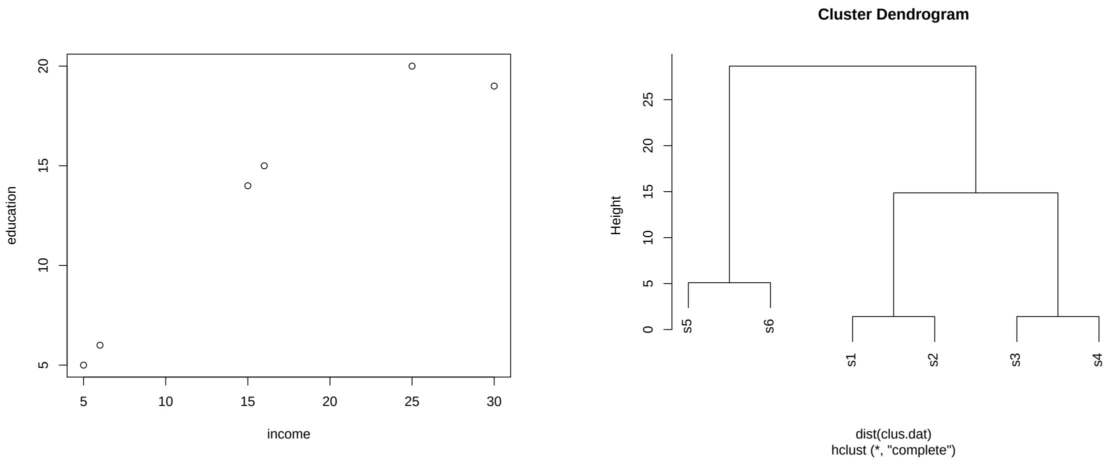
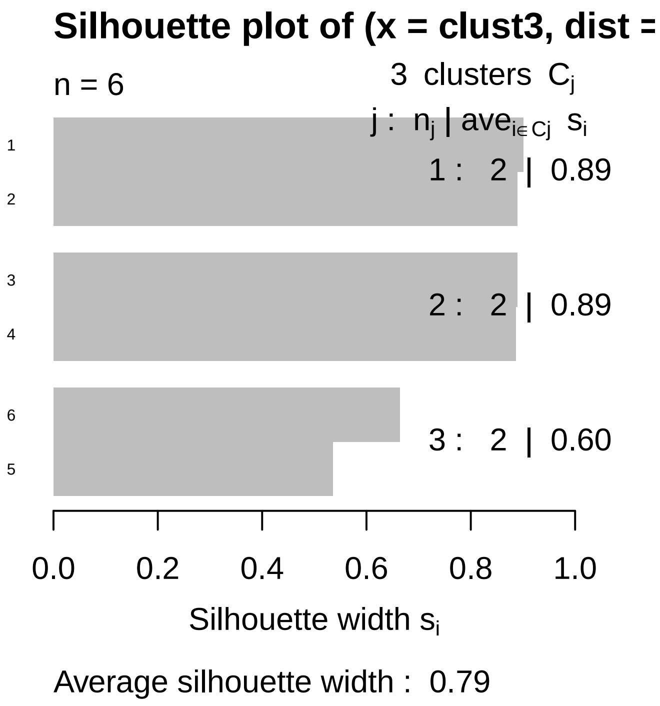
CLUSTER ANALYSIS

```
> ########################################################################
> #cluster analysis with toy example
> ########################################################################
> library(cluster)
> clus.dat <- matrix(c(5,5,6,6,15,14,16,15,25,20,30,19),nrow = 6, ncol=2,byrow=TRUE)
> rownames(clus.dat) <- c("s1","s2","s3","s4","s5","s6")
> colnames(clus.dat) <- c("income","education")
> clus.dat

   income education
s1      5         5
s2      6         6
s3     15        14
s4     16        15
s5     25        20
s6     30        19

> clus.compl<-hclust(dist(clus.dat), method = "complete") #or single, average, centroid, ward
> # names(clus.compl)

> par(mfrow=c(1,2))
> par(mar=c(5,5,5,5))
> plot(clus.dat)    #plot data
> plot(clus.compl,cex=0.5)  #plot dendogram
```



```
> clus.compl$merge  #shows the iterations

     [,1] [,2]
[1,]   -1   -2
[2,]   -3   -4
[3,]   -5   -6
[4,]    1    2
[5,]    3    4

> clust3 <- cutree(clus.compl, k=3) # cut tree into 3 clusters
> clust3
```

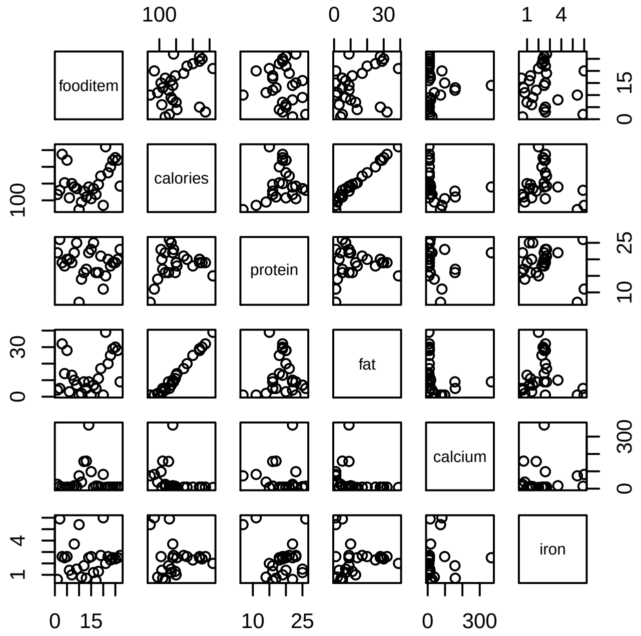
```
s1 s2 s3 s4 s5 s6
 1  1  2  2  3  3

> dd = dd= dist(clus.dat) #computes the distance matrix
> silout= silhouette(clust3,dd)  #computes information for silhouette plot for 3 clusters

> plot(silout,cex.main=1,cex = 0.5)  #plot silhouette
```



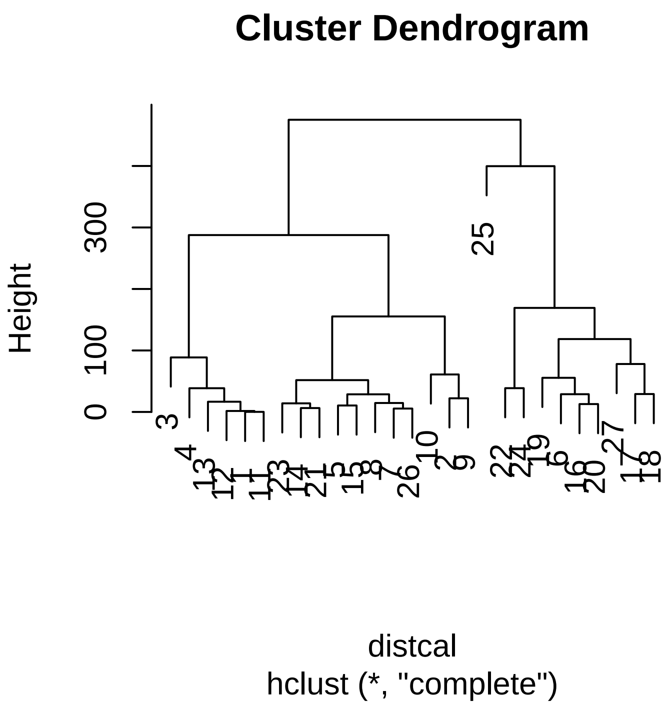
```
> ########################################################################
> #cluster analysis with calories etcetera of food items
> ########################################################################
>
> dcal <- read.table("C:/R/rmmva/calories.txt", header=T, quote="\"")
> attach(dcal)
> dcal

             fooditem calories protein fat calcium iron
1         Braised_beef      340      20  28       9  2.6
2            Hamburger      245      21  17       9  2.7
