Class 07: Clustering and Principal Component Analysis (PCA)

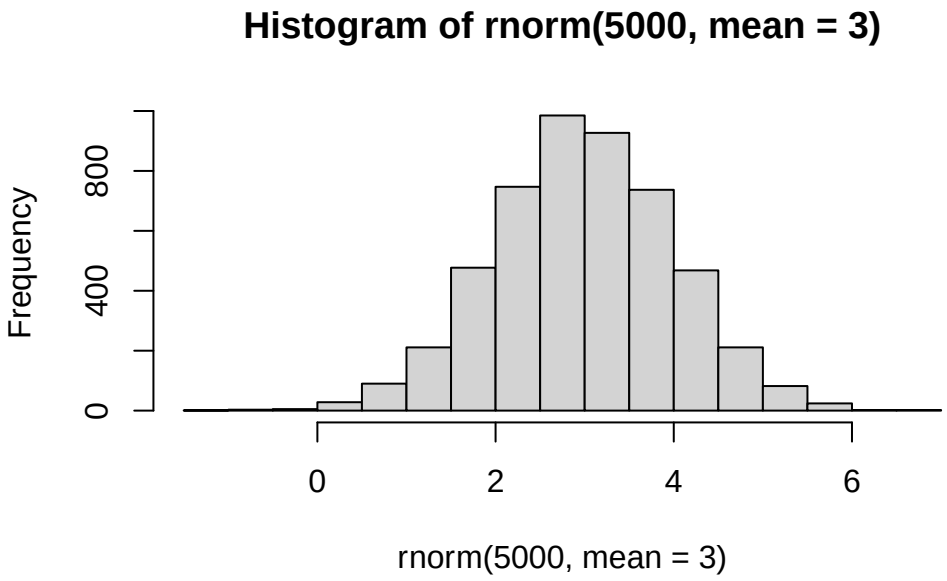
Katherine Lim (A15900881)

Clustering

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

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

```
hist(rnorm(5000, mean = 3))
```



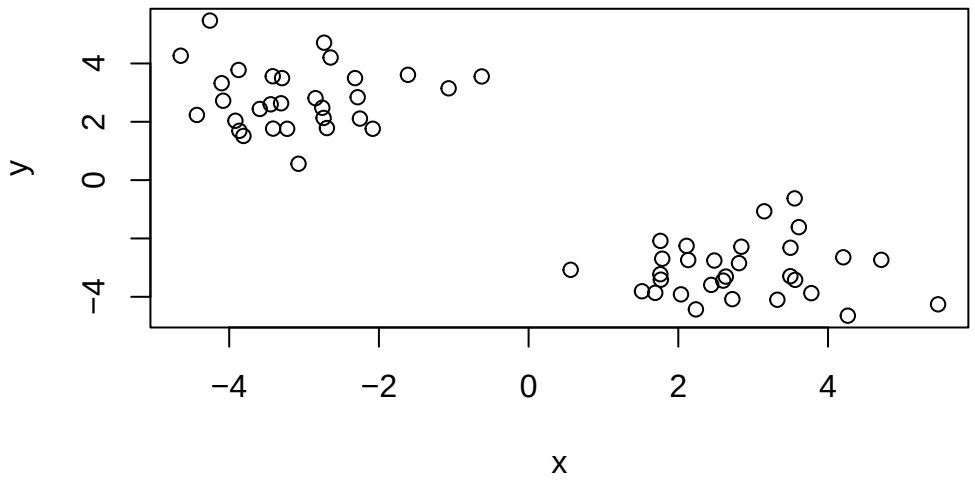
Let's get 30 points with a mean of 3 and another 30 points with a mean of -3.

