Sane's STATS Academy of Statistics

Cluster Analysis

Need for Clustering

- Groups available for grouping Customers
  - Male and female
  - High Income, Medium Income, Low Income
  - Having average invoice amounts as < 1Lac, between 5 Lac and 10 Lac etc.
  - Customers making Cheque, Cash or card payments
  - Any other type of grouping
- As per our convenience we categorize the entities based on some attributes

Need for Clustering

- There may be a business need of grouping the business entities on some variables.
- Such as we may group the products not just on the profit margin but also considering the variables such as quantity sold, average shelf life, maintenance cost etc.

Sane's STATS Academy of Statistics

Why is Clustering done?

- Cluster analysis is used to form groups or clusters of similar records based on several measurements made on these records.

Clustering in Marketing

- Market Segmentation: Customers are segmented based on demographic and transaction history information and a marketing strategy is tailored for each segment.
- Market structure analysis: Identifying groups of similar products according to competitive measures of similarity.

Clustering in Finance

- Creating balanced portfolios: Given data on a variety of investment opportunities (e.g., stocks), one may find clusters based on financial performance variables such as return (daily, weekly, or monthly), volatility and other characteristics, such as industry and market capitalization.

Clustering in Pure Sciences

- Biologists have made extensive use of classes and subclasses to organize species.
- In chemistry, Mendeleeyev's periodic table of the elements.

Distance Method

- For record i we have the vector of p measurements (xi1, xi2, . . . , xip), while for record j we have the vector of measurements (xj1, xj2, . . . , xjp).
- The most popular distance measure is the Euclidean distance, dij , which between two cases, i and j, is defined by

$$dij = \sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\ldots+(x_{ip}-x_{jp})^2}$$

Sane's STATS Academy of Statistics

Other Distance Measures

- Numerical Data
  - Correlation-based similarity
  - Statistical distance (also called Mahalanobis distance)
  - Manhattan distance (“city block”)
  - Maximum coordinate distance
- Categorical Data
  - Matching coefficient: (a + d)/p
  - Jaquard’s coefficient: d/(b+c+d)

Distance Calculation in R

- The distance can be calculated with dist( ) function in the base installation of R

Syntax : dist (x , method, …)

Where

x : numeric matrix / data frame / dist object

method : the distance measure to be used. Possible values are "euclidean", "manhattan", "canberra", "binary" , "minkowski“ or “maximum”

Program & Output: Distance Calculation & Scaling

```
> # Calculating Distances
> data(milk, package="flexclust")
> head(milk, 2)
          water protein fat lactose  ash
HORSE      90.1     2.6 1.0     6.9 0.35
ORANGUTAN  88.5     1.4 3.5     6.0 0.24
> d <- dist(milk)
> as.matrix(d)[1:5,1:5]
             HORSE ORANGUTAN   MONKEY   DONKEY     HIPPO
HORSE     0.000000  3.327477 2.493772 1.225765 4.759464
ORANGUTAN 3.327477  0.000000 1.205653 2.793850 2.798142
MONKEY    2.493772  1.205653 0.000000 2.374532 3.715696
DONKEY    1.225765  2.793850 2.374532 0.000000 3.762978
HIPPO     4.759464  2.798142 3.715696 3.762978 0.000000
```

```
> milk.scaled <- scale(milk)
> d <- dist(milk.scaled)
```

Sane's STATS Academy of Statistics

Scaling the data

- If the variables in analysis vary in range, the variables with large values will have larger impact on the results.
- To avoid this undesirable, we should scale the data
- Variable can be scaled in the following ways:
  - Subtract mean from each value(centering) and divide it by its standard deviation(scaling)
  - Divide each value in the variable by maximum value of the variable
  - Subtract mean from each value(centering) and divide it by its mean deviation about mean(scaling)

Scaling and Centering in R

- Scaling and Centering can be done by first alternative directly with function scale( )

Syntax : scale (x , center=TRUE , scale = TRUE)

Where

x : numeric matrix

center , scale : logical values

Steps in Clustering

1. Choosing appropriate attributes (variables)
2. Scaling and/or centering the data
3. Screening for outliers
4. Calculating Distances
5. Selecting a Cluster Algorithm
6. Obtaining one or more cluster solutions
7. Determining the number of clusters present
8. Interpreting the clusters

Types of Clustering Methods

- Hierarchical methods
  - **Agglomerative**
  - Divisive
- Nonhierarchical methods
  - **K-Means**
  - K-Medoids

Sane's STATS Academy of Statistics

Hierarchical Clustering

Distance Between Two Clusters

- Minimum distance (single linkage): the distance between the pair of records Ai and Bj that are closest.
- Maximum distance (complete linkage): the distance between the pair of records Ai and Bj that are farthest.
- Average distance (average linkage): the average distance of all possible distances between records in one cluster and records in the other cluster.

