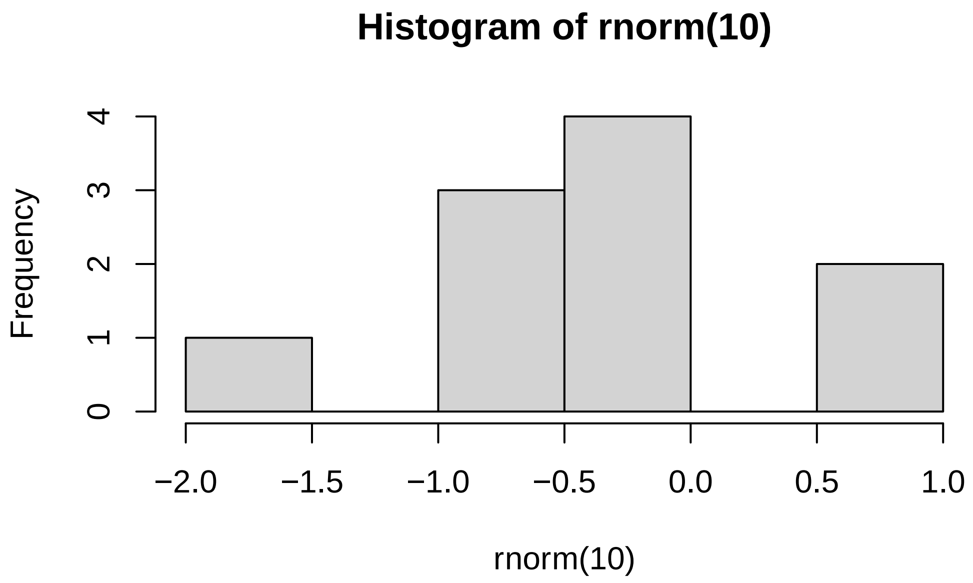
Class07:Machine Learning1

Ebru Robinson

Today we will begin our exploration of some "classical" machine learning approches. We will start with clustering:

Let's first make up some data to cluster where we know what the answer should be.

```
hist(rnorm(10))
```

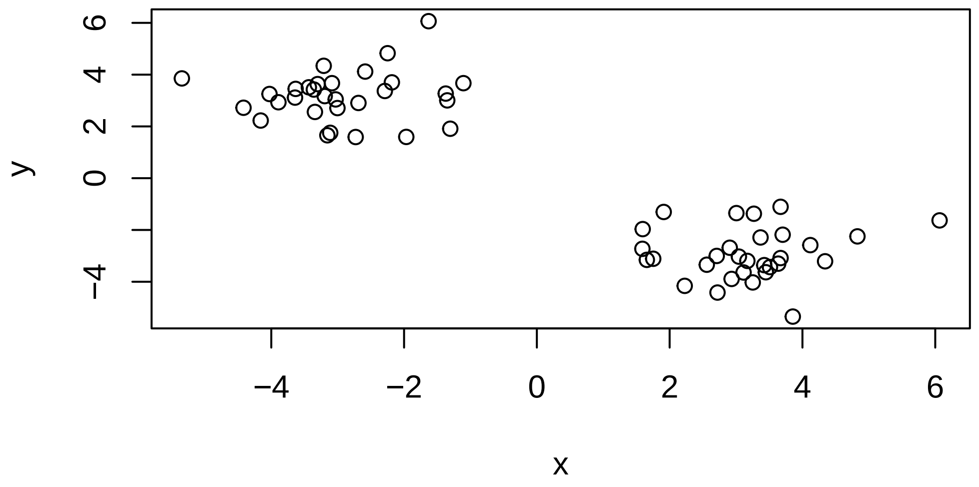


```
x <- c(rnorm(30,mean=-3),rnorm(30,mean=3))
y <- rev(x)

x <- cbind(x,y)
head(x)
```

```
             x        y
[1,] -3.030354 3.042981
[2,] -3.112372 1.752562
[3,] -1.966876 1.593968
[4,] -3.155117 1.656351
[5,] -5.346527 3.854675
[6,] -3.342669 2.559018
```