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

```
 [1]  2.7221749  1.7610130  3.1479589  3.3230351  4.7118915  3.5609192
 [7]  5.4704404  2.8423294  2.6344734  0.5615510  2.1103464  3.4984296
[13]  2.4809643  1.7665275  1.5146378  4.2646871  1.7851993  2.1322905
[19]  1.6917206  3.7772593  3.6102570  2.4401436  3.4971513  2.0356523
[25]  1.7601399  2.2352283  3.5543269  4.2038271  2.8108897  2.5994548
[31] -3.4455646 -2.8467094 -2.6456092 -0.6265974 -4.4319945 -3.2259492
[37] -3.9172722 -3.2940387 -3.5913853 -1.6120082 -3.8750914 -3.8642971
[43] -2.7390023 -2.6947555 -4.6477614 -3.8096290 -3.4132813 -2.7565017
[49] -2.3186777 -2.2536329 -3.0742690 -3.3068268 -2.2834392 -4.2577535
[55] -3.4183168 -2.7324113 -4.1007673 -1.0686109 -2.0824257 -4.0806748
```

Put two of these together.

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



K-mean clustering.

Very popluar clustering method, especially for big data sets, that we can use with the `kmeans()` function in base R.

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

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

Cluster means:
          x         y
1 -3.080508  2.816831
2  2.816831 -3.080508

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 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] 60.08489 60.08489
 (between_SS / total_SS =  89.7 %)

Available components:

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

Q. What 'component' of your result object details:

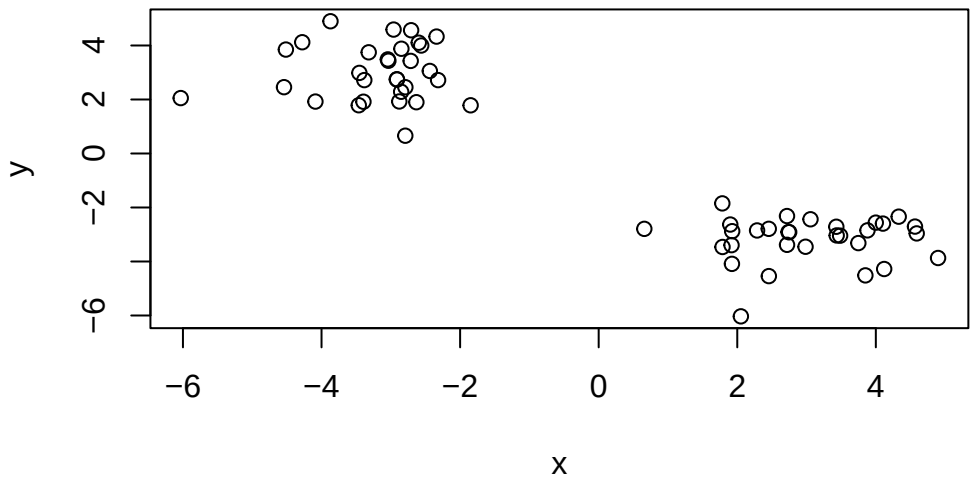
- cluster size?

```
km$size
```

```
[1] 30 30
```

- cluster assignment/membership?

```
tmp <- c(rnorm(30, -3), rnorm(30, 3))
x <- data.frame(x=tmp, y=rev(tmp))

plot(x)
```



