STATISTICAL DATA MINING – ASSIGNMENT 1

Name: Pratik Sanghvi

UB#: 50290451

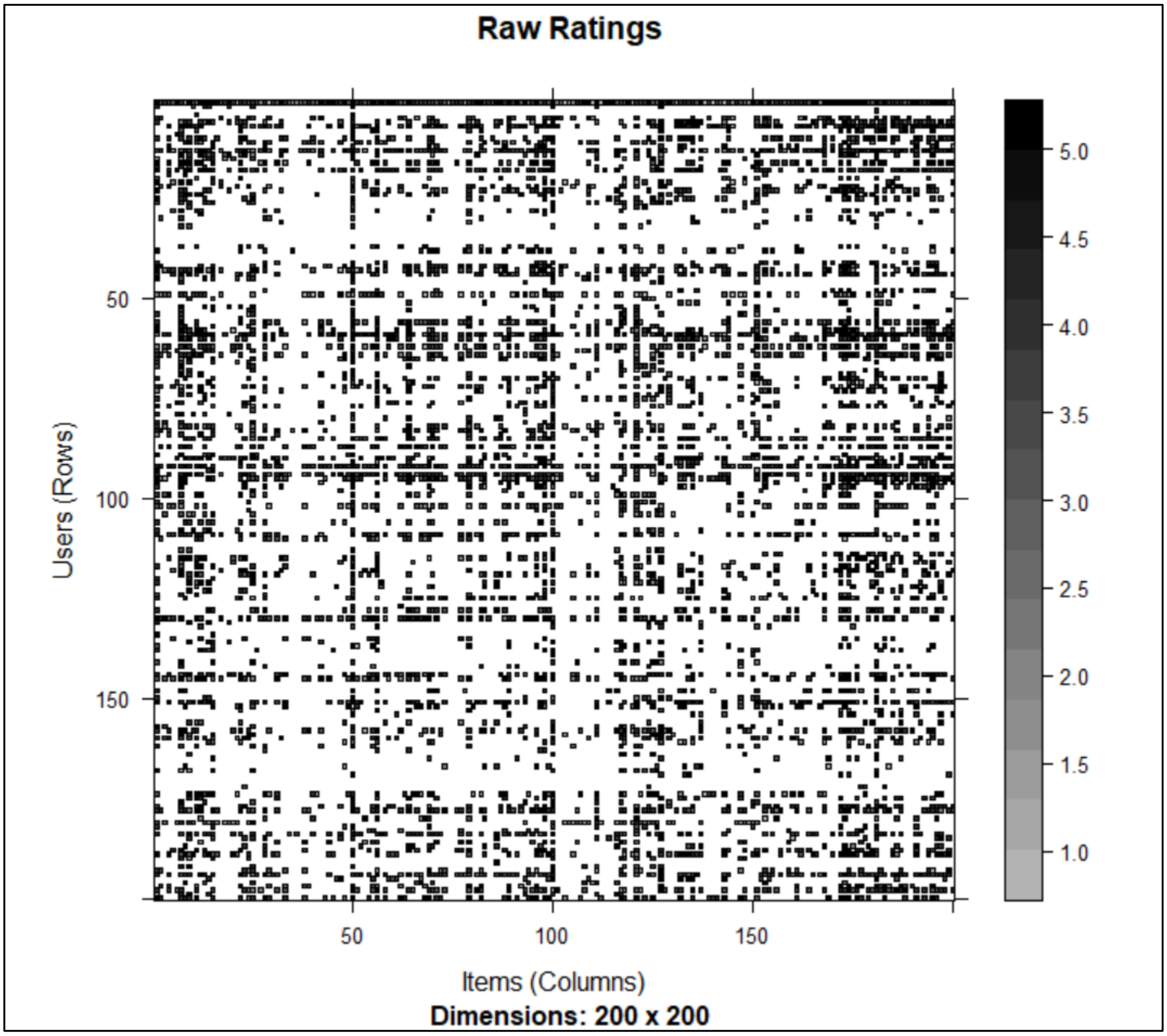
Question 1

In this problem, we design and evaluate our own recommendation system.

We start with by analyzing the data. Checking class, dimensions and data itself using head function.

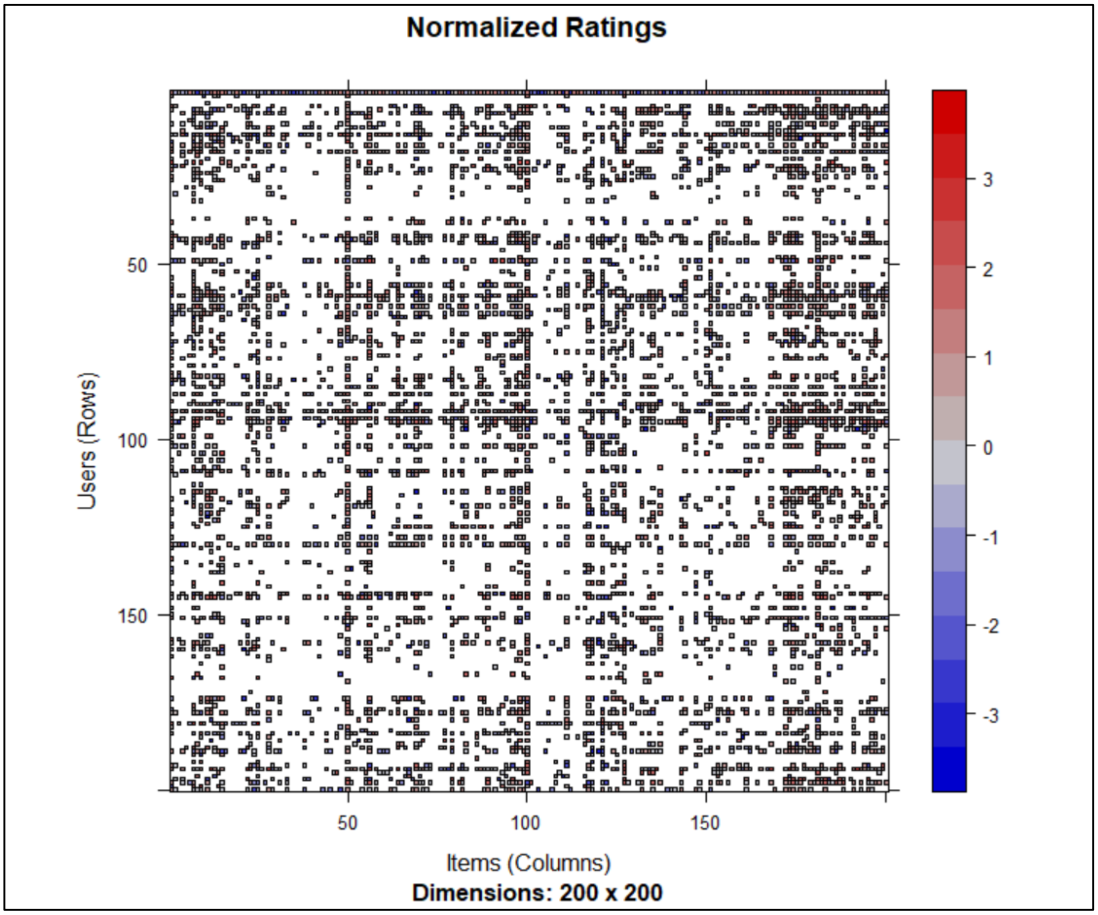
```
> class(movlen)
[1] "realRatingMatrix"
attr(,"package")
[1] "recommenderlab"
>
> head(movlen)
1 x 1664 rating matrix of class ‘realRatingMatrix’ with 271 ratings.
>
> dim(movlen)
[1]  943 1664
>
```

