Class 7: Machine Learning 1

Wade Ingersoll (PID: 69038080)

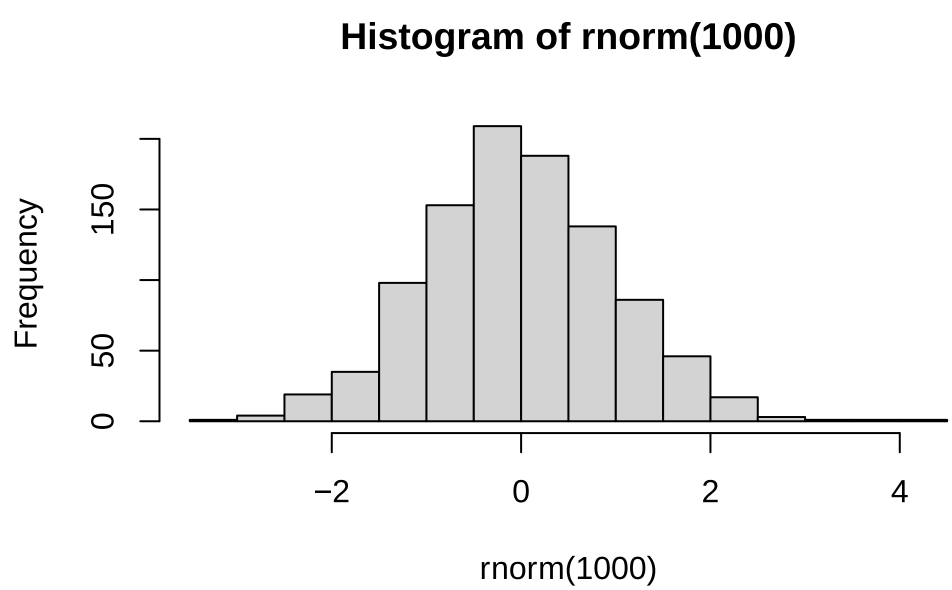
Today we will begin our exploration of some "classical" machine learning approaches. We will start with clustering.

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

```r
rnorm(10)
```

```
 [1] -1.0681029 -1.5655056 -1.2788473  0.1913744  1.9038961 -1.1879588
 [7] -0.4541442 -0.5998826  0.8935763 -0.1164368
```

```r
hist( rnorm(1000) )
```

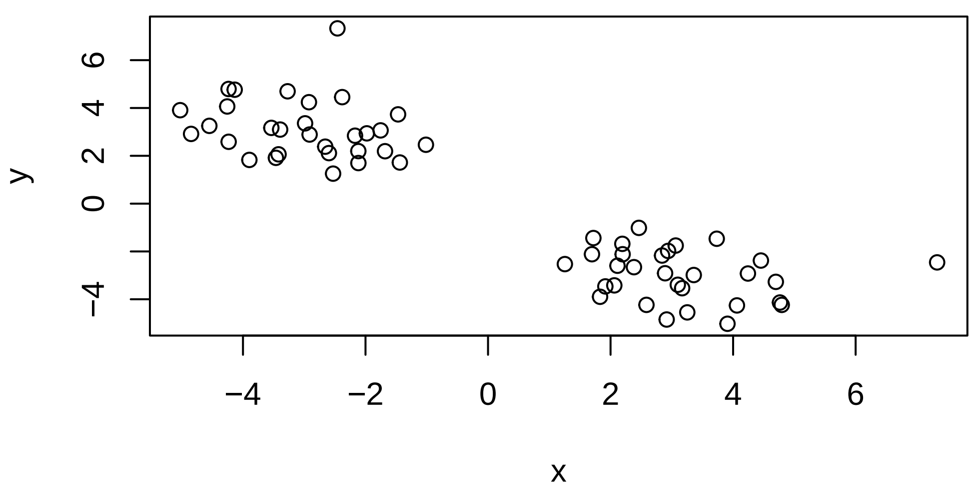


```r
x <- c( rnorm(30, mean=-3), rnorm(30, mean=3) )
y <- rev(x)

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

