Runa Muderrisoglu & Murtaza Gohari

Professor Berg

CSC 260

2/28/2024

K-Means Clustering Analysis Report for Walmart Customer Segmentation

Introduction

In our collaborative effort, we've undertaken a K-means clustering analysis of the "K_Means_Study.csv" dataset. The goal was to identify distinct customer segments that could be targeted with tailored marketing strategies by a retail company such as Walmart.

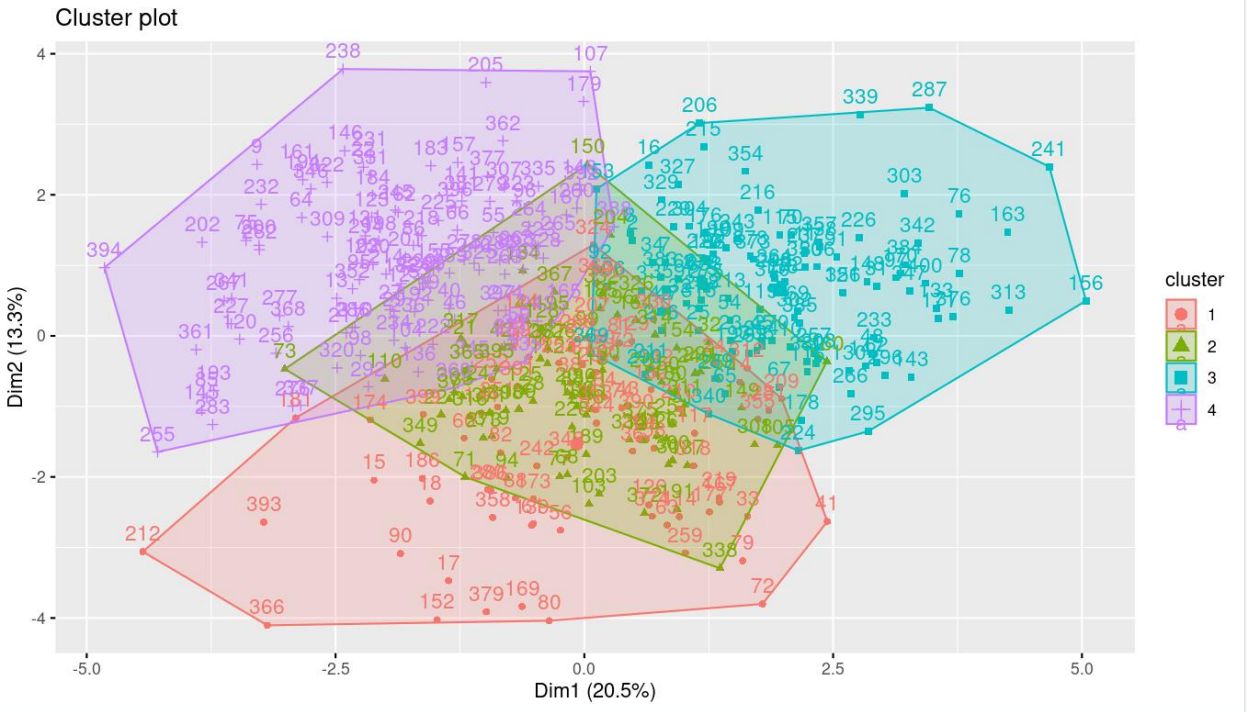
Methodology

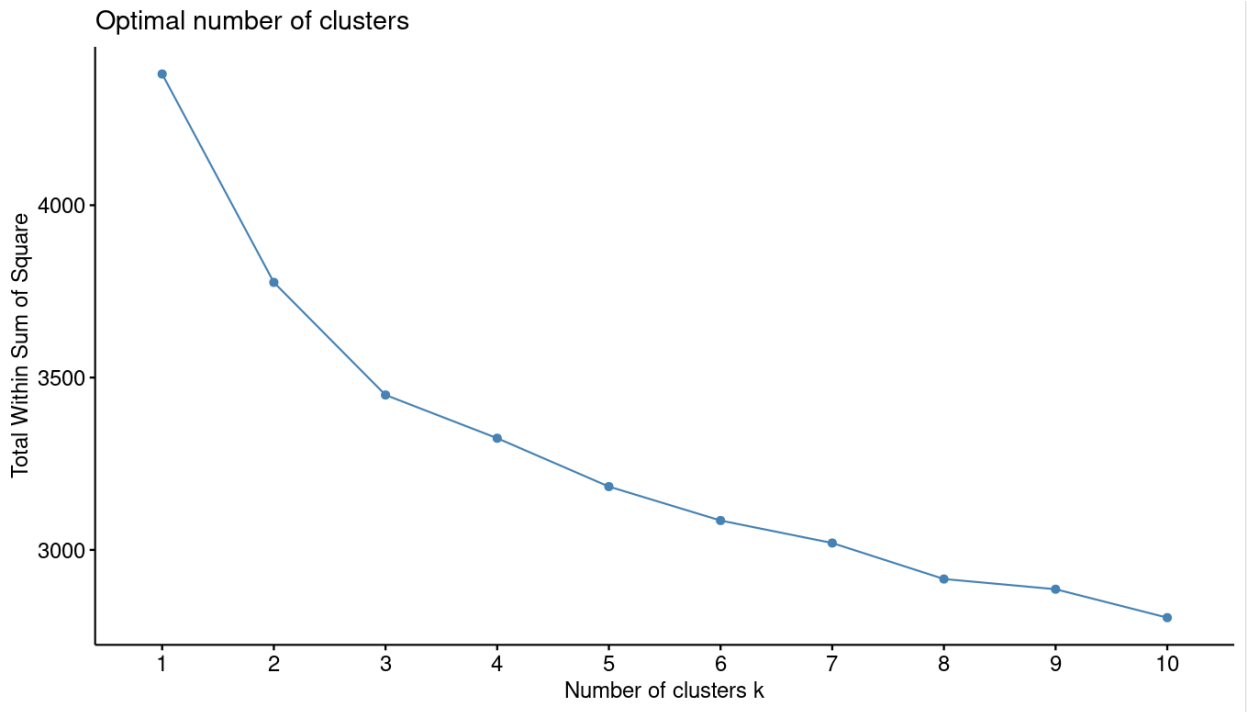
We engaged in a methodical approach, utilizing R programming to cluster the dataset and employed various methods, including the silhouette, gap, and elbow methods, to determine the optimal number of clusters. Graphical representations of our findings were generated to support our analysis.

