R Code:

```
install.packages("dplyr")
install.packages("readr")
install.packages("ggplot2")
install.packages("scales")
install.packages("ggthemes")
install.packages("tidyr")
install.packages("AppliedPredictiveModeling")
install.packages("outliers")
install.packages("factoextra")
install.packages("cluster")
install.packages("ggfortify")
install.packages("fpc")
install.packages("NbClust")
library(dplyr)
library(readr)
library(ggplot2)
library(scales)
library(ggthemes)
library(tidyr)
library(NbClust)
library(cluster)
library(fpc)
library(outliers)
library(factoextra)
#--------- read -------
help(read.csv)
?read.csv
```

```
googleplaystore <- read.csv("C:/Users/mrymh/OneDrive/Desktop/Data Mining Final submission/IT326-Dataset-section-52845-Group[2]/googleplaystore.csv")


#------------ Rating vs.price scatter plot-------


{r echo= FALSE, message=FALSE, warning=FALSE, Bivariate_Plots}
#plot of rating vs. Price
ggplot(aes(x = Rating , y = Price), data = df)+
  geom_jitter(alpha = 0.3, color = 'royalblue1')+
  ylim(0,25)+
  geom_line(stat = 'summary', fun.y = mean)+  ggtitle('Rating vs. Price')



#--------- Plot for top genres vs. category  --------------
ggplot(aes (x = Genres), data = topgenres)+
  geom_bar(aes(fill = Category))+
  coord_flip()+
  ggtitle('Top genres and Category')


#--------- Pie chart for Type ---------------


{r echo= FALSE,message=FALSE, warning=FALSE}
#There are two types paid and unpaid
df_type = subset(df, (Type == 'Free' | Type == 'Paid'))
temp <- df_type%>%
  group_by(Type)%>%
  summarise(n = n())
#pie chart
```

```
ggplot(aes(x = '', y = n, fill = Type), data = temp )+
  geom_bar(stat = 'identity')+
  coord_polar('y', start = 0)+
  theme_void()+
  ggtitle('Type')


#--------------preprocessing-------------------------
# ---------Replace missing values---------
googleplaystore$Rating[is.nan(googleplaystore$Rating)]<-0.0
#---------------Remove outliers -----------
outrev = outlier(googleplaystore$Rating, logical = TRUE)
sum(outrev)
Find_outlier = which(outrev ==TRUE, arr.ind = TRUE)
googleplaystore= googleplaystore[-Find_outlier,]
#--------------------
outGen = outlier(googleplaystore$Genres, logical = TRUE)
sum(outGen)
Find_outlier = which(outGen ==TRUE, arr.ind = TRUE)
googleplaystore= googleplaystore[-Find_outlier,]
#------------------
outCon = outlier(googleplaystore$Content.Rating, logical = TRUE)
sum(outCon)
Find_outlier = which(outCon ==TRUE, arr.ind = TRUE)
googleplaystore= googleplaystore[-Find_outlier,]
#--------------------
outcat = outlier(googleplaystore$Category, logical = TRUE)
sum(outcat)
Find_outlier = which(outcat ==TRUE, arr.ind = TRUE)
googleplaystore= googleplaystore[-Find_outlier,]
```

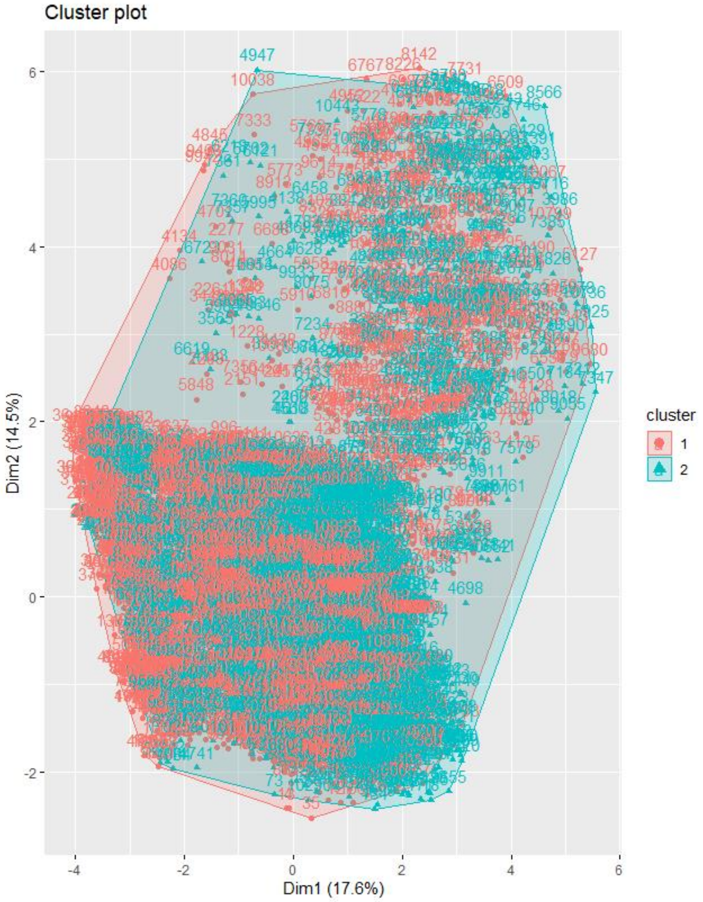
```
outrevi = outlier(googleplaystore$Reviews, logical = TRUE)

sum(outrevi)

Find_outlier = which(outrevi ==TRUE, arr.ind = TRUE)

googleplaystore= googleplaystore[-Find_outlier,]


#----- ENCODING CATEGORICAL DATA------


googleData$Type<- sapply(googleData$Type,as.numeric)


##---------------------Data mining task #----------------------

#-------- convert to numeric --------------------------



googleplaystore$Category <- sapply(googleplaystore$Category,as.numeric)

googleplaystore$Reviews <- sapply(googleplaystore$Reviews,as.numeric)

googleplaystore$Size <- sapply(googleplaystore$Size,as.numeric)

googleplaystore$Installs <- sapply(googleplaystore$Installs,as.numeric)

googleplaystore$Type <- sapply(googleplaystore$Type,as.numeric)

googleplaystore$Price <- sapply(googleplaystore$Price,as.numeric)

googleplaystore$Content.Rating <- sapply(googleplaystore$Content.Rating,as.numeric)

googleplaystore$Genres <- sapply(googleplaystore$Genres,as.numeric)

googleplaystore$Last.Updated <- sapply(googleplaystore$Last.Updated,as.numeric)

googleplaystore$Current.Ver<- sapply(googleplaystore$Current.Ver,as.numeric)

googleplaystore$Android.Ver<- sapply(googleplaystore$Android.Ver,as.numeric)

googleplaystore$App<- sapply(googleplaystore$App,as.numeric)
```

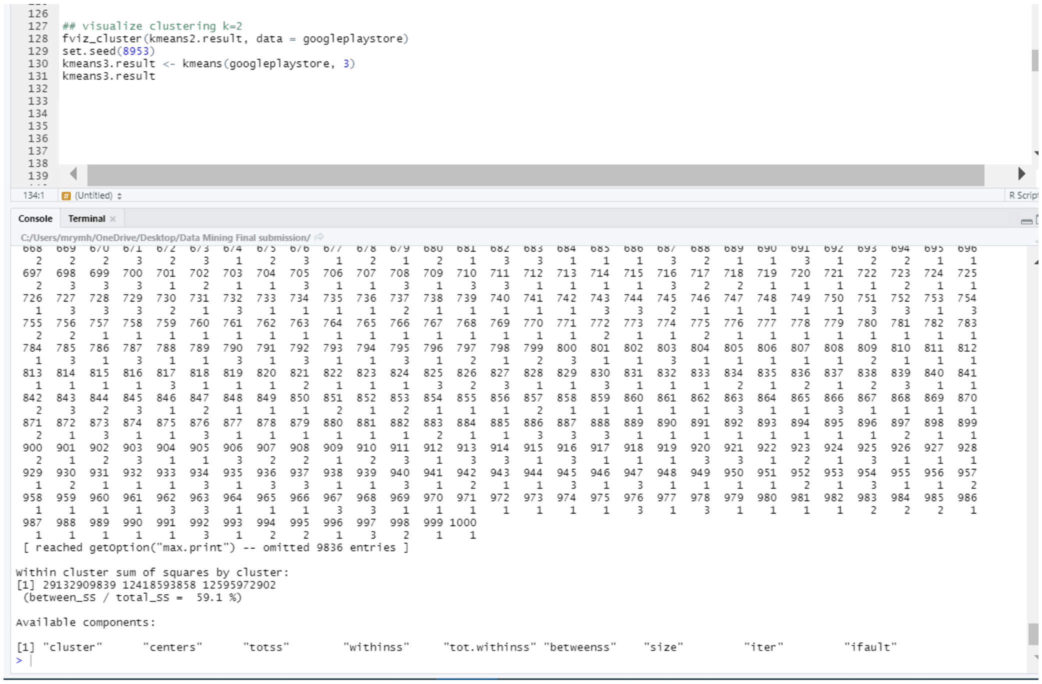
```
#------------------
kmeans2.result <- kmeans(googleplaystore,2)
kmeans2.result

## visualize clustering k=2
fviz_cluster(kmeans2.result, data = googleplaystore)
set.seed(8953)
kmeans3.result <- kmeans(googleplaystore, 3)
kmeans3.result

## visualize clustering k=3
fviz_cluster(kmeans3.result, data = googleplaystore)
kmeans4.result <- kmeans(googleplaystore, 4)
kmeans4.result

## visualize clustering k=4
fviz_cluster(kmeans4.result, data = googleplaystore)
kmeans5.result <- kmeans(googleplaystore, 5)
kmeans5.result
## visualize clustering k=5
fviz_cluster(kmeans5.result, data = googleplaystore)
##--------------------------
# group into 4 clusters
pam1.result <- pam(googleplaystore, 2)
plot(pam1.result)
fviz_cluster(pam1.result, data = googleplaystore)
pam2.result <- pam(googleplaystore, 3)
fviz_cluster(pam2.result, data = googleplaystore)
```

```
plot(pam2.result)

pam3.result <- pam(googleplaystore, 4)

fviz_cluster(pam3.result, data = googleplaystore)


plot(pam3.result)

pam4.result <- pam(googleplaystore, 5)

fviz_cluster(pam4.result, data = googleplaystore)


plot(pam4.result)


#---Evaluation

##--------------------------

# group into 4 clusters

pam.result <- pam(googleplaystore, 2)

plot(pam.result)

pam.result <- pam(googleplaystore, 3)

plot(pam.result)

pam.result <- pam(googleplaystore, 4)

plot(pam.result)

pam.result <- pam(googleplaystore, 5)

plot(pam.result)

##for all clusters


fviz_nbclust(googleplaystore, kmeans, method = "silhouette")+ labs(subtitle = "Silhouette method")


#extra heririchal clustering


install.packages("pvcluster")
```

```
library(pvclust)

dd <- dist(scale(googleplaystore), method = "euclidean")
hc <- hclust(dd, method = "ward.D")
plot(hc)


set.seed(1234)
result <- pvclust(googleplaystore[1:100, 1:10], method.dist="euclidean",
                 method.hclust="average", nboot=10)
plot(result)
pvrect(result)

dend <- googleplaystore[1:30,-5] %>% scale %>% dist %>%
  hclust %>% as.dendrogram %>%
  set("branches_k_color", k=3) %>% set("branches_lwd", 1.2) %>%
  set("labels_colors") %>% set("labels_cex", c(.9,1.2)) %>%
  set("leaves_pch", 19) %>% set("leaves_col", c("blue", "red"))
# plot the dend in usual "base" plotting engine:
plot(dend)
```

Figures:

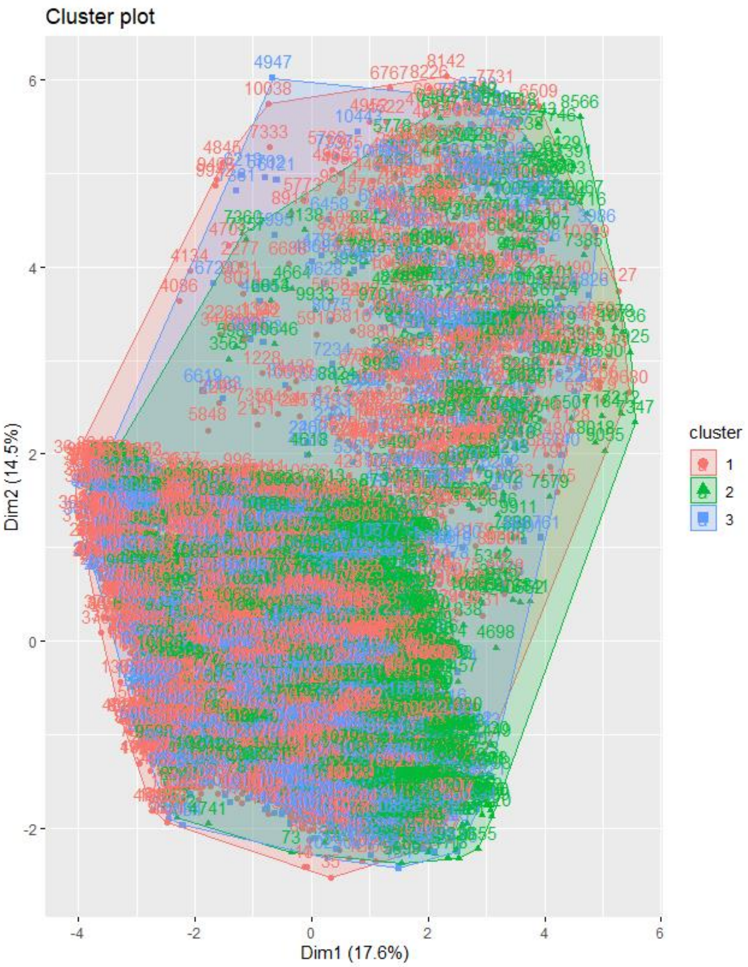


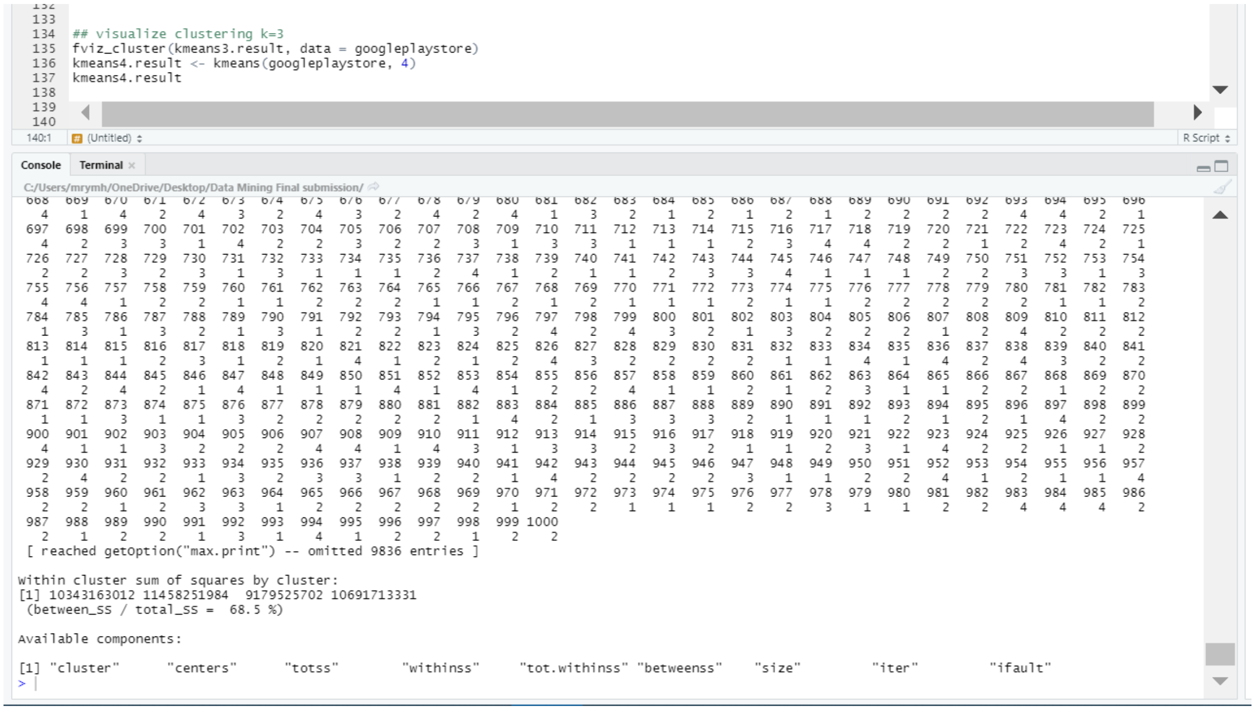


```
## visualize clustering k=2
fviz_cluster(kmeans2.result, data = googleplaystore)
set.seed(8953)
kmeans3.result <- kmeans(googleplaystore, 3)
kmeans3.result
```

134:1
(Untitled)
R Script
Console
Terminal
C:/Users/mrymh/OneDrive/Desktop/Data Mining Final submission/

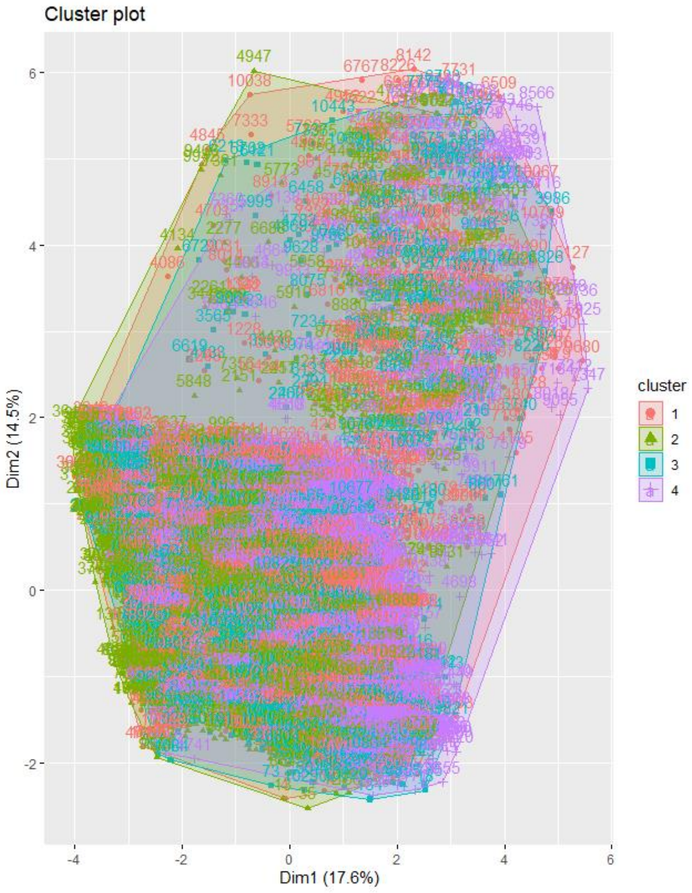
```
 668  669  670  671  672  673  674  675  676  677  678  679  680  681  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696 
   2    2    2    3    2    3    1    2    3    1    2    1    2    1    3    3    1    1    1    3    2    1    1    3    1    2    2    1    1 
 697  698  699  700  701  702  703  704  705  706  707  708  709  710  711  712  713  714  715  716  717  718  719  720  721  722  723  724  725 
   2    3    3    3    1    2    1    1    3    1    1    3    1    3    3    1    1    1    1    3    2    2    1    1    1    1    2    1    1 
 726  727  728  729  730  731  732  733  734  735  736  737  738  739  740  741  742  743  744  745  746  747  748  749  750  751  752  753  754 
   1    3    3    3    2    1    3    1    1    1    1    2    1    1    1    1    1    3    3    2    1    1    1    1    1    3    3    1    3 
 755  756  757  758  759  760  761  762  763  764  765  766  767  768  769  770  771  772  773  774  775  776  777  778  779  780  781  782  783 
   2    2    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    2    1    1    2    1    1    1    1    1    1    1    1 
 784  785  786  787  788  789  790  791  792  793  794  795  796  797  798  799  800  801  802  803  804  805  806  807  808  809  810  811  812 
   1    3    1    3    1    1    3    1    3    1    1    3    1    2    1    2    3    1    1    3    1    1    1    1    1    2    1    1    1 
 813  814  815  816  817  818  819  820  821  822  823  824  825  826  827  828  829  830  831  832  833  834  835  836  837  838  839  840  841 
   1    1    1    1    3    1    1    1    2    1    1    1    3    2    3    1    1    3    1    1    1    2    1    2    1    2    3    1    1 
 842  843  844  845  846  847  848  849  850  851  852  853  854  855  856  857  858  859  860  861  862  863  864  865  866  867  868  869  870 
   2    3    2    3    1    2    1    1    1    2    1    2    1    1    1    2    1    1    1    1    1    3    1    1    3    1    1    1    1 
 871  872  873  874  875  876  877  878  879  880  881  882  883  884  885  886  887  888  889  890  891  892  893  894  895  896  897  898  899 
   2    1    3    1    1    3    1    1    1    1    1    1    2    1    1    3    3    3    1    1    1    1    1    1    1    1    2    1    1 
 900  901  902  903  904  905  906  907  908  909  910  911  912  913  914  915  916  917  918  919  920  921  922  923  924  925  926  927  928 
   2    1    2    3    1    1    3    2    2    1    2    3    1    3    3    1    3    1    1    1    3    3    1    2    1    3    1    1    1 
 929  930  931  932  933  934  935  936  937  938  939  940  941  942  943  944  945  946  947  948  949  950  951  952  953  954  955  956  957 
   1    2    1    1    1    3    1    3    3    1    1    3    1    2    1    1    3    1    3    1    1    1    1    2    1    3    1    1    2 
 958  959  960  961  962  963  964  965  966  967  968  969  970  971  972  973  974  975  976  977  978  979  980  981  982  983  984  985  986 
   1    1    1    1    3    3    1    1    1    3    3    1    1    1    1    1    1    1    3    1    3    1    1    1    1    2    2    2    1 
 987  988  989  990  991  992  993  994  995  996  997  998  999 1000 
   1    1    1    1    1    3    1    2    2    1    3    2    1    1 
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 29132909839 12418593858 12595972902
 (between_SS / total_SS =  59.1 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"      
> 
```

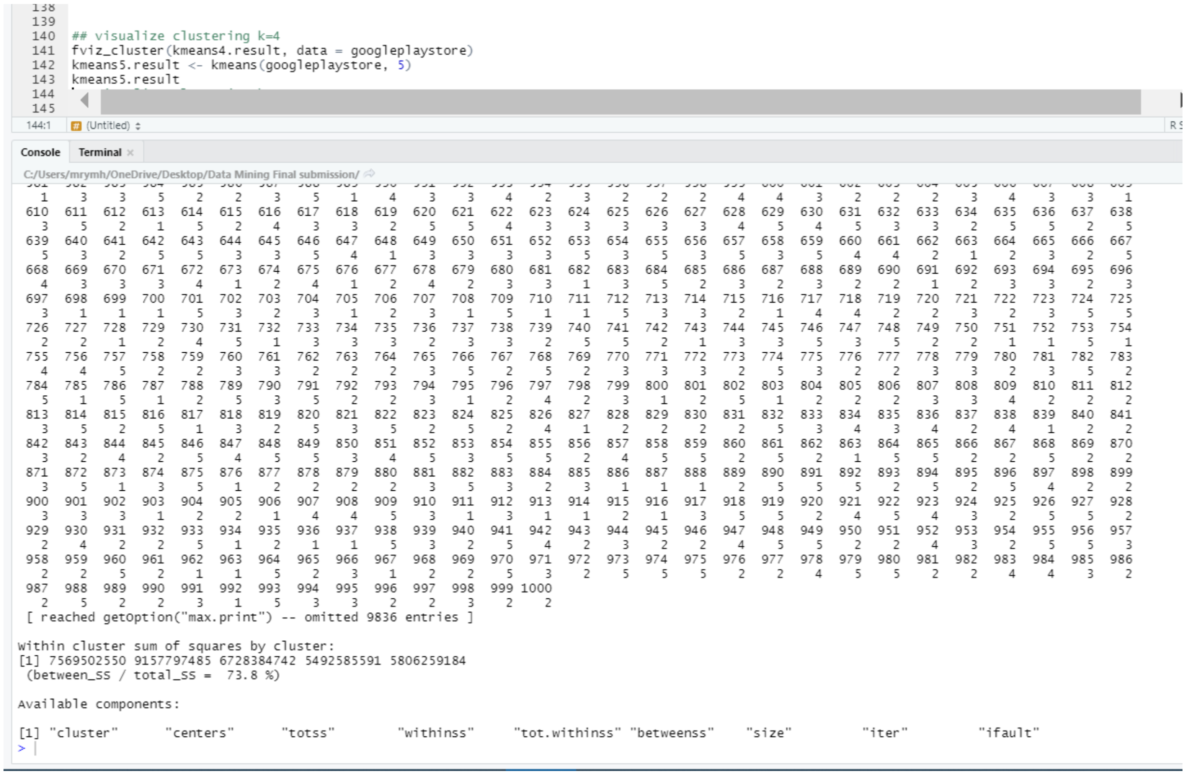



140:1
(Untitled)
R Script
Console
Terminal
C:/Users/mrymh/OneDrive/Desktop/Data Mining Final submission/

```r
## visualize clustering k=3
fviz_cluster(kmeans3.result, data = googleplaystore)
kmeans4.result <- kmeans(googleplaystore, 4)
kmeans4.result
```

```
 668  669  670  671  672  673  674  675  676  677  678  679  680  681  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696 
   4    1    4    2    4    3    2    4    3    2    4    2    4    1    3    2    1    2    1    2    1    2    2    2    2    4    4    2    1 
 697  698  699  700  701  702  703  704  705  706  707  708  709  710  711  712  713  714  715  716  717  718  719  720  721  722  723  724  725 
   4    2    3    3    1    4    2    2    3    2    2    3    1    3    3    1    1    1    2    3    4    4    2    2    1    2    4    2    1 
 726  727  728  729  730  731  732  733  734  735  736  737  738  739  740  741  742  743  744  745  746  747  748  749  750  751  752  753  754 
   2    2    3    2    3    1    3    1    1    1    2    4    1    2    1    1    2    3    3    4    1    1    1    2    2    3    3    1    3 
 755  756  757  758  759  760  761  762  763  764  765  766  767  768  769  770  771  772  773  774  775  776  777  778  779  780  781  782  783 
   4    4    1    2    2    1    1    2    2    2    1    1    2    1    2    1    1    1    2    1    1    2    2    2    2    2    1    1    2 
 784  785  786  787  788  789  790  791  792  793  794  795  796  797  798  799  800  801  802  803  804  805  806  807  808  809  810  811  812 
   1    3    1    3    2    1    3    1    2    2    1    3    2    4    2    4    3    2    1    3    2    2    2    1    2    4    2    2    2 
 813  814  815  816  817  818  819  820  821  822  823  824  825  826  827  828  829  830  831  832  833  834  835  836  837  838  839  840  841 
   1    1    1    2    3    1    2    1    4    1    2    1    2    4    3    2    2    2    2    1    1    4    1    4    2    4    3    2    2 
 842  843  844  845  846  847  848  849  850  851  852  853  854  855  856  857  858  859  860  861  862  863  864  865  866  867  868  869  870 
   4    2    4    2    1    4    1    1    1    4    1    4    1    2    2    4    1    1    2    1    2    3    1    1    2    2    1    2    2 
 871  872  873  874  875  876  877  878  879  880  881  882  883  884  885  886  887  888  889  890  891  892  893  894  895  896  897  898  899 
   1    1    3    1    1    3    2    2    2    2    2    1    4    2    1    3    3    3    2    1    1    1    2    1    2    1    4    2    2 
 900  901  902  903  904  905  906  907  908  909  910  911  912  913  914  915  916  917  918  919  920  921  922  923  924  925  926  927  928 
   4    1    1    3    2    2    2    4    4    1    4    3    1    3    3    2    3    2    1    1    2    3    1    4    2    2    1    1    2 
 929  930  931  932  933  934  935  936  937  938  939  940  941  942  943  944  945  946  947  948  949  950  951  952  953  954  955  956  957 
   2    4    2    2    1    3    2    3    3    1    2    2    1    4    2    2    2    2    3    1    1    2    2    4    1    2    1    1    4 
 958  959  960  961  962  963  964  965  966  967  968  969  970  971  972  973  974  975  976  977  978  979  980  981  982  983  984  985  986 
   2    2    1    2    3    3    1    2    2    2    2    2    1    2    2    1    1    1    2    2    3    1    1    2    2    4    4    4    2 
 987  988  989  990  991  992  993  994  995  996  997  998  999 1000 
   2    1    2    2    1    3    1    4    1    2    2    1    2    2 
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 10343163012 11458251984  9179525702 10691713331
 (between_SS / total_SS =  68.5 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"      
> 
```