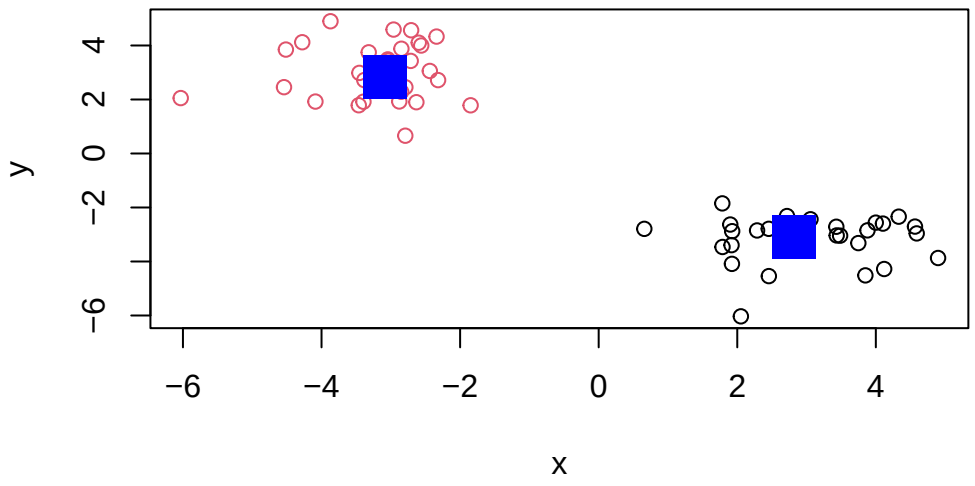
- cluster center?

```
km$centers
```

```
          x         y
1 -3.080508  2.816831
2  2.816831 -3.080508
```

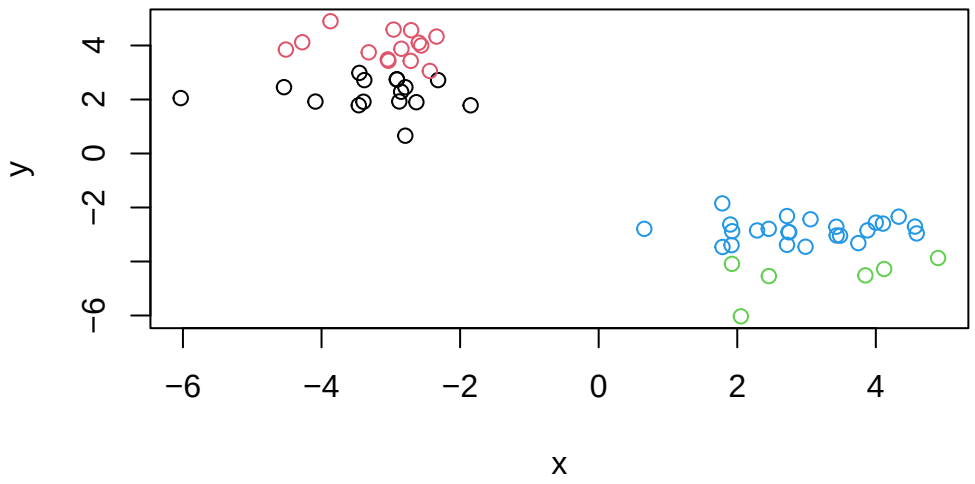
Q. Plot x colored by the kmeans cluster assignment and add cluster centers as blue points.

```
plot(x, col = km$cluster)
points(km$centers, col = "blue", pch = 15, cex = 3)
```



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

```
km <- kmeans(x, centers = 4)
plot(x, col = km$cluster)
```



Hierarchical clustering

We can use the `hclust()` function for hierarchical clustering.