a wee peak at

```r
plot(x)
```



then main functiion in “base” R for k-means clustering is called `kmeans()`.

```r
k <- kmeans(x, centers=4)
k
```

```
K-means clustering with 4 clusters of sizes 12, 10, 20, 18

Cluster means:
          x         y
1  2.594342 -2.188377
```

```
2 -1.928161  3.827909
3 -3.426093  2.838432
4  3.550869 -3.419053

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

Within cluster sum of squares by cluster:
[1] 14.15911 15.44459 19.85046 25.12928
 (between_SS / total_SS =  93.9 %)

Available components:

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

Q. How big are the clusters(i.e their size)?

```
k$size
```

```
[1] 12 10 20 18
```

Q. What clusters do my data points reside in?

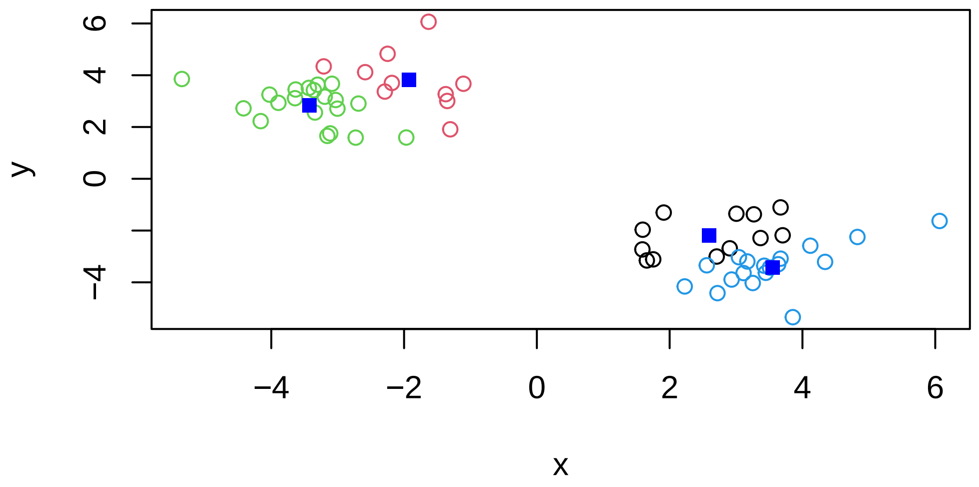
```
k$cluster
```

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

Q, Make a plot of our data colored by cluster assignment- i.e. make a result figure…

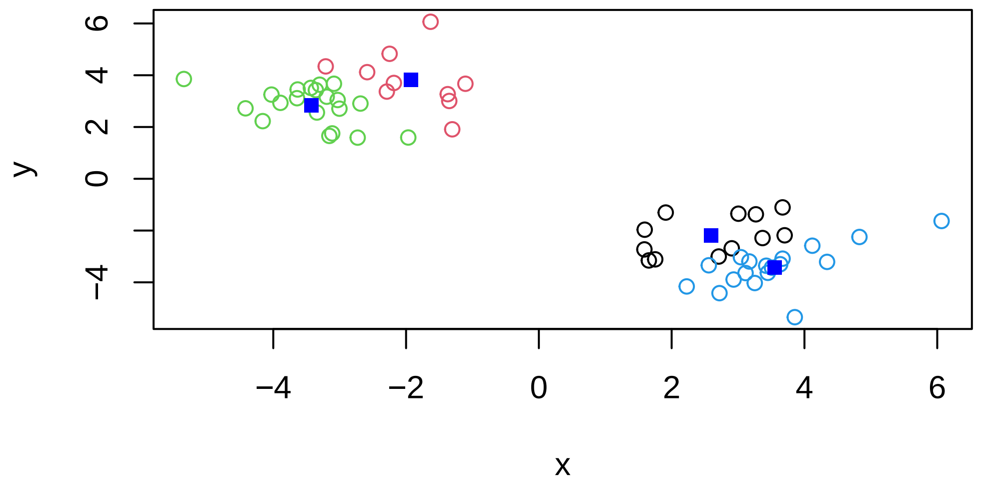
```
plot(x, col=k$cluster)
points(k$centers,centers=4,col="blue", pch=15)
```

