seperated models for the two independent variables. Compare the results. What do you think?

Solution: regression Ames housing data

```r
install.packages("AmesHousing") # 1)
ames_data <- AmesHousing::make_ames() # 1)
```

Three regression models

```
lm(Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data = ames_data)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data =
##
## Coefficients:
##   (Intercept)    Gr_Liv_Area  TotRms_AbvGrd
##       42767.6          139.4       -11025.9
```

```
lm(Sale_Price ~ Gr_Liv_Area, data = ames_data)$coefficients
```

```
## (Intercept) Gr_Liv_Area
##   13289.634     111.694
```

```
lm(Sale_Price ~ TotRms_AbvGrd, data = ames_data)$coefficients
```

```
##   (Intercept) TotRms_AbvGrd
##      18665.40      25163.83
```

Exercise: ridge regression (I)

1) Load the `lars` package and the `diabetes` dataset
2) Load the `glmnet` package to implement ridge regression.

The dataset has three matrices x, x2 and y. x has a smaller set of independent variables while x2 contains the full set with quadratic and interaction terms. y is the dependent variable which is a quantitative measure of the progression of diabetes.

3) Generate separate scatterplots with the line of best fit for all the predictors in x with y on the vertical axis.

4) Regress y on the predictors in x using OLS. We will use this result as benchmark for comparison.

Solution: ridge regression (I)

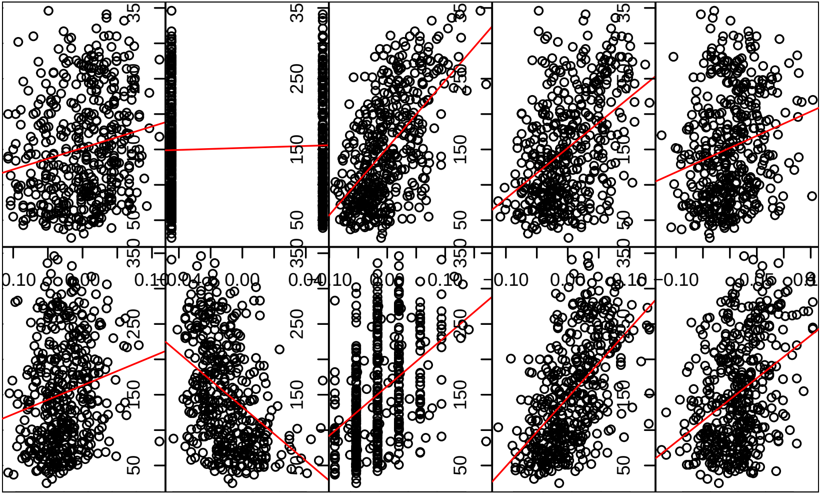
```
install.packages("lars")

library(lars) # 1)
data(diabetes)
attach(diabetes)

library(glmnet) #2)

# Create the scatterplots
set.seed(1234)
par(mfrow=c(2,5))
for(i in 1:10){ # 3)
  plot(x[,i], y)
  abline(lm(y~x[,i]))
}
```

Scatterplots



A OLS regression

```
model_ols <- lm(y ~ x) # 4)
summary(model_ols)
```

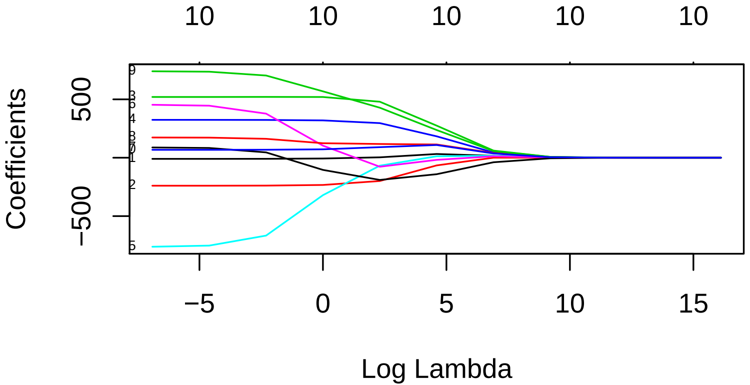
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -155.829  -38.534   -0.227   37.806  151.355
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   152.133      2.576  59.061  < 2e-16 ***
## xage          -10.012     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 ***
```

Exercise: ridge regression (II)

5) Fit the ridge regression model using the `glmnet` function and plot the trace of the estimated coefficients against lambdas. Note that coefficients are shrunk closer to zero for higher values of lambda.
6) Use the cv.glmnet function to get the cross validation curve and the value of lambda that minimizes the mean cross validation error.
7) Using the minimum value of lambda from the previous exercise, get the estimated beta matrix. Note that coefficients are lower than least squares estimates.
8) To get a more parsimonious model we can use a higher value of lambda that is within one standard error of the minimum. Use this value of lambda to get the beta coefficients. Note the shrinkage effect on the estimates.

Solution: ridge regression (Exercise 5)

