STAT 420: Project 1

Nico Kienawan, nicok2

4/13/2020

Part 1

```
library(faraway)
data(prostate)

RMSE = function(model) {
  sqrt(mean(resid(model) ^ 2))
}

model_1 = lm(lpsa ~ ., data = prostate) # all possible predictors
model_2 = lm(lpsa ~ lcavol, data = prostate) # only lcavol
model_3 = lm(lpsa ~ lcavol + lweight + svi, data = prostate) # best
model_4 = lm(lpsa ~ lcavol + lweight + age + lbph, data = prostate)
model_5 = lm(lpsa ~ lcavol + lweight + svi + lbph + lcp + pgg45, data = prostate)

rs_1 = summary(model_1)$r.squared
rs_2 = summary(model_2)$r.squared
rs_3 = summary(model_3)$r.squared
rs_4 = summary(model_4)$r.squared
rs_5 = summary(model_5)$r.squared
rs = c(rs_1, rs_2, rs_3, rs_4, rs_5)

rmse_1 = RMSE(model_1)
rmse_2 = RMSE(model_2)
rmse_3 = RMSE(model_3)
rmse_4 = RMSE(model_4)
rmse_5 = RMSE(model_5)
rmse = c(rmse_1, rmse_2, rmse_3, rmse_4, rmse_5)

summary(model_1)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = prostate)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7331 -0.3713 -0.0170  0.4141  1.6381
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.669337   1.296387   0.516  0.60693
## lcavol       0.587022   0.087920   6.677 2.11e-09 ***
## lweight      0.454467   0.170012   2.673  0.00896 **
## age         -0.019637   0.011173  -1.758  0.08229 .
## lbph         0.107054   0.058449   1.832  0.07040 .
## svi          0.766157   0.244309   3.136  0.00233 **
## lcp         -0.105474   0.091013  -1.159  0.24964
## gleason      0.045142   0.157465   0.287  0.77503
## pgg45        0.004525   0.004421   1.024  0.30886
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7084 on 88 degrees of freedom
## Multiple R-squared:  0.6548, Adjusted R-squared:  0.6234
## F-statistic: 20.86 on 8 and 88 DF,  p-value: < 2.2e-16
```

```
cbind(rs, rmse)
```

```
##             rs      rmse
## [1,] 0.6547541 0.6747510
## [2,] 0.5394319 0.7793386
## [3,] 0.6264403 0.7018742
## [4,] 0.5975750 0.7284869
## [5,] 0.6425760 0.6865484
```

I think model 3 (predictors: lcavol, lweight, svi) is the best model because the r squared is not that low and the RMSE is not that high compared to the other models. According to the table provided by `summary(model_1)`, the p-value of these 3 predictors are the lowest (<0.005), which means there is a significant relationship between these predictors and lpsa.