Distance Between Two Clusters

- Centroid distance: the distance between the two cluster centroids. A cluster centroid is the vector of measurement averages across all the records in that cluster.
- Ward: ANOVA sum of squares between the two clusters added up over the variables

Agglomerative Hierarchical Method

- Agglomerative method begins with n (No. of observations) clusters and sequentially merge similar clusters until a single cluster is left.

Algorithm

1. Start with n clusters (each observation = cluster).
2. The two closest observations are merged into one cluster.
3. At every step, the two clusters with the smallest distance are merged. This means that either single observations are added to existing clusters or two existing clusters are combined.

Hierarchical Clustering in R

- Hierarchical Clustering in R can be implemented using function hclust( )

Syntax : hclust (d , method , …)

Where

d : dissimilarity structure produced by dist () distance function

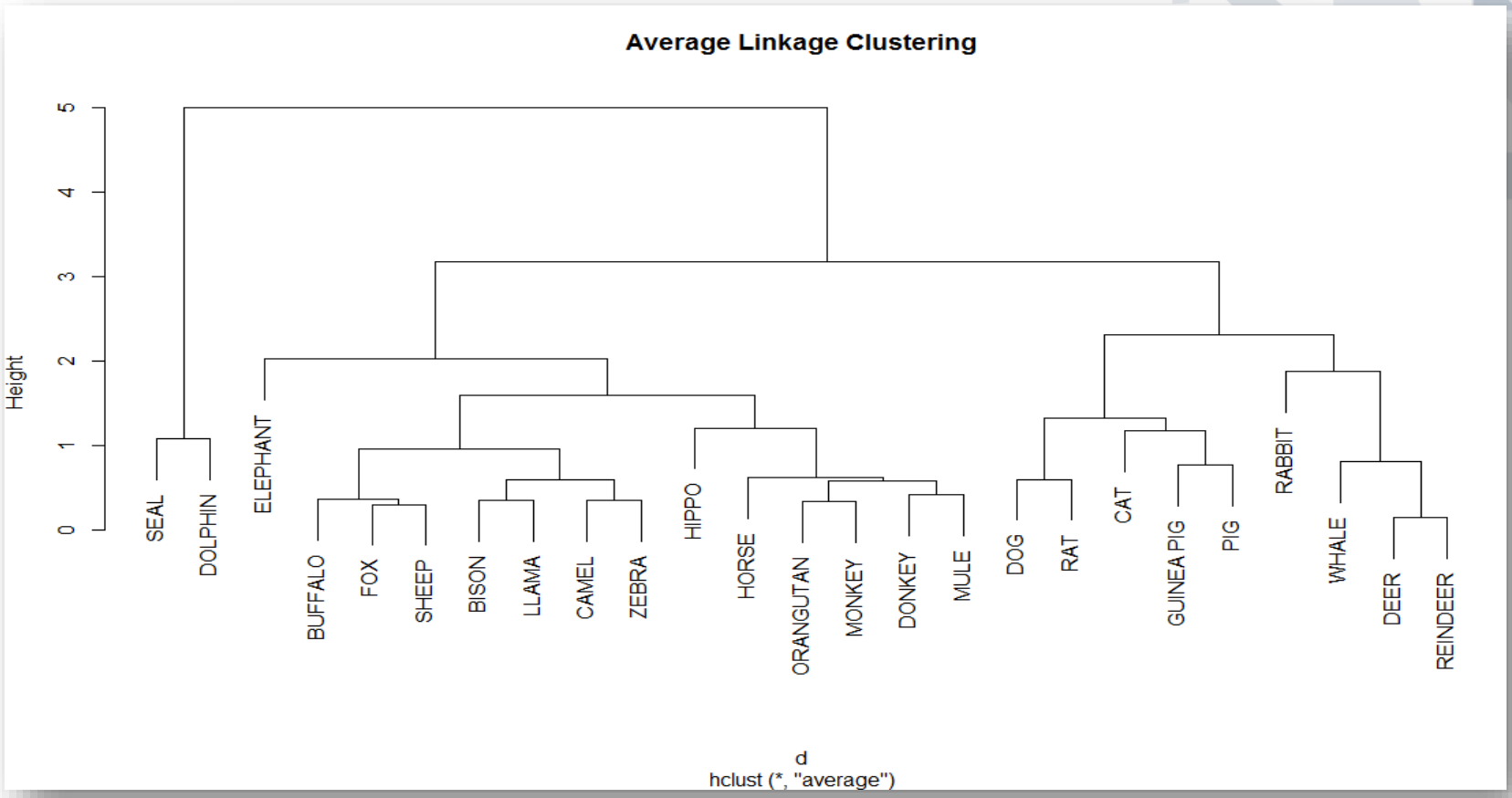
method : can values as “ward.D”(default), “single”, “complete”, “average”, etc.