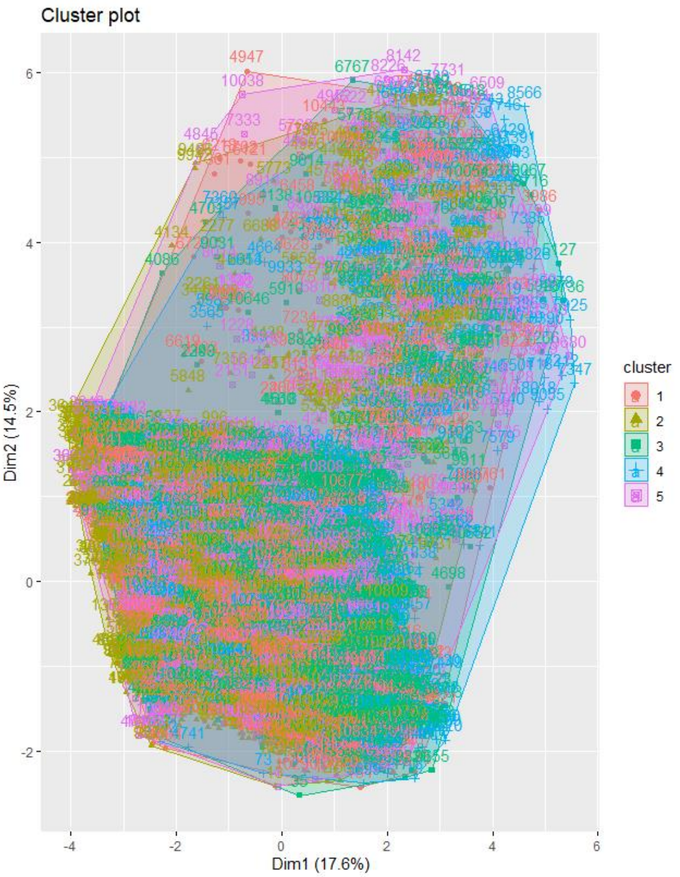


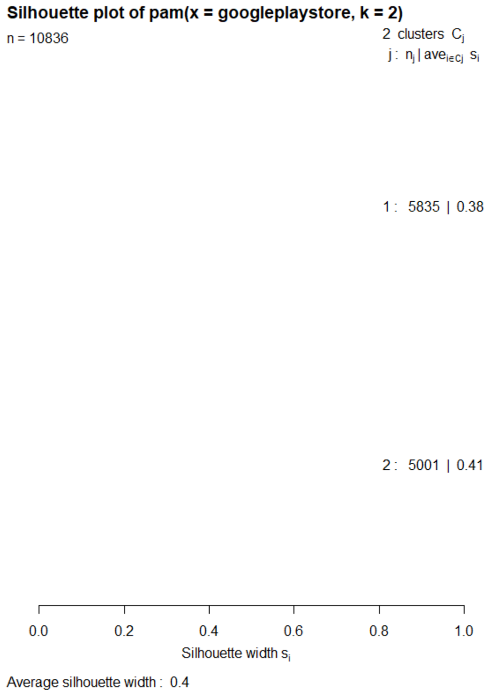


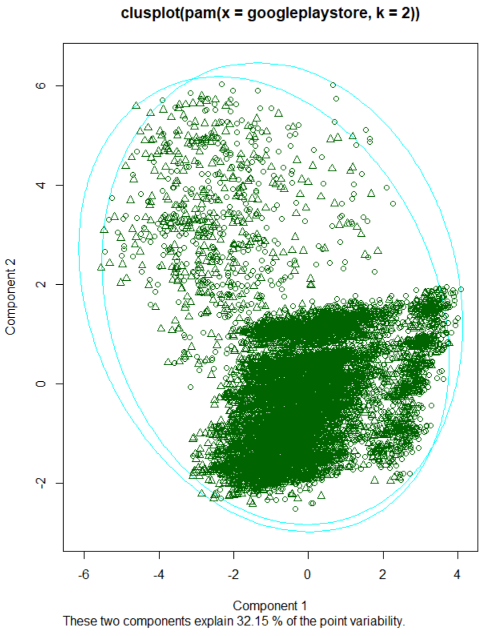
```
## visualize clustering k=4
fviz_cluster(kmeans4.result, data = googleplaystore)
kmeans5.result <- kmeans(googleplaystore, 5)
kmeans5.result
```

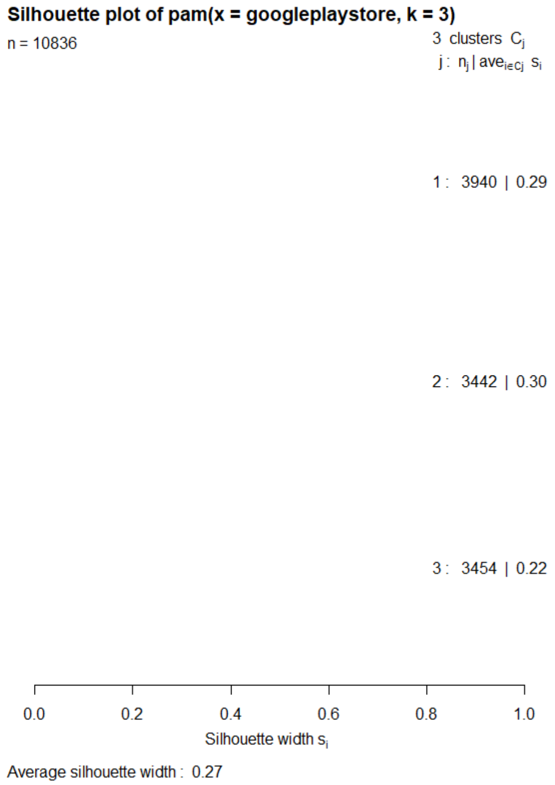
C:/Users/mrymh/OneDrive/Desktop/Data Mining Final submission/

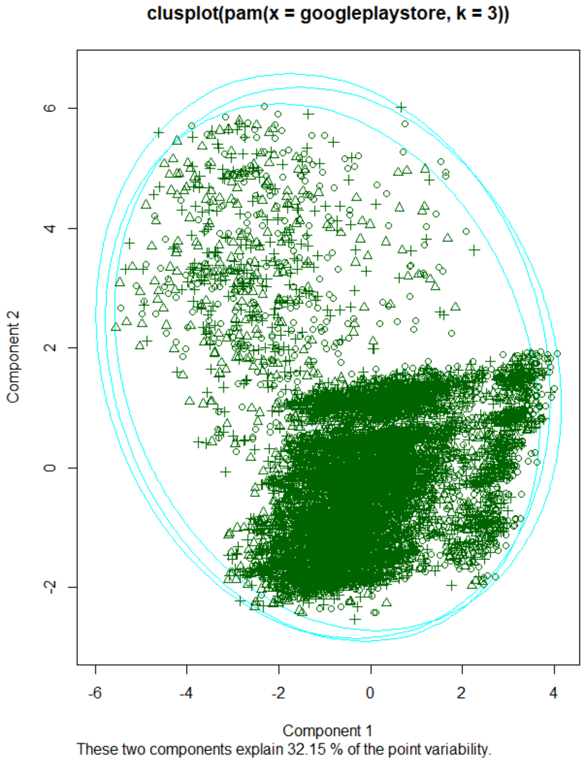
```
    1    3    3    5    2    2    3    5    1    4    3    3    4    2    3    2    2    2    4    4    3    2    2    2    3    4    3    3    1
  610  611  612  613  614  615  616  617  618  619  620  621  622  623  624  625  626  627  628  629  630  631  632  633  634  635  636  637  638
    3    5    2    1    5    2    4    3    3    2    5    5    4    3    3    3    3    3    4    5    4    5    3    3    2    5    5    2    5
  639  640  641  642  643  644  645  646  647  648  649  650  651  652  653  654  655  656  657  658  659  660  661  662  663  664  665  666  667
    5    3    2    5    5    3    3    5    4    1    3    3    3    3    5    3    5    3    5    3    5    4    4    2    1    2    3    2    5
  668  669  670  671  672  673  674  675  676  677  678  679  680  681  682  683  684  685  686  687  688  689  690  691  692  693  694  695  696
    4    3    3    3    4    1    2    4    1    2    4    2    3    3    1    3    5    2    3    2    3    2    2    1    2    3    3    2    3
  697  698  699  700  701  702  703  704  705  706  707  708  709  710  711  712  713  714  715  716  717  718  719  720  721  722  723  724  725
    3    1    1    1    5    3    2    3    1    2    3    1    5    1    1    5    3    3    2    1    4    4    2    2    3    2    3    5    5
  726  727  728  729  730  731  732  733  734  735  736  737  738  739  740  741  742  743  744  745  746  747  748  749  750  751  752  753  754
    2    2    1    2    4    5    1    3    3    3    2    3    3    2    5    5    2    1    3    3    5    3    5    2    2    1    1    5    1
  755  756  757  758  759  760  761  762  763  764  765  766  767  768  769  770  771  772  773  774  775  776  777  778  779  780  781  782  783
    4    4    5    2    2    3    3    2    2    2    3    5    2    5    2    3    3    3    2    5    3    2    2    3    3    2    3    5    2
  784  785  786  787  788  789  790  791  792  793  794  795  796  797  798  799  800  801  802  803  804  805  806  807  808  809  810  811  812
    5    1    5    1    2    5    3    5    2    2    3    1    2    4    2    3    1    2    5    1    2    2    2    3    3    4    2    2    2
  813  814  815  816  817  818  819  820  821  822  823  824  825  826  827  828  829  830  831  832  833  834  835  836  837  838  839  840  841
    3    5    2    5    1    3    2    5    3    5    2    5    2    4    1    2    2    2    2    5    3    4    3    4    2    4    1    2    2
  842  843  844  845  846  847  848  849  850  851  852  853  854  855  856  857  858  859  860  861  862  863  864  865  866  867  868  869  870
    3    2    4    2    5    4    5    5    3    4    5    3    5    5    2    4    5    5    2    5    2    1    5    5    2    2    5    2    2
  871  872  873  874  875  876  877  878  879  880  881  882  883  884  885  886  887  888  889  890  891  892  893  894  895  896  897  898  899
    3    5    1    3    5    1    2    2    2    2    3    5    3    2    3    1    1    1    2    5    5    5    2    5    2    5    4    2    2
  900  901  902  903  904  905  906  907  908  909  910  911  912  913  914  915  916  917  918  919  920  921  922  923  924  925  926  927  928
    3    3    3    1    2    2    1    4    4    5    3    1    3    1    1    2    1    3    5    5    2    4    5    4    3    2    5    5    2
  929  930  931  932  933  934  935  936  937  938  939  940  941  942  943  944  945  946  947  948  949  950  951  952  953  954  955  956  957
    2    4    2    2    5    1    2    1    1    5    3    2    5    4    2    3    2    2    4    5    5    2    2    4    3    2    5    5    3
  958  959  960  961  962  963  964  965  966  967  968  969  970  971  972  973  974  975  976  977  978  979  980  981  982  983  984  985  986
    2    2    5    2    1    1    5    2    3    1    2    2    5    3    2    5    5    5    2    2    4    5    5    2    2    4    4    3    2
  987  988  989  990  991  992  993  994  995  996  997  998  999 1000
    2    5    2    2    3    1    5    3    3    2    2    3    2    2
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 7569502550 9157797485 6728384742 5492585591 5806259184
 (between_SS / total_SS =  73.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"      
> 
```

