Class 7: Machine Learning I

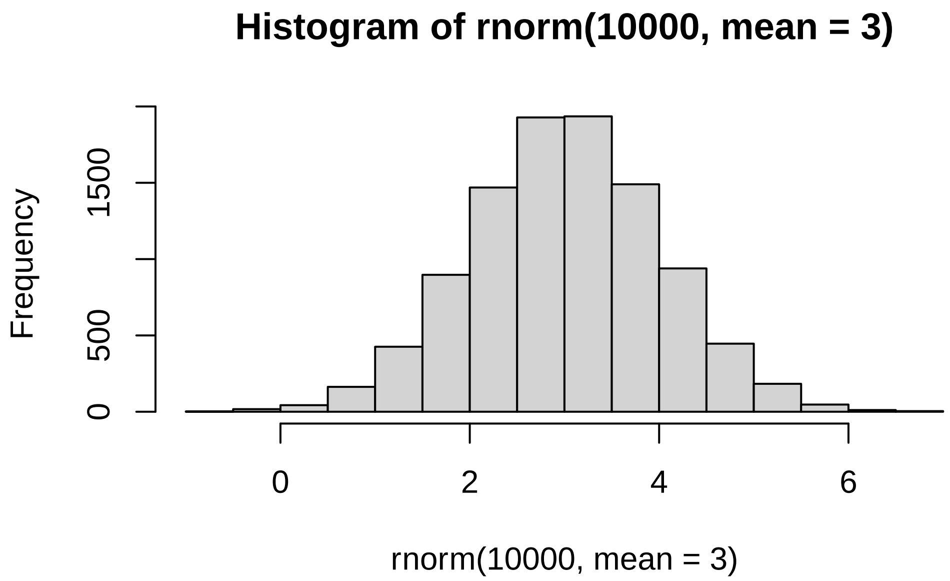
Isabel Hui - A16887852

Today we are going to learn how to apply different machine learning methods, beginning with clustering.

The goal here is to find groups/clusters in your input data.

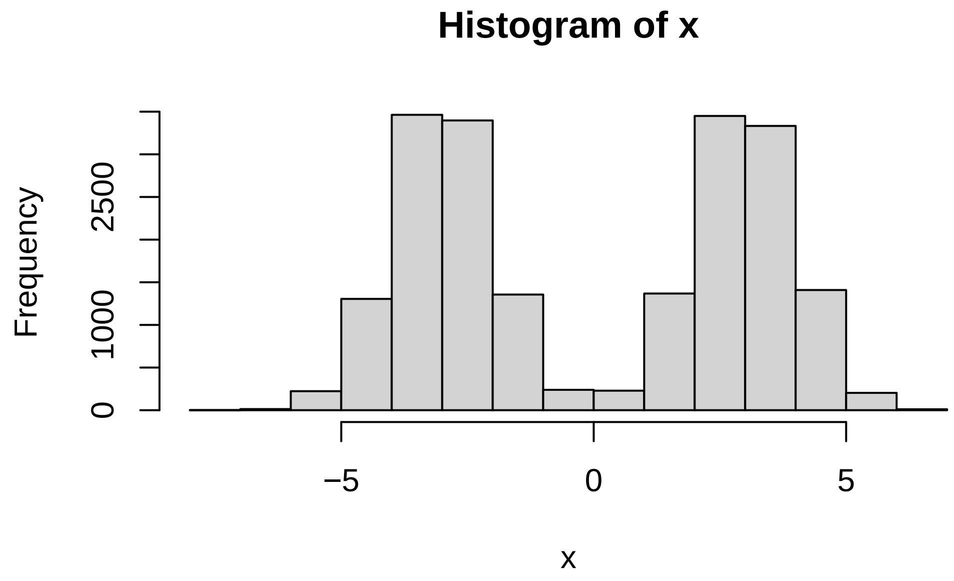
First I will make up some data with clear gorups. For this I will use the `rnorm()` function:

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



We can made two peaks on the plot by concatonating `c()` two `rnorm()` functions like so:

```
n <- 10000
x <- c(rnorm(n, -3), rnorm(n, +3))
hist(x)
```



