ml-project

suraj gadapa

2022-12-03

```
library(recommenderlab)   #for recommendation
```

```
## Warning: package 'recommenderlab' was built under R version 4.2.2
```

```
## Loading required package: Matrix
```

```
## Loading required package: arules
```

```
## Warning: package 'arules' was built under R version 4.2.2
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
## Loading required package: proxy
```

```
##
## Attaching package: 'proxy'
```

```
## The following object is masked from 'package:Matrix':
##
##     as.matrix
```

```
## The following objects are masked from 'package:stats':
##
##     as.dist, dist
```

```
## The following object is masked from 'package:base':
##
##     as.matrix
```

```
## Loading required package: registry
```

```
## Registered S3 methods overwritten by 'registry':
##   method               from
##   print.registry_field proxy
##   print.registry_entry proxy
```

```
library(reshape2)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
library(ggplot2)
```

#retrieving the data

```
movie_data<-read.csv("C:/Users/suraj/Downloads/archive (3)/imdb_top_1000.csv",stringsAsFactors = FALSE)
str(movie_data)
```

```
## 'data.frame':    1000 obs. of  16 variables:
##  $ Poster_Link  : chr  "https://m.media-amazon.com/images/M/MV5BMDFkYTc0MGEtZmNhMC00ZDIzLWFmNTEtODM1
##  $ Series_Title : chr  "The Shawshank Redemption" "The Godfather" "The Dark Knight" "The Godfather: I
##  $ Released_Year: chr  "1994" "1972" "2008" "1974" ...
##  $ Certificate  : chr  "A" "A" "UA" "A" ...
##  $ Runtime      : chr  "142 min" "175 min" "152 min" "202 min" ...
##  $ Genre        : chr  "Drama" "Crime, Drama" "Action, Crime, Drama" "Crime, Drama" ...
##  $ IMDB_Rating  : num  9.3 9.2 9 9 9 8.9 8.9 8.9 8.8 8.8 ...
##  $ Overview     : chr  "Two imprisoned men bond over a number of years, finding solace and eventual r
##  $ Meta_score   : int  80 100 84 90 96 94 94 94 74 66 ...
##  $ Director     : chr  "Frank Darabont" "Francis Ford Coppola" "Christopher Nolan" "Francis Ford Copp
##  $ Star1        : chr  "Tim Robbins" "Marlon Brando" "Christian Bale" "Al Pacino" ...
##  $ Star2        : chr  "Morgan Freeman" "Al Pacino" "Heath Ledger" "Robert De Niro" ...
##  $ Star3        : chr  "Bob Gunton" "James Caan" "Aaron Eckhart" "Robert Duvall" ...
##  $ Star4        : chr  "William Sadler" "Diane Keaton" "Michael Caine" "Diane Keaton" ...
##  $ No_of_Votes  : int  2343110 1620367 2303232 1129952 689845 1642758 1826188 1213505 2067042 185474
##  $ Gross        : chr  "28,341,469" "134,966,411" "534,858,444" "57,300,000" ...
```

```
md<- na.omit(movie_data) #gives the data after removing the missing values.
summary(md)
```

```
##  Poster_Link        Series_Title       Released_Year      Certificate
##  Length:843         Length:843         Length:843         Length:843
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##    Runtime              Genre             IMDB_Rating      Overview
##  Length:843         Length:843         Min.   :7.600   Length:843
##  Class :character   Class :character   1st Qu.:7.700   Class :character
##  Mode  :character   Mode  :character   Median :7.900   Mode  :character
##                                        Mean   :7.932
##                                        3rd Qu.:8.100
```

```
##                                       Max.   :9.300
##    Meta_score       Director           Star1              Star2
##  Min.   : 28.00   Length:843         Length:843         Length:843
##  1st Qu.: 70.00   Class :character   Class :character   Class :character
##  Median : 79.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 77.97
##  3rd Qu.: 87.00
##  Max.   :100.00
##     Star3              Star4            No_of_Votes         Gross
##  Length:843         Length:843         Min.   :  25198   Length:843
##  Class :character   Class :character   1st Qu.:  71024   Class :character
##  Mode  :character   Mode  :character   Median : 184966   Mode  :character
##                                        Mean   : 313187
##                                        3rd Qu.: 439631
##                                        Max.   :2343110
```

```
sd(md$IMDB_Rating)
```

```
## [1] 0.2837322
```

```
sd(md$Meta_score)
```

```
## [1] 12.3761
```

```
sd(md$No_of_Votes)
```

```
## [1] 341798.8
```

```
IQR(md$IMDB_Rating)
```

```
## [1] 0.4
```

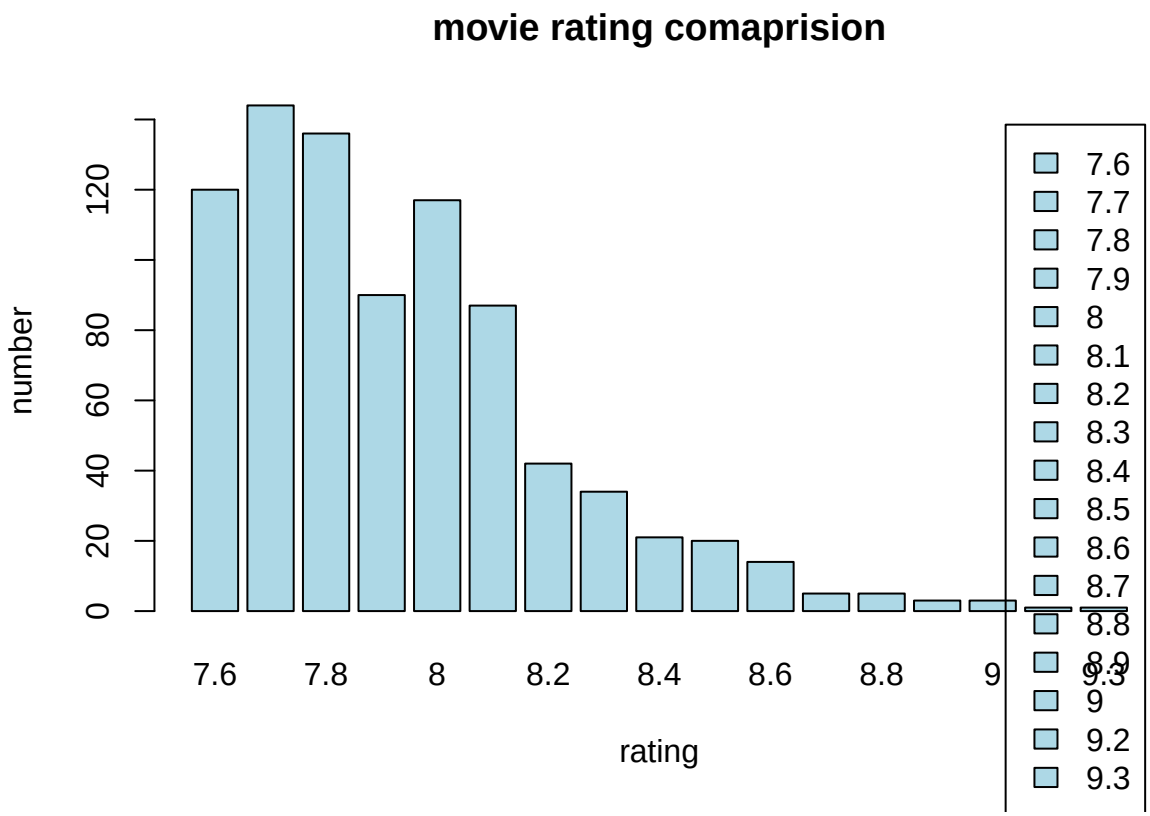
```
IQR(md$Meta_score)
```

```
## [1] 17
```

```
IQR(md$No_of_Votes)
```

```
## [1] 368607.5
```

```
rating<-table(md$IMDB_Rating)
barplot(rating,main="movie rating comaprision",ylab="number",
        xlab="rating",col=("lightblue"),legend=rownames(rating))
```




```
certificate_pie<-table(md$Certificate)
library(lessR)
```

