Class 7: Clustering and PCA

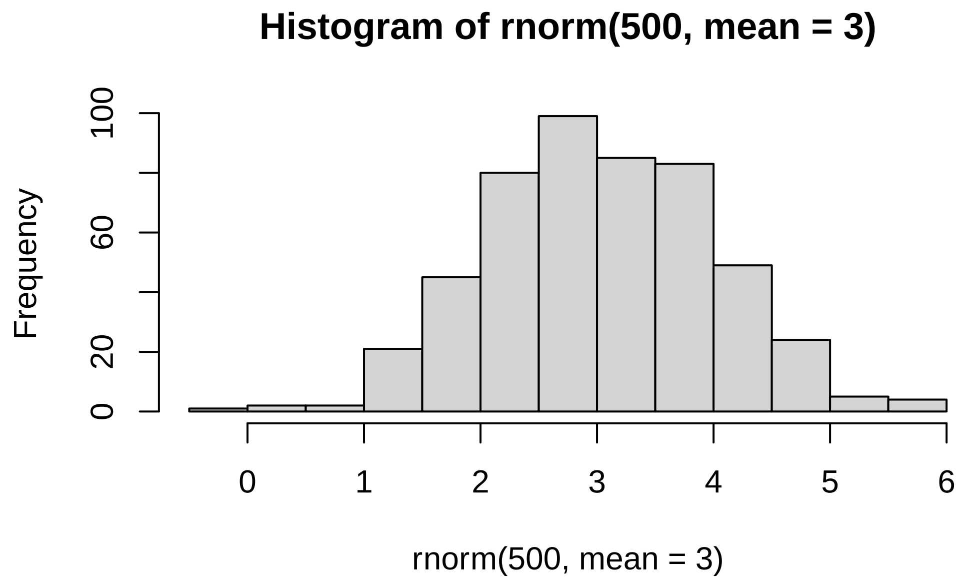
Emmanuel R

#CLustering

First let's make up some data to cluster so we can get a feel for these methods and how to work with them.

We can use the `rnorm()` function to get random numbers from a normal distriution around a given `mean`

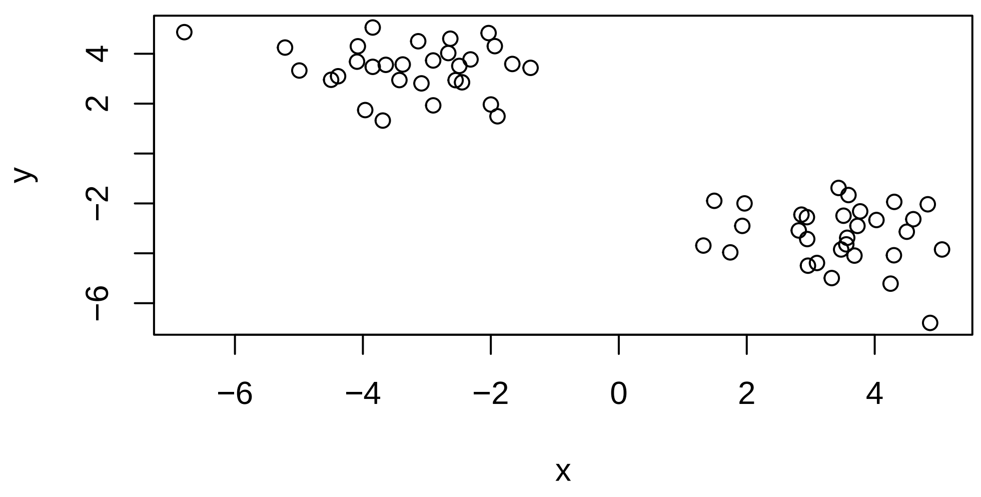
```
hist( rnorm(500, mean= 3))
```



Let's get 30 points for a mean of 3.

```r
tmp <- c(rnorm(30, mean= 3), rnorm(30, mean= -3))
```

```r
x <- cbind(x= tmp, y= rev(tmp))
plot(x)
```