Visualizing the rating Matrix for 200 users and 200 movies.



As expected, we notice that a lot of users have not rated a lot of movies.

We now normalize the ratings and visualize for 200 users and 200 movies to get a better understanding of the ratings.

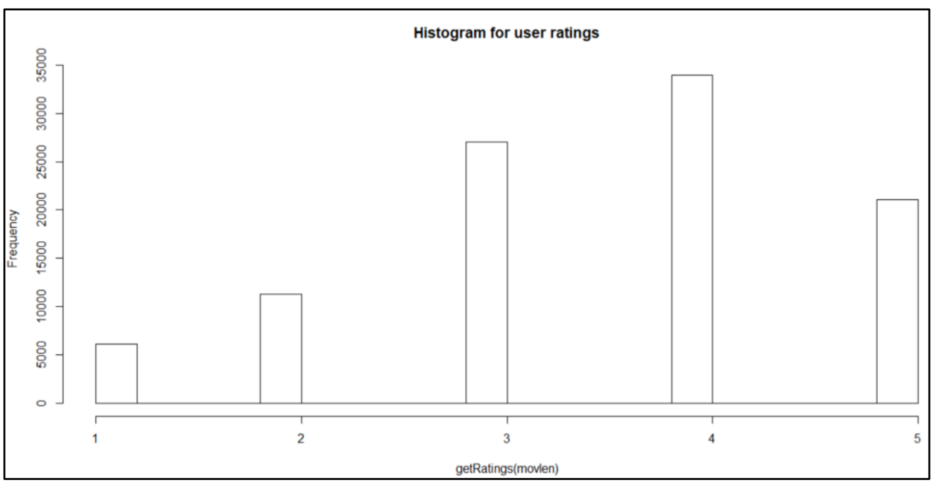


The ratings are evenly spaced, i.e. there are users who have given good ratings visible in red and users who have given lower ratings in blue.

Now, to get another view of the ratings, we use the getRatingMatrix to see the ratings for 30 users and their ratings for 30 movies.