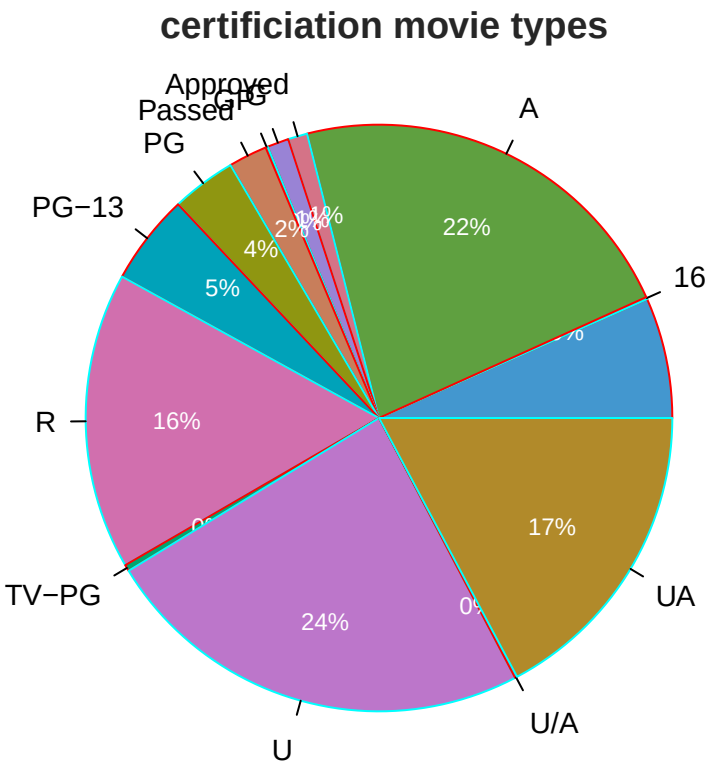
```
## Warning: package 'lessR' was built under R version 4.2.2
```

```
##
## lessR 4.2.4                        feedback: gerbing@pdx.edu
## ------------------------------------------------------------
## > d <- Read("")   Read text, Excel, SPSS, SAS, or R data file
##   d is default data frame, data= in analysis routines optional
##
## Learn about reading, writing, and manipulating data, graphics,
## testing means and proportions, regression, factor analysis,
## customization, and descriptive statistics from pivot tables.
##   Enter:  browseVignettes("lessR")
##
## View changes in this and recent versions of lessR.
##   Enter: news(package="lessR")
##
## **New Feature**: Interactive analysis of your data
##   Enter: interact()

##
## Attaching package: 'lessR'
```

```
## The following object is masked from 'package:data.table':
##
##     set

## The following object is masked from 'package:arules':
##
##     recode
```

```
PieChart(certificate_pie,hole = 0,values="%",data=md$Certificate,color = rainbow(2),main=" certificiatio
```

```
## >>> Note: certificate_pie is not in a data frame (table)
## >>> Note: certificate_pie is not in a data frame (table)
```




```
## >>> suggestions
## piechart(certificate_pie, hole=0)  # traditional pie chart
## piechart(certificate_pie, values="%")  # display %'s on the chart
## piechart(certificate_pie)  # bar chart
## plot(certificate_pie)  # bubble plot
## plot(certificate_pie, values="count")  # lollipop plot
##
## --- certificate_pie ---
##
## certificate_p Count   Prop
## ----------------------
```

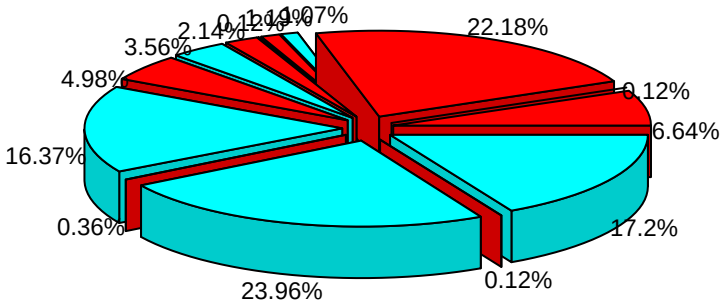
```
##             56   0.066
##       16     1   0.001
##        A   187   0.222
## Approved     9   0.011
##        G    10   0.012
##       GP     1   0.001
##   Passed    18   0.021
##       PG    30   0.036
##    PG-13    42   0.050
##        R   138   0.164
##    TV-PG     3   0.004
##        U   202   0.240
##      U/A     1   0.001
##       UA   145   0.172
## ----------------------
##    Total   843   1.000
##
## Chi-squared test of null hypothesis of equal probabilities
##   Chisq = 1185.738, df = 13, p-value = 0.000
```

```
library(plotrix)
```

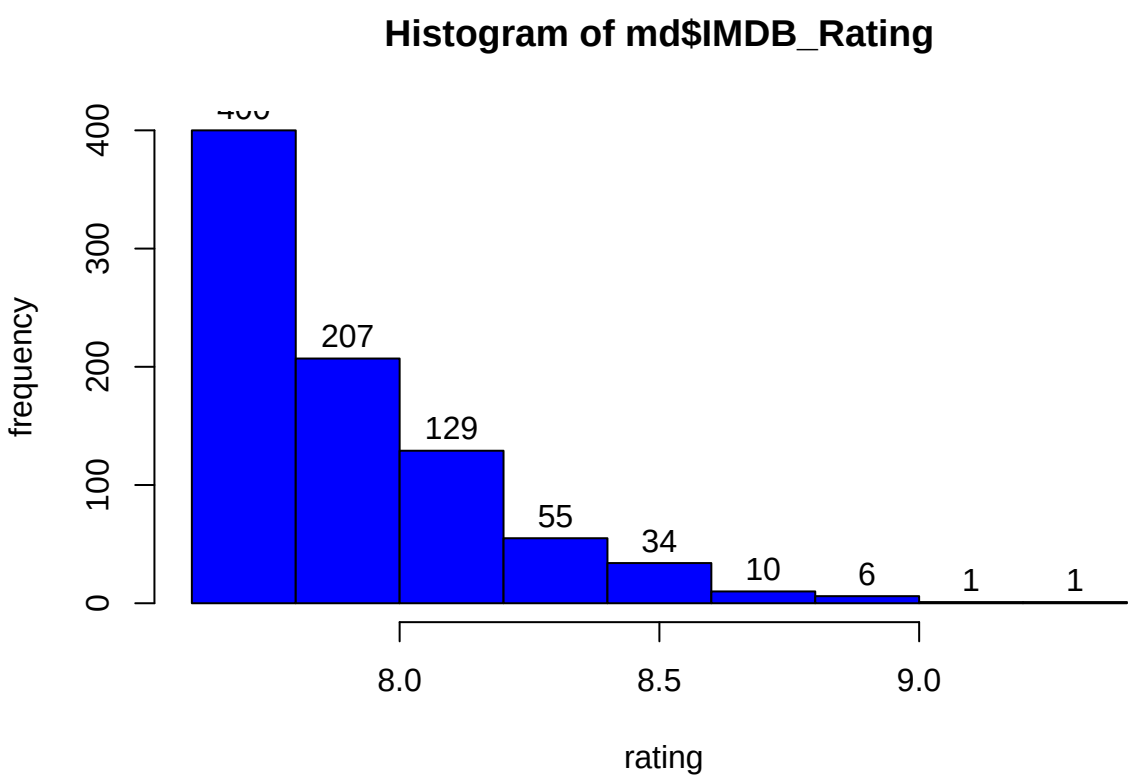
```
##
## Attaching package: 'plotrix'
```

```
## The following object is masked from 'package:lessR':
##
##     rescale
```

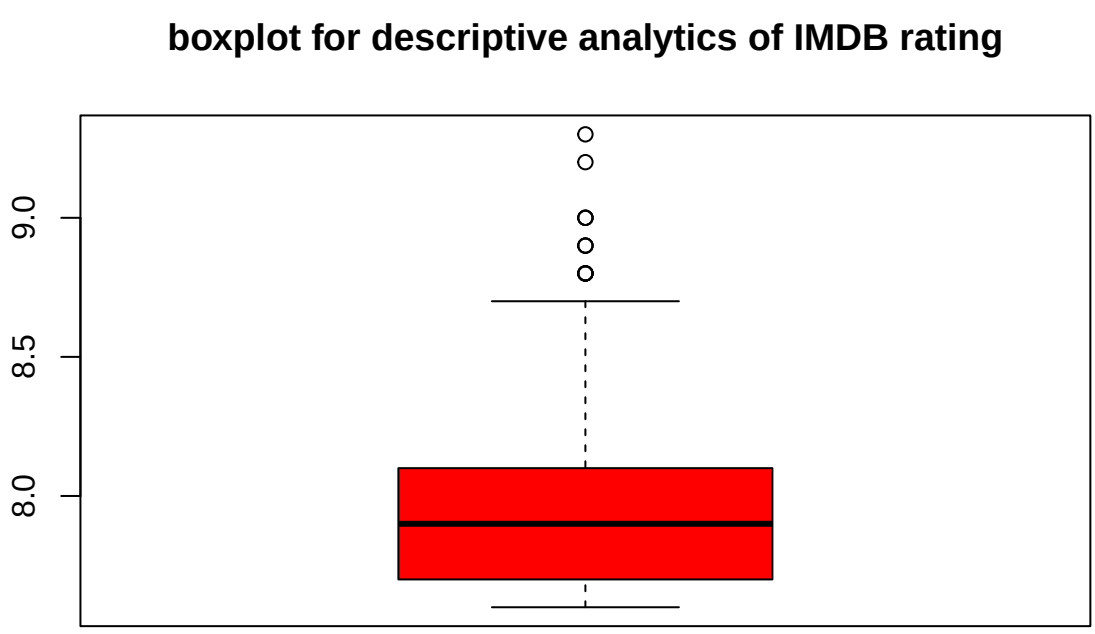
```
lab<-paste0(round(certificate_pie/sum(certificate_pie)*100,2),"%")

pie3D(certificate_pie,col= rainbow(2),labels=lab ,labelcex = 0.75,explode = 0.1)
```



