Class 07 Unsupervised learning

Andre Modolo

In this class we will explore clustering and dimensional reduction methods.

## K-means (K=number of clusters)

Make up input data where we know what the answer should be.

tmp <- c(rnorm(30, -3), rnorm(30, 3))  
# make it into a 2 dimensional thing  
x <- cbind(x=tmp, y=rev(tmp))  
# rev(tmp) flips the vector   
rev(tmp)

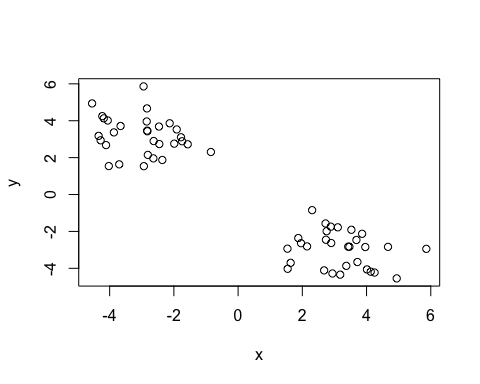
[1] 2.8991538 1.6362968 3.3686467 3.6842056 1.8736334 1.9591686  
 [7] 3.4654623 3.1048232 2.7231510 2.3056306 2.7574076 5.8610264  
[13] 3.1792991 3.8619907 3.9605646 3.5274241 4.1295692 3.4292391  
[19] 2.6792261 2.1474762 4.2498852 2.7357443 1.5366285 4.0127872  
[25] 1.5428144 3.7150114 2.9386080 4.9392024 2.8872510 4.6681940  
[31] -2.8379967 -1.7434181 -4.5467671 -4.2809171 -3.6596423 -4.0261784  
[37] -4.0599653 -2.9345128 -2.4536508 -4.2290156 -2.8111426 -4.1120691  
[43] -2.8259391 -4.1831423 -1.9113372 -2.8461242 -2.1304977 -4.3433768  
[49] -2.9439570 -1.9882731 -0.8469759 -1.5692192 -1.7780004 -2.8312950  
[55] -2.6384763 -2.3601613 -2.4634074 -3.8673657 -3.7020591 -2.6293157

head(x)

x y  
[1,] -2.629316 2.899154  
[2,] -3.702059 1.636297  
[3,] -3.867366 3.368647  
[4,] -2.463407 3.684206  
[5,] -2.360161 1.873633  
[6,] -2.638476 1.959169

Quick plot of x to see the 2 groups around (-3, 3) and (3, -3)

plot(x)



Use the kmeans() function setting k to 2 and nstart=20 (do the picking points and finding the distances to decide a potential cluster 20 times before deciding a winning set of clusters)

km <- kmeans(x,center=2, nstart=20 )  
km

K-means clustering with 2 clusters of sizes 30, 30  
  
Cluster means:  
 x y  
1 -2.985140 3.192651  
2 3.192651 -2.985140  
  
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 2 2 2 2 2 2 2 2  
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2  
  
Within cluster sum of squares by cluster:  
[1] 58.97352 58.97352  
 (between\_SS / total\_SS = 90.7 %)  
  
Available components:  
  
[1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
[6] "betweenss" "size" "iter" "ifault"

Clustering means: gives us the mean point of each cluster

km$size

[1] 30 30

Size of the clusters found: 30 and 30 Clustering vector: lables each component of the vector as the first or second cluster

Q. What component of your result object details? - cluster assignment/membership (1 or 2 in this case)

km$cluster

[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 2 2 2 2 2 2 2 2  
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

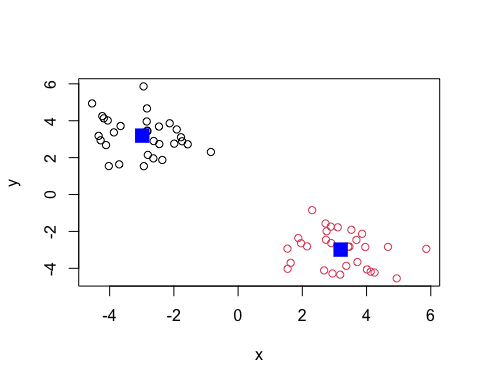
-Cluster center?

km$center

x y  
1 -2.985140 3.192651  
2 3.192651 -2.985140

Q. plot x colored by the kmeans cluster assignment and add cluster centers as blue points

plot(x, col=km$cluster)  
points(km$centers, col="blue", pch=15, cex=2)

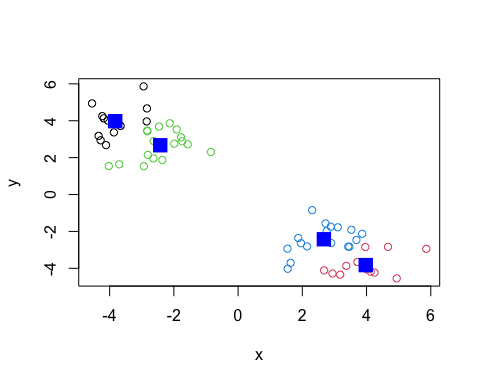


What if I ask for more than 2 clusters?