3           Roast_beef      420      15  39       7  2.0
4           Beef_steak      375      19  32       9  2.6
5          Canned_beef      180      22  10      17  3.7
6      Broiled_chicken      115      20   3       8  1.4
7       Canned_chicken      170      25   7      12  1.5
8           Beef_heart      160      26   5      14  5.9
9       Roast_lamb_leg      265      20  20       9  2.6
10 Roast_lamb_shoulder      300      18  25       9  2.3
11          Smoked_ham      340      20  28       9  2.5
12          Roast_pork      340      19  29       9  2.5
```

```
13      Simmered_pork     355      19  30        9  2.4
14        Beef_tongue     205      18  14        7  2.5
15        Veal_cutlet     185      23   9        9  2.7
16     Baked_bluefish     135      22   4       25  0.6
17          Raw_clams      70      11   1       82  6.0
18       Canned_clams      45       7   1       74  5.4
19    Canned_crabmeat      90      14   2       38  0.8
20      Fried_haddock     135      16   5       15  0.5
21  Broiled_mackerel     200      19  13        5  1.0
22    Canned_mackerel     155      16   9      157  1.8
23        Fried_perch     195      16  11       14  1.3
24      Canned_salmon     120      17   5      159  0.7
25    Canned_sardines     180      22   9      367  2.5
26        Canned_tuna     170      25   7        7  1.2
27     Canned_shrimp      110      23   1       98  2.6
```

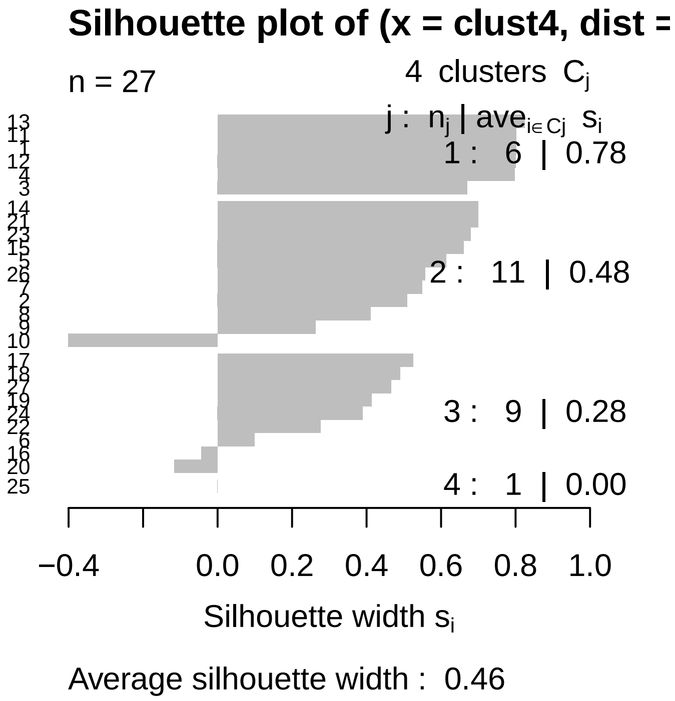
```
>   plot(dcal)
```



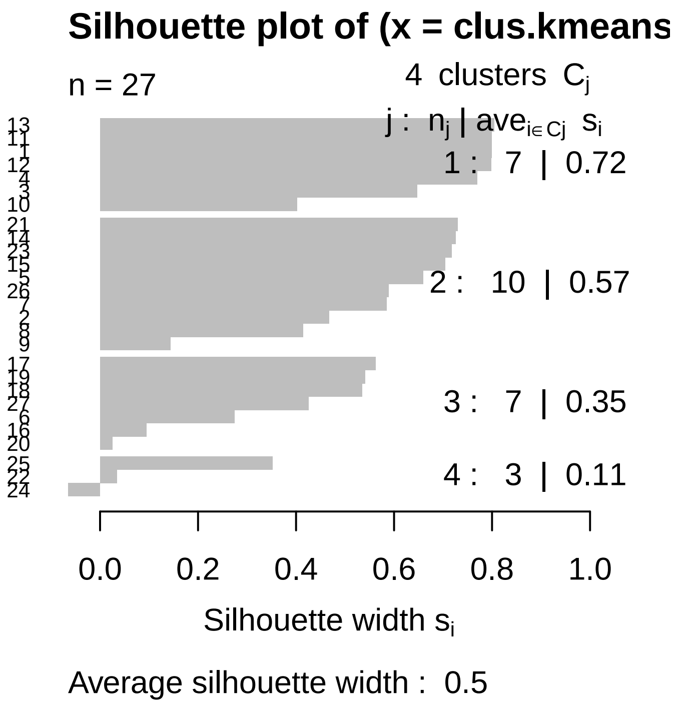
```
> #########################################################################
> #complete linkage
> #########################################################################
> distcal <- dist(dcal)
> # distcal
> clus.compl<-hclust(distcal, method = "complete") #or single, average, centroid, ward
> #names(clus.compl)

