Class 7: Hands on with Principal Component Analysis (PCA)

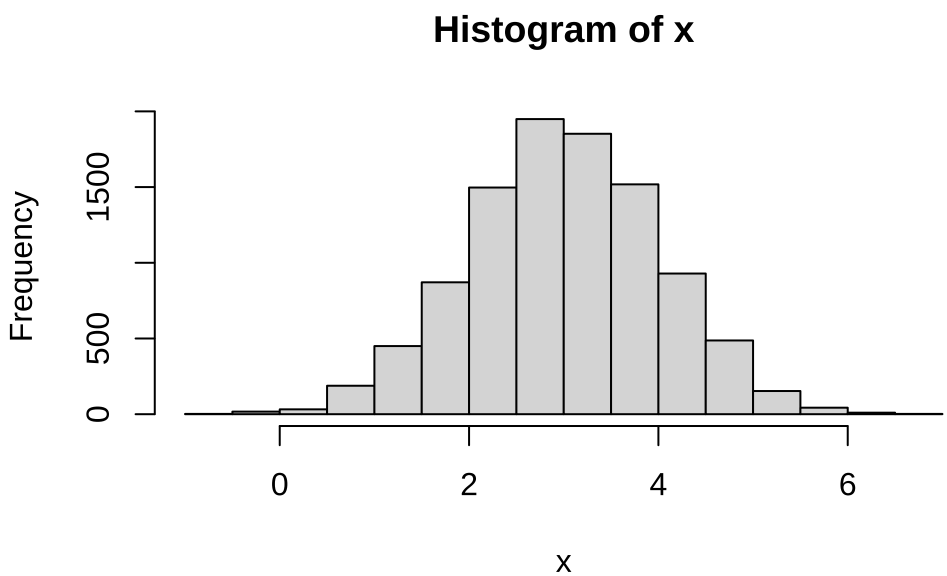
Liana Melikian (A16675734)

#Clustering

We will start today's lab with clustering methods, in particular so-called K-means. The main function for this in R is `kmeans()`

Let's try it on some made up data where we know what the answer should be.

```
x=rnorm(10000,mean=3)
hist(x)
```



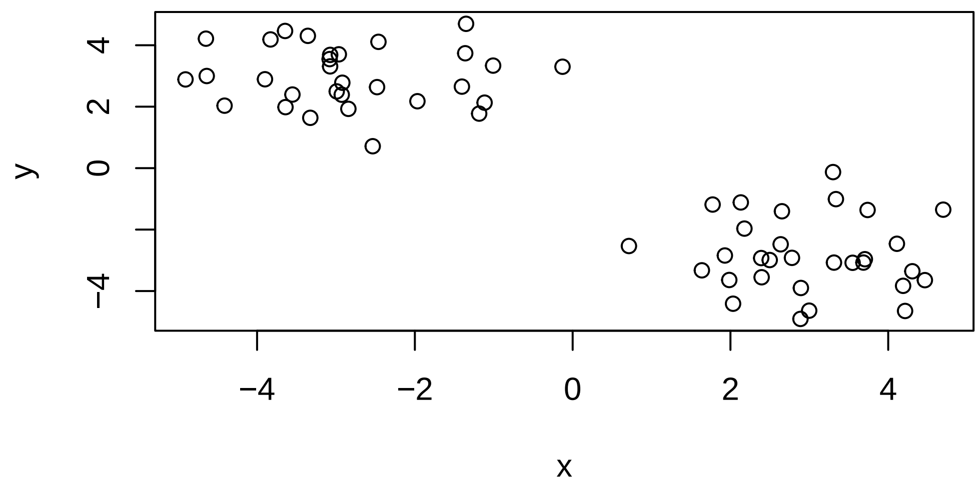
60 points

```
tmp=c(rnorm(30,mean=3), rnorm(30,-3))
x=cbind(x=tmp,y=rev(tmp))
head(x)
```

```
            x         y
[1,] 4.110307 -2.461553
[2,] 4.464750 -3.645201
[3,] 2.998319 -4.638297
[4,] 1.985344 -3.640050
[5,] 3.548596 -3.080086
[6,] 1.637930 -3.325448
```

