Clustering Models

Dr. Jeffrey Strickland

9/2/2018

Hierarchical Clustering

First load the package.

votes.repub is a data frame with the percent of votes given to the republican candidate in presidential elections from 1856 to 1976. Rows represent the 50 states, and columns the 31 elections.

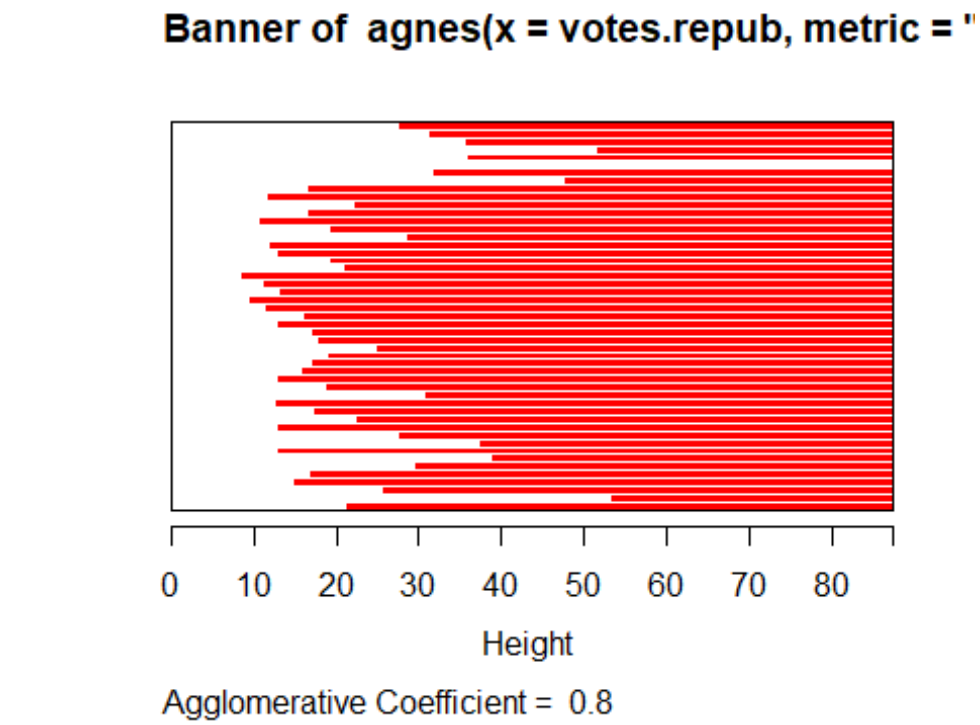
Agglomerative Nesting (Hierarchical Clustering)

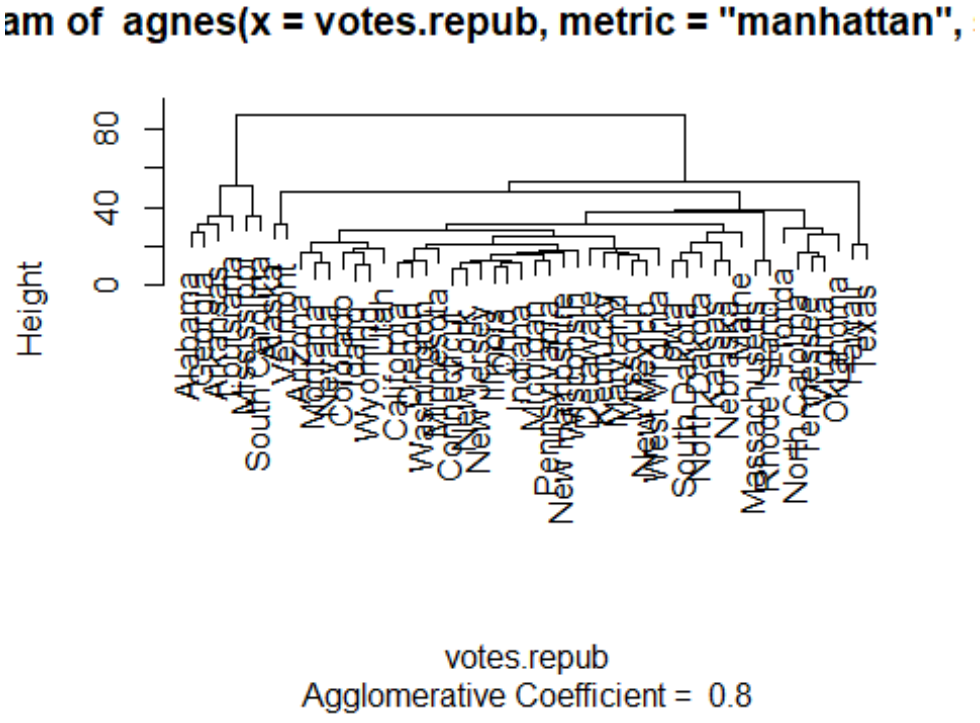
Here, we are using the agnes function computes agglomerative hierarchical clustering of the dataset.

```
library(cluster)
data(votes.repub)
agn1 <- agnes(votes.repub, metric = "manhattan", stand = TRUE)
agn1
```

```
## Call:     agnes(x = votes.repub, metric = "manhattan", stand = TRUE)
## Agglomerative coefficient:  0.7977555
## Order of objects:
##  [1] Alabama        Georgia        Arkansas       Louisiana
##  [5] Mississippi    South Carolina Alaska         Vermont
##  [9] Arizona        Montana        Nevada         Colorado
## [13] Idaho          Wyoming        Utah           California
## [17] Oregon         Washington     Minnesota      Connecticut
## [21] New York       New Jersey     Illinois       Ohio
## [25] Indiana        Michigan       Pennsylvania   New Hampshire
## [29] Wisconsin      Delaware       Kentucky       Maryland
## [33] Missouri       New Mexico     West Virginia  Iowa
## [37] South Dakota   North Dakota   Kansas         Nebraska
## [41] Maine          Massachusetts  Rhode Island   Florida
## [45] North Carolina Tennessee      Virginia       Oklahoma
## [49] Hawaii         Texas
## Height (summary):
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   8.382  12.804  18.528  23.118  28.411  87.455
##
## Available components:
## [1] "order"     "height"    "ac"        "merge"     "diss"      "call"
## [7] "method"    "order.lab" "data"
```

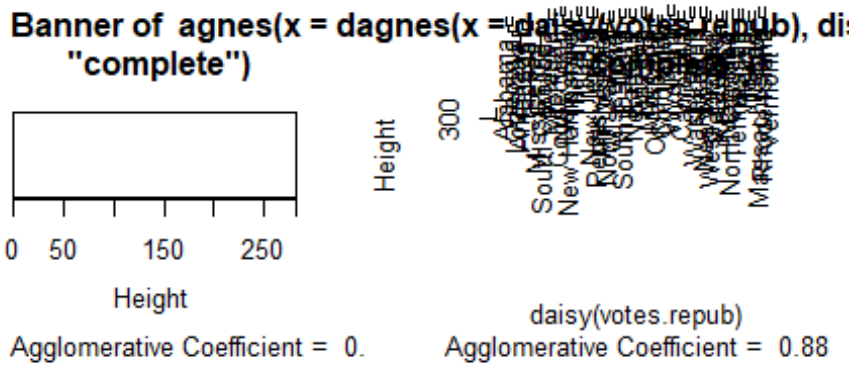
```
plot(agn1)
```


Banner of agnes(x = votes.repub, metric = "
0
10
20
30
40
50
60
70
80
Height
Agglomerative Coefficient = 0.8


am of agnes(x = votes.repub, metric = "manhattan",
Height
0
40
80
votes.repub
Agglomerative Coefficient = 0.8

Plot the Data

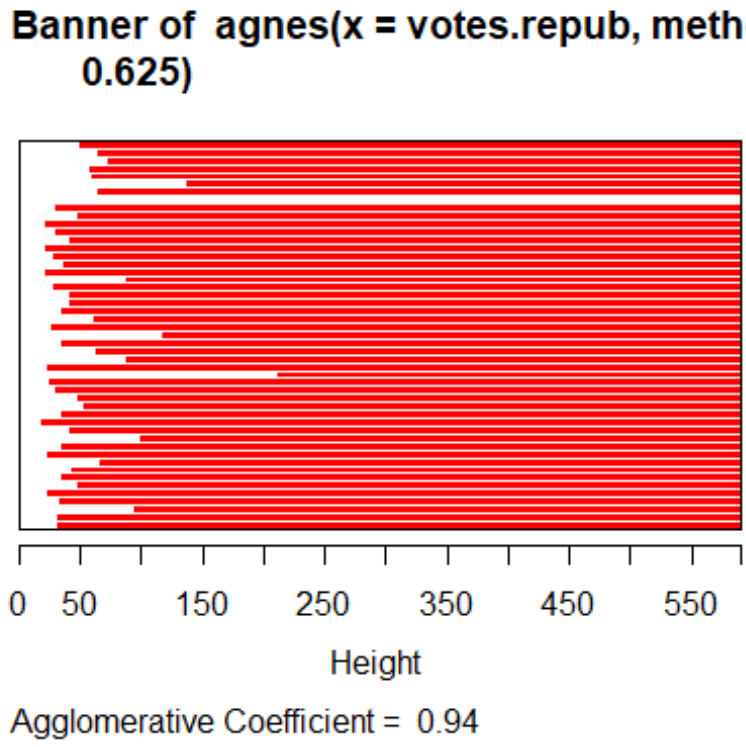
```
op <- par(mfrow=c(2,2))
agn2 <- agnes(daisy(votes.repub), diss = TRUE, method = "complete")
plot(agn2)
```

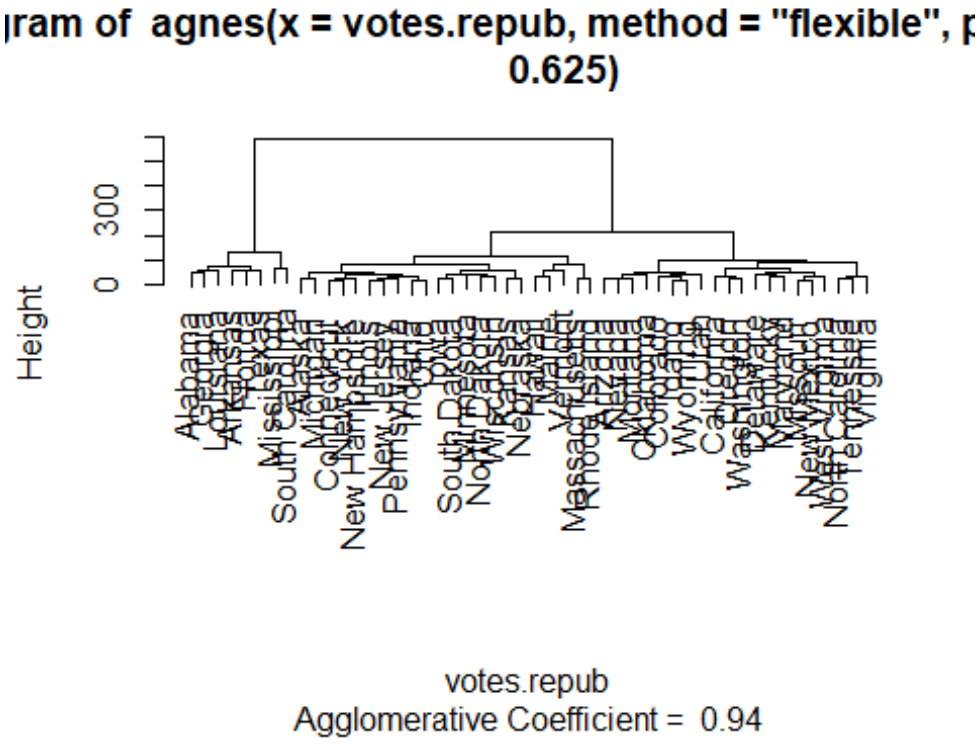


Plot with Parameters

alpha = 0.625 ==beta = -1/4 is “recommended” by some

```
agnS <- agnes(votes.repub, method = "flexible", par.meth = 0.625)
plot(agnS)
```


Banner of agnes(x = votes.repub, method =
0.625)
0
50
150
250
350
450
550
Height
Agglomerative Coefficient = 0.94


ıram of agnes(x = votes.repub, method = "flexible", ı
0.625)
Height
0
300
votes.repub
Agglomerative Coefficient = 0.94

“show” equivalence of three “flexible” special cases

Daisy computes all the pairwise dissimilarities (distances) between observations in the data set. The original variables may be of mixed types. In that case, or whenever metric = “gower” is set, a generalization of Gower’s formula is used.

```
d.vr <- daisy(votes.repub)
a.wgt <- agnes(d.vr, method = "weighted")
a.sing <- agnes(d.vr, method = "single")
a.comp <- agnes(d.vr, method = "complete")
iC <- -(6:7) # not using 'call' and 'method' for comparisons
stopifnot(all.equal(a.wgt [iC], agnes(d.vr, method="flexible",par.method =
0.5)[iC]), all.equal(a.sing[iC], agnes(d.vr, method="flex", par.method=
c(.5,.5,0, -.5))[iC]), all.equal(a.comp[iC], agnes(d.vr, method="flex",
par.method= c(.5,.5,0, +.5))[iC]))
```

The class “dendrogram” provides general functions for handling tree-like structures. It is intended as a replacement for similar functions in hierarchical clustering and classification/regression trees, such that all of these can use the same engine for plotting or cutting trees.