>   plot(clus.compl,cex=0.5)  #plot dendogram
```



```
>   #clus.compl$merge  #shows the iterations
>   clust4 <- cutree(clus.compl, k=4) # cut tree into 4 clusters
> clust4

 [1] 1 2 1 1 2 3 2 2 2 2 1 1 1 2 2 3 3 3 3 3 2 3 2 3 4 2 3

>   silout= silhouette(clust4,distcal)  #computes information for silhouette plot for 4 clusters

>   plot(silout,cex = 0.7)  #plot silhouette
```



```
> #########################################################################
> #k means clustering
> #########################################################################
> # 3 cluster solution and random initial clusters
> clus.kmeans <- kmeans(dcal[,-1], 4) # 3 cluster solution and random initial clusters
> clus.kmeans

K-means clustering with 4 clusters of sizes 7, 10, 7, 3

Cluster means:
  calories  protein       fat    calcium     iron
1 352.8571 18.57143 30.142857   8.714286 2.414286
2 197.5000 21.50000 11.300000  10.300000 2.510000
3 100.0000 16.14286  2.428571  48.571429 2.471429
4 151.6667 18.33333  7.666667 227.666667 1.666667

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

Within cluster sum of squares by cluster:
[1]  8433.126 10712.669 14706.660 30972.313
 (between_SS / total_SS =  84.9 %)

Available components:

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

> ss = silhouette(clus.kmeans$cluster,distcal)
>   plot(ss,cex = 0.7)  #plot silhouette
```



```
> ########################################################################
> #PAM partitioning around mediods
> ########################################################################
>
> library(MASS)
> library(cluster)
> library(fpc)
> p=pam(dcal[,-1],3)
> p

Medoids:
     ID calories protein fat calcium iron
[1,] 11      340      20  28       9  2.5
[2,]  7      170      25   7      12  1.5
[3,] 27      110      23   1      98  2.6
Clustering vector:
 [1] 1 2 1 1 2 2 2 2 2 1 1 1 1 1 2 2 2 3 3 3 2 2 3 2 3 3 2 3