We can pass this to the R `plot()` function for a quick.

```
plot(x)
```




```
k=kmeans(x,centers=2,nstart=20)
k
```

```
K-means clustering with 2 clusters of sizes 30, 30

Cluster means:
          x         y
1 -2.824148  2.970697
2  2.970697 -2.824148
```

```
Clustering vector:
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Within cluster sum of squares by cluster:
[1] 69.25431 69.25431
 (between_SS / total_SS =  87.9 %)

Available components:

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

Q1. How many points are in each cluster?

```r
k$size
```

```
[1] 30 30
```

Q2. Cluster membership?

```r
k$cluster
```

```
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

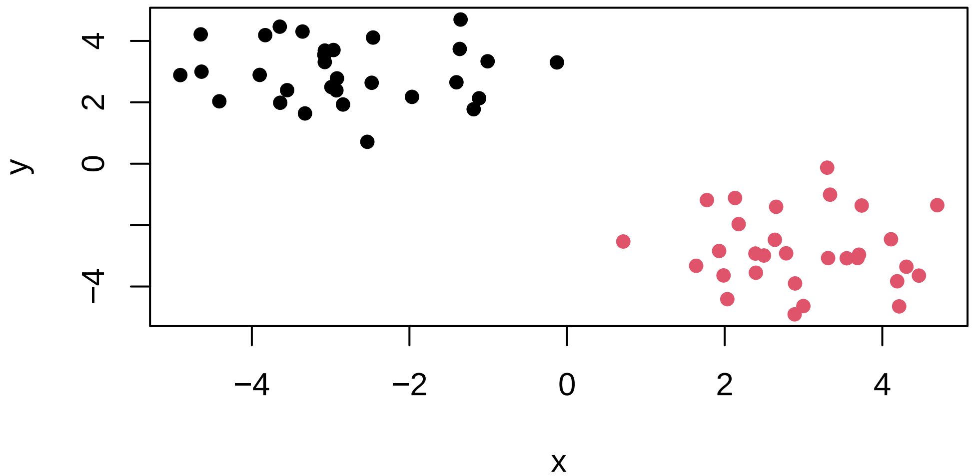
Q3. Cluster centers?

```r
k$centers
```

```
          x         y
1 -2.824148  2.970697
2  2.970697 -2.824148
```

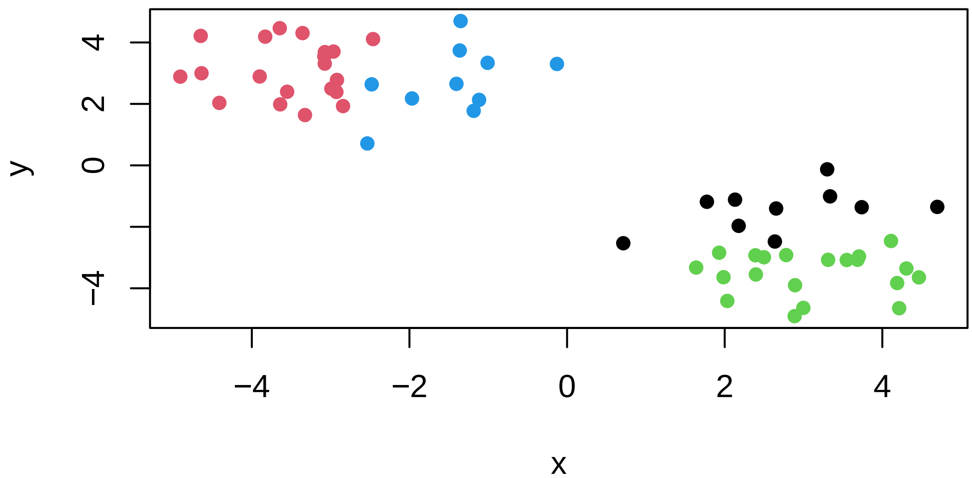
Q4. PLot my clustering results

```r
plot(x,col=k$cluster,pch=16)
```



