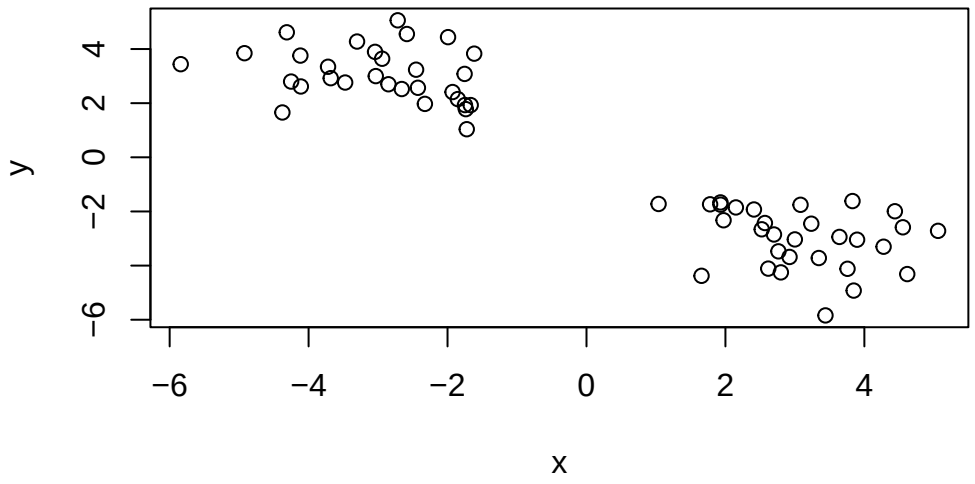
Class7

Liz

K-means Clustering

Let's make up some data to cluster.

```
tmp <- c(rnorm(30, -3), rnorm(30, 3) )
x <- cbind(x=tmp, y=rev(tmp))
plot(x)
```




```
rev( c("a", "b", "c"))
```

```
[1] "c" "b" "a"
```

The function to do k-means clustering in base R is called `kmeans()`. We give this our input data for clustering and the number of clusters we want `centers`.

```
km <- kmeans(x, centers=4, nstart=20)
km
```

```
K-means clustering with 4 clusters of sizes 13, 17, 17, 13

Cluster means:
          x         y
1 -2.057032  2.395831
2 -3.672211  3.566410
3  3.566410 -3.672211
4  2.395831 -2.057032

Clustering vector:
 [1] 2 2 1 1 1 1 2 2 2 2 1 1 1 1 2 1 1 2 2 2 2 2 2 1 2 2 2 1 1 2 3 4 4 3 3 3 4 3
[39] 3 3 3 3 3 4 4 3 4 4 4 4 3 3 3 3 4 4 4 4 3 3

Within cluster sum of squares by cluster:
[1]  8.427895 26.731148 26.731148  8.427895
 (between_SS / total_SS =  94.2 %)

Available components:

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

Q. What `component` of your result object details -cluster size?