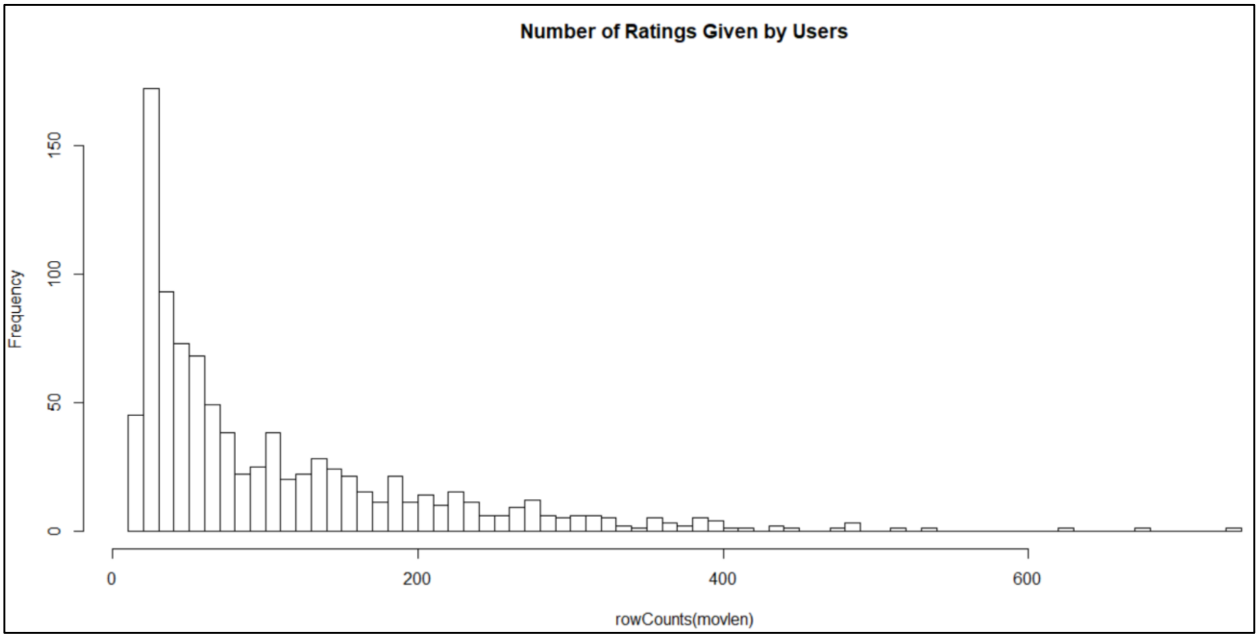
```
> getRatingMatrix(movlen)[1:30,1:30]
30 x 30 sparse Matrix of class "dgCMatrix"
   [[ suppressing 30 column names ‘Toy Story (1995)’, ‘GoldenEye (1
1995)’ ... ]]

1  5 3 4 3 3 5 4 1 5 3 2 5 5 5 5 5 3 4 5 4 1 4 4 3 4 3 2 4 1 3
2  4 . . . . . . . . 2 . . 4 4 . . . . 3 . . . . . 4 . . . . .
3  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
4  . . . . . . . . . . 4 . . . . . . . . . . . . . . . . . . .
5  4 3 . . . . . . . . . . . . . . 4 . . . 3 . . 4 3 . . . 4 .
6  4 . . . . . 2 4 4 . . 4 2 5 3 . . . 4 . 3 3 4 . . . . 2 . .
7  . . . 5 . . 5 5 5 4 3 5 . . . . . . . . . 5 3 . 3 . 4 5 3 .
8  . . . . . . 3 . . . 3 . . . . . . . . . . 5 . . . . . . . .
9  . . . . . 5 4 . . . . . . . . . . . . . . . . . . . . . . .
10 4 . . 4 . . 4 . 4 . 4 5 3 . . 4 . . . . . 5 5 . . . . . . .
11 . . . . . . . 4 5 . 2 2 . . 5 . . . . . . 4 . 3 3 . . 5 3 .
12 . . . 5 . . . . . . . . . . 5 . . . . . . . . . . . . 5 . .
13 3 3 . 5 1 . 2 4 3 . 1 5 5 4 . . 1 . . . 3 4 5 1 1 . 3 5 2 .
14 . . . . . . 5 . 4 . . 5 4 3 4 . . 3 5 . . 3 5 . 2 . . . . .
15 1 . . . . . 1 . 4 . . . 1 4 4 . . 1 . 3 . . . . 3 . . . . .
16 5 . . 5 . . 5 5 5 . 5 5 . . 5 . . . . . . 5 . . . . 2 5 . .
17 4 . . . . . 4 . 3 . . . 3 . . . . . . . . . . . . . . . . .
18 5 . . 3 . 5 . 5 5 . . 5 5 5 4 . . . 3 . . 5 4 . 3 4 . 3 . .
19 . . . 4 . . . 5 . . . . . . . . . . . . . . . . . . . . . .
20 3 . . . . . . . . . 2 . . . 4 . . . . . . 5 . . . . . . . .
21 5 . . . 2 . 5 . 5 . . . . . 4 . 4 . . . . . . . . . . . . .
22 . 2 . 5 . . . . . . . . . . . . 4 . . . 4 . . 5 . . . . 1 .
23 5 . . . . . 4 4 . . . . 4 4 . . . . 4 . . . . . . . . 3 . .
24 . . . . . . 4 5 5 . 5 5 . . . . . . . . . . . . 4 . . . . .
25 5 . . . . . 4 4 . . . . 4 . . . . . . . . . 4 . 5 . . . . .
26 3 . . . . . 3 . 4 . . . 3 3 4 . . . . . . . . 3 3 . . . . .
27 . . . . . . . . 4 . . . . . . . . . . . . . . . . . . . . .
28 . . . . 3 . 5 . . . 4 4 . . . . . . . . . . . . . . . 4 . .
29 . . . . . . . . . . . 5 . . . . . . . . . . . . . . . . . .
30 . 3 . . . . 4 . . . . . . . . . . . . . . . . . . . . 4 3 .
>
```

This gives a better understanding of how each user just rate a handful of movies.



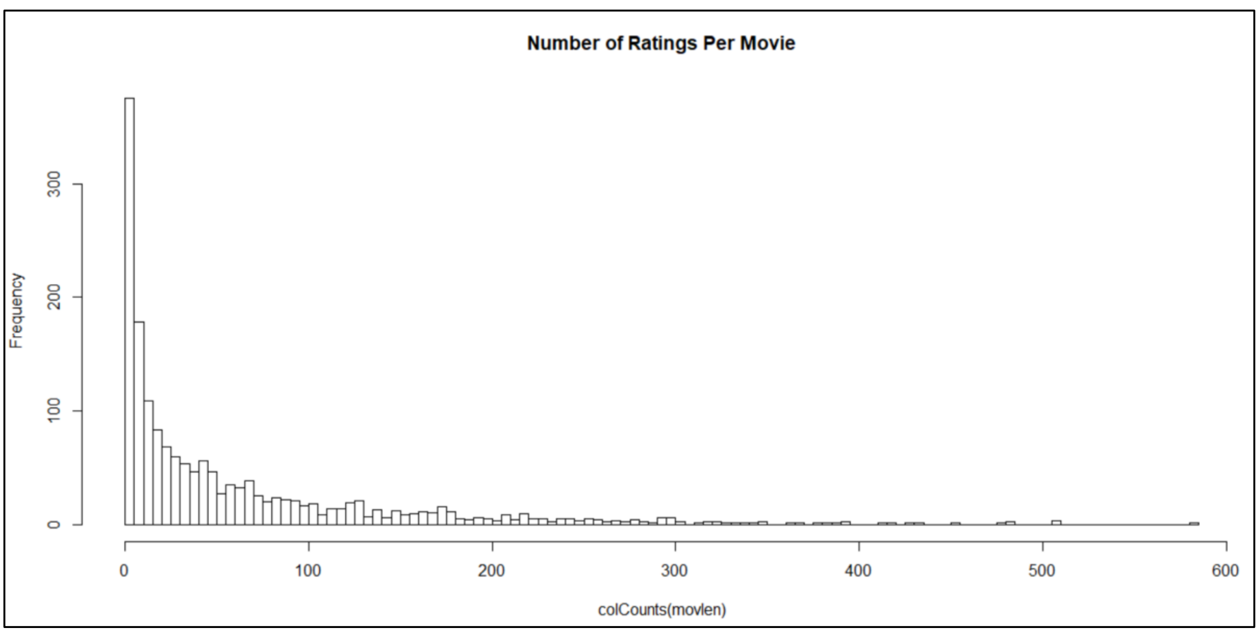
The above histogram shows the frequency of ratings given by the users. Users have generally given higher ratings and ratings of 1 or 2 is fairly less.

Now, visualizing the number of ratings given by users:



As stated earlier, we can see a very high percentage of users rate only a handful of movies, evident by the peaks at the left of the graph.

Similarly, visualizing number of ratings for each movie:



Now, we build our own recommendation system, using the method ‘User Based Collaborative Filtering’.

