Code

Statistical Analysis Handout for the Market Segmentation Lecture

- Cluster Analysis
- Reading and outputing data
- Hierarchical Clustering Analysis
  - The four-cluster solution
  - Three-Cluster Solution
  - Number of Clusters
  - Targeting the Clusters/segments
    - Demographics
    - Choice
- K-Means
  - Three Cluster Solution obtained using K-Means
  - Find the optimal number of clusters
  - Results Comparison
    - Demographics
    - Choice
    - Hierarchical Clustering
- Latent Class Analysis
  - Find the optimal model
  - Results Comparison
    - Demographics
    - Choice
    - Hierarchical Clustering
    - K-Means Clustering

This handout is designed to help you replicate the statistical analyses that were covered in the Market Segmentation lecture. You should have this handout handy when you work on the (Market Segmentation) programming assignment where you will be asked to apply the learnings from the lecture on a different dataset.

Cluster Analysis

Cluster Analysis refers to a class of techniques used to classify individuals into groups such that:

- Individuals within a group should be as similar as possible
- Individuals belonging to different groups should be as dissimilar as possible

This handout shows three different cluster analysis techniques:

1. Hierarchical clustering
2. K-Means
3. Latent Class Analysis

In order to run these statistical methods, you need to install these R packages:

- NbClust
- mclust
- gmodels

Hide

```
set.seed(1990)
library(NbClust)
library(mclust)
library(gmodels)
```

Reading and outputing data

The dataset includes data from 73 students (24 MBAs and 49 undergrads). These students were asked to allocate 100 points across six automobile attributes (Trendy, Styling, Reliability, Sportiness, Performance, and Comfort) in a way that reflects their importance in the purchase decision of which car to buy. We use this dataset to answer the following questions:

1. Are there different benefit segments among this student population?
2. How many segments?
3. How are they different in their constant-sum allocation?
4. How can we transform this information into actionable levers from a managerial standpoint?

Let us start by reading the data. Remember to start by setting the appropriate directory using the following code

Hide

```
setwd("your_directory")
```

We can now read the raw data:

Hide

```
seg_data <- read.csv(file = "SegmentationData.csv",row.names=1)
head(seg_data)
```

	Trendy <int>	Styling <int>	Reliability <int>	Sportiness <int>	Performance <int>	Comfort <int>	M… <fctr>	Choice <fctr>
1	10	20	35	5	20	10	MBA	Lexus
2	25	5	25	5	25	15	MBA	BMW
3	10	20	30	10	10	20	MBA	Lexus
4	10	15	30	10	20	15	MBA	BMW
5	20	10	40	1	14	15	MBA	Mercedes
6	20	30	10	20	10	10	MBA	Lexus

6 rows

Hierarchical Clustering Analysis

Hierarchical Clustering Analysis is one of the most popular technique used for market segmentation. It is a numerical procedure which attempts to separate a set of observations into clusters from the bottom-up by joining single individuals sequentially until we obtain one large cluster. Hence, this technique doesn't require the pre-specification of the number of clusters,which can be assessed through the “dendogram” (a tree-like representation of the data).

More specifically, the algorithm works as follow:

1. Each respondent is initially assigned to his or her own cluster
2. Identify the distance between each cluster (intially between pairs of respondents)
3. The two closest clusters are combined into one
4. Repeat steps 2 and 3 until there is one unique cluster containing all the observations
5. Represent the clusters in a dendogram

A key aspect of hierarchical clustering consists in choosing how to compute the distance between two clusters. Is it equal to the maximal distance between two points from each of these clusters? Or the minimal distance? What about the distance between two points? In this handout, we will use Ward's criterion which aims to minimize the total variance within-cluster. To do so, we use the R function hclust. We start by standardizing the data so that every variable is on the same scale. We then compute the euclidean distance between observations.

Hide

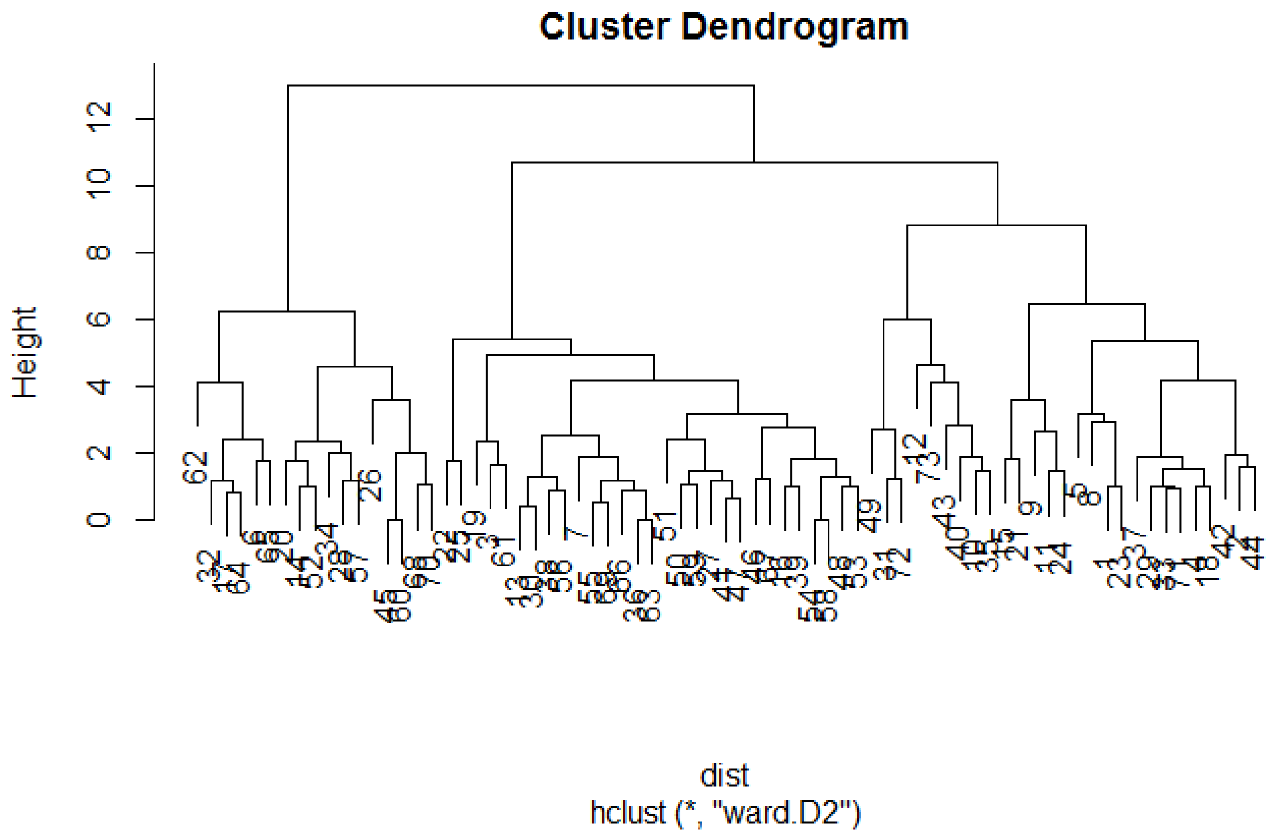
```
std_seg_data <- scale(seg_data[,c("Trendy", "Styling", "Reliability", "Sportiness", "Performa
nce", "Comfort")])
dist <- dist(std_seg_data, method = "euclidean")
as.matrix(dist)[1:5,1:5]
```

```
         1        2        3        4        5
1 0.000000 3.730216 2.802191 1.775616 2.746615
2 3.730216 0.000000 4.218662 3.017462 2.984534
3 2.802191 4.218662 0.000000 1.974683 3.331082
4 1.775616 3.017462 1.974683 0.000000 2.924141
5 2.746615 2.984534 3.331082 2.924141 0.000000
```

We now use the function hclust() to apply hierarchical clustering on our data. We use the Ward criterion which aims to minimize the within-cluster variance. \ We obtain the dendogram below which can help us decide the number of clusters to retain. This number seems to be either 3 or 4. \ Note: It is important to set the seed to a specific value. This way you would always get the same labeling of the clusters. Otherwise, cluster 1 in one analysis may correspond to cluster 3 in another.