Example: Milk (dataset in package *flexclust*)

- The data set contains the ingredients of mammal's milk of 25 animals.
- A data frame with 25 observations on the following 5 variables (all in percent)
  - water
  - protein
  - fat
  - lactose
  - ash
- Here, we are interested in grouping the 25 mammals based on the above given 5 nutrient measures

Program & Output: Dendogram

```
fit.average <- hclust(d, method="average")
plot(fit.average, main="Average Linkage Clustering")
```

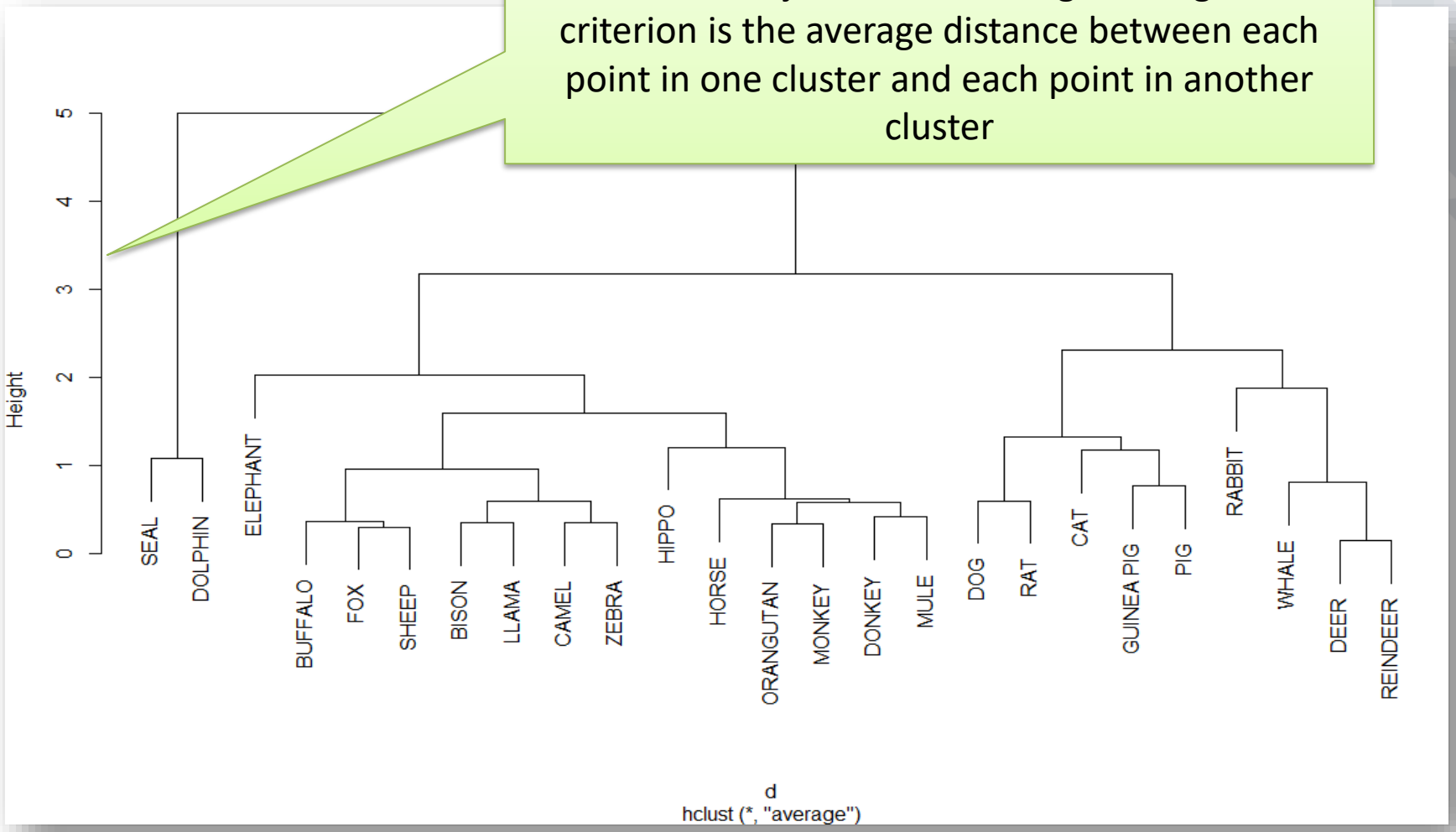


Dendrogram

- Dendrogram displays how items are combined into clusters and is read from the bottom up.
- Each observation starts as its own cluster.
- Two the observations that are closest are combined
- Then the clustered observations are combined with the closest further
- This continues until all observations are combined into a single cluster