Graphical Analysis

Our examination of the gap statistic plot indicated an optimal cluster count at k=3, which signifies a balance in intra-cluster cohesion and inter-cluster separation. However, the elbow method suggested a bend at k=4, hinting that additional clusters beyond this point would not significantly enhance the variance explained. Meanwhile, the silhouette analysis favored k=2 for its higher average silhouette width, suggesting well-defined clusters at this level.

The scatter plots for k=2, k=3, k=4, k=5, k=6, and k=8 illustrated the segmentation of the dataset. While k=2 and k=3 demonstrated clearer separations, the distinction blurred as the number of clusters increased. The individual cluster plot for k=4 provided an in-depth view of how segments are divided, showing potentially meaningful customer groupings despite some overlap.





Findings

The varying recommendations from the silhouette, gap, and elbow methods presented us with a challenge in determining the optimal number of clusters. After careful consideration of the graphical representations and the interpretability of the results:

- The **Gap Statistic** emphasized the simplicity of 3 clusters.
- The **Elbow Method** illustrated diminishing returns after 4 clusters.
- The **Silhouette Method** showed the highest clarity at 2 clusters.

Recommendations

We recommend adopting **4 clusters** for the following reasons:

The Elbow Method provides a trade-off between specificity and generalizability, which is crucial for actionable marketing insights.

Four clusters offer a more granular understanding of customer preferences, which can align with Walmart's diverse product offerings.

It enables Walmart to develop nuanced marketing strategies tailored to each segment's unique characteristics.

We suggest that Walmart consider these clusters as follows:

Cluster 1: **Entertainment Enthusiasts**

High interest in theater and live music shows.

Marketing strategy could include promoting entertainment-related products and organizing in-store events.

Cluster 2: **Homebodies**

High interest in gardening and watching TV.

Could be targeted with home and garden products, as well as electronics for home entertainment.

Cluster 3: **Social Readers**

Enjoy reading and attending social functions.

Strategies may involve book signings and reading clubs, along with social gathering spaces in stores.

Cluster 4: **Active Shoppers**

Show a balanced interest across activities.

A general approach with varied promotions could work well for this group.

Conclusion

The K-means clustering analysis, reinforced by our graphical evaluations, has yielded a robust framework for Walmart to segment its customer base effectively. While further investigation could refine these insights, our analysis provides a strong foundation for strategic marketing initiatives. As Walmart seeks to serve its customers with precision and care, the segmentation strategy informed by our study stands to significantly bolster its marketing efforts.