Q5. Cluster the data again with kmeans() into 4 groups and plot the results.

```r
k4=kmeans(x,centers=4, nstart=20)
plot(x,col=k4$cluster, pch=16)
```



K-means is very popular mostly because it is fast and relatively straightforward to run and understand. It has a big limitation in that you need to tell it how many groups (k, or centers) you want.

#Hierarchical clustering

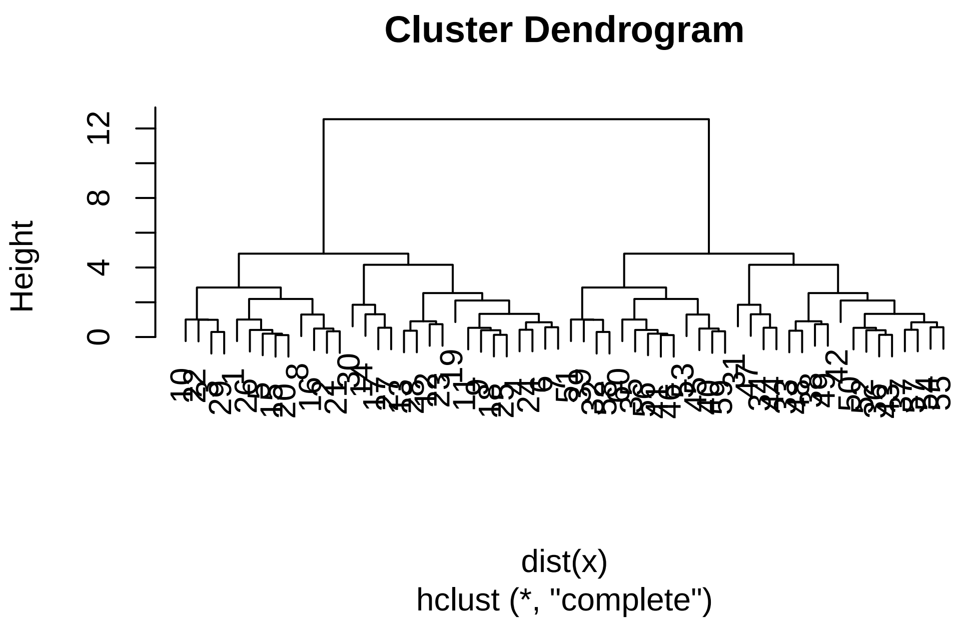
The main function in base R is called `hclust()`. You have to pass it in a "distance matrix" not just your input data.

You can generate a distance matrix with the `dist()` function.

```
hc=hclust(dist(x))
hc
```

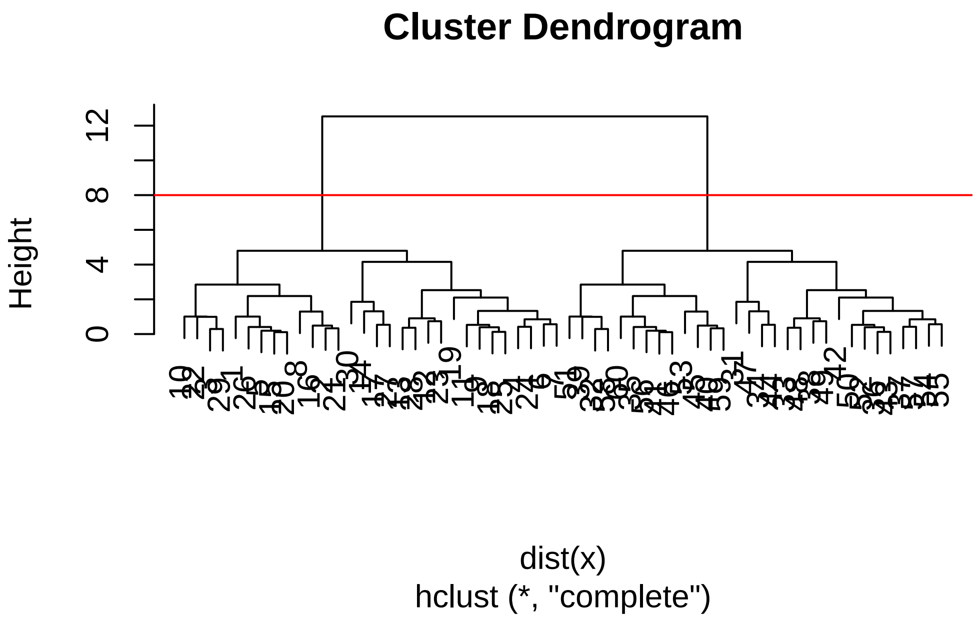
```
Call:
hclust(d = dist(x))

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