```
km$cluster
```

```
 [1] 2 2 1 1 1 1 2 2 2 2 1 1 1 1 2 1 1 2 2 2 2 2 2 1 2 2 2 1 1 2 3 4 4 3 3 3 4 3
[39] 3 3 3 3 3 4 4 3 4 4 4 4 3 3 3 3 4 4 4 4 3 3
```

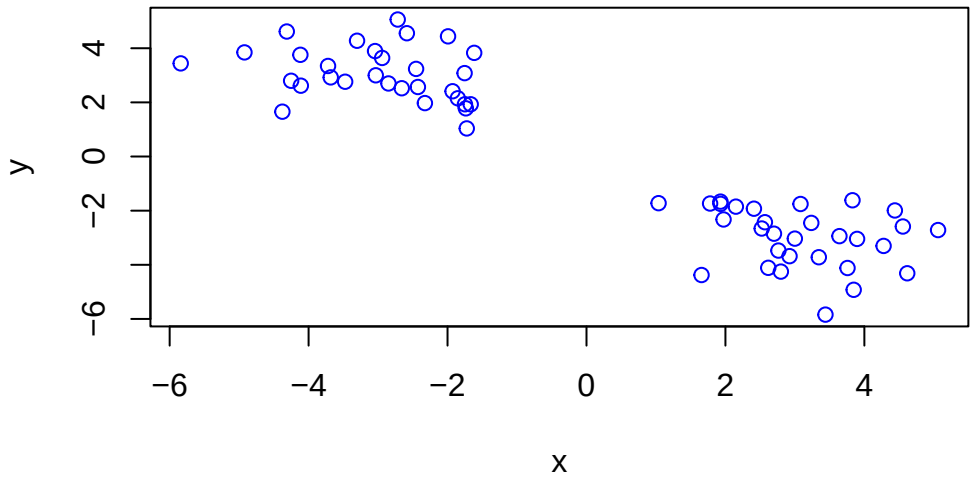
center?

```
km$center
```

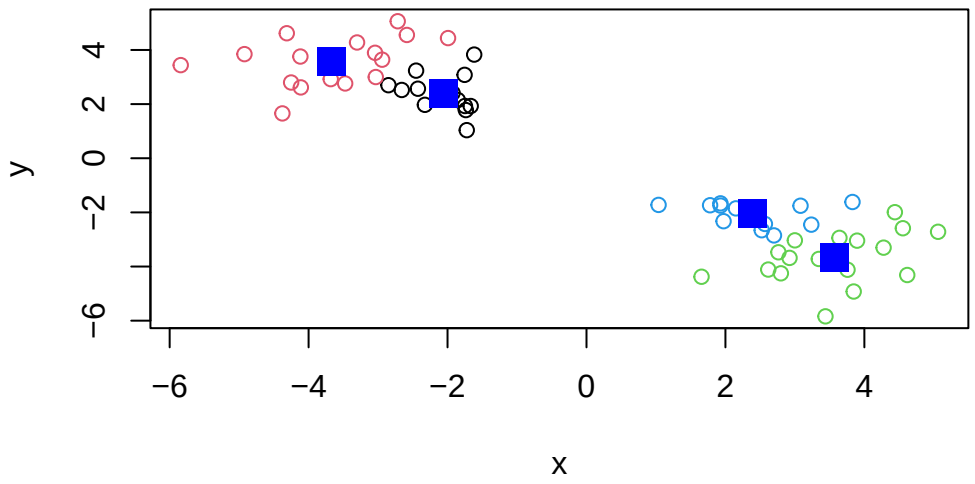
```
          x         y
1 -2.057032  2.395831
2 -3.672211  3.566410
3  3.566410 -3.672211
4  2.395831 -2.057032
```

cluster center as blue points

```
plot(x, col= "blue")
```




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



Hierarchial Clustering

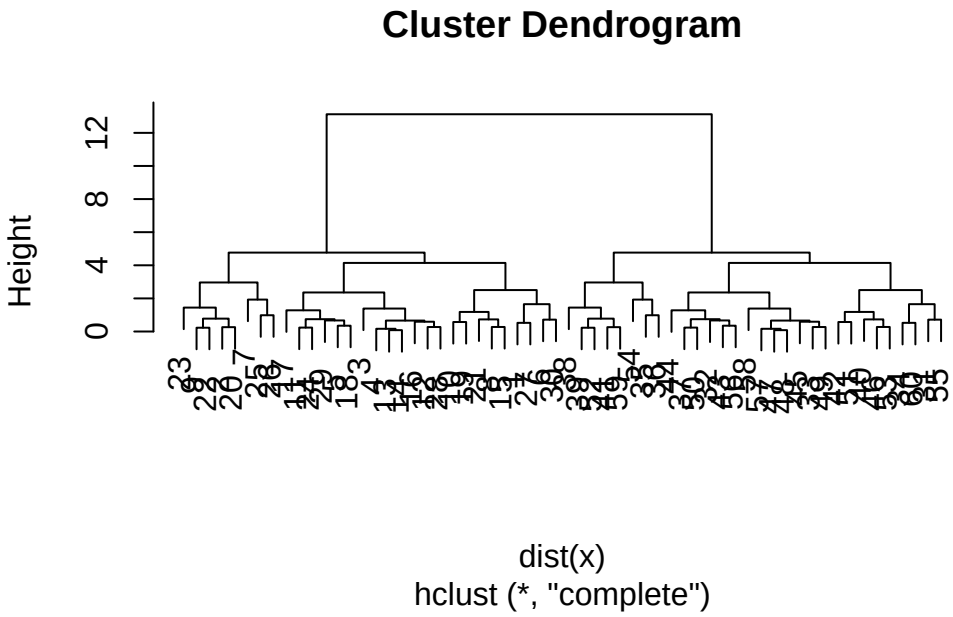
The `hclust()` function performs hierarchical clustering. The big advantage here is I don't need to tell it "k" the number of clusters. To run `hcluster()` I need to provide a distance

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

