mtcars

Shiva Sankar Modala

2023-03-01

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-6

# Creating 80-20 Training Testing Split, createDataPartition() returns the indices  
# Perform a basic 80/20 test-train split on the data (you may use caret, the sample method, or manually)  
initial\_train = createDataPartition(mtcars$mpg,times=1,p=0.8,list=FALSE)  
# Training data  
training\_data= mtcars[initial\_train, ]  
# Testing data (note the minus sign)  
testing\_data= mtcars[-initial\_train, ]  
training\_data$am = factor(training\_data$am)  
is.factor(training\_data$am)

## [1] TRUE

# Fitting linear model  
# Fit a linear model with mpg as the target response,  
testing\_data$am = factor(testing\_data$am)  
lm.fit = lm(mpg~.,data=training\_data)  
#MSE on test set  
mean((predict(lm.fit,testing\_data)-testing\_data$mpg)^2)

## [1] 11.26835

# What features are selected as relevant based on resulting t-statistics?  
# Analyze the t-stat and p-values to select relevant features  
summary(lm.fit)

##   
## Call:  
## lm(formula = mpg ~ ., data = training\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.9424 -1.7282 -0.2225 1.0956 5.4001   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 24.62378 19.65034 1.253 0.227  
## cyl -0.41449 1.09482 -0.379 0.710  
## disp 0.01090 0.01814 0.601 0.556  
## hp -0.03299 0.02539 -1.299 0.211  
## drat 0.88507 1.76755 0.501 0.623  
## wt -2.73163 2.07093 -1.319 0.205  
## qsec 0.18021 0.86946 0.207 0.838  
## vs 0.08982 2.36188 0.038 0.970  
## am1 1.15988 2.43176 0.477 0.639  
## gear 0.85259 1.54799 0.551 0.589  
## carb -0.41727 0.91617 -0.455 0.655  
##   
## Residual standard error: 2.632 on 17 degrees of freedom  
## Multiple R-squared: 0.8699, Adjusted R-squared: 0.7934   
## F-statistic: 11.37 on 10 and 17 DF, p-value: 1.079e-05

cat(" We will select wt as a predictor based on the statistics as it has the lowest p value.")

## We will select wt as a predictor based on the statistics as it has the lowest p value.

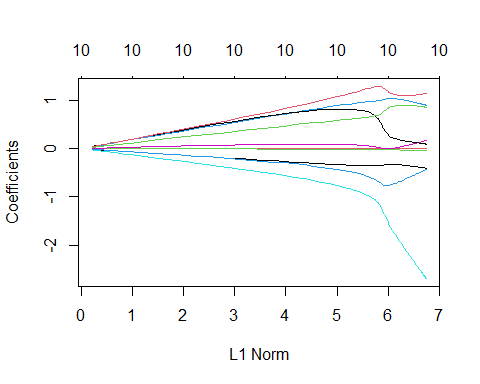
# coefficient values for relevant features  
lm.fit$coefficients

## (Intercept) cyl disp hp drat wt   
## 24.62378182 -0.41448777 0.01090413 -0.03298694 0.88506840 -2.73162674   
## qsec vs am1 gear carb   
## 0.18021450 0.08982164 1.15987939 0.85259465 -0.41726838

lambda\_seq = 10^seq(3, -3, by= -.06)  
# Perform a ridge regression using the glmnet package  
ridge\_regression<-glmnet(model.matrix(training\_data$mpg~.,data = training\_data)[, - 1],training\_data$mpg,alpha=0,lambda=lambda\_seq)  
summary(ridge\_regression)

## Length Class Mode   
## a0 101 -none- numeric  
## beta 1010 dgCMatrix S4   
## df 101 -none- numeric  
## dim 2 -none- numeric  
## lambda 101 -none- numeric  
## dev.ratio 101 -none- numeric  
## nulldev 1 -none- numeric  
## npasses 1 -none- numeric  
## jerr 1 -none- numeric  
## offset 1 -none- logical  
## call 5 -none- call   
## nobs 1 -none- numeric

plot(ridge\_regression)



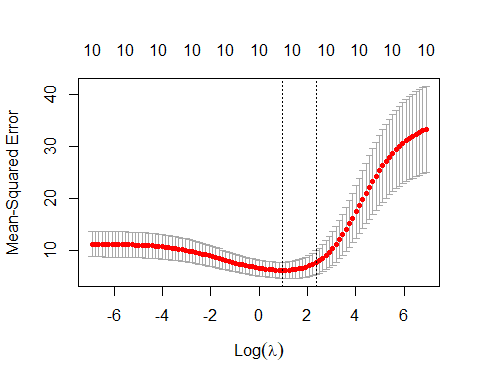
# Use cross-validation (via cv.glmnet) to determine the minimum value for lambda - what do you obtain  
cross\_validation<-cv.glmnet(model.matrix(training\_data$mpg~.,data = training\_data)[,- 1],training\_data$mpg,alpha=0,lambda = lambda\_seq,grouped = FALSE)  
cat("\n The best lambda: %s",cross\_validation$lambda.min)

##   
## The best lambda: %s 2.630268

lambda\_bst<-cross\_validation$lambda.min  
summary(cross\_validation)

## Length Class Mode   
## lambda 101 -none- numeric   
## cvm 101 -none- numeric   
## cvsd 101 -none- numeric   
## cvup 101 -none- numeric   
## cvlo 101 -none- numeric   
## nzero 101 -none- numeric   
## call 6 -none- call   
## name 1 -none- character  
## glmnet.fit 12 elnet list   
## lambda.min 1 -none- numeric   
## lambda.1se 1 -none- numeric   
## index 2 -none- numeric

# Plot training MSE as a function of lambda  
plot(cross\_validation)



# What is out-of-sample test set performance (using predict)  
testing\_predict<-predict(ridge\_regression,s=lambda\_bst,newx = model.matrix(testing\_data$mpg~.,data = testing\_data)[, -1])  
mean((testing\_data$mpg-testing\_predict)^2)

## [1] 11.50495

coef(cross\_validation)

## 11 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) 20.376767880  
## cyl -0.321695828  
## disp -0.004734022  
## hp -0.010568861  
## drat 0.866619485  
## wt -0.732029668  
## qsec 0.091940966  
## vs 0.816459140  
## am1 1.039344279  
## gear 0.570440867  
## carb -0.405025502

# Has ridge regression performed shrinkage, variable selection, or both?  
cat("\n As we can see that new coefficients are smaller, we can say that the ridge regression performs shrinkage.")

##   
## As we can see that new coefficients are smaller, we can say that the ridge regression performs shrinkage.