```
plot(hc)
```



To find the clusters (cluster membership vector) from a `hclust()` result we can "cut" the tree at a certain height that we like. For this we can use the `cutree()` function.

```r
plot(hc)
abline(h=8,col="red")
```



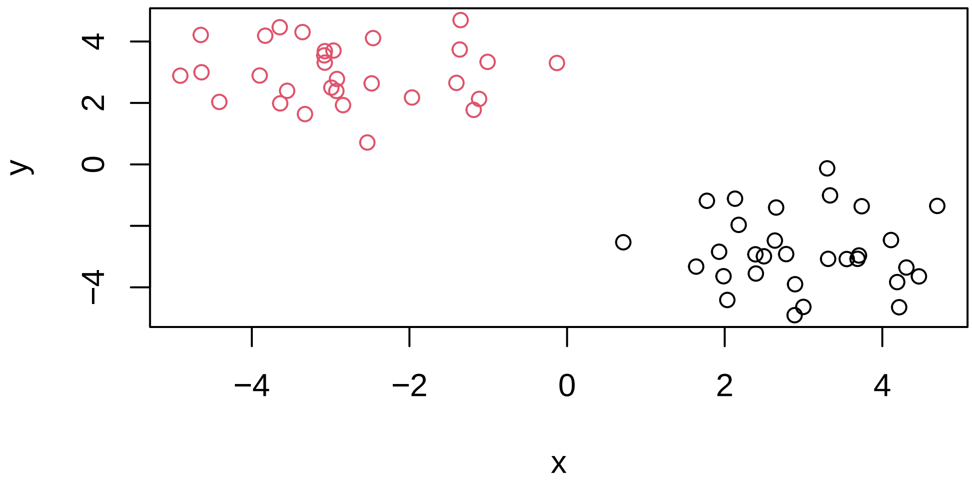

```r
grps =cutree(hc,h=8)
```

```r
table(grps)
```

```
grps
 1  2 
30 30 
```

Q6. Plot our hclust results.

```r
plot(x,col=grps)
```



#PCA Component Analysis

##PCA of UK food data

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)
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
7      Fresh_potatoes      720   874      566      1033
8           Fresh_Veg      253   265      171       143
9           Other_Veg      488   570      418       355
10 Processed_potatoes      198   203      220       187
11      Processed_Veg      360   365      337       334
12        Fresh_fruit     1102  1137      957       674
13            Cereals     1472  1582     1462      1494
14          Beverages       57    73       53        47
15        Soft_drinks     1374  1256     1572      1506
16   Alcoholic_drinks      375   475      458       135
17      Confectionery       54    64       62        41
```

Q1.

```r
dim(x)
```

```
[1] 17  5
```

```r
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
```

Note how the minus indexing works

```r
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
```