```
Call:
hclust(d = dist(x))

Cluster method   : complete
Distance         : euclidean
Number of objects: 60
```

```
plot(hc)
```



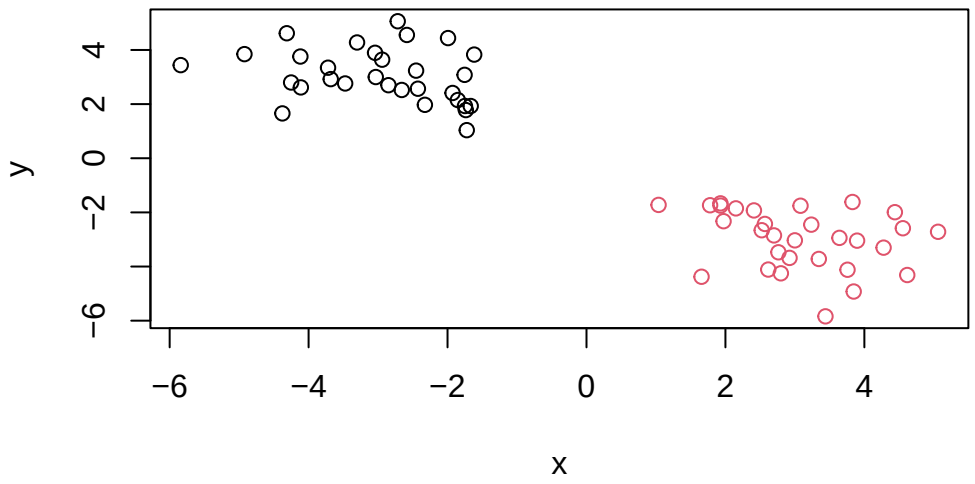
To get my "main" result (cluster members) I want to "cut" this tree to yields "branches" who's leaves are the members of the clusters.

```
grsp <- cutree(hc, k=2)
```

More often we will use `cutree()` with k=2 for example

Make a plot of our `hclust()` results i.e. our data colored bu cluster assignment!

```
plot(x, col = grsp)
```



Principal Component Analysis (PCA)

Read data for UK food trends from online

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