First, we predict the top 5 recommendations for all users using predict function. Below is the screenshot for the top 5 rated movies for first 5 users. The entire list of top 5 movies for all users is written in a csv file.

```
> top5_movies_rec[1:5]
$`1`
[1] "Titanic (1997)"              "Air Force One (1997)"       "English Patient, The (1996)"
[4] "Game, The (1997)"            "Rainmaker, The (1997)"

$`2`
[1] "Return of the Jedi (1983)" "Graduate, The (1967)"      "Blade Runner (1982)"
[4] "Schindler's List (1993)"   "Casablanca (1942)"

$`3`
[1] "Raiders of the Lost Ark (1981)"
[2] "Blade Runner (1982)"
[3] "Star Wars (1977)"
[4] "Empire Strikes Back, The (1980)"
[5] "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"

$`4`
[1] "Full Monty, The (1997)" "Fargo (1996)"           "Godfather, The (1972)"  "Blade Runner (1982)"
[5] "Pulp Fiction (1994)"

$`5`
[1] "Titanic (1997)"           "Contact (1997)"           "Boogie Nights (1997)"     "Good Will Hunting (1997)"
[5] "L.A. Confidential (1997)"
```

Now, we predict ratings for all the movies that have not been rated by the user, with NA values for movies that have already been rated.

```
> top_movies_with_na[1:5,1:10]
  Toy Story (1995) GoldenEye (1995) Four Rooms (1995) Get Shorty (1995) Copycat (1995)
1               NA               NA                NA                NA             NA
2               NA         3.669613          3.707367          3.704918       3.724054
3         2.777905         2.769734          2.764706          2.631979       2.764706
4         4.373704         4.304348          4.304348          4.357281       4.304348
5               NA               NA          2.874286          2.886138       2.874286
  Shanghai Triad (Yao a yao yao dao waipo qiao) (1995) Twelve Monkeys (1995) Babe (1995) Dead Man Walking (1995)
1                                                   NA                    NA          NA                      NA
2                                             3.704918              3.796620    3.799374                3.704918
3                                             2.764706              2.814385    2.799283                2.906929
4                                             4.304348              4.424833    4.424090                4.363758
5                                             2.874286              2.840346    2.874286                2.949924
  Richard III (1995)
1                 NA
2                 NA
3           2.789459
4           4.402399
5           2.890794
>
```

The entire matrix of predicted ratings for all movies and users is written in a csv file.

To tackle the NA values, we use another method in which type is given as ratingMatrix. The entire matrix of predicted ratings for all movies and users is written in a csv file.

```
> top_movies_without_na[1:5,1:10]
  Toy Story (1995) GoldenEye (1995) Four Rooms (1995) Get Shorty (1995) Copycat (1995)
1         1.394834       -0.6051661         0.3948339        -0.6051661     -0.6051661
2         0.295082        3.6696126         3.7073670         3.7049180      3.7240536
3         2.777905        2.7697345         2.7647059         2.6319791      2.7647059
4         4.373704        4.3043478         4.3043478         4.3572808      4.3043478
5         1.125714        0.1257143         2.8742857         2.8861376      2.8742857
  Shanghai Triad (Yao a yao yao dao waipo qiao) (1995) Twelve Monkeys (1995) Babe (1995) Dead Man Walking (1995)
1                                             1.394834             0.3948339   -2.605166                1.394834
2                                             3.704918             3.7966199    3.799374                3.704918
3                                             2.764706             2.8143849    2.799283                2.906929
4                                             4.304348             4.4248330    4.424090                4.363758
5                                             2.874286             2.8403463    2.874286                2.949924
  Richard III (1995)
1         -0.6051661
2         -1.7049180
3          2.7894592
4          4.4023986
5          2.8907937
>
```

Now, to test the performance of recommendation system, we use evaluationScheme method that creates an evaluation scheme object. This scheme can then be split into k number of cross validation samples. Then use the recommender on training data and check the error on the unknown data.

Following is the performance of the recommender system.

```
> ERROR
         RMSE      MSE       MAE
UBCF 1.058666 1.120774 0.8441363
> ?evaluate
```

Question 2

In this question, we deal with predicting the unspecified ratings for a user using 2 different approaches.

In user based collaborative filtering we use Pearson correlation for similarity while for item based collaborative filtering we use the adjusted cosine similarity.

We consider the following ratings table:

Item-Id ⇒	1	2	3	4	5	6
1	5	6	7	4	3	?
2	4	?	3	?	5	4
3	?	3	4	1	1	?
4	7	4	3	6	?	4
5	1	?	3	2	2	5

We use the recommenderlab package. Initially, start by creating a csv file and input it as a dataframe using read.delim. Then, convert into a matrix and then to a realRatingMatrix.

```
> getRatingMatrix(demog_rrm)
5 x 7 sparse Matrix of class "dgCMatrix"
     User Movie1 Movie2 Movie3 Movie4 Movie5 Movie6
[1,]    1      5      6      7      4      3      .
[2,]    2      4      .      3      .      5      4
[3,]    3      .      3      4      1      1      .
[4,]    4      7      4      3      6      .      4
[5,]    5      1      .      3      2      2      5
>
```

Using the recommender function, we then learn a recommender model on our data. We use the user based collaborative filtering method and apply the Pearson correlation for similarity. Using the predict function we then predict the ratings of user2 for the movies that user hasn't rated.

```
> as(demog_pred_ubcf,"matrix")
     User V1       V2 V3       V4 V5 V6
[1,]   NA NA 3.842212 NA 3.446494 NA NA
>
```

Using user based collaborative filtering, we predict a rating of 3.8 (4 when rounded) for item-2 and 3.44 (3 when rounded) for item-4.

For the second approach, we use the method, item based collaborative filtering and apply the Cosine similarity. Using the predict function we then predict the ratings of user2 for the movies that user hasn't rated.

```
> as(demog_pred_ibcf,"matrix")
     User V1       V2 V3       V4 V5 V6
[1,]   NA NA 3.584778 NA 3.891537 NA NA
>
```

Using item based collaborative filtering, we predict a rating of 3.58 (4 when rounded) for item-2 and 3.89 (4 when rounded) for item-4.

