Statistical Data Mining
Spring

Assignment -1

Name : Kartik Bapna
UB ID :50291058
Class No 05

The State University of New York at Buffalo

Engineering Sciences - Data Science

Question 1) Consider the MovieLense data in the "recommenderlab" package.
Design and evaluate your own recommendation system based

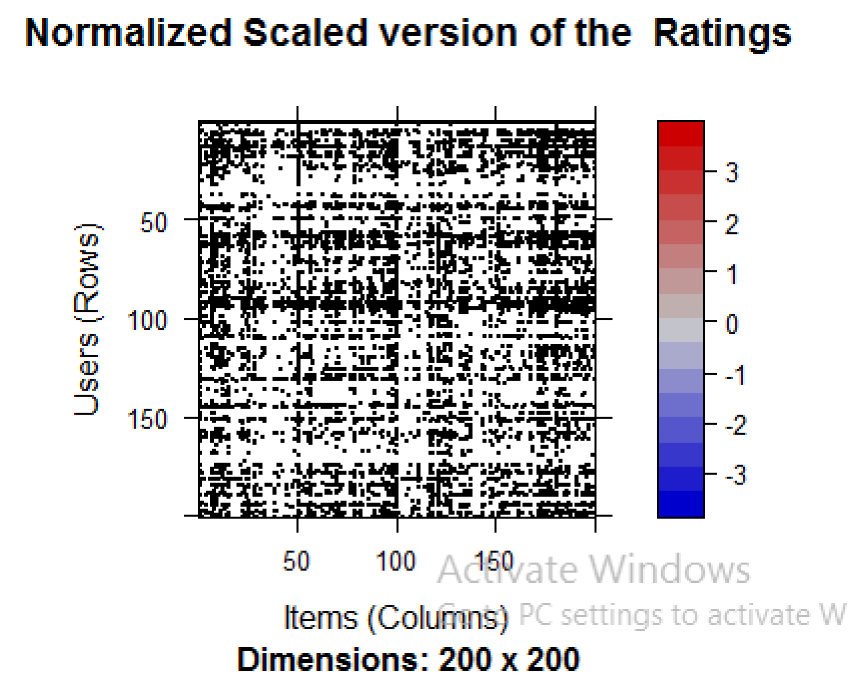
For each user "i" and each movie "j" they did not see, find top "k" most similar users to "i" who have seen "j" and then use them to infer the user "i" 's rating on movie. Handle all exceptions in a reasonable way and report your strategy if you did so; e.g., if you cannot find "k" users for some movie "j", then take all users who have seen it.