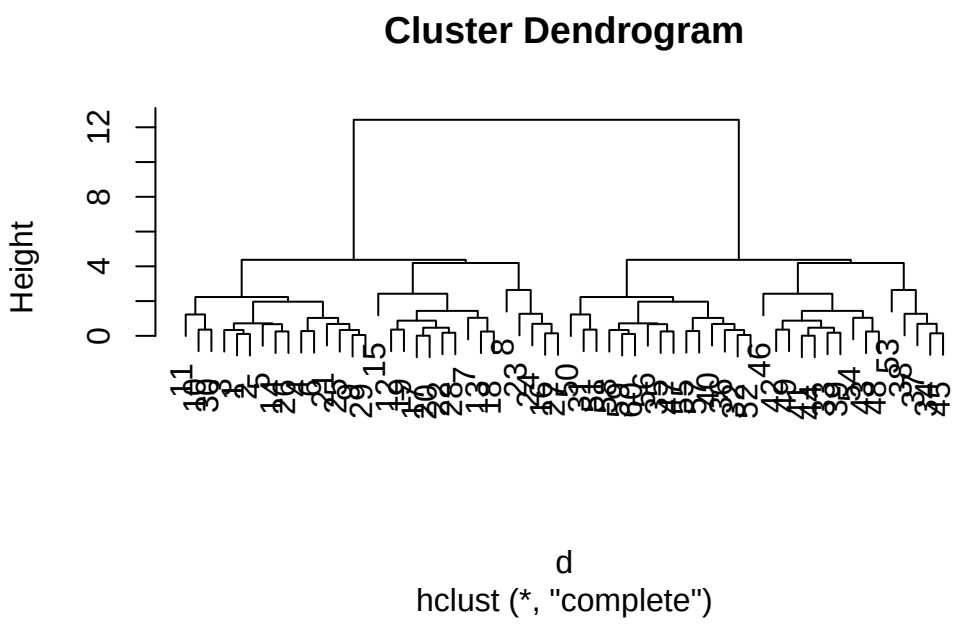
Unlike `kmeans()` where we could just pass our data as input, we need to give `hclust()` a "distance matrix".

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 with the `cutree()` to yield a cluster membership vector.

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

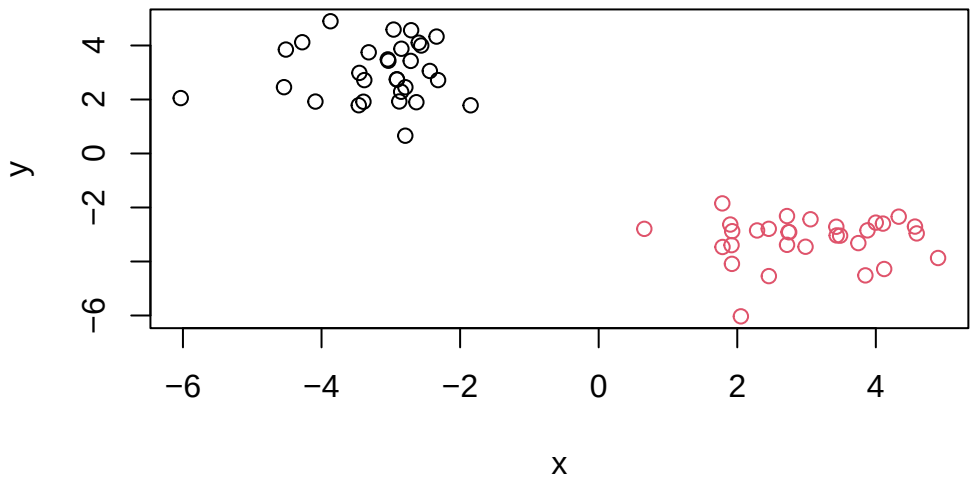
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

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

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

```
 [1] 1 1 1 1 1 1 2 3 1 1 1 2 2 1 2 4 2 2 2 2 1 2 4 4 1 1 4 2 1 1 5 5 6 7 5 5 7 7
[39] 6 5 6 6 6 6 7 6 5 6 6 5 5 5 8 6 5 5 5 5 5 5
```

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



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?

```
# Show the number of rows.
nrow(x)
```

```
[1] 17
```

```
# Show the number of columns.
ncol(x)
```

```
[1] 5
```

```
# Show the dimensions of the data frame.
dim(x)
```

```
[1] 17  5
```