Part 2

```
library(MASS)
data(Boston)

set.seed(42)
train_index = sample(1:nrow(Boston), 400)
train = Boston[train_index,]
test = Boston[-train_index,]

RMSE_2 = function(model, data) {
  yi = data[,c(length(data))]
  yi_hat = predict(model, data)
  sqrt(sum((yi_hat - yi)^2)/nrow(data))
}

model_1 = lm(medv ~ ., data = train) # all possible predictors
model_2 = lm(medv ~ crim, data = train) # only crim
```

```r
model_3 = lm(medv ~ crim + nox + rm + dis + rad + ptratio + lstat, data = train)
model_4 = lm(medv ~ crim + nox + rm + dis + rad + ptratio + lstat + tax + zn + black, data = train) # b
model_5 = lm(medv ~ tax + zn + black, data = train)

train1_rmse = RMSE_2(model_1, train)
train2_rmse = RMSE_2(model_2, train)
train3_rmse = RMSE_2(model_3, train)
train4_rmse = RMSE_2(model_4, train)
train5_rmse = RMSE_2(model_5, train)
train_rmse = c(train1_rmse, train2_rmse, train3_rmse, train4_rmse, train5_rmse)

test1_rmse = RMSE_2(model_1, test)
test2_rmse = RMSE_2(model_2, test)
test3_rmse = RMSE_2(model_3, test)
test4_rmse = RMSE_2(model_4, test)
test5_rmse = RMSE_2(model_5, test)
test_rmse = c(test1_rmse, test2_rmse, test3_rmse, test4_rmse, test5_rmse)

summary(model_1)
```

```
##
## Call:
## lm(formula = medv ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.3126  -2.7134  -0.5522   1.5431  25.5431
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  40.211363   5.823305   6.905 2.07e-11 ***
## crim         -0.121911   0.034032  -3.582 0.000384 ***
## zn            0.037754   0.016166   2.335 0.020038 *
## indus         0.002787   0.069150   0.040 0.967867
## chas          1.918167   0.999327   1.919 0.055663 .
## nox         -17.987178   4.304668  -4.179 3.63e-05 ***
## rm            3.478935   0.457299   7.608 2.16e-13 ***
## age          -0.003087   0.014798  -0.209 0.834880
## dis          -1.456826   0.230828  -6.311 7.60e-10 ***
## rad           0.310637   0.074539   4.167 3.81e-05 ***
## tax          -0.011081   0.004234  -2.617 0.009212 **
## ptratio      -0.996107   0.148701  -6.699 7.45e-11 ***
## black         0.007692   0.003214   2.393 0.017194 *
## lstat        -0.533910   0.055318  -9.652  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.759 on 386 degrees of freedom
## Multiple R-squared:  0.7262, Adjusted R-squared:  0.7169
## F-statistic: 78.73 on 13 and 386 DF,  p-value: < 2.2e-16
```

```
cbind(train_rmse, test_rmse)
```

```
##      train_rmse test_rmse
## [1,]   4.675465  4.767746
## [2,]   8.238496  9.318085
## [3,]   4.812485  5.232587
## [4,]   4.698149  4.884765
## [5,]   7.705864  8.000684
```

I think model 4 (predictors: crim, nox, rm, dis, rad, ptratio, lstat, tax, zn, black) is the best model because the RMSE is the second lowest. According to the table provided by `summary(model_1)`, the p-value of these predictors are < 0.001, which means there is a significant relationship between these predictors and medv.

Part 3

(a)

```
set.seed(42)
n = 25

x0 = rep(1, n)
x1 = runif(n, 0, 10)
x2 = runif(n, 0, 10)
x3 = runif(n, 0, 10)
x4 = runif(n, 0, 10)
X = cbind(x0, x1, x2, x3, x4)
C = solve(t(X) %*% X)
y = rep(0, n)
ex_4_data = data.frame(y, x1, x2, x3, x4)

diag(C)
```

```
##          x0          x1          x2          x3          x4
## 0.744784994 0.004573055 0.005091328 0.005898213 0.005058979
```

```
ex_4_data[10,]
```

```
##    y       x1         x2       x3       x4
## 10 0 7.050648 0.03948339 5.144129 7.758234
```

(b)

```
beta_hat_1 = numeric(1500)
beta_2_pval = numeric(1500)
beta_3_pval = numeric(1500)
```

(c)

```
for (i in 1:1500) {
  ex_4_data[, 1] = 2 + 3*x1 + 4*x2 + 0*x3 + 1*x4 + rnorm(25, 0, 4)
  y = ex_4_data[,1]
  model = lm(y ~ x1 + x2 + x3 + x4)
  beta_hat_1[i] = summary(model)$coef[2]
  beta_2_pval[i] = summary(model)$coef[3,4]
  beta_3_pval[i] = summary(model)$coef[4,4]
}
```

