CL7 Assignment5

43141 (Sahil Naphade)

10/10/2020

Adding the libraries and reading the file

```
library("lattice")
library("Metrics")
library("DAAG")
library("leaps")
library("ISLR")
#df<-df("advertising.csv")
df <- read.csv("./advertising.csv")
head(df)
```

```
##      TV Radio Newspaper Sales
## 1 230.1  37.8      69.2  22.1
## 2  44.5  39.3      45.1  10.4
## 3  17.2  45.9      69.3  12.0
## 4 151.5  41.3      58.5  16.5
## 5 180.8  10.8      58.4  17.9
## 6   8.7  48.9      75.0   7.2
```

```
dim(df) # Dimensions of the dataset
```

```
## [1] 200   4
```

Divide the dataset into training and testing

```
train_dataset=df[1:120,]
head(train_dataset)
```

```
##      TV Radio Newspaper Sales
## 1 230.1  37.8      69.2  22.1
## 2  44.5  39.3      45.1  10.4
## 3  17.2  45.9      69.3  12.0
## 4 151.5  41.3      58.5  16.5
## 5 180.8  10.8      58.4  17.9
## 6   8.7  48.9      75.0   7.2
```

```
test_dataset=df[121:200,]
head(test_dataset)
```

```
##        TV Radio Newspaper Sales
## 121 141.3  26.8      46.2  15.5
## 122  18.8  21.7      50.4   7.0
## 123 224.0   2.4      15.6  16.6
## 124 123.1  34.6      12.4  15.2
## 125 229.5  32.3      74.2  19.7
## 126  87.2  11.8      25.9  10.6
```

Columns for binding for predictioin

```
S=cbind("TV","Newspaper","Radio")
```

Linear regression using Least Square method on TV

```
TV1=lm(Sales~TV,data=train_dataset)
TV1
```

```
##
## Call:
## lm(formula = Sales ~ TV, data = train_dataset)
##
## Coefficients:
## (Intercept)           TV
##     7.22851      0.05516
```

```
  # Display the attributes of Linear regression of TV
attributes(TV1)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

```
TV1$coefficients[1]
```

```
## (Intercept)
##    7.228508
```

```
TV1$coefficients[2]
```

```
##         TV
## 0.05516365
```

Linear regression using Least Square method on Radio

```
Radio1=lm(Sales~Radio,data=train_dataset)
Radio1
```

```
##
## Call:
## lm(formula = Sales ~ Radio, data = train_dataset)
##
## Coefficients:
## (Intercept)        Radio
##     12.0555       0.1394
```

```
  # Display the attributes of Linear regression of Radio
attributes(Radio1)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
```

```
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

```
Radio1$coefficients[1]
```

```
## (Intercept)
##    12.05553
```

```
Radio1$coefficients[2]
```

```
##     Radio
## 0.1394216
```

Linear regression using Least Square method on Newspapers

```
# Linear regression using Least Square method on Newspapers
Newspaper1=lm(Sales~Newspaper,data=train_dataset)
Newspaper1
```

```
##
## Call:
## lm(formula = Sales ~ Newspaper, data = train_dataset)
##
## Coefficients:
## (Intercept)    Newspaper
##    14.54379      0.02681
```

```
  # Display the attributes of Linear regression of Radio
attributes(Newspaper1)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