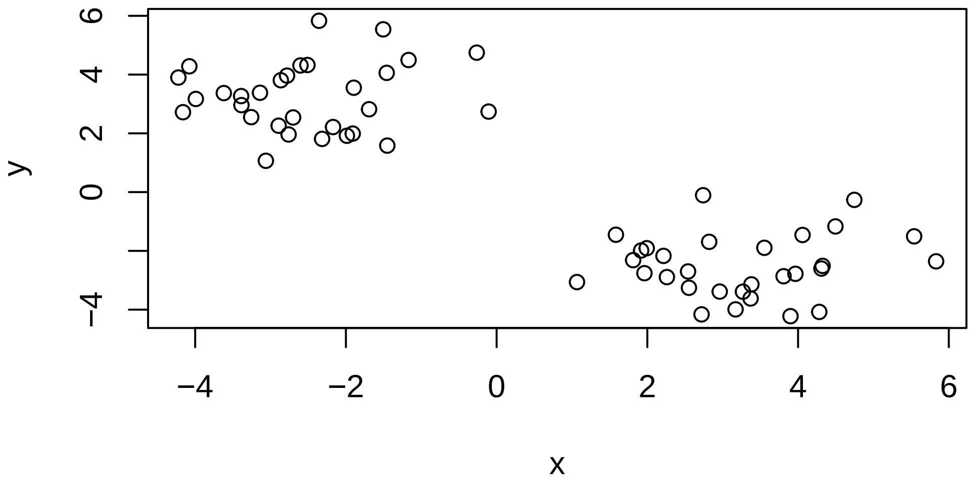
We can also make a cluster plot to show the groupings along axes that are both (+) and (-).

```r
n <- 30
x <- c(rnorm(n, -3), rnorm(n, +3))
y <- rev(x)

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

```
             x        y
[1,] -1.986921 1.917910
[2,] -1.692491 2.820572
[3,] -2.863494 3.807655
[4,] -2.602680 4.310190
[5,] -2.356578 5.831886
[6,] -1.895515 3.552803
```

```r
plot(z)
```



Use the `kmeans()` function setting k to 2 and `nstart = 20`.

Inspect/print the results.

> Q. How many points are in each cluster?

> Q. What 'component' of your result object details - cluster size? - cluster assignment/membership? - cluster center?

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

```r
km <- kmeans(z, centers = 2)
km
```

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

Cluster means:
          x         y
1 -2.523405  3.238189
2  3.238189 -2.523405

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] 72.07229 72.07229
 (between_SS / total_SS =  87.4 %)

Available components:

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

Results in kmenas object `km`.

```
attributes(km)
```

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

$class
[1] "kmeans"
```

Cluster size?

```
km$size
```

```
[1] 30 30
```

Cluster assignment/membership?

```
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
```

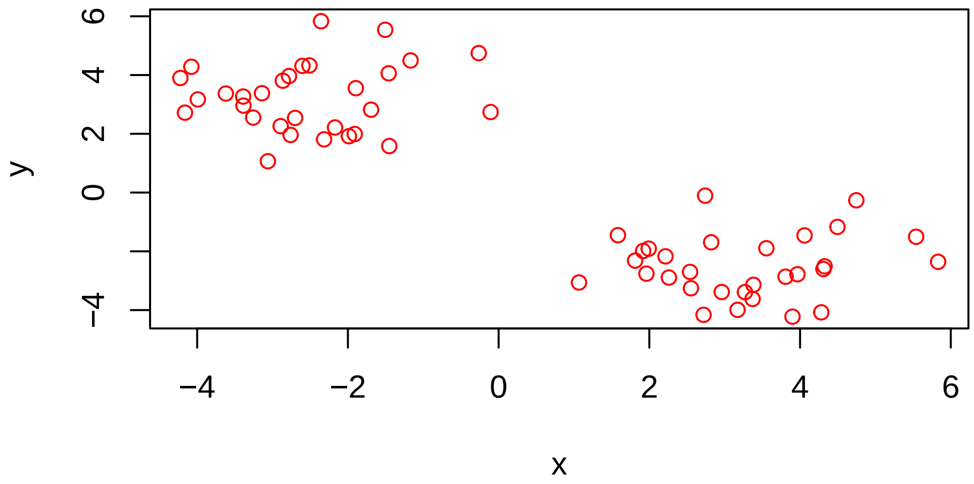
Cluster center?

```
km$centers
```

```
          x         y
1 -2.523405  3.238189
2  3.238189 -2.523405
```