Dendrogram

Height indicates the criterion value at which clusters are joined. For average linking, this criterion is the average distance between each point in one cluster and each point in another cluster



NbClust Package

- NbClust package offers many indices for finding the best number of clusters. Function NbClust ( ) is provided.
- The indices might not be consistent with each other, but the results can be a guide for selecting the optimal value of number of clusters

Syntax : NbClust(data, distance, min.nc = 2, max.nc = 15, method, …)

Where

data : data frame / matrix

distance : Distance method

min.nc : minimum number of clusters

max.nc : maximum number of clusters to be tried

method : method of linkage of clusters

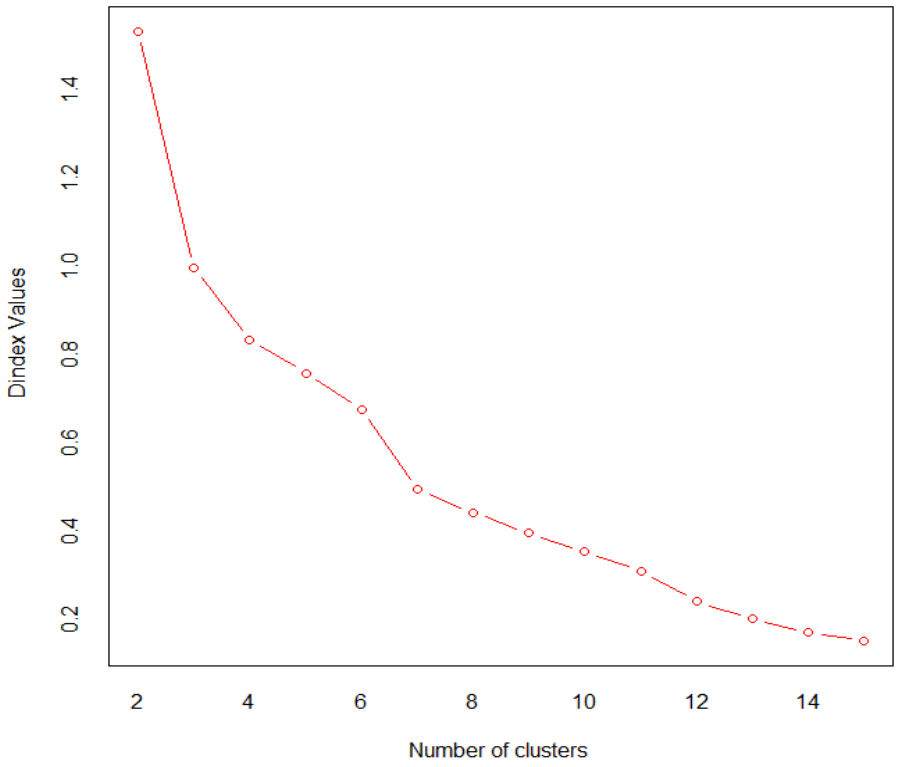
Program & Output : NBClust( )

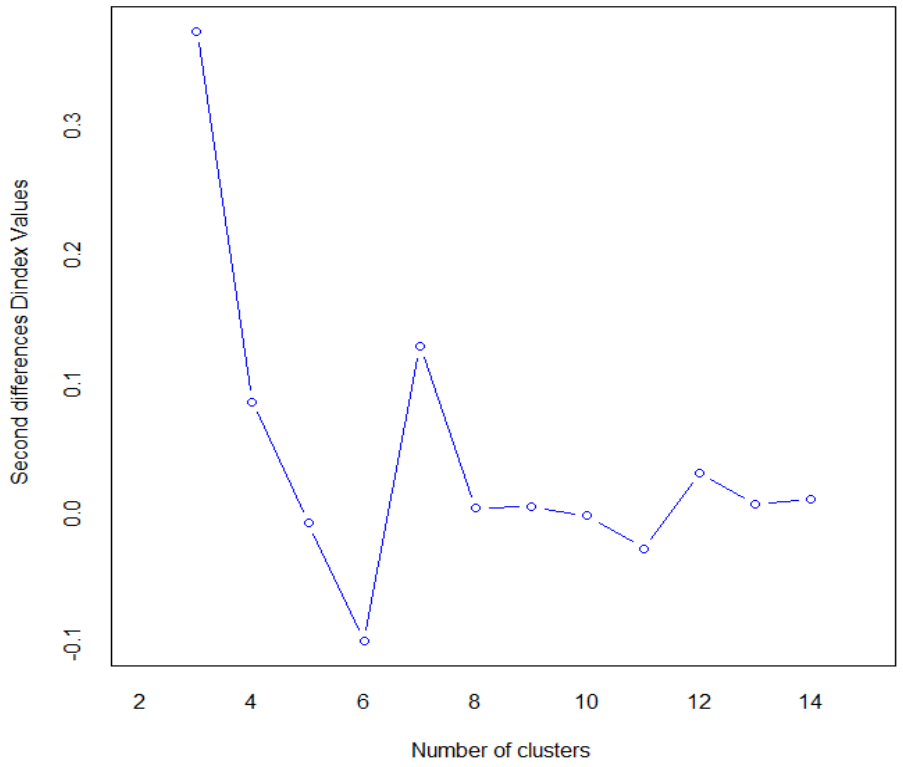
```
library(NbClust)
nc <- NbClust(milk.scaled, distance="euclidean",
              min.nc=2, max.nc=15, method="average")
```

```
*******************************************************************
* Among all indices:
* 2 proposed 2 as the best number of clusters
* 10 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 2 proposed 11 as the best number of clusters
* 1 proposed 12 as the best number of clusters
* 2 proposed 13 as the best number of clusters
* 2 proposed 14 as the best number of clusters
* 3 proposed 15 as the best number of clusters

                   ***** Conclusion *****

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


*******************************************************************
```

