Homework3

Margo Kim

2024-02-20

Load Libraries and Datasets

```r
# Load library
library('class')
library('dplyr')
library('glmnet')
library('ggplot2')
```

```r
## Load prostate data
prostate <- read.csv("./prostate.csv")
```

Train the model

```r
## Subset to training examples
prostate_train <- subset(prostate, train==TRUE)
x_train <- model.matrix(lpsa ~ . -train, data = prostate_train)
y_train <- prostate_train$lpsa

# Subset the data for testing
prostate_test <- subset(prostate, train == FALSE)
x_test <- model.matrix(lpsa ~ . -train, data = prostate_test)
y_test <- prostate_test$lpsa
```

```r
## Train the model
model <- lm(lpsa ~ . -train, data = prostate_train)
```

Calculate Test Error

```r
# Make predictions on the testing set
test_predictions <- predict(model, prostate_test)
```

```r
# Compute the average squared-error loss
test_error <- mean((prostate_test$lpsa - test_predictions)^2)

print(paste("Test Error (MSE):", test_error))
```

```
## [1] "Test Error (MSE): 0.521274005507601"
```

Train a Ridge Regression model and tune parameters

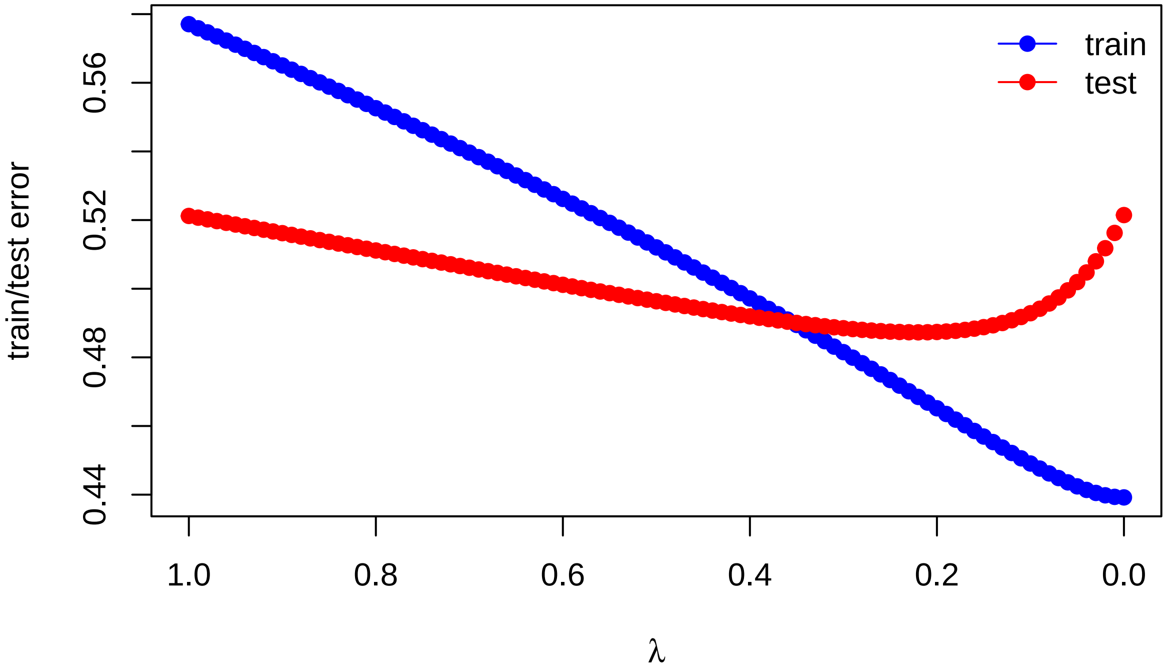
```r
## use glmnet to fit ridge
## glmnet fits using penalized L2 loss
## first create an input matrix and output vector
grid = seq(1, 0, -0.01)
form  <- lpsa ~  lweight + age + lbph + lcp + pgg45 + lcavol + svi + gleason
x_inp <- model.matrix(form, data=prostate_train)
y_out <- prostate_train$lpsa
fit <- glmnet(x=x_inp, y=y_out, alpha = 0, lambda=grid)

## functions to compute testing/training error w/lm
L2_loss <- function(y, yhat)
  (y-yhat)^2
error <- function(dat, fit, loss=L2_loss)
  mean(loss(dat$lpsa, predict(fit, newdata=dat)))

## functions to compute testing/training error with glmnet
error <- function(dat, fit, lam, form, loss=L2_loss) {
  x_inp <- model.matrix(form, data=dat)
  y_out <- dat$lpsa
  y_hat <- predict(fit, newx=x_inp, s=lam)  ## see predict.elnet
  mean(loss(y_out, y_hat))
}
```

```r
## compute training and testing errors as function of lambda
err_train_1 <- sapply(fit$lambda, function(lam)
  error(prostate_train, fit, lam, form))
err_test_1 <- sapply(fit$lambda, function(lam)
  error(prostate_test, fit, lam, form))

# Find the index of the minimum MSE
min_mse_index <- which.min(err_test_1)

# Find the corresponding lambda value
optimal_lambda <- grid[min_mse_index]

# Print the optimal lambda
print(paste("Optimal lambda:", optimal_lambda))
```

```
## [1] "Optimal lambda: 0.22"
```

