HW 4

r12631055

2024-10-07

Problem 1

Investigate the car price (cars.dat) using linear regression and lasso and optimize the value of the tuning parameter so that the resulting model has smallest residuals.

```
cars_data <- read.table("cars.dat", header = TRUE)
```

```
str(cars_data)
```

```
## 'data.frame':    74 obs. of  14 variables:
##  $ Model: chr  "AMC_Concord" "AMC_Pacer" "AMC_Spirit" "Audi_5000" ...
##  $ P    : int  4099 4749 3799 9690 6295 9735 4816 7827 5788 4453 ...
##  $ M    : int  22 17 22 17 23 25 20 15 18 26 ...
##  $ R78  : chr  "3" "3" "." "5" ...
##  $ R77  : chr  "2" "1" "." "2" ...
##  $ H    : num  2.5 3 3 3 2.5 2.5 4.5 4 4 3 ...
##  $ R    : num  27.5 25.5 18.5 27 28 26 29 31.5 30.5 24 ...
##  $ Tr   : int  11 11 12 15 11 12 16 20 21 10 ...
##  $ W    : int  2930 3350 2640 2830 2070 2650 3250 4080 3670 2230 ...
##  $ L    : int  186 173 168 189 174 177 196 222 218 170 ...
##  $ T    : int  40 40 35 37 36 34 40 43 43 34 ...
##  $ D    : int  121 258 121 131 97 121 196 350 231 304 ...
##  $ G    : num  3.58 2.53 3.08 3.2 3.7 3.64 2.93 2.41 2.73 2.87 ...
##  $ C    : int  1 1 1 3 3 3 1 1 1 1 ...
```

```
install.packages("caret")
```

```
## 將程式套件安載入 'C:/Users/Paul/AppData/Local/R/win-library/4.4'
## (因為 'lib' 沒有被指定)
```

```
## 程式套件 'caret' 開啟成功，MD5 和檢查也透過
```

```
## Warning: 無法將拆除原來安裝的程式套件 'caret'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): 複製
## C:\Users\Paul\AppData\Local\R\win-library\4.4\00LOCK\caret\libs\x64\caret.dll
## 到 C:\Users\Paul\AppData\Local\R\win-library\4.4\caret\libs\x64\caret.dll
## 時出了問題：Permission denied
```

```
## Warning: 回覆了 'caret'
```

```
##
## 下載的二進位程式套件在
##  C:\Users\Paul\AppData\Local\Temp\RtmpWkMQIJ\downloaded_packages 裡
```

```
library(glmnet)
```

```
## Warning: 套件 'glmnet' 是用 R 版本 4.4.1 來建造的
```

```
## 載入需要的套件：Matrix
```

```
## Loaded glmnet 4.1-8
```

```
library(caret)
```

```
## Warning: 套件 'caret' 是用 R 版本 4.4.1 來建造的
```

```
## 載入需要的套件：ggplot2
```

```
## 載入需要的套件：lattice
```

```
library(dplyr)
```

```
## Warning: 套件 'dplyr' 是用 R 版本 4.4.1 來建造的
```

```
##
## 載入套件：'dplyr'
```

```
## 下列物件被遮斷自 'package:stats':
##
##     filter, lag
```

```
## 下列物件被遮斷自 'package:base':
##
##     intersect, setdiff, setequal, union
```

Let’s begin pre-processing it seems R77 & R78 represents repair record in 1977 & 1978

```
cars_data$R77 <- as.numeric(ifelse(cars_data$R77 == ".", 0, cars_data$R77))
cars_data$R78 <- as.numeric(ifelse(cars_data$R78 == ".", 0, cars_data$R78))

# Ordinal Encoding
cars_data$R77 <- factor(cars_data$R77, ordered = TRUE)
cars_data$R78 <- factor(cars_data$R78, ordered = TRUE)

cars_data$R78 <- as.numeric(cars_data$R78)
cars_data$R77 <- as.numeric(cars_data$R77)

# Remove rows with missing values
cars_data_clean <- cars_data %>% na.omit()
```

```
str(cars_data_clean)
```

```
## 'data.frame':    74 obs. of  14 variables:
##  $ Model: chr  "AMC_Concord" "AMC_Pacer" "AMC_Spirit" "Audi_5000" ...
##  $ P    : int  4099 4749 3799 9690 6295 9735 4816 7827 5788 4453 ...
##  $ M    : int  22 17 22 17 23 25 20 15 18 26 ...
##  $ R78  : num  4 4 1 6 4 5 4 5 4 1 ...
##  $ R77  : num  3 2 1 3 4 5 4 5 5 1 ...
##  $ H    : num  2.5 3 3 3 2.5 2.5 4.5 4 4 3 ...
##  $ R    : num  27.5 25.5 18.5 27 28 26 29 31.5 30.5 24 ...
##  $ Tr   : int  11 11 12 15 11 12 16 20 21 10 ...
##  $ W    : int  2930 3350 2640 2830 2070 2650 3250 4080 3670 2230 ...
##  $ L    : int  186 173 168 189 174 177 196 222 218 170 ...
##  $ T    : int  40 40 35 37 36 34 40 43 43 34 ...
##  $ D    : int  121 258 121 131 97 121 196 350 231 304 ...
##  $ G    : num  3.58 2.53 3.08 3.2 3.7 3.64 2.93 2.41 2.73 2.87 ...
##  $ C    : int  1 1 1 3 3 3 1 1 1 1 ...
```

```
X <- model.matrix(P ~ . - Model - P, data = cars_data_clean)  # Predictor variables
y <- cars_data_clean$P  # Target variable (price)
```