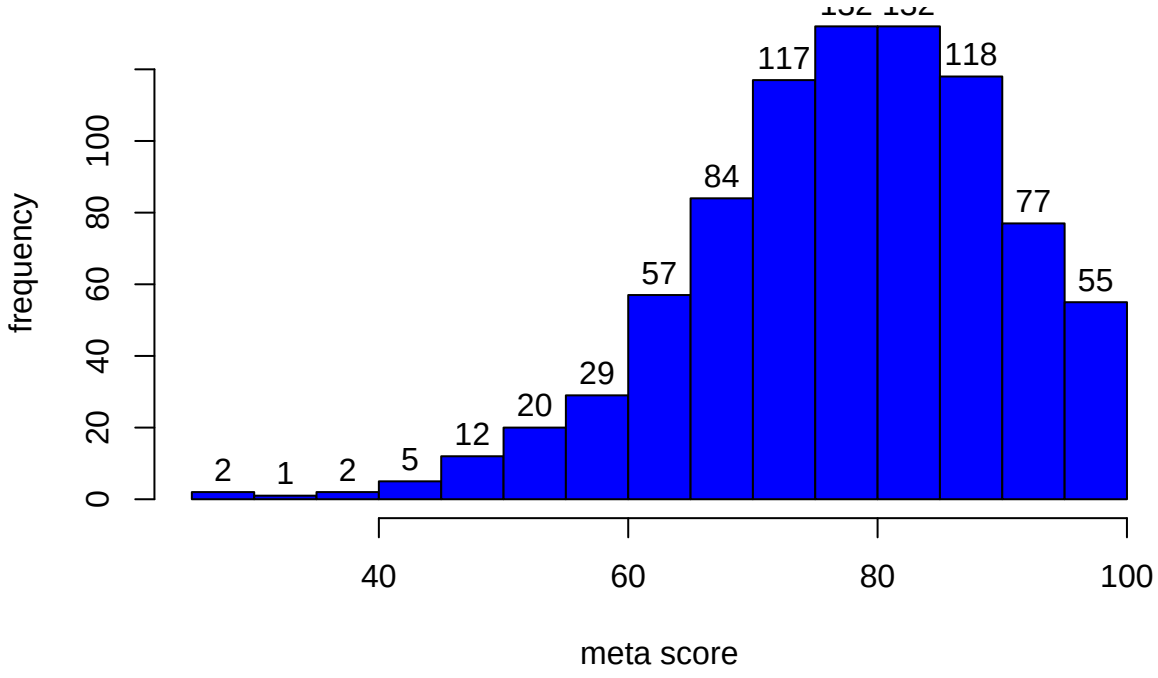

```
hist(md$IMDB_Rating, col = "BLUE", xlab = " rating",ylab = "frequency",labels = TRUE)
```



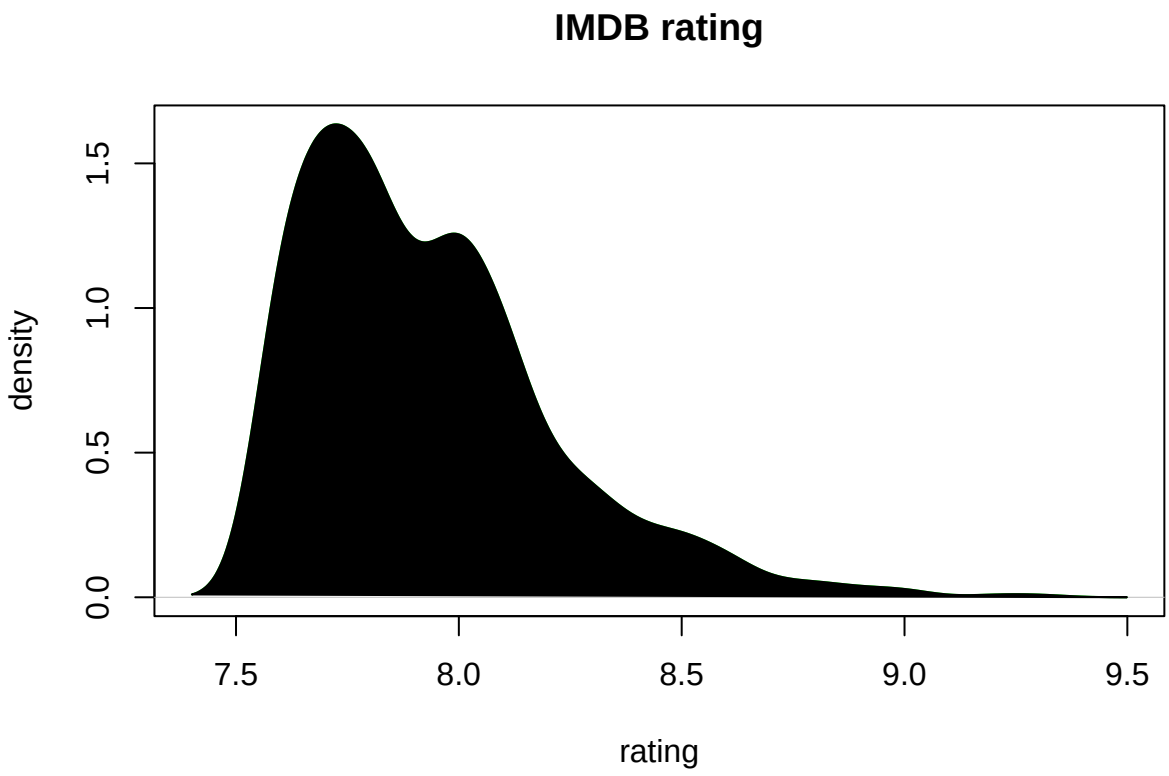

```
boxplot(md$IMDB_Rating, col = "red", border = "black",main="boxplot for descriptive analytics of IMDB ra
```