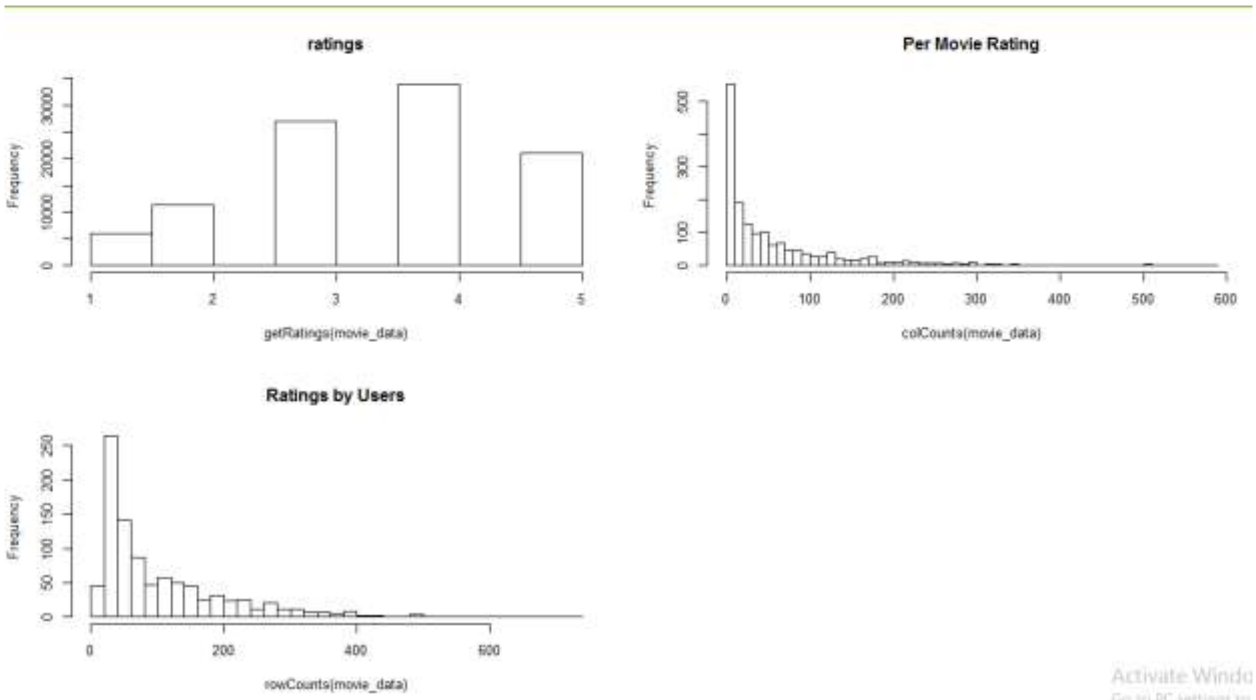
Solution 1

Step1 : Uploading and Analyzing data

```
movie_data <- MovieLense

## analyzing data
head(movie_data)

dim(movie_data) ## 943* 1664

image(movie_data[1:200,1:200], main= "Intial Ratings")
```

Step 2 : Scaling the data and Making Plots and visualization data





Step 3 Making Recommender using UBCF data

```
recomm_movie <- Recommender(movie_data, method= "UBCF")
summary(recomm_movie)
```

```
> summary(recomm_movie)
     Length       Class        Mode
          1 Recommender          S4
> 
```

Step 4 Predicting top 7 movies

```
# predicting best 7 values

best_pred <- predict(recomm_movie, movie_data, n = 7)

best_movie <- bestN(best_pred, n=7)

best_movies_pred <- as(best_movie, "list")

best_movies_pred[1:7]
```

```
$`1`
[1] "Titanic (1997)"             "Air Force One (1997)"       "English Patient, The (1996)"
[4] "Game, The (1997)"           "Rainmaker, The (1997)"      "Wedding Singer, The (1998)"
[7] "Anna Karenina (1997)"

$`2`
[1] "Return of the Jedi (1983)"
[2] "Graduate, The (1967)"
[3] "Blade Runner (1982)"
[4] "Schindler's List (1993)"
[5] "Casablanca (1942)"
[6] "Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)"
[7] "Silence of the Lambs, The (1991)"

$`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)"
[6] "Wrong Trousers, The (1993)"
[7] "Close Shave, A (1995)"
```

Part B

Test the performance of your system using cross-validation. For each data set, the MovieLens database already provides a split of the initial data set into N = 5 folds. This means you will run your algorithm N times; in each step, use the training partition to make predictions for each user on all terms rated in the test partition (by that user). When you complete all N iterations, you will have a large number of user-movie pairs from the 5 test partitions on which you can evaluate the performance of your system. Measure the performance of your recommendation system

Step 5 Apply cross validation

```
# cross validation
eval_cal<- evaluationScheme(movie_data,method = "cross-validation",given = 15, train=0.5, goodRating=4, k=5)

# recommender model

model<- Recommender(getData(eval_cal,"train"), "UBCF")
summary(model)
```

Step 6 Making model and see prediction

```
model<- Recommender(getData(eval_cal,"train"), "UBCF")
summary(model)

Prediction<- predict(model, getData(eval_cal, "known"), type="ratings")

# converting in matrix
new_pred <- as(Prediction, "matrix")
```

Step 7 Error Calculation

```
#Top 5
new_pred[1:5,1:5]

error_cal<- rbind(UBCF = calcPredictionAccuracy(Prediction, getData(eval_cal,"unknown")))
error_cal
```

Measuring the performance of RMSE is 1.09

```
> error_cal
         RMSE      MSE      MAE
[1,] 1.090735 1.189702 0.874382
>
```

Question 2) Consider the following ratings table between five users and six items.

Step 1) Make an excel file and read the file in R

```
  User V1 V2 V3 V4 V5 V6
1    1  5  6  7  4  3 NA
2    2  4 NA  3 NA  5  4
3    3 NA  3  4  1  1 NA
4    4  7  4  3  6 NA  4
5    5  1 NA  3  2  2  5
```

Step 2) Convert the file into matrix form

Step3)Convert to rating matrix

```
# convert to rating matrix
rrm_matrix <- as(matrix,"realRatingMatrix")
```

Step 4)Predict the value using pearson method

```
pearson_recc <- Recommender(rrm_matrix, method = "UBCF",param= list(method= "Pearson"))

pearson_predict <- predict(pearson_recc, rrm_matrix, type="ratings")
```

step 5) Check the matrix with predicted value for user 2 and others

as we can see all the missing value are getting predicted

```
     User       V1       V2 V3       V4       V5       V6
[1,]   NA       NA       NA NA       NA       NA 4.611513
[2,]   NA       NA 3.842212 NA 3.446494       NA       NA
[3,]   NA 2.499412       NA NA       NA       NA 2.793285
[4,]   NA       NA       NA NA       NA 4.37253        NA
[5,]   NA       NA 3.290321 NA       NA       NA       NA
```