```
Newspaper1$coefficients[1]
```

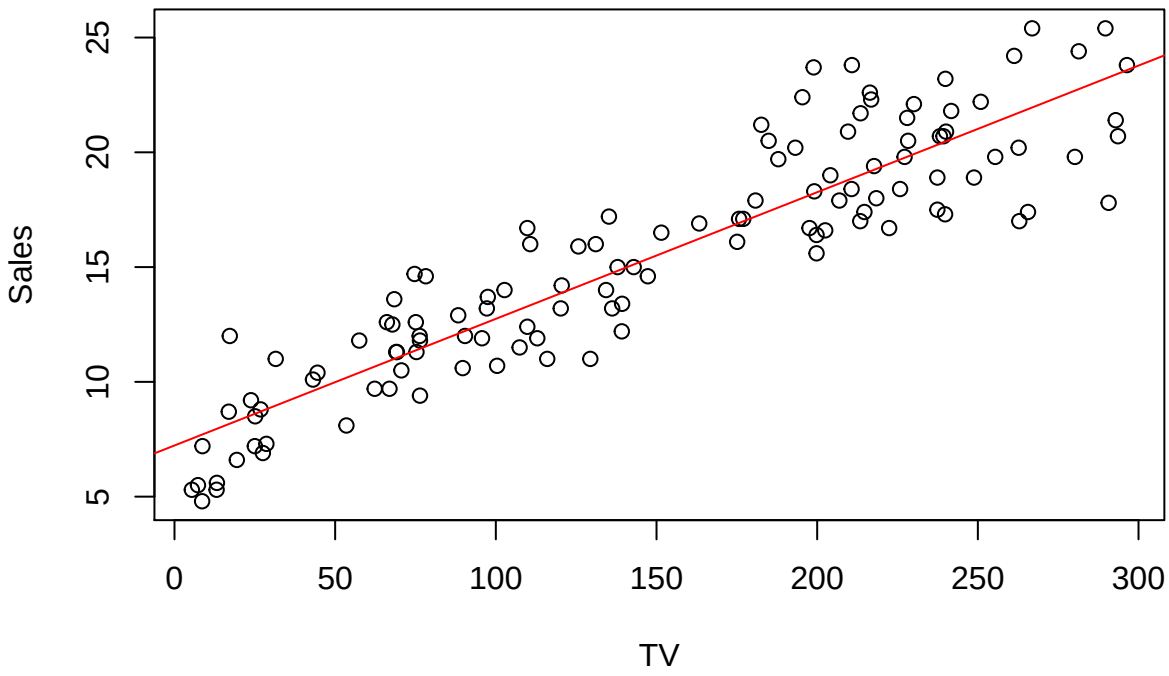
```
## (Intercept)
##    14.54379
```