km4 <- kmeans(x, 4, nstart=20)  
km4

K-means clustering with 4 clusters of sizes 12, 12, 18, 18  
  
Cluster means:  
 x y  
1 -3.825862 3.975168  
2 3.975168 -3.825862  
3 -2.424659 2.670972  
4 2.670972 -2.424659  
  
Clustering vector:  
 [1] 3 3 1 3 3 3 3 3 3 3 3 1 1 3 1 3 1 3 1 3 1 3 3 1 3 1 1 1 3 1 2 4 2 2 2 4 2 4  
[39] 4 2 4 2 4 2 4 2 4 2 2 4 4 4 4 4 4 4 4 2 4 4  
  
Within cluster sum of squares by cluster:  
[1] 13.05430 13.05430 19.53628 19.53628  
 (between\_SS / total\_SS = 94.8 %)  
  
Available components:  
  
[1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
[6] "betweenss" "size" "iter" "ifault"

plot(x, col=km4$cluster)  
points(km4$centers, col="blue", pch=15, cex=2)



#Hierarchical Clustering

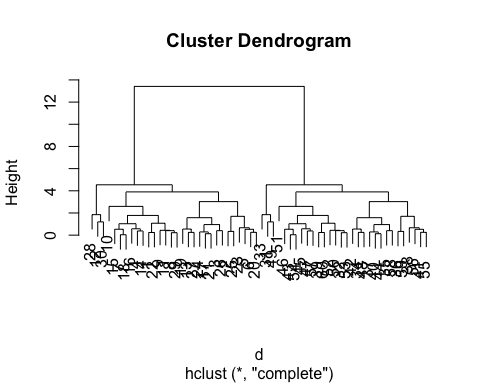
Super useful and widely employed clustering method which has the advantage over kmeans because it can show you a little something about the true nature of the clustering in your data You need to give it a “d” distance matrix as an input (how far apart the values are). get it using dist()

d <- dist(x)  
hc <- hclust(d)  
hc

Call:  
hclust(d = d)  
  
Cluster method : complete   
Distance : euclidean   
Number of objects: 60

There is a plot method for hcluster results:

plot(hc)

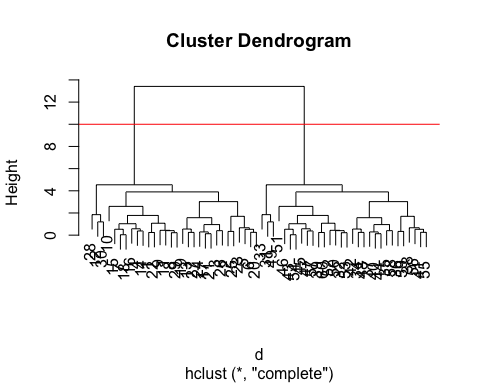


You get 2 overall branches with 1:30 on one branch and 31:60 on the other branch. This makes sense because in the vector we made the first 30 numbers have a set mean and the second 30 numbers have another set mean.

Long goal post = big jump between the things you grouped together and the next group.

How do I get an actual result out of this? cut the longest post, and you are left with “subtrees” in this case you are left with 2 subtrees.

plot(hc)  
#cut the tree with this line   
abline(h=10, col="red")

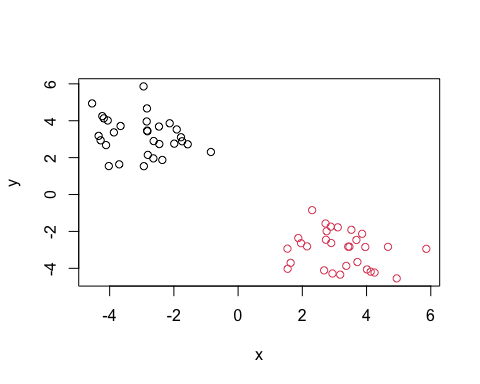


To get the cluster membership vector, I need to “cut my tree” to yield subtrees with the function cutree() with h=height to cut, **or** k= number of clusters you want after the cut

grps <- cutree(hc, h=10)  
grps

[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 2 2 2 2 2 2 2 2  
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

plot(x, col=grps)



# Principal Component Analysis (PCA)

The base R function to do PCA is called prcomp()

Import the food data from the 4 countries

url <- "https://tinyurl.com/UK-foods"  
x <- read.csv(url)  
dim(x)

[1] 17 5

There are 17 row and 5 columns

head(x)