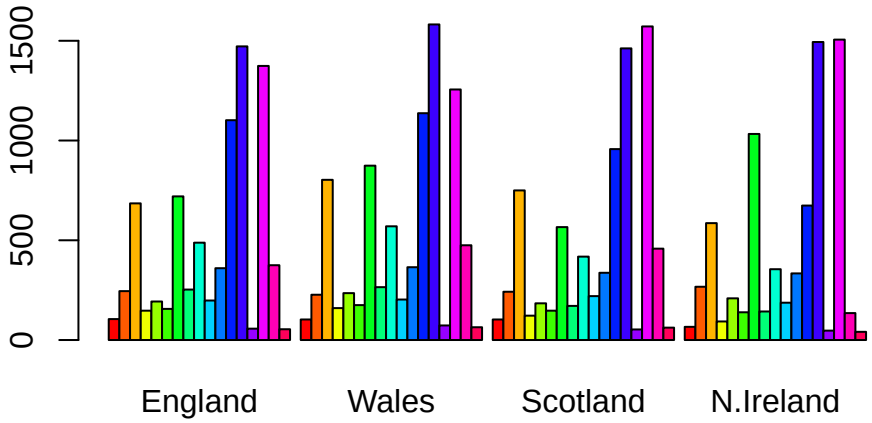
Q2. Which approach to solving the 'row-names problem' mentioned above do you prefer and why? Is one approach more robust than another under certain circumstances?

```
# Properly set row names
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
```

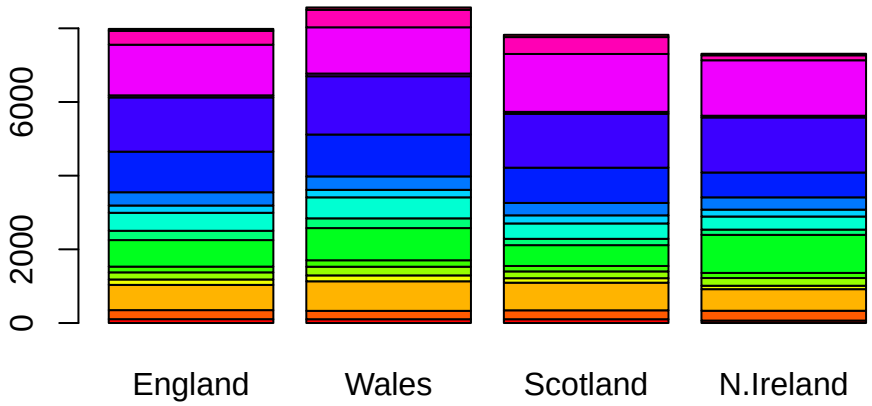
I prefer to establish row names when I read in a csv file because it is less destructive and properly sets row names from the start.

```
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?

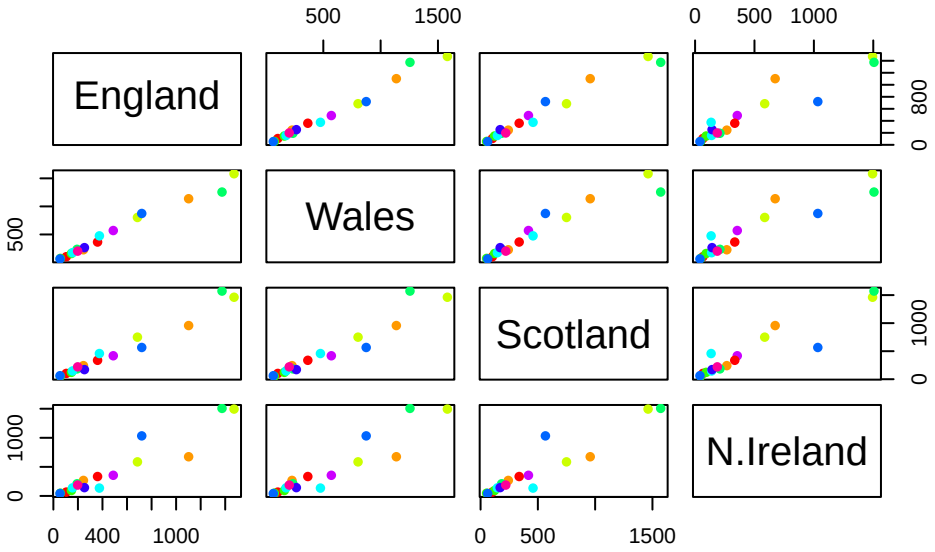
```
barplot(as.matrix(x), beside = F, col = rainbow(nrow(x)))
```



Changing beside from TRUE to FALSE results in the above plot.

> 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)
```



If a given point lies on the diagonal for a given plot, that means the two variables are comparable.

