Kunal Mehta
50352960
kunalkam

Homework 2

Problem 1

a. This data was provided to build a predictive model for the rating variable. The first step was to clear the environment and load the MASS and leaps libraries. I downloaded and stored the dataset locally, and since it is a CSV file, I next used the read.csv command to load the file and examined it a bit.

```
> names(dats)
 [1] "name"     "mfr"      "type"     "calories" "protein"  "fat"      "sodium"   "fiber"
 [9] "carbo"    "sugars"   "potass"   "vitamins" "shelf"    "weight"   "cups"     "rating"
> head(dats)
                       name mfr type calories protein fat sodium fiber carbo sugars potass
1                 100% Bran   N    C       70       4   1    130  10.0   5.0      6    280
2         100% Natural Bran   Q    C      120       3   5     15   2.0   8.0      8    135
3                  All-Bran   K    C       70       4   1    260   9.0   7.0      5    320
4 All-Bran with Extra Fiber   K    C       50       4   0    140  14.0   8.0      0    330
5            Almond Delight   R    C      110       2   2    200   1.0  14.0      8     -1
6   Apple Cinnamon Cheerios   G    C      110       2   2    180   1.5  10.5     10     70
  vitamins shelf weight cups   rating
1       25     3      1 0.33 68.40297
2        0     3      1 1.00 33.98368
3       25     3      1 0.33 59.42551
4       25     3      1 0.50 93.70491
5       25     3      1 0.75 34.38484
6       25     1      1 0.75 29.50954
> |
```

I divided the data next, with a 70% training and 30% testing ratio, setting a seed of 123 such that my results can be replicated. Since the mfr and type variables are categorical, and since there isn't much variance in type, I have removed the first the 3 columns. I next fit a simple linear model to rating, and then used the predict function. To calculate the mean square error, I took the mean of the square of the residuals component. My result is 6.6207e-14.

```
> print(MSE)
[1] 6.620702e-14
> |
```

b. For part b, I performed forward subset selection. The default value for method in the regsubsets function is exhaustive, as mentioned in the lab, therefore I gave the value forward to the method parameter. A summary of the selection shows the best subset selections with different number of variables.

```
shelf        FALSE      FALSE
weight       FALSE      FALSE
cups         FALSE      FALSE
1 subsets of each size up to 12
Selection Algorithm: forward
          calories protein fat sodium fiber carbo sugars potass vitamins shelf weight cups
1  ( 1 )  " "      " "     " " " "    " "   " "   "*"    " "    " "      " "   " "    " "
2  ( 1 )  " "      " "     " " " "    "*"   " "   "*"    " "    " "      " "   " "    " "
3  ( 1 )  " "      " "     " " "*"    "*"   " "   "*"    " "    " "      " "   " "    " "
4  ( 1 )  " "      " "     "*" "*"    "*"   " "   "*"    " "    " "      " "   " "    " "
5  ( 1 )  " "      "*"     "*" "*"    "*"   " "   "*"    " "    " "      " "   " "    " "
6  ( 1 )  " "      "*"     "*" "*"    "*"   " "   "*"    " "    "*"      " "   " "    " "
7  ( 1 )  " "      "*"     "*" "*"    "*"   " "   "*"    "*"    "*"      " "   " "    " "
8  ( 1 )  " "      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " "
9  ( 1 )  "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " "
10  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   " "    " "
11  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   " "    "*"
12  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   "*"    "*"
> |
```