Question 3

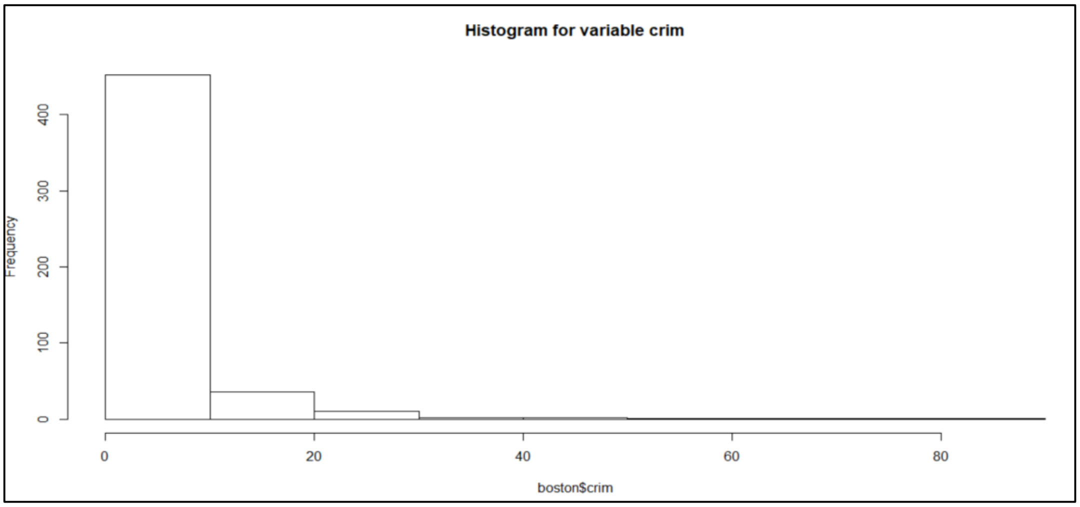
This question works with the renowned Boston dataset, where we work with arules package to apply the apriori algorithm.

Part a

We start by using the correlation matrix, looking at the correlation between different variables. Variable 'dis' doesn't have a strong correlation with variable 'chas'. Hence, we remove the variable 'chas' from our analysis.

Now visualizing the variables using histograms. Looking at the summary statistics and the histograms we categorize the data into 3 or 4 categories. For example, we variable 'crim' following is the summary statistic and histogram plot:

```
> summary(boston$crim)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
 0.00632  0.08204  0.25651  3.61352  3.67708 88.97620 
>
```



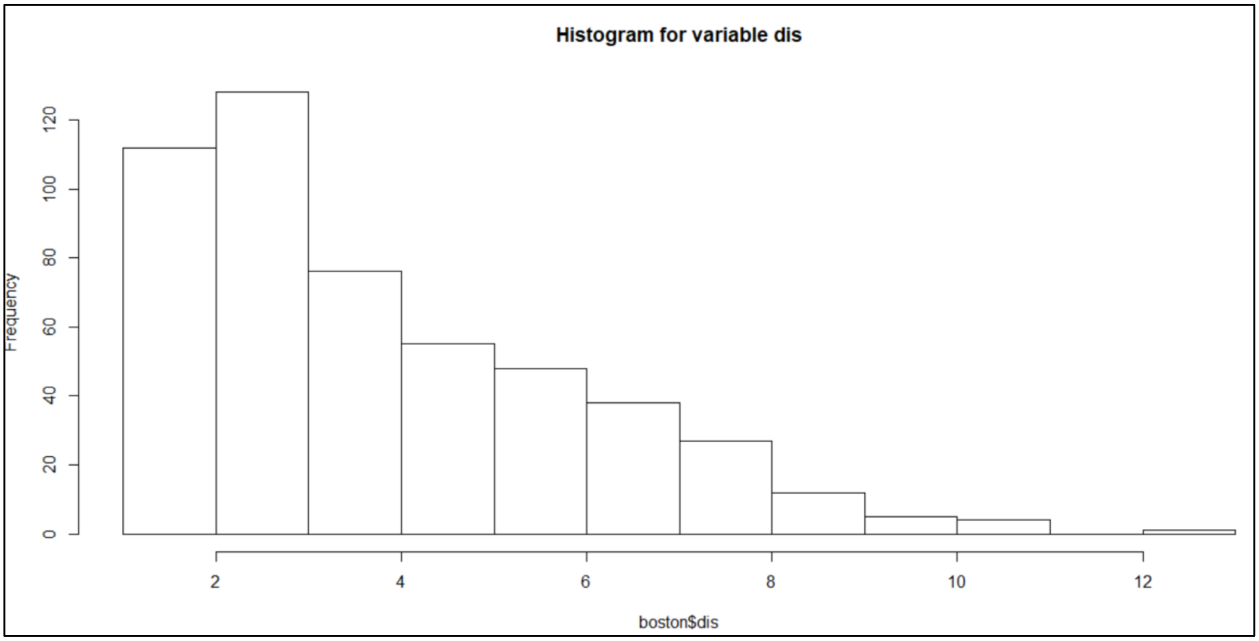
Looking at the above graph and summary, we categorize in 0-0.08 as safe, 0.08-0.25 as moderate, 0.25-3.67 as cautious, 3.67-89 as dangerous.

Similarly, this is done for all the other variables.

Variable **dis** categorized as:

```
Levels: Nearby < Reachable < Far < Very Distant
```

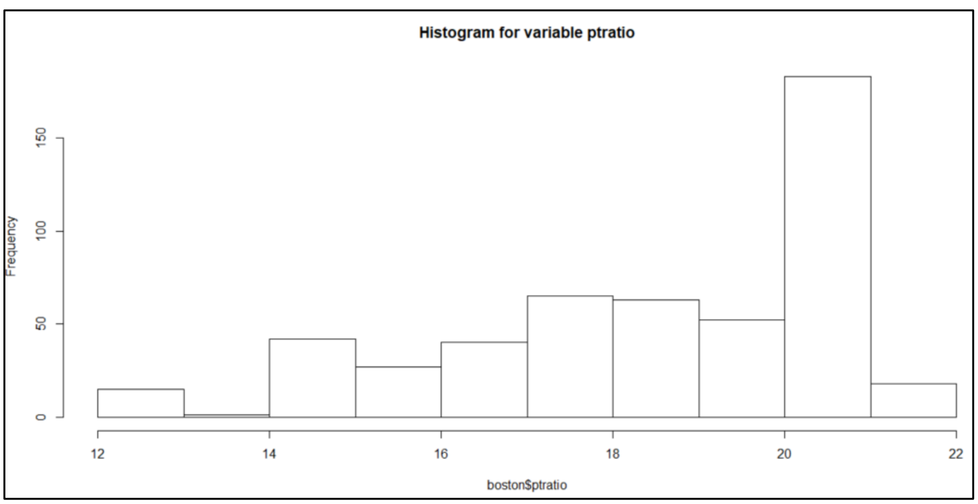
```
> summary(boston$dis)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.130   2.100   3.207   3.795   5.188  12.127 
```