```
Warning in plot.xy(xy.coords(x, y), type = type, ...): "centers" is not a
graphical parameter
```



```
k4 <- kmeans(x,centers=4)
plot(x, col=k$cluster)
points(k$centers,centers=4,col="blue", pch=15)
```

```
Warning in plot.xy(xy.coords(x, y), type = type, ...): "centers" is not a
graphical parameter
```



Q. Run kmeans with values center(i.e values of k) equal 1 to 6

```r
k1 <- kmeans(x,centers=1)$tot.withinss
k2 <- kmeans(x,centers=2)$tot.withinss
k3 <- kmeans(x,centers=3)$tot.withinss
k4 <- kmeans(x,centers=4)$tot.withinss
k5 <- kmeans(x,centers=5)$tot.withinss
k6 <- kmeans(x,centers=6)$tot.withinss
ans <- c(k1,k2,k3,k4,k5,k6)
```

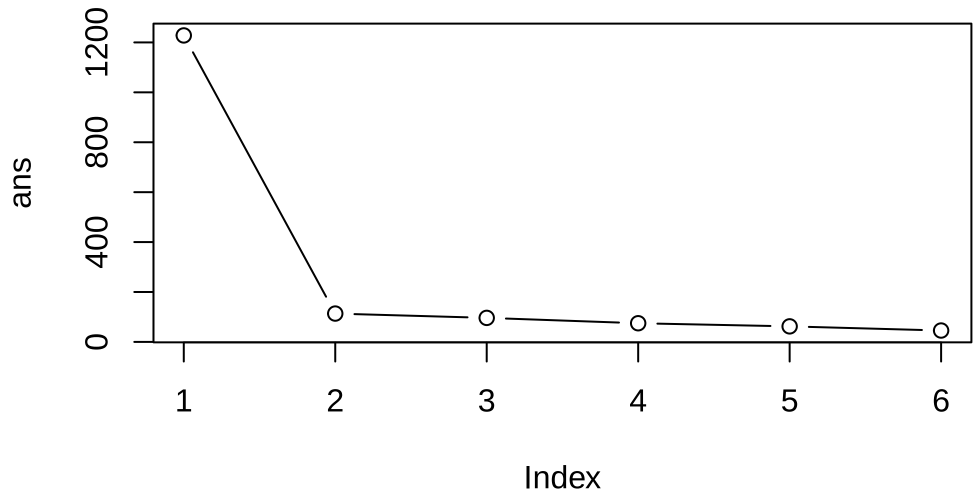
or use a for loop

```r
ans <- NULL
for(i in 1:6) {
  ans <- c(ans, kmeans(x,centers=i)$tot.withinss)
}
ans
```

```
[1] 1228.04720  113.56166   96.32075   74.83497   62.34236   45.60488
```

Make a "scree-plot"