```
Newspaper1$coefficients[2]
```

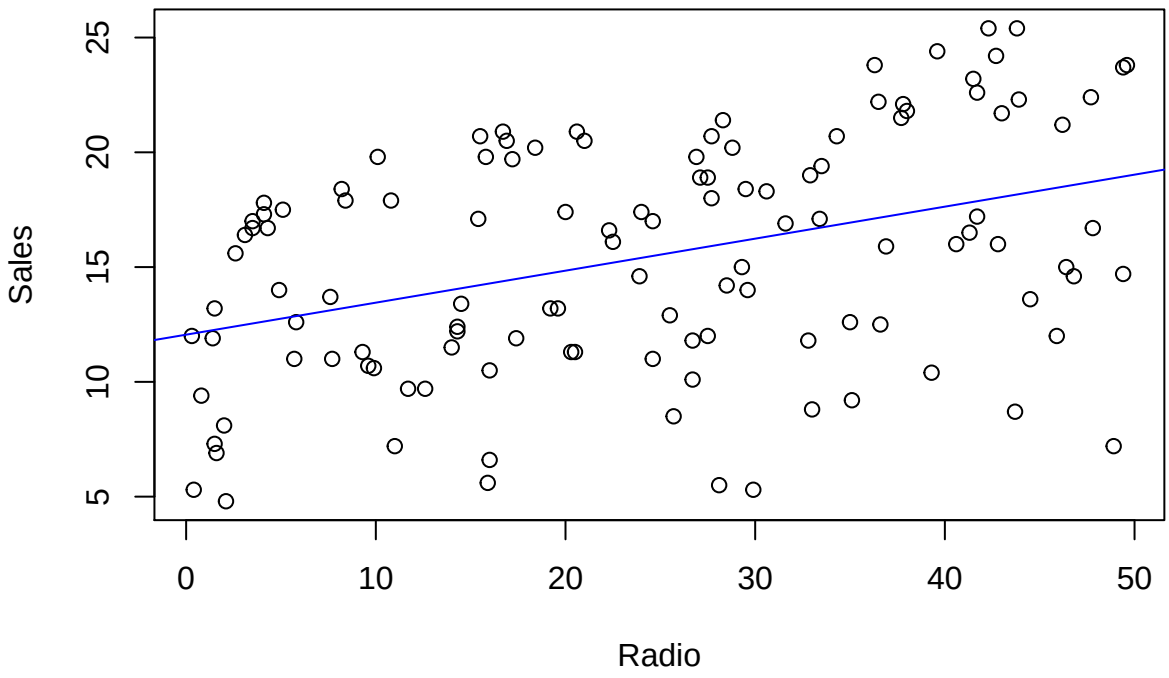
```
##  Newspaper
## 0.02681309
```

Plotting graphs of parts of dataset vs sales

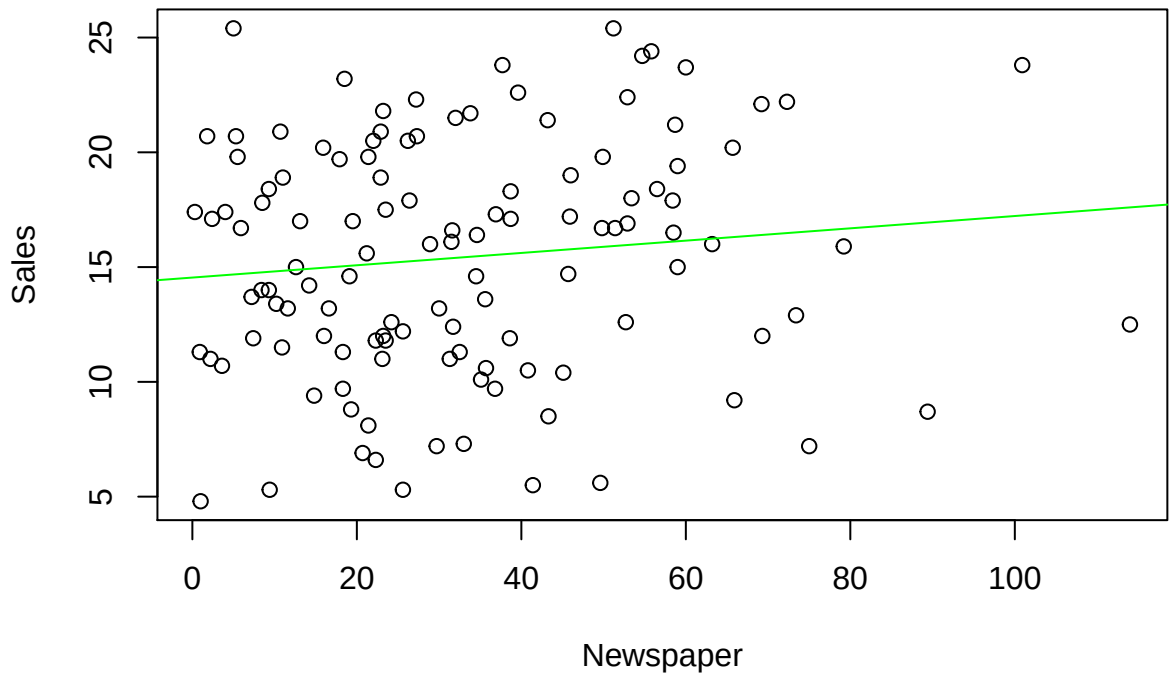
```
# 1. With TV
plot(train_dataset$Sales~train_dataset$TV,xlab="TV",ylab = "Sales")
abline(TV1, col="red")
```




```
# 2. With Radio
plot(train_dataset$Sales~train_dataset$Radio,xlab="Radio",ylab = "Sales")
abline(Radio1, col="blue")
```




```
# 3. with Newspaper
plot(train_dataset$Sales~train_dataset$Newspaper,xlab="Newspaper",ylab="Sales")
abline(Newspaper1, col="green")
```




```
summary(TV1)
```

```
##
## Call:
## lm(formula = Sales ~ TV, data = train_dataset)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4646 -1.5615  0.1003  1.4325  5.4994
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.228508   0.393172   18.39   <2e-16 ***
## TV          0.055164   0.002302   23.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.125 on 118 degrees of freedom
## Multiple R-squared:  0.8296, Adjusted R-squared:  0.8281
## F-statistic: 574.3 on 1 and 118 DF,  p-value: < 2.2e-16
```

```
#tidy(TV1)