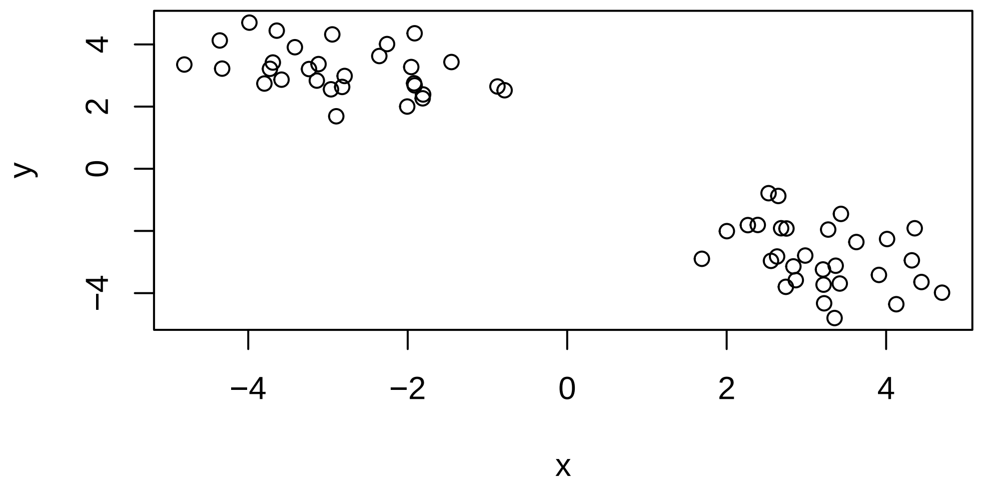
K-means clustering

Very popular clustering method that we can use with the `kmeans()` function in base R.

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

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

Cluster means:
          x         y
1 -3.26312  3.41505
2  3.41505 -3.26312

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] 70.52749 70.52749
 (between_SS / total_SS =  90.5 %)

Available components:

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

```r
# Generate some example data for clustering
tmp1 <- c(rnorm(30,-3), rnorm(30,3))
x <- data.frame(x=tmp1, y=rev(tmp1))

plot(x)
```



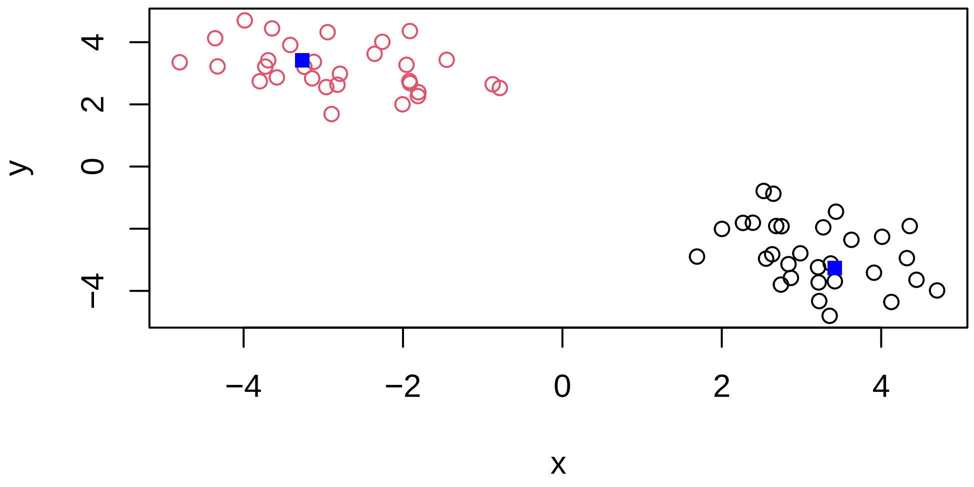
Generate some example data for clustering

```
km$size
```

```
[1] 30 30
```