(d)

```
true_var = (16*C)[2,2]
true_var
```

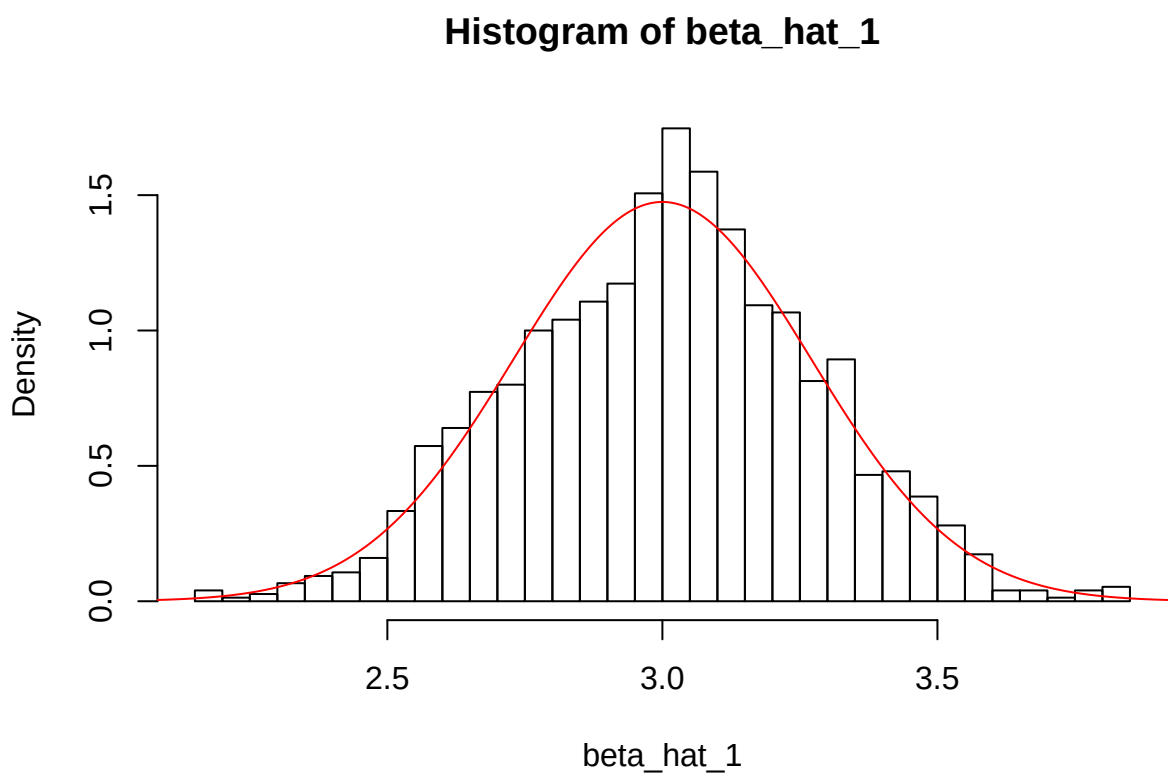
```
## [1] 0.07316889
```

True distribution of $\hat{\beta}_1$ is mean: 3 and variance: 0.0731689.

(e)

```
beta_hat_1_mean = mean(beta_hat_1)
beta_hat_1_var = var(beta_hat_1)

hist(beta_hat_1, prob = TRUE, breaks = 50)
x = seq(0, 6, length=1000)
y = dnorm(x, 3, sqrt(true_var))
lines(x,y, col = "red")
```



```r
beta_hat_1_mean
```

```
## [1] 3.006391
```

```r
beta_hat_1_var
```

```
## [1] 0.07303341
```

The mean and variance of `beta_hat_1` are close to 3 and 0.0731689.
The curve also matches the histogram.

(f)

```r
prop = mean(beta_3_pval < 0.05)
prop
```

```
## [1] 0.04666667
```

The proportion is 0.0466667, which means most of the tests fail to reject the null hypothesis ($H_0 : \beta_3 = 0$) with $\alpha = 0.05$.
It is expected because the true value of β_3 is 0.

(g)

```r
mean(beta_2_pval < 0.05)
```

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

The proportion is 1, which means all of the tests reject the null hypothesis ($H_0 : \beta_2 = 0$) with $\alpha = 0.05$.
It is expected because the true value of β_2 is 4, which is far from 0.