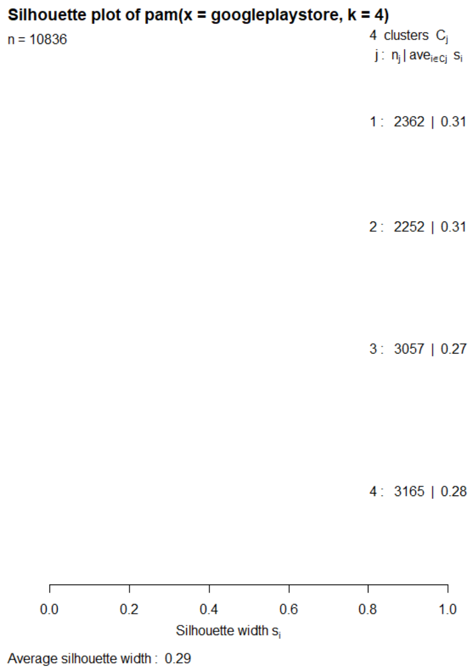

Cluster plot
Dim2 (14.5%)
Dim1 (17.6%)
6
4
2
0
-2
-4
-2
0
2
4
6
cluster
1
2
3
4
5
4947
10038
6767
8142
8226
7731
6509
8566
7333
4846
5848
4086
4134
3565
4698
7347
4741

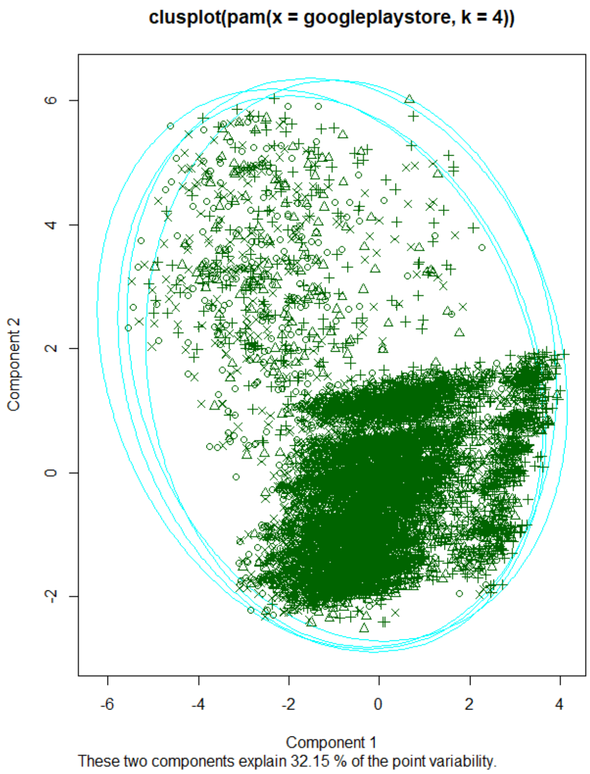

Silhouette plot of pam(x = googleplaystore, k = 2)
n = 10836
2 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 5835 | 0.38
2 : 5001 | 0.41
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.4

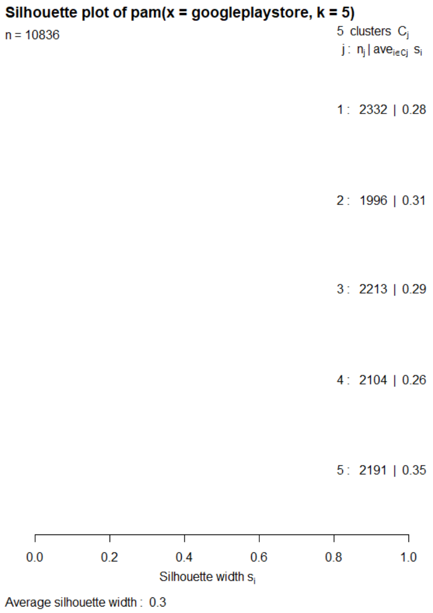

clusplot(pam(x = googleplaystore, k = 2))
Component 2
6
4
2
0
-2
-6
-4
-2
0
2
4
Component 1
These two components explain 32.15 % of the point variability.

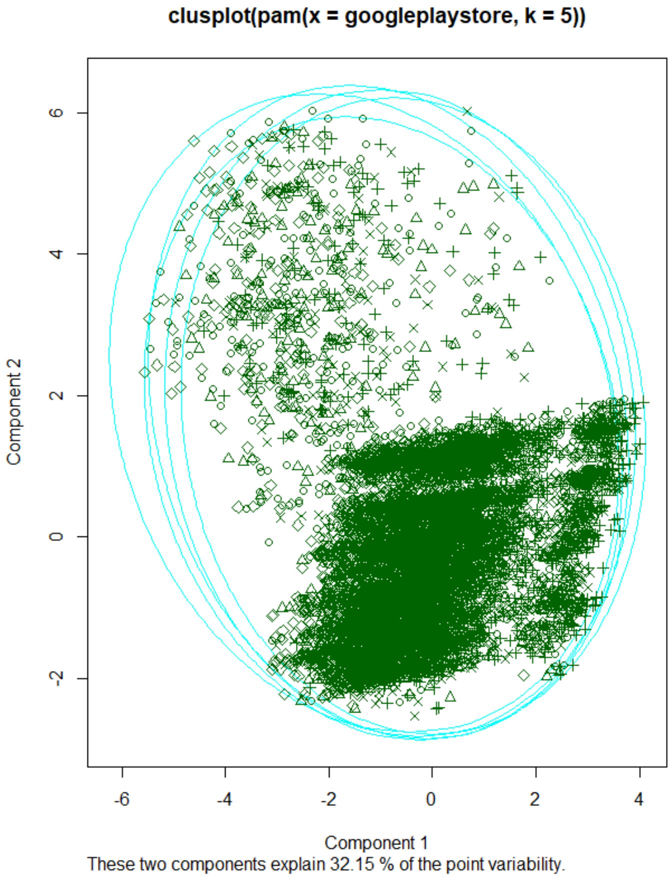

Silhouette plot of pam(x = googleplaystore, k = 3)
n = 10836
3 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 3940 | 0.29
2 : 3442 | 0.30
3 : 3454 | 0.22
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.27

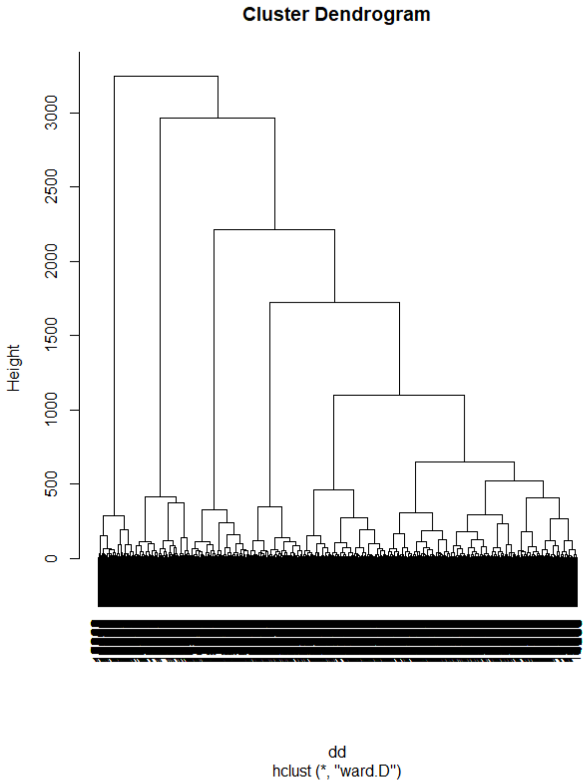

clusplot(pam(x = googleplaystore, k = 3))
Component 2
6
4
2
0
-2
-6
-4
-2
0
2
4
Component 1
These two components explain 32.15 % of the point variability.


Silhouette plot of pam(x = googleplaystore, k = 4)
n = 10836
4 clusters C_j
j : n_j | $ave_{i \in C_j}$ s_i
1 : 2362 | 0.31
2 : 2252 | 0.31
3 : 3057 | 0.27
4 : 3165 | 0.28
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.29


clusplot(pam(x = googleplaystore, k = 4))
Component 2
6
4
2
0
-2
-6
-4
-2
0
2
4
Component 1
These two components explain 32.15 % of the point variability.


Silhouette plot of pam(x = googleplaystore, k = 5)
n = 10836
5 clusters C_j
j : n_j | $ave_{i \in C_j}$ s_i
1 : 2332 | 0.28
2 : 1996 | 0.31
3 : 2213 | 0.29
4 : 2104 | 0.26
5 : 2191 | 0.35
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.3


clusplot(pam(x = googleplaystore, k = 5))
Component 2
6
4
2
0
-2
-6
-4
-2
0
2
4
Component 1
These two components explain 32.15 % of the point variability.



Output:

```
406 407 408 409 410 411 412 413 414 415 416 417 418 419 420
  1   2   1   2   2   1   1   2   2   1   1   1   1   2   1
421 422 423 424 425 426 427 428 429 430 431 432 433 434 435
  1   1   1   1   2   1   1   1   2   2   2   1   1   1   2
436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
  1   2   1   1   1   2   1   2   2   2   1   1   1   2   1
451 452 453 454 455 456 457 458 459 460 461 462 463 464 465
  2   1   1   1   1   2   2   1   1   1   1   2   1   1   1
466 467 468 469 470 471 472 473 474 475 476 477 478 479 480
  1   1   1   1   1   1   1   1   1   1   1   1   2   1   1
481 482 483 484 485 486 487 488 489 490 491 492 493 494 495
  2   2   1   1   1   1   1   2   2   1   2   1   2   2   2
496 497 498 499 500 501 502 503 504 505 506 507 508 509 510
  2   2   1   1   1   2   2   2   1   2   2   1   1   1   1
511 512 513 514 515 516 517 518 519 520 521 522 523 524 525
  2   2   1   1   2   2   1   2   1   1   2   1   1   1   1
526 527 528 529 530 531 532 533 534 535 536 537 538 539 540
  1   1   1   2   1   1   1   2   2   1   1   2   2   1   2
541 542 543 544 545 546 547 548 549 550 551 552 553 554 555
  1   2   1   2   1   1   1   2   2   2   2   1   1   1   1
556 557 558 559 560 561 562 563 564 565 566 567 568 569 570
  1   1   1   1   2   2   2   2   1   2   1   1   2   2   1
571 572 573 574 575 576 577 578 579 580 581 582 583 584 585
  2   1   1   1   2   2   2   1   1   1   2   1   1   1   1
586 587 588 589 590 591 592 593 594 595 596 597 598 599 600
  1   1   1   2   2   1   1   2   1   2   1   1   1   2   2
601 602 603 604 605 606 607 608 609 610 611 612 613 614 615
  1   1   1   1   2   2   1   1   2   1   1   1   2   1   1
616 617 618 619 620 621 622 623 624 625 626 627 628 629 630
  2   1   1   1   1   1   2   1   2   1   1   2   2   1   2
```

```
631 632 633 634 635 636 637 638 639 640 641 642 643 644 645
  1   2   1   1   1   1   1   1   1   1   1   1   1   1   1
646 647 648 649 650 651 652 653 654 655 656 657 658 659 660
  1   2   2   1   1   2   1   1   1   1   1   1   2   1   2
661 662 663 664 665 666 667 668 669 670 671 672 673 674 675
  2   1   2   1   1   1   1   2   1   2   1   2   2   1   2
676 677 678 679 680 681 682 683 684 685 686 687 688 689 690
  2   1   2   1   2   1   2   2   1   1   1   1   1   1   1
691 692 693 694 695 696 697 698 699 700 701 702 703 704 705
  1   1   2   2   1   1   2   1   2   2   1   2   1   1   2
706 707 708 709 710 711 712 713 714 715 716 717 718 719 720
  1   1   2   1   2   2   1   1   1   1   2   2   2   1   1
721 722 723 724 725 726 727 728 729 730 731 732 733 734 735
  1   1   2   1   1   1   1   2   1   2   1   2   1   1   1
736 737 738 739 740 741 742 743 744 745 746 747 748 749 750
  1   2   1   1   1   1   1   2   2   2   1   1   1   1   1
751 752 753 754 755 756 757 758 759 760 761 762 763 764 765
  2   2   1   2   2   2   1   1   1   1   1   1   1   1   1
766 767 768 769 770 771 772 773 774 775 776 777 778 779 780
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
781 782 783 784 785 786 787 788 789 790 791 792 793 794 795
  1   1   1   1   2   1   2   1   1   2   1   1   1   1   2
796 797 798 799 800 801 802 803 804 805 806 807 808 809 810
  1   2   1   2   2   1   1   2   1   1   1   1   1   2   1
811 812 813 814 815 816 817 818 819 820 821 822 823 824 825
  1   1   1   1   1   1   2   1   1   1   2   1   1   1   1
826 827 828 829 830 831 832 833 834 835 836 837 838 839 840
  2   2   1   1   1   1   1   1   2   1   2   1   2   2   1
841 842 843 844 845 846 847 848 849 850 851 852 853 854 855
  1   2   1   2   1   1   2   1   1   1   2   1   1   1   1
856 857 858 859 860 861 862 863 864 865 866 867 868 869 870
  1   2   1   1   1   1   1   2   1   1   1   1   1   1   1
871 872 873 874 875 876 877 878 879 880 881 882 883 884 885
  1   1   2   1   1   2   1   1   1   1   1   1   2   1   1
886 887 888 889 890 891 892 893 894 895 896 897 898 899 900
  2   2   2   1   1   1   1   1   1   1   1   2   1   1   2
901 902 903 904 905 906 907 908 909 910 911 912 913 914 915
  1   1   2   1   1   1   2   2   1   2   2   1   2   2   1
916 917 918 919 920 921 922 923 924 925 926 927 928 929 930
  2   1   1   1   1   2   1   2   1   1   1   1   1   1   2
931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
  1   1   1   2   1   2   2   1   1   1   1   2   1   1   1
946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
  1   2   1   1   1   1   2   1   1   1   1   2   1   1   1
961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
  1   2   2   1   1   1   1   1   1   1   1   1   1   1   1
976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
  1   1   2   1   1   1   1   2   2   2   1   1   1   1   1
991 992 993 994 995 996 997 998 999 1000
  1   2   1   2   1   1   1   1   1   1
 [ reached getOption("max.print") -- omitted 9836 entries ]
```

```
Within cluster sum of squares by cluster:
[1] 35675021689 33475326394
 (between_SS / total_SS =  47.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```
> ## visualize clustering k=2
> fviz_cluster(kmeans2.result, data = googleplaystore)
> set.seed(8953)
> kmeans3.result <- kmeans(googleplaystore, 3)
> kmeans3.result
K-means clustering with 3 clusters of sizes 3069, 4644, 3123
```