Objective function:
   build     swap
44.21261 44.02536

Available components:
 [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"  "clusinfo"   "silinfo"
 [8] "diss"       "call"       "data"

> # plot(p,cex=0.5,cex.main=0.5) #creates (among others) silhouette plots
> #######################################################################
> #Mixture modelling
> #######################################################################
> library(FisherEM)
> #remove canned sardines
> dcalwcs = subset(dcal,calcium < 360)
> #dcalwcs
>
> #only seems to work if nr of variables larger than nr of clusters
> #the function fem fits a finite mixture model and at the same time
> #reduces the number of dimensions to plot the results in 2 dimensions
>
>
> #res = fem(dcalwcs[,-1],3,model="AkB")
> res = fem(dcalwcs[,-1],3,model="all")
```

```
 model: DkBk    bic: -356.1142
 model: DkB    bic: -388.0045
 model: DBk    bic: -400.2387
 model: DB    bic: -421.7061
 model: AkjBk    bic: -368.9642
 model: AkjB    bic: -407.1148
 model: AkBk    bic: -403.2533
 model: AkB    bic: -411.1244
 model: AjBk    bic: -409.9689
 model: AjB    bic: -440.6903
 model: ABk    bic: -410.0535
 model: AB    bic: -450.4718
 The best model is: DkBk with a bic equal to: -356.1142
```

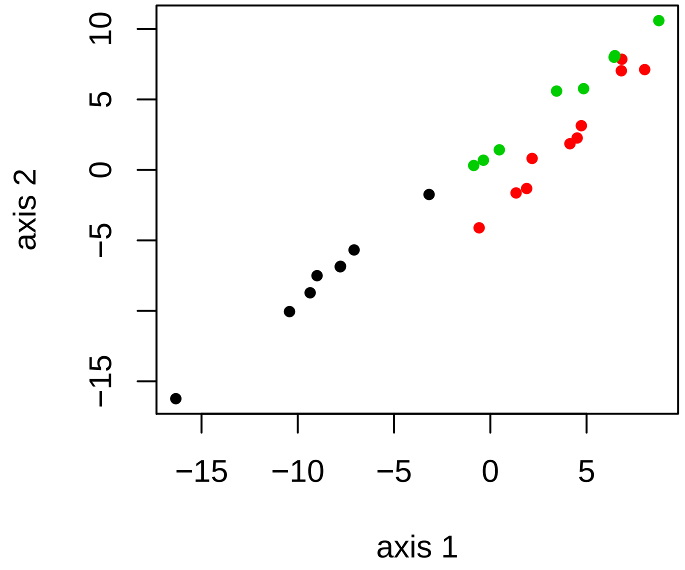
```
> res$cls
```

```
 [1] 1 3 1 1 3 2 3 3 1 1 1 1 1 3 3 2 2 2 2 2 3 2 2 2 3 2
```

```
> round(res$P,5) #P are the posterior probabilities, print 5 decimals
```

```
         [,1]    [,2]    [,3]
 [1,] 1.00000 0.00000 0.00000
 [2,] 0.00434 0.00000 0.99566
 [3,] 1.00000 0.00000 0.00000
 [4,] 1.00000 0.00000 0.00000
 [5,] 0.00000 0.00000 1.00000
 [6,] 0.00000 0.99870 0.00130
 [7,] 0.00000 0.00053 0.99947
 [8,] 0.00000 0.00908 0.99092
 [9,] 0.99939 0.00000 0.00061
[10,] 1.00000 0.00000 0.00000
[11,] 1.00000 0.00000 0.00000
[12,] 1.00000 0.00000 0.00000