Variable **ptratio** categorized as:

```
Levels: low < Moderate < High < Very High
```

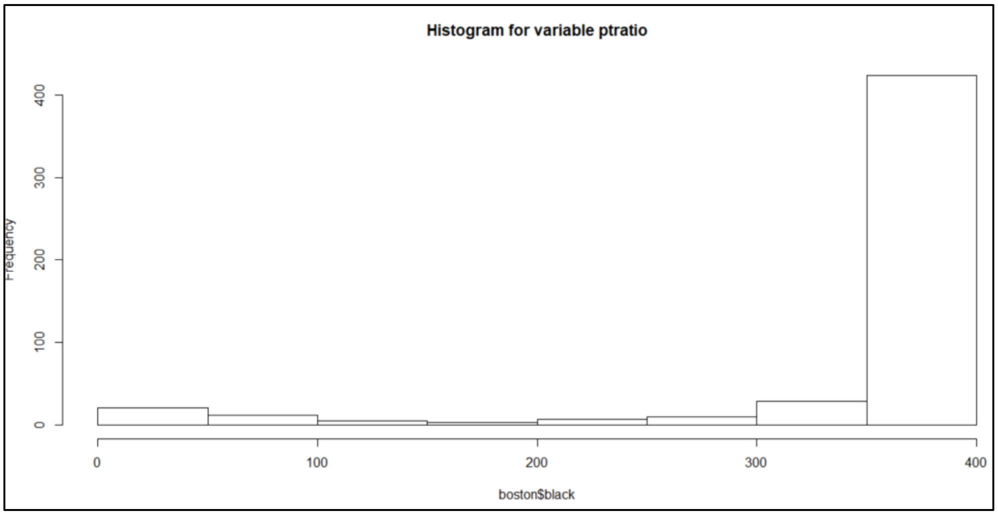
```
> summary(boston$ptratio)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  12.60   17.40   19.05   18.46   20.20   22.00 
```



Variable **black** categorized as:

```
Levels: low < Moderate < High < Very High
```

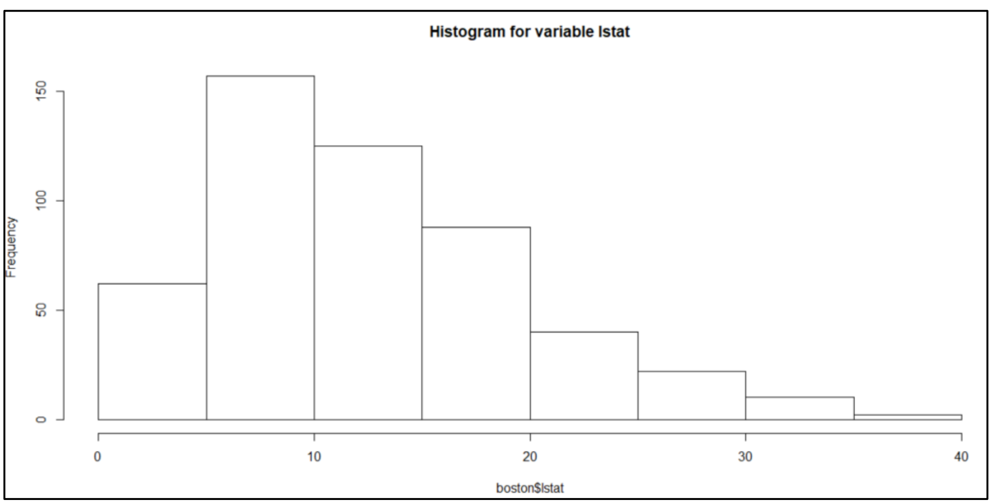
```
> summary(boston$black)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   0.32  375.38  391.44  356.67  396.23  396.90 
>
```



Variable **lstat** categorized as:

```
Levels: low < Moderate < High < Very High
```

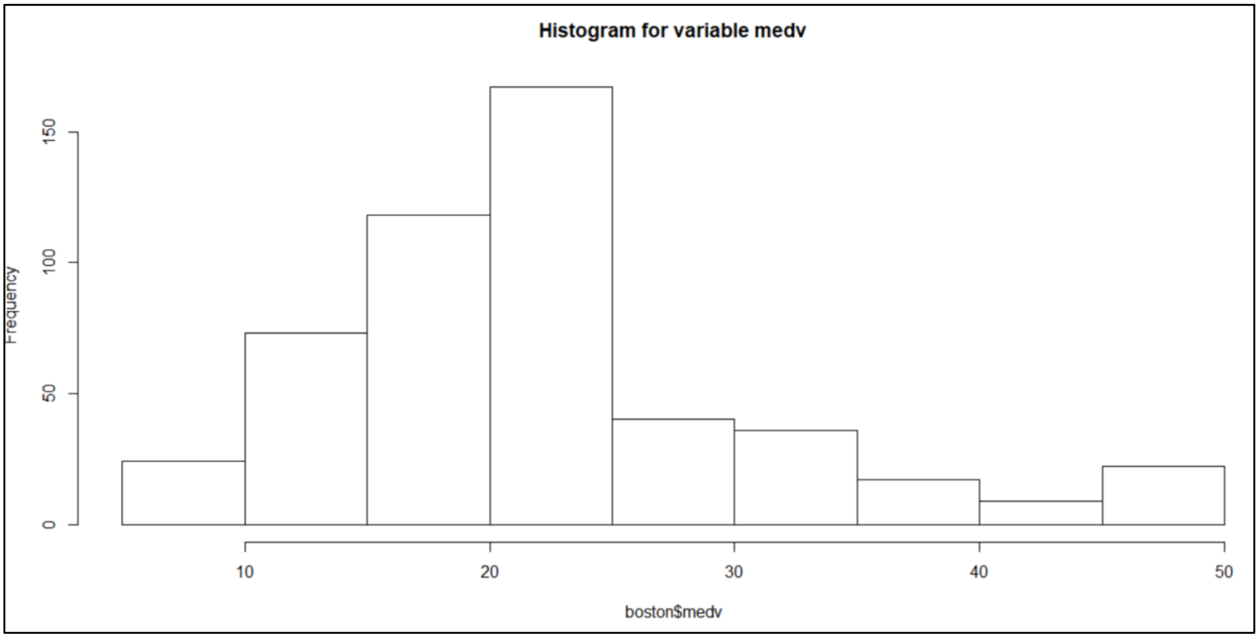
```
> summary(boston$lstat)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   1.73    6.95   11.36   12.65   16.95   37.97 
```