Hide

```
set.seed(1990)
clust <- hclust(dist, method = "ward.D2")
plot(clust)
```

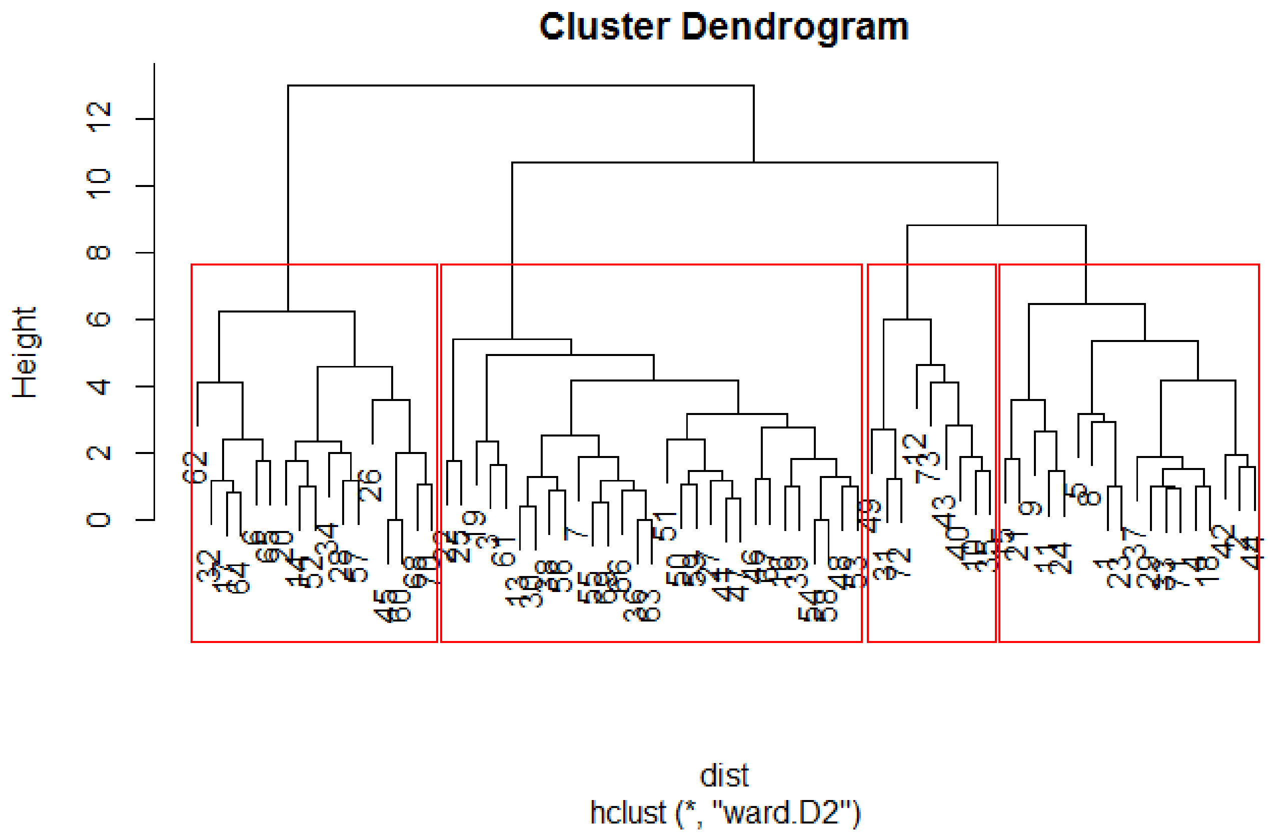


The four-cluster solution

We start by 4 clusters as we see below:

Hide

```
set.seed(1990)
clust <- hclust(dist, method = "ward.D2")
plot(clust)
h_cluster <- cutree(clust, 4)
rect.hclust(clust, k=4, border="red")
```



Let us now look at some description of this clustering. The table below informs us with the number of individuals in each cluster:

Hide

```
table(h_cluster)
```

```
h_cluster
 1  2  3  4 
18 29 17  9 
```

The table below reports the profiles of the four clusters (i.e., the clustering variables means by cluster). Looking at this table, we can describe the clusters as follows:

1. Cluster 1 values reliability and Performance
2. Cluster 2 values Sportiness and Comfort
3. Cluster 3 values Trendiness and Style
4. Cluster 4 values Style and Sportiness

Hence, it seems that Cluster 4 is a combination of Clusters 2 and 3. This suggests that 3 clusters may be better at capturing the heterogeneity of the subjects in this dataset.

Hide

```
hclust_summary <- aggregate(std_seg_data[,c("Trendy", "Styling", "Reliability", "Sportiness", "Performance", "Comfort")],by=list(h_cluster),FUN=mean)
hclust_summary
```

Group.1 <int>	Trendy <dbl>	Styling <dbl>	Reliability <dbl>	Sportiness <dbl>	Performance <dbl>	Comfort <dbl>
1	-0.50357227	-0.6837159	1.09976574	-0.94569654	0.6548024	0.08642535