Tvp=predict(TV1,train_dataset)
Radiop=predict(Radio1,train_dataset)
Newspaperp=predict(Newspaper1,train_dataset)
```

```
# Predict on the test data-set
Tvt=predict(TV1,test_dataset)
Radiot=predict(Radio1,test_dataset)
Newspapert=predict(Newspaper1,test_dataset)
```

Finding mean square error on Training dataset

```
TVtrain_mse=mse(train_dataset$Sales,Tvp)
TVtrain_mse
```

```
## [1] 4.438338
```

```
Radiotrain_mse=mse(train_dataset$Sales,Radiop)
Radiotrain_mse
```

```
## [1] 21.89934
```

```
Newspapertrain_mse=mse(train_dataset$Sales,Newspaperp)
Newspapertrain_mse
```

```
## [1] 25.68104
```

Finding mean square error on testing dataset

```
TVtest_mse=mse(test_dataset$Sales,Tvt)
TVtest_mse
```

```
## [1] 6.498146
```

```
Radiotest_mse=mse(test_dataset$Sales,Radiot)
Radiotest_mse
```

```
## [1] 28.30533
```

```
Newspapertest_mse=mse(test_dataset$Newspaper,Newspapert)
Newspapertest_mse
```

```
## [1] 526.0659
```

Combines the arguments to form a vector

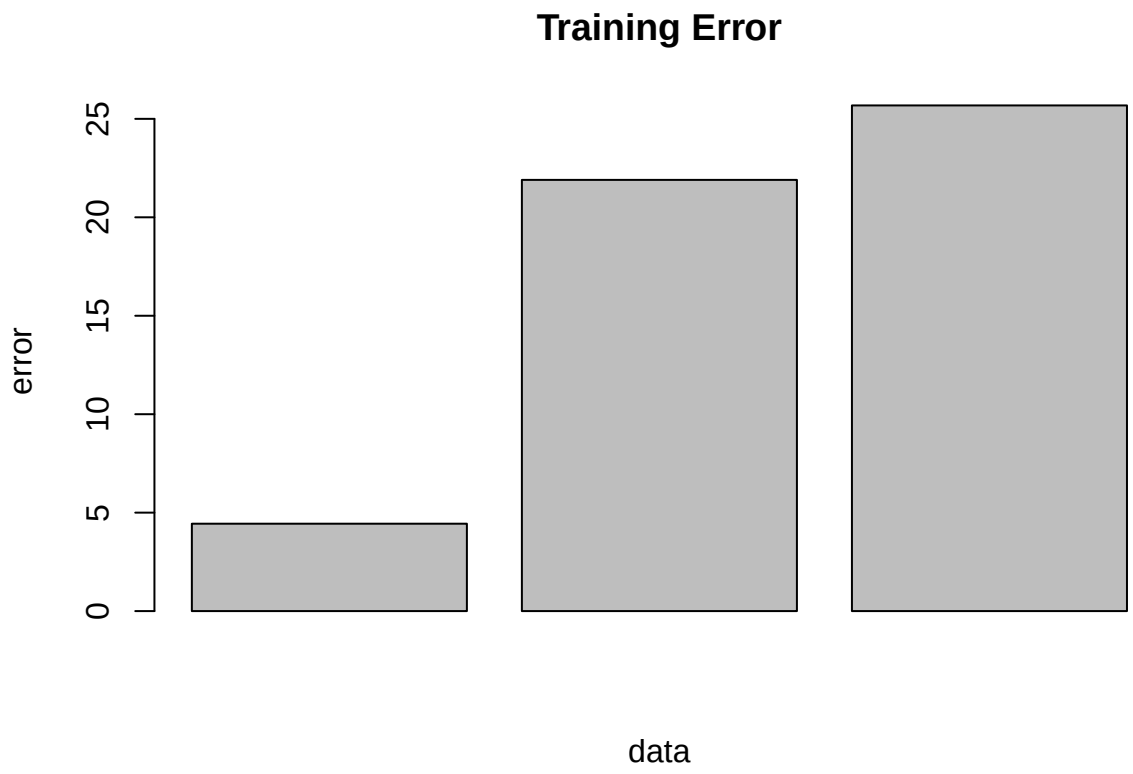
```
TrainMSE=c(TVtrain_mse,Radiotrain_mse,Newspapertrain_mse)
TrainMSE
```