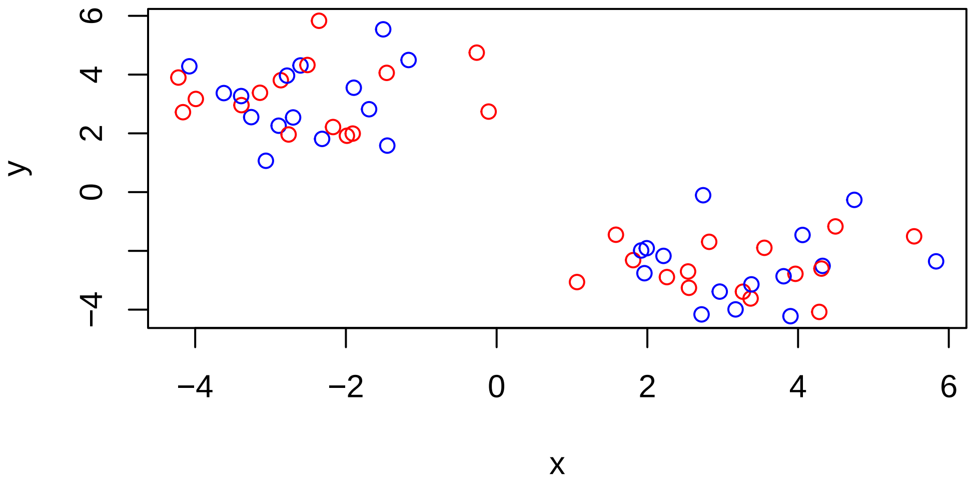
> Q. Plot x colored by the kmeans cluster assignment and add cluster centers as blue points.

```
plot(z, col = "red")
```



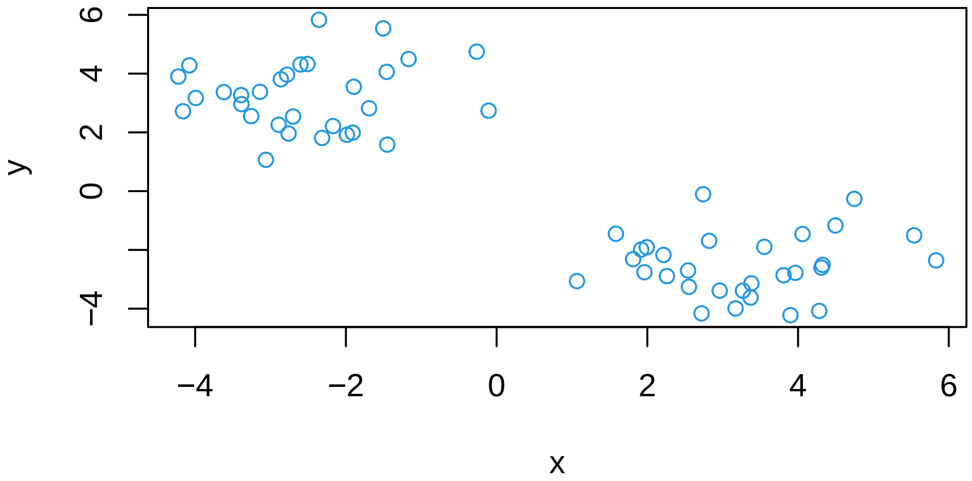
R will recycle the shorter color vector to be the same length as the longer (length of data points) in z.

```
plot(z, col = c("red", "blue"))
```

