Final project: Regression

Jose Antonio Alvarez Ocete

For fractions:

```
library(MASS)
options(digits=4)
```

R Markdown

```
#import data
data <- read.table("data/battery.dat")

p <- 5

#set up X
X <- data.matrix(data[,1:p])
n <- length(X[,1])
Z <- cbind(rep(1,n), X)
Y <- data.matrix(data[,p+1])

names <- c('','Z1','Z2','Z3','Z4','Z5')
colnames(Z)<- names

sampleMean<-function(X, n) {
  Ones <- rep(1,n)
  return (1/n * t(X)%*%Ones)
}

sampleCovariance<-function(X, n, sample_mean) {
  Ones <- rep(1,n)
  return (1/(n-1) * t(X - Ones%*%t(sample_mean))%*%(X - Ones%*%t(sample_mean)))
}
```

(1) Find the least square estimate beta_hat

```
# least square estimates
beta_hat <- solve(t(Z)%*%Z)%*%t(Z)%*%Y
rownames(beta_hat) <- c('beta0','beta1','beta2','beta3','beta4','beta5')
beta_hat
```

```
##              [,1]
## beta0 -2937.7571
## beta1   -33.7934
## beta2    -0.1798
## beta3    -1.7397
## beta4     7.0627
## beta5  1529.2897
```

(2) Find the R^2 statistic

```r
# R^2 statistic
R_square <- sum((Y - Z%*%beta_hat)^2)/sum((Y-mean(Y))^2)
R_square
```

```
## [1] 0.4799
```

(3) Find sigma_hat_square and estimated Cov(beta_square)

```r
# sigma_hat_square
sigma_hat_square <- sum((Y - Z%*%beta_hat)^2)/(n-p-1)
cat('Sigma hat square:', sigma_hat_square, fill=TRUE)
```

```
## Sigma hat square: 7138
```

```r
# estimated covariance of hat{beta}
covariance_hat <- sigma_hat_square * solve(t(Z)%*%Z)
cat('Covariance Hat:', fill=TRUE)
```

```
## Covariance Hat:
```

```r
covariance_hat
```

```
##                     Z1        Z2        Z3         Z4         Z5
##     1.633e+07 -2933.74  2980.4460 -34.78143 -991.73637 -8.160e+06
## Z1 -2.934e+03  1880.55    18.5503  17.34897   10.28445 -1.764e+02
## Z2  2.980e+03    18.55   193.4117  -3.23257    0.34449 -1.696e+03
## Z3 -3.478e+01    17.35    -3.2326   1.79944   -0.08092 -4.242e+01
## Z4 -9.917e+02    10.28     0.3445  -0.08092    3.89193  4.549e+02
## Z5 -8.160e+06  -176.39 -1695.7845 -42.42251  454.86652  4.081e+06
```

NOT USED:

```r
# t-test for single coefficient
# H_0: beta_j = 0, H_a: beta_j != 0

j <- 1
t_stat <- (beta_hat[j+1] - 0)/sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j+1])
t_stat
```

```
## [1] -0.7793
```

```r
alpha <- 0.05
cval_t <- qt(1-alpha/2, n-p-1)
cval_t
```

```
## [1] 2.145
```

(4) 95% confidence interval for the beta:

```r
# One-at-a-time confidence interval for beta_j

for (j in 0:p) {
  cat('Beta hat', j,': [',
    beta_hat[j+1] - qt(1-alpha/2, n-p-1)*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j+1]),
    ',',
    beta_hat[j+1] + qt(1-alpha/2, n-p-1)*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j+1]),
    ']', fill=TRUE)
}
```

```
## Beta hat 0 : [ -11604 , 5729 ]
## Beta hat 1 : [ -126.8 , 59.22 ]
## Beta hat 2 : [ -30.01 , 29.65 ]
## Beta hat 3 : [ -4.617 , 1.137 ]
## Beta hat 4 : [ 2.831 , 11.29 ]
## Beta hat 5 : [ -2804 , 5862 ]
```

