HW11

Lucy Lin

2022-11-29

##Problem 1

```
data = read.csv("used_car.csv")
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
folds <- createFolds(data$price, 10)

folds <- createFolds(data$price, 10)
rmse <- matrix(0, 10, 2)
for (k in 1:10) {
  train <- data[-folds[[k]], ]
  validation <- data[folds[[k]], ]


  fit1 = lm(price ~ . , data = data)
  pred1 <- predict(fit1, newdata = validation)
  rmse[k, 1] <- sqrt(mean((validation$price - pred1)^2))

  fit2 = lm(log(price) ~ . , data = data)
  pred2 <- predict(fit2, newdata = validation)
  rmse[k, 2] <- sqrt(mean((validation$price - exp(pred2))^2))

}

#fit1 and fit 2 RMSE scores
print(rmse)
```

```
##           [,1]     [,2]
##  [1,] 1586.786 1934.863
##  [2,] 1571.835 1850.644
##  [3,] 1629.137 1987.352
##  [4,] 1605.368 1915.405
##  [5,] 1615.490 1890.475
##  [6,] 1582.400 1771.555
##  [7,] 1593.941 1773.516
##  [8,] 1611.223 1864.247
##  [9,] 1638.860 1887.941
## [10,] 1556.380 1748.235
```

```
#fit1 and fit2 mean RMSE
colMeans(rmse)
```

```
## [1] 1599.142 1862.423
```

```
#https://stackoverflow.com/questions/18047896/column-standard-deviation-r
#fit1 and fit2 RMSE standard deviation
apply(rmse, 2,sd)
```

```
## [1] 25.74122 77.77630
```

##Problem 2 #Procedure 1 already uses a model selection process at step 1. The folds may be redundant and can overfit. Procedure 2 independently creates different folds to get candidate models.

##Problem 3 a) True, as lambda decreases(towards model 2), beta ridge decreases so the ratio (beta ridge vs beta ols) approaches 1 b) True, model 1 has a higher lambda (increases penalty term and therefore bias) c) True, in sample SSE/training decreases when lambda decreases d) True, we cannot determine test errors with the current given information

##Problem 4 a)Yes, I expect collinearity because brozek and siri are derived in a similar formula and body fat may be spread out on the various body measurements.

```
data = data(fat, package = "faraway")
help(fat)
```

```
## No documentation for 'fat' in specified packages and libraries:
## you could try '??fat'
```

```
##??faraway::fat
```

b)

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-6
```

```
data(fat, package = "faraway")
x <- model.matrix(brozek ~ ., data = fat)[, -1]
y <- fat$brozek
```

c)

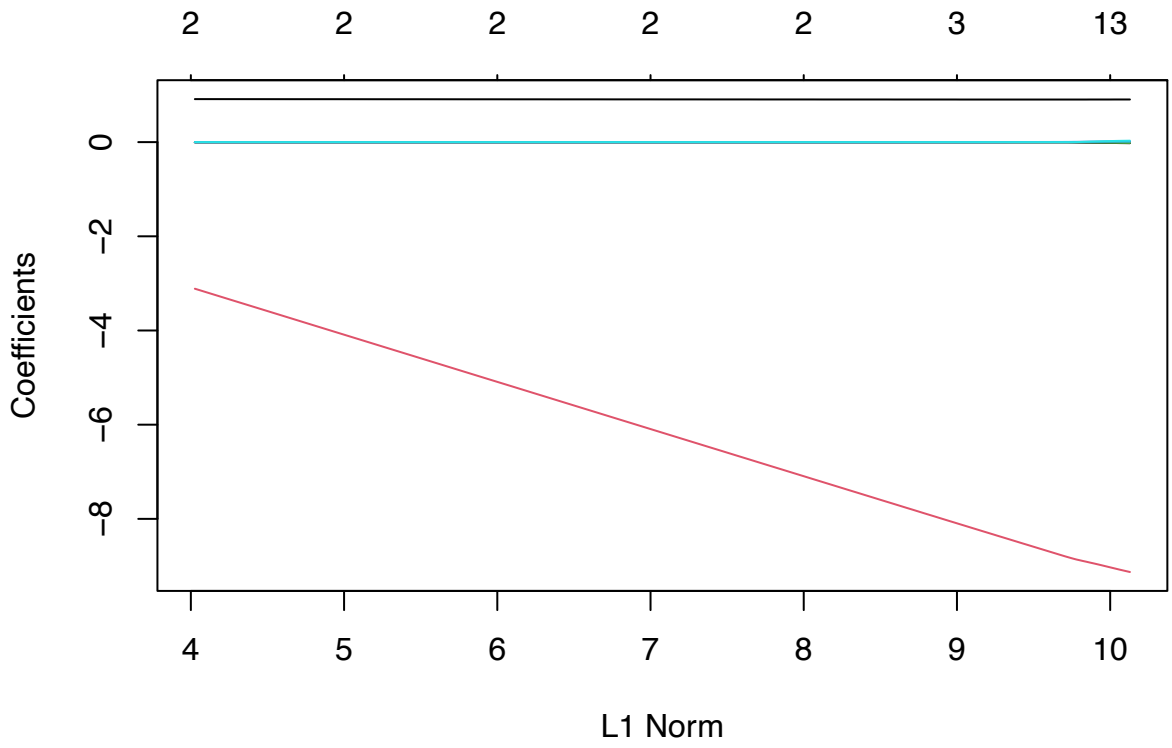
```
#fat.df <- data.frame(fat)


