Homework 4

PSTAT Winter 2023

Haocheng Zhang

2023-03-17

```r
library(ISLR)
data(Auto)
```

```r
## Q1(a):
# Qualitative predictors: cylinders, year, origin, (name, but it is excluded
# from the linear regression model)

# Quantitative predictors: displacement, horsepower, weight, acceleration
```

```r
## Q1(b) Fit the MLR model.
lmod<- lm(mpg~. - name, Autod)
# Then use the anova() function in R to perform an analysis of variance
anova_lmod <- anova(lmod)

print(anova_lmod)
```

```
## Analysis of Variance Table
##
## Response: mpg
##               Df  Sum Sq Mean Sq  F value    Pr(>F)
## cylinders      4 15274.5  3818.6 470.9373 < 2.2e-16 ***
## displacement   1  1098.0  1098.0 135.4069 < 2.2e-16 ***
## horsepower     1   588.0   588.0  72.5161  4.25e-16 ***
## weight         1   715.1   715.1  88.1961 < 2.2e-16 ***
## acceleration   1     7.7     7.7   0.9457    0.3315
## year          12  2960.4   246.7  30.4251 < 2.2e-16 ***
## origin         2   183.2    91.6  11.2972  1.73e-05 ***
## Residuals    369  2992.1     8.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Based on the anova table and p-values, we can make the following observations about the
# predictors and their linear association with 'mpg' conditional on the other
# predictors in the model:

# cylinders: p-value < 2.2e-16<0.05, reject the null hypothesis
# displacement: p-value < 2.2e-16<0.05, reject the null hypothesis
# horsepower: p-value = 4.25e-16<0.05, reject the null hypothesis
# weight: p-value < 2.2e-16<0.05, reject the null hypothesis
```

```r
# acceleration: p-value = 0.3315>0.05, fail to reject the null hypothesis
# year: p-value < 2.2e-16<0.05, reject the null hypothesis
# origin: p-value = 1.73e-05<0.05, reject the null hypothesis
```

```r
# Q1(c) To predict the mpg with given specifications, we use predict() function.

# Create a new data frame with the desired values for each predictor:
Japanese_car <- data.frame(
  name = factor("Jpcar", levels = levels(Autod$name)),
  cylinders = factor(3, levels = levels(Autod$cylinders)),
  displacement = 100,
  horsepower = 85,
  weight = 3000,
  acceleration = 20,
  year = factor(80, levels = levels(Autod$year)),
  origin = factor(3, levels = levels(Autod$origin))
)

# Then, use the predict() function with the fitted MLR model
predicted_mpg <- predict(lmod, newdata = Japanese_car)

# Print the predicted mpg
show(predicted_mpg)
```

```
##        1
## 24.64804
```

```r
# Q1(d) We may extract the coefficients from the fitted MLR model,
# and compute the difference
diff_origin <- coef(lmod)["origin3"] - coef(lmod)["origin2"]

# Print the difference between Japanese cars and European cars.
show(diff_origin)
```

```
##   origin3
## 0.5996414
```

```r
# Q1(e) To fit a model to predict mpg using origin, horsepower,
# and their interaction
interaction_model <- lm(mpg ~ origin * horsepower, data = Autod)

# Print the summary of the interaction_model
summary(interaction_model)
```

```
##
## Call:
## lm(formula = mpg ~ origin * horsepower, data = Autod)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.7415  -2.9547  -0.6389   2.3978  14.2495
```

```
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         34.476496   0.890665  38.709  < 2e-16 ***
## origin2             10.997230   2.396209   4.589 6.02e-06 ***
## origin3             14.339718   2.464293   5.819 1.24e-08 ***
## horsepower          -0.121320   0.007095 -17.099  < 2e-16 ***
## origin2:horsepower -0.100515   0.027723  -3.626 0.000327 ***
## origin3:horsepower -0.108723   0.028980  -3.752 0.000203 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.422 on 386 degrees of freedom
## Multiple R-squared:  0.6831, Adjusted R-squared:  0.679
## F-statistic: 166.4 on 5 and 386 DF,  p-value: < 2.2e-16
```

```r
# Based on the summary output of the fitted model, the fitted linear model
# using origin, horsepower, and their interaction can be written as:
# mpg = 34.4765 + 10.9972 * origin2 + 14.3397 * origin3 - 0.1213 * horsepower
#      - 0.1005 * (origin2 * horsepower) - 0.1087 * (origin3 * horsepower)
#      + $\epsilon$
# Here, the variables are:
# mpg: miles per gallon
# origin2: 1 if the car is European, 0 otherwise
# origin3: 1 if the car is Japanese, 0 otherwise
# horsepower: engine horsepower
# $\epsilon$: random error term
```