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

The blue and orange plots are the main differences between N. Ireland and other countries of the UK. N. Ireland consumes less of the blue dot and more of the orange dot.

> Q7. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points.

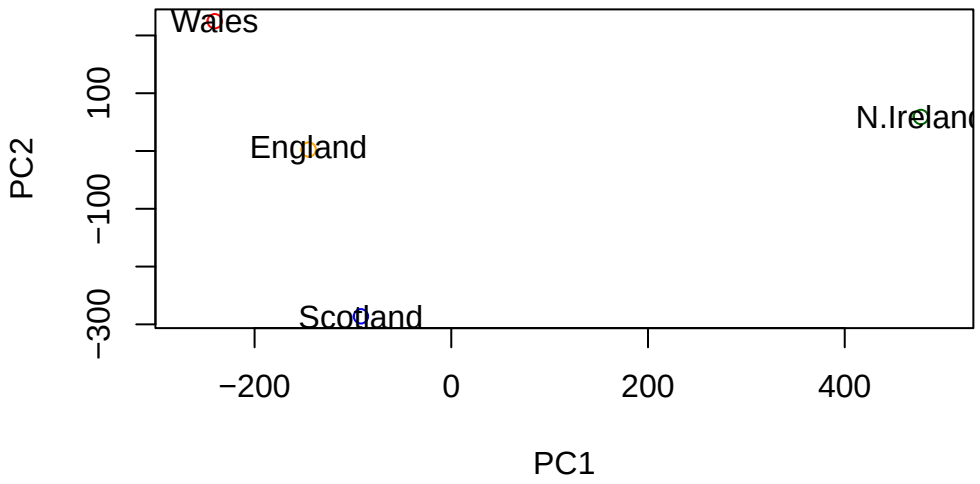
```
pca <- prcomp( t(x) )
summary(pca)
```

```
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
```

```
attributes(pca)
```

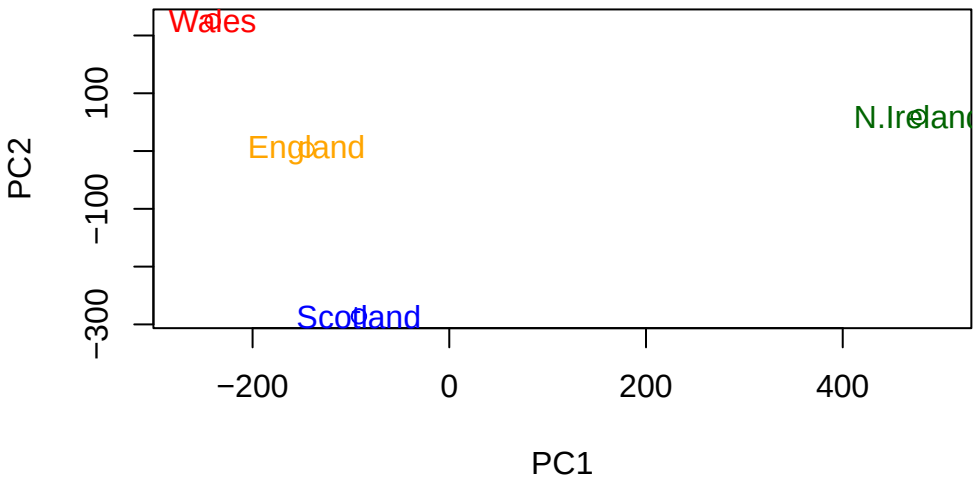
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

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



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.

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


PC2
100
−100
−300
−200
0
200
400
PC1