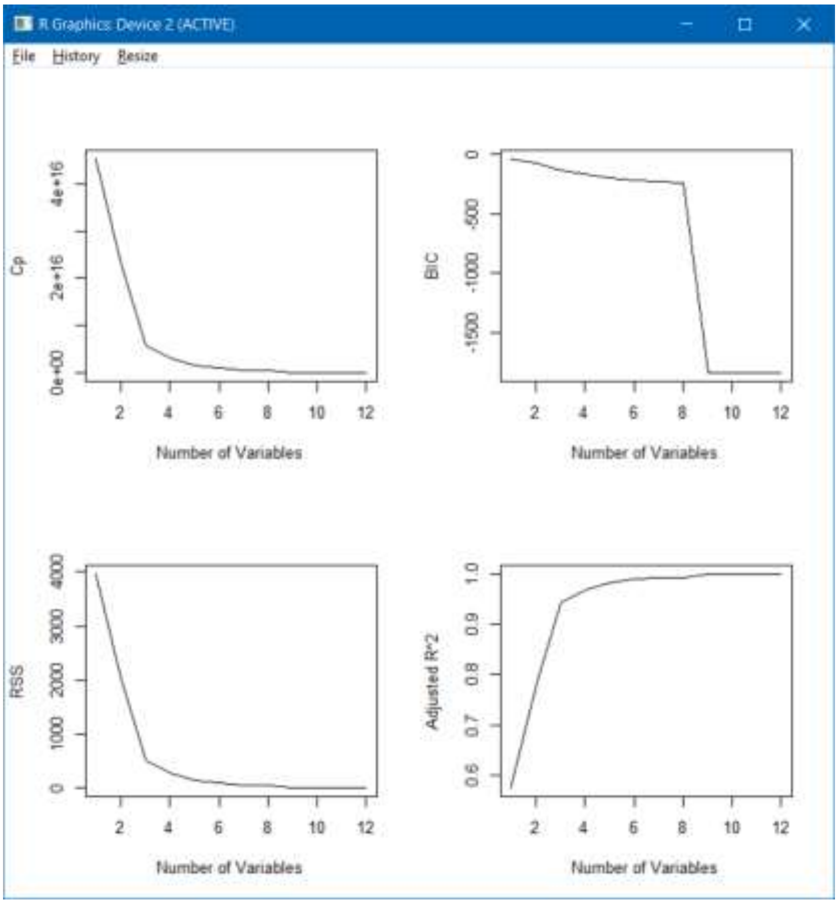
I experimented a bit to see if changing the number of best subsets for each size revealed anything unexpected, but it didn't, so I let the best single subset of each size stay.

Kunal Mehta
50352960
kunalkam

c. For part c as well I did the same as for part b.