(5) 95% simultaneous confidence intervals for all betas based on the confidence region

```
# confidence region based simultaneous confidence intervals

for (j in 0:p) {
cat('Beta hat', j,': [',
      beta_hat[j+1] - sqrt((p+1)*qf(1-alpha, p+1, n-p-1))*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j
      ',',
      beta_hat[j+1] + sqrt((p+1)*qf(1-alpha, p+1, n-p-1))*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j
      ']', fill=TRUE)
}
```

```
## Beta hat 0 : [ -19640 , 13764 ]
## Beta hat 1 : [ -213 , 145.5 ]
## Beta hat 2 : [ -57.67 , 57.31 ]
## Beta hat 3 : [ -7.285 , 3.805 ]
## Beta hat 4 : [ -1.092 , 15.22 ]
## Beta hat 5 : [ -6822 , 9880 ]
```

(6) 95% simultaneous confidence intervals for all betas based on the Bonferroni correction

```
# Bonferroni correction based simultaneous confidence intervals

for (j in 0:p) {
  cat('Beta hat', j,': [',
    beta_hat[j+1] - qt(1-alpha/(2*(p+1)), n-p-1)*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j+1]),
    ',',
    beta_hat[j+1] + qt(1-alpha/(2*(p+1)), n-p-1)*sqrt(sigma_hat_square * solve(t(Z)%*%Z)[j+1,j+1]),
    ']', fill=TRUE)
}
```

```
## Beta hat 0 : [ -15338 , 9462 ]
## Beta hat 1 : [ -166.9 , 99.29 ]
## Beta hat 2 : [ -42.86 , 42.5 ]
## Beta hat 3 : [ -5.856 , 2.377 ]
## Beta hat 4 : [ 1.009 , 13.12 ]
## Beta hat 5 : [ -4670 , 7729 ]
```

(7) Test H_0: beta_1 = beta_2 = 0 at significance level alpha = 0.05

```
# F-test
# H_0: beta_1 = beta_2 = 0

C <- cbind(rep(0,p), diag(p))

df_1 <- qr(C)$rank # df_1: rank of matrix R

f_stat <- t(C%*%beta_hat)%*%solve(C%*%solve(t(Z)%*%Z)%*%t(C))%*%(C%*%beta_hat)

cval_f <- sigma_hat_square*df_1*qf(1-alpha, 2, n-p-1)
```

```r
cat (f_stat, '<', cval_f, '?', fill=TRUE)
```

```
## 108296 < 133445 ?
```

```r
# (equivalent) F-test by comparing residuals

# fit the reduced model
beta_hat_reduced <- solve(t(Z[,1])%*%Z[,1])%*%t(Z[,1])%*%Y

f_stat_reduced <- ((sum((Y - Z[,1]%*%beta_hat_reduced)^2) - sum((Y - Z%*%beta_hat)^2))/2)

cat (f_stat_reduced, '<', cval_f, '?', fill=TRUE)
```

```
## 54148 < 133445 ?
```

(8) 95% confidence interval for the mean response given z_0

```r
# confidence interval for z_0^T beta

X_mean <- sampleMean(X,n)
z_0 <- rbind(1, X_mean)

cat('C.I. for the mean response z_0[',
  t(z_0)%*%beta_hat - sqrt(sigma_hat_square)*sqrt(t(z_0)%*%solve(t(Z)%*%Z)%*%z_0)*qt(1-alpha/2, n-p-1),
  ',',
  t(z_0)%*%beta_hat + sqrt(sigma_hat_square)*sqrt(t(z_0)%*%solve(t(Z)%*%Z)%*%z_0)*qt(1-alpha/2, n-p-1),
  ']')
```

```
## C.I. for the mean response z_0[ 71.78 , 152.8 ]
```

(9) 95% confidence interval for a new response given z_0

```r
# prediction interval for Y_0 = z_0^T beta + epsilon_0

cat('Prediction interval for Y_0 = z_0^T beta + epsilon_0: [',
  t(z_0)%*%beta_hat - sqrt(sigma_hat_square)*sqrt(1+t(z_0)%*%solve(t(Z)%*%Z)%*%z_0)*qt(1-alpha/2, n-p-1
  ',',
  t(z_0)%*%beta_hat + sqrt(sigma_hat_square)*sqrt(1+t(z_0)%*%solve(t(Z)%*%Z)%*%z_0)*qt(1-alpha/2, n-p-1
  ']')
```

```
## Prediction interval for Y_0 = z_0^T beta + epsilon_0: [ -73.38 , 298 ]
```