Cluster means:

```
      App Category   Rating  Reviews     Size  Installs     Type    Price
1 6867.025 18.59759 3.490974 1152.814 215.1854 10.190942 2.185728 87.33529
2 2148.195 18.59281 3.442442 2685.810 192.6906 10.423127 2.141473 88.20457
3 7078.711 19.04323 4.016971 4395.316 227.2856  9.988793 2.119757 88.63305
  Content.Rating   Genres Last.Updated Current.Ver Android.Ver
1       3.491365 65.31248     621.7908    1277.691    19.94037
2       3.399871 65.71835     652.2808    1195.169    18.77239
3       3.536663 66.64521     611.2389    1464.661    20.57797
```

Clustering vector:

```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15 
  1   2   3   1   3   1   1   3   1   1   1   3   3   3   2 
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 
  3   3   2   1   1   3   2   3   3   1   1   2   2   1   2 
 31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 
  1   2   2   2   3   3   3   3   1   3   2   1   3   1   1 
 46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 
  2   1   3   3   3   1   3   1   2   3   1   3   1   1   3 
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75 
  2   2   2   1   2   2   1   3   3   3   1   1   2   3   3 
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90 
  1   3   3   2   1   1   1   1   1   3   2   2   1   2   3 
 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 
  3   3   3   2   3   2   1   1   1   3   1   2   3   2   3 
106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 
  3   3   3   1   1   3   2   1   1   2   1   3   2   3   3 
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 
  2   3   1   1   1   2   3   2   3   3   3   2   2   2   3 
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 
  3   3   1   3   3   2   2   3   2   2   2   1   3   2   2 
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 
  2   1   1   2   3   3   3   3   2   2   1   3   1   2   2 
166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 
  3   3   2   2   2   1   2   2   3   2   1   1   2   1   1 
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 
  1   3   3   2   2   1   2   1   3   1   2   3   2   3   1 
```

196	197	198	199	200	201	202	203	204	205	206	207	208	209	210
1	1	2	3	2	3	1	3	3	2	3	2	1	3	3

211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
3	1	2	3	2	2	1	1	3	1	2	3	3	1	2

226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
1	1	3	3	3	3	3	1	3	1	1	2	3	1	3

241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
3	1	1	1	3	1	3	2	2	3	2	2	3	3	1

256	257	258	259	260	261	262	263	264	265	266	267	268	269	270
2	3	3	2	3	3	3	2	2	3	2	3	1	3	3

271	272	273	274	275	276	277	278	279	280	281	282	283	284	285
1	1	1	3	1	3	2	2	3	2	2	3	3	1	3

286	287	288	289	290	291	292	293	294	295	296	297	298	299	300
3	3	3	1	3	1	1	2	1	3	3	1	1	1	3

301	302	303	304	305	306	307	308	309	310	311	312	313	314	315
1	3	3	1	1	1	3	2	3	1	2	2	2	2	2

316	317	318	319	320	321	322	323	324	325	326	327	328	329	330
2	1	1	3	2	3	2	1	3	2	1	2	1	1	1

331	332	333	334	335	336	337	338	339	340	341	342	343	344	345
3	2	1	3	3	3	3	1	3	1	3	3	1	1	2

346	347	348	349	350	351	352	353	354	355	356	357	358	359	360
3	3	3	2	2	3	3	2	3	3	2	1	1	2	1

361	362	363	364	365	366	367	368	369	370	371	372	373	374	375
2	3	3	1	3	3	3	1	3	2	3	1	3	3	2

376	377	378	379	380	381	382	383	384	385	386	387	388	389	390
2	1	3	1	1	1	3	3	3	1	1	3	1	2	3

391	392	393	394	395	396	397	398	399	400	401	402	403	404	405
3	1	3	1	1	1	2	1	2	3	3	3	1	1	3

406	407	408	409	410	411	412	413	414	415	416	417	418	419	420
3	2	1	2	2	3	3	2	2	3	1	3	3	2	3

421	422	423	424	425	426	427	428	429	430	431	432	433	434	435
1	1	1	3	2	1	1	1	2	1	2	1	3	1	1

436	437	438	439	440	441	442	443	444	445	446	447	448	449	450
1	2	1	1	3	2	3	2	2	2	3	1	1	2	3

451	452	453	454	455	456	457	458	459	460	461	462	463	464	465
2	3	3	3	3	2	2	1	1	3	1	2	3	1	3

466	467	468	469	470	471	472	473	474	475	476	477	478	479	480
3	3	1	1	3	3	1	3	1	1	1	1	2	1	1

481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
2	2	3	1	3	1	3	2	2	3	2	3	2	1	2

496	497	498	499	500	501	502	503	504	505	506	507	508	509	510
2	2	3	3	1	2	1	1	1	2	2	1	3	3	1

511	512	513	514	515	516	517	518	519	520	521	522	523	524	525
2	2	3	1	1	2	1	2	3	1	2	3	1	1	3

526	527	528	529	530	531	532	533	534	535	536	537	538	539	540
3	1	1	2	3	3	1	2	2	1	3	2	2	3	2

541	542	543	544	545	546	547	548	549	550	551	552	553	554	555
3	1	1	2	1	3	3	2	2	2	2	3	3	3	3

556	557	558	559	560	561	562	563	564	565	566	567	568	569	570
3	1	1	3	2	2	2	1	1	2	1	3	2	2	3

571	572	573	574	575	576	577	578	579	580	581	582	583	584	585
2	1	1	1	2	2	2	1	1	3	2	1	1	1	3

586	587	588	589	590	591	592	593	594	595	596	597	598	599	600
3	1	1	2	2	3	1	2	3	1	3	3	3	2	2

601	602	603	604	605	606	607	608	609	610	611	612	613	614	615
1	3	3	3	3	2	1	1	2	1	3	1	2	3	3

616	617	618	619	620	621	622	623	624	625	626	627	628	629	630
2	1	1	3	1	1	2	1	1	1	1	2	2	1	2

631	632	633	634	635	636	637	638	639	640	641	642	643	644	645
1	1	1	3	1	1	3	1	1	1	3	1	1	1	1

646	647	648	649	650	651	652	653	654	655	656	657	658	659	660
1	2	2	1	1	1	1	1	1	3	1	1	1	1	2

661	662	663	664	665	666	667	668	669	670	671	672	673	674	675
2	3	2	3	1	3	3	2	1	2	3	2	2	3	2

676	677	678	679	680	681	682	683	684	685	686	687	688	689	690
2	3	2	1	1	1	2	3	3	3	1	3	1	3	3

691	692	693	694	695	696	697	698	699	700	701	702	703	704	705
3	3	1	2	3	1	1	3	2	2	1	2	3	3	2

706	707	708	709	710	711	712	713	714	715	716	717	718	719	720
3	3	2	1	3	3	1	1	1	3	2	2	2	3	3

721	722	723	724	725	726	727	728	729	730	731	732	733	734	735
1	3	2	3	3	3	3	3	3	2	1	3	1	1	1

736	737	738	739	740	741	742	743	744	745	746	747	748	749	750
3	2	1	3	1	1	3	2	2	2	1	1	1	3	3

751	752	753	754	755	756	757	758	759	760	761	762	763	764	765
2	2	3	2	2	2	1	3	3	1	1	3	3	3	1

```
766 767 768 769 770 771 772 773 774 775 776 777 778 779 780
  1   3   1   3   1   1   1   3   1   1   3   3   3   3   3
781 782 783 784 785 786 787 788 789 790 791 792 793 794 795
  1   1   3   1   2   1   2   3   1   2   1   3   3   1   2
796 797 798 799 800 801 802 803 804 805 806 807 808 809 810
  3   2   3   1   2   3   1   2   3   3   3   1   3   2   3
811 812 813 814 815 816 817 818 819 820 821 822 823 824 825
  3   3   1   1   1   3   2   1   3   1   2   1   3   1   3
826 827 828 829 830 831 832 833 834 835 836 837 838 839 840
  2   2   3   3   3   3   1   1   2   1   2   3   2   2   3
841 842 843 844 845 846 847 848 849 850 851 852 853 854 855
  3   2   3   2   3   1   2   1   1   1   2   1   1   3   3
856 857 858 859 860 861 862 863 864 865 866 867 868 869 870
  3   2   3   3   3   1   3   2   1   1   3   3   1   3   3
871 872 873 874 875 876 877 878 879 880 881 882 883 884 885
  1   1   2   1   1   2   3   3   3   3   3   1   2   3   1
886 887 888 889 890 891 892 893 894 895 896 897 898 899 900
  2   2   2   3   1   3   1   3   1   3   1   2   3   3   2
901 902 903 904 905 906 907 908 909 910 911 912 913 914 915
  1   1   2   3   3   3   2   2   1   2   2   1   2   2   3
916 917 918 919 920 921 922 923 924 925 926 927 928 929 930
  2   3   1   1   3   2   3   2   3   3   1   1   3   3   2
931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
  3   3   3   2   3   2   2   1   3   3   3   2   3   3   3
946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
  3   2   1   1   3   3   2   1   3   1   1   1   3   3   3
961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
  3   2   2   1   3   3   3   3   3   3   3   3   3   1   1
976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
  3   3   2   1   1   3   3   2   2   1   3   3   3   3   3
991 992 993 994 995 996 997 998 999 1000
  1   2   1   2   1   3   3   1   3   3
 [ reached getOption("max.print") -- omitted 9836 entries ]
Within cluster sum of squares by cluster:
[1] 13261402224 27056449783 13830733105
 (between_SS / total_SS =  59.1 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> plot(kmeans3.result, data = googleplaystore)
Error in xy.coords(x, y, xlabel, ylabel, log) :
  'x' is a list, but does not have components 'x' and 'y'
> ##---------------------
> ##Data mining task
> #----------------------
> googleplaystore$Category <- sapply(googleplaystore$Category,as.numeric)
> googleplaystore$Reviews <- sapply(googleplaystore$Reviews,as.numeric)

> googleplaystore$Size <- sapply(googleplaystore$Size,as.numeric)

> googleplaystore$Installs <- sapply(googleplaystore$Installs,as.numeric)

> googleplaystore$Type <- sapply(googleplaystore$Type,as.numeric)

> googleplaystore$Price <- sapply(googleplaystore$Price,as.numeric)

> googleplaystore$Content.Rating <- sapply(googleplaystore$Content.Rating,as.numeric)

> googleplaystore$Genres <- sapply(googleplaystore$Genres,as.numeric)

> googleplaystore$Last.Updated <- sapply(googleplaystore$Last.Updated,as.numeric)

> googleplaystore$Current.Ver<- sapply(googleplaystore$Current.Ver,as.numeric)

> googleplaystore$Android.Ver<- sapply(googleplaystore$Android.Ver,as.numeric)

> googleplaystore$App<- sapply(googleplaystore$App,as.numeric)

> #------------------

> kmeans2.result <- kmeans(googleplaystore,2)

> kmeans2.result

K-means clustering with 2 clusters of sizes 5601, 5235


Cluster means:

    App Category   Rating  Reviews     Size Installs     Type    Price

1 7237.547 18.80950 3.830673 2858.093 225.4133 10.06017 2.152473 87.99125

2 2410.779 18.63247 3.398262 2622.594 191.5056 10.41624 2.142693 88.17880

  Content.Rating   Genres Last.Updated Current.Ver Android.Ver

1      3.526335 65.79147     615.4306    1393.747    20.37475

2      3.399809 65.95511     649.3486    1191.854    18.81987


Clustering vector:


1   2   3   4   5   6   7   8   9  10  11  12  13  14  15

 1   2   1   1   1   1   1   1   1   1   1   1   1   1   2

 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30

 1   1   2   2   1   1   2   1   1   1   1   2   2   1   2

31  32  33  34  35  36  37  38  39  40  41  42  43  44  45

 1   2   2   2   1   1   1   2   1   1   2   1   1   1   1

46  47  48  49  50  51  52  53  54  55  56  57  58  59  60

 2   1   1   1   1   1   1   1   2   1   1   1   2   1   1
```

61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
2	2	2	2	2	2	1	1	1	1	2	1	2	1	1

76	77	78	79	80	81	82	83	84	85	86	87	88	89	90
1	1	1	2	1	1	1	1	1	1	2	2	1	2	1

91	92	93	94	95	96	97	98	99	100	101	102	103	104	105
1	1	1	2	1	2	1	1	1	1	1	2	1	2	1

106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
2	1	1	1	1	1	2	1	1	2	1	1	2	1	1

121	122	123	124	125	126	127	128	129	130	131	132	133	134	135
2	1	1	1	1	2	1	2	1	1	1	2	2	2	1

136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
1	1	1	1	1	2	2	1	2	2	2	1	1	2	2

151	152	153	154	155	156	157	158	159	160	161	162	163	164	165
2	2	1	2	1	1	1	1	2	2	1	1	1	2	2

166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
1	1	2	2	2	2	2	2	1	2	1	2	2	1	1