[13,] 1.00000 0.00000 0.00000
[14,] 0.00000 0.00000 1.00000
[15,] 0.00000 0.00040 0.99960
[16,] 0.00000 0.99990 0.00010
[17,] 0.00000 1.00000 0.00000
```

```
[18,] 0.00000 1.00000 0.00000
[19,] 0.00000 1.00000 0.00000
[20,] 0.00000 1.00000 0.00000
[21,] 0.00000 0.00000 1.00000
[22,] 0.00000 1.00000 0.00000
[23,] 0.00000 1.00000 0.00000
[24,] 0.00000 1.00000 0.00000
[25,] 0.00000 0.00037 0.99963
[26,] 0.00000 1.00000 0.00000

> res$prms$prop

[1] 0.3078357 0.3849612 0.3072030

> res$prms$my#estimated mean in the original space

         [,1]     [,2]      [,3]      [,4]     [,5]
[1,] 341.8283 18.75112 28.869238  8.750116 2.437630
[2,] 117.0466 16.20941  4.201328 66.952491 2.113502
[3,] 189.3736 22.37073 10.252225  9.995831 2.646188

> res$prms$mean  #estimated mean in the subspace

          [,1]      [,2]
[1,] -8.866252 -7.951395
[2,]  3.988375  2.308575
[3,]  3.643949  5.055173

>   plot.fem(res,dcalwcs[,-1])
```



```
> ##########################################################################
> #Model based clustering
> ##########################################################################
>
>
> library(mclust)
> # The R function Mclust performs model-based clustering for a range of models
> # and a variety of values of k:
>
> mclustout <- Mclust(dcalwcs[,-1], G=2:9)
> # By default, the models considered are:
> # "EII": spherical, equal volume
> # "VII": spherical, unequal volume
> # "EEI": diagonal, equal volume and shape
> # "VEI": diagonal, varying volume, equal shape
> # "EVI": diagonal, equal volume, varying shape
> # "VVI": diagonal, varying volume and shape
> # "EEE": ellipsoidal, equal volume, shape, and orientation
> # "EEV": ellipsoidal, equal volume and equal shape
> # "VEV": ellipsoidal, equal shape
> # "VVV": ellipsoidal, varying volume, shape, and orientation
>
> # Plotting the BIC values:
> plot(mclustout, data=dcalwcs, what="BIC")
> mclustout