Step 6) Repeat the same process for cosine and predict the values for user 2 and others

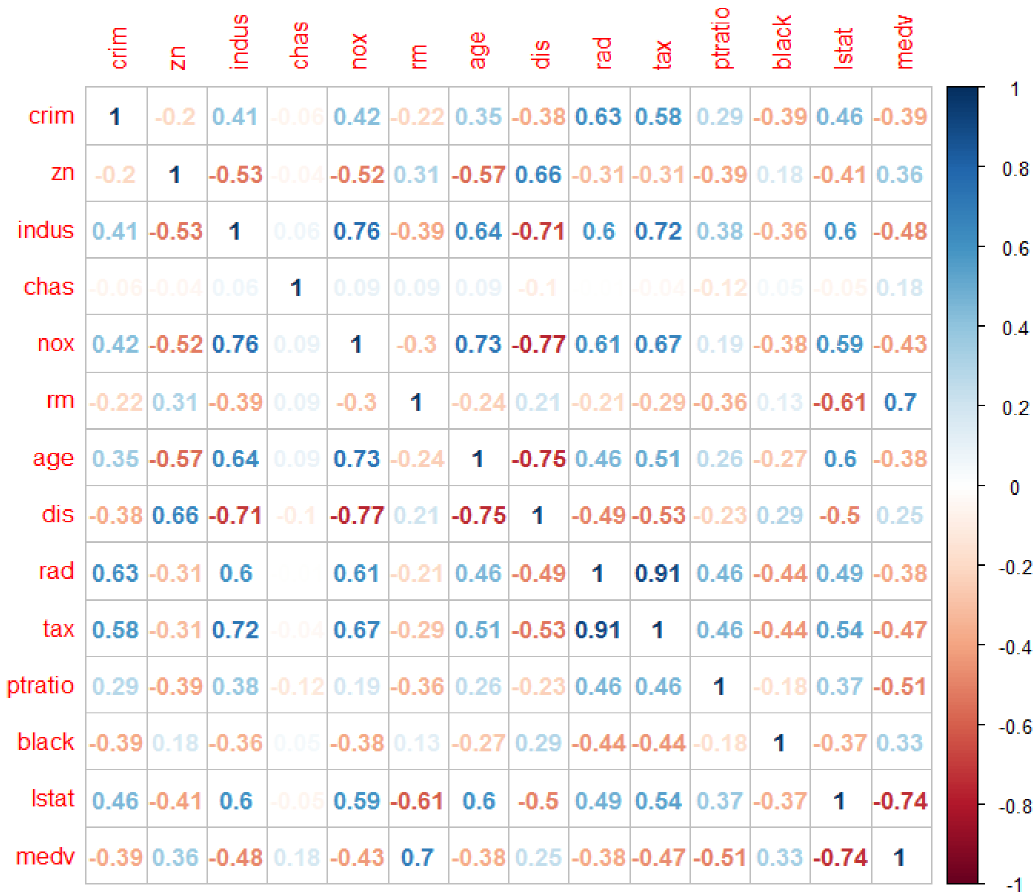
```
### predicting using cosine

cos_recc <- Recommender(rrm_matrix, method = "IBCF",param= list(method= "Cosine"))

cosine_predict <- predict(cos_recc, rrm_matrix, type="ratings")

as(cosine_predict,"matrix")
```

```
as(cosine_predict,"matrix")
     User      V1       V2 V3       V4       V5       V6
[1,]   NA      NA       NA NA       NA       NA 4.273599
[2,]   NA      NA 3.584778 NA 3.891537       NA       NA
[3,]   NA 1.96269       NA NA       NA       NA 2.286019
[4,]   NA      NA       NA NA       NA 4.598693       NA
[5,]   NA      NA 3.521462 NA       NA       NA       NA
```

question (3) (10 points) Consider the Boston Housing Data. This data can be accessed in the ElemStatLearn package (available through CRAN).