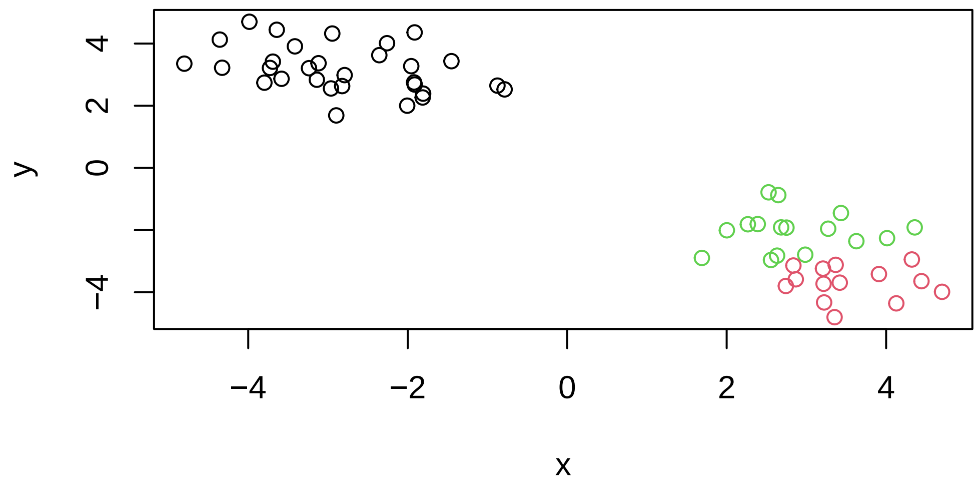
Q. Plot x colored by the means

```
mycols <- c(1, 2)
plot(x, col= km$cluster)
points(km$centers, col= "blue", pch =15, )
```



Q. Let's cluster into 3 groups or some `xdata` and make a plot.

```
km <- kmeans(x, centers= 3)
plot(x, col= km$cluster)
```



Hierarchial cluster

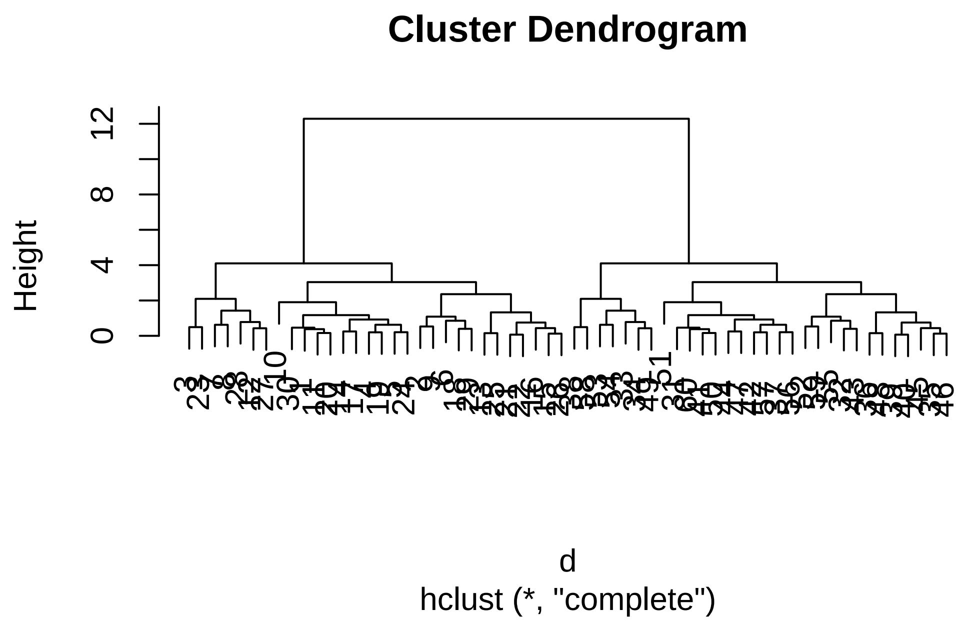
We can use the `hclust()` function for Hierarchial Clustering. Unlike `kmeans()`, where we could just passin our data as input, we need to give `hclust()` a “distance mark”.

We will use the `dist()` function to start with.