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

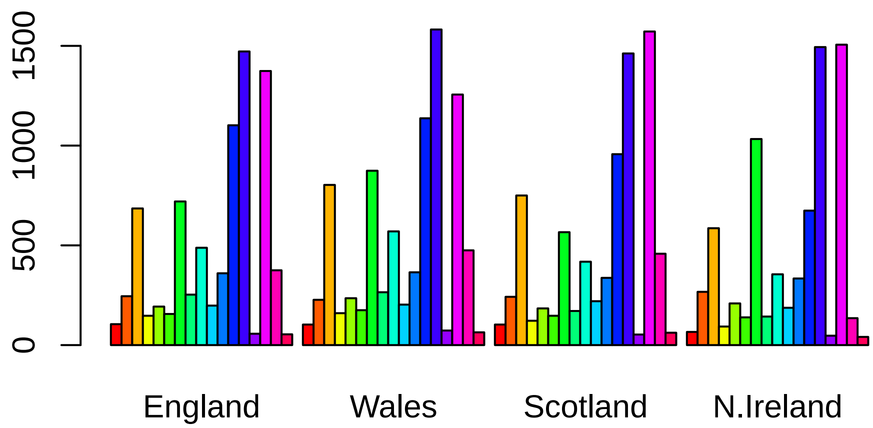
```
                England Wales Scotland N.Ireland
Cheese              105   103      103        66
Carcass_meat        245   227      242       267
Other_meat          685   803      750       586
Fish                147   160      122        93
Fats_and_oils       193   235      184       209
Sugars              156   175      147       139
```

```
dim(x)
```

```
[1] 17  4
```

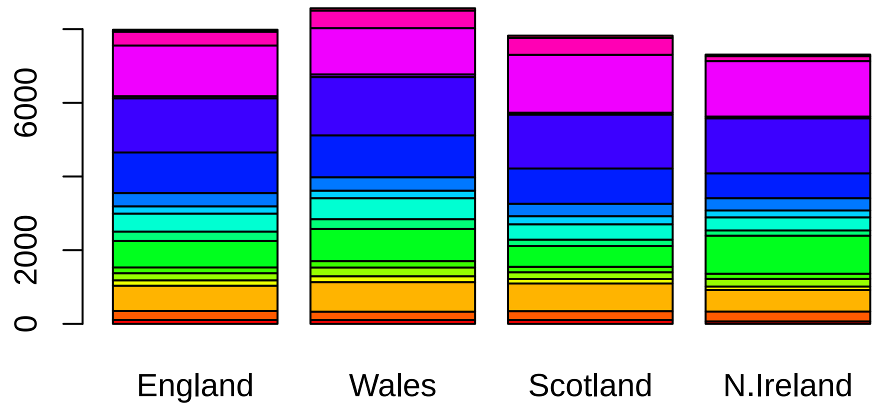
Q2. I prefer the second approach because if the first approach is run more than once, it keeps removing a column with every run until there is an error.

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



Q3.

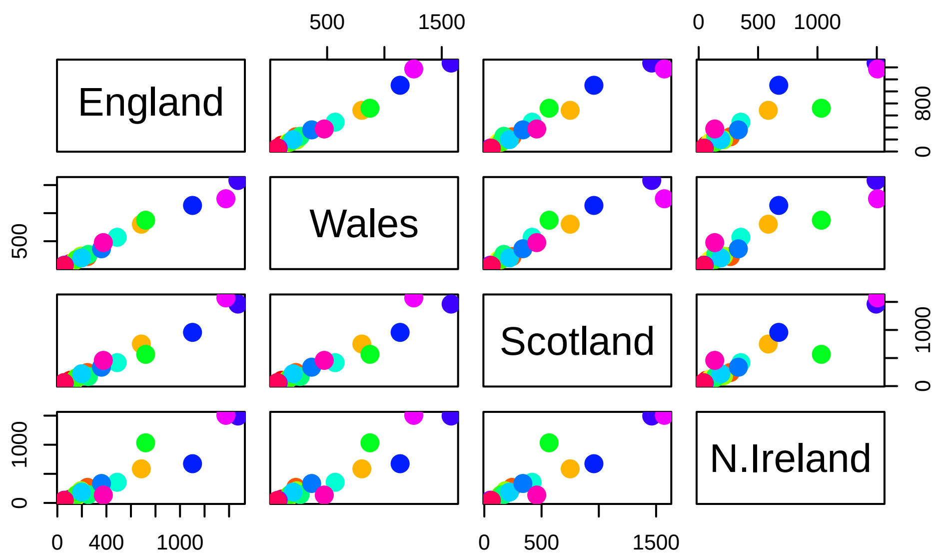
```
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```



A pairs plot can be useful if we don't have too many dimensions.

Q5.