Solution 3)

Step1 : Load the Boston data and see the correlation plot

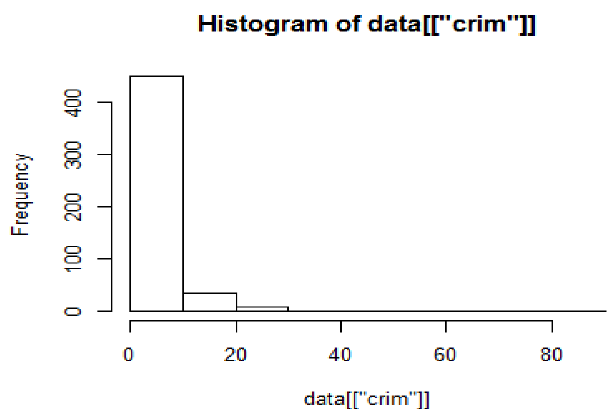


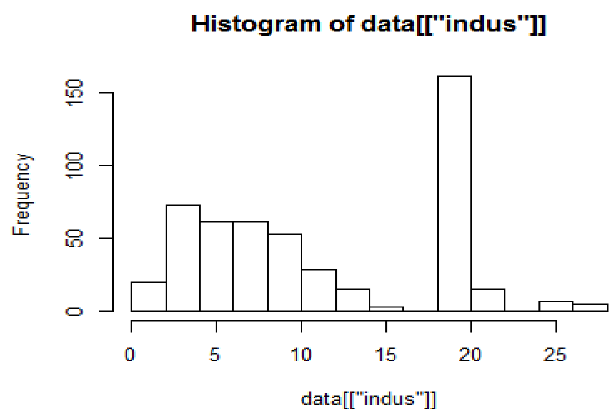
	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
crim	1	-0.2	0.41	-0.06	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39
zn	-0.2	1	-0.53	-0.04	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36
indus	0.41	-0.53	1	0.06	0.76	-0.39	0.64	-0.71	0.6	0.72	0.38	-0.36	0.6	-0.48
chas	-0.06	-0.04	0.06	1	0.09	0.09	0.09	-0.1		-0.04	-0.12	0.05	-0.05	0.18
nox	0.42	-0.52	0.76	0.09	1	-0.3	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43
rm	-0.22	0.31	-0.39	0.09	-0.3	1	-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.7
age	0.35	-0.57	0.64	0.09	0.73	-0.24	1	-0.75	0.46	0.51	0.26	-0.27	0.6	-0.38
dis	-0.38	0.66	-0.71	-0.1	-0.77	0.21	-0.75	1	-0.49	-0.53	-0.23	0.29	-0.5	0.25
rad	0.63	-0.31	0.6		0.61	-0.21	0.46	-0.49	1	0.91	0.46	-0.44	0.49	-0.38
tax	0.58	-0.31	0.72	-0.04	0.67	-0.29	0.51	-0.53	0.91	1	0.46	-0.44	0.54	-0.47
ptratio	0.29	-0.39	0.38	-0.12	0.19	-0.36	0.26	-0.23	0.46	0.46	1	-0.18	0.37	-0.51
black	-0.39	0.18	-0.36	0.05	-0.38	0.13	-0.27	0.29	-0.44	-0.44	-0.18	1	-0.37	0.33
lstat	0.46	-0.41	0.6	-0.05	0.59	-0.61	0.6	-0.5	0.49	0.54	0.37	-0.37	1	-0.74
medv	-0.39	0.36	-0.48	0.18	-0.43	0.7	-0.38	0.25	-0.38	-0.47	-0.51	0.33	-0.74	1

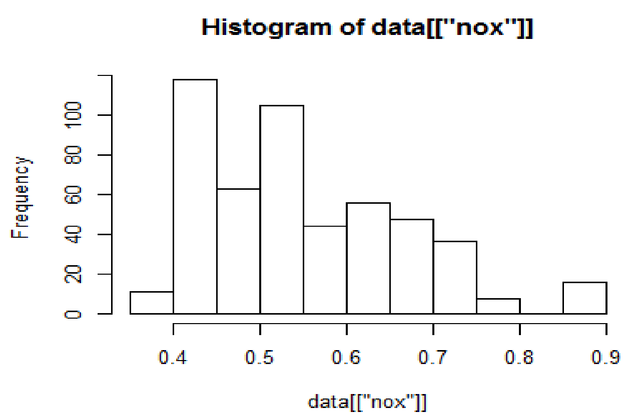
Step 2: Removing the chas as it is not correlated with respect to other variable and plotting histogram for other variables

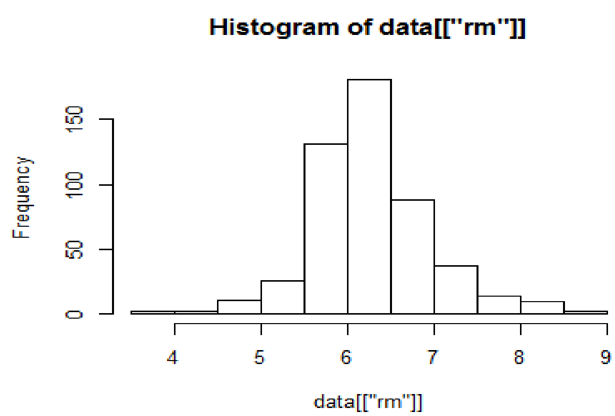
Part a

Visualize the data using histograms of the different variables in the data set. Transform the data into a binary incidence matrix, and justify the choices you make in grouping categories.