```
lambdas <- 10^seq(7, -3)
model_ridge <- glmnet(x, y, alpha = 0, lambda = lambdas)
plot.glmnet(model_ridge, xvar = "lambda", label = TRUE)
```

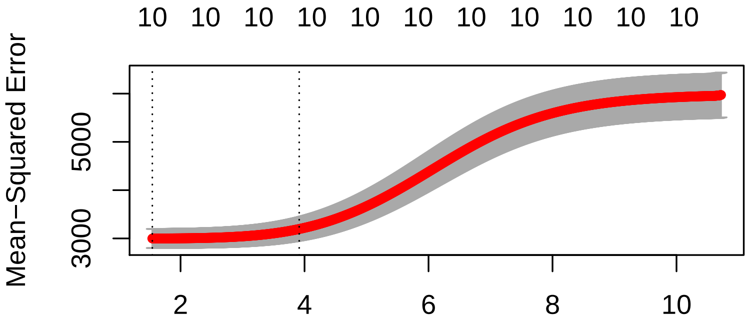


Solution: ridge regression (Exercise 6)

```
cv_fit <- cv.glmnet(x=x, y=y, alpha = 0, nlambda = 1000)
cv_fit$lambda.min
```

```
## [1] 4.685655
```

```
plot.cv.glmnet(cv_fit)
```



Solution: ridge regression (Exercise 7)

```
fit <- glmnet(x=x, y=y, alpha = 0, lambda=cv_fit$lambda.min)
fit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## age   -1.776856
## sex -218.078516
## bmi  503.649512
## map  309.268174
## tc  -116.815822
## ldl  -51.664814
## hdl -181.472590
## tch  113.468602
## ltg  470.871223
## glu   80.969338
```

Solution: ridge regression (Exercise 8)

```
fit <- glmnet(x=x, y=y, alpha = 0, lambda=cv_fit$lambda.1se)
fit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                s0
## age   23.935931
## sex -113.200049
## bmi  355.815095
## map  229.598721
## tc    -6.723554
## ldl  -47.972296
## hdl -167.389042
## tch  121.092669
## ltg  304.387870
## glu  112.675894
```

Exercise: ridge regression (III)

9) Split the data randomly between a training set (80%) and test set (20%). We will use these to get the prediction standard error for least squares and ridge regression models.
10) Fit the ridge regression model on the training and get the estimated beta coefficients for both the minimum lambda and the higher lambda within 1-standard error of the minimum.
11) Get predictions from the ridge regression model for the test set and calculate the prediction standard error. Do this for both the minimum lambda and the higher lambda within 1-standard error of the minimum.
12) Fit the least squares model on the training set.
13) Get predictions from the least squares model for the test set and calculate the prediction standard error.

Solution: ridge regression (Exercise 9)

```
library(caret)
intrain <- createDataPartition(y=diabetes$y,
                                        p = 0.8,
                                        list = FALSE)
training <- diabetes[intrain,]
testing <- diabetes[-intrain,]
```

Solution: ridge regression (Exercise 10a)

```
cv_ridge <- cv.glmnet(x=training$x, y=training$y,
                      alpha = 0, nlambda = 1000)
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = 0, lambda=cv_ridge$lambda.min)
ridge_reg$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## age    31.33299
## sex  -210.53163
## bmi   423.74733
## map   337.34705
## tc    -99.82516
## ldl   -46.24579
## hdl  -241.00979
## tch    67.98749
## ltg   432.93107
## glu   118.84876
```

Solution: ridge regression (Exercise 10b)

```
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = 0, lambda=cv_ridge$lambda.1se)
ridge_reg$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##             s0
## age   43.80905
## sex -111.99653
## bmi  322.61859
## map  252.51659
## tc   -17.89400
## ldl  -43.95097
## hdl -180.01382
## tch  104.13127
## ltg  300.13712
## glu  127.46322
```

Solution: ridge regression (Exercise 11a)

```
ridge_reg <- glmnet(x=training$x, y=training$y,
                alpha = 0, lambda=cv_ridge$lambda.min)
ridge_pred<-predict.glmnet(ridge_reg,
               s = cv_ridge$lambda.min,newx = testing$x)
sd((ridge_pred - testing$y)^2)/sqrt(length(testing$y))
```

```
## [1] 354.8159
```

Solution: ridge regression (Exercise 11b)

```
ridge_reg <- glmnet(x=training$x, y=training$y,
              alpha = 0, lambda=cv_ridge$lambda.1se)
ridge_pred <- predict.glmnet(ridge_reg,
              s = cv_ridge$lambda.1se, newx = testing$x)
sd((ridge_pred - testing$y)^2)/sqrt(length(testing$y))
```

```
## [1] 380.1932
```

Solution: ridge regression (Exercise 12)

```
ols_reg <- lm(y ~ x, data = training)
summary(ols_reg)

##
## Call:
## lm(formula = y ~ x, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -151.967  -39.604   -2.989   40.180  156.881
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  153.3004     2.9338  52.253  < 2e-16 ***
## xage          23.1119    67.9522   0.340 0.733974
## xsex        -243.0285    68.8648  -3.529 0.000474 ***
## xbmi         432.5835    77.2865   5.597 4.46e-08 ***
## xmap         367.4892    75.8545   4.845 1.92e-06 ***
```

Solution: ridge regression (Exercise 13)

```
ols_pred <- predict(ols_reg, newdata=testing$x,
                    type = "response")
sd((ols_pred - testing$y)^2)/sqrt(length(testing$y))
```

```
## [1] 358.3017
```