```r
# Q1(f) use the Akaike Information Criterion (AIC) to compare different degrees
# of polynomial regression models and choose the one with the lowest AIC value.

# Maximum polynomial degree to test
max_degree <- 5

# Create an empty vector to store AIC values
aic_values <- numeric(max_degree)

# Loop through different polynomial degrees and fit models
for (degree in 1:max_degree) {
  poly_model <- lm(mpg ~ poly(weight, degree, raw = TRUE), data = Autod)
  aic_values[degree] <- AIC(poly_model)
}

# Find the degree with the lowest AIC value
best_degree <- which.min(aic_values)

# Print the AIC values and the best degree
print(aic_values)
```

```
## [1] 2265.939 2238.115 2240.113 2241.657 2242.807
```

```r
print(best_degree)
```

```
## [1] 2
```

```r
# From the outcome above, we know 2nd degree is a proper degree of polynomial,
# because it has the lowest AIC value.
```

```r
# Q1(g) Load the MASS package for further use.
library(MASS)

# Full model with all predictors (except name)
full_model <- lm(mpg ~ . - name, data = Autod)

# Perform a backward selection using 'stepAIC()' function
best_model <- stepAIC(full_model, direction = "backward")
```

```
## Start:  AIC=842.72
## mpg ~ (cylinders + displacement + horsepower + weight + acceleration +
##     year + origin + name) - name
##
##                Df Sum of Sq    RSS     AIC
## - acceleration  1      0.01 2992.1  840.72
## <none>                      2992.1  842.72
## - displacement  1     24.70 3016.8  843.94
## - horsepower    1     73.45 3065.5  850.23
## - origin        2    183.21 3175.3  862.02
## - cylinders     4    472.77 3464.8  892.23
## - weight        1    558.60 3550.7  907.82
## - year         12   2831.60 5823.7 1079.78
##
## Step:  AIC=840.72
## mpg ~ cylinders + displacement + horsepower + weight + year +
##     origin
##
##                Df Sum of Sq    RSS     AIC
## <none>                      2992.1  840.72
## - displacement  1     24.88 3017.0  841.97
## - horsepower    1    115.58 3107.7  853.58
## - origin        2    183.45 3175.5  860.05
## - cylinders     4    476.39 3468.5  890.64
## - weight        1    730.02 3722.1  924.31
## - year         12   2841.52 5833.6 1078.45
```

```r
# Print the summary of the best model
summary(best_model)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     year + origin, data = Autod)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.931 -1.671 -0.049  1.448 11.612
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.9706782  1.9703396  15.718  < 2e-16 ***