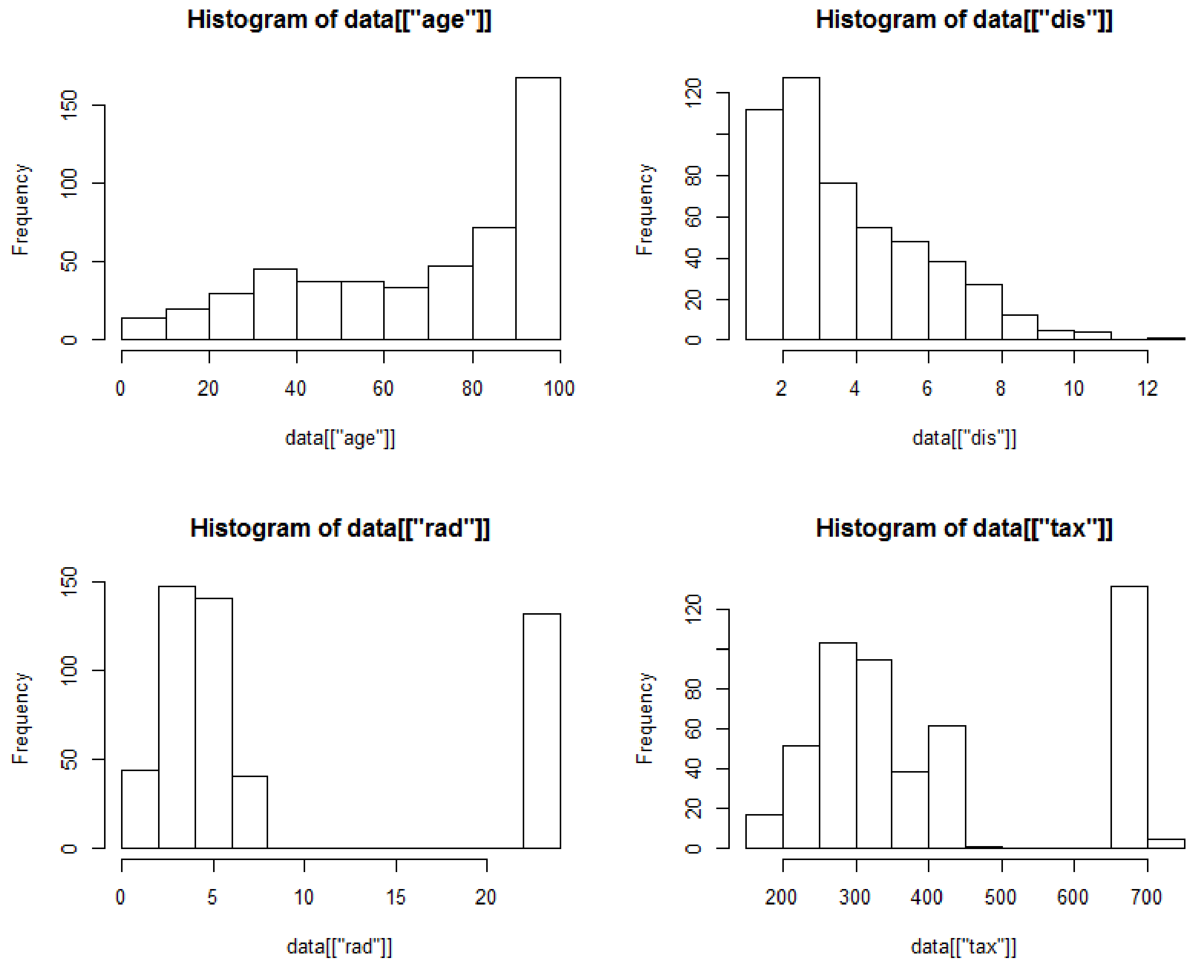






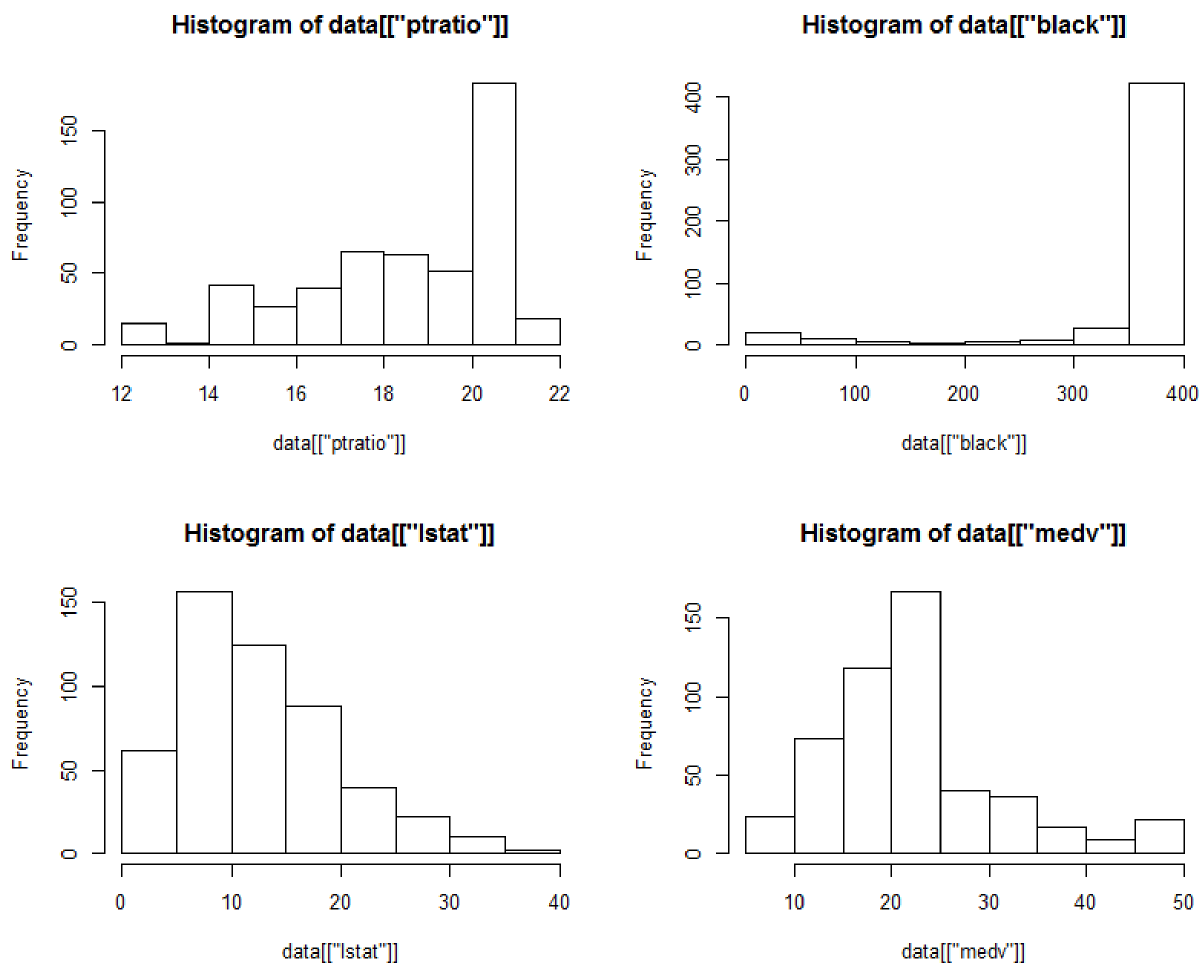
As per the above histogram the categories are classified as Low, Mid and High
as we can see for all the four histogram there are values in low level, mid level and then High Level

Name	Categorised
crim	Low, Mid, High
Indus	Low, Mid, High
Nox	Low, Mid, High
Rm	Low, Mid, High



As per the Histogram this is classified as , However for Dis attribute it is converted into 4 categories as the value are fluctuating within the histogram and for rad it is converted in two level only

Name	Categorised
age	Low, Mid, High
dis	very low, Low, Mid, High
rad	near , extreme
tax	Low, Mid, High