```
pairs(x, col=rainbow(17), pch=16,cex=2)
```



Q6. There is greater spread between N. Ireland and other countries. It is an outlier compared with the other countries when comparing different foods.

##Principal Component Analysis (PCA)

PCA can help us make sense of these types of datasets. Let's see how it works.

The main function in "base" R is called `prcomp()`. In this case we want to first take the transpose of our input `x` so the columns are the food types and the countries are the rows.

```r
head(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 5.552e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

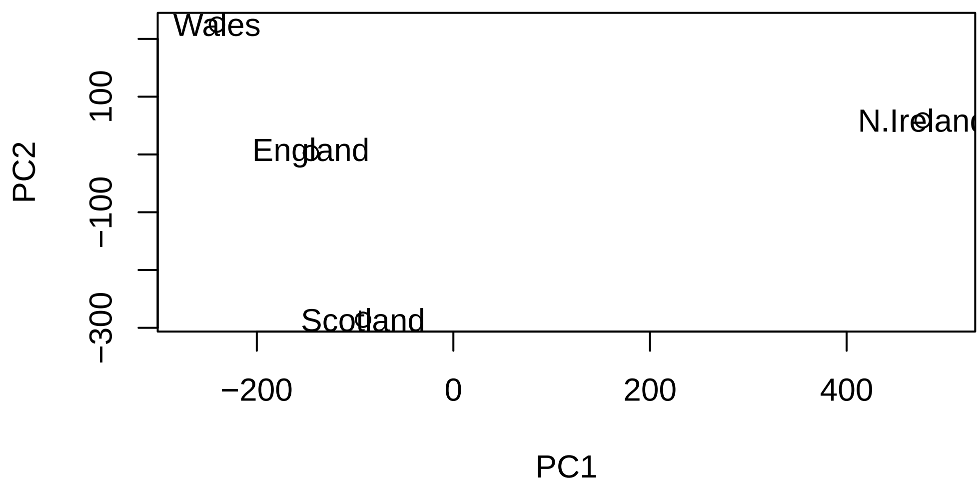
Q7.

```r
pca$x
```

```
                PC1         PC2         PC3           PC4
England  -144.99315    2.532999 -105.768945  1.042460e-14
```

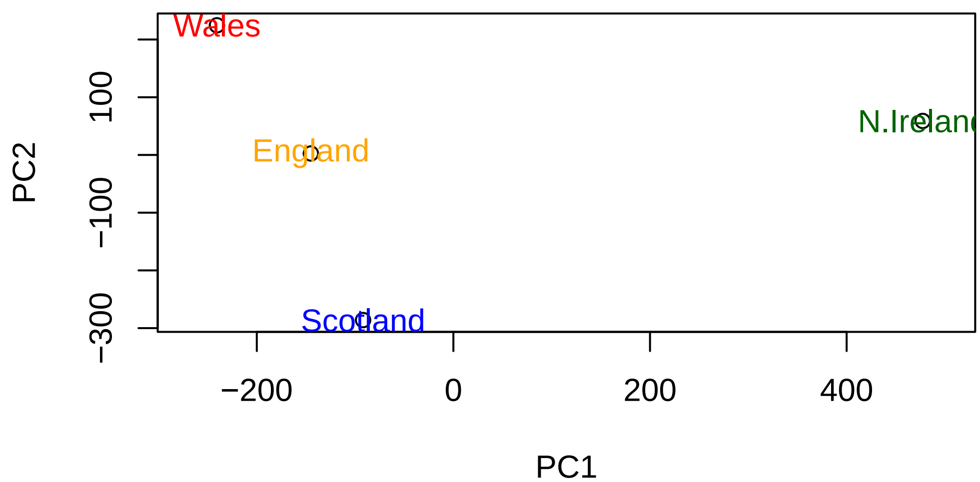
```
Wales      -240.52915  224.646925   56.475555  9.556806e-13
Scotland    -91.86934 -286.081786   44.415495 -1.257152e-12
N.Ireland   477.39164   58.901862    4.877895  2.872787e-13
```

```
plot(pca$x[,1],pca$x[,2], xlab="PC1", ylab="PC2", xlim=c(-270,500))
text(pca$x[,1], pca$x[,2], colnames(x))
```