181	182	183	184	185	186	187	188	189	190	191	192	193	194	195
1	2	1	2	2	1	2	1	1	1	2	1	2	1	1

196	197	198	199	200	201	202	203	204	205	206	207	208	209	210
1	1	2	1	2	1	2	1	1	2	1	2	1	1	1

211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
1	1	2	1	2	2	1	1	1	1	2	1	1	1	2

226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
1	1	1	1	1	1	1	2	1	1	1	2	1	1	1

241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
1	2	1	1	1	1	1	2	2	1	2	2	1	1	1

256	257	258	259	260	261	262	263	264	265	266	267	268	269	270
2	1	1	2	1	1	1	2	2	1	2	1	1	1	1

271	272	273	274	275	276	277	278	279	280	281	282	283	284	285
2	1	1	1	1	1	2	2	1	2	2	1	1	1	1

286	287	288	289	290	291	292	293	294	295	296	297	298	299	300
1	1	1	2	1	1	1	2	1	1	1	1	1	1	1

301	302	303	304	305	306	307	308	309	310	311	312	313	314	315
1	1	1	1	1	1	1	2	1	1	2	2	2	2	2

316	317	318	319	320	321	322	323	324	325	326	327	328	329	330
2	1	1	1	2	1	2	2	1	2	1	2	1	2	1

331	332	333	334	335	336	337	338	339	340	341	342	343	344	345
1	2	1	1	1	1	1	1	1	1	1	1	1	1	2

346	347	348	349	350	351	352	353	354	355	356	357	358	359	360
1	1	1	2	2	1	1	2	1	1	2	1	1	2	1

361	362	363	364	365	366	367	368	369	370	371	372	373	374	375
2	1	1	1	1	1	1	1	1	2	1	1	1	1	2

376	377	378	379	380	381	382	383	384	385	386	387	388	389	390
2	1	1	1	1	1	1	1	1	1	1	1	1	2	1

391	392	393	394	395	396	397	398	399	400	401	402	403	404	405
1	1	1	1	1	1	2	1	2	1	1	1	1	1	1

406	407	408	409	410	411	412	413	414	415	416	417	418	419	420
1	2	1	2	2	1	1	2	2	1	1	1	1	2	1

421	422	423	424	425	426	427	428	429	430	431	432	433	434	435
1	1	1	1	2	1	1	1	2	2	2	1	1	1	2

436	437	438	439	440	441	442	443	444	445	446	447	448	449	450
1	2	1	1	1	2	1	2	2	2	1	1	1	2	1

451	452	453	454	455	456	457	458	459	460	461	462	463	464	465
2	1	1	1	1	2	2	1	1	1	1	2	1	1	1

466	467	468	469	470	471	472	473	474	475	476	477	478	479	480
1	1	1	1	1	1	1	1	1	1	1	1	2	1	1

481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
2	2	1	1	1	1	1	2	2	1	2	1	2	2	2

496	497	498	499	500	501	502	503	504	505	506	507	508	509	510
2	2	1	1	1	2	2	2	1	2	2	1	1	1	1

511	512	513	514	515	516	517	518	519	520	521	522	523	524	525
2	2	1	1	2	2	1	2	1	1	2	1	1	1	1

526	527	528	529	530	531	532	533	534	535	536	537	538	539	540
1	1	1	2	1	1	1	2	2	1	1	2	2	1	2

541	542	543	544	545	546	547	548	549	550	551	552	553	554	555
1	2	1	2	1	1	1	2	2	2	2	1	1	1	1

556	557	558	559	560	561	562	563	564	565	566	567	568	569	570
1	1	1	1	2	2	2	2	1	2	1	1	2	2	1

571	572	573	574	575	576	577	578	579	580	581	582	583	584	585
2	1	1	1	2	2	2	1	1	1	2	1	1	1	1

586	587	588	589	590	591	592	593	594	595	596	597	598	599	600
1	1	1	2	2	1	1	2	1	2	1	1	1	2	2

601	602	603	604	605	606	607	608	609	610	611	612	613	614	615
1	1	1	1	2	2	1	1	2	1	1	1	2	1	1

616	617	618	619	620	621	622	623	624	625	626	627	628	629	630
2	1	1	1	1	1	2	1	2	1	1	2	2	1	2

```
631 632 633 634 635 636 637 638 639 640 641 642 643 644 645
  1   2   1   1   1   1   1   1   1   1   1   1   1   1   1
646 647 648 649 650 651 652 653 654 655 656 657 658 659 660
  1   2   2   1   1   2   1   1   1   1   1   1   2   1   2
661 662 663 664 665 666 667 668 669 670 671 672 673 674 675
  2   1   2   1   1   1   1   2   1   2   1   2   2   1   2
676 677 678 679 680 681 682 683 684 685 686 687 688 689 690
  2   1   2   1   2   1   2   2   1   1   1   1   1   1   1
691 692 693 694 695 696 697 698 699 700 701 702 703 704 705
  1   1   2   2   1   1   2   1   2   2   1   2   1   1   2
706 707 708 709 710 711 712 713 714 715 716 717 718 719 720
  1   1   2   1   2   2   1   1   1   1   2   2   2   1   1
721 722 723 724 725 726 727 728 729 730 731 732 733 734 735
  1   1   2   1   1   1   1   2   1   2   1   2   1   1   1
736 737 738 739 740 741 742 743 744 745 746 747 748 749 750
  1   2   1   1   1   1   1   2   2   2   1   1   1   1   1
751 752 753 754 755 756 757 758 759 760 761 762 763 764 765
  2   2   1   2   2   2   1   1   1   1   1   1   1   1   1
766 767 768 769 770 771 772 773 774 775 776 777 778 779 780
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
781 782 783 784 785 786 787 788 789 790 791 792 793 794 795
  1   1   1   1   2   1   2   1   1   2   1   1   1   1   2
796 797 798 799 800 801 802 803 804 805 806 807 808 809 810
  1   2   1   2   2   1   1   2   1   1   1   1   1   2   1
811 812 813 814 815 816 817 818 819 820 821 822 823 824 825
  1   1   1   1   1   1   2   1   1   1   2   1   1   1   1
826 827 828 829 830 831 832 833 834 835 836 837 838 839 840
  2   2   1   1   1   1   1   1   2   1   2   1   2   2   1
841 842 843 844 845 846 847 848 849 850 851 852 853 854 855
  1   2   1   2   1   1   2   1   1   1   2   1   1   1   1
856 857 858 859 860 861 862 863 864 865 866 867 868 869 870
  1   2   1   1   1   1   1   2   1   1   1   1   1   1   1
871 872 873 874 875 876 877 878 879 880 881 882 883 884 885
  1   1   2   1   1   2   1   1   1   1   1   1   2   1   1
886 887 888 889 890 891 892 893 894 895 896 897 898 899 900
  2   2   2   1   1   1   1   1   1   1   1   2   1   1   2
901 902 903 904 905 906 907 908 909 910 911 912 913 914 915
  1   1   2   1   1   1   2   2   1   2   2   1   2   2   1
916 917 918 919 920 921 922 923 924 925 926 927 928 929 930
  2   1   1   1   1   2   1   2   1   1   1   1   1   1   2
931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
  1   1   1   2   1   2   2   1   1   1   1   2   1   1   1
946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
  1   2   1   1   1   1   2   1   1   1   1   2   1   1   1
961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
  1   2   2   1   1   1   1   1   1   1   1   1   1   1   1
976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
  1   1   2   1   1   1   1   2   2   2   1   1   1   1   1
991 992 993 994 995 996 997 998 999 1000
  1   2   1   2   1   1   1   1   1   1
 [ reached getOption("max.print") -- omitted 9836 entries ]
```

Within cluster sum of squares by cluster:

```
[1] 35675021689 33475326394
 (between_SS / total_SS =  47.8 %)
```

Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```
> ## visualize clustering k=2
> fviz_cluster(kmeans2.result, data = googleplaystore)
> set.seed(8953)
> kmeans3.result <- kmeans(googleplaystore, 3)
> kmeans3.result
K-means clustering with 3 clusters of sizes 3069, 4644, 3123

Cluster means:
      App Category   Rating  Reviews     Size  Installs     Type    Price
1 6867.025 18.59759 3.490974 1152.814 215.1854 10.190942 2.185728 87.33529
2 2148.195 18.59281 3.442442 2685.810 192.6906 10.423127 2.141473 88.20457
3 7078.711 19.04323 4.016971 4395.316 227.2856  9.988793 2.119757 88.63305
  Content.Rating   Genres Last.Updated Current.Ver Android.Ver
1       3.491365 65.31248     621.7908    1277.691    19.94037
2       3.399871 65.71835     652.2808    1195.169    18.77239
3       3.536663 66.64521     611.2389    1464.661    20.57797

Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
  1   2   3   1   3   1   1   3   1   1   1   3   3   3   2
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
  3   3   2   1   1   3   2   3   3   1   1   2   2   1   2
 31  32  33  34  35  36  37  38  39  40  41  42  43  44  45
  1   2   2   2   3   3   3   3   1   3   2   1   3   1   1
 46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
  2   1   3   3   3   1   3   1   2   3   1   3   1   1   3
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75
  2   2   2   1   2   2   1   3   3   3   1   1   2   3   3
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  1   3   3   2   1   1   1   1   1   3   2   2   1   2   3
 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105
  3   3   3   2   3   2   1   1   1   3   1   2   3   2   3
106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
  3   3   3   1   1   3   2   1   1   2   1   3   2   3   3
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135
  2   3   1   1   1   2   3   2   3   3   3   2   2   2   3
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
  3   3   1   3   3   2   2   3   2   2   2   1   3   2   2
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165
  2   1   1   2   3   3   3   3   2   2   1   3   1   2   2
166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
  3   3   2   2   2   1   2   2   3   2   1   1   2   1   1
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195
  1   3   3   2   2   1   2   1   3   1   2   3   2   3   1
196 197 198 199 200 201 202 203 204 205 206 207 208 209 210
  1   1   2   3   2   3   1   3   3   2   3   2   1   3   3
```

211 212 213 214 215 216 217 218 219 220 221 222 223 224 225

3 1 2 3 2 2 1 1 3 1 2 3 3 1 2

226 227 228 229 230 231 232 233 234 235 236 237 238 239 240

1 1 3 3 3 3 3 1 3 1 1 2 3 1 3

241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

3 1 1 1 3 1 3 2 2 3 2 2 3 3 1

256 257 258 259 260 261 262 263 264 265 266 267 268 269 270

2 3 3 2 3 3 3 2 2 3 2 3 1 3 3

271 272 273 274 275 276 277 278 279 280 281 282 283 284 285

1 1 1 3 1 3 2 2 3 2 2 3 3 1 3

286 287 288 289 290 291 292 293 294 295 296 297 298 299 300

3 3 3 1 3 1 1 2 1 3 3 1 1 1 3

301 302 303 304 305 306 307 308 309 310 311 312 313 314 315

1 3 3 1 1 1 3 2 3 1 2 2 2 2 2

316 317 318 319 320 321 322 323 324 325 326 327 328 329 330

2 1 1 3 2 3 2 1 3 2 1 2 1 1 1

331 332 333 334 335 336 337 338 339 340 341 342 343 344 345

3 2 1 3 3 3 3 1 3 1 3 3 1 1 2

346 347 348 349 350 351 352 353 354 355 356 357 358 359 360

3 3 3 2 2 3 3 2 3 3 2 1 1 2 1

361 362 363 364 365 366 367 368 369 370 371 372 373 374 375

2 3 3 1 3 3 3 1 3 2 3 1 3 3 2

376 377 378 379 380 381 382 383 384 385 386 387 388 389 390

2 1 3 1 1 1 3 3 3 1 1 3 1 2 3

391 392 393 394 395 396 397 398 399 400 401 402 403 404 405

3 1 3 1 1 1 2 1 2 3 3 3 1 1 3

406 407 408 409 410 411 412 413 414 415 416 417 418 419 420

3 2 1 2 2 3 3 2 2 3 1 3 3 2 3

421 422 423 424 425 426 427 428 429 430 431 432 433 434 435

1 1 1 3 2 1 1 1 2 1 2 1 3 1 1

436 437 438 439 440 441 442 443 444 445 446 447 448 449 450

1 2 1 1 3 2 3 2 2 2 3 1 1 2 3

451 452 453 454 455 456 457 458 459 460 461 462 463 464 465

2 3 3 3 3 2 2 1 1 3 1 2 3 1 3

466 467 468 469 470 471 472 473 474 475 476 477 478 479 480

3 3 1 1 3 3 1 3 1 1 1 1 2 1 1

481 482 483 484 485 486 487 488 489 490 491 492 493 494 495

2 2 3 1 3 1 3 2 2 3 2 3 2 1 2

496 497 498 499 500 501 502 503 504 505 506 507 508 509 510

2 2 3 3 1 2 1 1 1 2 2 1 3 3 1

511 512 513 514 515 516 517 518 519 520 521 522 523 524 525

2 2 3 1 1 2 1 2 3 1 2 3 1 1 3

526 527 528 529 530 531 532 533 534 535 536 537 538 539 540

3 1 1 2 3 3 1 2 2 1 3 2 2 3 2

541 542 543 544 545 546 547 548 549 550 551 552 553 554 555

3 1 1 2 1 3 3 2 2 2 2 3 3 3 3

556 557 558 559 560 561 562 563 564 565 566 567 568 569 570

3 1 1 3 2 2 2 1 1 2 1 3 2 2 3

571 572 573 574 575 576 577 578 579 580 581 582 583 584 585

2 1 1 1 2 2 2 1 1 3 2 1 1 1 3

586 587 588 589 590 591 592 593 594 595 596 597 598 599 600

3 1 1 2 2 3 1 2 3 1 3 3 3 2 2

601 602 603 604 605 606 607 608 609 610 611 612 613 614 615

1 3 3 3 3 2 1 1 2 1 3 1 2 3 3

616 617 618 619 620 621 622 623 624 625 626 627 628 629 630

2 1 1 3 1 1 2 1 1 1 1 2 2 1 2

631 632 633 634 635 636 637 638 639 640 641 642 643 644 645

1 1 1 3 1 1 3 1 1 1 3 1 1 1 1

646 647 648 649 650 651 652 653 654 655 656 657 658 659 660

1 2 2 1 1 1 1 1 1 3 1 1 1 1 2

661 662 663 664 665 666 667 668 669 670 671 672 673 674 675

2 3 2 3 1 3 3 2 1 2 3 2 2 3 2

676 677 678 679 680 681 682 683 684 685 686 687 688 689 690

2 3 2 1 1 1 2 3 3 3 1 3 1 3 3

691 692 693 694 695 696 697 698 699 700 701 702 703 704 705

3 3 1 2 3 1 1 3 2 2 1 2 3 3 2

706 707 708 709 710 711 712 713 714 715 716 717 718 719 720

3 3 2 1 3 3 1 1 1 3 2 2 2 3 3

721 722 723 724 725 726 727 728 729 730 731 732 733 734 735

1 3 2 3 3 3 3 3 3 2 1 3 1 1 1

736 737 738 739 740 741 742 743 744 745 746 747 748 749 750

3 2 1 3 1 1 3 2 2 2 1 1 1 3 3

751 752 753 754 755 756 757 758 759 760 761 762 763 764 765

2 2 3 2 2 2 1 3 3 1 1 3 3 3 1

766 767 768 769 770 771 772 773 774 775 776 777 778 779 780

1 3 1 3 1 1 1 3 1 1 3 3 3 3 3