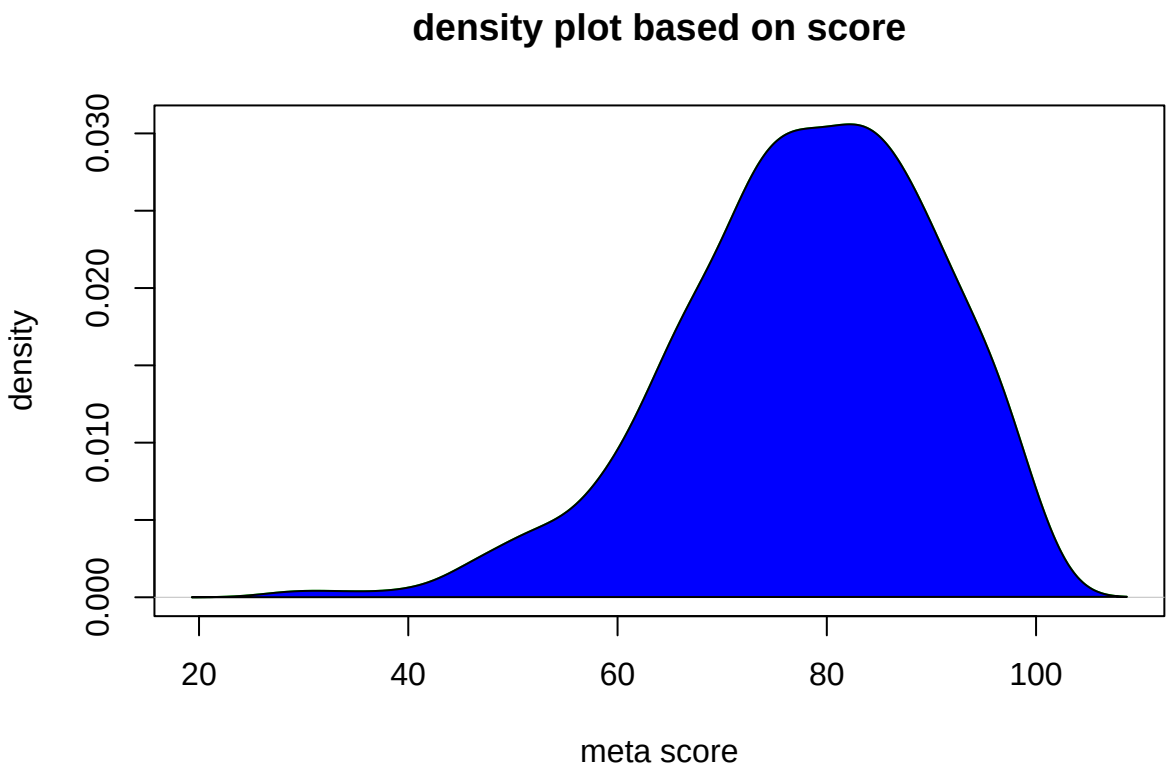

```
hist(md$Meta_score, col = "BLUE", xlab = "meta score",ylab = "frequency",labels = TRUE)
```




```
plot(density(md$IMDB_Rating),
     col="green",
     main="IMDB rating",
     xlab="rating",
     ylab="density")
polygon(density(md$IMDB_Rating),col = "black")
```




```
plot(density(md$Meta_score),
     col="green",
     main="density plot based on score",
     xlab="meta score",
     ylab="density")
polygon(density(md$Meta_score),col = "blue")
```




```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
md1<-na.omit(md)
```

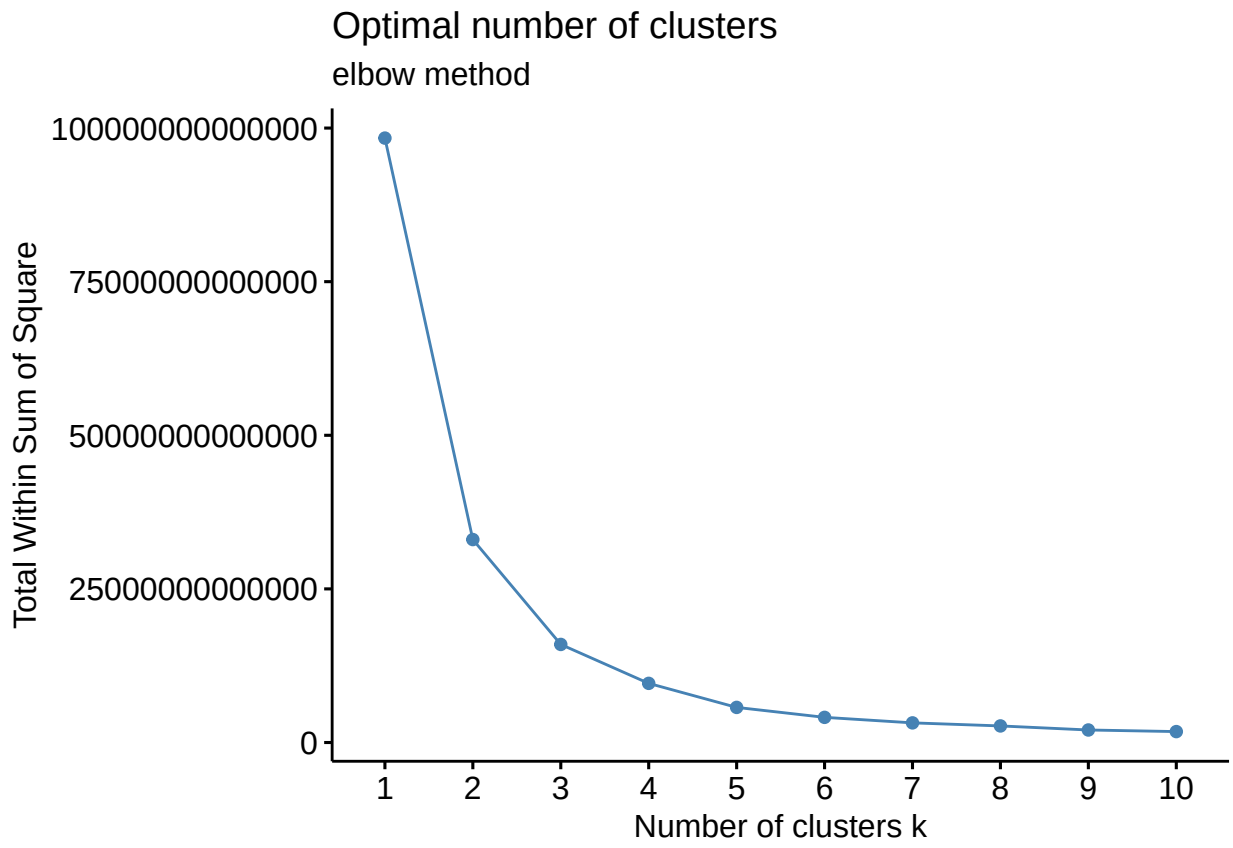
```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:lessR':
##
##     recode, rename

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:arules':
##
##     intersect, recode, setdiff, setequal, union
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
md2<-select_if(md1,is.numeric)
```

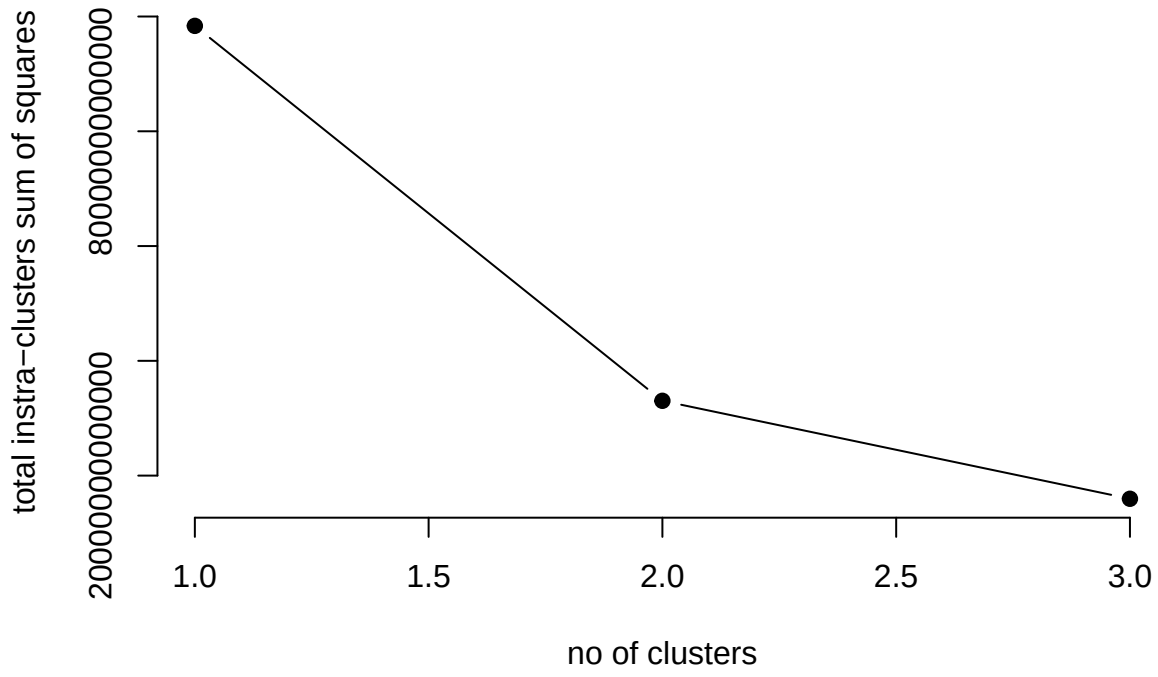
```
library(dplyr)
fviz_nbclust(md2,kmeans,method="wss")+labs(subtitle = "elbow method")
```




```
library(purrr)
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:data.table':
##
##     transpose
```

```r
set.seed(123)
#functionto calculate total intra-cluster sum of squares(euclidean distance)
ics<-function(k){
  kmeans(md2[,1:3],k,iter.max =100,nstart = 100,algorithm = "Lloyd" )$tot.withinss

}
k_values<-1:3
ics_values<-map_dbl(k_values,ics)

plot(k_values,ics_values,
     type="b",pch=19,frame=FALSE,
     xlab="no of clusters",
     ylab="total instra-clusters sum of squares"
       )
```