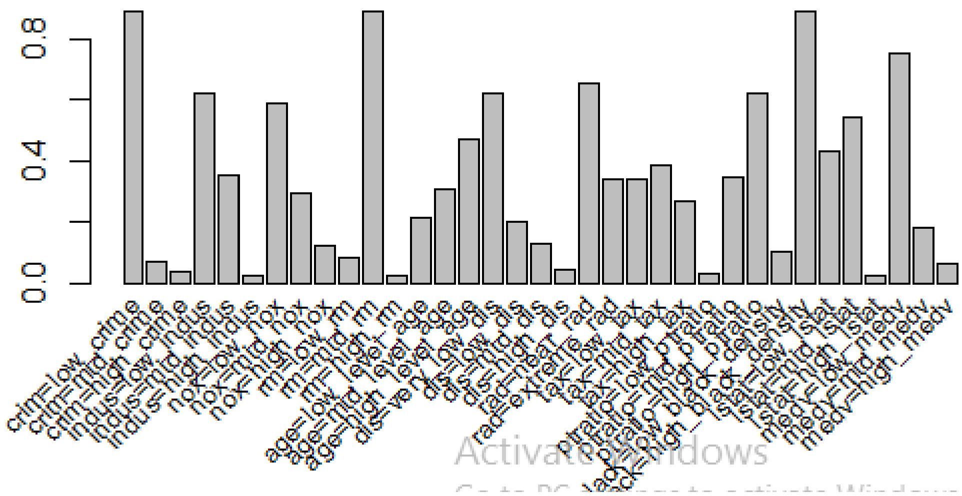
Based on the Histogram the attributes are classified in low, mid and high level however for black adn lstat we can categorize them in two level

Name	Categorised
ptratio	Low, Mid, High
black	Low, High
lstat	near , extreme
Medv	Low, Mid, High

Step 3 : Converting to binary incidence matrix

Part b

Visualize the data using the itemFrequencyPlot in the “arules” package.
Apply the apriori algorithm (Do not forget to specify parameters in your write up).



Step 4 : apply apriori rules

```
> rules<-apriori(boston_mat,parameter = list(support=.005,confidence =.5))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.5    0.1    1 none FALSE            TRUE       5   0.005      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 2

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[35 item(s), 506 transaction(s)] done [0.00s].
sorting and recoding items ... [35 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.08s].
writing ... [693893 rule(s)] done [0.32s].
creating S4 object  ... done [0.50s].
```

Step 5: Checking summary of Rules

```
set of 693893 rules

rule length distribution (lhs + rhs):sizes
     1      2      3      4      5      6      7      8      9     10
    10    344   4004  22070  69325 136906 178584 156666  91653  34331

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    6.00    7.00    7.12    8.00   10.00

summary of quality measures:
    support           confidence          lift             count
 Min.   :0.005929   Min.   :0.5000   Min.   : 0.5597   Min.   :  3.00
 1st Qu.:0.007905   1st Qu.:0.8000   1st Qu.: 1.1195   1st Qu.:  4.00
 Median :0.013834   Median :1.0000   Median : 1.6013   Median :  7.00
 Mean   :0.026168   Mean   :0.8923   Mean   : 2.0616   Mean   : 13.24
 3rd Qu.:0.027668   3rd Qu.:1.0000   3rd Qu.: 2.1013   3rd Qu.: 14.00
 Max.   :0.893281   Max.   :1.0000   Max.   :42.1667   Max.   :452.00

mining info:
       data ntransactions support confidence
 boston_mat           506   0.005        0.5
```

Part c

A student is interested is a low crime area as close to the city as possible (as measured by “dis”). What can you advise on this matter through the mining of association rules?

```
> rules_1 <- subset(rules, subset = rhs %in% "crim=low_crime" & lhs %in% "dis=very_low_dis" & l
ift >.5)
> summary(rules_1)
set of 19090 rules

rule length distribution (lhs + rhs):sizes
   2    3    4    5    6    7    8    9   10
   1   27  258 1210 3139 4940 4964 3227 1324

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   7.000   7.000   7.469   8.000  10.000

summary of quality measures:
    support           confidence          lift            count
 Min.   :0.005929   Min.   :0.5000   Min.   :0.5597   Min.   :  3.00
 1st Qu.:0.007905   1st Qu.:1.0000   1st Qu.:1.1195   1st Qu.:  4.00
 Median :0.013834   Median :1.0000   Median :1.1195   Median :  7.00
 Mean   :0.024446   Mean   :0.9623   Mean   :1.0773   Mean   : 12.37
 3rd Qu.:0.027668   3rd Qu.:1.0000   3rd Qu.:1.1195   3rd Qu.: 14.00
 Max.   :0.517787   Max.   :1.0000   Max.   :1.1195   Max.   :262.00

mining info:
       data ntransactions support confidence
 boston_mat           506   0.005        0.5
```

```
> rules_1 <- subset(rules, subset = rhs %in% "crim=low_crime" & lhs %in% "dis=very_low_dis" & li
t >.5)
> inspect(head(sort(rules_1, by ='lift'),n = 6))
    lhs                                      rhs              support    confidence
[1] {indus=high_indus,dis=very_low_dis}    => {crim=low_crime} 0.02371542 1
[2] {rm=high_rm,dis=very_low_dis}          => {crim=low_crime} 0.02173913 1
[3] {dis=very_low_dis,ptratio=low_ptratio} => {crim=low_crime} 0.02371542 1
[4] {dis=very_low_dis,medv=high_medv}      => {crim=low_crime} 0.04347826 1
[5] {age=low_level_age,dis=very_low_dis}   => {crim=low_crime} 0.02371542 1
[6] {dis=very_low_dis,tax=low_tax}         => {crim=low_crime} 0.14426877 1
    lift     count
[1] 1.119469 12
[2] 1.119469 11
[3] 1.119469 12
[4] 1.119469 22
[5] 1.119469 12
[6] 1.119469 73
```