Two main branches ‘dendrogram’ with 2 branches and 50 members total, at height 281.9508. The first branch ‘dendrogram’ with 2 branches and 8 members total, at height 116.7048. The 2nd one { 8 + 42 = 50 } ‘dendrogram’ with 2 branches and 42 members total, at height 178.4119. The first sub-branch of branch 1 .. and shorter form ‘dendrogram’ with 2 branches and 6 members total, at height 72.92212

```
(d2 <- as.dendrogram(agn2))

## 'dendrogram' with 2 branches and 50 members total, at height 281.9508

d2[[1]] #

## 'dendrogram' with 2 branches and 8 members total, at height 116.7048

d2[[2]] #

## 'dendrogram' with 2 branches and 42 members total, at height 178.4119

d2[[1]][[1]]#

## 'dendrogram' with 2 branches and 6 members total, at height 72.92212

identical(d2[[c(1,1)]], d2[[1]][[1]])

## [1] TRUE
```

A “textual picture” of the dendrogram

```
str(d2)

## --[dendrogram w/ 2 branches and 50 members at h = 282]
##   |--[dendrogram w/ 2 branches and 8 members at h = 117]
```

```
##    |  |--[dendrogram w/ 2 branches and 6 members at h = 72.9]
##    |  |  |--[dendrogram w/ 2 branches and 3 members at h = 60.9]
##    |  |  |  |--[dendrogram w/ 2 branches and 2 members at h = 48.2]
##    |  |  |  |  |--leaf "Alabama"
##    |  |  |  |  `--leaf "Georgia"
##    |  |  |  `--leaf "Louisiana"
##    |  |  `--[dendrogram w/ 2 branches and 3 members at h = 58.8]
##    |  |     |--[dendrogram w/ 2 branches and 2 members at h = 56.1]
##    |  |     |  |--leaf "Arkansas"
##    |  |     |  `--leaf "Florida"
##    |  |     `--leaf "Texas"
##    |  `--[dendrogram w/ 2 branches and 2 members at h = 63.1]
##    |     |--leaf "Mississippi"
##    |     `--leaf "South Carolina"
##    `--[dendrogram w/ 2 branches and 42 members at h = 178]
##       |--[dendrogram w/ 2 branches and 37 members at h = 121]
##       |  |--[dendrogram w/ 2 branches and 31 members at h = 80.5]
##       |  |  |--[dendrogram w/ 2 branches and 17 members at h = 64.5]
##       |  |  |  |--[dendrogram w/ 2 branches and 13 members at h = 56.4]
##       |  |  |  |  |--[dendrogram w/ 2 branches and 10 members at h = 47.2]
##       |  |  |  |  |  |--[dendrogram w/ 2 branches and 2 members at h = 28.1]
##       |  |  |  |  |  |  |--leaf "Alaska"
##       |  |  |  |  |  |  `--leaf "Michigan"
##       |  |  |  |  |  `--[dendrogram w/ 2 branches and 8 members at h = 39.2]
##       |  |  |  |  |     |--[dendrogram w/ 2 branches and 5 members at h = 36.8]
##       |  |  |  |  |     |  |--[dendrogram w/ 2 branches and 3 members at h = 32.9]
##       |  |  |  |  |     |  |  |--[dendrogram w/ 2 branches and 2 members at h=19.4]
##       |  |  |  |  |     |  |  |  |--leaf "Connecticut"
##       |  |  |  |  |     |  |  |  `--leaf "New York"
##       |  |  |  |  |     |  |  `--leaf "New Hampshire"
##       |  |  |  |  |     |  `--[dendrogram w/ 2 branches and 2 members at h = 20.2]
##       |  |  |  |  |     |     |--leaf "Indiana"
##       |  |  |  |  |     |     `--leaf "Ohio"
##       |  |  |  |  |     `--[dendrogram w/ 2 branches and 3 members at h = 25.3]
##       |  |  |  |  |        |--[dendrogram w/ 2 branches and 2 members at h = 20.9]
##       |  |  |  |  |        |  |--leaf "Illinois"
##       |  |  |  |  |        |  `--leaf "New Jersey"
##       |  |  |  |  |        `--leaf "Pennsylvania"
##       |  |  |  |  `--[dendrogram w/ 2 branches and 3 members at h = 42.2]
##       |  |  |  |     |--leaf "Minnesota"
##       |  |  |  |     `--[dendrogram w/ 2 branches and 2 members at h = 33.7]
##       |  |  |  |        |--leaf "North Dakota"
##       |  |  |  |        `--leaf "Wisconsin"
##       |  |  |  `--[dendrogram w/ 2 branches and 4 members at h = 37.5]
##       |  |  |     |--[dendrogram w/ 2 branches and 2 members at h = 26.2]
##       |  |  |     |  |--leaf "Iowa"
##       |  |  |     |  `--leaf "South Dakota"
##       |  |  |     `--[dendrogram w/ 2 branches and 2 members at h = 25.9]
##       |  |  |        |--leaf "Kansas"
##       |  |  |        `--leaf "Nebraska"
##       |  |  `--[dendrogram w/ 2 branches and 14 members at h = 70.5]
##       |  |     |--[dendrogram w/ 2 branches and 8 members at h = 48]
##       |  |     |  |--[dendrogram w/ 2 branches and 4 members at h = 43.4]
##       |  |     |  |  |--[dendrogram w/ 2 branches and 3 members at h = 27.8]
##       |  |     |  |  |  |--[dendrogram w/ 2 branches and 2 members at h = 23.4]
```

```
##      |  |     |  |  |  |  |--leaf "Arizona"
##      |  |     |  |  |  |  `--leaf "Nevada"
##      |  |     |  |  |  `--leaf "Montana"
##      |  |     |  |  `--leaf "Oklahoma"
##      |  |     |  `--[dendrogram w/ 2 branches and 4 members at h = 43.7]
##      |  |     |     |--leaf "Colorado"
##      |  |     |     `--[dendrogram w/ 2 branches and 3 members at h = 31.2]
##      |  |     |        |--[dendrogram w/ 2 branches and 2 members at h = 17.2]
##      |  |     |        |  |--leaf "Idaho"
##      |  |     |        |  `--leaf "Wyoming"
##      |  |     |        `--leaf "Utah"
##      |  |     `--[dendrogram w/ 2 branches and 6 members at h = 54.3]
##      |  |        |--[dendrogram w/ 2 branches and 3 members at h = 33.2]
##      |  |        |  |--leaf "California"
##      |  |        |  `--[dendrogram w/ 2 branches and 2 members at h = 22.2]
##      |  |        |     |--leaf "Oregon"
##      |  |        |     `--leaf "Washington"
##      |  |        `--[dendrogram w/ 2 branches and 3 members at h = 35.1]
##      |  |           |--[dendrogram w/ 2 branches and 2 members at h = 21.1]
##      |  |           |  |--leaf "Missouri"
##      |  |           |  `--leaf "New Mexico"
##      |  |           `--leaf "West Virginia"
##      |  `--[dendrogram w/ 2 branches and 6 members at h = 66.8]
##      |     |--[dendrogram w/ 2 branches and 3 members at h = 43.4]
##      |     |  |--leaf "Delaware"
##      |     |  `--[dendrogram w/ 2 branches and 2 members at h = 33.5]
##      |     |     |--leaf "Kentucky"
##      |     |     `--leaf "Maryland"
##      |     `--[dendrogram w/ 2 branches and 3 members at h = 30.2]
##      |        |--[dendrogram w/ 2 branches and 2 members at h = 29.5]
##      |        |  |--leaf "North Carolina"
##      |        |  `--leaf "Tennessee"
##      |        `--leaf "Virginia"
##      `--[dendrogram w/ 2 branches and 5 members at h = 83.1]
##         |--[dendrogram w/ 2 branches and 4 members at h = 55.4]
##         |  |--[dendrogram w/ 2 branches and 2 members at h = 32.8]
##         |  |  |--leaf "Hawaii"
##         |  |  `--leaf "Maine"
##         |  `--[dendrogram w/ 2 branches and 2 members at h = 22.6]
##         |     |--leaf "Massachusetts"
##         |     `--leaf "Rhode Island"
##         `--leaf "Vermont"
```

Multiple Plots

```
plot(agnes(agriculture), ask = TRUE)
data(animals)
aa.a <- agnes(animals) # default method = "average"
aa.ga <- agnes(animals, method = "gaverage")
op <- par(mfcol=1:2, mgp=c(1.5, 0.6, 0), mar=c(.1+ c(4,3,2,1)),cex.main=0.8)
plot(aa.a, which.plot = 2)
```

k-Means Clustering

The wine dataset contains the results of a chemical analysis of wines grown in a specific area of Italy. Three types of wine are represented in the 178 samples, with the results of 13 chemical analyses recorded for each sample. The Type variable has been transformed into a categoric variable.

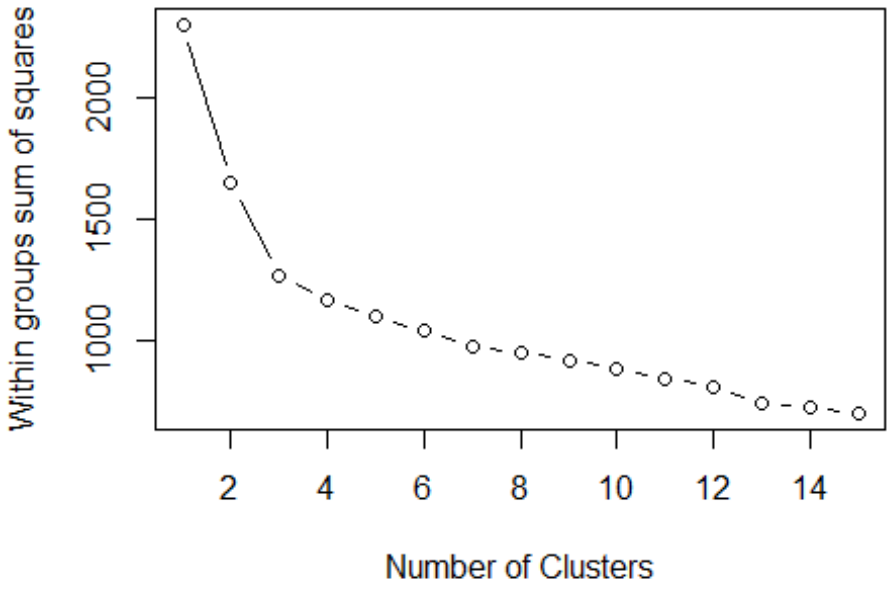
Load and View the Data

```
library(rattle)

## Rattle: A free graphical interface for data science with R.
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(rattle.data)
data(wine)
head(wine)

##   Type Alcohol Malic  Ash Alcalinity Magnesium Phenols Flavanoids
## 1    1   14.23  1.71 2.43       15.6       127    2.80       3.06
## 2    1   13.20  1.78 2.14       11.2       100    2.65       2.76
## 3    1   13.16  2.36 2.67       18.6       101    2.80       3.24
## 4    1   14.37  1.95 2.50       16.8       113    3.85       3.49
## 5    1   13.24  2.59 2.87       21.0       118    2.80       2.69
## 6    1   14.20  1.76 2.45       15.2       112    3.27       3.39
##   Nonflavanoids Proanthocyanins Color  Hue Dilution Proline
## 1          0.28            2.29  5.64 1.04     3.92    1065
## 2          0.26            1.28  4.38 1.05     3.40    1050
## 3          0.30            2.81  5.68 1.03     3.17    1185
## 4          0.24            2.18  7.80 0.86     3.45    1480
## 5          0.39            1.82  4.32 1.04     2.93     735
## 6          0.34            1.97  6.75 1.05     2.85    1450
```

A fundamental question is how to determine the value of the parameter k. If we looks at the percentage of variance explained as a function of the number of clusters, one should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the "elbow criterion".