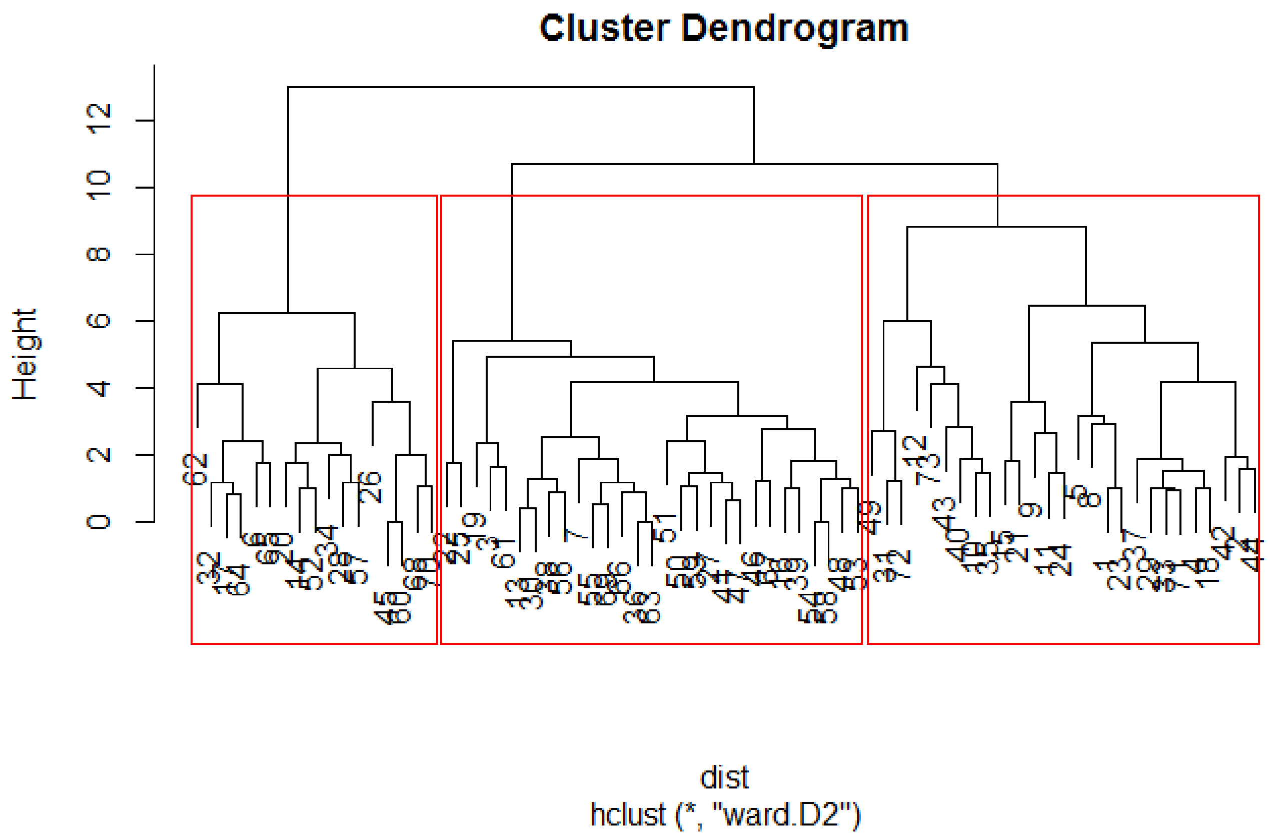
Group.1 <int>	Trendy <dbl>	Styling <dbl>	Reliability <dbl>	Sportiness <dbl>	Performance <dbl>	Comfort <dbl>
2	-0.01577854	-0.4249072	-0.28158545	0.50052114	-0.0989237	0.58621035
3	1.14725137	0.8552172	-0.65660558	0.16346240	-0.9192806	-0.69794311
4	-1.10904387	1.1211667	-0.05194561	-0.03015957	0.7455682	-0.74341374

4 rows

Three-Cluster Solution

Hide

```
plot(clust)
h_cluster <- cutree(clust, 3)
rect.hclust(clust, k=3, border="red")
```



Hide

```
table(h_cluster)
```

```
h_cluster
 1  2  3 
27 29 17 
```

Hide

```
hclust_summary <- aggregate(std_seg_data[,c("Trendy", "Styling", "Reliability", "Sportiness",
"Performance", "Comfort")],by=list(h_cluster),FUN=mean)
hclust_summary
```

Group.1 <int>	Trendy <dbl>	Styling <dbl>	Reliability <dbl>	Sportiness <dbl>	Performance <dbl>	Comfort <dbl>
1	-0.70539614	-0.08208834	0.7158620	-0.6405175	0.6850577	-0.1901877
2	-0.01577854	-0.42490717	-0.2815854	0.5005211	-0.0989237	0.5862104
3	1.14725137	0.85521724	-0.6566056	0.1634624	-0.9192806	-0.6979431

3 rows

This solution seems to have clusters of similar sizes. In addition, we can easily caracterize each of them. The first cluster cares about Performance and Reliability while Cluster 2 values Comfort and Sportiness. Finally, the third cluster cares about the appearance. Below, we rename those clusters according to their characteristics.

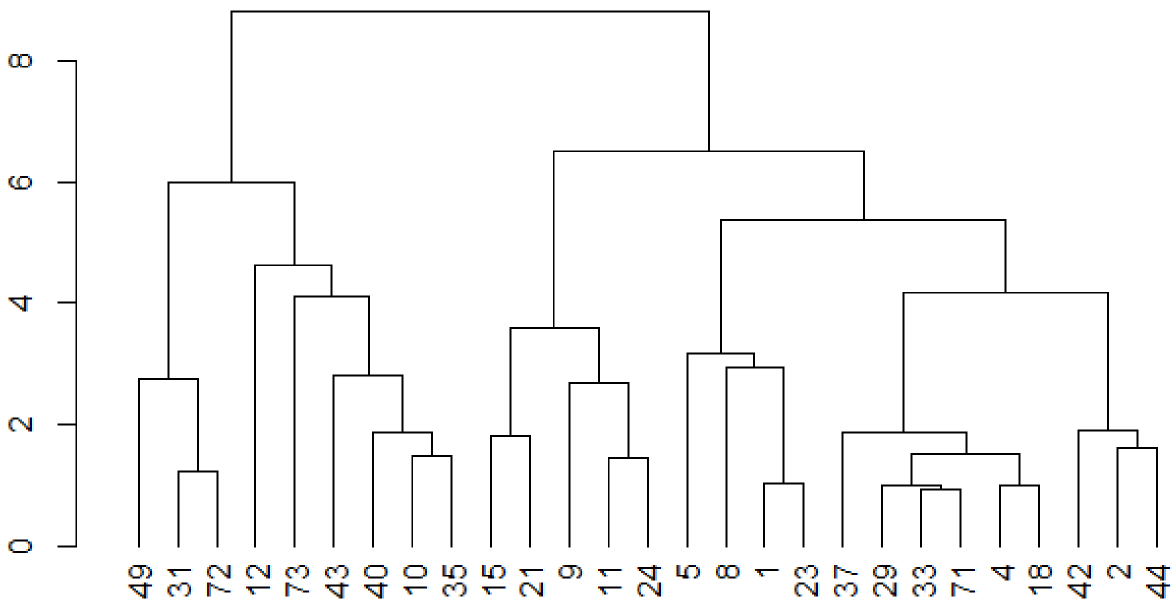
Hide

```
h_cluster <- factor(h_cluster,levels = c(1,2,3),
                    labels = c("Perf.", "Comfort", "Appearance"))
```

We can also focus on a given cluster by using the following code. Here the first one on the left:

Hide

```
plot(cut(as.dendrogram(clust), h=9)$lower[[3]])
```



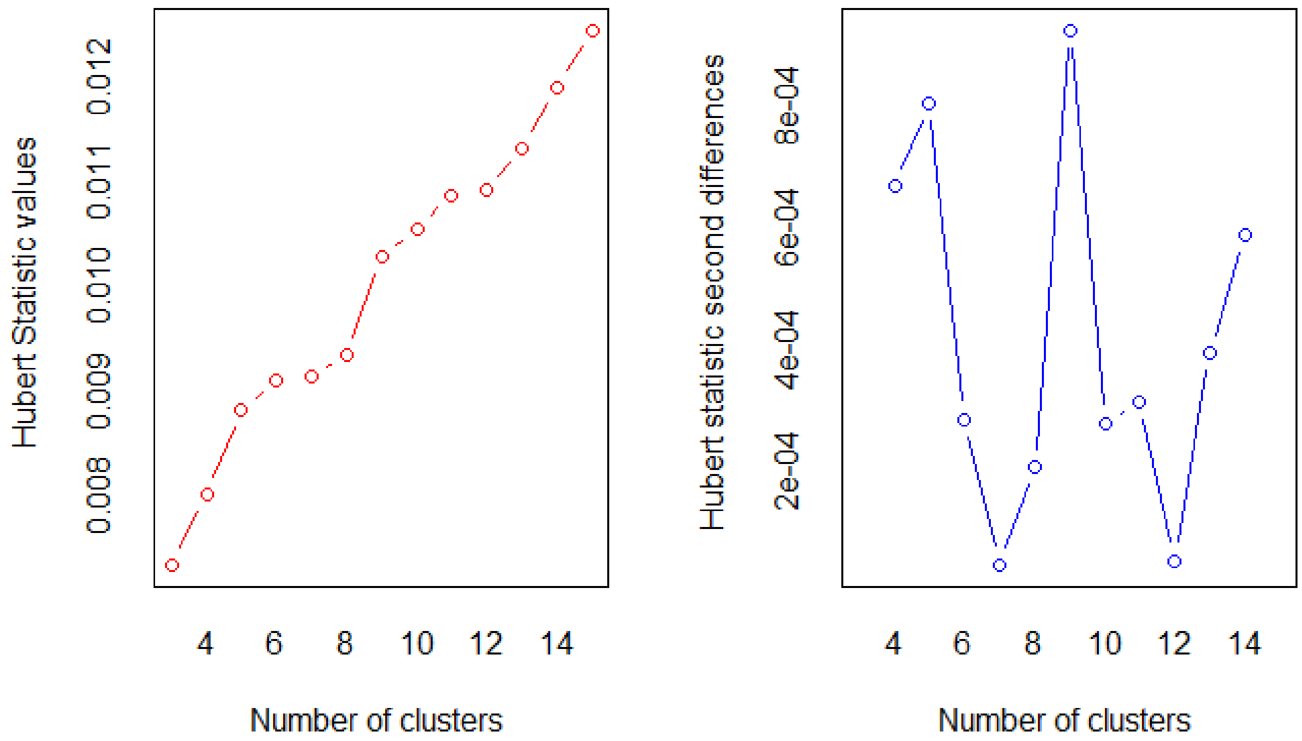
Number of Clusters

As seen above, one can use the dendogram to decide on the appropriate number of clusters. The function NbClust examines all the indexes/criteria used to determine the optimal number of clusters and outputs the optimal number based on the majority rule. Note that since it's a constant-sum allocation, we must use only 5 variables to avoid collinearity issues.