'Mclust' model object:
 best model: ellipsoidal, equal shape (VEV) with 6 components

> mclustout$classification

 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 26 27
 1  2  1  1  3  4  3  3  2  2  1  1  1  2  3  4  5  5  4  4  2  6  2  6  3  5

> names(mclustout)

 [1] "call"           "modelName"      "n"              "d"              "G"
 [6] "BIC"            "bic"            "loglik"         "df"             "parameters"
[11] "classification" "uncertainty"    "z"

> round(mclustout$parameters$pro,2)

[1] 0.23 0.23 0.19 0.15 0.12 0.08

> round(mclustout$parameters$mean,2)

           [,1]   [,2]  [,3]   [,4]  [,5]   [,6]
calories 361.67 235.00 173.0 118.75 75.00 137.50
protein   18.67  18.67  24.2  18.00 13.67  16.50
fat       31.00  16.67   7.6   3.50  1.00   7.00
calcium    8.67   8.83  11.8  21.50 84.67 158.00
iron       2.43   2.07   3.0   0.82  4.67   1.25

> # This gives the probabilities of belonging to each cluster for every object:
> round(mclustout$z,2)
```

```
   [,1] [,2] [,3] [,4] [,5] [,6]
1     1    0    0    0    0    0
2     0    1    0    0    0    0
3     1    0    0    0    0    0
4     1    0    0    0    0    0
5     0    0    1    0    0    0
6     0    0    0    1    0    0
7     0    0    1    0    0    0
8     0    0    1    0    0    0
9     0    1    0    0    0    0
10    0    1    0    0    0    0
11    1    0    0    0    0    0
12    1    0    0    0    0    0
13    1    0    0    0    0    0
14    0    1    0    0    0    0
15    0    0    1    0    0    0
16    0    0    0    1    0    0
17    0    0    0    0    1    0
18    0    0    0    0    1    0
19    0    0    0    1    0    0
20    0    0    0    1    0    0
21    0    1    0    0    0    0
22    0    0    0    0    0    1
23    0    1    0    0    0    0
24    0    0    0    0    0    1
26    0    0    1    0    0    0
27    0    0    0    0    1    0
```