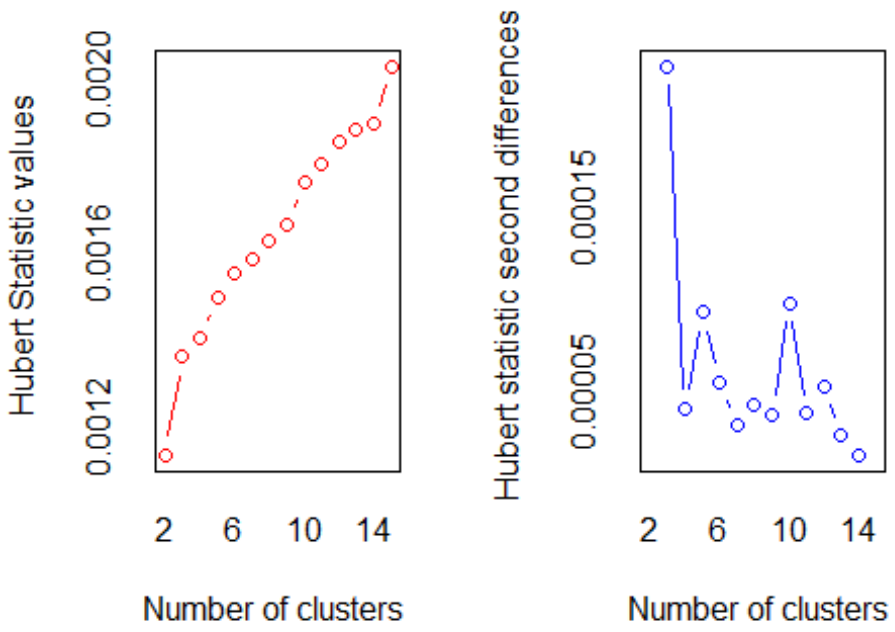
```
df <- scale(wine[-1])
wssplot <- function(data, nc=15, seed=1234){
               wss <- (nrow(data)-1)*sum(apply(data,2,var))
               for (i in 2:nc){
                    set.seed(seed)
                    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
                plot(1:nc, wss, type="b", xlab="Number of Clusters",
```

```
                     ylab="Within groups sum of squares")}
wssplot(df)
```

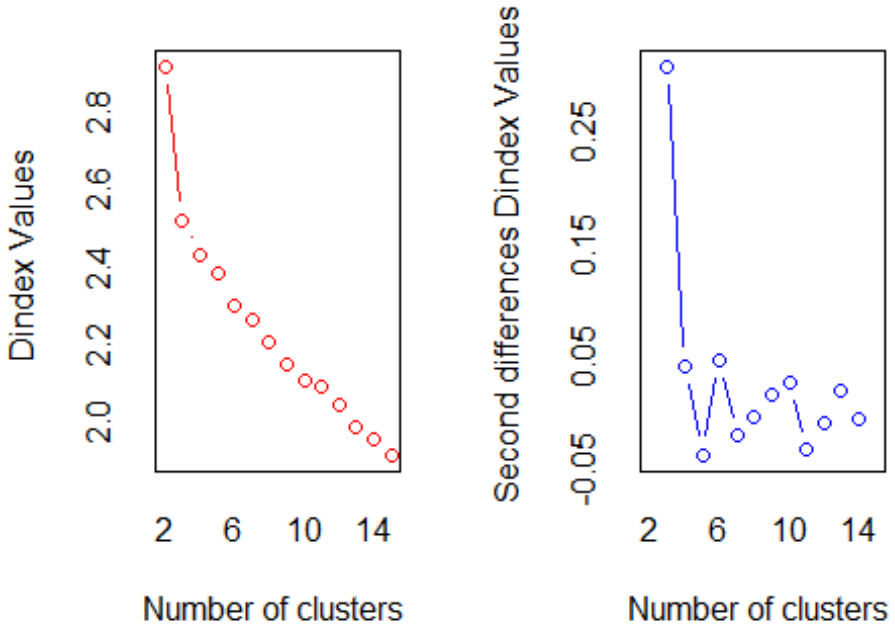


NbClust package provides 30 indices for determining the number of clusters and proposes to user the best clustering scheme from the different results obtained by varying all combinations of number of clusters, distance measures, and clustering methods.

```
library(NbClust)
set.seed(1234)
nc <- NbClust(df, min.nc=2, max.nc=15, 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.
##
```

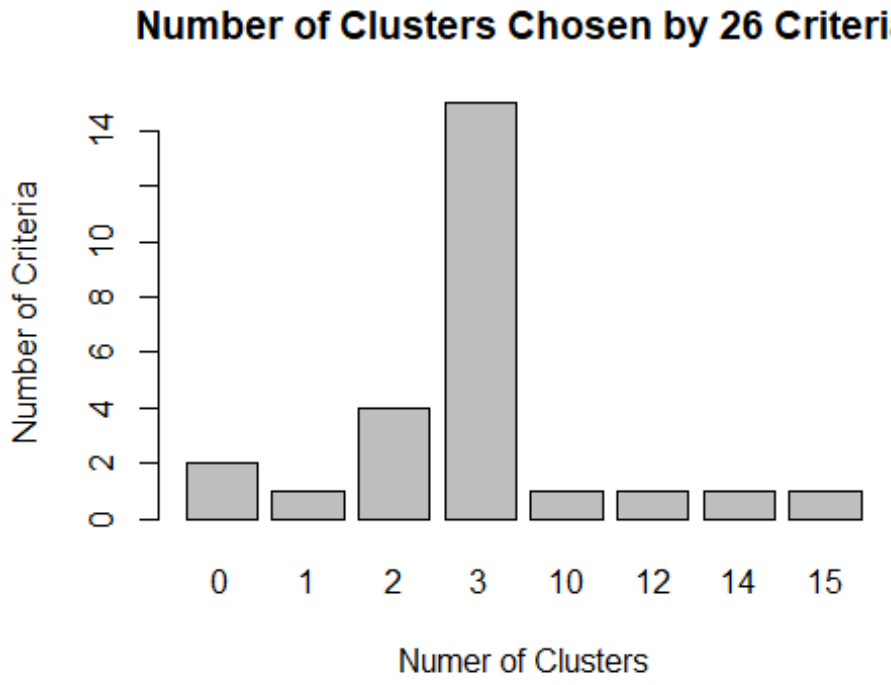



```
## *** : The D index is a graphical method of determining the number of
clusters.
##                 In the plot of D index, we seek a significant knee (the
significant peak in Dindex
##                 second differences plot) that corresponds to a significant
increase of the value of
##                 the measure.
##
## *****************************************************************
## * Among all indices:
## * 4 proposed 2 as the best number of clusters
## * 15 proposed 3 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
## * 1 proposed 12 as the best number of clusters
## * 1 proposed 14 as the best number of clusters
## * 1 proposed 15 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *****************************************************************
```

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 Dindex second differences plot) that corresponds to a significant increase of the value of the measure. All 178 observations were used.

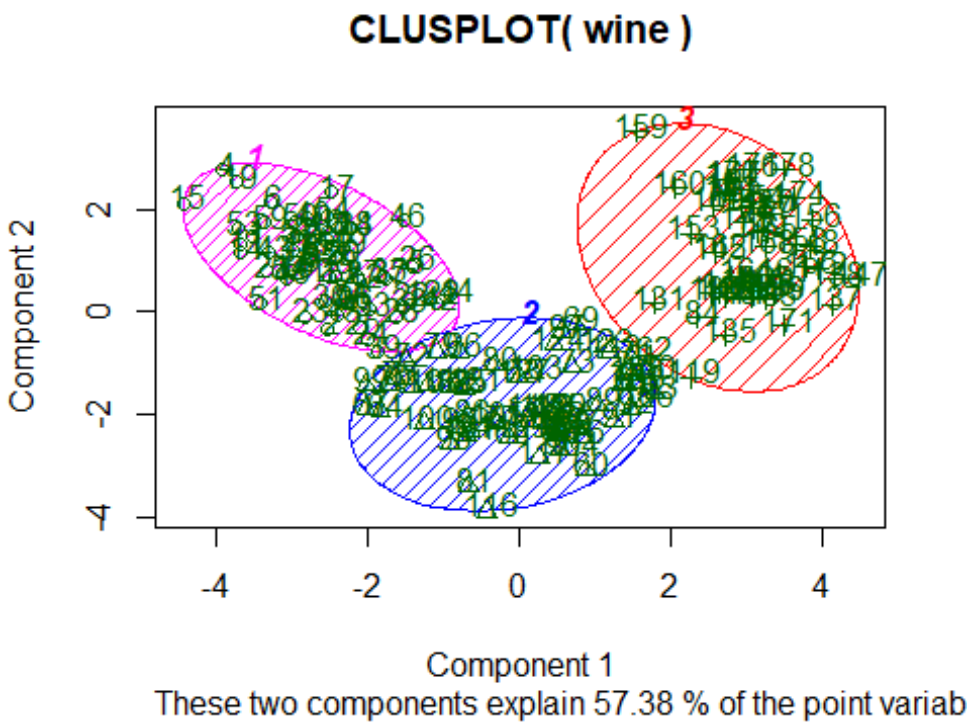
```
table(nc$Best.n[1,])

##
##  0  1  2  3 10 12 14 15
##  2  1  4 15  1  1  1  1

barplot(table(nc$Best.n[1,]), xlab="Numer of Clusters", ylab="Number of
Criteria", main="Number of Clusters Chosen by 26 Criteria")
```




```
library(cluster)
set.seed(1234)
fit.km <- kmeans(df, 3, nstart=25)
fit.km$size

## [1] 62 65 51

fit.km$centers

##      Alcohol      Malic        Ash Alcalinity   Magnesium     Phenols
## 1  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724
## 2 -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891
## 3  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548
```

```
##    Flavanoids Nonflavanoids Proanthocyanins      Color        Hue
## 1  0.97506900   -0.56050853      0.57865427  0.1705823  0.4726504
## 2  0.02075402   -0.03343924      0.05810161 -0.8993770  0.4605046
## 3 -1.21182921    0.72402116     -0.77751312  0.9388902 -1.1615122
##     Dilution    Proline
## 1  0.7770551  1.1220202
## 2  0.2700025 -0.7517257
## 3 -1.2887761 -0.4059428
```

```
clusplot(wine, fit.km$cluster, color=TRUE, shade=TRUE,  labels=2, lines=0)
```



Hierarchical methods use a distance matrix as an input for the clustering algorithm. The choice of an appropriate metric will influence the shape of the clusters, as some elements may be close to one another according to one distance and farther away according to another.

