LAB 13

ABDUL RAUF

2023-02-05

Contents

__NAME__:-**ABDUL RAUF** \
__ENRL NO__:-**GL6092** \
__ROLL NO__:-**22DSMSA116** \

1 RIDGE REGRESSION:

Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. When the issue of multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values being far away from the actual values.

To perform lasso regression, we'll use functions from the `glmnet()` package. This package requires the response variable to be a vector and the set of predictor variables to be of the class data.matrix.

use the `glmnet()` function to fit the ridge regression model and specify alpha=0.

Note that setting alpha equal to 0 is equivalent to using ridge regression and setting alpha to some value between 0 and 1 is equivalent to using an elastic net.

To determine what value to use for lambda, we'll perform k-fold cross-validation and identify the lambda value that produces the lowest test mean squared error (MSE). `cv.glmnet()` automatically performs k-fold cross validation using k = 10 folds.

2 Load the Data

```
library(glmnet)
library(ISLR2)

#Define predictor and response variables
y <- mtcars$hp
x <- data.matrix(mtcars[, c('mpg', 'wt', 'drat', 'qsec')])
```

3 Fit the ridge regression model

```
model <- glmnet(x, y, alpha = 0)
summary(model)
```

```
##           Length Class     Mode
## a0        100    -none-    numeric
## beta      400    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
```

4 Performing k-fold cross-validation

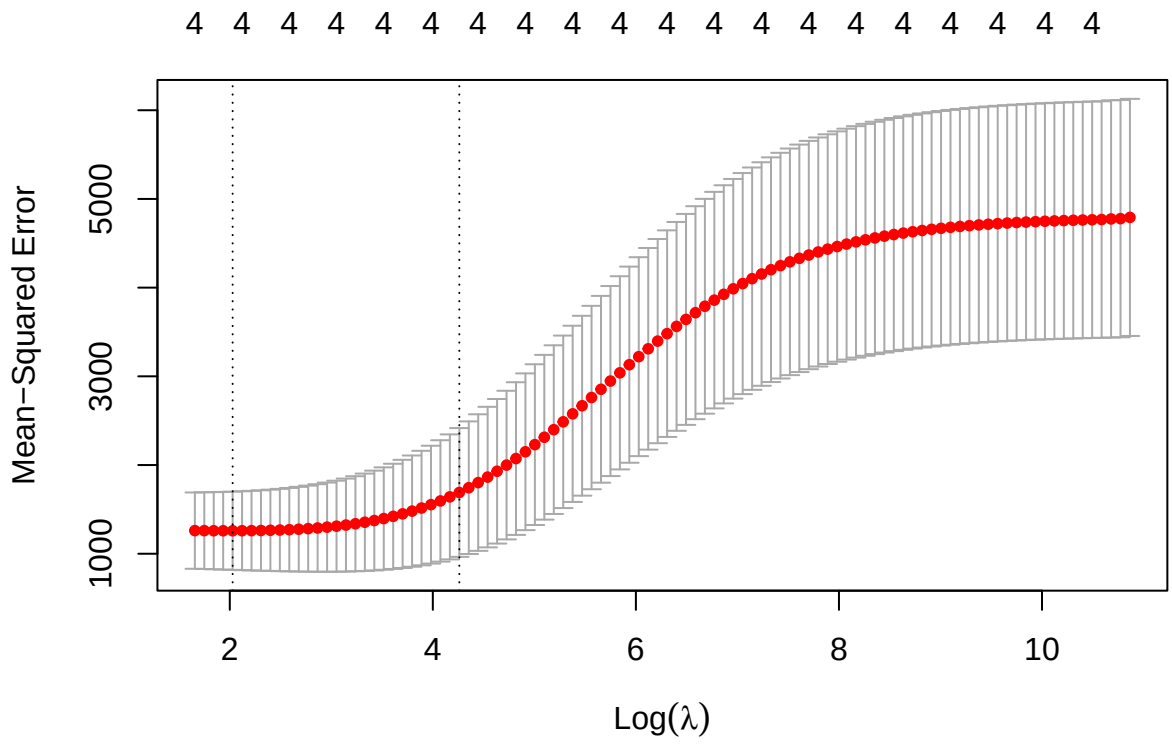
```
cv_model <- cv.glmnet(x, y, alpha = 0)
best_lambda <- cv_model$lambda.min

#display optimal lambda value
best_lambda
```

```
## [1] 7.599184
```

The lambda value that minimizes the test MSE turns out to be 11.02511.