index <- sample(nrow(fat), round(nrow(fat) * 0.8))
xtrain <- x[index, ]
```

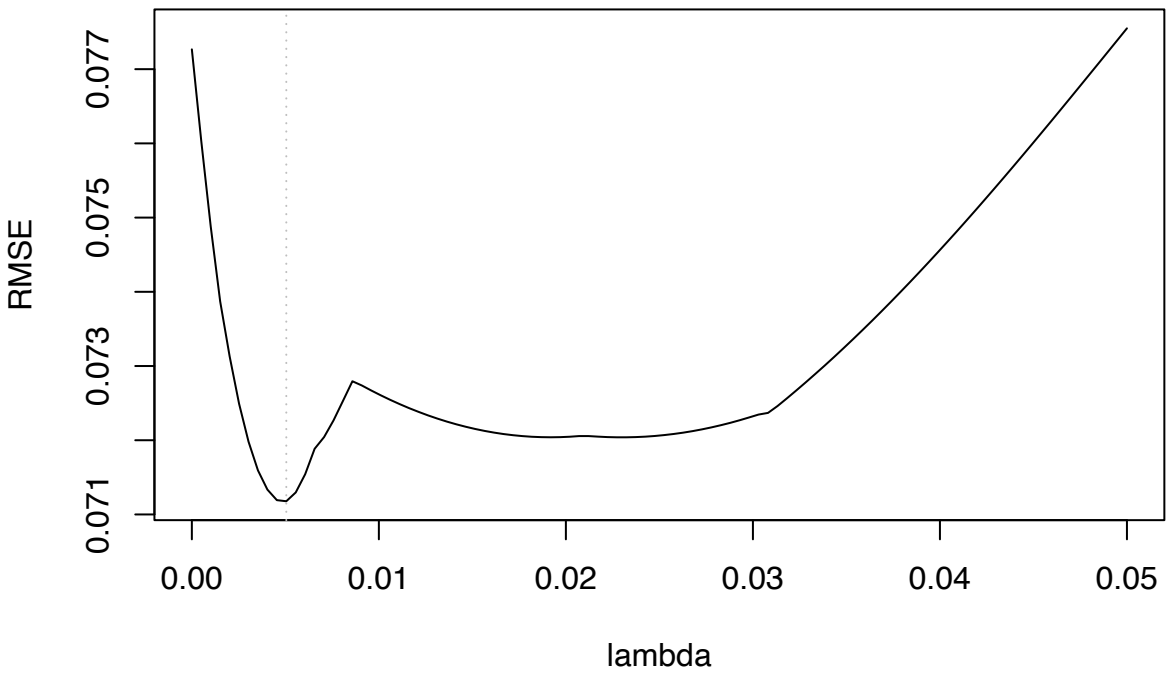
```
ytrain <- y[index]
xvalidation <- x[-index, ]
yvalidation <- y[-index]
grid = seq(0, 0.05, length = 100)
fit.lasso <- glmnet(x, y, alpha = 1, lambda = grid)
rmse <- rep(0, length(grid))
for (i in 1:length(grid)) {
pred <- predict(fit.lasso, s = grid[i], newx = xvalidation)
rmse[i] <- sqrt(mean((yvalidation - pred)^2))
}
min(rmse) #optimal rmse
```

```
## [1] 0.07117786
```

```
plot(fit.lasso)
```




```
plot(grid, rmse, "l", xlab = "lambda", ylab = "RMSE")
abline(v = grid[which.min(rmse)], col = "gray", lty = 3)
```




```
fit.lasso.cv <- cv.glmnet(x, y, alpha = 1, lambda = grid)
fit.lasso.cv$lambda.min
```

```
## [1] 0.03787879
```

d) Yes, lambda changes a lot because there are a lot of collinear terms, which results in a lower beta values