

EXERCISES REGRESSION

Jan-Philipp Kolb

27 Mai, 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 separated models for the two independent variables. Compare the results. What do you think?

SOLUTION: REGRESSION AMES HOUSING DATA

```
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 , x_2 and y . x has a smaller set of independent variables while x_2 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)

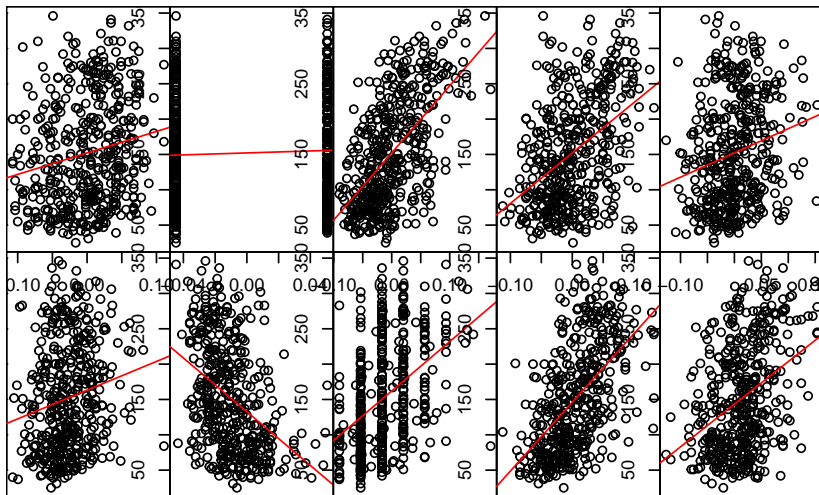
```
install.packages("lars")

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:
```

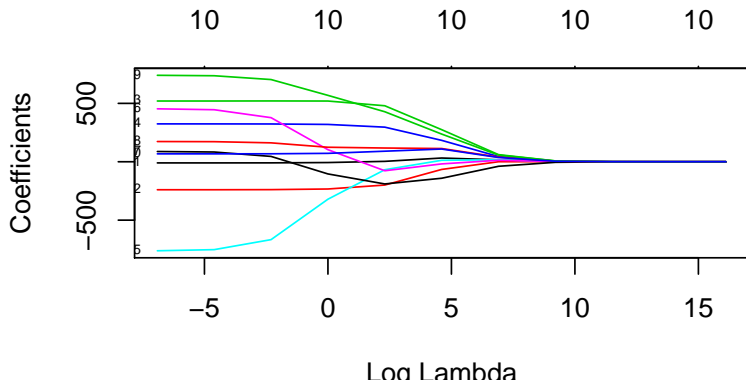
	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	152.133	2.576	59.061	< 2e-16	***
xage	-10.012	59.749	-0.168	0.867000	
xsex	-239.819	61.222	-3.917	0.000104	***
xbmi	519.840	66.534	7.813	4.30e-14	***
xmar	324.390	65.422	4.958	1.02e-06	***

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

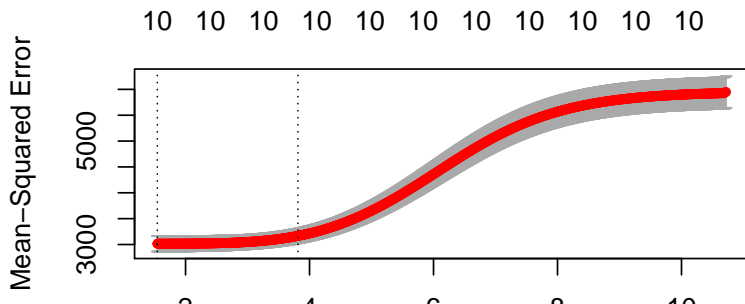


SOLUTION: RIDGE REGRESSION (EXERCISE 6)

```
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    22.463960
## sex -120.242424
## bmi  366.769877
## map  235.675888
## tc   -9.896792
## ldl  -52.093091
## hdl -170.482272
## tch  121.536668
## ltg  313.810749
## glu  112.152682
```

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    31.78324
## sex -209.01835
## bmi   524.74216
## map   323.20860
## tc    -136.96600
## ldl   -29.57508
## hdl  -170.95306
## tch    90.47813
## ltg   482.18291
## glu    56.68782
```


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    51.063186
```

```
## sex   -86.691190
```

```
## bmi   331.897059
```

```
## map   227.556673
```

```
## tc    -2.250169
```

```
## ldl   -37.166071
```

```
## hdl  -150.681629
```

```
## tch   114.454304
```

```
## ltg   279.018402
```

```
## glu   105.476529
```

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] 392.4241
```

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] 357.1851
```

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
## -156.660  -38.582   -0.434   35.392  152.598
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    152.56      2.87   53.156 < 2e-16 ***
## xage           26.46     67.04    0.395  0.69329
## xsex          -236.06     67.97   -3.473  0.00058 ***
## xbmj           542.48     74.50    7.282 2.26e-12 ***
## xman           335.80     72.30    4.644 4.86e-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] 414.2004
```