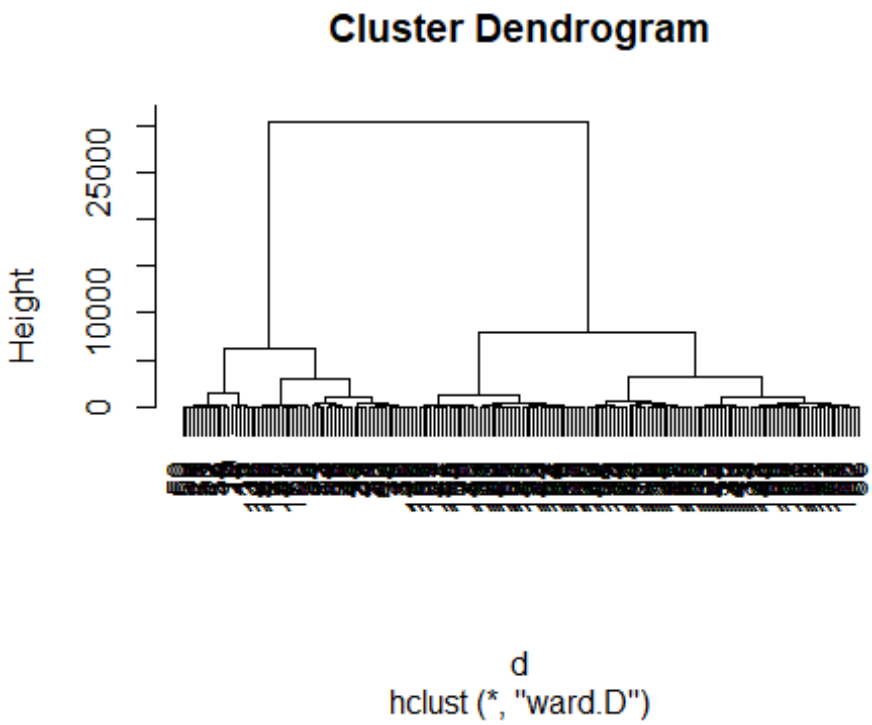
```
d <- dist(wine, method = "euclidean") # Euclidean distance matrix.
```

We use the Euclidean distance as an input for the clustering algorithm (Ward's minimum variance criterion minimizes the total within-cluster variance):

```
H.fit <- hclust(d, method="ward.D")
H2.fit<- hclust(d, method="ward.D2")
```

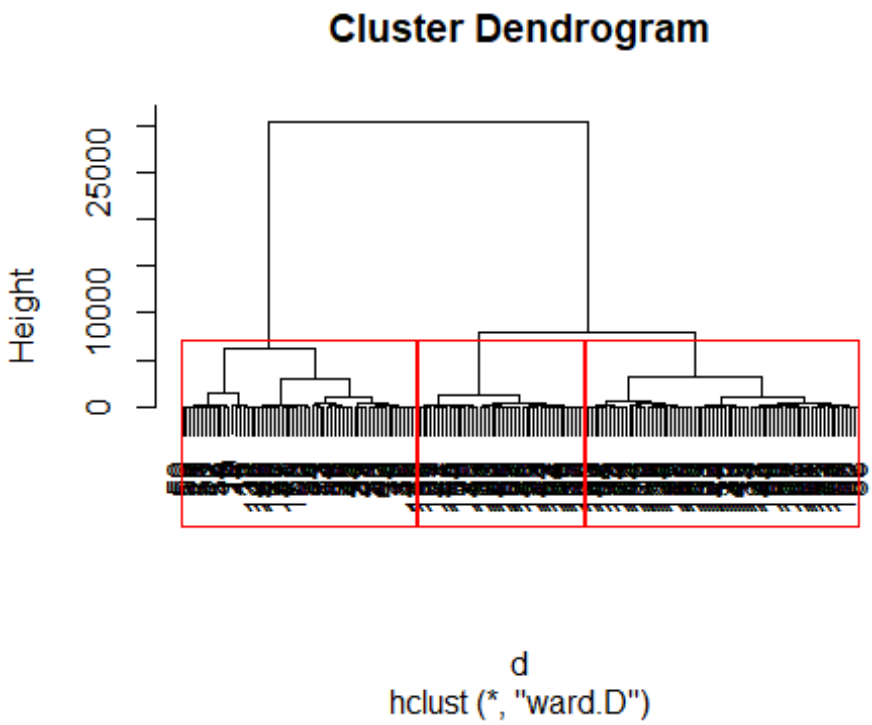
The clustering output can be displayed in a dendrogram.

```
plot(H.fit)
```

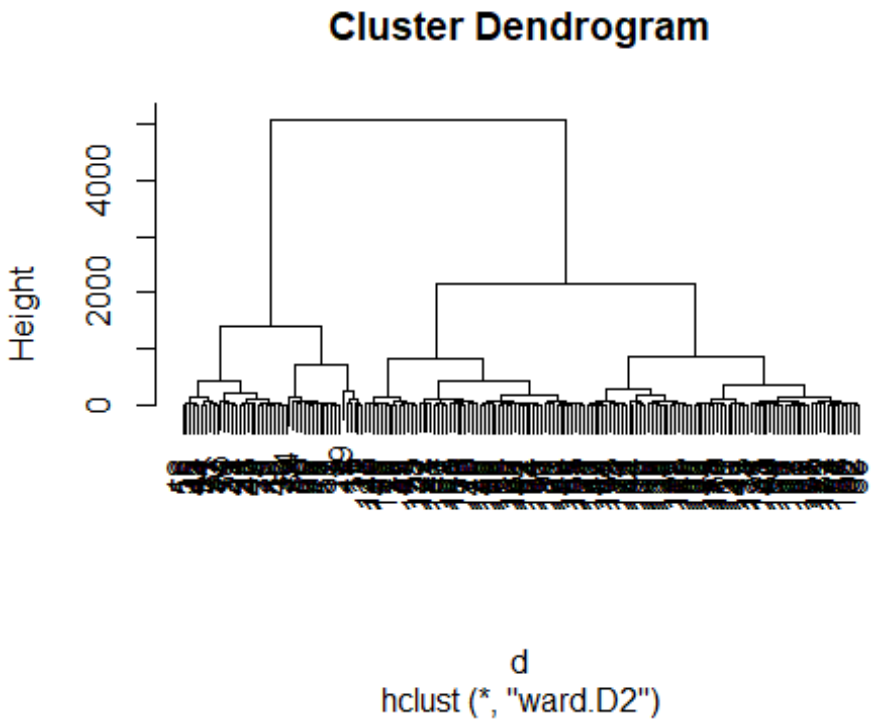


First, we draw dendogram with red borders around the 5 clusters.

```
plot(H.fit)
groups <- cutree(H.fit, k=3) # cut tree into 5 clusters
rect.hclust(H.fit, k=3, border="red")
```

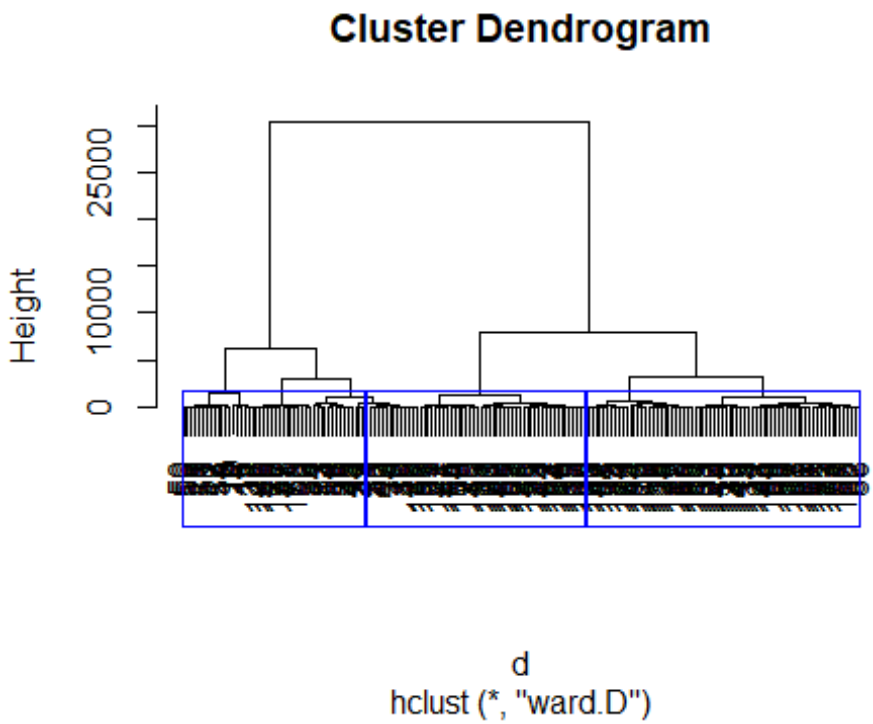



```
plot(H2.fit) # display dendogram
```



We now draw dendogram with blue borders around the 5 clusters.

```
plot(H.fit)
rect.hclust(H2.fit, k=3, border="blue")
```




```
groups <- cutree(H2.fit, k=3) # cut tree into 5 clusters
```

Social Network Clustering Analysis

For this analysis, we will be using a dataset representing a random sample of 30.000 U.S. high school students who had profiles on a well-known Social Network in from 2006 to 2009. From the top 500 words appearing across all pages, 36 words were chosen to represent five categories of interests, namely extracurricular activities, fashion, religion, romance, and antisocial behavior. The 36 words include terms such as football, sexy, kissed, bible, shopping, death, and drugs. The final dataset indicates, for each person, how many times each word appeared in the person's SNS profile.

```
social<-
read.csv(file="C:/Users/Jeff/Documents/VIT_University/data/social_net.csv",he
ader = T)
head(social,3)

##   gradyear gender    age friends basketball football soccer softball
## 1     2006      M 18.982       7          0        0      0        0
## 2     2006      F 18.801       0          0        1      0        0
## 3     2006      M 18.335      69          0        1      0        0
##   volleyball swimming cheerleading baseball tennis sports cute sex sexy
## 1          0        0            0        0      0      0    0   0    0
## 2          0        0            0        0      0      0    1   0    0
```

```
## 3          0        0            0        0      0      0    0   0    0
##   hot kissed dance band marching music rock god church jesus bible hair
## 1   0      0     1    0        0     0    0   0      0     0     0    0
## 2   0      0     0    0        0     2    2   1      0     0     0    6
## 3   0      0     0    2        0     1    0   0      0     0     0    0
##   dress blonde mall shopping clothes hollister abercrombie die death drunk
## 1     0      0    0        0       0         0           0   0     0     0
## 2     4      0    1        0       0         0           0   0     0     0
## 3     0      0    0        0       0         0           0   0     1     0
##   drugs
## 1     0
## 2     0
## 3     0

dim(social)

## [1] 10599    40
```

Let's also take a quick look at the specifics of the data. The first several lines of the str() output are as follows:

```
str(social)

## 'data.frame':    10599 obs. of  40 variables:
##  $ gradyear    : int  2006 2006 2006 2006 2006 2006 2006 2006 2006 2006
...
##  $ gender      : Factor w/ 2 levels "F","M": 2 1 2 1 NA 1 1 2 1 1 ...
##  $ age         : num  19 18.8 18.3 18.9 19 ...
##  $ friends     : int  7 0 69 0 10 142 72 17 52 39 ...
##  $ basketball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ football    : int  0 1 1 0 0 0 0 0 0 0 ...
##  $ soccer      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ softball    : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ volleyball  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ swimming    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cheerleading: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ baseball    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ tennis      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sports      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cute        : int  0 1 0 1 0 0 0 0 0 1 ...
##  $ sex         : int  0 0 0 0 1 1 0 2 0 0 ...
##  $ sexy        : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ hot         : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ kissed      : int  0 0 0 0 5 0 0 0 0 0 ...
##  $ dance       : int  1 0 0 0 1 0 0 0 0 0 ...
##  $ band        : int  0 0 2 0 1 0 1 0 0 0 ...
##  $ marching    : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ music       : int  0 2 1 0 3 2 0 1 0 1 ...
##  $ rock        : int  0 2 0 1 0 0 0 1 0 1 ...
##  $ god         : int  0 1 0 0 1 0 0 0 0 6 ...
##  $ church      : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ jesus       : int  0 0 0 0 0 0 0 0 0 2 ...
##  $ bible       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hair        : int  0 6 0 0 1 0 0 0 0 1 ...
##  $ dress       : int  0 4 0 0 0 1 0 0 0 0 ...
##  $ blonde      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ mall        : int  0 1 0 0 0 0 2 0 0 0 ...
##  $ shopping    : int  0 0 0 0 2 1 0 0 0 1 ...
##  $ clothes     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hollister   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ abercrombie : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ die         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ death       : int  0 0 1 0 0 0 0 0 0 0 ...
##  $ drunk       : int  0 0 0 0 1 1 0 0 0 0 ...
##  $ drugs       : int  0 0 0 0 1 0 0 0 0 0 ...
```

As we had expected, the data include 30,000 teenagers with four variables indicating personal characteristics and 36 words indicating interests. Note that there are some NA's in the variable gender.

```
summary(social$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   4.928  18.010  18.453  18.907  18.867 106.927    1690
```

We will skip all the data with missing values:

```
social = na.omit(social)
dim(social)
```

```
## [1] 8586   40
```

We'll start our cluster analysis by considering only the 36 features that represent the number of times various interests appeared on the SNS profiles of teens. For convenience, let's make a data frame containing only these features:

```
interests <- social[5:40]
```

To apply z-score standardization to the interests data frame, we can use the scale() function with lapply(), as follows:

```
interests_z <- as.data.frame(lapply(interests, scale))
```

To divide teens into five clusters, we can use the following command:

```
social_clusters <- kmeans(interests_z, 5)
```