```
nrow(x)
```

```
[1] 17
```

```
ncol(x)
```

```
[1] 5
```

```
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
```

```
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
```

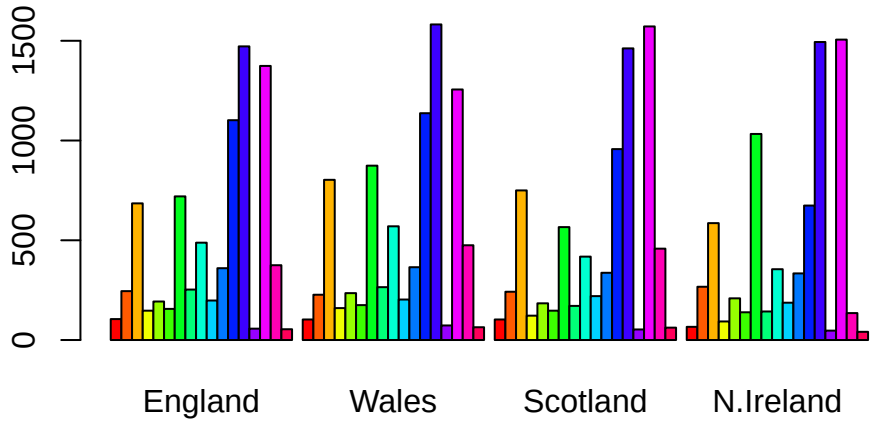
```
dim(x)
```

```
[1] 17  4
```

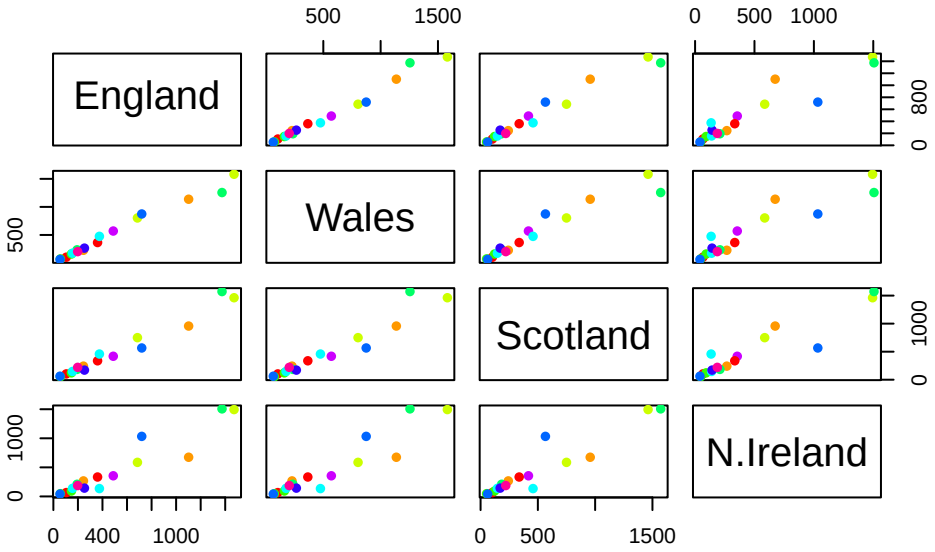
```
x <- read.csv(url, row.names=1)
head(x)
```

```
              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
```

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




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



```
log2(20/20)
```

```
[1] 0
```

```
log2(20/10)
```

```
[1] 1
```

```
(20/10)
```

```
[1] 2
```

```
log2(10/20)
```

```
[1] -1
```

##PCA to the rescue!

The main function in base R to do PCA is called `prcomp()`. One issue with the `prcomp()` function is that it expects the transpose of our data as input.