From the above we can see the for low crime area as close to the city(very_low_dist)
the top rule suggest
as per
rule 1) Indus needs to be high as per that
select area where proportion of non-retail business acres per town as Indus is high

rule2) Ptratio should be low
select area where pupil-teacher ratio by town which is low

rule 3) medv should be high
select an area where Median value of owner-occupied homes in $1000's is high
and same can inferred from other rules

part d

A family is moving to the area, and has made schooling a priority. They want schools with low pupil-teacher ratios. What can you advise on this matter through the mining of association rules ?

```
> #part d
> rulesLowPTRatio <- subset(rules, subset = rhs %in% "ptratio=low_ptratio" & lift >.7)
> inspect(head(sort(rulesLowPTRatio, by ='lift'),n = 6))
    lhs                      rhs                   support confidence     lift count
[1] {nox=mid_nox,
     rm=high_rm,
     tax=low_tax}         => {ptratio=low_ptratio} 0.005928854          1 33.73333     3
[2] {indus=low_indus,
     nox=mid_nox,
     rm=high_rm}          => {ptratio=low_ptratio} 0.005928854          1 33.73333     3
[3] {nox=mid_nox,
     tax=low_tax,
     medv=high_medv}      => {ptratio=low_ptratio} 0.009881423          1 33.73333     5
[4] {indus=low_indus,
     nox=mid_nox,
     medv=high_medv}      => {ptratio=low_ptratio} 0.009881423          1 33.73333     5
[5] {age=high_level_age,
     tax=low_tax,
     medv=high_medv}      => {ptratio=low_ptratio} 0.005928854          1 33.73333     3
```

As per rule 1 we can say the family should select area where nitric oxides concentration is mid level ,average number of dwelling (rm) is high and full-value property-tax rate per $10,000 is low

As per rule 2 we can say where proportion of non-retail business acres per town (indus) is low , nitric oxides concentration is mid level and average number of dwelling (rm) is high

Similary inferences can be seen from other rules

Question e

Use a regression model to solve part d. Are you results comparable? Which provides an easier interpretation? When would regression be preferred, and when would association models be preferred?

After applying regression

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

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 
F-statistic: 37.58 on 13 and 492 DF,  p-value: < 2.2e-16
```

We see there Ptratio has strong relation with Zn, indus, nox, rad and medv

While comparing with linear regression some results are matching and not all also the attributes which are matching are not in details when compared to arules as each attribute is not further classified in low, mid and high , I found association rule to be better while trying to find better results with proper attributes as they provide insights for lift, support and confidence and we can take better decision

question 4)

(10 points) (Modified Exercise 14.4) Cluster the demographic data of Table 14.1 using a classification tree. Specifically, generate a reference sample the same size as the training set. Build a classification tree to the training sample (class 1) and the reference sample (class 0) and describe the terminal nodes having highest estimated class 1 probability.

Solution 1)

Step1 : Loading Marketing Data

```
library(ElemStatLearn)
library(rpart)
data("marketing")

## marketing data
new_data=marketing

str(new_data)

head(new_data)
```

Step 2 : Replacing 2694 NA values with median

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

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

Step 3 : generating random value for all the variable using sample and adding target column with 1

Step 4 : Replicating the data and sampling all the variable and adding target column with value 0

step 5 : Merging the both the data frame using rbind

```
final_data = rbind(new_data_frame, data_frame);
```

step 6 : Converting all the categorical features as factor

Step 7 : Making model using rpart and seeing summary

```
> model = rpart(target~., final_data)
> summary(model)
Call:
rpart(formula = target ~ ., data = final_data)
  n= 17986

    CP nsplit rel error xerror xstd
1 0.01      0         1      0    0

Node number 1: 17986 observations
  predicted class=0  expected loss=0.5  P(node) =1
    class counts:  8993  8993
   probabilities: 0.500 0.500

> 
```

Step 8: Predicting the values

```
> model.predict = predict(model, final_data[,-c(15)])
> model.predict
           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
```

As we can see the node value is 0.5 we cannot reach to any conclusion and using rpart is not advisable for this dataset.