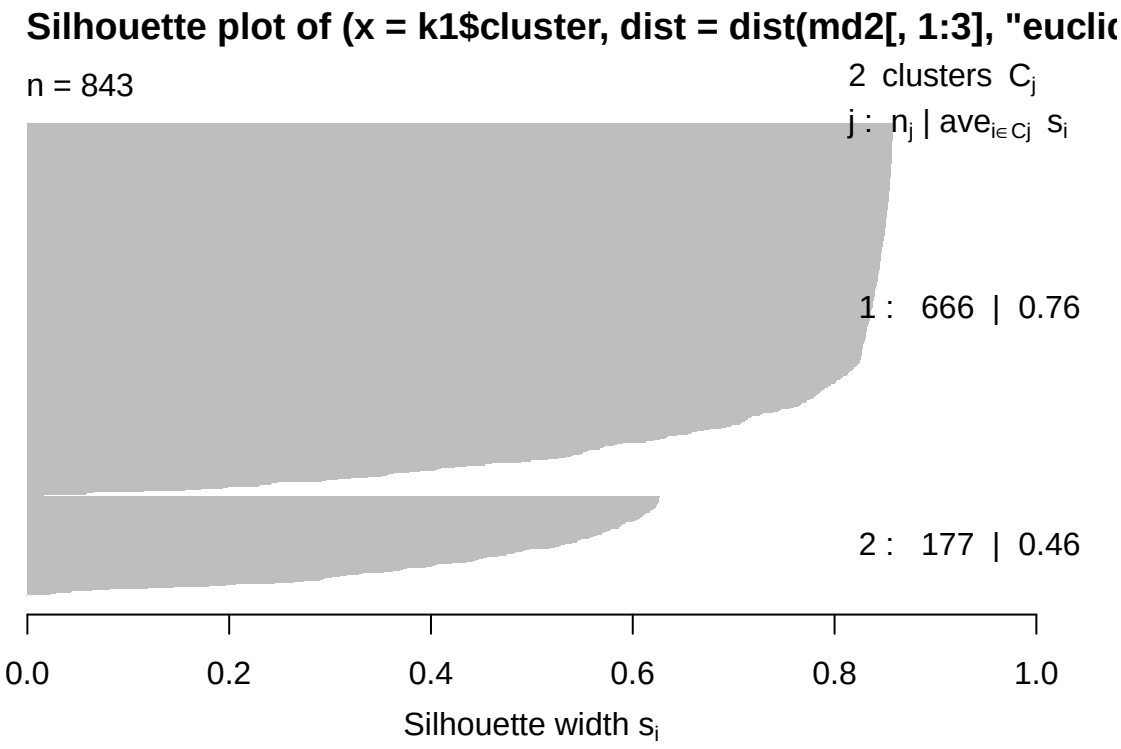



```r
library(cluster)
library(gridExtra)
```

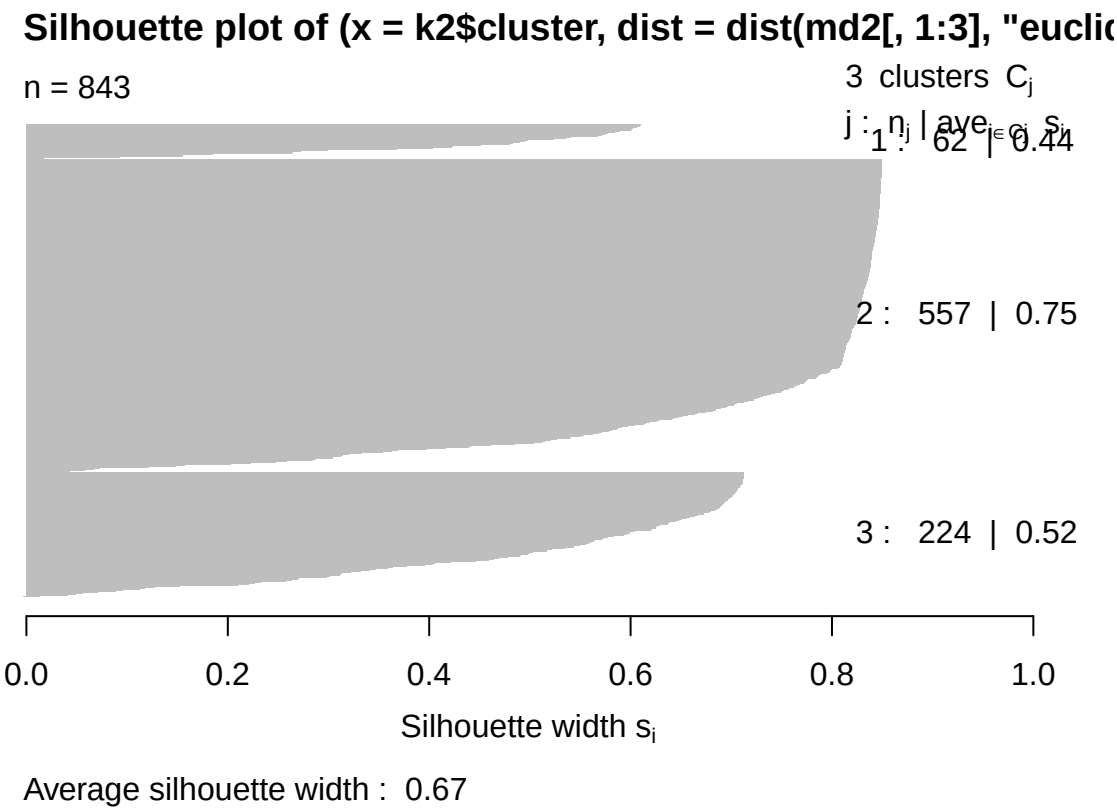
```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(grid)
k1<-kmeans(md2[,1:3],2,iter.max=100,nstart=50,algorithm="Lloyd")

s1<-plot(silhouette(k1$cluster,dist(md2[,1:3],"euclidean")))
```

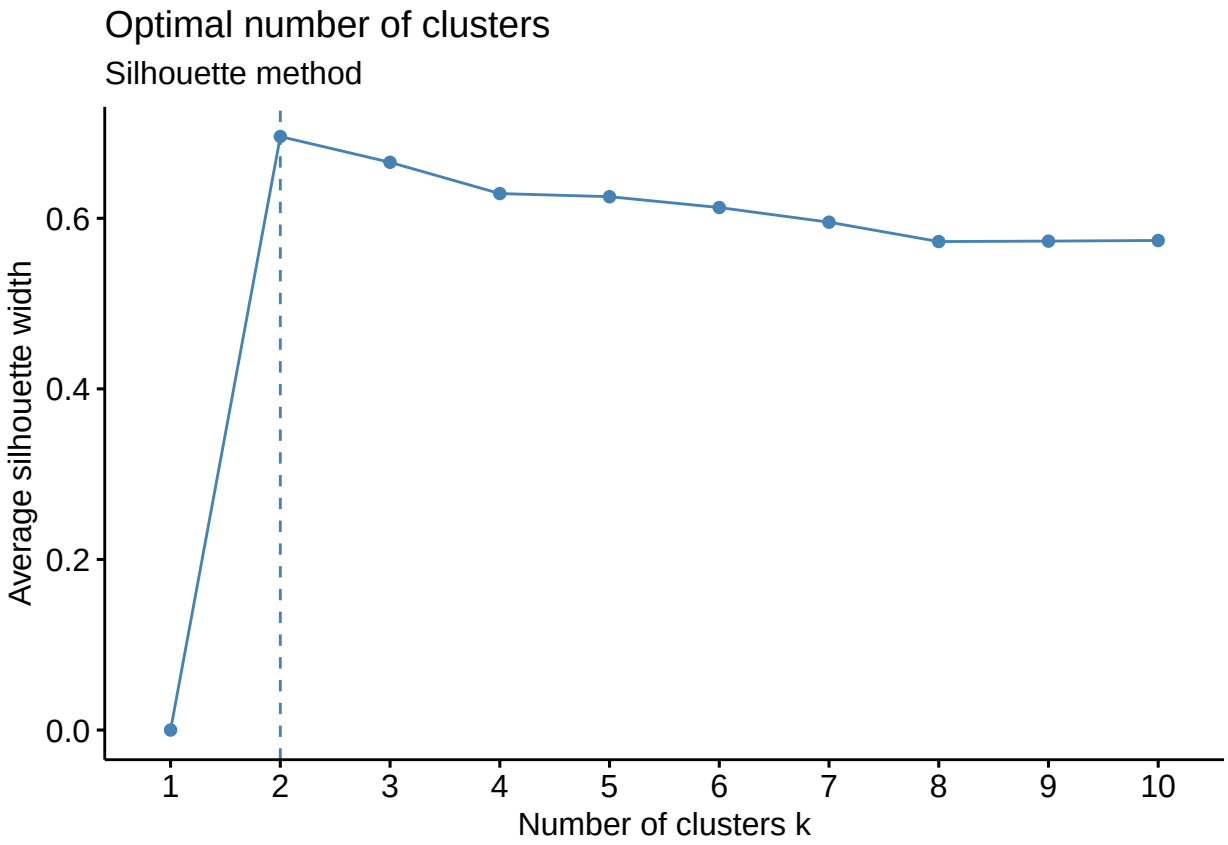



```
k2<-kmeans(md2[,1:3],3 ,iter.max = 100,nstart = 50,algorithm = "Lloyd")
s2<-plot(silhouette(k2$cluster,dist(md2[,1:3],"euclidean")))
```