```
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
```

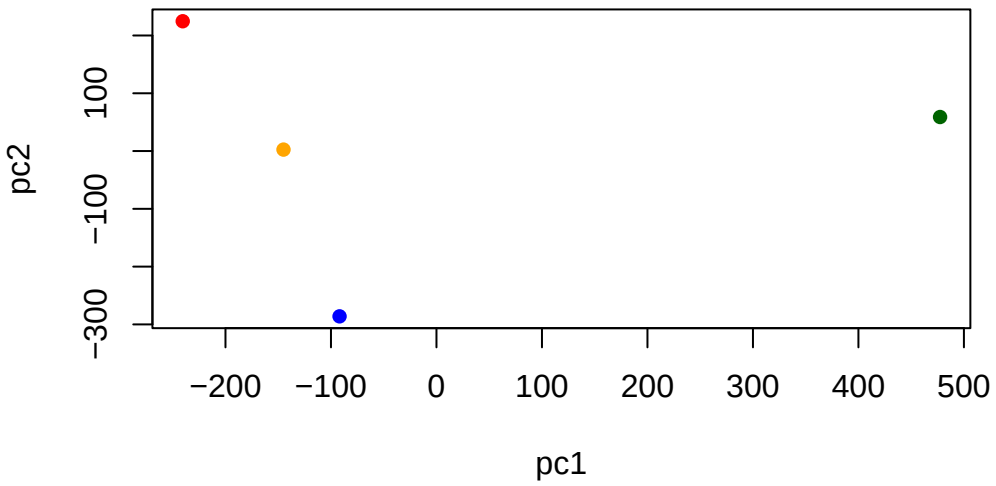
```
N.Ireland            334          674     1494        47         1506
          Alcoholic_drinks  Confectionery
England                375             54
Wales                  475             64
Scotland               458             62
N.Ireland              135             41
```

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

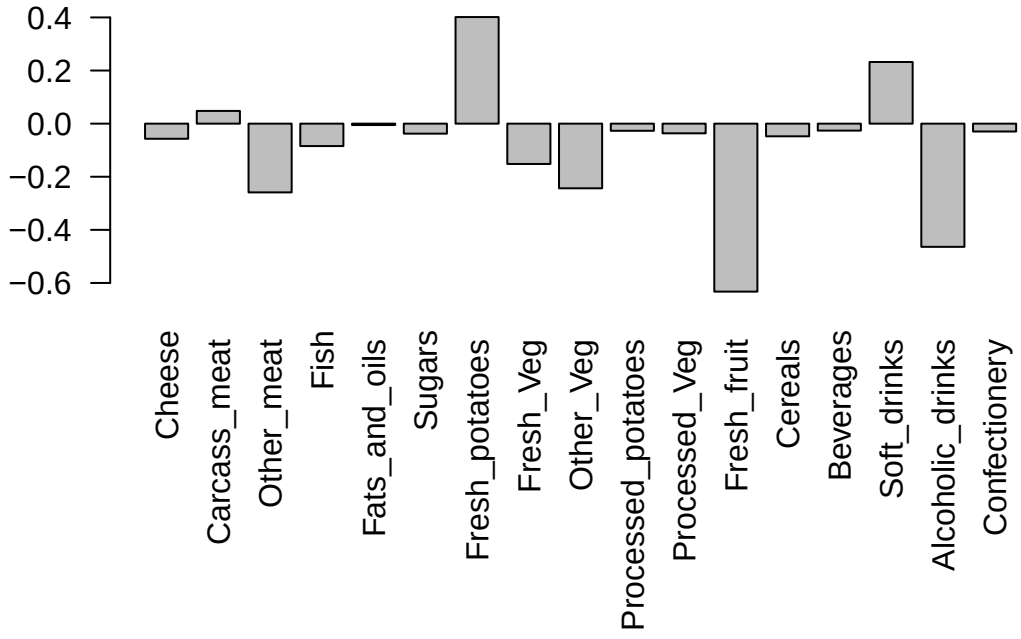
```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 4.189e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

The object returned by `prcomp()` has our results that include a $x component. This is our "scores" along the PCs (i.e. The plot our data along the new PC axis)

```r
plot(pca$x[,1], pca$x[,2],
     xlab= "pc1", ylab= "pc2",
     col=c("orange", "red", "blue","darkgreen"),
           pch=16)
```




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



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

```
[1] 67 29  4  0
```

```
z <- summary(pca)
z$importance
```

```
                             PC1       PC2      PC3           PC4
Standard deviation     324.15019 212.74780 73.87622 4.188568e-14
Proportion of Variance   0.67444   0.29052  0.03503 0.000000e+00
Cumulative Proportion    0.67444   0.96497  1.00000 1.000000e+00
```

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