R Code:

```
> data = read.csv("K_Means_Study.csv")
>
> install.packages("amap")
Installing package into ‘/cloud/lib/x86_64-pc-linux-gnu-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'http://rspm/default/__linux__/focal/latest/src/contrib/amap_0.8-
19.tar.gz'
Content type 'application/x-gzip' length 175548 bytes (171 KB)
==================================================
downloaded 171 KB

* installing *binary* package ‘amap’ ...
* DONE (amap)

The downloaded source packages are in
      ‘/tmp/RtmpMtwcpE/downloaded_packages’
> library(amap)
> set.seed(456)
> clusters = kmeans(data, 4, nstart = 30)
> print(clusters)
K-means clustering with 4 clusters of sizes 83, 83, 114, 120

Cluster means:
  ATTEND.THEATER PARTIES.OR.SOCIAL.FUNCTIONS READING..MAGAZINES READING..BOOKS
1       1.493976                    2.228916           2.397590       2.289157
2       1.650602                    2.120482           2.987952       2.963855
3       1.219298                    1.798246           1.728070       1.780702
4       1.750000                    2.675000           2.683333       2.525000
  GO.OUT.WITH.FRIENDS NEWS.PAPERS TRAVEL.ON.VACATION VISIT.RELATIVES LISTEN.TO.RADIO
1            2.277108    2.530120           2.554217        3.012048        3.518072
2            2.108434    3.084337           2.506024        2.433735        3.000000
3            1.877193    2.000000           2.070175        2.175439        2.833333
4            3.375000    2.491667           2.958333        2.233333        3.583333
  LIVE.MUSIC.SHOWS   GARDEN WATCH.TV GO.TO.BARS LISTEN.TO.MUSIC OUT.WITH.THE.FAMILY
1         1.734940 3.180723 3.469880   1.807229        2.939759            3.132530
2         1.506024 1.831325 3.120482   1.481928        2.662651            2.939759
3         1.280702 1.640351 3.052632   1.517544        2.289474            2.289474
4         2.358333 1.625000 3.008333   2.841667        3.250000            2.300000
  ATTEND.RELIGIOUS.SERVICES
1                  2.578313
2                  1.915663
3                  1.894737
4                  1.800000

Clustering vector:
  [1] 2 3 3 3 2 2 3 3 4 3 3 4 4 1 1 3 1 1 4 2 2 4 3 1 3 2 4 2 4 2 3 2 1 3 2 1 2 1 4 4
1 2 1 2
 [45] 3 4 3 3 3 1 2 3 4 3 4 1 1 1 4 1 1 3 1 4 3 4 3 2 3 3 2 1 2 1 4 3 2 3 1 1 1 1 2 1
```

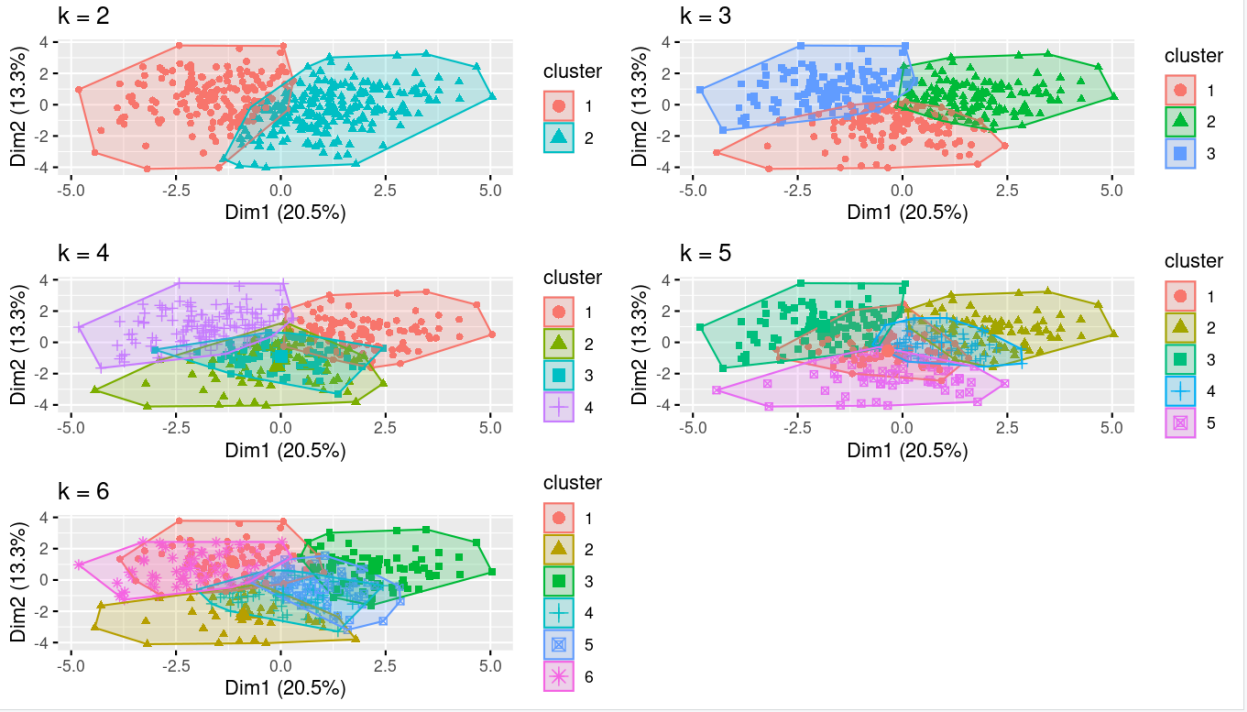
```
4 4 1 1
 [89] 4 1 3 3 4 2 4 4 3 4 2 2 2 4 2 4 2 3 4 2 2 2 2 2 1 1 3 3 1 2 3 4 3 3 4 1 2 1 1 2
1 3 4 1
[133] 3 2 2 4 2 3 1 4 4 1 3 1 4 4 3 3 2 2 1 1 3 2 4 3 4 3 1 2 4 4 3 2 4 3 1 4 1 3 3 1
1 1 3 3
[177] 1 3 4 2 1 4 4 4 2 1 4 3 2 3 2 4 4 4 2 2 3 4 3 4 4 4 2 2 4 3 1 3 1 1 3 1 3 4 3 3
4 4 1 3
[221] 2 4 4 3 4 3 4 2 3 4 4 4 3 4 4 4 3 4 4 4 3 1 4 4 4 3 2 2 2 2 2 1 1 3 4 4 3 2 1 4
2 2 4 4
[265] 4 3 4 4 1 3 4 3 2 4 3 4 4 4 4 1 1 4 4 1 4 4 3 3 3 3 2 4 3 4 3 3 1 3 4 2 2 3 3 3
2 3 4 2
[309] 4 4 2 1 3 2 3 2 2 1 2 4 3 4 4 1 4 2 3 4 3 1 4 2 3 2 4 3 4 2 3 3 4 3 3 1 4 4 3 1
2 3 1 4
[353] 1 3 1 3 3 1 3 4 4 4 1 3 2 1 2 4 3 3 2 2 3 1 2 3 4 3 1 1 4 3 3 3 3 1 2 4 4 1 1 4
1 4 2 4
[397] 4 2 1 3

Within cluster sum of squares by cluster:
[1]  790.3855  554.6747  919.0614 1031.8583
 (between_SS / total_SS =  24.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
"betweenss"
[7] "size"         "iter"         "ifault"
> newDataSet = cbind(data, cluster = clusters$cluster)
>
> install.packages("factoextra")
Error in install.packages : Updating loaded packages
> library(factoextra)
> fviz_cluster(clusters, data = data)
> k2 = kmeans(data, 2, nstart = 30)
> k3 = kmeans(data, 3, nstart = 30)
> k4 = kmeans(data, 4, nstart = 30)
> k5 = kmeans(data, 5, nstart = 30)
> k6 = kmeans(data, 6, nstart = 30)
> plot1 = fviz_cluster(k2, geom = "point", data = data) + ggtitle("k = 2")
> plot2 = fviz_cluster(k3, geom = "point", data = data) + ggtitle("k = 3")
> plot3 = fviz_cluster(k4, geom = "point", data = data) + ggtitle("k = 4")
> plot4 = fviz_cluster(k5, geom = "point", data = data) + ggtitle("k = 5")
> plot5 = fviz_cluster(k6, geom = "point", data = data) + ggtitle("k = 6")
>
> library(gridExtra)
> grid.arrange(plot1,plot2,plot3,plot4,plot5)
> set.seed(456)
> fviz_nbclust(data, kmeans, method="wss")
> fviz_nbclust(data, kmeans, method="silhouette")
>
> install.packages("cluster")
```