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

```

Call:
hclust(d = d)

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

```
plot(hc)
```



I can now cut my tree to yield a custer membership function with `cutree()`.

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

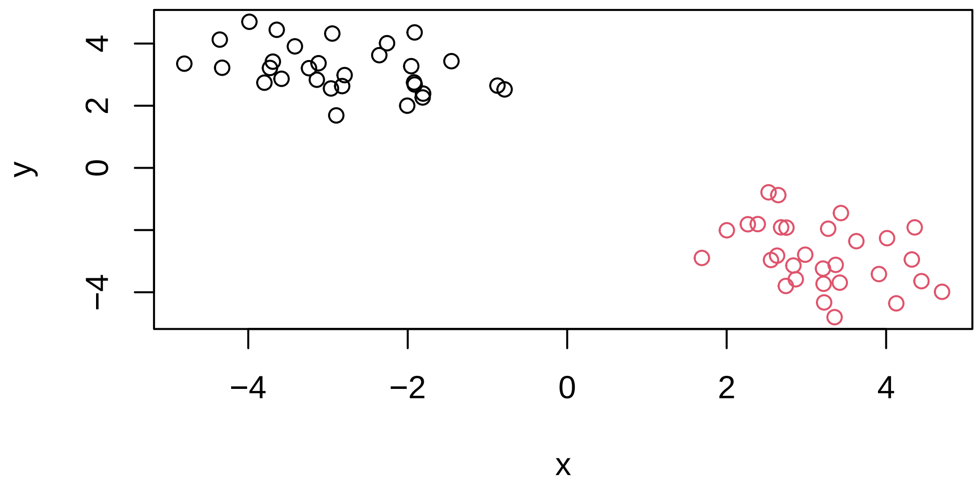
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

You can also tell 'cutree()' to cut where it yields "k" groups.

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

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



Principal Component Analysis (PCA)

Start of Lab Part 1

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url, row.names = 1)
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
Fresh_potatoes          720   874      566      1033
Fresh_Veg               253   265      171       143
Other_Veg               488   570      418       355
Processed_potatoes      198   203      220       187
Processed_Veg           360   365      337       334
```

```
Fresh_fruit            1102  1137      957       674
Cereals                1472  1582     1462      1494
Beverages                57    73       53        47
Soft_drinks            1374  1256     1572      1506
Alcoholic_drinks        375   475      458       135
Confectionery            54    64       62        41
```

> 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
dim(x)
```

```
[1] 17  4
```

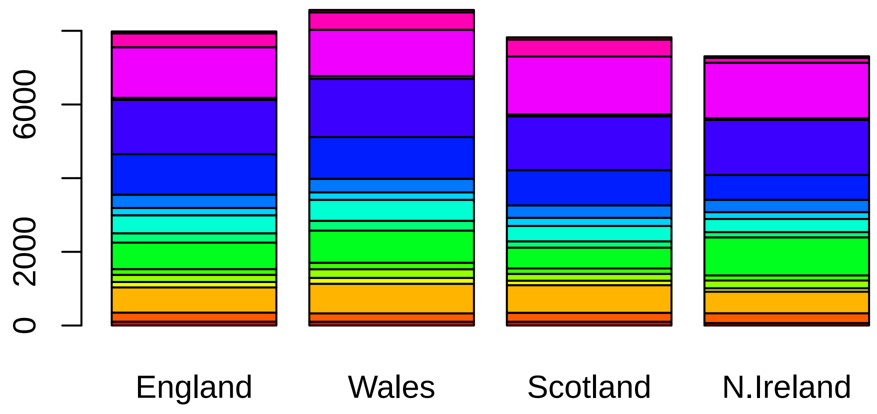
```r
head(x)
```

```
               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 prefer the second method. Much faster and easier.