Output of Graphs

```
*** : 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.
```





Sane's STATS Academy of Statistics

Program & Output: Best No. of Clusters

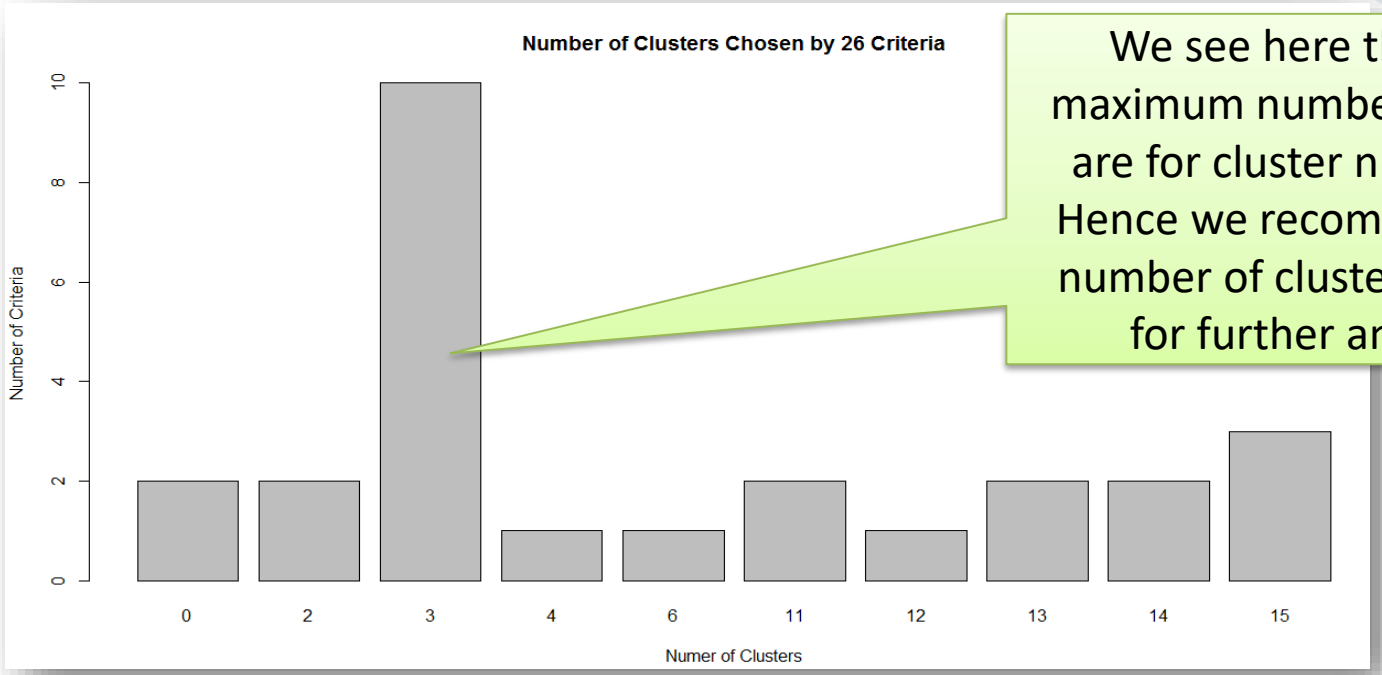
- Best.nc object gives the Best number of clusters proposed by each index and the corresponding index value.

```
> nc$Best.nc
                     KL      CH Hartigan     CCC   Scott  Marriot   TrCovW  TraceW Friedman   Rubin  Cindex
Number_clusters 3.0000 15.0000   3.0000 13.0000 11.0000   3.0000   3.0000  3.0000   11.000 12.0000 15.0000
Value_Index     4.9442 73.3096  26.2519  4.1918 62.7375 884.4857 976.9965 38.7123 1971.538 -6.8448  0.3739
                     DB Silhouette   Duda PseudoT2  Beale Ratkowsky    Ball PtBiserial   Frey McClain
Number_clusters 14.0000    14.0000 3.0000   3.0000 4.0000    3.0000  3.0000     3.0000 2.0000  2.0000
Value_Index      0.2892     0.7003 0.3747  11.6841 0.2171    0.4979 29.3569     0.6743 1.5602  0.0838
                   Dunn Hubert SDindex Dindex    SDbw
Number_clusters 13.0000      0  6.0000      0 15.0000
Value_Index      0.8298      0  1.4481      0  0.0061
```