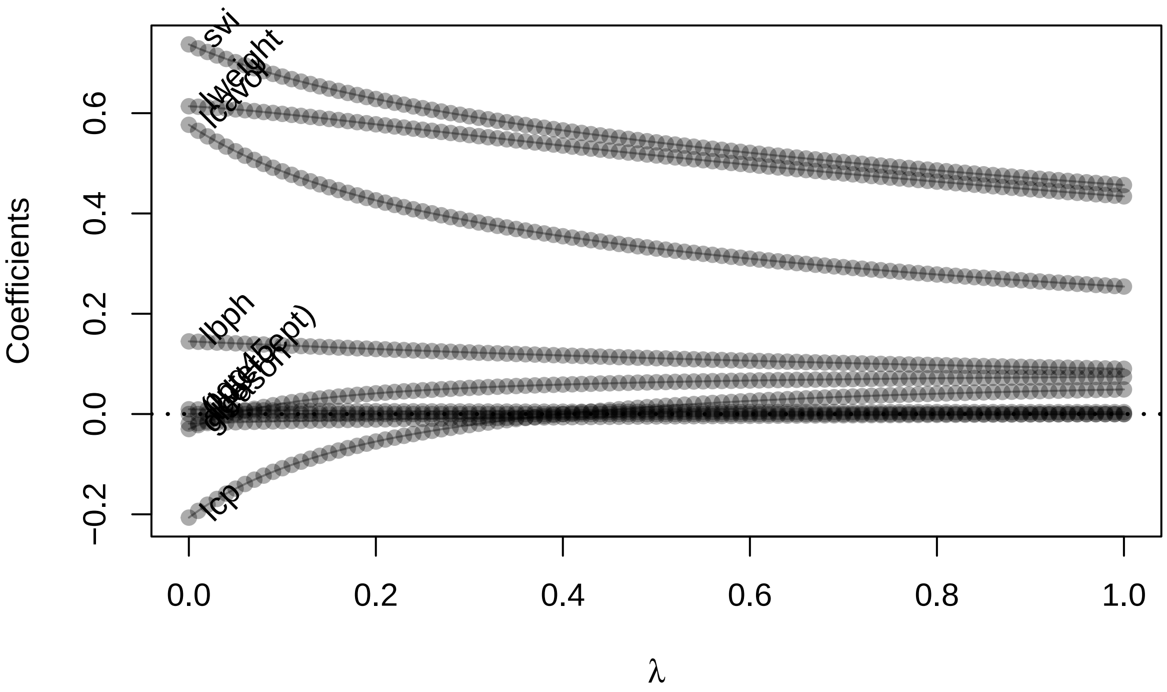
PLOT1

```r
## plot test/train error
plot(x=range(fit$lambda),
     y=range(c(err_train_1, err_test_1)),
     xlim=rev(range(fit$lambda)),
     type='n',
     xlab=expression(lambda),
     ylab='train/test error')
points(fit$lambda, err_train_1, pch=19, type='b', col='blue')
points(fit$lambda, err_test_1, pch=19, type='b', col='red')
legend('topright', c('train','test'), lty=1, pch=19,
       col=c('blue','red'), bty='n')
```



PLOT2

```
plot(x=range(fit$lambda),
     y=range(as.matrix(fit$beta)),
     type='n',
     xlab=expression(lambda),
     ylab='Coefficients')
for(i in 1:nrow(fit$beta)) {
  points(x=fit$lambda, y=fit$beta[i,], pch=19, col='#00000055')
  lines(x=fit$lambda, y=fit$beta[i,], col='#00000055')
}
text(x=0, y=fit$beta[,ncol(fit$beta)],
     labels=rownames(fit$beta),
     xpd=NA, pos=4, srt=45)
abline(h=0, lty=3, lwd=2)
```


svi
lweight
lcavol
lbph
(Intercept)
pgg45
gleason
age
lcp
Coefficients
−0.2
0.0
0.2
0.4
0.6
0.0
0.2
0.4
0.6
0.8
1.0
λ