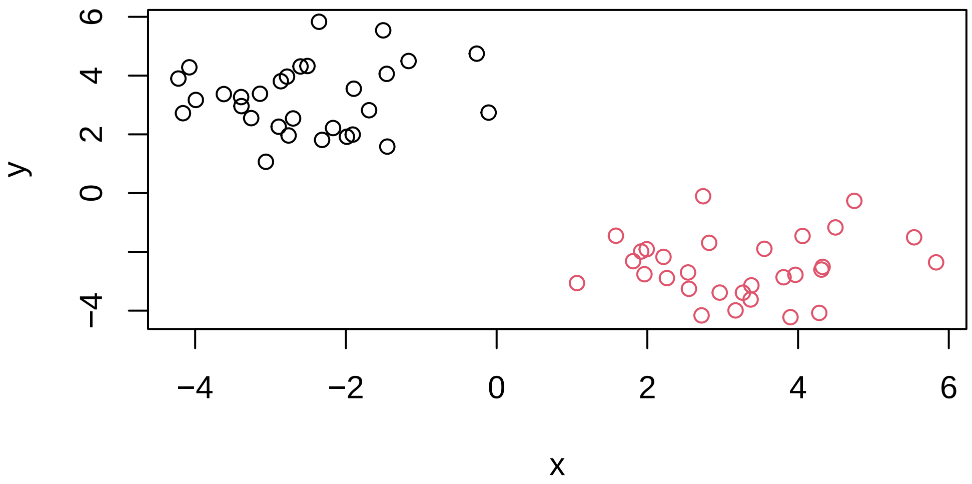


R corresponds numbers to colors.

```
plot(z, col = (4))
```

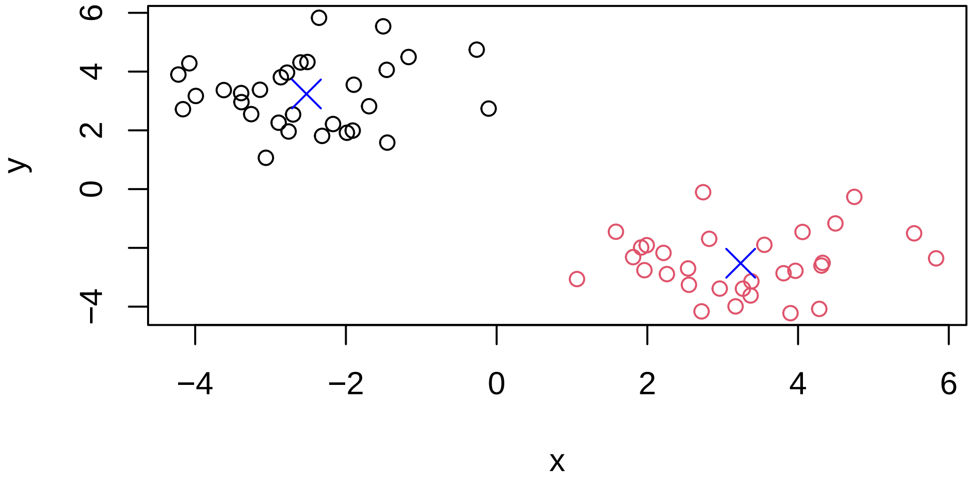


```
plot(z, col = km$cluster)
```



We can use the **points()** function to add new points to an existing plot… like the cluster centers. We can change the color using `col`, shape using `pch`, and size using `cex`.

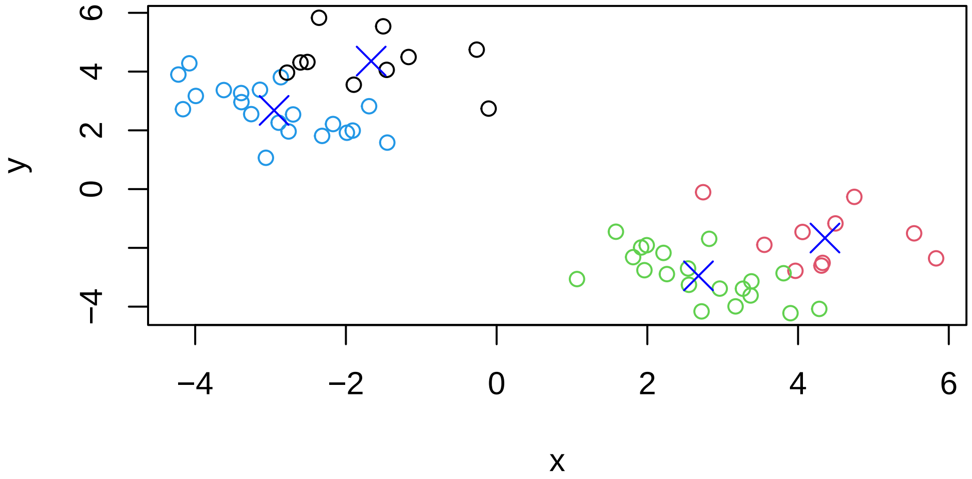
```
plot(z, col = km$cluster)
points(km$centers, col = "blue", pch = 4, cex = 2)
```



> Q. Can you run `kmeans` and ask for 4 clusters and plot the results like we have done before?