X England Wales Scotland N.Ireland  
1 Cheese 105 103 103 66  
2 Carcass\_meat 245 227 242 267  
3 Other\_meat 685 803 750 586  
4 Fish 147 160 122 93  
5 Fats\_and\_oils 193 235 184 209  
6 Sugars 156 175 147 139

We can remove the x column and only get the 4 counties as columns by using this code

rownames(x) <- x[,1]  
x<- x[,-1]  
head(x)

England Wales Scotland N.Ireland  
Cheese 105 103 103 66  
Carcass\_meat 245 227 242 267  
Other\_meat 685 803 750 586  
Fish 147 160 122 93  
Fats\_and\_oils 193 235 184 209  
Sugars 156 175 147 139

Be careful with this approach because if you keep running it multiple times, it will keep removing the name of the first column

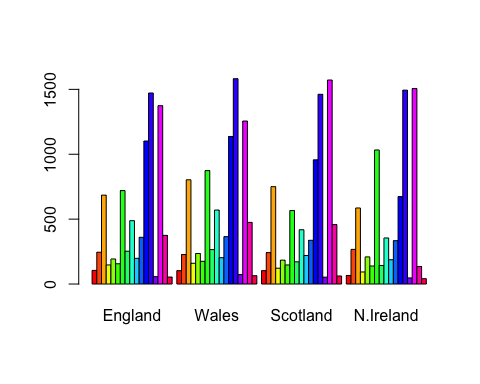
A more robust way of doing it would be using this method, setting the row name as 1, so you can rerun the code and it won’t delete any more column names

x <- read.csv(url, row.names=1)  
head(x)

England Wales Scotland N.Ireland  
Cheese 105 103 103 66  
Carcass\_meat 245 227 242 267  
Other\_meat 685 803 750 586  
Fish 147 160 122 93  
Fats\_and\_oils 193 235 184 209  
Sugars 156 175 147 139

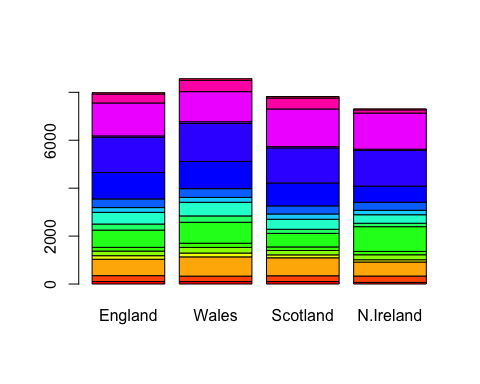
Spotting the major differences and trends using a bar plot

barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))



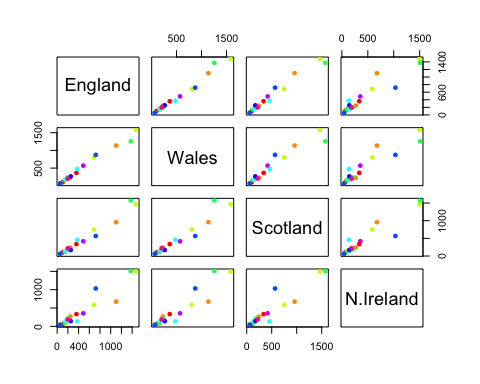
Doing beside=False you get this kind of bar plot

barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))



What about plotting it this way?

pairs(x, col=rainbow(10), pch=16)



What does it mean when a point lies on a diagonal of a given plot? This gives a matrix of scatterplots comparing the countries as an x and a y variable in each situation. This way you only have to look at bottom left or top right half depending in which country you want to be on which axis.

If the point lies on the diagonal of a scatterplot, this means that the two countries have a similar amount of consumption for that specific food group (color)

The main difference in food consumption between N. Ireland and the other countries is in the food colored blue

#PCA to the rescue

Take the transpose of x to flip the rows and columns

t(x)

Cheese Carcass\_meat Other\_meat Fish Fats\_and\_oils Sugars  
England 105 245 685 147 193 156  
Wales 103 227 803 160 235 175  
Scotland 103 242 750 122 184 147  
N.Ireland 66 267 586 93 209 139  
 Fresh\_potatoes Fresh\_Veg Other\_Veg Processed\_potatoes   
England 720 253 488 198  
Wales 874 265 570 203  
Scotland 566 171 418 220  
N.Ireland 1033 143 355 187  
 Processed\_Veg Fresh\_fruit Cereals Beverages Soft\_drinks   
England 360 1102 1472 57 1374  
Wales 365 1137 1582 73 1256  
Scotland 337 957 1462 53 1572  
N.Ireland 334 674 1494 47 1506  
 Alcoholic\_drinks Confectionery   
England 375 54  
Wales 475 64  
Scotland 458 62  
N.Ireland 135 41

Now do prcomp() and print out the summary

pca <- prcomp(t(x))  
summary(pca)

Importance of components:  
 PC1 PC2 PC3 PC4  
Standard deviation 324.1502 212.7478 73.87622 4.189e-14  
Proportion of Variance 0.6744 0.2905 0.03503 0.000e+00  
Cumulative Proportion 0.6744 0.9650 1.00000 1.000e+00

Proportion of Variance: 67.4% of all the variance is captured on the new axis made.