Hide

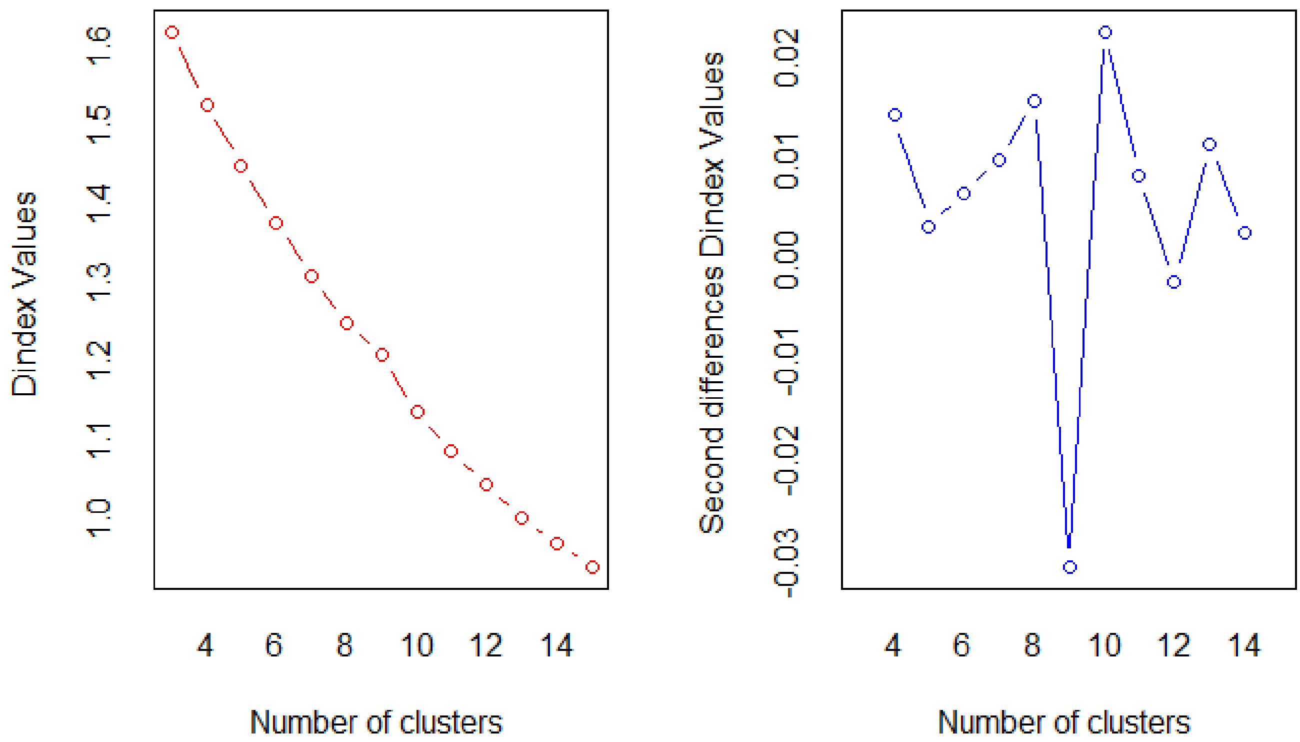
```
set.seed(1990)
NbClust(data=std_seg_data[,1:5], min.nc=3, max.nc=15, index="all", method="ward.D2")
```

```
*** : The Hubert index is a graphical method of determining the number of clusters.
                In the plot of Hubert index, we seek a significant knee that corresponds to a
                significant increase of the value of the measure i.e the significant peak in
Hubert
                index second differences plot.
```



```
*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the significant peak in D
index
                second differences plot) that corresponds to a significant increase of the va
lue of
                the measure.

*******************************************************************
* Among all indices:
* 7 proposed 3 as the best number of clusters
* 2 proposed 4 as the best number of clusters
* 2 proposed 5 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 4 proposed 9 as the best number of clusters
* 3 proposed 12 as the best number of clusters
* 1 proposed 13 as the best number of clusters
* 2 proposed 15 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  3


*******************************************************************
$All.index
       KL      CH Hartigan     CCC    Scott   Marriot    TrCovW   TraceW Friedman  Rubin Cind
ex     DB Silhouette   Duda Pseudot2   Beale
3  4.2356 19.1708   9.1557 -3.8247 125.3102 546888028 3415.0147 232.5975   5.4783 1.5477 0.40
26 1.6869     0.1987 0.6935   6.6287  1.2967
4  0.3189 17.2540   8.3534 -5.0953 159.1836 611300402 2593.8512 205.6938   7.0207 1.7502 0.38
25 1.5521     0.2111 0.7443   7.2126  1.0261
5  0.5079 16.3550   8.9849 -6.0343 202.7795 525669059 2134.0463 183.4809   8.8405 1.9621 0.35
60 1.5191     0.2206 0.5410   6.7863  2.3600
6  1.2239 16.3655   7.4554 -5.4291 240.3721 452300734 1641.4307 162.0669  10.4134 2.2213 0.44
27 1.3403     0.2335 0.3851  11.1763  4.3725
7  0.9987 16.1533   6.8924 -5.0795 272.0101 399115542 1309.7081 145.8389  11.7806 2.4685 0.44
29 1.1950     0.2414 0.6398   6.7563  1.6266
8  0.8637 16.0296   7.0554 -4.7518 305.5157 329419867 1063.7346 132.0490  13.4009 2.7263 0.42
68 1.1583     0.2445 0.7768   8.9079  0.8713
9  1.0446 16.1775   6.6456 -4.2941 344.2663 245198975  894.1088 119.1193  15.5956 3.0222 0.39
21 1.1251     0.2550 0.7288  10.7909  1.1258
10 0.9691 16.3520   6.6364 -3.8667 376.7922 193878168  720.2895 107.9138  17.7061 3.3360 0.36
30 1.2087     0.2093 0.4678   4.5515  2.8491
11 1.2274 16.6619   5.7269 -3.3850 405.2744 158805977  566.4595  97.6295  19.4921 3.6874 0.40
62 1.1481     0.2253 0.3736   5.0300  3.9357
12 1.4727 16.7917   4.4283 -3.0689 430.4837 133803651  464.2880  89.3741  21.0350 4.0280 0.47
49 1.0742     0.2312 0.7289   7.8114  1.1113
13 0.9527 16.6021   4.4291 -3.0135 462.2226 101664547  418.1911  83.3250  23.7505 4.3204 0.45
93 1.1319     0.2094 0.5973   3.3715  1.7587
14 1.0413 16.5171   4.2289 -2.9133 490.8040  79707962  370.8162  77.5969  25.9989 4.6394 0.44
58 1.1454     0.2167 0.5413   5.9328  2.3211
15 0.9571 16.4550   4.2829 -2.8227 517.1570  63774583  329.0855  72.4071  28.2284 4.9719 0.43
03 1.1209     0.2158 1.9784  -0.9891 -1.0319
   Ratkowsky    Ball Ptbiserial    Frey McClain   Dunn Hubert SDindex Dindex   SDbw
3     0.3432 77.5325     0.4553 -0.0754  1.2360 0.1677 0.0074  2.5719 1.6213 0.9774
4     0.3267 51.4234     0.4936  0.1486  1.3198 0.1677 0.0080  2.2785 1.5292 0.9946
5     0.3123 36.6962     0.5216 -0.0562  1.5473 0.1686 0.0088  2.2161 1.4514 0.6734
```

```
6     0.3017 27.0112     0.5383  0.0428  1.5724 0.2159 0.0091  1.9374 1.3769 0.5657
7     0.2912 20.8341     0.5458  0.1763  1.6005 0.2200 0.0092  1.7638 1.3091 0.4389
8     0.2811 16.5061     0.5535  0.0382  1.6962 0.2200 0.0094  1.7668 1.2513 0.4271
9     0.2725 13.2355     0.5828  0.9510  1.7837 0.2200 0.0103  1.7508 1.2091 0.4453
10    0.2644 10.7914     0.5028  0.0019  2.7486 0.1690 0.0106  2.1167 1.1366 0.4186
11    0.2573  8.8754     0.5089  0.0231  2.7520 0.1935 0.0109  2.1415 1.0868 0.3715
12    0.2502  7.4478     0.5125  1.7152  2.7587 0.2300 0.0109  2.0783 1.0454 0.3555
13    0.2429  6.4096     0.4248  0.0688  4.2473 0.2300 0.0113  2.4720 1.0019 0.3382
14    0.2365  5.5426     0.4268  0.1162  4.3133 0.2300 0.0119  2.5598 0.9698 0.3191
15    0.2305  4.8271     0.4257  0.0279  4.4504 0.2300 0.0125  2.6614 0.9406 0.3038

$All.CriticalValues
   CritValue_Duda CritValue_PseudoT2 Fvalue_Beale
3          0.4234            20.4305       0.2745
4          0.4864            22.1752       0.4062
5          0.2868            19.8896       0.0573
6          0.2552            20.4342       0.0034
7          0.3776            19.7832       0.1668
8          0.5502            25.3445       0.5019
9          0.5399            24.7090       0.3493
10         0.1164            30.3727       0.0422
11         0.0442            64.8953       0.0177
12         0.4864            22.1752       0.3589
13         0.1725            23.9899       0.1581
14         0.2552            20.4342       0.0638
15        -0.0536           -39.3104       1.0000

$Best.nc
                    KL      CH Hartigan     CCC   Scott  Marriot   TrCovW TraceW Friedman   R
ubin Cindex      DB Silhouette   Duda PseudoT2
Number_clusters 3.0000  3.0000   6.0000 15.0000  5.0000        9   4.0000 6.0000  13.0000 12.
0000  5.000 12.0000      9.000 3.0000   3.0000
Value_Index     4.2356 19.1708   1.5295 -2.8227 43.5959 32900086 821.1635 5.1859   2.7154 -0.
0482  0.356  1.0742      0.255 0.6935   6.6287
                 Beale Ratkowsky    Ball PtBiserial Frey McClain  Dunn Hubert SDindex Dindex
SDbw
Number_clusters 3.0000    3.0000  4.0000     9.0000    2   3.000 12.00      0  9.0000      0
15.0000
Value_Index     1.2967    0.3432 26.1091     0.5828   NA   1.236  0.23      0  1.7508      0
0.3038

$Best.partition
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
 1  1  1  2  1  3  2  2  1  2  1  2  2  3  1  2  3  2  3  3  1  2  1  1  3  3  2  3  2  2  3
3  2  3  3  2  1  2  2  1  2  1  2  1  3  2  2  2
49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73
 1  2  1  3  2  2  2  2  3  2  2  3  3  3  2  3  3  2  2  3  2  3  2  3  1
```



Targeting the Clusters/segments

We can now study our demographics and choice data in light of these cluster assignments using the funcion CrossTable:

Demographics

Hide

```
CrossTable(seg_data$MBA,h_cluster,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  73


             | h_cluster
seg_data$MBA |      Perf. |    Comfort | Appearance |  Row Total |
-------------|------------|------------|------------|------------|
         MBA |         14 |          6 |          4 |         24 |
             |      0.583 |      0.250 |      0.167 |      0.329 |
             |      0.519 |      0.207 |      0.235 |            |
-------------|------------|------------|------------|------------|
   Undergrad |         13 |         23 |         13 |         49 |
             |      0.265 |      0.469 |      0.265 |      0.671 |
             |      0.481 |      0.793 |      0.765 |            |
-------------|------------|------------|------------|------------|
Column Total |         27 |         29 |         17 |         73 |
             |      0.370 |      0.397 |      0.233 |            |
-------------|------------|------------|------------|------------|


Statistics for All Table Factors


Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 =  7.03013     d.f. =  2     p =  0.02974588
```

Choice

Hide

```
CrossTable(h_cluster,seg_data$Choice,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  73


             | seg_data$Choice
   h_cluster |       BMW |     Lexus |  Mercedes | Row Total |
-------------|-----------|-----------|-----------|-----------|
       Perf. |        14 |         9 |         4 |        27 |
             |     0.519 |     0.333 |     0.148 |     0.370 |
             |     0.438 |     0.409 |     0.211 |           |
-------------|-----------|-----------|-----------|-----------|
     Comfort |        10 |         8 |        11 |        29 |
             |     0.345 |     0.276 |     0.379 |     0.397 |
             |     0.312 |     0.364 |     0.579 |           |
-------------|-----------|-----------|-----------|-----------|
  Appearance |         8 |         5 |         4 |        17 |
             |     0.471 |     0.294 |     0.235 |     0.233 |
             |     0.250 |     0.227 |     0.211 |           |
-------------|-----------|-----------|-----------|-----------|
Column Total |        32 |        22 |        19 |        73 |
             |     0.438 |     0.301 |     0.260 |           |
-------------|-----------|-----------|-----------|-----------|


Statistics for All Table Factors


Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 =  4.095664     d.f. =  4     p =  0.393214
```

See lecture on how to identify variables for targeting.