Note that running this multiple times results in a different result each time; it just runs again and again, only make two clusters.

```r
km4 <- kmeans(z, centers = 4)
plot(z, col = km4$cluster)
points(km4$centers, col = "blue", pch = 4, cex = 2)
```



Hierarchical Clustering

Let's take our same made-up data `z` and see how hclust works.

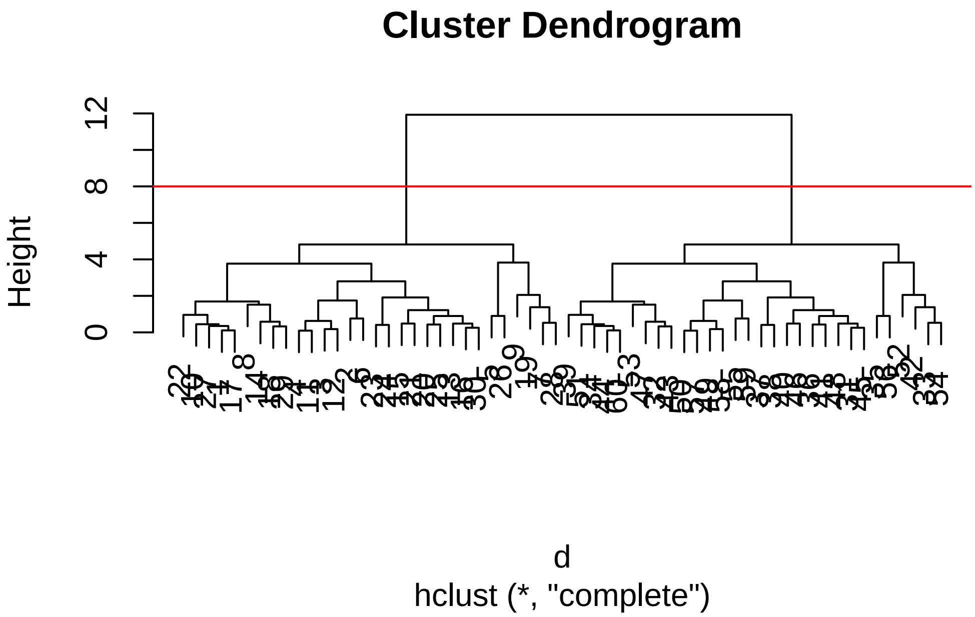
First we need a distance matrix of our data to be clustered.

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

```
Call:
hclust(d = d)

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

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



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