Cumulative Proportion: adding 2 or 3 PCs together you capture basically all the variance from the plots (ex: PC2 with 96.5%!)

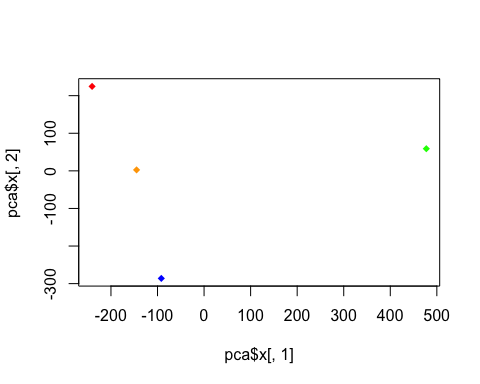
A “PCA plot” (a.k.a “Score Plot”, PC1vsPC2 plot, etc.)

pca$x

PC1 PC2 PC3 PC4  
England -144.99315 2.532999 -105.768945 2.842865e-14  
Wales -240.52915 224.646925 56.475555 7.804382e-13  
Scotland -91.86934 -286.081786 44.415495 -9.614462e-13  
N.Ireland 477.39164 58.901862 4.877895 1.448078e-13

Plot the PC1 vs PC2 and color the countries Irland is green

plot(pca$x[,1], pca$x[,2], col=c("orange","red", "blue", "green"), pch=18)



You see that N. Ireland is actually different than the other countries in their food consumption.

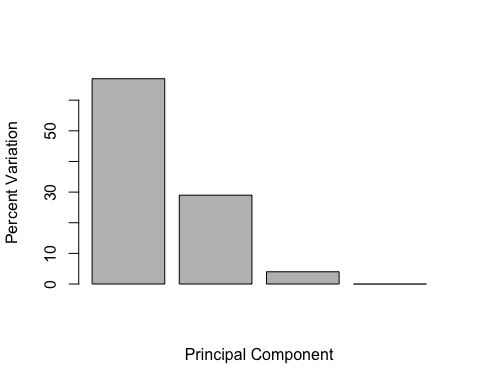
Below we can use the square of pca$sdev , which stands for “standard deviation”, to calculate how much variation in the original data each PC accounts for:

v <- round( pca$sdev^2/sum(pca$sdev^2) \* 100 )  
v

[1] 67 29 4 0

This information can be summarized in a plot of the variances (eigenvalues) with respect to the principal component number (eigenvector number), which is given below.

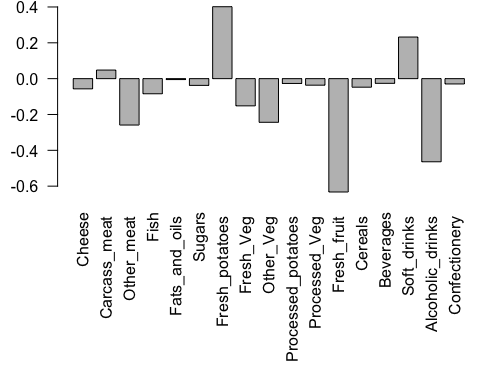
barplot(v, xlab="Principal Component", ylab="Percent Variation")



We can also consider the influence of each of the original variables upon the principal components (typically known as loading scores). This information can be obtained from the prcomp() returned $rotation component

Using PC1 we can get this barplot:

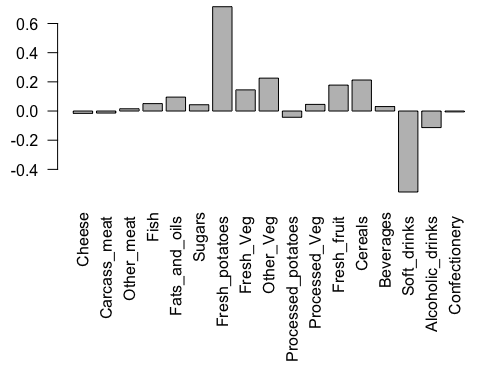
par(mar=c(10, 3, 0.35, 0))  
barplot( pca$rotation[,1], las=2 )



Now we can see what foods that make N. Ireland more different than the rest if the countries.

Generate a similar ‘loadings plot’ for PC2. What two food groups feature prominantely and what does PC2 maninly tell us about?

par(mar=c(10, 3, 0.35, 0))  
barplot( pca$rotation[,2], las=2 )



Using PC2 it tells us that N. Ireland eats more fresh potatoes and drinks less soft drinks