```
             x        y
[1,] -1.681652 2.192150
[2,] -1.439419 1.719849
[3,] -3.536872 3.168177
[4,] -4.233827 2.585800
[5,] -2.170789 2.840339
[6,] -4.135237 4.764704
```

A look at x with `plot()`

```r
plot(x)
```



Then main function in "base" R for K-means clustering is called `kmeans()`

```r
k <- kmeans(x, centers = 2)
k
```

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

Cluster means:
          x         y
1  3.130587 -2.981431
2 -2.981431  3.130587

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] 80.65708 80.65708
 (between_SS / total_SS =  87.4 %)

Available components:

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

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

```r
k$size
```

```
[1] 30 30
```

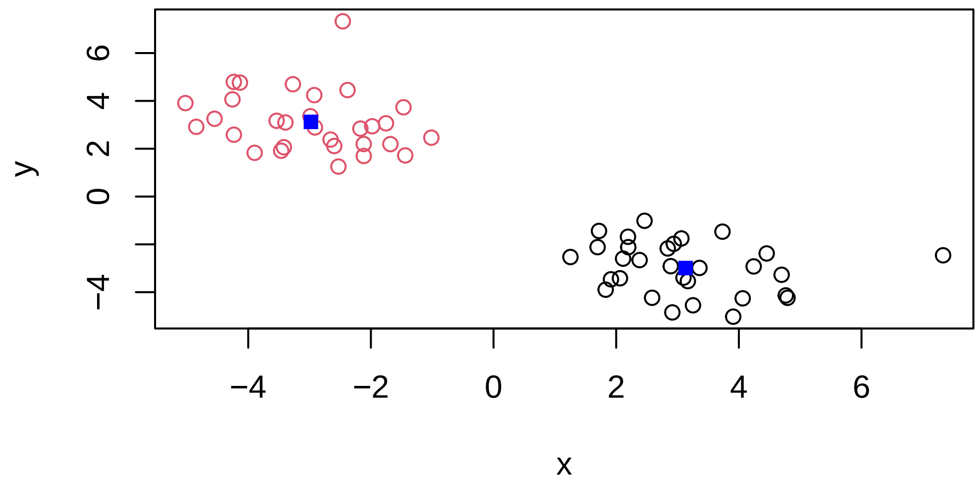
> Q. What clusters do my data points reside in?

```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
```

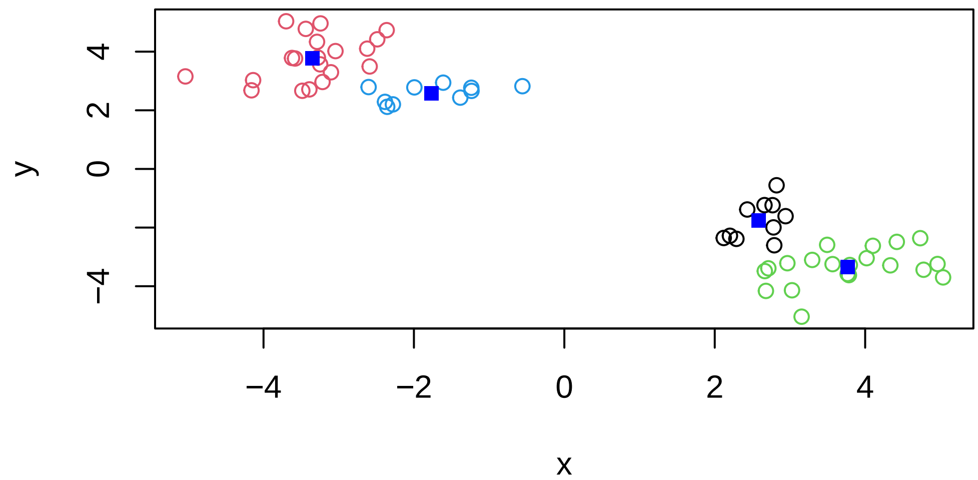
> Q. Make a plot of our data colored by cluster assignment - i.e. Make a result figure…

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



Q. Cluster with k-means into 4 clusters and plot the results as above.

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

k <- kmeans(x, centers = 4)

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



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