Variable **medv** categorized as:

```
Levels: low < Moderate < High < Very High
```

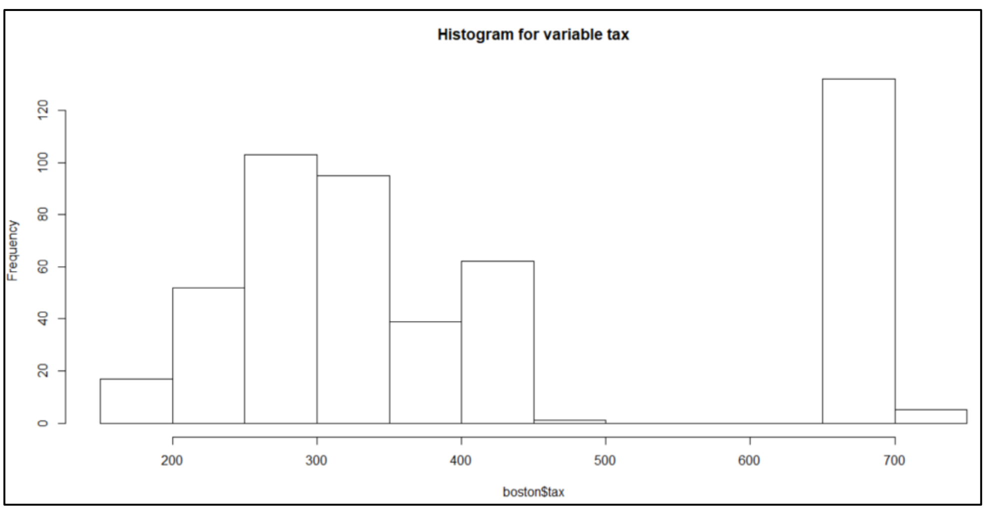
```
> summary(boston$medv)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   5.00   17.02   21.20   22.53   25.00   50.00 
```



Variable **tax** categorized as:

```
4 Levels: low_tax_rate < Moderate_tax_rate < ... < Very_High_tax_rate
>
```

```
> summary(boston$tax)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  187.0   279.0   330.0   408.2   666.0   711.0 
```

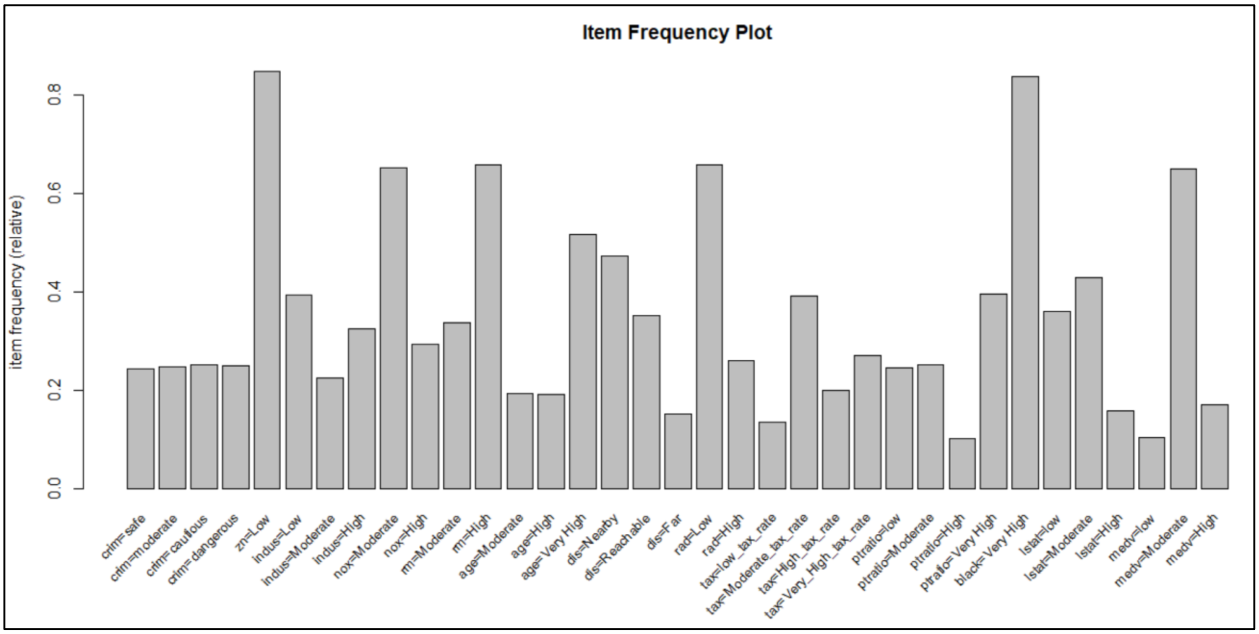


The other variables have been done in a similar way.

Part b

Transforming the data into a binary incidence matrix.

We now apply the apriori algorithm. After a number of iterations, we decide on the support of 0.1 and confidence of 0.6.