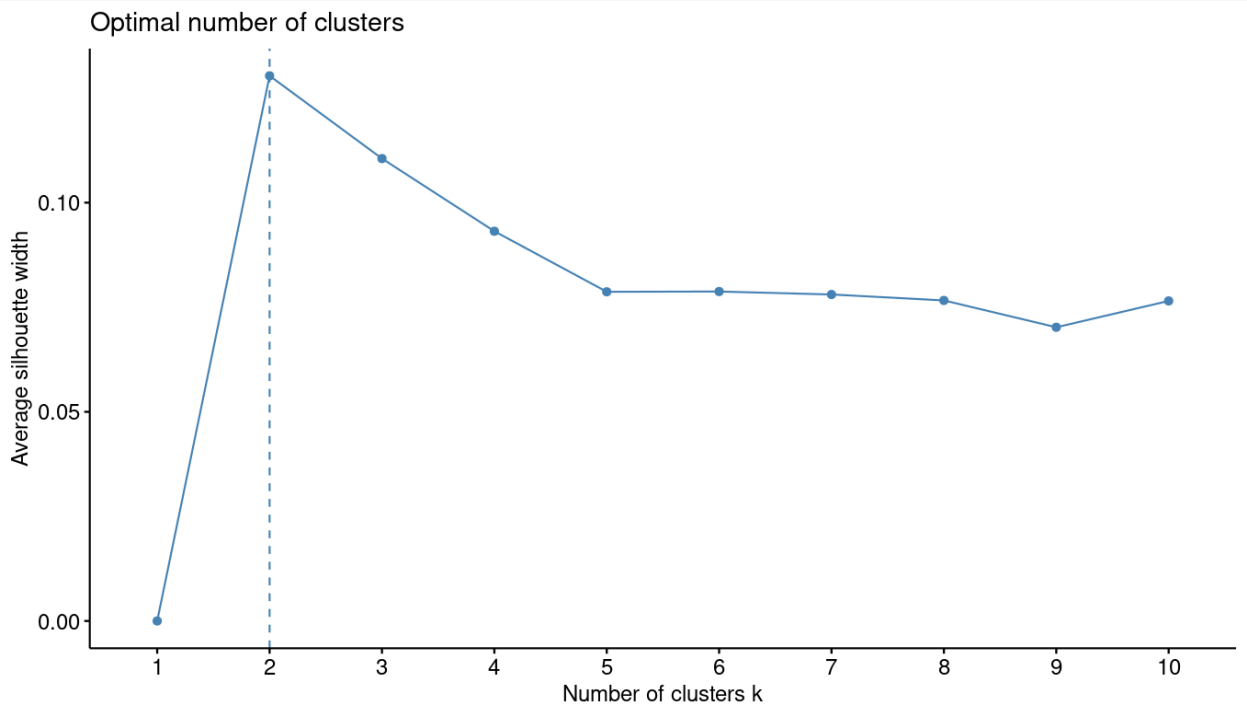
```
Error in install.packages : Updating loaded packages
> library(cluster)
> set.seed(456)
> GapStat = clusGap(data, FUN=kmeans,nstart=30, B=60, K.max =12)
Clustering k = 1,2,..., K.max (= 12): .. done
Bootstrapping, b = 1,2,..., B (= 60)  [one "." per sample]:
.................................................. 50
.......... 60
Warning messages:
1: did not converge in 10 iterations
2: did not converge in 10 iterations
3: did not converge in 10 iterations
4: did not converge in 10 iterations
5: did not converge in 10 iterations
6: did not converge in 10 iterations
> fviz_gap_stat(GapStat)
>
> k3 = kmeans(data, centers = 3, nstart = 30)
> k2 = kmeans(data, centers = 2, nstart = 30)
> k8 = kmeans(data, centers = 8, nstart = 30)
> plot3 = fviz_cluster(k3, geom = "point",
+                      data = data) + ggtitle("k = 3")
> plot2 = fviz_cluster(k2, geom = "point",
+                      data = data) + ggtitle("k = 2")
> plot8 = fviz_cluster(k8, geom = "point",
+                      data = data) + ggtitle("k = 8")
> grid.arrange(plot3, plot2, plot8, nrow = 2)
> print(k3)
K-means clustering with 3 clusters of sizes 135, 130, 135

Cluster means:
  ATTEND.THEATER PARTIES.OR.SOCIAL.FUNCTIONS READING..MAGAZINES READING..BOOKS
1       1.303704                    1.800000           1.792593       1.859259
2       1.738462                    2.669231           2.646154       2.515385
3       1.540741                    2.200000           2.814815       2.696296
  GO.OUT.WITH.FRIENDS NEWS.PAPERS TRAVEL.ON.VACATION VISIT.RELATIVES LISTEN.TO.RADIO
1            1.829630    2.088889           2.074074        2.251852        2.851852
2            3.338462    2.500000           2.907692        2.230769        3.538462
3            2.237037    2.859259           2.614815        2.770370        3.325926
  LIVE.MUSIC.SHOWS   GARDEN WATCH.TV GO.TO.BARS LISTEN.TO.MUSIC OUT.WITH.THE.FAMILY
1         1.325926 1.755556 3.014815   1.614815        2.311111            2.325926
2         2.338462 1.638462 3.015385   2.807692        3.230769            2.323077
3         1.592593 2.577778 3.385185   1.511111        2.844444            3.148148
  ATTEND.RELIGIOUS.SERVICES
1                  1.918519
2                  1.792308
3                  2.318519

Clustering vector:
  [1] 3 1 1 1 3 3 1 1 2 1 1 2 2 1 3 1 3 3 2 3 2 2 1 1 1 3 2 3 2 3 1 1 3 1 3 3 3 3 2 2
3 2 3 2
```