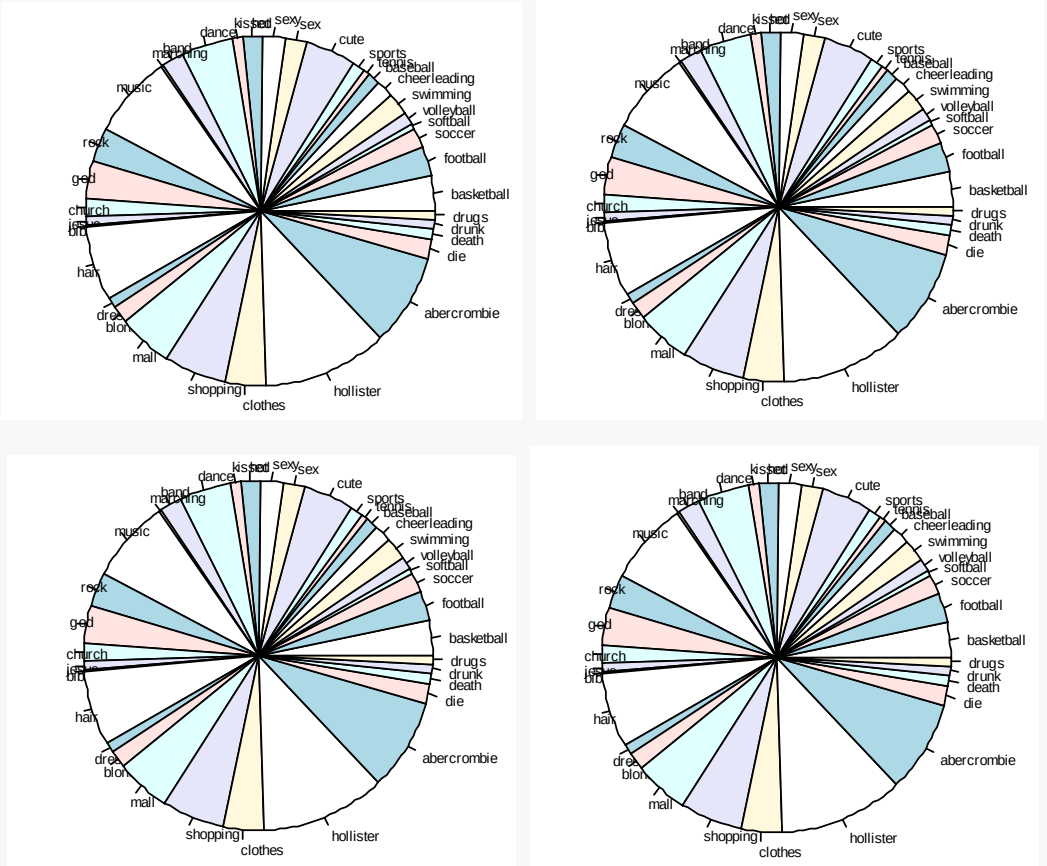
The number of examples falling in each of the groups is provided below. If the groups are too large or too small, then they are not likely to be very useful. To obtain the size of the kmeans() clusters, use the teen_clusters$size component as follows:

```
social_clusters$size
```

```
## [1]  251 6320 1578  176  261
```

The cluster characterization can be obtained with pie charts:

```
par(mfrow=c(2,2))
pie(colSums(interests[social_clusters$cluster==1,]),cex=0.5)
pie(colSums(interests[social_clusters$cluster==2,]),cex=0.5)
pie(colSums(interests[social_clusters$cluster==3,]),cex=0.5)
pie(colSums(interests[social_clusters$cluster==4,]),cex=0.5)
```



Customer Segmentation

Customer segmentation is as simple as it sounds: grouping customers by their characteristics - and why would you want to do that? To better serve their needs! Our example is to do with e-mail marketing.

```
offers<-read.table("C:/Users/Jeff/Documents/VIT_University/data/offers.csv",
sep = ',', header=T)
head(offers)
```

```
##   OfferID Campaign           Varietal MinimumQt Discount      Origin
## 1       1  January              Malbec        72       56      France
```

```
## 2       2  January         Pinot Noir        72       17      France
## 3       3 February          Espumante       144       32      Oregon
## 4       4 February          Champagne        72       48      France
## 5       5 February Cabernet Sauvignon       144       44 New Zealand
## 6       6    March           Prosecco       144       86       Chile
##   PastPeak
## 1    FALSE
## 2    FALSE
## 3     TRUE
## 4     TRUE
## 5     TRUE
## 6    FALSE
```

```
transactions<-
read.table("C:/Users/Jeff/Documents/VIT_University/data/transactions.csv",
sep = ',', header=T)
head(transactions)
```

```
##   CustomerLastName OfferID
## 1            Smith       2
## 2            Smith      24
## 3          Johnson      17
## 4          Johnson      24
## 5          Johnson      26
## 6         Williams      18
```

Step 1: Organizing the information

We have two data sets: one for the offers and the other for the transactions. First what we need to do is create a transaction matrix. That means, we need to put the offers we mailed out next to the transaction history of each customer. This is easily achieved with a pivot table.

```
library(reshape)
pivot<-melt(transactions[1:2])#Using CustomerLastName as id variables

## Using CustomerLastName as id variables

pivot<-(cast(pivot,value~CustomerLastName,fill=0,fun.aggregate=function(x)
length(x)))
pivot<-cbind(offers,pivot[-1])
# write.csv(file="pivot.csv",pivot) # to save your data
cluster.data<-pivot[,8:length(pivot)]
cluster.data<-t(cluster.data)
head(cluster.data)
```

```
##          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
## Adams    0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0
## Allen    0 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## Anderson 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0
## Bailey   0 0 0 0 0 0 1 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
## Baker    0 0 0 0 0 0 1 0 0  1  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
## Barnes   0 0 0 0 0 0 0 0 0  1  0  0  0  0  0  0  0  0  0  0  1  1  0  0  0
##          26 27 28 29 30 31 32
## Adams     0  0  0  1  1  0  0
## Allen     0  1  0  0  0  0  0
## Anderson  1  0  0  0  0  0  0
## Bailey    0  0  0  0  1  0  0
## Baker     0  0  0  0  0  1  0
## Barnes    0  0  0  0  0  1  0
```

In the clustering data set, rows represents costumers and columns are different wine brands/types.

Step 2: Distances and Clusters

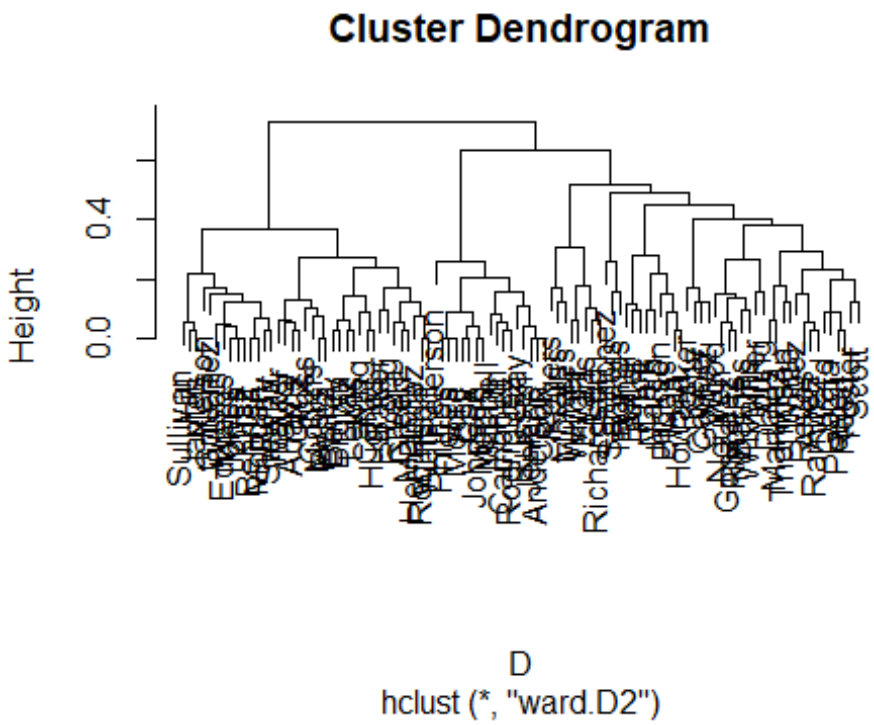
We will use k=4 indicating that we will use 4 clusters. This is somewhat arbitrary, but the number you pick should be representative of the number of segments you can handle as a business. So 100 segments does not make sense for an e-mail marketing campaign. We need to calculate how far away each customer is from the cluster's mean. To do this we could use many distances/dissimilarity index, one of which is the Gower dissimilarity.

```
library(cluster)
D=daisy(cluster.data, metric='gower')
```

```
## Warning in daisy(cluster.data, metric = "gower"): binary variable(s) 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 treated as interval scaled
```

After the creation of a distance matrix, we implement a Ward's hierarchical clustering procedure:

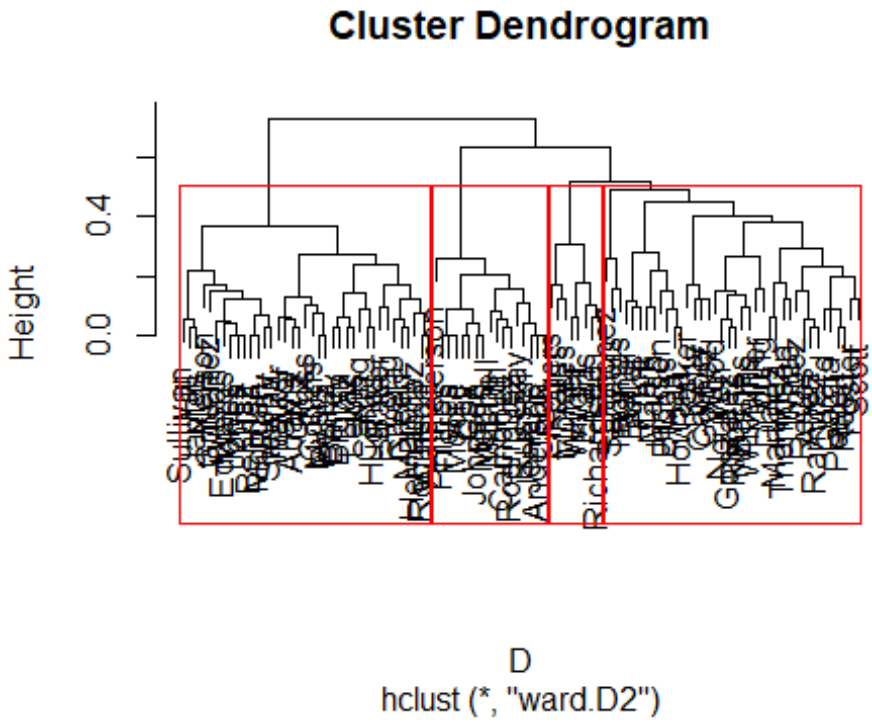
```
H.fit <- hclust(D, method="ward.D2")
plot(H.fit) # display dendrogram
```




```
groups <- cutree(H.fit, k=4) # cut tree into 4 clusters
```

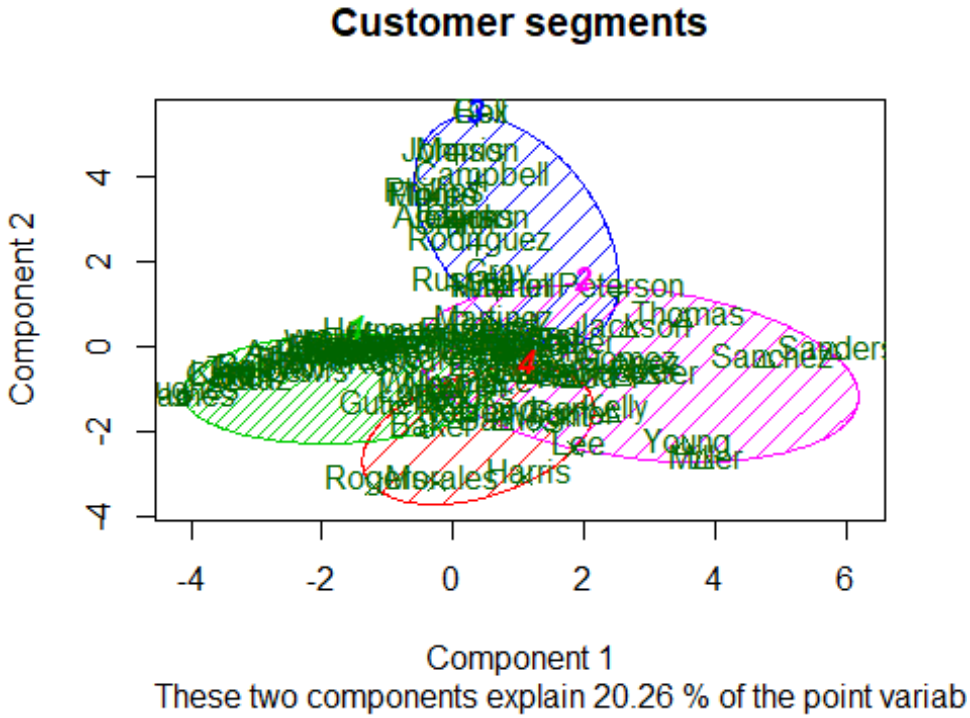
Now we draw dendogram with red borders around the 4 clusters

```
plot(H.fit) # display dendrogram
groups <- cutree(H.fit, k=4)
rect.hclust(H.fit, k=4, border="red")
```



2D representation of the Segmentation:

```
clusplot(cluster.data, groups, color=TRUE, shade=TRUE,
         labels=2, lines=0, main= 'Customer segments')
```



Generally, the way K-Means algorithms work is via an iterative refinement process: - Each data point is randomly assigned to a cluster (number of clusters is given before hand). - Each cluster's centroid (mean within cluster) is calculated. - Each data point is assigned to its nearest centroid (iteratively to minimise the within-cluster variation) until no major differences are found.