Since linear regression does not require hyperparameter tuning, we directly calculate its fit quality.

```
# Fit the linear regression model
linear_model <- lm(P ~ . - Model, data = cars_data_clean)

# Summary of the linear regression model
summary(linear_model)
```

```
##
## Call:
## lm(formula = P ~ . - Model, data = cars_data_clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3809.1 -1076.6  -300.2   898.5  4822.1
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12512.212   6406.880   1.953 0.055419 .
## M               7.728     70.660   0.109 0.913267
## R78          -343.323    311.944  -1.101 0.275399
## R77           344.614    273.874   1.258 0.213081
## H            -593.542    361.563  -1.642 0.105819
## R             147.522    103.667   1.423 0.159817
## Tr              2.649     90.360   0.029 0.976708
## W               6.588      1.295   5.086 3.74e-06 ***
## L             -98.155     39.042  -2.514 0.014587 *
## T            -335.623    130.005  -2.582 0.012252 *
## D               5.122      6.410   0.799 0.427327
## G             -87.345    968.673  -0.090 0.928448
## C            1822.714    514.275   3.544 0.000762 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1803 on 61 degrees of freedom
## Multiple R-squared:  0.6853, Adjusted R-squared:  0.6234
## F-statistic: 11.07 on 12 and 61 DF,  p-value: 2.76e-11
```

```
# Predictions for linear regression
linear_preds <- predict(linear_model, cars_data_clean)

# Calculate residuals
linear_residuals <- y - linear_preds

# Sum of squared residuals for linear regression
linear_ssr <- sum(linear_residuals^2)
cat("Linear Regression SSR:", linear_ssr, "\n")
```

```
## Linear Regression SSR: 198296553
```

There is some problem with intercept, so we take it out.

```
# Standardize the predictor variables
X_no_intercept <- X[, -1]

colnames(X_no_intercept)
```

```
##  [1] "M"   "R78" "R77" "H"   "R"   "Tr"  "W"   "L"   "T"   "D"   "G"   "C"
```

```
X_scaled <- scale(X_no_intercept)
```

```
# Identify columns with zero variance
zero_var_cols <- apply(X_scaled, 2, var) == 0

# Remove columns with zero variance
X_scaled_clean <- X_scaled[, !zero_var_cols]

# Check if there are any NaN values after cleaning
any(is.nan(X_scaled_clean))  # This should return FALSE
```

```
## [1] FALSE
```

```
any(is.infinite(X_scaled_clean))
```

```
## [1] FALSE
```

```
str(X_scaled_clean)
```

```
##  num [1:74, 1:12] 0.121 -0.743 0.121 -0.743 0.294 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:74] "1" "2" "3" "4" ...
##   ..$ : chr [1:12] "M" "R78" "R77" "H" ...
```

For Lasso regression, we use cross-validation to find the best regularization parameter lambda to minimize residuals.This code uses the cv.glmnet() function for cross-validation to find the best lambda that minimizes the sum of squared residuals.

```
# Perform Lasso regression with cross-validation
set.seed(123)
lasso_cv <- cv.glmnet(X_scaled_clean, y, alpha = 1)  # alpha = 1 for Lasso

# Get the best lambda that minimizes cross-validation error
best_lambda <- lasso_cv$lambda.min
cat("best lambda:", best_lambda, "\n")
```

```
## best lambda: 26.84201
```

```
# Fit the Lasso model with the optimal lambda
lasso_model <- glmnet(X_scaled_clean, y, alpha = 1, lambda = best_lambda)

# Get the coefficients of the Lasso model
coef(lasso_model)
```

```
## 13 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept)  6192.28378
## M               .
## R78          -224.85423
## R77           269.86604
## H            -445.97461
## R             339.60783
## Tr            -27.38893
## W            4357.41427
## L           -1497.38795
## T           -1302.89338
## D             568.13331
## G               .
## C            1327.99793
```

```
# Predictions for Lasso regression
lasso_preds <- predict(lasso_model, newx = X_scaled_clean)

# Calculate residuals for Lasso regression
lasso_residuals <- y - lasso_preds

# Sum of squared residuals for Lasso regression
lasso_ssr <- sum(lasso_residuals^2)
cat("Lasso Regression SSR:", lasso_ssr, "\n")
```

```
## Lasso Regression SSR: 202502001
```

Summary

Linear Regression: We simply perform the regression and calculate the SSR without tuning any parameters.

Lasso Regression: We use cross-validation to find the optimal regularization parameter lambda to minimize the SSR.