```r
plot(ans, typ="b")
```



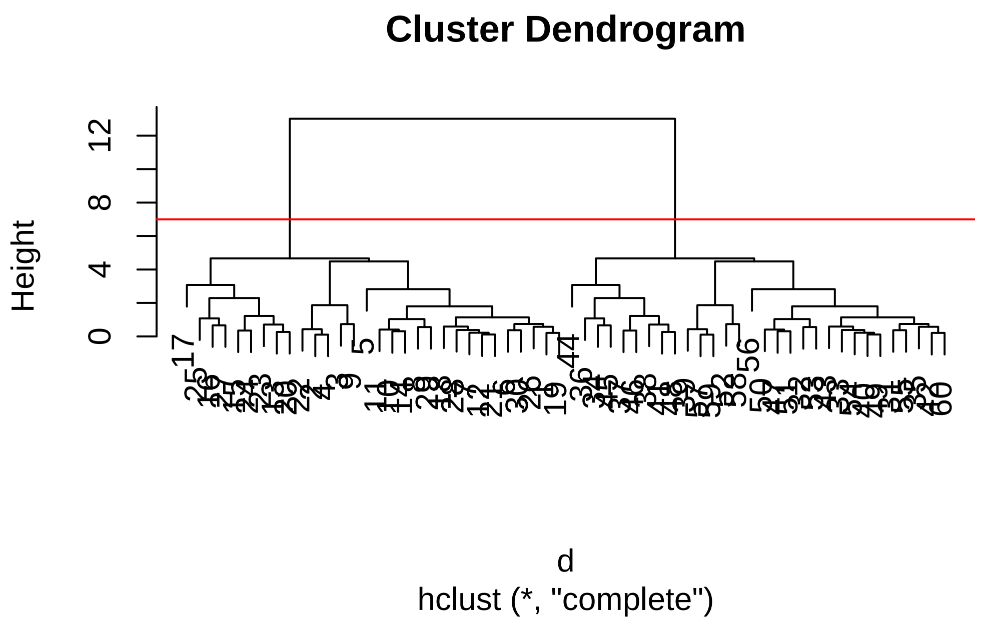
##Hierarchical Clustering

The main function in "base" R for this is called `hclust()`

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

```

Call:
hclust(d = d)

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

```r
plot(hc)
abline(h=7, col="red")
```



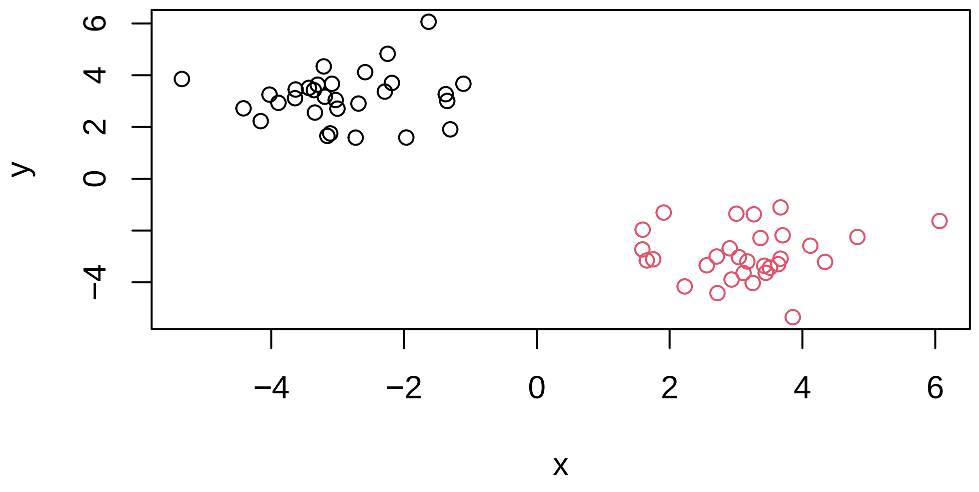
to obtain clusters from our **hclust** result object **hc** we “cut” the tree to yield different sub branches. For this we use the **cutree()** function

```
grps <- cutree(hc,h=7)
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 2 2
```

Results figure

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



```
#install.packages("pheatmap")
```

```
library("pheatmap")
```

Principal Component Analysis (PCA)

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

> Q1. How many rows and columns are in your new data frame named x? What R functions could you use to answer this questions?

This will return both the number of rows and columns in the data frame x.

```
dim(x)
```

```
[1] 17  5
```

```r
## Preview the first 6 rows
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
```

```r
# Note how the minus indexing works
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
```