```
## [1]  4.438338 21.899336 25.681035
```

```
TestMSE=c(TVtest_mse,Radiotest_mse,Newspapertest_mse)
TestMSE
```

```
## [1]   6.498146  28.305334 526.065890
```

```
barplot(TrainMSE,width = 0.02,xlab="data",ylab="error",main="Training Error")
```




```
barplot(TestMSE,width=0.02,xlab = "data",ylab="error",main="Testing Error")
```

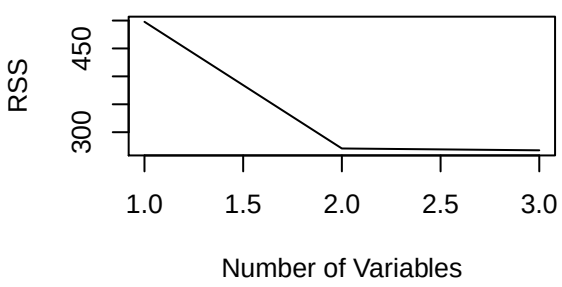
Using Subset selection method

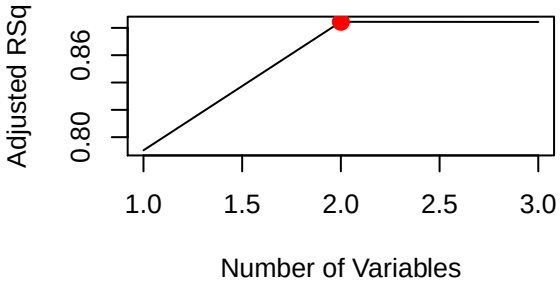
```
regfit_full = regsubsets(Sales~., data= test_dataset)
reg_summary = summary(regfit_full)
reg_summary$rsq
```

```
## [1] 0.7930429 0.8873720 0.8887412
```

Set up a 2x2 grid so we can look at 4 plots at once

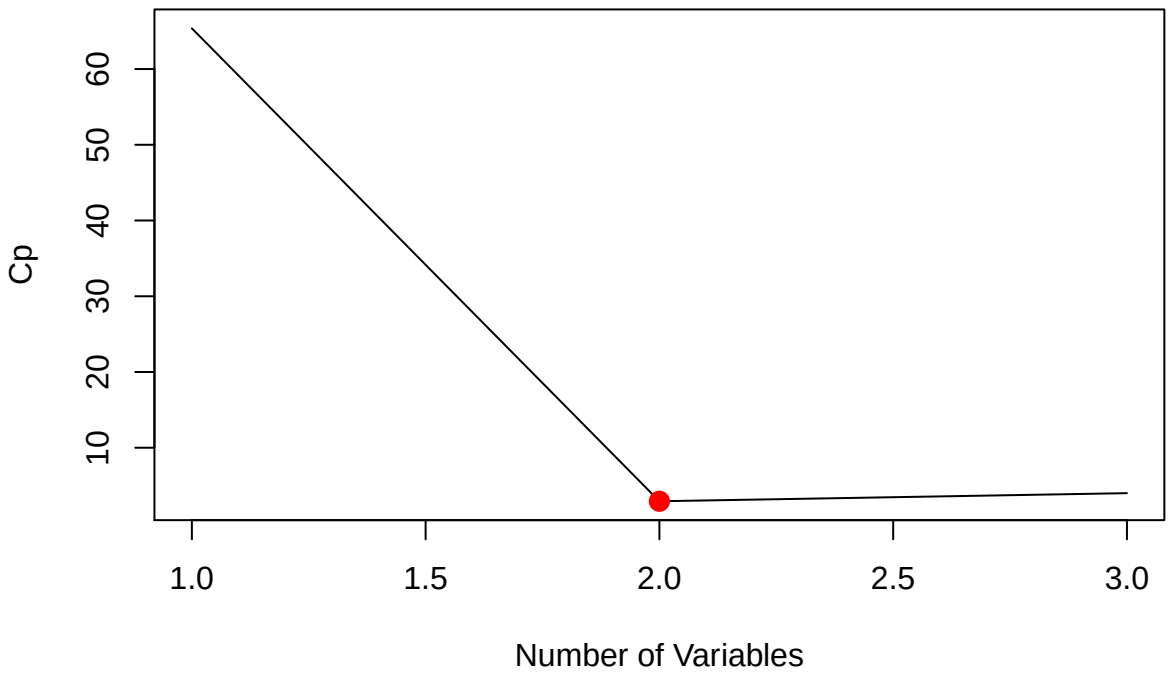
```
par(mfrow = c(2,2))
plot(reg_summary$rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(reg_summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")
# We will now plot a red dot to indicate the model with the largest adjusted R^2 statistic.
# The which.max() function can be used to identify the location of the maximum point of a vector
adj_r2_max = which.max(reg_summary$adjr2) # 11