```
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

answer <- c(k1, k2, k3, k4, k5, k6)
```

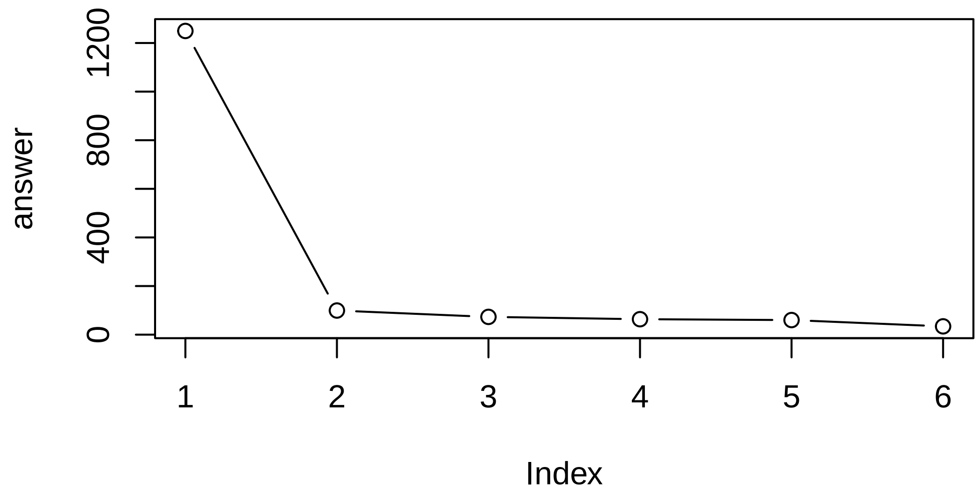
Or use a for loop

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

```
[1] 1249.56371   99.25738   73.11373   63.51748   60.02594   34.09707
```

Make a "scree-plot"

```
plot(answer, typ="b")
```



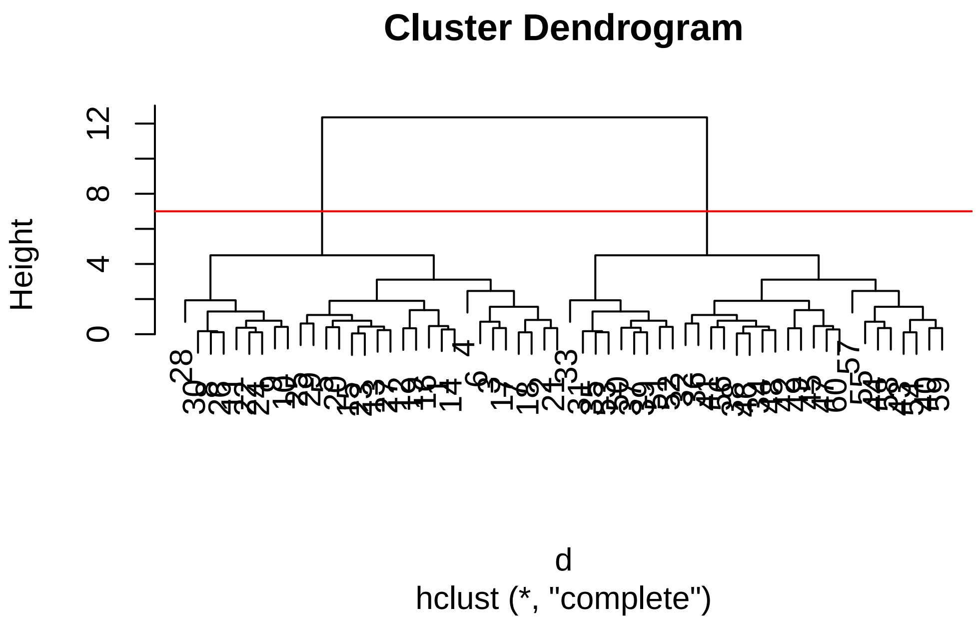
Hierarchical Clustering

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

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