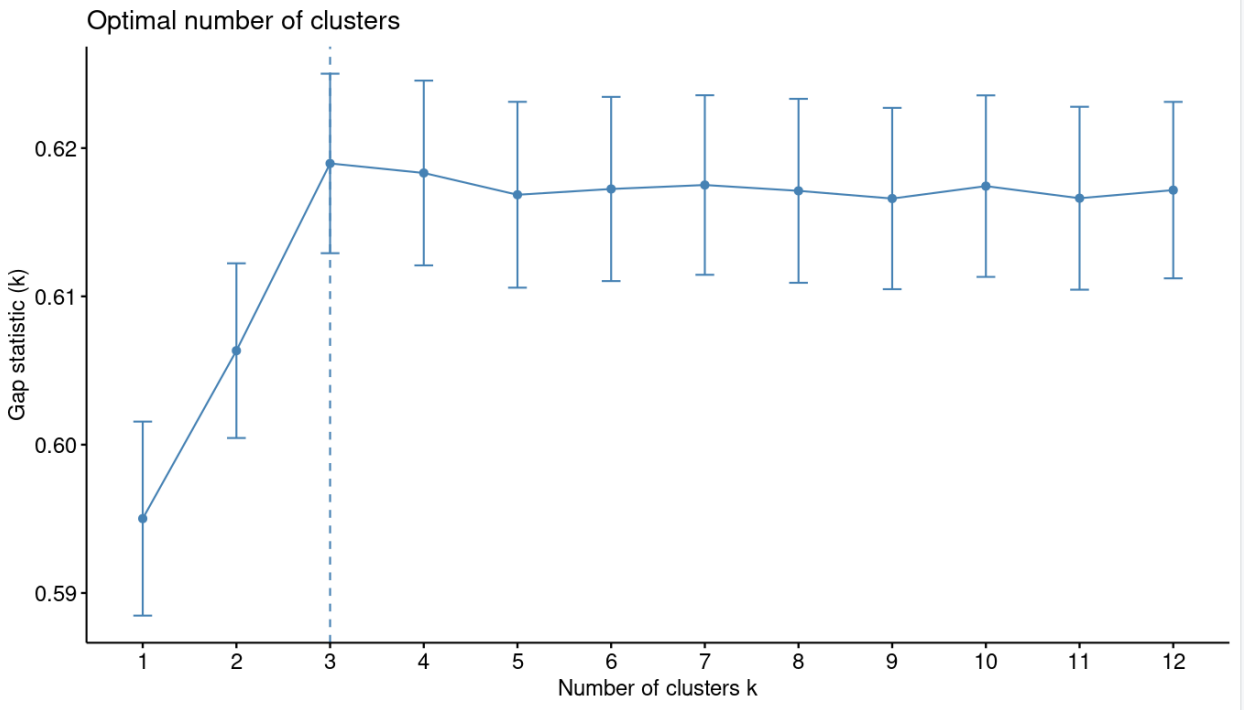
```
 [45] 1 2 1 1 1 1 3 1 2 1 2 3 3 3 2 3 3 1 3 2 1 2 1 3 1 1 3 3 2 3 2 1 3 1 3 3 3 3 3 3
2 2 3 3
 [89] 2 3 1 2 2 3 2 2 1 2 3 3 3 2 3 2 3 1 2 3 3 3 3 3 3 3 1 1 3 3 1 2 1 1 2 2 3 3 1 3
3 1 2 3
[133] 1 2 3 2 3 1 3 2 2 3 1 3 2 2 1 1 1 1 3 3 2 3 2 1 2 1 3 1 2 2 1 1 2 1 3 2 3 1 1 3
3 3 1 1
[177] 3 1 2 1 2 2 2 2 1 3 2 1 3 1 3 2 2 2 3 1 1 2 1 2 2 2 3 2 2 1 3 1 1 1 1 3 1 2 1 1
2 2 3 1
[221] 3 2 2 1 2 1 2 3 1 2 2 2 1 2 2 2 1 2 2 2 1 3 2 2 2 1 3 3 1 3 3 3 3 1 2 2 1 3 3 2
3 3 2 2
[265] 2 1 2 2 3 1 3 1 3 2 1 3 2 2 2 3 3 2 2 3 2 2 1 1 1 1 3 2 1 2 1 1 3 1 2 3 3 1 1 1
3 1 2 3
[309] 2 2 3 1 1 1 1 3 3 3 3 2 1 2 2 2 2 1 1 2 1 1 2 1 1 3 2 1 2 3 1 1 2 1 1 3 2 2 1 3
3 1 3 2
[353] 3 1 1 1 1 3 1 2 2 2 1 1 3 3 2 2 1 1 3 3 1 3 3 1 2 1 3 1 2 1 1 1 1 3 3 2 2 3 3 2
3 2 3 2
[397] 2 3 3 1

Within cluster sum of squares by cluster:
[1] 1121.644 1156.138 1171.600
 (between_SS / total_SS =  21.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
"betweenss"
[7] "size"         "iter"         "ifault"
> print(k2)
K-means clustering with 2 clusters of sizes 233, 167

Cluster means:
  ATTEND.THEATER PARTIES.OR.SOCIAL.FUNCTIONS READING..MAGAZINES READING..BOOKS
1       1.360515                    1.914163           2.163090       2.171674
2       1.754491                    2.640719           2.766467       2.610778
  GO.OUT.WITH.FRIENDS NEWS.PAPERS TRAVEL.ON.VACATION VISIT.RELATIVES LISTEN.TO.RADIO
1            1.922747     2.39485           2.253219        2.442060        3.034335
2            3.203593     2.60479           2.910180        2.389222        3.514970
  LIVE.MUSIC.SHOWS   GARDEN WATCH.TV GO.TO.BARS LISTEN.TO.MUSIC OUT.WITH.THE.FAMILY
1         1.377682 2.150215 3.180258   1.532189        2.489270            2.686695
2         2.257485 1.778443 3.083832   2.574850        3.209581            2.485030
  ATTEND.RELIGIOUS.SERVICES
1                  2.111588
2                  1.874251

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 1 2 2 1 1 1 1 2 1 2 1 1 1 1 1 2 1 2 1 2 2
1 2 1 2
 [45] 1 2 1 1 1 1 1 1 2 1 2 1 1 1 2 2 1 1 1 2 1 2 1 1 1 1 2 1 2 1 2 1 1 1 1 1 1 1 1 1
2 2 1 1
 [89] 2 2 1 2 2 1 2 2 1 2 1 1 1 2 1 2 1 1 2 1 2 2 1 1 2 1 1 1 1 1 1 2 1 1 2 2 1 1 1 1
```