K-Means

We now focus on a different method called K-Means. This method, which requires us to specify in advance the number of clusters, aims to group the observations based on their similarity using an optimization procedure. Indeed, the aim is to minimize the within-cluster variation which is defined as the sum of square of the euclidean distance between each data point to the centroid of its cluster. More precisely, the algorithm works as follow:

1. Start by assigning each point to a cluster randomly
2. Compute the centroid of each cluster and the distances of each point to each centroid
3. Reassign each observation to the closest Centroid

4. Repeat Steps 2 and 3 until the within-cluster variance is minimized

Three Cluster Solution obtained using K-Means

Let us start by observing how the algorithm works on our data for 3 segments. We use the function kmeans(). Don't forget to set the seed to a specific vaule (e.g., 1990).

Hide

```
set.seed(1990)
car_Cluster3 <-kmeans(std_seg_data, 3, iter.max=100,nstart=100)
car_Cluster3
```

```
K-means clustering with 3 clusters of sizes 18, 32, 23

Cluster means:
        Trendy    Styling Reliability Sportiness Performance     Comfort
1 -0.637247817 -0.6837159   1.1781135 -1.0328905   0.7785740  0.08642535
2 -0.003271873 -0.3788069  -0.3496669  0.4977728  -0.0445069  0.53615835
3  0.503267855  1.0621176  -0.4355087  0.1157956  -0.5473961 -0.81359668

Clustering vector:
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
 1  1  2  1  1  3  2  1  1  3  1  2  2  3  1  2  3  2  2  3  1  2  1  1  2  3  2  3  1  2  3
3  1  3  3  2  1  2  2  3  2  1  3  1  2  1  2  2
49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73
 3  2  2  3  2  2  2  2  3  2  2  2  2  3  2  3  3  2  2  3  3  3  2  3  1

Within cluster sum of squares by cluster:
[1]  81.39207  83.90060 111.49649
 (between_SS / total_SS =  35.9 %)

Available components:

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

Hide

```
Kmean_Cluster<-factor(car_Cluster3$cluster,levels = c(1,2,3),
                    labels = c("Perf. KM", "Comfort KM", "Appearance KM"))
```

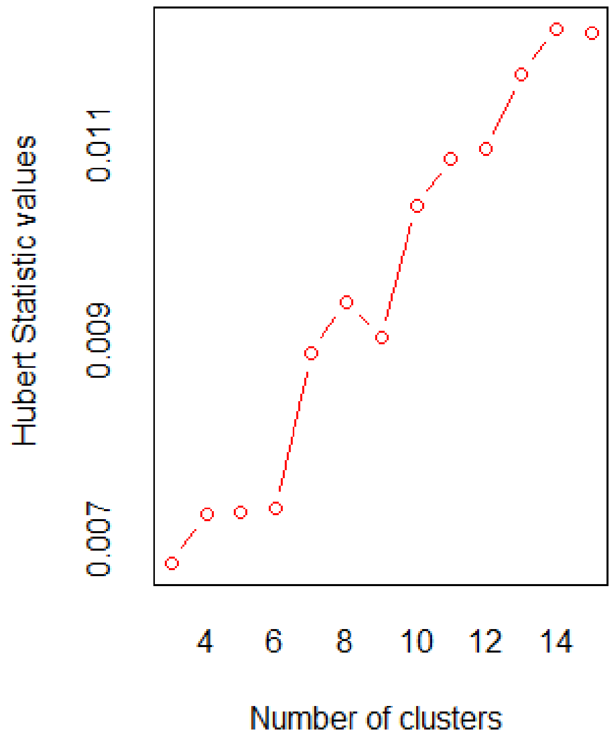
Find the optimal number of clusters

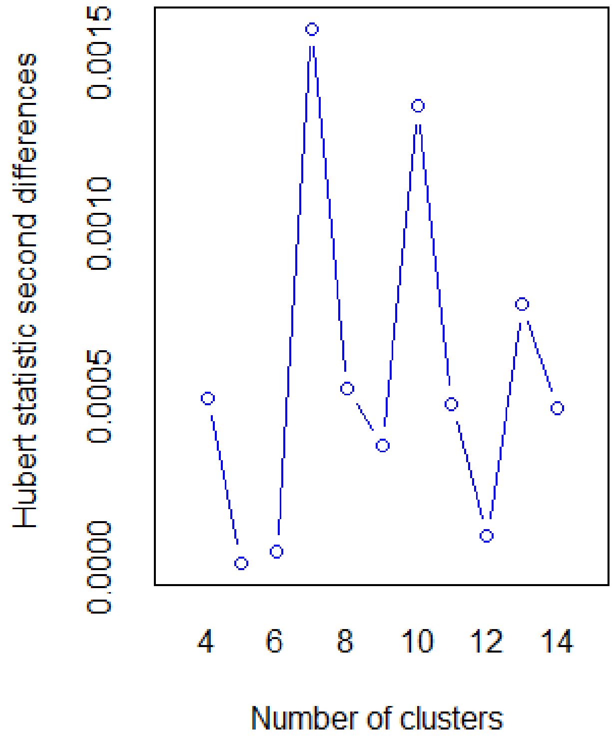
A key question when using the K-Means clustering technique consists in choosing the optimal number of segments. In order to do that, we can use the function NbClust() as in hierarchical clustering by specifying the method kmeans as below. From the output, We see that the three-cluster solution is best.

Hide