Q8.

```
plot(pca$x[,1],pca$x[,2], xlab="PC1", ylab="PC2", xlim=c(-270,500))
text(pca$x[,1], pca$x[,2],  colnames(x),col=c("orange","red","blue","darkgreen"),pch=16)
```



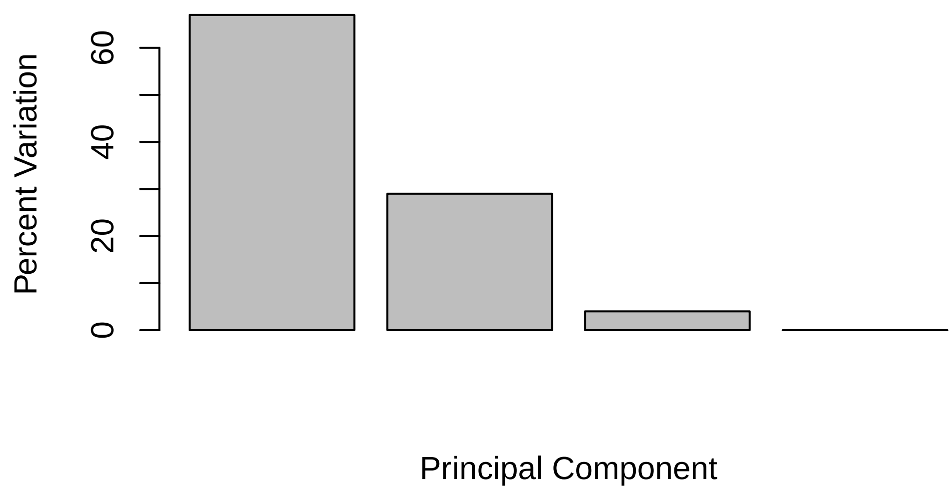
```r
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

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

```r
## or the second row here...
z <- summary(pca)
z$importance
```

```
                             PC1       PC2      PC3          PC4
Standard deviation     324.15019 212.74780 73.87622 5.551558e-14
Proportion of Variance   0.67444   0.29052  0.03503 0.000000e+00
Cumulative Proportion    0.67444   0.96497  1.00000 1.000000e+00
```

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



The “loadings” tell us how much the original variables (in our case the foods) contribute to the new variables i.e. the PCs.

```r
head(pca$rotation)
```

```
                        PC1         PC2         PC3          PC4
Cheese         -0.056955380 -0.01601285 -0.02394295 -0.537717586
Carcass_meat    0.047927628 -0.01391582 -0.06367111  0.827327785
Other_meat     -0.258916658  0.01533114  0.55384854 -0.054885657
Fish           -0.084414983  0.05075495 -0.03906481 -0.017195729
Fats_and_oils  -0.005193623  0.09538866  0.12522257  0.039441462
Sugars         -0.037620983  0.04302170  0.03605745  0.002788534
```

```
## Lets focus on PC1 as it accounts for > 90% of variance
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,1], las=2 )
```