```
 [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
```

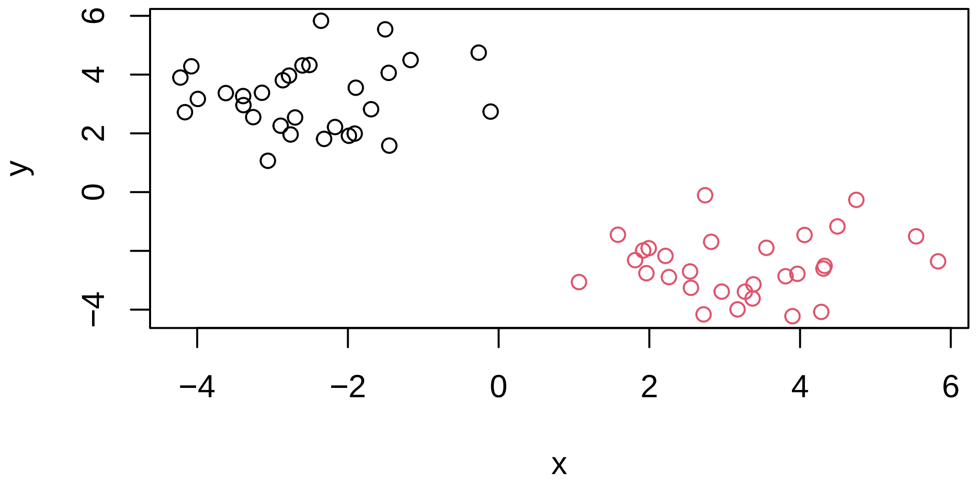
I can get my cluster membership vector by "cutting the tree" with the `cutre()` function like so:

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

Can you plot `z` colored by our hclust results.

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



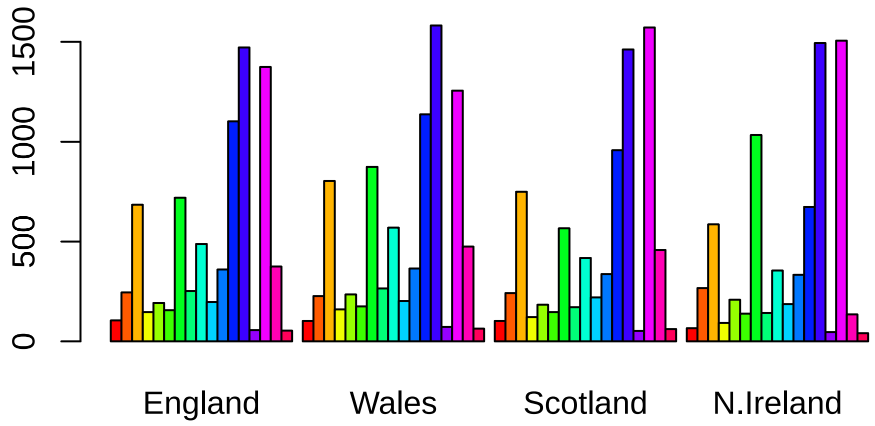
PCA of UK Food Data

Read data from the UK on food consumption in different parts of the UK.

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url, row.names=1)
head(x)
```

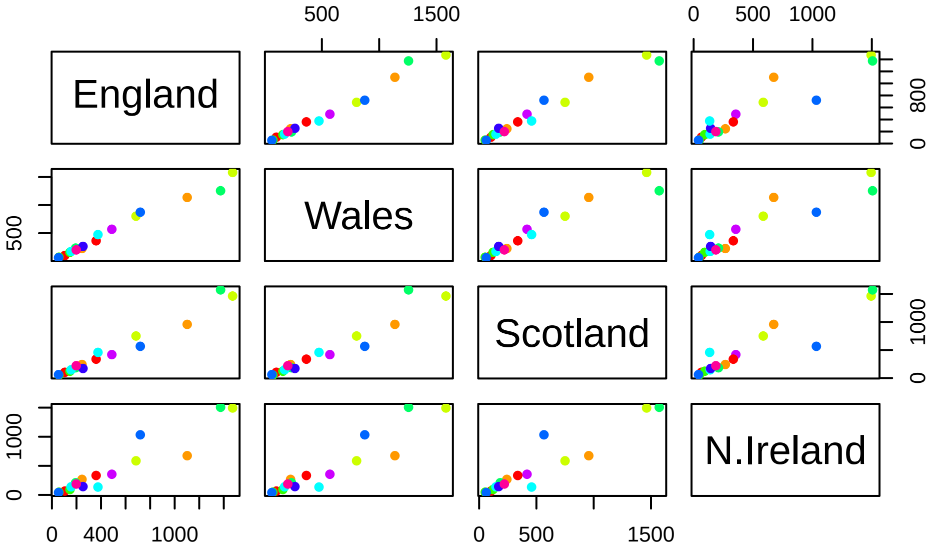
```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars  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

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



A so-called "Pairs" plot can be useful for small datasets like this one.

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



It is hard to see structure and trends in even this small dataset. How will we ever do this when we big datasets with 1,000s or 10s or thousands of things we are measuring…

PCA to the rescue

Let's see how PCA deals with this dataset. So main function in base R to do PCA is called `prcromp()`.

We can take the transpose of the data, flipping so the columns are foods using `t()`.

```r
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
```

Let's see what is inside this `pca` object that we created from running `prcomp()`.