Program & Output: Best No. of Clusters

- Hence we extract the clustering indices along with best number of clusters proposed by each and plot it.

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



We see here that the maximum number of votes are for cluster number 3. Hence we recommend the number of clusters to be 3 for further analysis

Obtaining Clusters

- The cluster IDs for observations can be obtained with the function cutree()

Syntax : cutree (tree , k)

Where

tree : object produced by hclust()

k : Cluster number used to cut the tree

Sane's STATS Academy of Statistics

Program & Output : Cluster IDs

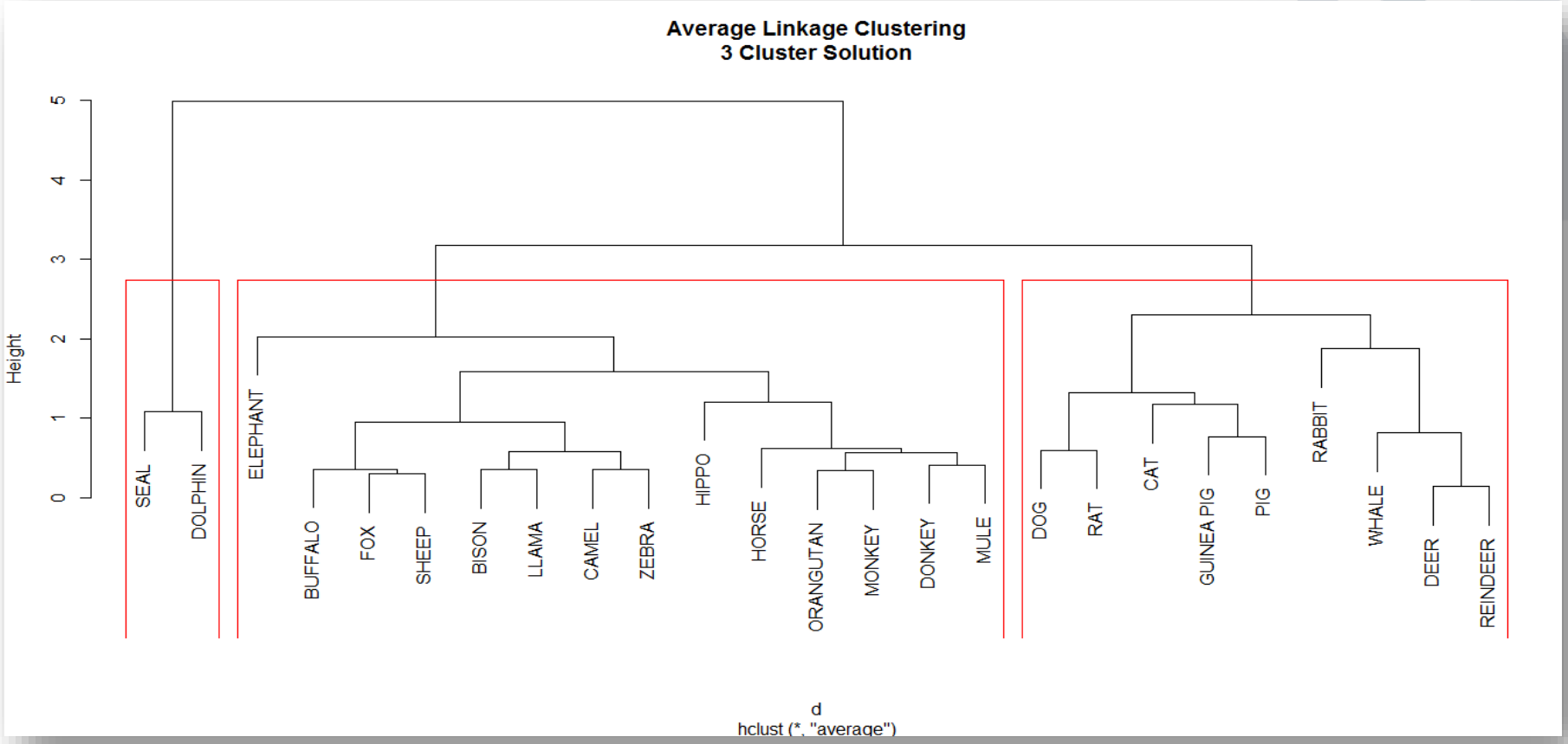
```
# Obtaining the final cluster solution
clusterID <- cutree(fit.average, k=3)
# Assigning ClusterID to observations
MilkClust <- data.frame(milk , clusterID)
```