Let's have a look at an example in R using the Chatterjee-Price Attitude Data from the library(datasets) package. The dataset is a survey of clerical employees of a large financial organization. The data are aggregated from questionnaires of approximately 35 employees for each of 30 (randomly selected) departments. The numbers give the percent proportion of favourable responses to seven questions in each department. For more details, see ?attitude.

Load necessary libraries

```
library(datasets)
```

Inspect data structure

```
str(attitude)
```

```
## 'data.frame':    30 obs. of  7 variables:
##  $ rating    : num  43 63 71 61 81 43 58 71 72 67 ...
##  $ complaints: num  51 64 70 63 78 55 67 75 82 61 ...
##  $ privileges: num  30 51 68 45 56 49 42 50 72 45 ...
##  $ learning  : num  39 54 69 47 66 44 56 55 67 47 ...
##  $ raises    : num  61 63 76 54 71 54 66 70 71 62 ...
##  $ critical  : num  92 73 86 84 83 49 68 66 83 80 ...
##  $ advance   : num  45 47 48 35 47 34 35 41 31 41 ...
```

Summarise data

```
summary(attitude)
```

```
##      rating        complaints     privileges       learning   
##  Min.   :40.00   Min.   :37.0   Min.   :30.00   Min.   :34.00  
##  1st Qu.:58.75   1st Qu.:58.5   1st Qu.:45.00   1st Qu.:47.00  
##  Median :65.50   Median :65.0   Median :51.50   Median :56.50  
##  Mean   :64.63   Mean   :66.6   Mean   :53.13   Mean   :56.37  
##  3rd Qu.:71.75   3rd Qu.:77.0   3rd Qu.:62.50   3rd Qu.:66.75  
##  Max.   :85.00   Max.   :90.0   Max.   :83.00   Max.   :75.00  
##      raises         critical        advance      
##  Min.   :43.00   Min.   :49.00   Min.   :25.00  
##  1st Qu.:58.25   1st Qu.:69.25   1st Qu.:35.00  
##  Median :63.50   Median :77.50   Median :41.00  
##  Mean   :64.63   Mean   :74.77   Mean   :42.93  
##  3rd Qu.:71.00   3rd Qu.:80.00   3rd Qu.:47.75  
##  Max.   :88.00   Max.   :92.00   Max.   :72.00
```

As we've seen, this data gives the percent of favourable responses for each department. For example, in the summary output above we can see that for the variable privileges among all 30 departments the minimum percent of favourable responses was 30 and the maximum

was 83. In other words, one department had only 30% of responses favourable when it came to assessing 'privileges' and one department had 83% of favorable responses when it came to assessing 'privileges', and a lot of other favourable response levels in between.

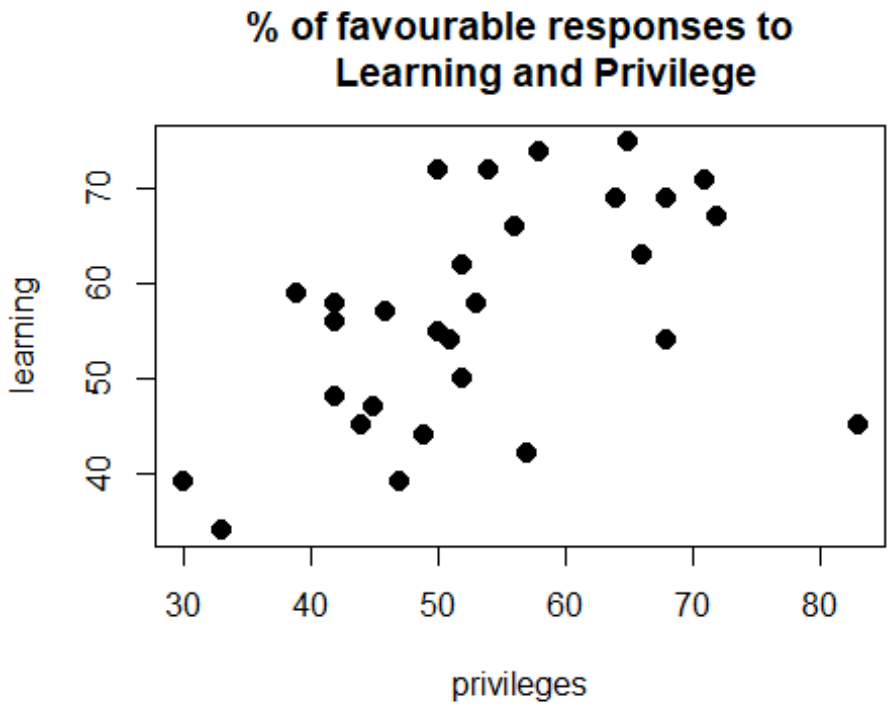
In light of the example, we'll take a subset of the attitude dataset and consider only two variables in our K-Means clustering exercise. So imagine that we would like to cluster the attitude dataset with the responses from all 30 departments when it comes to 'privileges' and 'learning' and we would like to understand whether there are commonalities among certain departments when it comes to these two variables.

Subset the attitude data

```
dat = attitude[,c(3,4)]
```

Plot subset data

```
plot(dat, main = "% of favourable responses to
     Learning and Privilege", pch =20, cex =2)
```



With the data subset and the plot above we can see how each department's score behave across Privilege and Learning compare to each other. In the most simplistic sense, we can apply K-Means clustering to this data set and try to assign each department to a specific number of clusters that are "similar".

Let's use the kmeans function from R base stats package: Perform K-Means with 2 clusters

```
set.seed(7)
km1 = kmeans(dat, 2, nstart=100)
```

Plot results

```
plot(dat, col =(km1$cluster +1) , main="K-Means result with 2 clusters",
pch=20, cex=2)
```

As mentioned before, one of the key decisions to be made when performing K-Means clustering is to decide on the numbers of clusters to use. In practice, there is no easy answer and it's important to try different ways and numbers of clusters to decide which options is the most useful, applicable or interpretable solution.

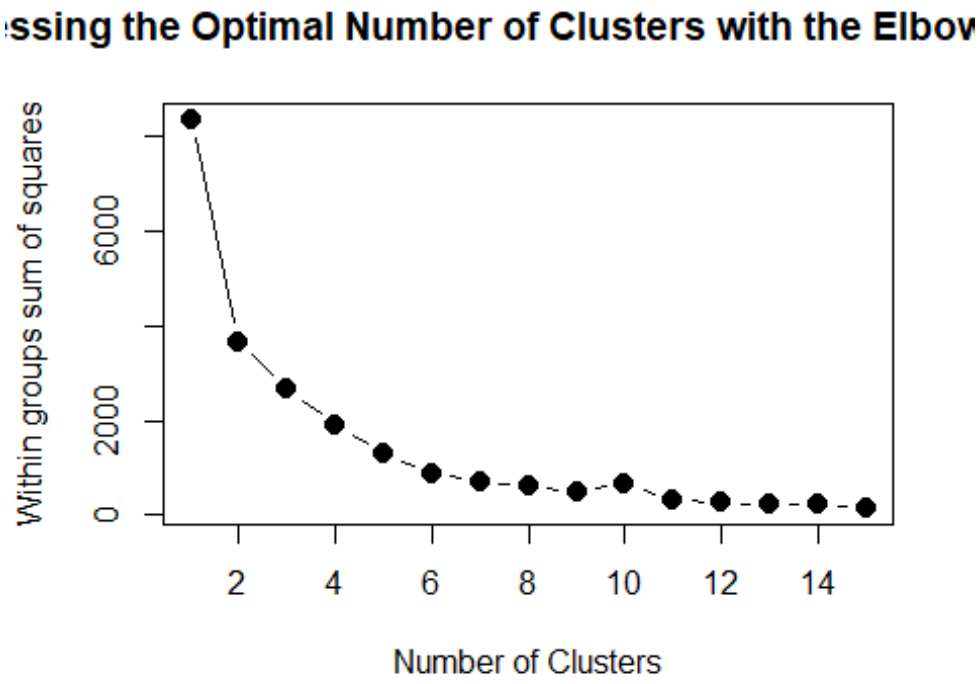
In the plot above, we randomly chose the number of clusters to be 2 for illustration purposes only.

However, one solution often used to identifiy the optimal number of clusters is called the Elbow method and it involves observing a set of possible numbers of clusters relative to how they minimise the within-cluster sum of squares. In other words, the Elbow method examines the within-cluster dissimilarity as a function of the number of clusters. Below is a visual representation of the method:

Check for the optimal number of clusters given the data.

```
mydata <- dat
wss <- (nrow(mydata)-1)*sum(apply(mydata,2,var))
  for (i in 2:15) wss[i] <- sum(kmeans(mydata,
                                       centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares",
```

```
     main="Assessing the Optimal Number of Clusters with the Elbow Method",
     pch=20, cex=2)
```



We can say that after 6 clusters the observed difference in the within-cluster dissimilarity is not substantial. Consequently, we can say with some reasonable confidence that the optimal number of clusters to be used is 6.

Perform K-Means with the optimal number of clusters identified from the Elbow method

```
set.seed(7)
km2 = kmeans(dat, 6, nstart=100)
```

Examine the result of the clustering algorithm

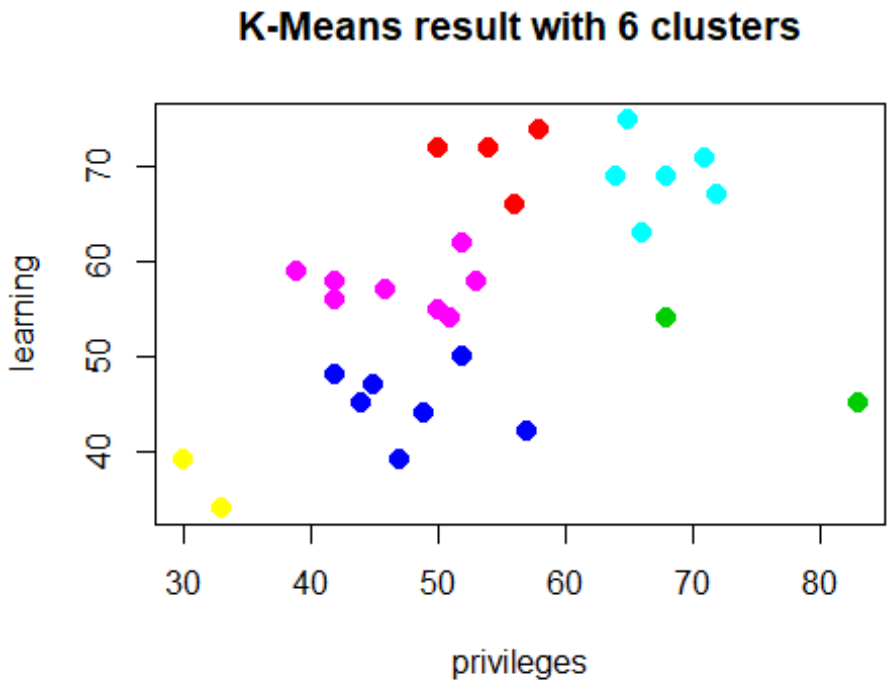
```
km2
```

```
## K-means clustering with 6 clusters of sizes 4, 2, 8, 6, 8, 2
##
## Cluster means:
##   privileges learning
## 1   54.50000   71.000
## 2   75.50000   49.500
## 3   47.62500   45.250
## 4   67.66667   69.000
## 5   46.87500   57.375
## 6   31.50000   36.500
##
## Clustering vector:
```

```
##  [1] 6 5 4 3 1 3 5 5 4 3 5 3 3 2 1 1 4 4 5 2 6 5 3 5 3 4 1 3 4 5
##
## Within cluster sum of squares by cluster:
## [1]  71.0000 153.0000 255.3750 133.3333 244.7500  17.0000
##  (between_SS / total_SS =  89.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

##Plot results