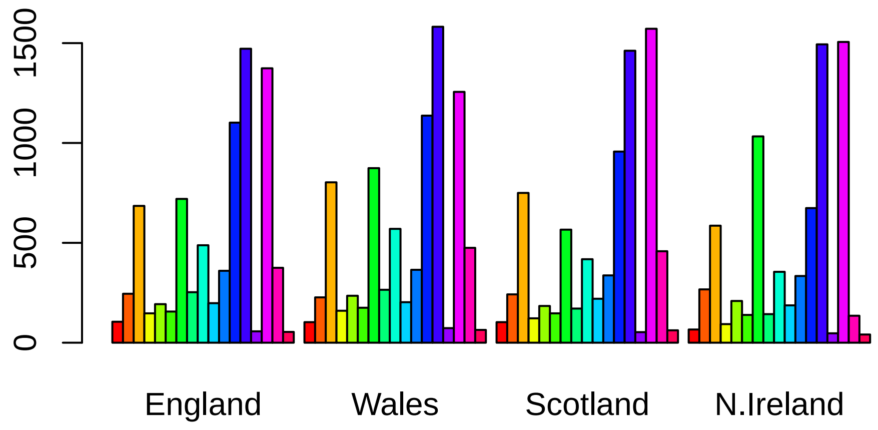
Q2. Which approach to solving the 'row-names problem' mentioned above do you prefer and why? Is one approach more robust than another under certain circumstances?

I like fixing it up front when reading the data…

I prefer the import-time approach: set row names when you read the file, e.g. read.csv(url, row.names = 1). It is more robust and reproducible.

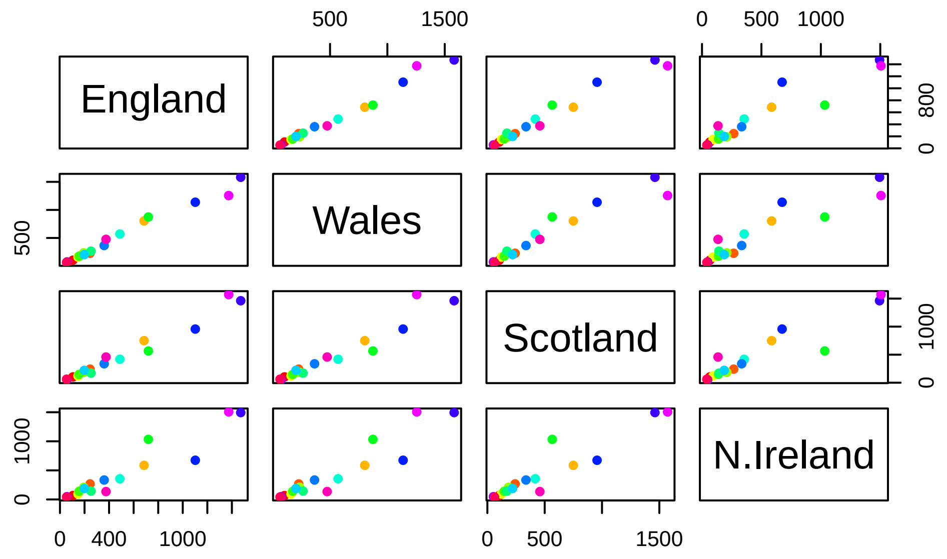
##spotting major differences and trends

```r
# Using base R
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```

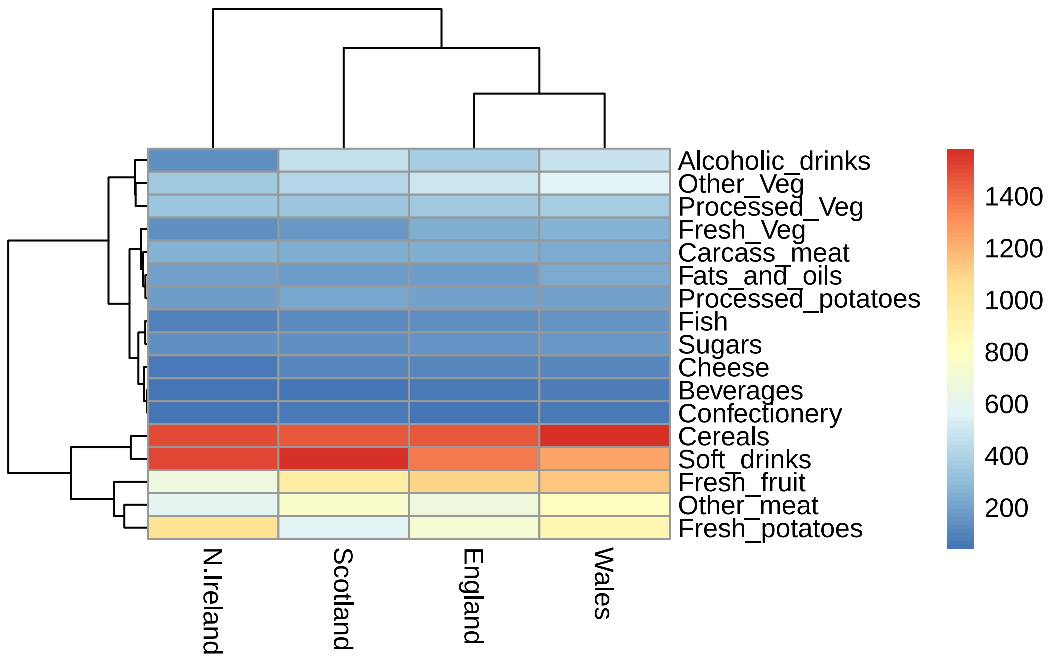


## Pairs plots and heatmaps Scatterplot matrices ("pairs plots") can be useful for relatively small datasets like this one. lets have a look:

```r
pairs(x, col=rainbow(nrow(x)), pch=16)
```



```r
library(pheatmap)

pheatmap( as.matrix(x) )
```



> Q6. Based on the pairs and heatmap figures, which countries cluster together and what does this suggest about their food consumption patterns? Can you easily tell what the main differences between N. Ireland and the other countries of the UK in terms of this data-set?

It looks like Wales and England are quite similar in their consumption of these foods. It is still quite diffucult to tell what is going on in the dataset

##PCA to the rescue

The main function in "base" R for PCA is called `prcomp()`.

As we want to do PCA on the food data for the different countries we will want the foods in the columns

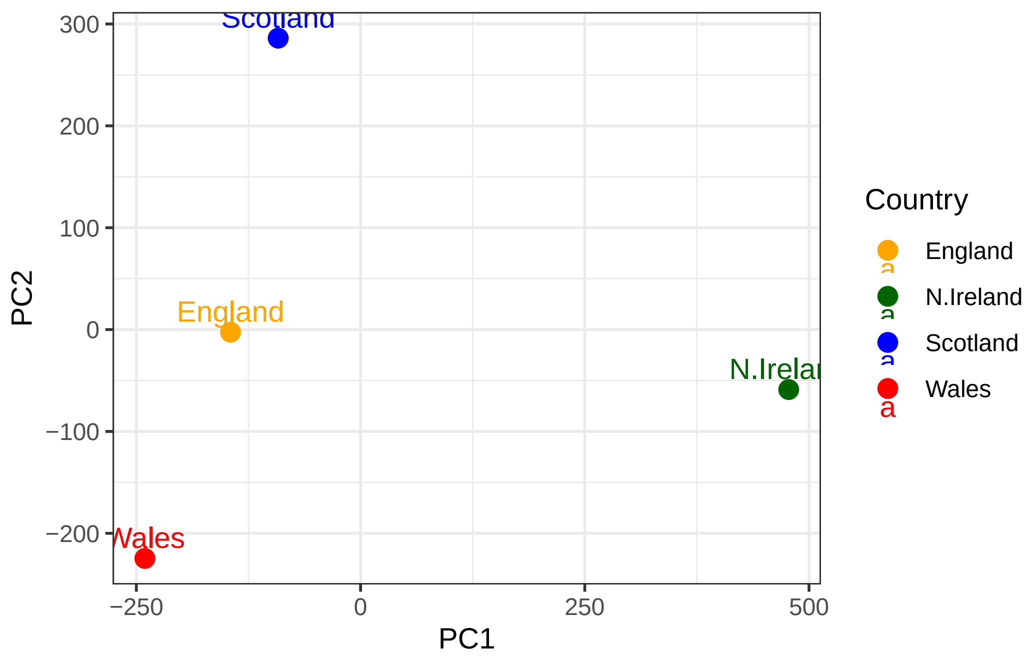
```
# Use the prcomp() PCA function
pca <- prcomp(t(x))
summary(pca)
```