Part c

For part c, a student is interested in a low crime area as close to city as possible. We subset the rules where “dis=Reachable” and “crim=safe” and then using inspect look at the rules to draw insights.

```
> inspect(sort(rule_subset, by="confidence"))
    lhs                    rhs                 support confidence     lift count
[1] {crim=safe,
     dis=Reachable}     => {nox=Moderate}    0.1205534  1.0000000 1.533333    61
[2] {crim=safe,
     dis=Reachable}     => {black=Very High} 0.1205534  1.0000000 1.193396    61
[3] {crim=safe,
     indus=Low,
     dis=Reachable}     => {nox=Moderate}    0.1007905  1.0000000 1.533333    51
[4] {crim=safe,
     indus=Low,
     dis=Reachable}     => {black=Very High} 0.1007905  1.0000000 1.193396    51
[5] {crim=safe,
     rm=High,
     dis=Reachable}     => {nox=Moderate}    0.1067194  1.0000000 1.533333    54
[6] {crim=safe,
     dis=Reachable,
     rad=Low}           => {nox=Moderate}    0.1106719  1.0000000 1.533333    56
[7] {crim=safe,
     nox=Moderate,
     dis=Reachable}     => {black=Very High} 0.1205534  1.0000000 1.193396    61
[8] {crim=safe,
     dis=Reachable,
     black=Very High} => {nox=Moderate}      0.1205534  1.0000000 1.533333    61
```

```
      black=Very High} => {nox=Moderate}      0.1106719  1.0000000 1.533333    56
[17] {crim=safe,
      dis=Reachable}   => {rad=Low}           0.1106719  0.9180328 1.394969    56
[18] {crim=safe,
      nox=Moderate,
      dis=Reachable}   => {rad=Low}           0.1106719  0.9180328 1.394969    56
[19] {crim=safe,
      dis=Reachable,
      black=Very High} => {rad=Low}           0.1106719  0.9180328 1.394969    56
[20] {crim=safe,
      nox=Moderate,
      dis=Reachable,
      black=Very High} => {rad=Low}           0.1106719  0.9180328 1.394969    56
[21] {crim=safe,
      dis=Reachable}   => {rm=High}           0.1067194  0.8852459 1.345148    54
[22] {crim=safe,
      nox=Moderate,
      dis=Reachable}   => {rm=High}           0.1067194  0.8852459 1.345148    54
[23] {crim=safe,
```

From the association rules, we can see that areas with low crime rate and as close to city as possible, will have moderate level of NOX concentrations, while it can also mean that the proportion of blacks in that town will be highest. Another advice that can be given is that these towns are then easily accessible to the radial highways.

Part d

For this part, we need to advice a family who needs low pupil to teacher ratio.

```
> inspect(head(sort(rule_lowptratio, by="confidence")))
    lhs                           rhs                support confidence     lift count
[1] {crim=safe,
     ptratio=low}              => {black=Very High} 0.1047431          1 1.193396    53
[2] {tax=Moderate_tax_rate,
     ptratio=low}              => {black=Very High} 0.1324111          1 1.193396    67
[3] {indus=Low,
     ptratio=low}              => {black=Very High} 0.1561265          1 1.193396    79
[4] {nox=Moderate,
     ptratio=low}              => {black=Very High} 0.1482213          1 1.193396    75
[5] {zn=Low,
     ptratio=low}              => {rad=Low}         0.1541502          1 1.519520    78
[6] {crim=safe,
     rad=Low,
     ptratio=low}              => {black=Very High} 0.1027668          1 1.193396    52
>
```

By mining the association rules, we see that where there are ptratio is low, proportion of blacks is very high. Also, these areas are very accessible to the radial highways. Areas where the ptratio is low, generally is safe and has low proportion of residential land.

Part e

For this part, we have to apply linear regression to the data to compare our results from part d, to get inferences for the family.

```
> summary(bos_lm)

Call:
lm(formula = boston_e$ptratio ~ ., data = boston_e)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.1190 -1.0126 -0.0060  0.8961  4.8945 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  2.484e+01  1.352e+00  18.379  < 2e-16 ***
crim        -1.578e-02  1.085e-02  -1.454  0.14661    
zn          -2.473e-02  4.408e-03  -5.611 3.35e-08 ***
indus        5.722e-02  1.997e-02   2.865  0.00434 ** 
chas        -2.824e-01  2.846e-01  -0.992  0.32152    
nox         -1.050e+01  1.187e+00  -8.848  < 2e-16 ***
rm          -7.076e-02  1.479e-01  -0.478  0.63255    
age          7.198e-03  4.313e-03   1.669  0.09577 .  
dis         -2.187e-02  6.883e-02  -0.318  0.75084    
rad          1.177e-01  2.154e-02   5.465 7.35e-08 ***
tax          6.983e-04  1.244e-03   0.561  0.57491    
black        1.573e-03  8.873e-04   1.773  0.07692 .  
lstat       -3.770e-02  1.824e-02  -2.067  0.03929 *  
medv        -1.021e-01  1.402e-02  -7.283 1.31e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.554 on 492 degrees of freedom
Multiple R-squared:  0.4982,	Adjusted R-squared:  0.485
```

From using the lm function on data, we see that variables, zn, nox, rad and medv are the important ones. However, the results are not comparable, as linear regression would give us a prediction of response variable whereas association rules would give us inferences/ suggestions for how would other variables behave. Hence for this case, an association rule would be preferred over linear regression.

Question 4

This problem works with marketing data from the package ElemStatLearn. We have to build a classification tree on training as well as reference sample.

```
> dim(demo_data)
[1] 8993   14
```

We create a vector of length 8993 called class and assign a value of 1 in every row. Next step is to merge it to training data. Then, we check for missing values.

```
> sum(is.na(demo_data))
[1] 2694
>
```

We replace the missing values with the median value.

```
> sum(is.na(demo_data))
[1] 0
```

Now, we sample data into a reference_data data frame from original demographic data. Using rbind we combine the data and then build a classification tree on it.

```
> class_tree$cptable
  CP nsplit rel error xerror xstd
1  0      0         1      0    0
>
```

We observe that we don’t get any splits for the data.

```
> class_pred <- predict(class_tree, data[,-c(15)])
> class_pred
            0   1
    [1,] 0.5 0.5
    [2,] 0.5 0.5
    [3,] 0.5 0.5
    [4,] 0.5 0.5
    [5,] 0.5 0.5
    [6,] 0.5 0.5
    [7,] 0.5 0.5
    [8,] 0.5 0.5
    [9,] 0.5 0.5
   [10,] 0.5 0.5
   [11,] 0.5 0.5
```

Now when we predict the class on the entire data, we observe that we get a 0.5 probability for both classes. Hence the variables cannot help with classification.