## cylinders4    6.9489835  1.5189655   4.575 6.51e-06 ***
## cylinders5    6.6467365  2.3240427   2.860 0.004477 **
## cylinders6    4.3050676  1.6932440   2.542 0.011413 *
## cylinders8    6.3723261  1.9615936   3.249 0.001266 **
## displacement  0.0117929  0.0067234   1.754 0.080256 .
## horsepower   -0.0395543  0.0104627  -3.781 0.000182 ***
## weight       -0.0051675  0.0005439  -9.501  < 2e-16 ***
## year71        0.9057500  0.8066633   1.123 0.262236
## year72       -0.4921367  0.8015260  -0.614 0.539593
## year73       -0.5550656  0.7185757  -0.772 0.440340
## year74        1.2376123  0.8470486   1.461 0.144840
## year75        0.8654150  0.8276285   1.046 0.296402
## year76        1.4923994  0.7942429   1.879 0.061027 .
## year77        2.9948793  0.8136242   3.681 0.000267 ***
## year78        2.9703034  0.7736654   3.839 0.000145 ***
## year79        4.8922614  0.8182998   5.979 5.30e-09 ***
## year80        9.0552685  0.8695618  10.414  < 2e-16 ***
## year81        6.4527050  0.8525066   7.569 3.02e-13 ***
## year82        7.8336547  0.8429257   9.293  < 2e-16 ***
## origin2       1.6931856  0.5155093   3.284 0.001119 **
## origin3       2.2936695  0.4957722   4.626 5.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.844 on 370 degrees of freedom
## Multiple R-squared:  0.8744, Adjusted R-squared:  0.8673
## F-statistic: 122.6 on 21 and 370 DF,  p-value: < 2.2e-16
```

```r
# From the output of the summary above, the best model selected by the stepAIC()
# function includes the following predictor variables: cylinders, displacement,
# horsepower, weight, year, and origin. The model has an adjusted R-squared
# value of 0.8673, which indicates a good fit. The AIC value for this model
# is 840.72, lower than the full model with all predictors, suggesting that
# it is a better model according to the AIC criterion.
```

```r
# Q2(a) load the fat dataset:
library(faraway)
data(fat)

# remove every tenth observation for use as a test sample.
test_sample <- fat[seq(10, nrow(fat), by=10),]

# The remaining data will be used as a training sample for futher use:
training_sample <- fat[-seq(10, nrow(fat), by=10),]

# fit a linear regression model with all predictors,
# excluding "brozek" and "density", using the training data:
```

```r
training_model <- lm(siri ~ . - brozek - density, data=training_sample)

summary(training_model)
```

```
##
## Call:
## lm(formula = siri ~ . - brozek - density, data = training_sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8314 -0.6722  0.1828  0.9150  6.6619
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.591885   6.448868  -1.953 0.052193 .
## age           0.007978   0.012320   0.648 0.517983
## weight        0.362999   0.023314  15.570  < 2e-16 ***
## height        0.049026   0.040315   1.216 0.225315
## adipos       -0.514032   0.114074  -4.506 1.09e-05 ***
## free         -0.564773   0.014889 -37.933  < 2e-16 ***
## neck          0.016525   0.089863   0.184 0.854272
## chest         0.120219   0.039590   3.037 0.002694 **
## abdom         0.140108   0.042186   3.321 0.001056 **
## hip           0.006197   0.056101   0.110 0.912148
## thigh         0.195057   0.054460   3.582 0.000424 ***
## knee          0.106637   0.093534   1.140 0.255542
## ankle         0.125118   0.081303   1.539 0.125325
## biceps        0.096199   0.064656   1.488 0.138278
## forearm       0.230775   0.073332   3.147 0.001888 **
## wrist         0.139279   0.206804   0.673 0.501378
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.55 on 211 degrees of freedom
## Multiple R-squared:  0.9692, Adjusted R-squared:  0.967
## F-statistic: 442.5 on 15 and 211 DF,  p-value: < 2.2e-16
```

```r
# Q2(b)
library(MASS)

# Scale all predictors, excluding 'siri', 'brozek', and 'density'
training_sample_scaled <- scale(training_sample[ , !(colnames(training_sample)
                              %in% c("siri", "brozek", "density"))],
                              center = TRUE, scale = TRUE)
training_sample_scaled <- as.data.frame(training_sample_scaled)

# Add 'siri' back
training_sample_scaled$siri <- training_sample$siri

# Use 'lm.ridge()' function to fit a ridge regreassion model
rgmod <- lm.ridge(siri ~ ., data = training_sample_scaled,
                lambda = seq(0, 100, length.out = 100))
```

```
matplot(rgmod$lambda, coef(rgmod), type="l", xlab = "lambda", ylab = "Beta hat"
        , cex=0.8)
```