```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 2.921e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

Our result object is csalled `pca` and it has a `$x` component that will look at first

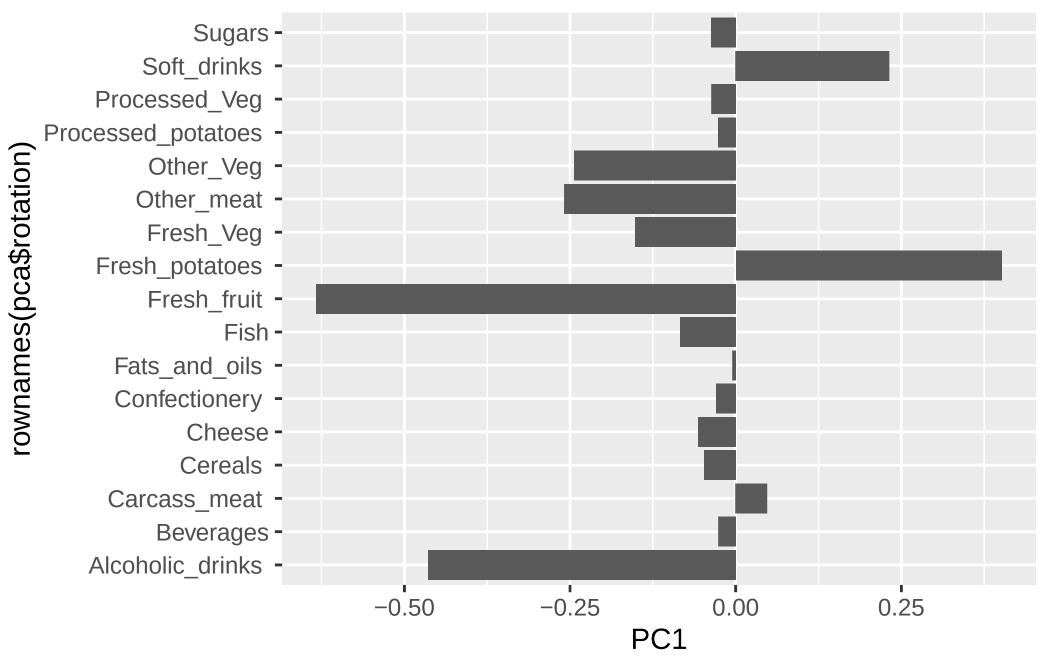
```
library(ggplot2)

# Convert PCA results to a data frame
df <- as.data.frame(pca$x)
df$Country <- rownames(df)

# Define colors in the same order as your countries appear in df
cols <- c("orange", "red", "blue", "darkgreen")
names(cols) <- df$Country   # optional: name colors explicitly for safety

# Plot
ggplot(df, aes(x = PC1, y = PC2, label = Country, color = Country)) +
  geom_point(size = 3) +
  geom_text(vjust = -0.5) +
  scale_color_manual(values = cols) +
  xlab("PC1") +
```

```r
  ylab("PC2") +
  theme_bw()
```



Another major result out of PCA is the so-called "variable" loadings or `$rotation` that tells us how the original variables (foods) contributes to the PC's (our new axis)

```r
ggplot(pca$rotation) +aes(PC1, rownames(pca$rotation))+ geom_col()
```


rownames(pca$rotation)
Sugars
Soft_drinks
Processed_Veg
Processed_potatoes
Other_Veg
Other_meat
Fresh_Veg
Fresh_potatoes
Fresh_fruit
Fish
Fats_and_oils
Confectionery
Cheese
Cereals
Carcass_meat
Beverages
Alcoholic_drinks
−0.50
−0.25
0.00
0.25
PC1