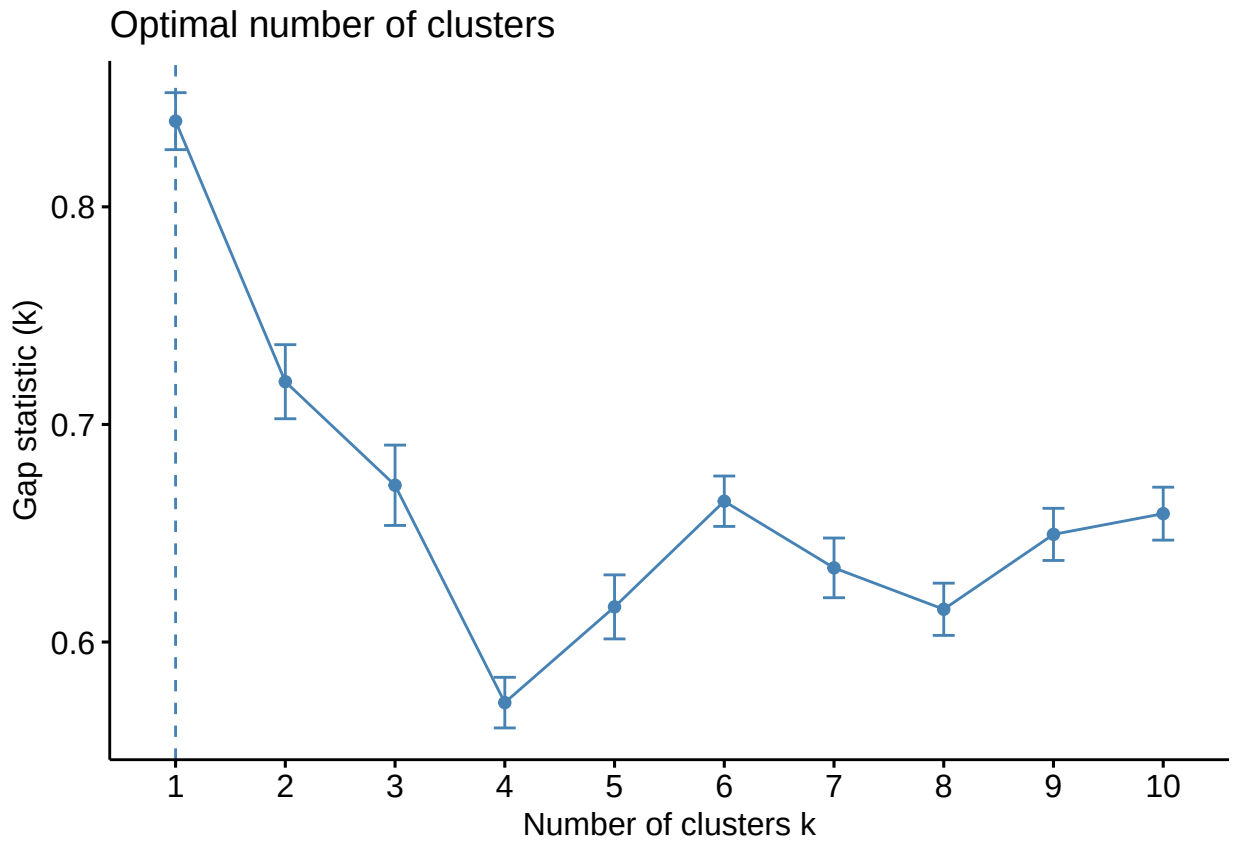
#we make use of the fviz_nbcluster() function #to determine and visualize the optimal number of cluster

```
library(NbClust)
library(factoextra)
fviz_nbclust(md2[,1:3], kmeans, method = "silhouette")+
labs(subtitle = "Silhouette method")
```



#gap statastic method

```
set.seed(123)
stat_gap<-clusGap(md2[,1:3],FUN=kmeans,nstart=25,K.max = 10,B=50)
fviz_gap_stat(stat_gap)
```