	water	protein	fat	lactose	ash	clusterID
HORSE	90.1	2.6	1.0	6.9	0.35	1
ORANGUTAN	88.5	1.4	3.5	6.0	0.24	1
MONKEY	88.4	2.2	2.7	6.4	0.18	1
DONKEY	90.3	1.7	1.4	6.2	0.40	1
HIPPO	90.4	0.6	4.5	4.4	0.10	1
CAMEL	87.7	3.5	3.4	4.8	0.71	1
BISON	86.9	4.8	1.7	5.7	0.90	1
BUFFALO	82.1	5.9	7.9	4.7	0.78	1
GUINEA PIG	81.9	7.4	7.2	2.7	0.85	2
CAT	81.6	10.1	6.3	4.4	0.75	2
FOX	81.6	6.6	5.9	4.9	0.93	1
LLAMA	86.5	3.9	3.2	5.6	0.80	1
MULE	90.0	2.0	1.8	5.5	0.47	1
PIG	82.8	7.1	5.1	3.7	1.10	2
ZEBRA	86.2	3.0	4.8	5.3	0.70	1
SHEEP	82.0	5.6	6.4	4.7	0.91	1
DOG	76.3	9.3	9.5	3.0	1.20	2
ELEPHANT	70.7	3.6	17.6	5.6	0.63	1
RABBIT	71.3	12.3	13.1	1.9	2.30	2
RAT	72.5	9.2	12.6	3.3	1.40	2
DEER	65.9	10.4	19.7	2.6	1.40	2
REINDEER	64.8	10.7	20.3	2.5	1.40	2
WHALE	64.8	11.1	21.2	1.6	1.70	2
SEAL	46.4	9.7	42.0	0.0	0.85	3
DOLPHIN	44.9	10.6	34.9	0.9	0.53	3

Sane's STATS Academy of Statistics

Program & Output

```
plot(fit.average,main="Average Linkage Clustering\n3 Cluster Solution")
rect.hclust(fit.average, k=3)
```



Limitations of Hierarchical Clustering

- Hierarchical clustering requires the computation and storage of an n × n distance matrix. For very large datasets, this can be expensive and slow.
- The hierarchical algorithm makes only one pass through the data. This means that records that are allocated incorrectly early in the process cannot be reallocated subsequently.

Limitations of Hierarchical Clustering

- With respect to the choice of distance between clusters, single and complete linkage are robust to changes in the distance metric (e.g., Euclidean, statistical distance) as long as the relative ordering is kept.
- Hierarchical clustering is sensitive to outliers.

Sane's STATS Academy of Statistics

K-Means Clustering

K-Means

- Forming good clusters is to pre-specify a desired number of clusters, k, and to assign each case to one of k clusters so as to minimize a measure of dispersion (variation) within the clusters.
- The method divides the sample into a predetermined number k of non-overlapping clusters so that clusters are as homogeneous as possible with respect to the measurements used.

Algorithm

1. Start with k initial clusters (user chooses k). The starting points are chosen by software
2. At every step, each record is reassigned to the cluster with the “closest” centroid.
3. Re-compute the centroids of clusters that lost or gained a record, and repeat step 2.
4. Stop when moving any more records between clusters increases cluster dispersion.

K-means in R

- K-means can be implemented in R by function kmeans( ) in *stat* package.

Syntax : kmeans (x , centers , …)