```
set.seed(1990)
NbClust(data=std_seg_data[,1:5], min.nc=3, max.nc=15, index="all", method="kmeans")
```

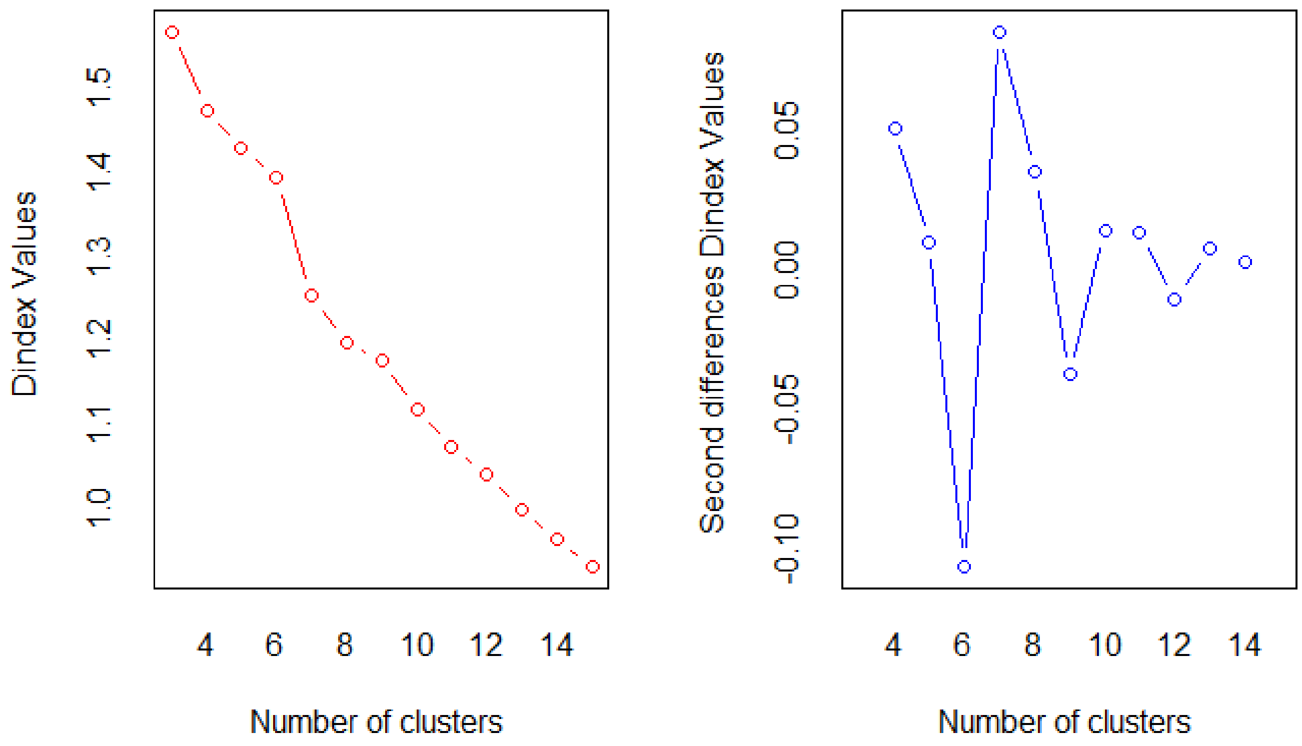
```
*** : The Hubert index is a graphical method of determining the number of clusters.
                In the plot of Hubert index, we seek a significant knee that corresponds to a
                significant increase of the value of the measure i.e the significant peak in
Hubert
                index second differences plot.
```





```
*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the significant peak in D
index
                second differences plot) that corresponds to a significant increase of the va
lue of
                the measure.

*******************************************************************
* Among all indices:
* 7 proposed 3 as the best number of clusters
* 2 proposed 4 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 3 proposed 7 as the best number of clusters
* 5 proposed 8 as the best number of clusters
* 1 proposed 10 as the best number of clusters
* 1 proposed 11 as the best number of clusters
* 1 proposed 12 as the best number of clusters
* 2 proposed 15 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  3


*******************************************************************
$All.index
         KL      CH Hartigan     CCC    Scott   Marriot    TrCovW   TraceW Friedman  Rubin Ci
ndex     DB Silhouette   Duda Pseudot2   Beale
3    2.6615 23.0027   9.9985 -2.3384 161.6503 332431565 3142.6116 217.2313   6.9183 1.6572 0.
4029 1.6368     0.2211 0.7619   6.8769  0.9339
4    2.6211 20.5604   5.9101 -3.4050 206.2171 320952678 2494.7351 190.0809   8.7864 1.8939 0.
3875 1.4764     0.2351 1.4900  -4.9326 -0.9006
5    0.3090 17.9546   3.5125 -5.0208 211.9276 463754305 1884.0863 175.0844   9.3054 2.0562 0.
3747 1.3972     0.2146 1.6044  -3.7672 -0.9825
6    0.1667 15.5757  15.2717 -5.9661 250.0332 396233308 1841.2783 166.4846  11.9435 2.1624 0.
3825 1.3828     0.1678 1.0555  -0.7881 -0.1542
7    2.8226 18.2077   7.6528 -3.7082 297.2619 282403511 1117.8685 135.5809  13.1919 2.6552 0.
3826 1.2398     0.2296 2.1023  -5.2433 -1.2308
8  115.6198 18.2290   3.1015 -3.2577 325.0955 251921280  850.0973 121.4935  14.7248 2.9631 0.
4170 1.1432     0.2537 1.9135  -9.0706 -1.2807
9    0.0103 16.8361   6.1543 -3.8303 341.9554 253084884  780.5087 115.9604  15.5678 3.1045 0.
4631 1.2340     0.1908 2.2632  -7.8141 -1.3975
10   0.7749 16.8213   7.0516 -3.5342 370.8399 210349050  629.2076 105.7877  17.4933 3.4030 0.
3716 1.2501     0.2030 0.6031   2.6329  1.7167
11   5.0114 17.2605   3.1097 -2.9648 406.5994 155949665  518.0238  95.1388  19.4733 3.7839 0.
4513 1.1235     0.2118 2.0940  -8.3592 -1.3081
12   0.2795 16.4907   5.3651 -3.2843 446.4373 107536543  493.6384  90.5948  24.4889 3.9737 0.
4432 1.1876     0.2075 0.9828   0.1929  0.0507
13   1.0804 16.6162   5.0360 -3.0034 473.8984  86637820  423.0730  83.2710  26.1131 4.3232 0.
4365 1.1268     0.2093 1.4350  -3.6377 -0.8433
14   1.9907 16.7292   3.2865 -2.7603 487.2320  83705207  340.7177  76.8230  26.1213 4.6861 0.
4219 1.1091     0.2127 1.3559  -1.3124 -0.5477
15   1.0396 16.3524   3.1017 -2.8976 510.4032  69956428  321.4077  72.7695  27.6550 4.9471 0.
4171 1.0293     0.2340 9.3984  -5.3616  0.0000
   Ratkowsky    Ball Ptbiserial     Frey McClain   Dunn Hubert SDindex Dindex   SDbw
3     0.3612 72.4104     0.4486   0.1157   1.2726 0.1677 0.0068   2.4006 1.5698 0.9262
4     0.3427 47.5202     0.4837   1.0258   1.5254 0.1342 0.0073   2.1293 1.4750 0.7671
```

```
5     0.3201 35.0169     0.4512  -1.0806  2.0027 0.1651 0.0073  2.0377 1.4305 0.6090
6     0.2949 27.7474     0.3805  -0.2084  2.5868 0.0905 0.0073  1.9698 1.3957 0.5129
7     0.2981 19.3687     0.4840  -0.2534  2.2978 0.1520 0.0089  1.8817 1.2548 0.4441
8     0.2876 15.1867     0.5323  -5.0634  2.1224 0.2019 0.0094  1.8598 1.1989 0.4061
9     0.2744 12.8845     0.3980   0.0564  3.8572 0.1324 0.0090  2.0189 1.1776 0.3576
10    0.2657 10.5788     0.4099  -0.3349  4.1242 0.1690 0.0104  2.2013 1.1187 0.3598
11    0.2586  8.6490     0.4576 -23.5430  3.4951 0.1321 0.0109  2.1409 1.0738 0.3560
12    0.2496  7.5496     0.4283   0.5173  4.0171 0.1460 0.0110  2.5475 1.0423 0.3064
13    0.2430  6.4055     0.4021   0.0993  4.7440 0.1487 0.0117  2.8269 0.9999 0.3015
14    0.2370  5.4874     0.4017  -0.2287  4.8915 0.1309 0.0122  2.6940 0.9648 0.2900
15    0.2305  4.8513     0.4087   0.6242  4.7589 0.1309 0.0121  2.6062 0.9323 0.2517