```
Call:
hclust(d = d)

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

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

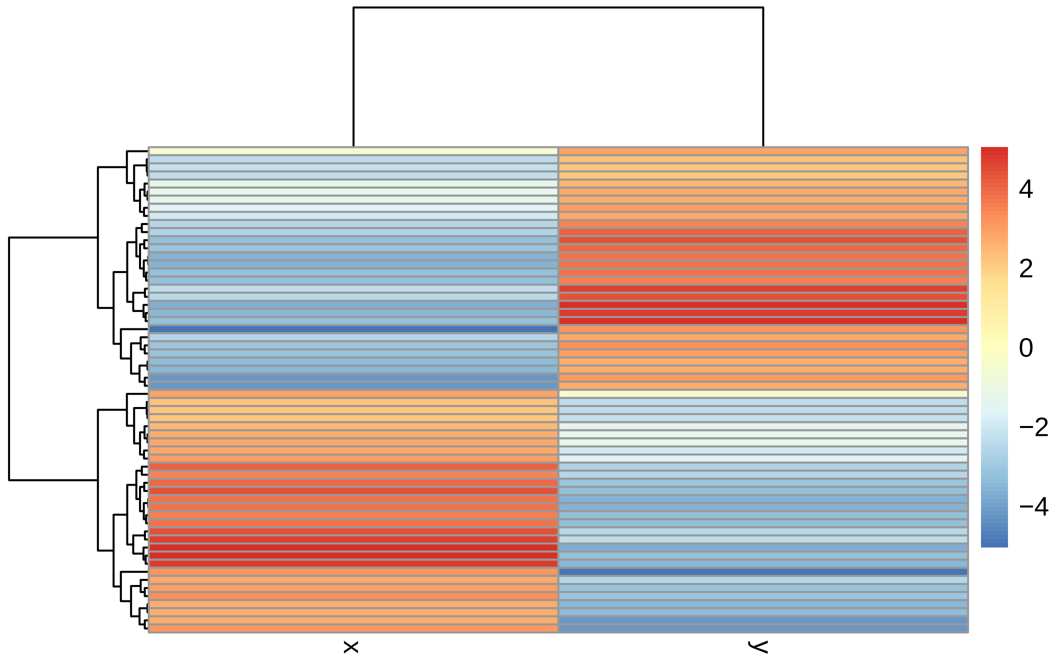


To obtain clusters from our `hclust()` result object **hc** we "cut" the tree to yield different sub branches. For this we use the `cutree()` function

```
cutree(hc, k=7)
```

```
 [1] 1 2 2 2 1 2 2 3 3 3 3 1 1 1 1 1 2 2 1 1 2 3 1 3 1 3 1 3 1 3 4 5 4 5 4 5 4 5
[39] 4 6 5 5 6 6 5 5 5 5 5 4 4 4 4 6 6 5 7 6 6 5
```

```
library(pheatmap)

pheatmap(x)
```



Principal Component Analysis (PCA)

```r
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?

```r
# Complete the following code to find out how many rows and columns are in x?
dim(x)
```

```
[1] 17  5
```

> A: There are 17 rows and 5 columns

Preview the first 6 rows

```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
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
```

> 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...