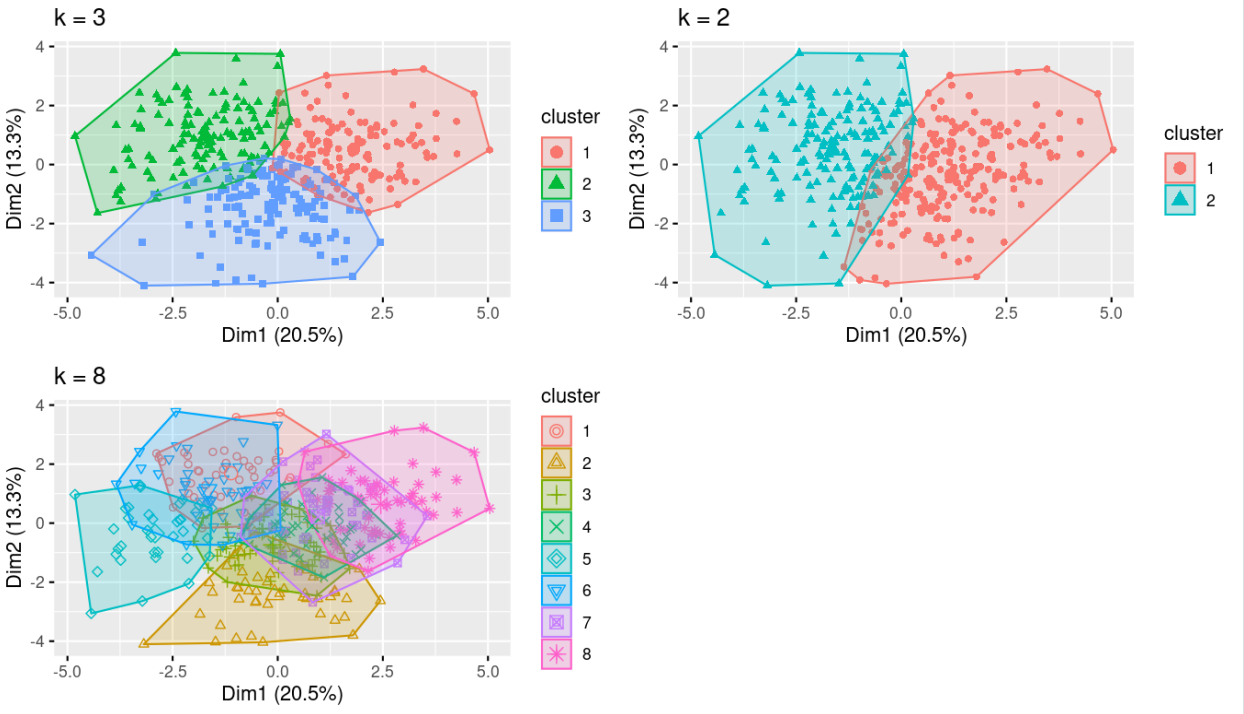
```
1 1 2 1
[133] 1 2 1 2 1 1 1 2 2 1 1 2 2 2 1 1 1 2 2 2 2 1 2 1 2 1 1 1 2 2 1 1 2 1 1 2 1 1 1 1
1 2 1 1
[177] 1 1 2 1 2 2 2 2 1 2 2 1 1 1 1 2 2 2 1 1 1 2 1 2 2 2 1 2 2 1 1 1 1 1 1 2 1 2 1 1
2 2 1 1
[221] 2 2 2 1 2 1 2 1 1 2 2 2 1 2 2 2 1 2 2 2 2 1 1 2 2 2 1 2 1 1 1 1 2 1 1 2 2 1 1 1 2
1 1 2 2
[265] 2 1 2 2 1 1 2 1 2 2 1 2 2 2 2 1 1 2 2 1 2 2 1 1 1 1 1 2 1 2 1 1 1 1 2 1 1 1 1 1
2 1 2 1
[309] 2 2 1 1 1 1 1 2 2 1 1 2 1 2 2 2 2 1 1 2 1 1 2 1 1 1 2 1 2 1 1 1 2 1 1 1 2 2 1 1
2 1 2 2
[353] 1 1 1 1 1 1 1 2 2 2 1 1 2 2 2 2 1 1 2 1 1 1 1 1 2 1 1 1 2 1 1 1 1 1 2 2 2 1 2 2
2 2 2 2
[397] 2 2 2 1

Within cluster sum of squares by cluster:
[1] 2152.867 1623.138
 (between_SS / total_SS =  13.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
"betweenss"
[7] "size"         "iter"         "ifault"
> print(k8)
K-means clustering with 8 clusters of sizes 46, 46, 68, 48, 36, 44, 49, 63

Cluster means:
  ATTEND.THEATER PARTIES.OR.SOCIAL.FUNCTIONS READING..MAGAZINES READING..BOOKS
1       1.847826                   2.456522           2.500000       2.521739
2       1.500000                   2.217391           2.847826       2.521739
3       1.661765                   2.132353           3.058824       2.941176
4       1.479167                   2.000000           1.875000       1.895833
5       2.000000                   2.805556           3.333333       3.250000
6       1.500000                   2.795455           2.431818       1.931818
7       1.244898                   2.000000           1.693878       2.081633
8       1.158730                   1.730159           1.777778       1.825397
  GO.OUT.WITH.FRIENDS NEWS.PAPERS TRAVEL.ON.VACATION VISIT.RELATIVES LISTEN.TO.RADIO
1            3.260870    2.391304           2.630435        1.847826        3.130435
2            2.152174    2.956522           2.652174        3.152174        3.369565
3            2.132353    3.073529           2.500000        2.470588        3.117647
4            2.104167    2.312500           2.270833        2.312500        3.270833
5            3.305556    3.194444           3.222222        2.638889        3.805556
6            3.454545    2.136364           3.113636        2.318182        3.772727
7            2.163265    1.734694           2.081633        2.510204        3.489796
8            1.761905    2.111111           2.126984        2.206349        2.412698
  LIVE.MUSIC.SHOWS   GARDEN WATCH.TV GO.TO.BARS LISTEN.TO.MUSIC OUT.WITH.THE.FAMILY