$All.CriticalValues
   CritValue_Duda CritValue_PseudoT2 Fvalue_Beale
3          0.4864            23.2312       0.4623
4          0.2552            43.7876       1.0000
5          0.1725            47.9798       1.0000
6          0.4234            20.4305       1.0000
7          0.0442           216.3177       1.0000
8          0.2177            68.2773       1.0000
9          0.1164           106.3046       1.0000
10         0.1725            19.1919       0.1675
11         0.1164           121.4910       1.0000
12         0.3776            18.1346       0.9983
13         0.2868            29.8345       1.0000
14        -0.0536           -98.2760       1.0000
15        -0.4373           -19.7211          NaN

$Best.nc
                     KL      CH Hartigan     CCC   Scott   Marriot   TrCovW  TraceW Friedman
Rubin  Cindex      DB Silhouette   Duda PseudoT2
Number_clusters   8.0000  3.0000   6.0000  3.0000  7.0000         4   7.0000  7.0000  12.0000
11.0000 10.0000 15.0000     8.0000 3.0000   3.0000
Value_Index     115.6198 23.0027  11.7591 -2.3384 47.2286 154280514 723.4098 16.8164   5.0156
-0.1911  0.3716  1.0293     0.2537 0.7619   6.8769
                 Beale Ratkowsky    Ball PtBiserial Frey McClain    Dunn Hubert SDindex Dindex
SDbw
Number_clusters 3.0000    3.0000  4.0000     8.0000    2  3.0000 8.0000      0  8.0000      0
15.0000
Value_Index     0.9339    0.3612 24.8902     0.5323   NA  1.2726 0.2019      0  1.8598      0
0.2517

$Best.partition
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
 2  2  1  2  2  1  3  2  2  3  2  3  3  1  2  3  1  3  3  1  2  3  2  2  3  1  3  1  2  3  1
1  2  1  1  3  2  3  3  3  3  2  3  2  1  2  3  3
49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73
 1  3  3  1  3  3  3  3  1  3  3  1  1  1  3  1  1  3  3  1  3  1  3  1  2
```



Results Comparison

Looking at the cluster means above, we see that the clusters defined with the kmeans function are characterized similarly as before. Thus, we relabel them to describe them more accurately. We can now compare this clustering to the demographics and choice as well as the hierarchical clustering.

Demographics

Hide

```
CrossTable(seg_data$MBA,Kmean_Cluster,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  73


             | Kmean_Cluster 
seg_data$MBA |      Perf. KM |    Comfort KM | Appearance KM |     Row Total | 
-------------|---------------|---------------|---------------|---------------|
         MBA |            11 |             8 |             5 |            24 | 
             |         0.458 |         0.333 |         0.208 |         0.329 | 
             |         0.611 |         0.250 |         0.217 |               | 
-------------|---------------|---------------|---------------|---------------|
   Undergrad |             7 |            24 |            18 |            49 | 
             |         0.143 |         0.490 |         0.367 |         0.671 | 
             |         0.389 |         0.750 |         0.783 |               | 
-------------|---------------|---------------|---------------|---------------|
Column Total |            18 |            32 |            23 |            73 | 
             |         0.247 |         0.438 |         0.315 |               | 
-------------|---------------|---------------|---------------|---------------|


Statistics for All Table Factors


Pearson's Chi-squared test 
------------------------------------------------------------
Chi^2 =  8.694824     d.f. =  2     p =  0.01294026 
```

Choice

Hide

```
CrossTable(Kmean_Cluster,seg_data$Choice,prop.chisq = FALSE, prop.r = T, prop.c = T,prop.t = F,chisq = T)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  73


              | seg_data$Choice
Kmean_Cluster |       BMW |     Lexus |  Mercedes | Row Total |
--------------|-----------|-----------|-----------|-----------|
     Perf. KM |         7 |         8 |         3 |        18 |
              |     0.389 |     0.444 |     0.167 |     0.247 |
              |     0.219 |     0.364 |     0.158 |           |
--------------|-----------|-----------|-----------|-----------|
   Comfort KM |        14 |         8 |        10 |        32 |
              |     0.438 |     0.250 |     0.312 |     0.438 |
              |     0.438 |     0.364 |     0.526 |           |
--------------|-----------|-----------|-----------|-----------|
Appearance KM |        11 |         6 |         6 |        23 |
              |     0.478 |     0.261 |     0.261 |     0.315 |
              |     0.344 |     0.273 |     0.316 |           |
--------------|-----------|-----------|-----------|-----------|
 Column Total |        32 |        22 |        19 |        73 |
              |     0.438 |     0.301 |     0.260 |           |
--------------|-----------|-----------|-----------|-----------|


Statistics for All Table Factors


Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 =  2.753465     d.f. =  4     p =  0.5998918
```

Hierarchical Clustering

Hide

```
CrossTable(h_cluster,Kmean_Cluster,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

Latent Class Analysis

Latent Class Analysis is a method to identify cluster membership of subjects using the observable variables that describe them. The approach consists in estimating for each individual the probability to belong to a “latent class” or cluster. In turn, each cluster is defined in terms of its “geometry” and “orientation” as cloud of points.

As such, this technique belongs to the family of gaussian finite mixture models. This approach relies on a different optimization procedure that aims to maximize the likelihood (versus minimize the distances between each point). Hence, the tools to assess the optimal number of classes differ. We now perform this analysis using the package mclust. We start by determining the optimal model based on BIC using the function mclustBIC().

Find the optimal model

Hide

```
set.seed(1990)
mclustBIC(std_seg_data[,1:5],verbose=F)
```

Hence, the optimal model is VEE with 2 segments. This means that the data can be clustered in two clusters which will both be modeled by a Normal distribution with the same covariance matrix. We obtain more details about the optimal model below:

Hide

```
set.seed(1990)
lca_clust <- Mclust(std_seg_data[,1:5],verbose = FALSE)
summary(lca_clust)
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VEE (ellipsoidal, equal shape and orientation) model with 2 components:

 log.likelihood  n df       BIC       ICL
      -431.8862 73 27 -979.6149 -990.9626

Clustering table:
 1  2
47 26
```

We now interpret each cluster and rename them to describe them accurately:

Hide

```
lca_clusters <- lca_clust$classification
lca_clust_summary <- aggregate(std_seg_data[,c("Trendy", "Styling", "Reliability", "Sportines
s", "Performance", "Comfort")],by=list(lca_clusters),FUN=mean)
lca_clust_summary
lca_clusters<-factor(lca_clusters,levels = c(1,2),
                    labels = c("Reliability LCA", "Comfort LCA"))
```

Results Comparison

Let us compare this solution to the demographics and choice data as well as the hierarchical clustering and K-Means:

Demographics

Hide

```
CrossTable(seg_data$MBA,lca_clusters,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

Choice

Hide

```
CrossTable(lca_clusters,seg_data$Choice,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

Hierarchical Clustering

Hide

```
CrossTable(h_cluster,lca_clusters,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```

K-Means Clustering

Hide

```
CrossTable(Kmean_Cluster,lca_clusters,prop.chisq = FALSE, prop.r = T, prop.c = T,
           prop.t = F,chisq = T)
```