```
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
```

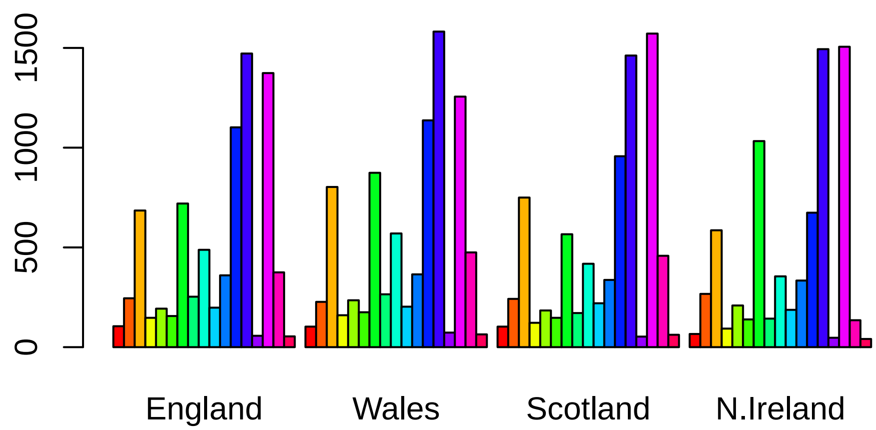
Spotting major differences and trends

```r
rainbow(5)
```

```
[1] "#FF0000" "#CCFF00" "#00FF66" "#0066FF" "#CC00FF"
```

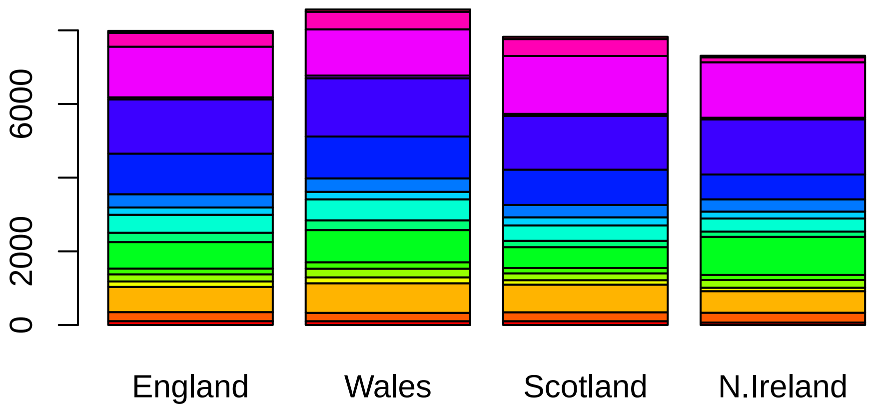
Using base R

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



Q3: Changing what optional argument in the above barplot() function results in the following plot?

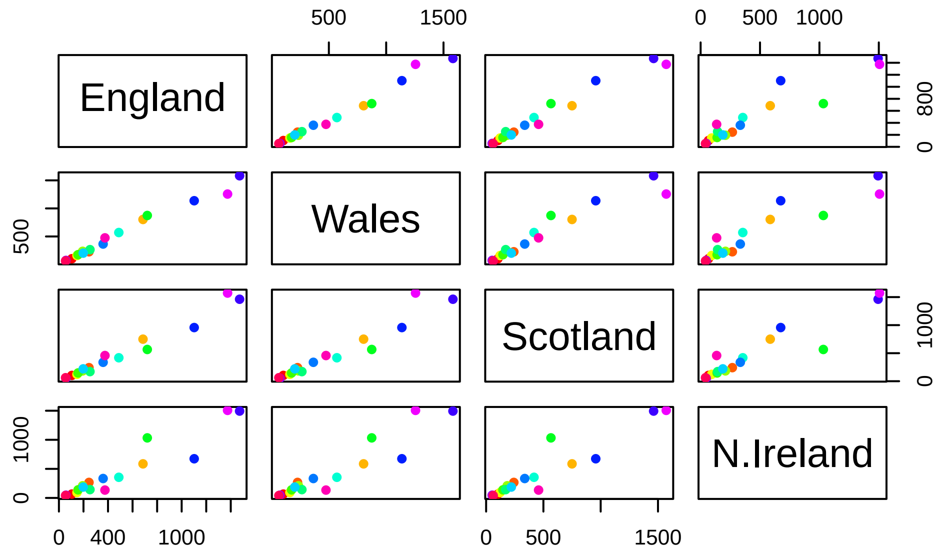
```r
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```