#select cluster2

```
k2
```

```
## K-means clustering with 3 clusters of sizes 62, 557, 224
##
## Cluster means:
##   IMDB_Rating Meta_score No_of_Votes
## 1    8.451613   78.37097   1220051.7
## 2    7.854937   78.42370    122482.7
## 3    7.978571   76.73661    536387.0
##
## Clustering vector:
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
##    1    1    1    1    3    1    1    1    1    1    1    1    3    1    1    1
##   17   18   19   20   22   23   24   25   26   27   28   29   30   31   32   33
##    1    1    2    3    1    3    3    1    1    3    1    1    1    2    2    3
##   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48   49
##    1    3    3    1    1    3    1    1    1    1    1    1    2    2    1    2
##   50   51   52   53   54   57   59   60   61   62   63   64   65   67   68   69
##    3    3    2    2    2    2    3    3    3    3    1    1    3    1    3    3
##   70   71   73   74   75   76   79   81   82   85   89   90   91   94   95   96
##    1    3    1    1    3    3    3    2    3    3    2    2    2    1    1    3
##   97   98   99  100  101  102  103  104  105  107  108  109  110  111  112  113
##    3    3    1    3    2    1    1    1    3    3    3    3    1    2    3    2
##  114  115  116  117  118  119  120  121  124  125  127  132  133  136  139  141
##    3    3    2    2    2    2    3    2    2    3    2    3    3    2    2    2
```

```
##  144  145  146  147  148  149  150  151  152  153  156  158  159  160  162  163
##    2    3    1    1    1    2    3    3    3    1    1    3    3    3    3    3
##  165  166  168  169  171  172  173  174  179  180  181  183  184  185  186  187
##    3    3    3    3    2    3    2    2    3    2    2    2    2    2    2    2
##  188  190  191  192  193  196  198  200  201  203  204  205  206  207  210  211
##    2    2    2    2    2    2    2    2    2    3    3    2    2    2    3    3
##  212  213  214  216  217  218  219  220  223  224  226  227  228  231  232  233
##    2    3    3    3    3    2    3    2    3    1    3    3    2    3    3    3
##  235  236  237  238  240  242  243  244  245  246  248  249  250  251  252  253
##    3    3    2    2    2    1    1    3    2    3    2    1    2    1    2    3
##  254  255  256  260  261  262  263  264  267  268  269  270  271  272  273  275
##    2    3    3    2    2    2    3    2    3    3    3    2    2    3    2    2
##  276  277  278  279  280  281  282  284  285  287  289  290  292  294  296  299
##    3    2    3    2    3    2    2    2    2    2    2    2    2    2    2    2
##  301  305  306  307  308  310  312  313  314  315  317  319  324  328  329  330
##    2    2    2    2    2    2    2    2    2    2    2    2    2    3    2    3
##  331  333  334  335  337  338  339  340  341  342  343  344  345  346  347  348
##    3    2    2    2    2    2    3    1    3    3    3    3    3    2    3    3
##  349  350  352  353  355  356  357  358  359  360  361  362  363  364  365  366
##    1    2    2    2    3    3    2    1    2    3    3    3    3    2    3    2
##  368  369  370  371  372  373  374  375  376  377  378  379  381  383  384  385
##    3    3    3    2    2    2    2    2    2    1    3    3    2    3    2    2
##  386  388  389  390  392  393  394  395  396  397  398  399  400  401  402  403
##    2    2    2    2    2    2    3    2    2    3    2    2    3    2    3    2
##  404  405  407  408  409  410  411  412  413  414  415  416  417  418  419  420
##    2    3    3    2    2    2    3    2    2    2    2    3    2    2    2    3
##  422  423  424  425  426  427  428  429  430  431  432  433  435  436  437  438
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  439  440  441  442  443  446  447  448  451  453  455  457  458  459  461  462
##    2    2    2    2    2    2    2    2    2    2    2    2    2    3    2    2
##  463  464  466  467  468  469  470  471  472  473  474  475  476  477  478  479
##    2    3    2    2    2    2    2    2    2    2    3    3    2    3    3    2
##  480  481  482  483  484  485  486  487  488  489  490  492  493  494  495  497
##    3    2    2    3    2    2    2    3    2    3    2    3    2    3    3    2
##  498  499  500  501  502  503  504  505  506  507  508  509  510  511  512  513
##    2    3    2    2    2    1    3    2    3    3    2    2    2    2    3    2
##  514  515  516  517  518  519  520  521  522  524  525  526  527  528  529  530
##    3    2    2    3    2    2    2    2    2    2    3    2    2    2    2    2
##  531  532  533  534  535  536  537  538  540  542  543  544  545  547  548  549
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  550  554  555  556  557  559  560  563  565  567  568  570  572  573  574  576
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    3    2
##  577  579  581  582  583  584  585  586  587  588  589  590  591  593  594  596
##    2    2    2    2    3    3    3    2    2    2    2    2    2    3    2    2
##  597  598  599  600  601  603  604  605  607  608  609  610  611  612  613  614
##    2    2    2    2    3    2    3    3    2    2    2    3    2    2    2    2
##  615  616  617  618  619  620  621  622  623  624  625  626  627  628  629  630
##    3    3    2    2    3    2    2    3    3    1    2    2    3    3    3    2
##  631  632  633  634  635  636  637  638  639  640  641  642  643  644  645  646
##    2    2    2    2    2    2    2    3    2    2    2    2    2    2    2    2
##  647  648  649  650  651  652  653  655  656  657  658  659  660  661  662  663
##    2    3    2    2    2    3    1    2    3    2    2    2    2    2    2    2
##  664  665  668  669  670  671  672  673  674  675  676  677  678  679  680  681
##    2    2    2    2    2    2    2    2    2    2    3    2    3    2    2    2
```

```
##  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696  697
##    2    2    3    2    3    2    2    3    2    2    2    2    2    2    2    2
##  698  699  700  701  702  703  704  705  706  707  708  713  714  715  716  717
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  718  719  720  721  722  723  724  725  726  727  728  729  730  731  732  733
##    2    2    2    2    2    3    2    3    2    2    2    2    3    3    3    2
##  734  735  736  737  738  740  741  742  743  744  745  746  747  748  749  750
##    3    2    2    2    3    2    3    2    2    3    2    3    3    2    3    3
##  751  752  753  754  755  756  757  758  759  760  761  764  765  766  767  768
##    3    3    3    3    3    3    2    2    2    2    2    2    3    2    2    3
##  769  770  771  772  773  774  775  776  777  778  779  780  781  782  783  784
##    2    2    2    3    2    2    2    3    2    3    3    2    3    3    3    2
##  785  786  787  788  789  790  791  792  793  794  795  796  797  798  800  801
##    3    2    2    2    2    2    3    2    2    2    2    3    2    2    2    2
##  802  803  804  806  807  808  809  810  811  812  814  815  816  817  818  819
##    2    3    2    2    2    3    2    2    2    2    2    2    2    2    2    2
##  820  821  822  823  824  825  826  827  828  829  831  832  833  834  835  836
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  837  839  840  841  842  843  844  845  846  847  848  849  850  851  852  853
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  854  855  856  857  859  860  861  862  863  864  866  867  868  869  870  871
##    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  874  876  877  878  879  880  881  882  883  884  885  886  887  888  889  890
##    2    2    2    2    2    3    2    2    2    2    2    2    2    3    3    2
##  891  892  893  894  895  896  897  898  899  901  902  903  904  905  906  907
##    2    2    2    3    2    2    2    2    3    2    2    2    3    2    2    3
##  908  909  911  912  913  915  916  917  918  919  920  921  922  923  924  925
##    2    3    3    2    3    3    2    2    2    2    2    2    3    2    2    2
##  926  927  928  929  930  931  932  933  934  935  936  937  938  939  940  941
##    3    2    3    3    2    3    2    3    2    2    2    2    2    2    2    2
##  942  943  944  945  946  947  948  949  950  951  952  953  954  955  956  957
##    2    3    3    2    2    2    3    3    2    2    3    2    3    2    2    2
##  958  959  960  961  962  963  964  965  966  967  968  969  970  971  972  973
##    2    2    2    2    2    2    3    2    2    2    2    2    2    2    2    2
##  974  975  976  977  978  979  980  981  982  983  984  985  986  987  988  989
##    3    3    2    2    2    2    2    2    2    2    2    2    2    2    2    2
##  990  991  992  993  994  995  996  997  998  999 1000
##    2    2    2    2    2    2    2    2    2    2    2
##
## Within cluster sum of squares by cluster:
## [1] 7418707707912 3814453102634 4729359642907
##  (between_SS / total_SS =  83.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
pcluster<-prcomp(md2[,1:3],scale. = FALSE)
summary(pcluster)
```

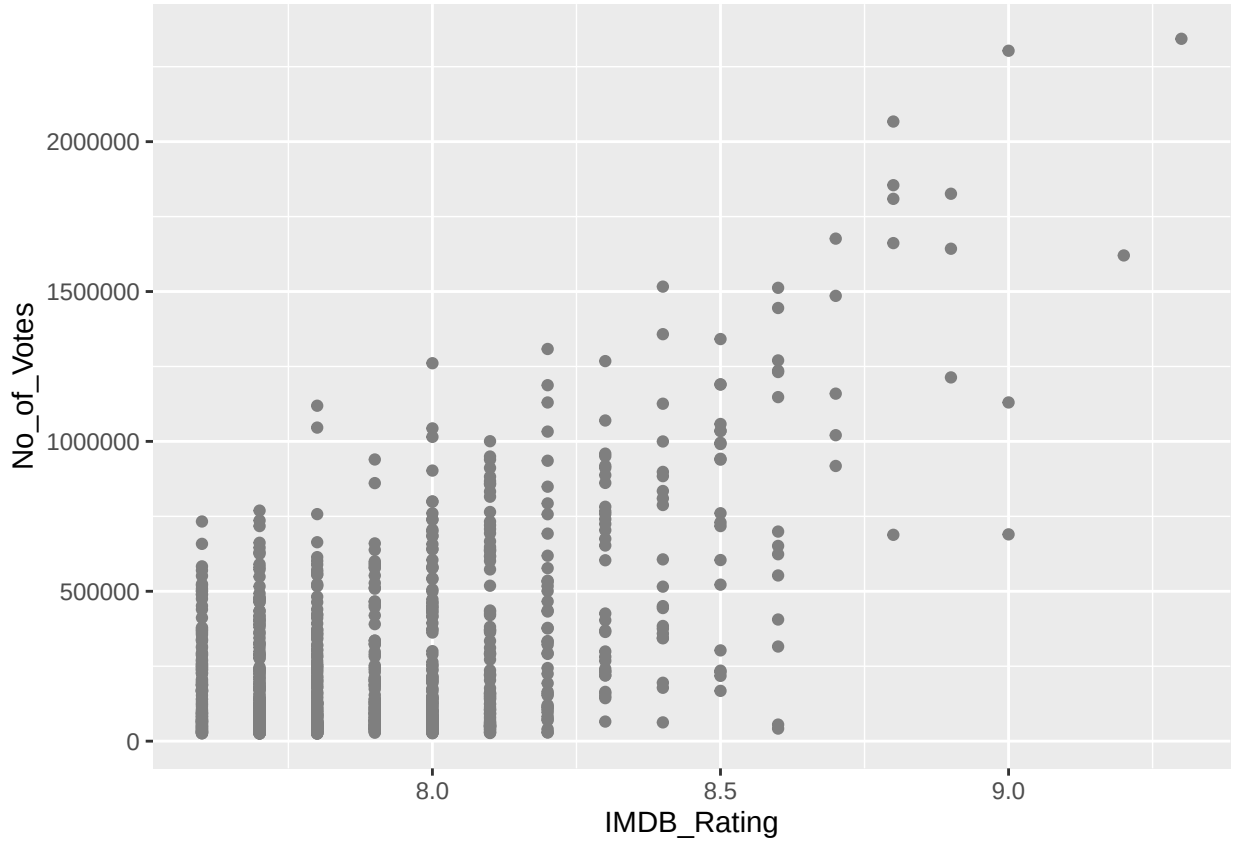
```
## Importance of components:
##                           PC1   PC2    PC3
## Standard deviation     341799 12.37 0.2157
```

```
## Proportion of Variance      1  0.00 0.0000
## Cumulative Proportion       1  1.00 1.0000
```