Where

x : numeric matrix of data, or an object that can be coerced to such a matrix

centers : number of clusters

Program & Output

```
# Kmeans
k4<-kmeans(milk.scaled, centers=4)
```

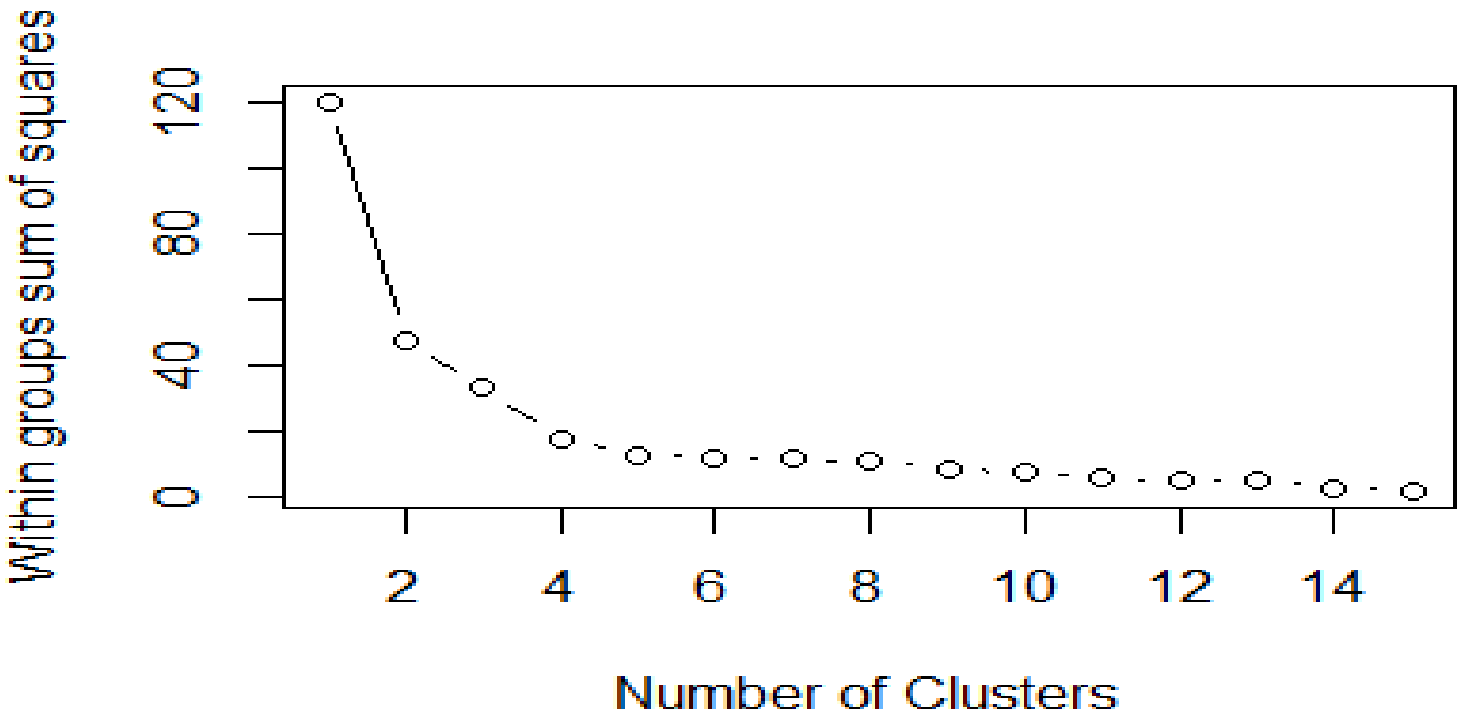
```
> k4
K-means clustering with 4 clusters of sizes 11, 6, 2, 6

Cluster means:
       water    protein        fat    lactose        ash
1  0.3544471 -0.1700432 -0.3793325  0.3299235 -0.07837109
2  0.8919289 -1.2216133 -0.7439312  0.9651552 -1.13544434
3 -2.5381667  1.0781517  2.6756044 -2.0100122 -0.34308960
4 -0.6956931  1.1739753  0.5475060 -0.9000109  1.39348787

Clustering vector:
     HORSE  ORANGUTAN     MONKEY     DONKEY      HIPPO      CAMEL      BISON    BUFFALO GUINEA PIG
         2          2          2          2          2          1          1          1          1
       CAT        FOX      LLAMA       MULE        PIG      ZEBRA      SHEEP        DOG   ELEPHANT
         1          1          1          2          1          1          1          4          1
    RABBIT        RAT       DEER   REINDEER      WHALE       SEAL    DOLPHIN
         4          4          4          4          4          3          3

Within cluster sum of squares by cluster:
[1] 9.4118196 1.7974564 0.5866378 5.9582975
 (between_SS / total_SS =  85.2 %)

Available components:

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

Within Sum of Squares

- Within Sum of Squares indicate the variation
- We can create a function to extract and plot it.
- Alternatively we can also use *NbClust()* function
- Lower the WSS better is the cluster variation.
- WSS can be extracted from the component *tot.withinSS* of **kmeans** object.

Program & Output

```
wssplot(milk.scaled)
```



Sane's STATS Academy of Statistics

Advantages of K-Means Clutering

- K-means clustering can handle larger datasets than hierarchical cluster approach.
- Observations are not permanently committed to any cluster but, they are changed at every iteration.

Limitations

- All the variables have to be continuous / numeric
- Clusters are severely affected by outliers
- Clusters are sensitive to initialization
- Clusters obtained are of differing densities