# The points() command works like the plot() command, except that it puts points
# on a plot that has already been created instead of creating a new plot
points(adj_r2_max, reg_summary$adjr2[adj_r2_max], col ="red", cex = 2, pch = 20)
```



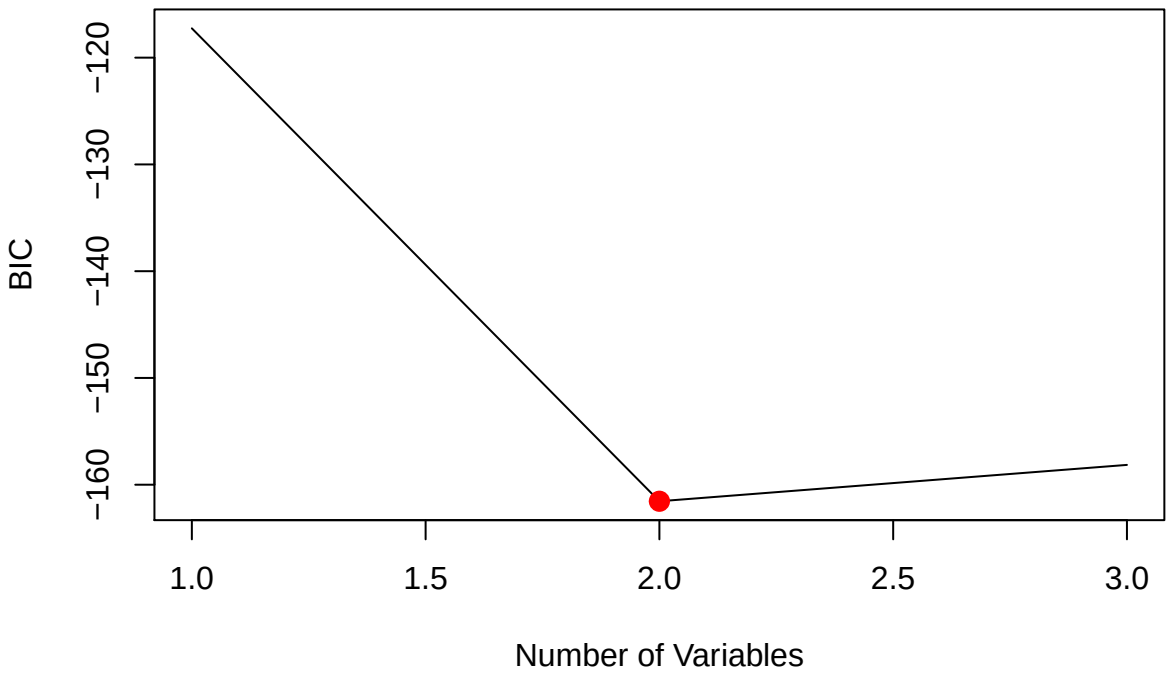


We'll do the same for C_p and BIC, this time looking for the models with the SMALLEST statistic

```
plot(reg_summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
cp_min = which.min(reg_summary$cp) # 10
points(cp_min, reg_summary$cp[cp_min], col = "red", cex = 2, pch = 20)
```




```
plot(reg_summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
bic_min = which.min(reg_summary$bic) # 6
points(bic_min, reg_summary$bic[bic_min], col = "red", cex = 2, pch = 20)
```


BIC
−120
−130
−140
−150
−160
1.0
1.5
2.0
2.5
3.0
Number of Variables