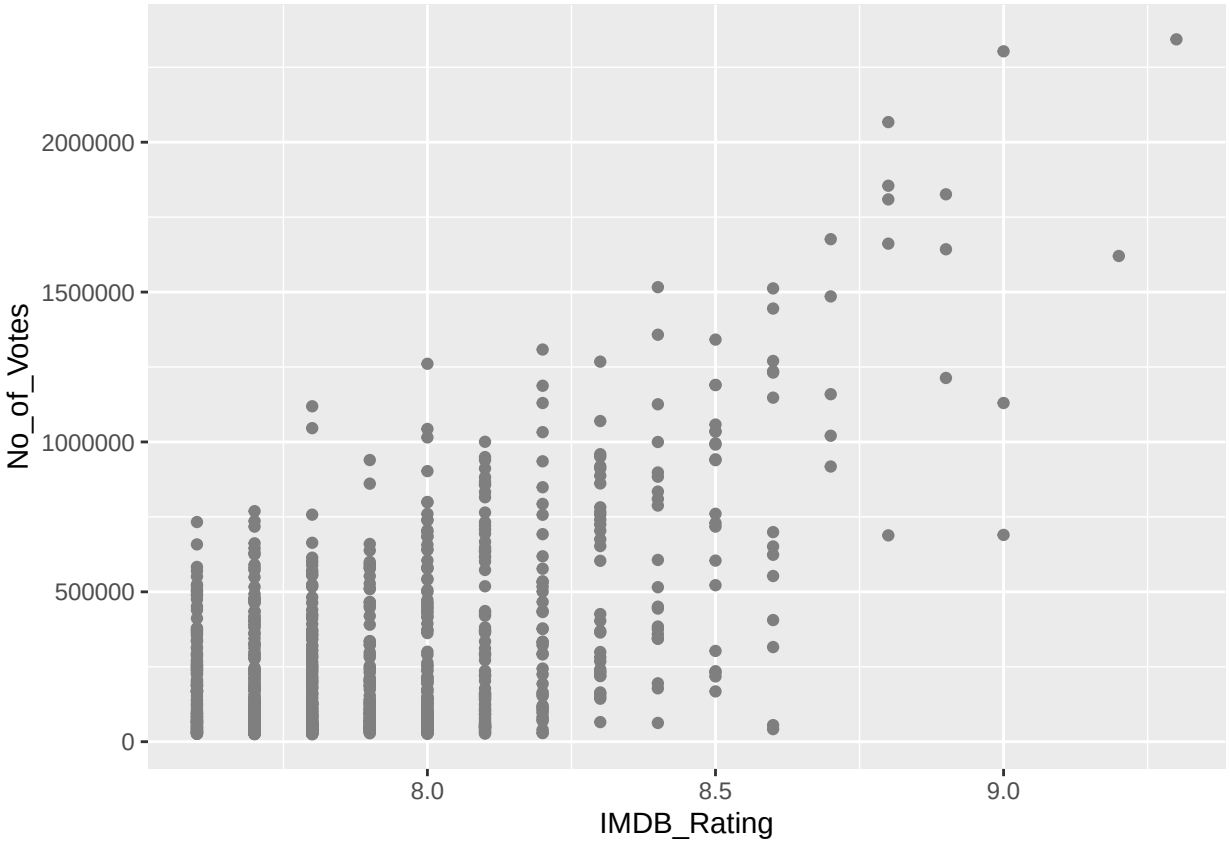
```
pcluster$rotation[,1:2]
```

```
##                            PC1              PC2
## IMDB_Rating  0.0000004869986 -0.0064092136358
## Meta_score  -0.0000006701138 -0.9999794607791
## No_of_Votes  0.9999999999997 -0.0000006669787
```

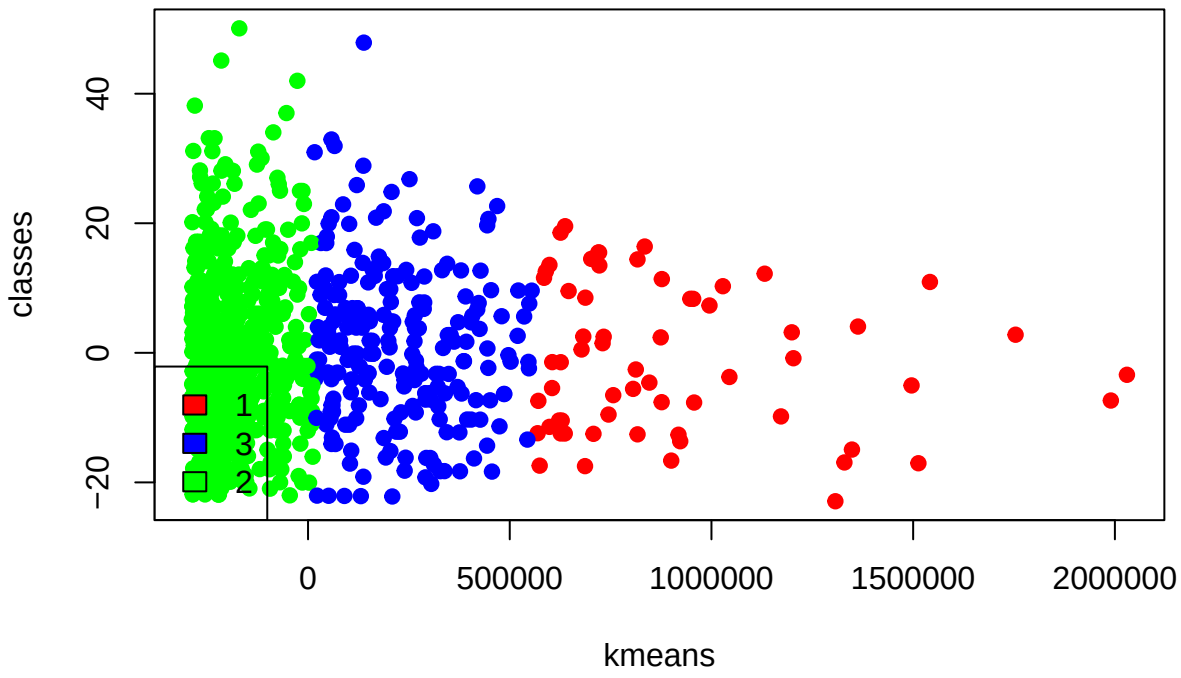
```
set.seed(123)
ggplot(md2,aes(x=IMDB_Rating,y=No_of_Votes))+
  geom_point(stat="identity",aes(color=as.factor(k2$cluster)))+
  scale_color_discrete(name="  ",breaks=c("1","2","3"),
                       labels=c("cluster1","cluster2","cluster3"), ggtitle("segments of demographics",su
```




```
set.seed(123)
ggplot(md2,aes(x=IMDB_Rating,y=No_of_Votes))+
  geom_point(stat="identity",aes(color=as.factor(k2$cluster)))+
  scale_color_discrete(name="  ",breaks=c("1","2","3"),
                       labels=c("cluster1","cluster2","cluster3"), ggtitle("segments of demographics",su
```




```
kcols=function(vec){cols=rainbow(length(unique(vec)))
return (cols[as.numeric(as.factor(vec))])}
digCluster<-k2$cluster; dignm<-as.character(digCluster);
plot(pcluster$x[,1:2],col=kcols(digCluster),pch=19,xlab="kmeans",ylab="classes")
legend("bottomleft",unique(dignm),fill = unique(kcols(digCluster)))
```


classes
40
20
0
−20
1
3
2
0
500000
1000000
1500000
2000000
kmeans