1         2.239130 1.760870 2.217391   2.804348        2.934783            2.086957
2         1.543478 3.217391 3.304348   1.369565        2.847826            3.434783
3         1.544118 1.632353 3.235294   1.514706        2.691176            2.897059
```

```
4         1.770833 3.187500 3.104167   2.062500        2.395833            2.583333
5         2.500000 2.111111 3.500000   2.666667        3.583333            2.666667
6         2.386364 1.363636 3.454545   3.068182        3.204545            2.363636
7         1.448980 1.510204 3.693878   1.673469        3.265306            2.612245
8         1.079365 1.507937 2.761905   1.269841        1.936508            2.190476
  ATTEND.RELIGIOUS.SERVICES
1                  1.543478
2                  3.043478
3                  1.882353
4                  1.895833
5                  1.777778
6                  2.022727
7                  1.959184
8                  2.000000

Clustering vector:
  [1] 3 8 4 4 3 3 4 7 6 8 7 5 1 4 5 8 2 2 5 3 3 1 4 4 4 3 5 3 6 2 8 3 2 7 3 7 3 4 3 1
2 3 7 3
 [45] 7 6 4 8 4 7 3 4 6 4 1 2 7 2 6 2 2 8 7 6 8 6 8 3 4 8 3 2 5 4 6 8 2 8 2 2 4 2 3 2
5 1 4 2
 [89] 6 2 8 7 6 2 5 1 8 5 4 3 3 6 2 1 2 7 1 3 3 3 3 3 2 2 8 7 7 3 7 6 8 8 6 4 3 2 7 3
4 8 6 4
[133] 8 3 3 5 3 8 2 1 1 7 8 2 5 6 4 8 3 1 7 2 7 3 1 8 1 4 4 8 1 1 8 3 7 8 2 1 2 8 7 4
2 5 7 8
[177] 2 8 6 3 5 6 1 1 4 2 1 8 3 8 3 6 5 1 3 3 8 1 7 5 6 6 2 1 1 7 4 8 4 4 7 5 4 1 1 8
5 6 2 8
[221] 3 6 6 8 1 8 5 3 7 1 6 6 8 5 7 3 8 6 6 1 8 2 7 6 1 8 3 3 3 3 3 4 4 4 5 5 8 4 2 1
3 3 7 1
[265] 6 8 5 6 4 7 1 7 3 1 8 5 5 6 6 2 4 5 5 2 6 5 8 7 8 7 4 6 8 6 7 7 4 4 6 3 3 7 8 4
3 4 1 3
[309] 1 5 3 4 8 3 7 3 3 4 3 5 7 6 1 4 6 3 7 1 7 4 1 3 7 3 1 8 5 2 8 8 5 8 8 7 6 1 8 2
3 8 2 5
[353] 7 1 7 8 8 2 7 6 5 6 4 8 3 2 3 5 6 7 3 2 7 2 3 8 1 7 2 4 6 8 7 8 8 2 3 1 1 4 4 1
5 5 7 1
[397] 6 3 5 8

Within cluster sum of squares by cluster:
[1] 343.6957 360.5217 409.7500 360.4792 284.3611 336.1818 366.2041 448.6667
 (between_SS / total_SS =  33.6 %)

Available components:

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


k = 2
k = 3
k = 4
k = 5
k = 6
Dim1 (20.5%)
Dim2 (13.3%)
cluster


Optimal number of clusters
Average silhouette width
Number of clusters k


Optimal number of clusters
Gap statistic (k)
0.62
0.61
0.60
0.59
1
2
3
4
5
6
7
8
9
10
11
12
Number of clusters k


k = 3
Dim2 (13.3%)
Dim1 (20.5%)
cluster
1
2
3
k = 2
Dim2 (13.3%)
Dim1 (20.5%)
cluster
1
2
k = 8
Dim2 (13.3%)
Dim1 (20.5%)
cluster
1
2
3
4
5
6
7
8