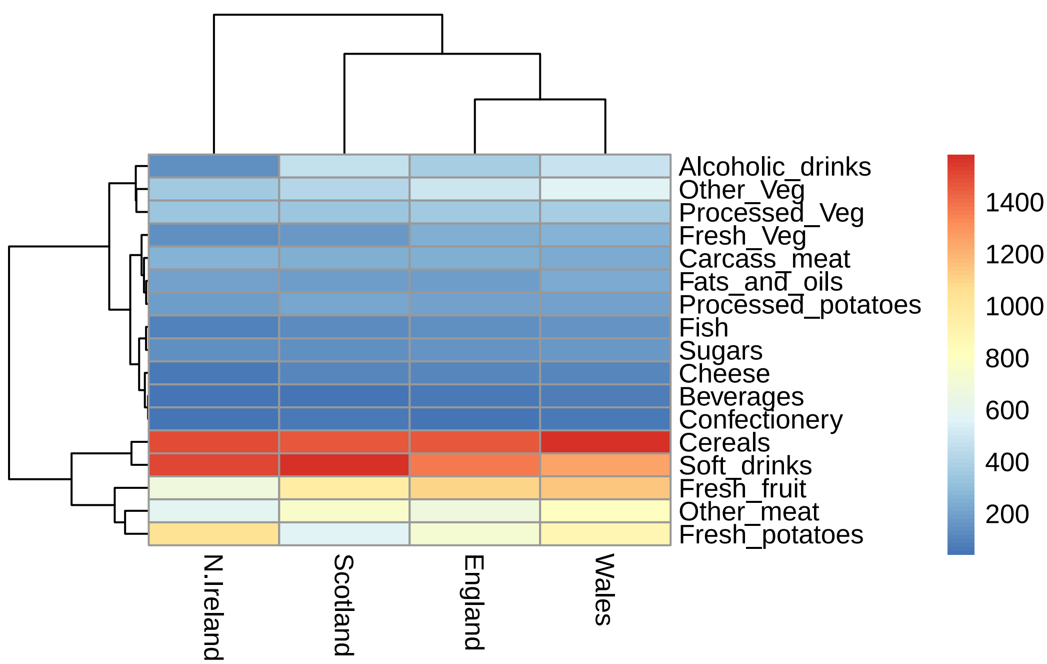
Q5: We can use the pairs() function to generate all pairwise plots for our countries. Can you make sense of the following code and resulting figure? What does it mean if a given point lies on the diagonal for a given plot?

A: Each point is a food and when they lie on the diagonal, it means the two countries eat that given food at the same rate

```
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?

A: It looks like Wales and England are quite similar in their consumption of these foods. It is still quite difficult 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.

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

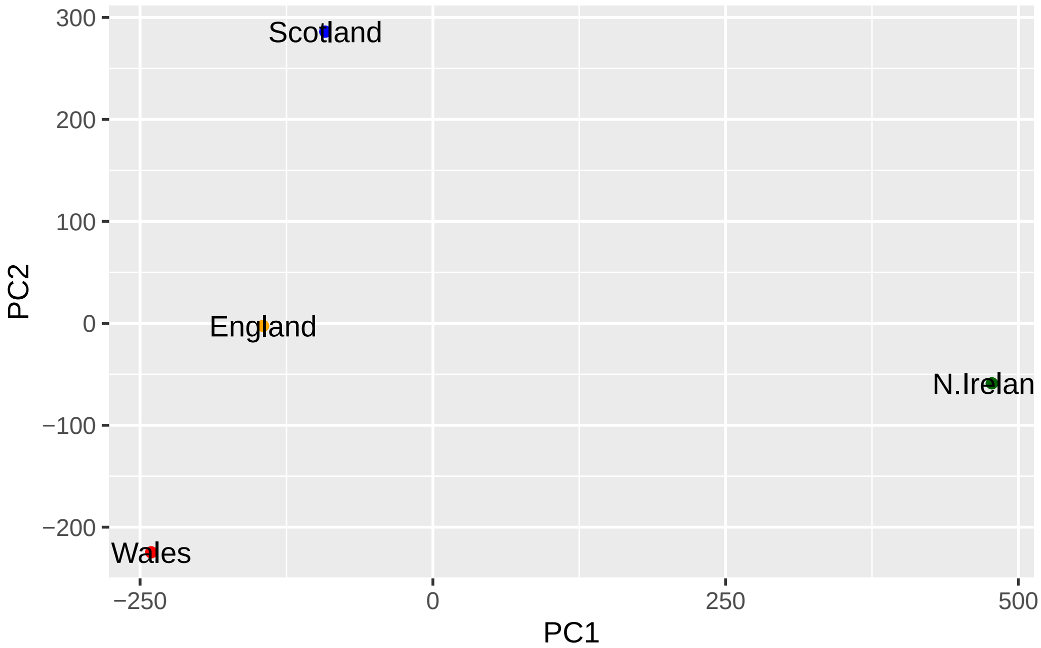
```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 3.176e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

```
pca$x
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -4.894696e-14
Wales     -240.52915 -224.646925 -56.475555  5.700024e-13
Scotland   -91.86934  286.081786 -44.415495 -7.460785e-13
N.Ireland  477.39164  -58.901862  -4.877895  2.321303e-13
```