5 Plot the test MSE's vs. lambda values

```
plot(cv_model)
```



6 Coefficients of best model

```
best_model <- glmnet(x, y, alpha = 0, lambda = best_lambda)
coef(best_model)
```

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept) 477.88658049
## mpg          -3.29585113
## wt           20.32422537
## drat         -0.09481675
## qsec        -18.49039687
```

the coefficient of the `dart` becomes exactly equal to 0 ,it get completely dropped the model.

7 Prediction for the response value of a new observation

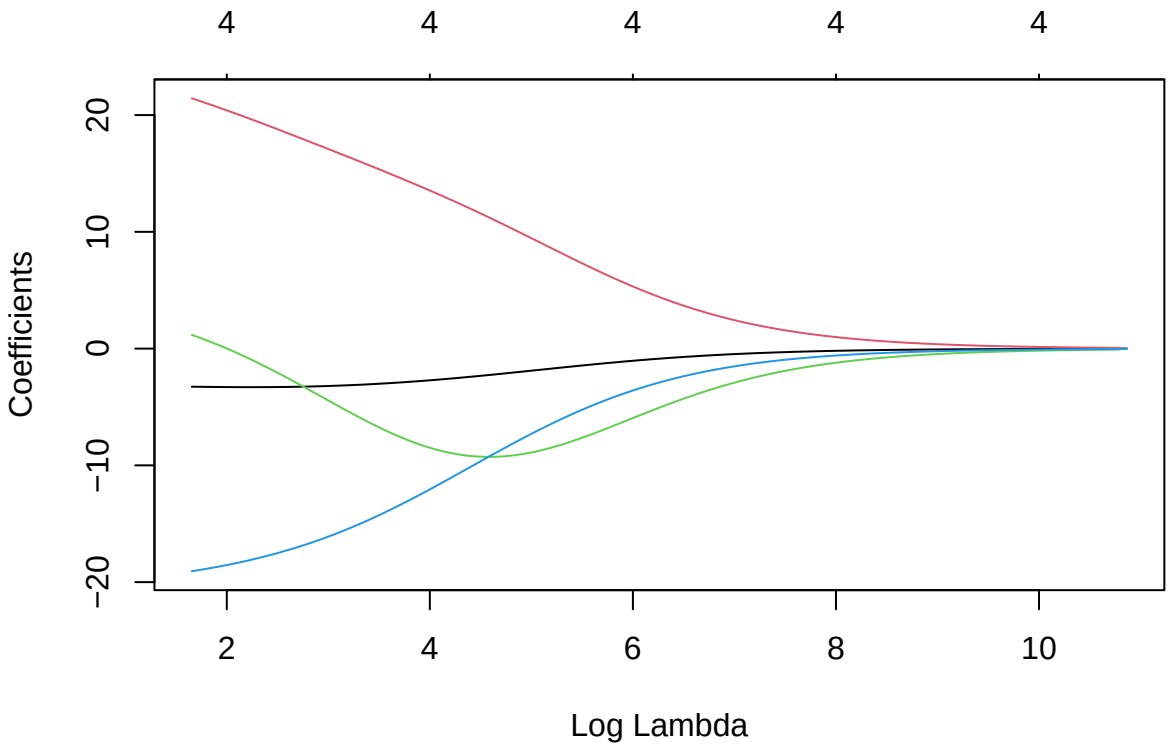
```
new = matrix(c(24, 2.8, 3.4, 18.4), nrow=1, ncol=4)
predict(best_model, s = best_lambda, newx = new)
```

```
##            s1
## [1,] 115.1483
```

based on the new input value as a test data , model predict that car have an hp value of 115.2554.

8 Ridge trace plot

```
plot(model,xvar = "lambda")
```



all the variables are approaching towards 0.

9 Find R-squared of model on training data

```
y_predicted <- predict(best_model, s = best_lambda, newx = x)

sst <- sum((y - mean(y))^2)
sse <- sum((y_predicted - y)^2)

rsq <- 1 - sse/sst
rsq
```

```
## [1] 0.8025993
```

R-squared turns out to be 0.7987891 which means 79.87% of the variation in the response is explained by model.