```
781 782 783 784 785 786 787 788 789 790 791 792 793 794 795
  1   1   3   1   2   1   2   3   1   2   1   3   3   1   2
796 797 798 799 800 801 802 803 804 805 806 807 808 809 810
  3   2   3   1   2   3   1   2   3   3   3   1   3   2   3
811 812 813 814 815 816 817 818 819 820 821 822 823 824 825
  3   3   1   1   1   3   2   1   3   1   2   1   3   1   3
826 827 828 829 830 831 832 833 834 835 836 837 838 839 840
  2   2   3   3   3   3   1   1   2   1   2   3   2   2   3
841 842 843 844 845 846 847 848 849 850 851 852 853 854 855
  3   2   3   2   3   1   2   1   1   1   2   1   1   3   3
856 857 858 859 860 861 862 863 864 865 866 867 868 869 870
  3   2   3   3   3   1   3   2   1   1   3   3   1   3   3
871 872 873 874 875 876 877 878 879 880 881 882 883 884 885
  1   1   2   1   1   2   3   3   3   3   3   1   2   3   1
886 887 888 889 890 891 892 893 894 895 896 897 898 899 900
  2   2   2   3   1   3   1   3   1   3   1   2   3   3   2
901 902 903 904 905 906 907 908 909 910 911 912 913 914 915
  1   1   2   3   3   3   2   2   1   2   2   1   2   2   3
916 917 918 919 920 921 922 923 924 925 926 927 928 929 930
  2   3   1   1   3   2   3   2   3   3   1   1   3   3   2
931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
  3   3   3   2   3   2   2   1   3   3   3   2   3   3   3
946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
  3   2   1   1   3   3   2   1   3   1   1   1   3   3   3
961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
  3   2   2   1   3   3   3   3   3   3   3   3   3   1   1
976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
  3   3   2   1   1   3   3   2   2   1   3   3   3   3   3
991 992 993 994 995 996 997 998 999 1000
  1   2   1   2   1   3   3   1   3   3
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 13261402224 27056449783 13830733105
 (between_SS / total_SS =  59.1 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> ## visualize clustering k=3
> fviz_cluster(kmeans3.result, data = googleplaystore)
> kmeans4.result <- kmeans(googleplaystore, 4)
> kmeans4.result
K-means clustering with 4 clusters of sizes 2841, 2747, 2497, 2751

Cluster means:
      App Category   Rating  Reviews     Size Installs     Type    Price
1 7097.168 18.98381 4.022527 4409.852 228.8508  9.94333 2.119676 88.72826
2 2554.132 18.44266 3.062723 1071.518 185.2756 10.36767 2.147434 88.01274
3 2264.517 18.82739 3.767121 4294.469 197.3444 10.48018 2.140969 88.26712
4 7387.924 18.64268 3.634206 1287.648 222.8957 10.17012 2.183206 87.31516
  Content.Rating   Genres Last.Updated Current.Ver Android.Ver
1       3.532207 66.35903     607.2925    1473.073    20.62443
2       3.397161 65.54132     654.5497    1129.056    18.74081
3       3.400881 66.37205     641.9828    1257.099    18.86304
4       3.522356 65.23955     625.2163    1315.974    20.16176
```

Clustering vector:

```
1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
  4   3   1   4   1   4   4   1   2   4   4   1   1   1   2
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
  1   1   3   2   4   1   3   1   1   4   4   2   2   4   2
 31  32  33  34  35  36  37  38  39  40  41  42  43  44  45
  4   2   3   2   1   1   1   1   4   1   2   4   4   4   4
 46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
  2   4   1   1   1   4   4   4   2   1   4   1   2   4   1
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75
  3   3   3   2   3   3   4   1   1   1   2   4   3   1   1
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  4   1   1   3   4   4   4   4   4   1   2   3   4   2   1
 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105
  1   1   1   3   1   2   4   2   4   1   4   2   1   2   1
106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
  3   1   1   4   4   1   3   4   4   3   4   1   2   1   1
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135
  3   1   4   4   4   2   1   2   1   1   1   3   2   3   1
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
  1   1   4   1   4   2   3   1   3   2   3   4   1   2   2
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165
  2   2   4   3   1   1   1   1   2   2   4   1   4   2   3
166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
  1   1   3   2   2   2   3   3   1   3   4   2   2   4   4
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195
  4   3   1   3   2   4   3   4   1   4   3   1   2   1   4
196 197 198 199 200 201 202 203 204 205 206 207 208 209 210
  4   4   3   1   3   1   2   1   1   2   1   2   4   1   1
211 212 213 214 215 216 217 218 219 220 221 222 223 224 225
  1   4   3   4   2   2   4   4   1   4   3   1   1   4   3
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
  4   4   1   1   1   1   4   2   1   4   4   2   4   1   1
241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
  1   2   4   4   1   4   1   3   2   1   3   3   1   1   4
256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
  2   4   1   2   1   4   1   3   2   1   2   4   1   1   1
271 272 273 274 275 276 277 278 279 280 281 282 283 284 285
  2   4   4   1   4   1   3   2   1   3   3   1   1   4   1
286 287 288 289 290 291 292 293 294 295 296 297 298 299 300
  1   1   4   2   1   4   4   2   4   1   1   4   4   4   1
301 302 303 304 305 306 307 308 309 310 311 312 313 314 315
  4   1   1   4   4   1   1   3   1   4   2   3   2   2   3
```

316	317	318	319	320	321	322	323	324	325	326	327	328	329	330
2	4	4	4	2	1	2	2	1	2	4	3	4	2	4
331	332	333	334	335	336	337	338	339	340	341	342	343	344	345
1	3	4	1	1	1	1	4	1	4	1	1	4	4	3
346	347	348	349	350	351	352	353	354	355	356	357	358	359	360
1	1	1	3	3	1	1	2	1	1	2	4	4	3	4
361	362	363	364	365	366	367	368	369	370	371	372	373	374	375
3	1	1	4	1	1	1	4	1	3	1	4	1	1	3
376	377	378	379	380	381	382	383	384	385	386	387	388	389	390
2	4	1	4	4	4	1	1	1	4	4	1	4	3	1
391	392	393	394	395	396	397	398	399	400	401	402	403	404	405
1	4	1	4	4	1	3	4	3	1	1	1	4	4	1
406	407	408	409	410	411	412	413	414	415	416	417	418	419	420
1	2	4	3	2	1	1	3	2	1	4	1	1	2	1
421	422	423	424	425	426	427	428	429	430	431	432	433	434	435
4	2	4	1	2	4	4	4	2	2	3	4	1	4	2
436	437	438	439	440	441	442	443	444	445	446	447	448	449	450
4	2	4	4	1	3	1	3	3	3	1	4	4	2	1
451	452	453	454	455	456	457	458	459	460	461	462	463	464	465
3	1	1	1	1	2	2	4	4	1	4	3	1	4	1
466	467	468	469	470	471	472	473	474	475	476	477	478	479	480
1	1	4	4	1	4	4	1	4	4	4	4	3	4	4
481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
3	2	1	4	1	4	1	3	2	1	3	1	3	2	3
496	497	498	499	500	501	502	503	504	505	506	507	508	509	510
2	2	1	1	4	3	2	2	4	2	2	4	1	1	4
511	512	513	514	515	516	517	518	519	520	521	522	523	524	525
2	2	1	4	2	2	4	3	1	4	3	1	4	4	1
526	527	528	529	530	531	532	533	534	535	536	537	538	539	540
1	4	4	3	1	1	4	2	2	4	1	2	3	1	3
541	542	543	544	545	546	547	548	549	550	551	552	553	554	555
1	2	4	3	4	1	1	2	3	3	2	1	1	1	1
556	557	558	559	560	561	562	563	564	565	566	567	568	569	570
1	4	4	1	3	3	3	2	4	2	4	1	3	2	1
571	572	573	574	575	576	577	578	579	580	581	582	583	584	585
3	4	4	4	2	2	3	4	4	1	3	4	4	4	1
586	587	588	589	590	591	592	593	594	595	596	597	598	599	600
1	4	4	3	2	1	4	2	1	2	1	1	1	2	2

601	602	603	604	605	606	607	608	609	610	611	612	613	614	615
4	1	1	1	1	3	4	4	3	4	1	1	3	4	1
616	617	618	619	620	621	622	623	624	625	626	627	628	629	630
2	4	4	1	4	4	2	4	2	4	4	2	2	4	2
631	632	633	634	635	636	637	638	639	640	641	642	643	644	645
4	2	4	1	4	4	1	4	4	4	1	4	4	4	4
646	647	648	649	650	651	652	653	654	655	656	657	658	659	660
4	2	3	4	4	2	4	4	4	4	4	4	2	4	2
661	662	663	664	665	666	667	668	669	670	671	672	673	674	675
2	1	3	1	4	1	1	2	4	2	1	3	3	1	2
676	677	678	679	680	681	682	683	684	685	686	687	688	689	690
3	1	2	1	2	4	3	1	4	1	4	1	4	1	1
691	692	693	694	695	696	697	698	699	700	701	702	703	704	705
1	1	2	2	1	4	2	1	3	3	4	2	1	1	3
706	707	708	709	710	711	712	713	714	715	716	717	718	719	720
1	1	3	4	3	3	4	4	4	1	3	2	2	1	1
721	722	723	724	725	726	727	728	729	730	731	732	733	734	735
4	1	2	1	4	1	1	3	1	3	4	3	4	4	4
736	737	738	739	740	741	742	743	744	745	746	747	748	749	750
1	2	4	1	4	4	1	3	3	2	4	4	4	1	1
751	752	753	754	755	756	757	758	759	760	761	762	763	764	765
3	3	4	3	2	2	4	1	1	4	4	1	1	1	4
766	767	768	769	770	771	772	773	774	775	776	777	778	779	780
4	1	4	1	4	4	4	1	4	4	1	1	1	1	1
781	782	783	784	785	786	787	788	789	790	791	792	793	794	795
4	4	1	4	3	4	3	1	4	3	4	1	1	4	3
796	797	798	799	800	801	802	803	804	805	806	807	808	809	810
1	2	1	2	3	1	4	3	1	1	1	4	1	2	1
811	812	813	814	815	816	817	818	819	820	821	822	823	824	825
1	1	4	4	4	1	3	4	1	4	2	4	1	4	1
826	827	828	829	830	831	832	833	834	835	836	837	838	839	840
2	3	1	1	1	1	4	4	2	4	2	1	2	3	1
841	842	843	844	845	846	847	848	849	850	851	852	853	854	855
1	2	1	2	1	4	2	4	4	4	2	4	2	4	1
856	857	858	859	860	861	862	863	864	865	866	867	868	869	870
1	2	4	4	1	4	1	3	4	4	1	1	4	1	1
871	872	873	874	875	876	877	878	879	880	881	882	883	884	885
4	4	3	4	4	3	1	1	1	1	1	4	2	1	4

```
886  887  888  889  890  891  892  893  894  895  896  897  898  899  900
  3    3    3    1    4    4    4    1    4    1    4    2    1    1    2
901  902  903  904  905  906  907  908  909  910  911  912  913  914  915
  4    4    3    1    1    1    2    2    4    2    3    4    3    3    1
916  917  918  919  920  921  922  923  924  925  926  927  928  929  930
  3    1    4    4    1    3    4    2    1    1    4    4    1    1    2
931  932  933  934  935  936  937  938  939  940  941  942  943  944  945
  1    1    4    3    1    3    3    4    1    1    4    2    1    1    1
946  947  948  949  950  951  952  953  954  955  956  957  958  959  960
  1    3    4    4    1    1    2    4    1    4    4    2    1    1    4
961  962  963  964  965  966  967  968  969  970  971  972  973  974  975
  1    3    3    4    1    1    1    1    1    4    1    1    4    4    4
976  977  978  979  980  981  982  983  984  985  986  987  988  989  990
  1    1    3    4    4    1    1    2    2    2    1    1    4    1    1
991  992  993  994  995  996  997  998  999 1000
  4    3    4    2    4    1    1    4    1    1
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 11453483107 10557233161  9307249817 10354205077
 (between_SS / total_SS =  68.5 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> ## visualize clustering k=4
> fviz_cluster(kmeans4.result, data = googleplaystore)
> kmeans5.result <- kmeans(googleplaystore, 5)
> kmeans5.result
K-means clustering with 5 clusters of sizes 2508, 1934, 2224, 2196, 1974