```
plot(dat, col =(km2$cluster +1) , main="K-Means result with 6 clusters",
pch=20, cex=2)
```



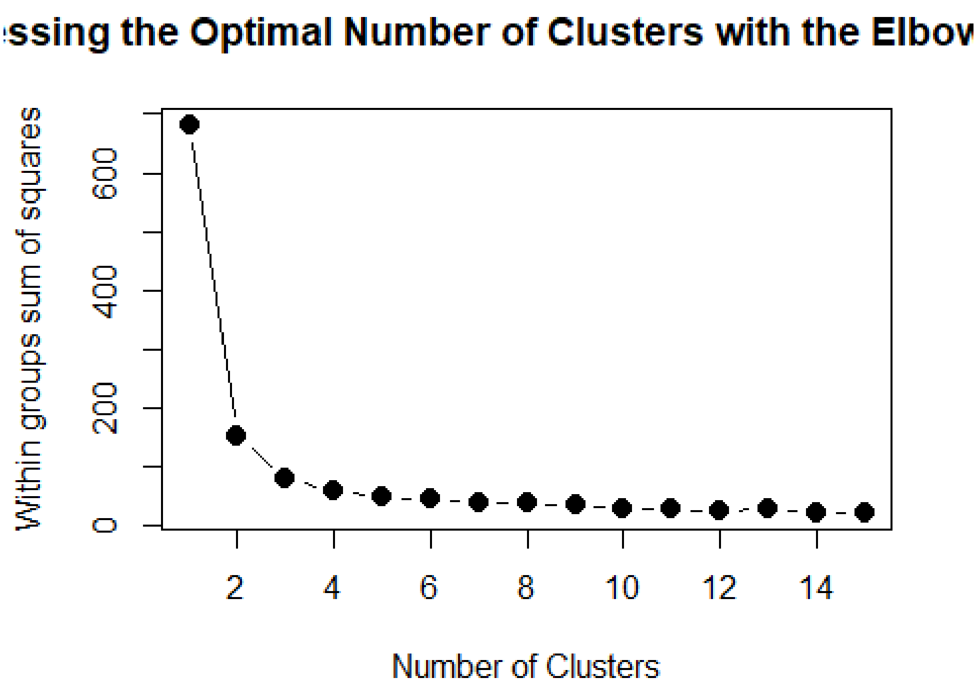
From the results above we can see that there is a relatively well defined set of groups of departments that are relatively distinct when it comes to answering favourably around Privileges and Learning in the survey. It is only natural to think the next steps from this sort of output. One could start to devise strategies to understand why certain departments rate these two different measures the way they do and what to do about it. But we will leave this to another exercise.

#Iris kmeans #1

```r
iris2 <- iris
iris2$Species <- NULL
mydata <- iris2
```

#How many clusters?

```r
wss <- (nrow(mydata)-1)*sum(apply(mydata,2,var))
  for (i in 2:15) wss[i] <- sum(kmeans(mydata,
                                        centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares",
     main="Assessing the Optimal Number of Clusters with the Elbow Method",
     pch=20, cex=2)
```



#K-Means

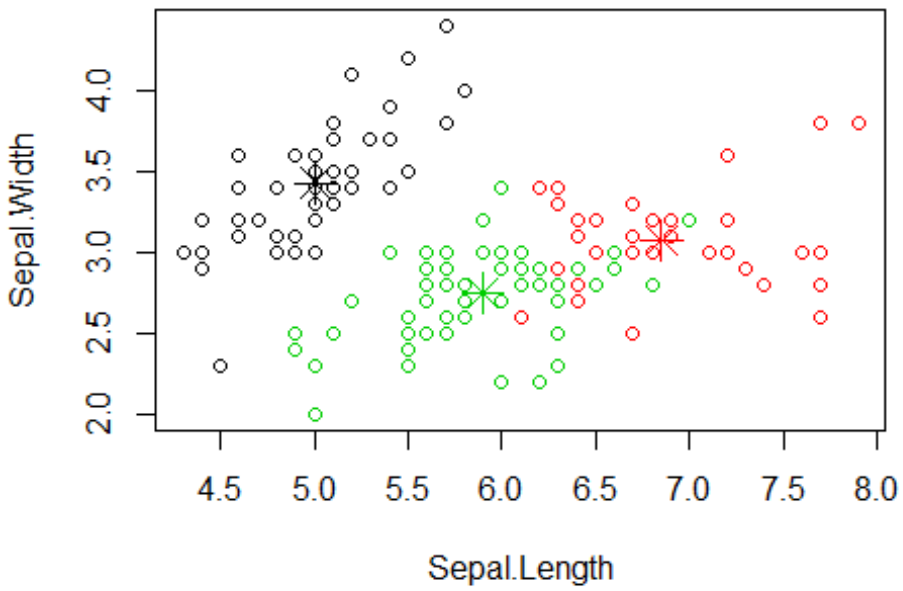
```r
(kmeans.result <- kmeans(iris2, 3))

## K-means clustering with 3 clusters of sizes 50, 38, 62
##
## Cluster means:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.006000    3.428000     1.462000    0.246000
## 2     6.850000    3.073684     5.742105    2.071053
## 3     5.901613    2.748387     4.393548    1.433871
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3
##  [71] 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 2 2
2
## [106] 2 3 2 2 2 2 2 2 3 3 2 2 2 2 3 2 3 2 3 2 2 3 3 2 2 2 2 2 3 2 2 2 2 3
2
## [141] 2 2 3 2 2 2 3 2 2 3
##
## Within cluster sum of squares by cluster:
## [1] 15.15100 23.87947 39.82097
##  (between_SS / total_SS =  88.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
table(iris$Species, kmeans.result$cluster)
```

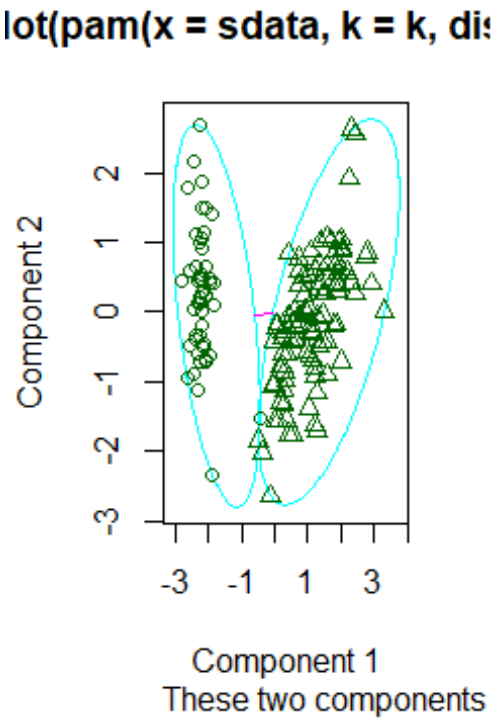
```
##
##                1  2  3
##   setosa      50  0  0
##   versicolor   0  2 48
##   virginica    0 36 14
```

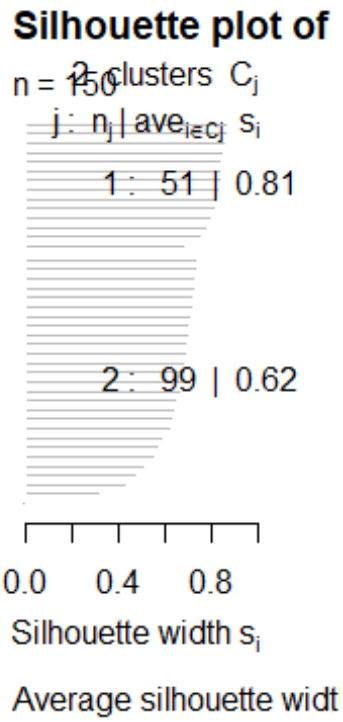
```
plot(iris2[c("Sepal.Length", "Sepal.Width")], col = kmeans.result$cluster)
points(kmeans.result$centers[,c("Sepal.Length", "Sepal.Width")], col = 1:3,
pch = 8, cex=2)
```



#Iris kmeans #2

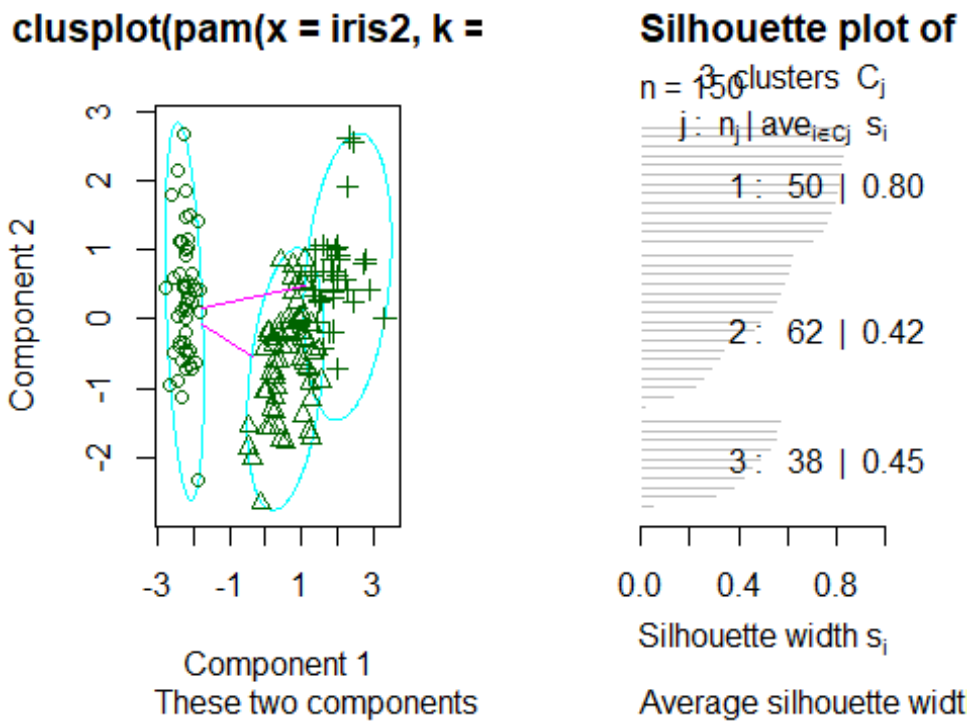
```
library(fpc)
pamk.result <- pamk(iris2)
pamk.result$nc

## [1] 2

table(pamk.result$pamobject$clustering, iris$Species)

##
##     setosa versicolor virginica
##   1     50          1         0
##   2      0         49        50

layout(matrix(c(1,2),1,2)) # 2 graphs per page
plot(pamk.result$pamobject)
```






```
layout(matrix(1)) # change back to one graph per page

#————————————————————————————-
```

```
library(fpc)
pam.result <- pam(iris2, 3)
table(pam.result$clustering, iris$Species)

##
##     setosa versicolor virginica
##   1     50          0         0
##   2      0         48        14
##   3      0          2        36

layout(matrix(c(1,2),1,2)) # 2 graphs per page
plot(pam.result)
```




```
layout(matrix(1)) # change back to one graph per page
```

#Iris kmeans #3

```
library(ggplot2)
set.seed(20)
irisCluster <- kmeans(iris2[, 3:4], 3, nstart = 20)
irisCluster
```

```
## K-means clustering with 3 clusters of sizes 50, 52, 48
##
## Cluster means:
##   Petal.Length Petal.Width
## 1     1.462000    0.246000
## 2     4.269231    1.342308
## 3     5.595833    2.037500
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [71] 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3
## [106] 3 2 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 2
## [141] 3 3 3 3 3 3 3 3 3 3
```