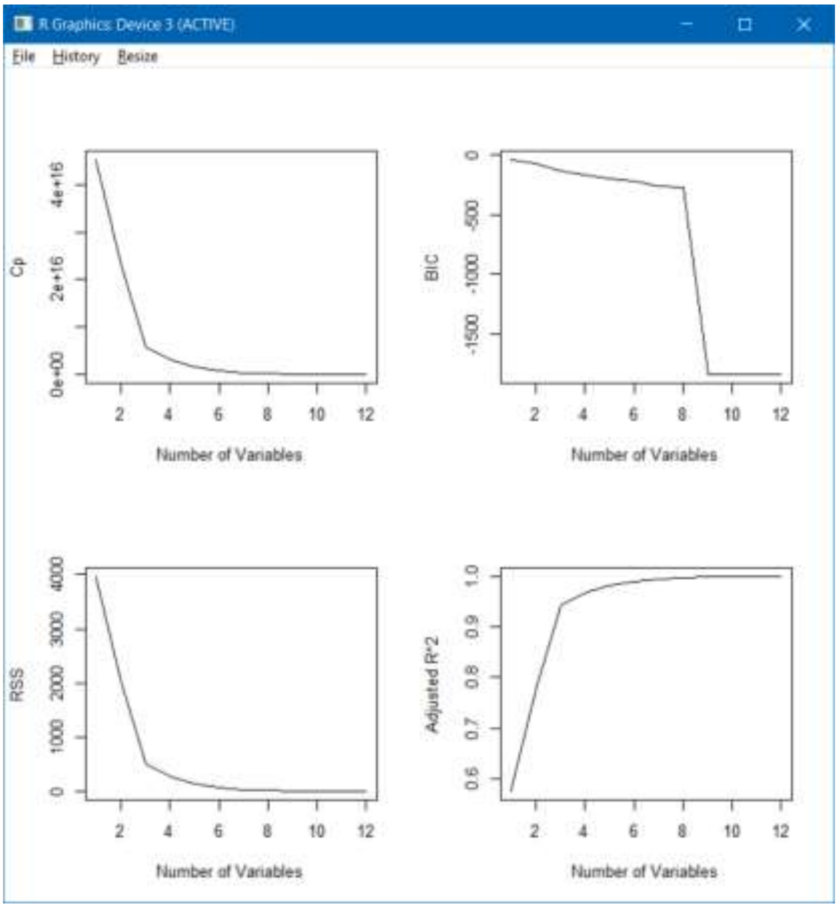
```
shelf          FALSE      FALSE
weight         FALSE      FALSE
cups           FALSE      FALSE
1 subsets of each size up to 12
Selection Algorithm: exhaustive
          calories protein fat sodium fiber carbo sugars potass vitamins shelf weight cups
1  ( 1 )  " "      " "     " " " "    " "   " "   "*"    " "    " "      " "   " "    " " 
2  ( 1 )  " "      " "     " " " "    "*"   " "   "*"    " "    " "      " "   " "    " " 
3  ( 1 )  " "      " "     " " "*"    "*"   " "   "*"    " "    " "      " "   " "    " " 
4  ( 1 )  " "      " "     "*" "*"    "*"   " "   "*"    " "    " "      " "   " "    " " 
5  ( 1 )  " "      "*"     "*" "*"    "*"   " "   "*"    " "    " "      " "   " "    " " 
6  ( 1 )  "*"      "*"     " " "*"    "*"   "*"   " "    " "    "*"      " "   " "    " " 
7  ( 1 )  "*"      "*"     " " "*"    "*"   "*"   " "    "*"    "*"      " "   " "    " " 
8  ( 1 )  "*"      "*"     " " "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " " 
9  ( 1 )  "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      " "   " "    " " 
10  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   " "    " " 
11  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   " "    "*" 
12  ( 1 ) "*"      "*"     "*" "*"    "*"   "*"   "*"    "*"    "*"      "*"   "*"    "*" 
> |
```

In addition to the subset selections, I also displayed the Cp, BIC, RSS and adjusted R squared for both forward and exhaustive.



Forward

Kunal Mehta
50352960
kunalkam



Exhaustive

As is clear from the graphs, the forward and exhaustive selections are quite similar. There is a sharp decline in BIC for both forward and exhaustive subset selections around the 9 variable mark. Based on my results, I would say that the exhaustive subset selection is the best model.

Problem 2

a. The first step was to load the train and test data. Since only 2s and 3s are to be focused on for this problem, I filtered the data to only include them. I also added all the k values in a list.

b. For linear regression, I used the V1 variable as the focus for fitting a line to the data. To check it's prediction accuracy, I ran a loop through the values of the predictions and essentially rounded off the values such that the prediction is binary, either 2 or 3. Then, I simply mathematically calculated the error and stored it in a variable. The error was roughly 4.12%

```
> error
[1] 0.04120879
>
```

Kunal Mehta
50352960
kunalkam

c. For knn classification, as suggested in the pseudo code, I ran a loop through the list of k values, using the knn algorithm function each time and mathematically calculating the error each time and appending it to the store_error list. The testing errors corresponding to ascending order of k values (1, 3, 5, 7, 9, 11, 13, 15):

```
> store_error
[1] 0.02472527 0.03021978 0.03021978 0.03021978 0.03571429 0.03571429
[7] 0.03296703 0.03846154
```

k-value	Error
1	2.47%
3	3.02%
5	3.02%
7	3.02%
9	3.57%
11	3.57%
13	3.29%
15	3.85%

It appears that knn classification is better for all cases than linear regression. The error percentage seems to increase with increase in k-value except for a decline in error between 11 and 13. Beyond 15 as well, the error follows an increasing trend.

```
> store_error2
[1] 0.000000000 0.004319654 0.005759539 0.005759539 0.007919366 0.007919366 0.007919366 0.009359251
```

The training error for each of the k values is as above