Cluster means:
      App Category   Rating  Reviews     Size  Installs      Type    Price
1 7260.772 19.02153 4.036722 4543.264 227.2791  9.940191 2.117225 88.76874
2 1555.544 18.55377 3.116960 1215.774 188.3283 10.394519 2.153568 87.72854
3 4850.503 18.40558 3.384442 1528.921 204.2963 10.265737 2.157374 88.12770
4 2319.006 18.75729 3.809107 4518.211 199.0437 10.499545 2.134791 88.35291
5 8135.481 18.83435 3.648126 1352.247 222.5805 10.108916 2.184397 87.20213
```

```
  Content.Rating   Genres Last.Updated Current.Ver Android.Ver
1       3.522727 66.28230     607.4725    1476.742    20.58413
2       3.385212 64.86194     662.0936    1199.965    18.90228
3       3.495504 65.44829     629.5544    1187.478    19.28552
4       3.413934 66.15710     644.0301    1250.067    18.74362
5       3.493414 66.49240     622.0456    1334.971    20.46960
```

Clustering vector:

```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15 
  5   4   1   5   1   5   5   1   3   3   5   1   1   1   2 
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 
  1   1   2   3   3   1   4   1   1   3   3   2   2   5   3 
 31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 
  5   2   4   3   3   1   1   3   5   1   3   5   5   5   5 
 46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 
  2   3   1   1   1   5   5   5   2   1   5   1   3   5   1 
 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75 
  4   4   3   3   4   2   5   1   1   1   3   5   2   1   1 
 76  77  78  79  80  81  82  83  84  85  86  87  88  89  90 
  5   1   1   4   5   5   5   5   5   1   2   4   5   2   1 
 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 
  1   1   1   2   1   2   3   3   3   1   5   2   1   2   1 
106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 
  4   1   1   5   5   1   4   5   5   3   5   1   2   3   1 
121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 
  4   3   5   3   5   3   1   3   1   1   1   4   3   4   3 
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 
  1   1   3   1   5   3   4   1   4   2   4   5   3   2   3 
151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 
  3   3   3   4   1   1   1   1   3   2   5   1   5   2   4 
166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 
  1   1   4   3   3   3   2   4   1   4   5   3   3   5   5 
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 
  5   4   1   2   2   5   3   5   1   5   4   1   3   1   5 
196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 
  5   3   4   1   4   1   3   1   1   2   1   2   3   1   1 
211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 
  1   5   4   5   3   3   3   5   1   3   4   1   1   5   4 
226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 
  5   5   1   1   1   1   5   3   1   5   5   2   5   3   1 
241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 
  1   3   3   5   1   5   1   4   2   1   4   2   1   1   3 
256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 
  2   5   1   2   1   5   1   4   2   1   2   5   3   1   1 
271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 
  3   3   5   1   5   1   4   2   1   4   2   1   1   3   1 
286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 
  1   1   5   3   1   5   5   3   5   1   1   5   3   3   1 
301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 
  5   1   1   3   3   3   1   4   3   5   2   4   2   2   3 
316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 
  2   3   3   5   2   1   2   3   1   2   3   4   5   3   3 
331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 
  1   4   5   1   1   1   1   3   1   3   1   3   5   5   3 
346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 
  1   1   1   4   4   5   1   2   1   1   2   5   5   3   3 
361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 
  4   1   1   5   1   1   1   5   1   4   5   3   1   1   2 
376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 
  2   3   1   5   5   3   1   1   1   3   5   3   5   4   1 
391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 
  1   5   5   5   3   3   4   3   4   3   1   3   5   3   3 
```

406	407	408	409	410	411	412	413	414	415	416	417	418	419	420
1	2	3	3	2	3	1	3	2	1	5	1	1	3	1

421	422	423	424	425	426	427	428	429	430	431	432	433	434	435
5	3	3	1	3	3	5	5	2	3	4	5	1	5	3

436	437	438	439	440	441	442	443	444	445	446	447	448	449	450
3	2	5	5	1	4	1	4	4	4	5	5	5	2	1

451	452	453	454	455	456	457	458	459	460	461	462	463	464	465
4	1	1	1	1	3	2	5	3	1	3	4	1	5	3

466	467	468	469	470	471	472	473	474	475	476	477	478	479	480
1	1	5	5	1	5	3	3	5	3	3	5	4	5	5

481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
4	2	1	5	1	3	1	4	2	1	4	3	3	3	4

496	497	498	499	500	501	502	503	504	505	506	507	508	509	510
2	2	1	4	5	2	3	3	5	2	2	3	1	3	5

511	512	513	514	515	516	517	518	519	520	521	522	523	524	525
2	3	4	5	3	2	5	4	1	5	4	1	3	5	3

526	527	528	529	530	531	532	533	534	535	536	537	538	539	540
1	5	5	4	4	1	5	2	2	3	1	2	4	1	4

541	542	543	544	545	546	547	548	549	550	551	552	553	554	555
1	3	5	4	3	1	1	2	4	4	2	1	1	3	1

556	557	558	559	560	561	562	563	564	565	566	567	568	569	570
1	5	5	1	4	4	2	3	5	2	5	1	2	2	1

571	572	573	574	575	576	577	578	579	580	581	582	583	584	585
4	5	5	3	2	3	4	3	3	1	4	3	3	5	1

586	587	588	589	590	591	592	593	594	595	596	597	598	599	600
1	3	5	4	2	3	3	2	1	3	1	1	1	2	2

601	602	603	604	605	606	607	608	609	610	611	612	613	614	615
3	1	1	1	3	2	3	3	4	3	5	1	4	5	1

616	617	618	619	620	621	622	623	624	625	626	627	628	629	630
2	3	3	1	5	5	2	3	3	3	3	3	2	5	2

631	632	633	634	635	636	637	638	639	640	641	642	643	644	645
5	3	3	1	5	5	1	5	5	3	1	5	5	3	3

646	647	648	649	650	651	652	653	654	655	656	657	658	659	660
5	2	4	3	3	3	3	5	3	5	3	5	3	5	2

661	662	663	664	665	666	667	668	669	670	671	672	673	674	675
2	1	4	1	3	1	5	2	3	3	3	2	4	1	2

676	677	678	679	680	681	682	683	684	685	686	687	688	689	690
4	1	2	1	3	3	4	3	5	1	3	1	3	1	1

691	692	693	694	695	696	697	698	699	700	701	702	703	704	705
4	1	3	3	1	3	3	4	4	4	5	3	1	3	4

706	707	708	709	710	711	712	713	714	715	716	717	718	719	720
1	3	4	5	4	4	3	3	3	1	4	2	2	1	1

721	722	723	724	725	726	727	728	729	730	731	732	733	734	735
3	1	3	5	5	1	1	4	1	2	5	4	3	3	3

736	737	738	739	740	741	742	743	744	745	746	747	748	749	750
1	3	3	1	5	5	1	4	3	3	5	3	5	1	1

751	752	753	754	755	756	757	758	759	760	761	762	763	764	765
4	4	5	4	2	2	5	1	1	3	3	1	1	1	3

766	767	768	769	770	771	772	773	774	775	776	777	778	779	780
5	1	5	1	3	3	3	1	5	3	1	1	3	3	1

781	782	783	784	785	786	787	788	789	790	791	792	793	794	795
3	5	1	5	4	5	4	1	5	3	5	1	1	3	4

796	797	798	799	800	801	802	803	804	805	806	807	808	809	810
1	2	1	3	4	1	5	4	1	1	1	3	3	2	1

811	812	813	814	815	816	817	818	819	820	821	822	823	824	825
1	1	3	5	1	5	4	3	1	5	3	5	1	5	1

826	827	828	829	830	831	832	833	834	835	836	837	838	839	840
2	4	1	1	1	1	5	3	2	3	2	1	2	4	1

841	842	843	844	845	846	847	848	849	850	851	852	853	854	855
1	3	1	2	1	5	2	5	5	3	2	5	3	5	5

856	857	858	859	860	861	862	863	864	865	866	867	868	869	870
1	2	5	5	1	5	1	4	5	5	1	1	5	1	1

871	872	873	874	875	876	877	878	879	880	881	882	883	884	885
3	5	4	3	5	4	1	1	1	1	3	5	3	1	3

886	887	888	889	890	891	892	893	894	895	896	897	898	899	900
4	4	4	1	5	5	5	1	5	1	5	2	1	1	3

901	902	903	904	905	906	907	908	909	910	911	912	913	914	915
3	3	4	1	1	4	2	2	5	3	4	3	4	4	1

916	917	918	919	920	921	922	923	924	925	926	927	928	929	930
4	3	5	5	1	2	5	2	3	1	5	5	1	1	2

931	932	933	934	935	936	937	938	939	940	941	942	943	944	945
1	1	5	4	1	4	4	5	3	1	5	2	1	3	1

946	947	948	949	950	951	952	953	954	955	956	957	958	959	960
1	2	5	5	1	1	2	3	1	5	5	3	1	1	5

961	962	963	964	965	966	967	968	969	970	971	972	973	974	975
1	4	4	5	1	3	4	1	1	5	3	1	5	5	5

```
976  977  978  979  980  981  982  983  984  985  986  987  988  989  990                 991  992  993  994  995  996  997  998  999 1000
  1    1    2    5    5    1    1    2    2    3    1    1    5    1    1                   3    4    5    3    3    1    1    3    1    1
 [ reached getOption("max.print") -- omitted 9836 entries ]

Within cluster sum of squares by cluster:
[1] 9152948774 5495589969 6808452033 7569502550 5727822822
 (between_SS / total_SS =  73.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> ## visualize clustering k=5
> fviz_cluster(kmeans5.result, data = googleplaystore)
> ##--------------------------
> # group into 4 clusters
> pam1.result <- pam(googleplaystore, 2)
> plot(pam1.result)
Hit <Return> to see next plot: fviz_cluster(pam1.result, data = googleplaystore)
Hit <Return> to see next plot: pam2.result <- pam(googleplaystore, 3)
> fviz_cluster(pam2.result, data = googleplaystore)
> plot(pam2.result)
Hit <Return> to see next plot: pam3.result <- pam(googleplaystore, 4)
Hit <Return> to see next plot: fviz_cluster(pam3.result, data = googleplaystore)
> plot(pam3.result)
Hit <Return> to see next plot: pam4.result <- pam(googleplaystore, 5)
Hit <Return> to see next plot: fviz_cluster(pam4.result, data = googleplaystore)
> plot(pam4.result)
Hit <Return> to see next plot: #---Evaluation
Hit <Return> to see next plot: ##--------------------------
> # group into 4 clusters
> pam.result <- pam(googleplaystore, 2)
> plot(pam.result)
Hit <Return> to see next plot: pam.result <- pam(googleplaystore, 3)
Hit <Return> to see next plot:
>
> pam.result <- pam(googleplaystore, 3)
> plot(pam.result)
Hit <Return> to see next plot: pam.result <- pam(googleplaystore, 4)
Hit <Return> to see next plot: plot(pam.result)
> pam.result <- pam(googleplaystore, 5)
> plot(pam.result)
Hit <Return> to see next plot: fviz_nbclust(googleplaystore, kmeans, method = "silhouette")+
labs(subtitle = "Silhouette method")
Hit <Return> to see next plot:
> install.packages("pvcluster")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and
install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Mashari Aljasser/Documents/R/win-library/3.6’
(as ‘lib’ is unspecified)
> library(pvclust)
> dd <- dist(scale(googleplaystore), method = "euclidean")
> hc <- hclust(dd, method = "ward.D")
> plot(hc)
> set.seed(1234)
> result <- pvclust(googleplaystore[1:100, 1:10], method.dist="euclidean",
+                  method.hclust="average", nboot=10)
Bootstrap (r = 0.5)... Done.
Bootstrap (r = 0.6)... Done.
Bootstrap (r = 0.7)... Done.
Bootstrap (r = 0.8)... Done.
Bootstrap (r = 0.9)... Done.
Bootstrap (r = 1.0)... Done.
Bootstrap (r = 1.1)... Done.
Bootstrap (r = 1.2)... Done.
Bootstrap (r = 1.3)... Done.
Bootstrap (r = 1.4)... Done.
> plot(result)
> pvrect(result)
>
> result <- pvclust(googleplaystore[1:100, 1:10], method.dist="euclidean",
+                  method.hclust="average", nboot=10)
Bootstrap (r = 0.5)... Done.
Bootstrap (r = 0.6)... Done.
Bootstrap (r = 0.7)... Done.
Bootstrap (r = 0.8)... Done.
Bootstrap (r = 0.9)... Done.
Bootstrap (r = 1.0)... Done.
Bootstrap (r = 1.1)... Done.
Bootstrap (r = 1.2)... Done.
Bootstrap (r = 1.3)... Done.
Bootstrap (r = 1.4)... Done.
> plot(result)
> pvrect(result)
> dend <- googleplaystore[1:30,-5] %>% scale %>% dist %>%
+   hclust %>% as.dendrogram %>%

+   set("branches_k_color", k=3) %>% set("branches_lwd", 1.2) %>%

+   set("labels_colors") %>% set("labels_cex", c(.9,1.2)) %>%

+   set("leaves_pch", 19) %>% set("leaves_col", c("blue", "red"))

> # plot the dend in usual "base" plotting engine:

> plot(dend)

>
```