```
##
## Within cluster sum of squares by cluster:
## [1]  2.02200 13.05769 16.29167
##  (between_SS / total_SS =  94.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
table(irisCluster$cluster, iris$Species)
```

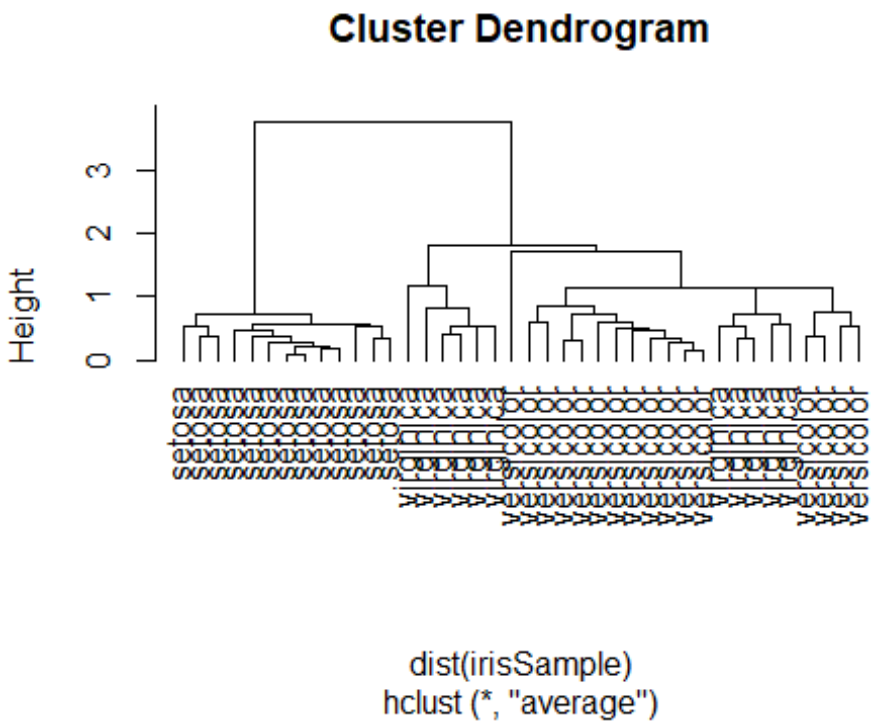
```
##
##     setosa versicolor virginica
##   1     50          0         0
##   2      0         48         4
##   3      0          2        46
```

```
irisCluster$cluster <- as.factor(irisCluster$cluster)
ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) +
geom_point()
```

#Hierarchical Clustering

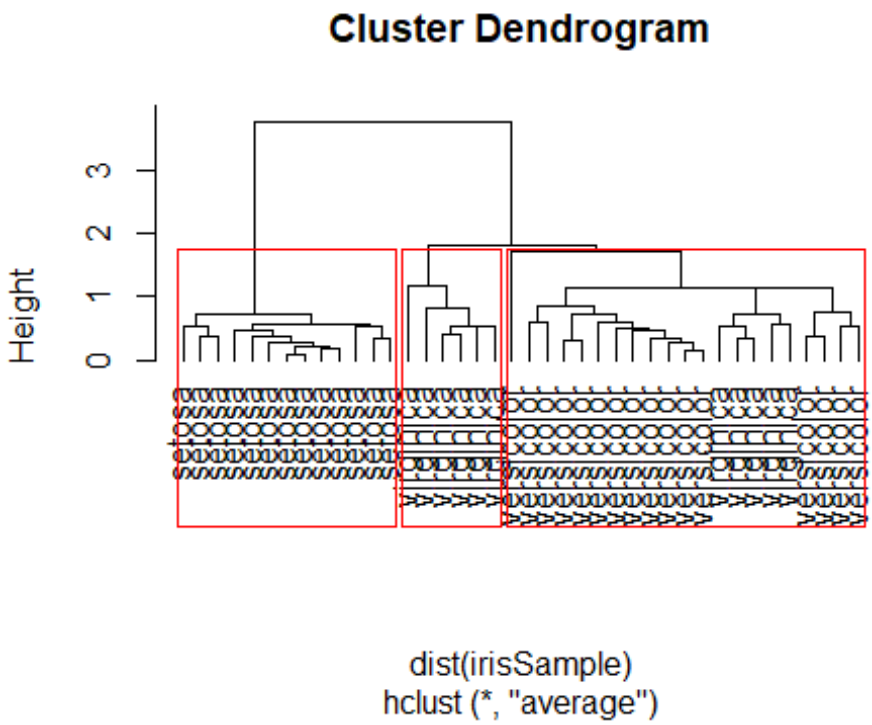
```
idx <- sample(1:dim(iris)[1], 40)
irisSample <- iris[idx,]
irisSample$Species <- NULL
```

```r
hc <- hclust(dist(irisSample), method="ave")
plot(hc, hang = -1, labels=iris$Species[idx])
```



Cut tree into 3 clusters

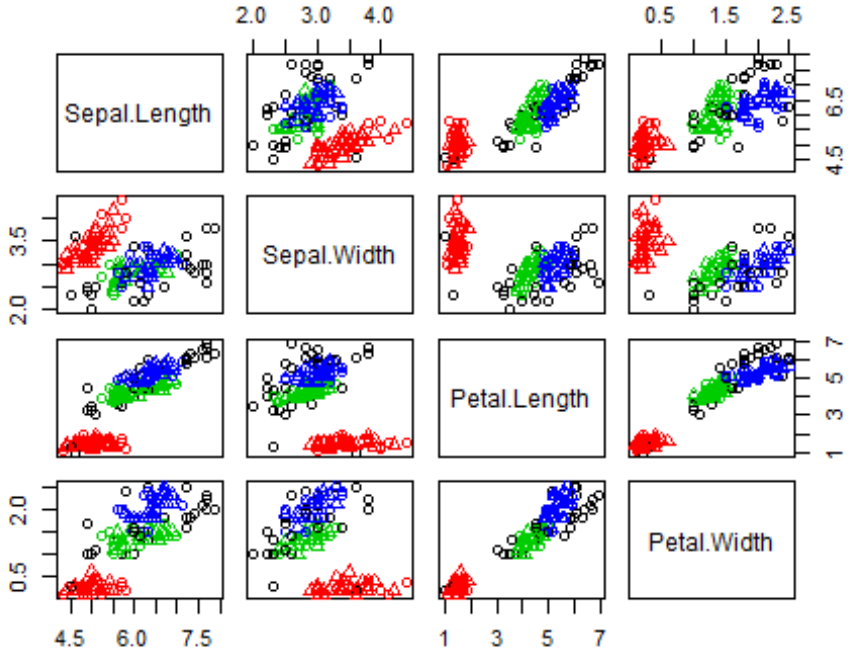
```r
plot(hc, hang = -1, labels=iris$Species[idx])
rect.hclust(hc, k=3)
```



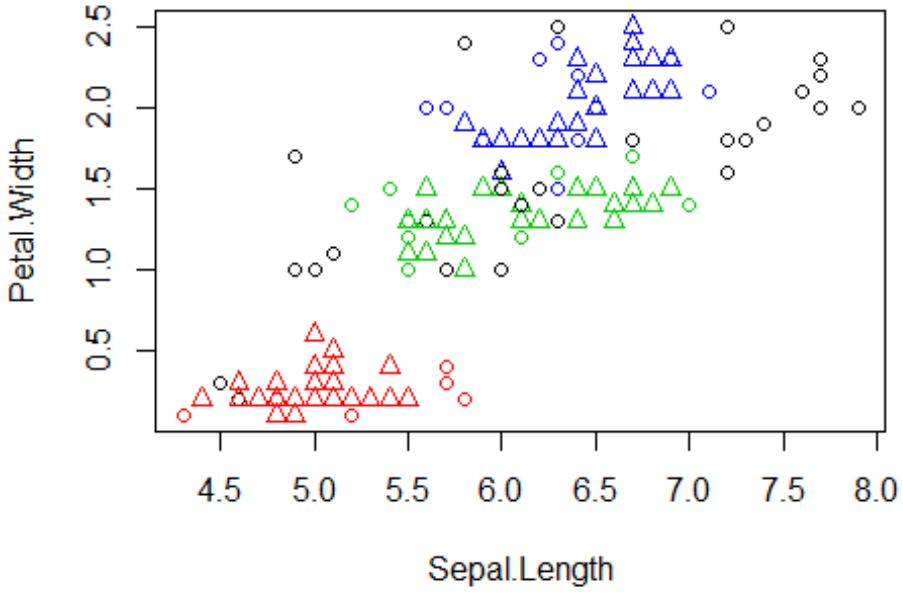

```
groups <- cutree(hc, k=3)
```

##Density-based Clustering

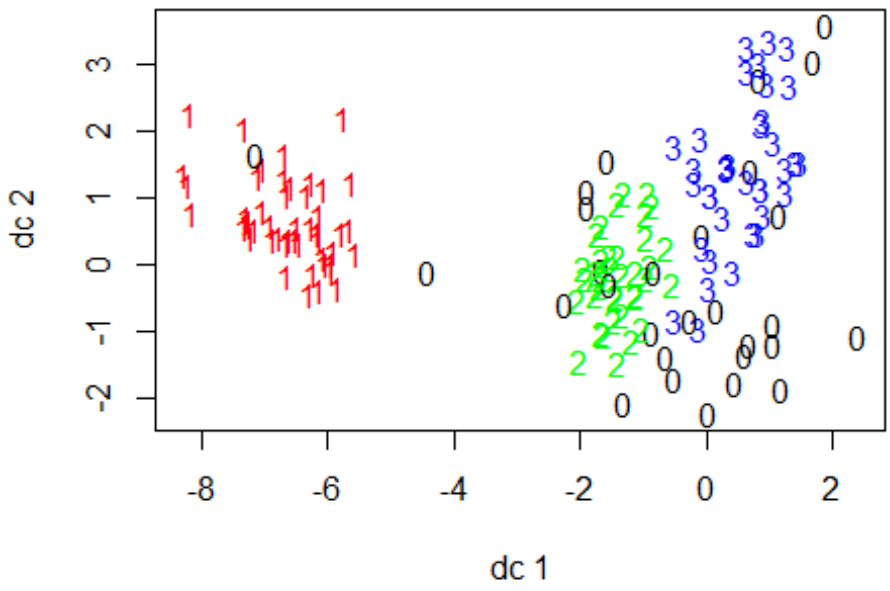
```
library(fpc)
iris2 <- iris[-5] # remove class tags

ds <- dbscan(iris2, eps=0.42, MinPts=5)
# compare clusters with original class labels
table(ds$cluster, iris$Species)

##
##     setosa versicolor virginica
##   0      2         10        17
##   1     48          0         0
##   2      0         37         0
##   3      0          3        33

plot(ds, iris2)
```




```
plot(ds, iris2[c(1,4)])
```




```
plotcluster(iris2, ds$cluster)
```



Now we create a new dataset for labeling

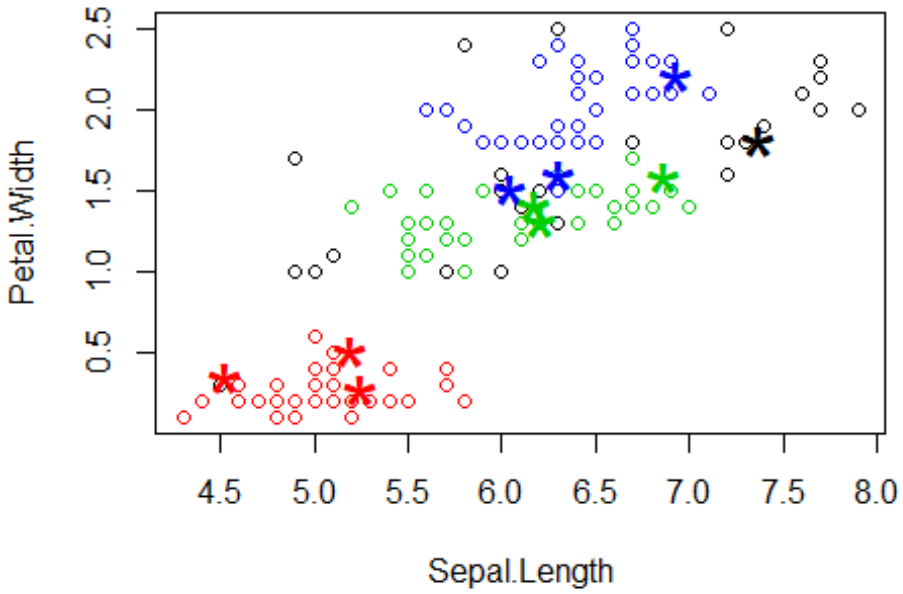
```
set.seed(435)
idx <- sample(1:nrow(iris), 10)
newData <- iris[idx,-5]
newData <- newData + matrix(runif(10*4, min=0, max=0.2), nrow=10, ncol=4)
```

Label new data

```
myPred <- predict(ds, iris2, newData)
```

Plot result

```
plot(iris2[c(1,4)], col=1+ds$cluster)
points(newData[c(1,4)], pch="*", col=1+myPred, cex=3)
```



check cluster labels

```
table(myPred, iris$Species[idx])
```

```
##
## myPred setosa versicolor virginica
##      0      0          0         1
##      1      3          0         0
##      2      0          3         0
##      3      0          1         2
```