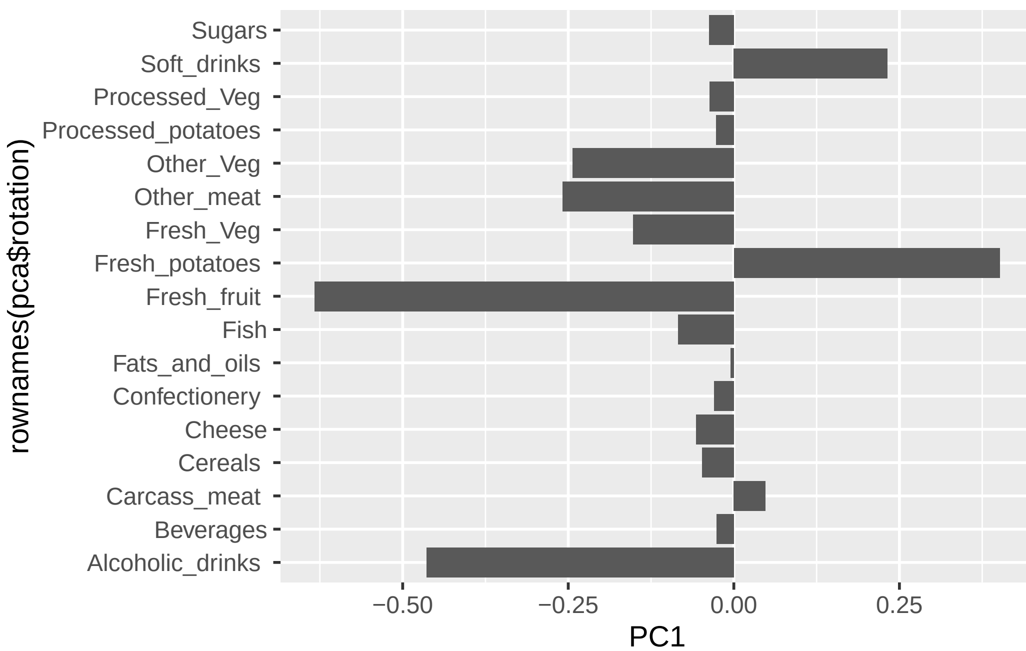
Our result object is called `pca` and it has a `$x` component that we will look at first

```
library(ggplot2)

cols <- c("orange", "red", "blue", "darkgreen")

ggplot(pca$x) +
  aes(PC1, PC2, label=rownames(pca$x)) +
  geom_point(col=cols) +
  geom_text()
```



Another major result out of the PCA is the so-called "variable loadings" or `$rotation` that tells us how the original variables (foods) contribute to the PCs (i.e. our new axis).

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


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
rownames(pca$rotation)
−0.50
−0.25
0.00
0.25
PC1