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

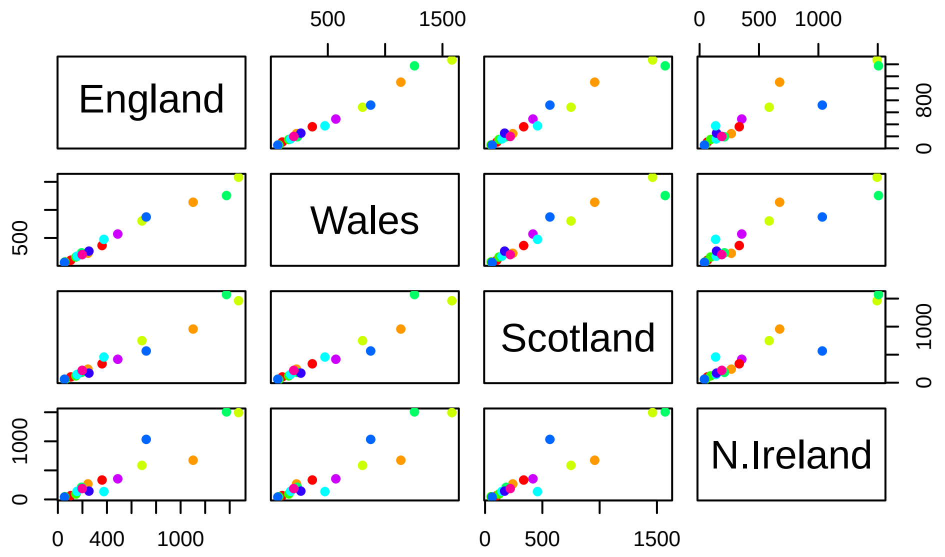


> Q3: Changing what optional argument in the above barplot() function results in the following plot?

Changing the `beside` argument will give the second plot result.

> Q5: Generating all pairwise plots may help somewhat. 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?

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



> Q6. What is the main differences between N. Ireland and the other countries of the UK in terms of this data-set?

The main PCA function in base R is called `prcomp()` it expects the transpose of our data.

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

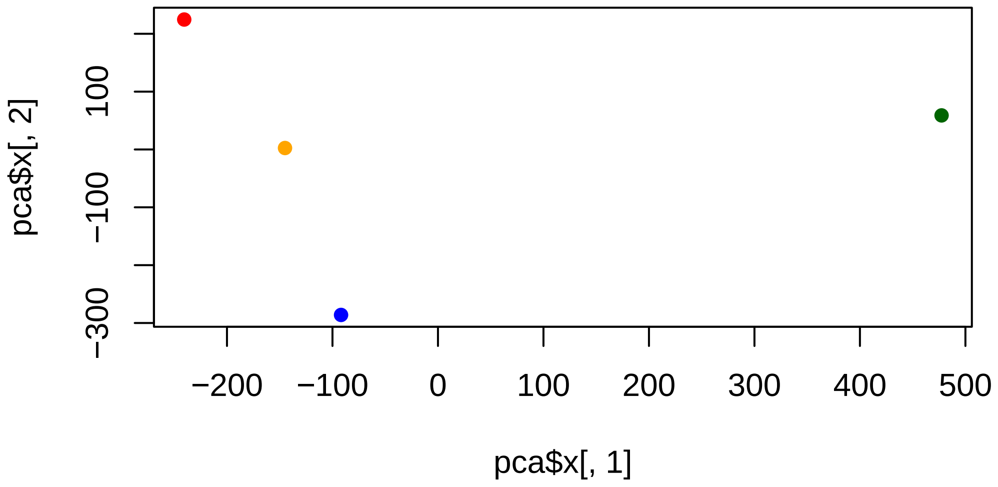
```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 4.189e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

```
pca$x
```

```
                PC1         PC2         PC3           PC4
England   -144.99315    2.532999 -105.768945  2.842865e-14
Wales     -240.52915  224.646925   56.475555  7.804382e-13
Scotland   -91.86934 -286.081786   44.415495 -9.614462e-13
N.Ireland  477.39164   58.901862    4.877895  1.448078e-13
```

Q7. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points. Q8. Customize your plot so that the colors of the country names match the colors in our UK and Ireland map and table at start of this document.

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


pca$x[, 2]
pca$x[, 1]