```r
attributes(pca)
```

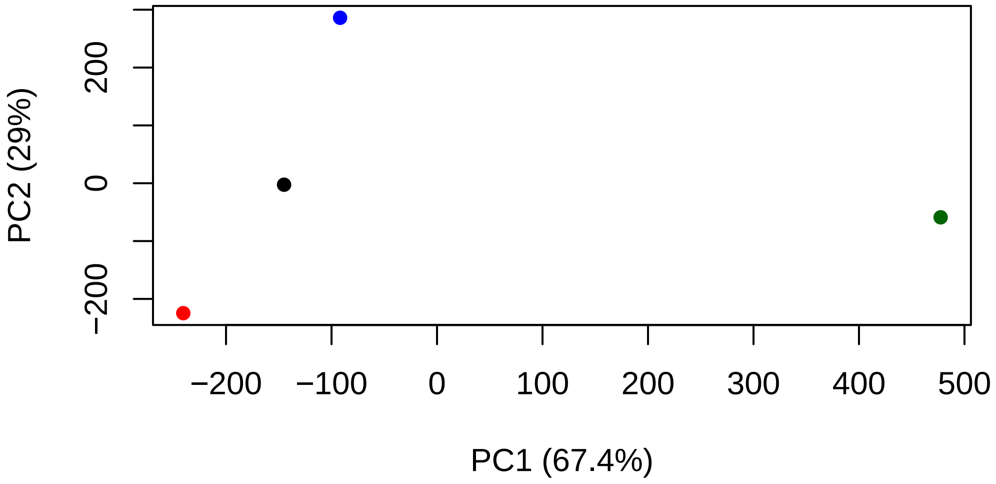
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"       

$class
[1] "prcomp"
```

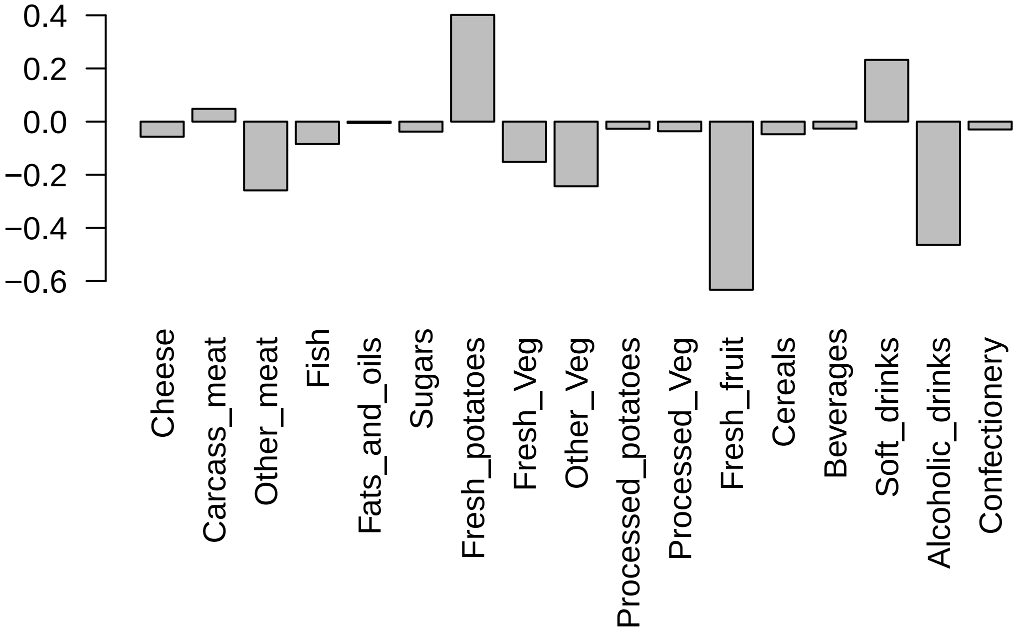
```r
pca$x
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -9.152022e-15
Wales     -240.52915 -224.646925 -56.475555  5.560040e-13
Scotland   -91.86934  286.081786 -44.415495 -6.638419e-13
N.Ireland  477.39164  -58.901862  -4.877895  1.329771e-13
```

```r
plot(pca$x[,1], pca$x[,2], col=c("black", "red", "blue", "darkgreen"), pch=16,
     xlab="PC1 (67.4%)", ylab="PC2 (29%)")
```



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


Cheese
Carcass_meat
Other_meat
Fish
Fats_and_oils
Sugars
Fresh_potatoes
Fresh_Veg
Other_Veg
Processed_potatoes
Processed_Veg
Fresh_fruit
Cereals
Beverages
Soft_drinks
Alcoholic_drinks
Confectionery