Class 7 Machine Learning 1

Karleen Guerrero

Quarto

Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see https://quarto.org.

Running Code

When you click the **Render** button a document will be generated that includes both content and the output of embedded code. You can embed code like this:

```
1 + 1
```

```
[1] 2
```

You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).

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

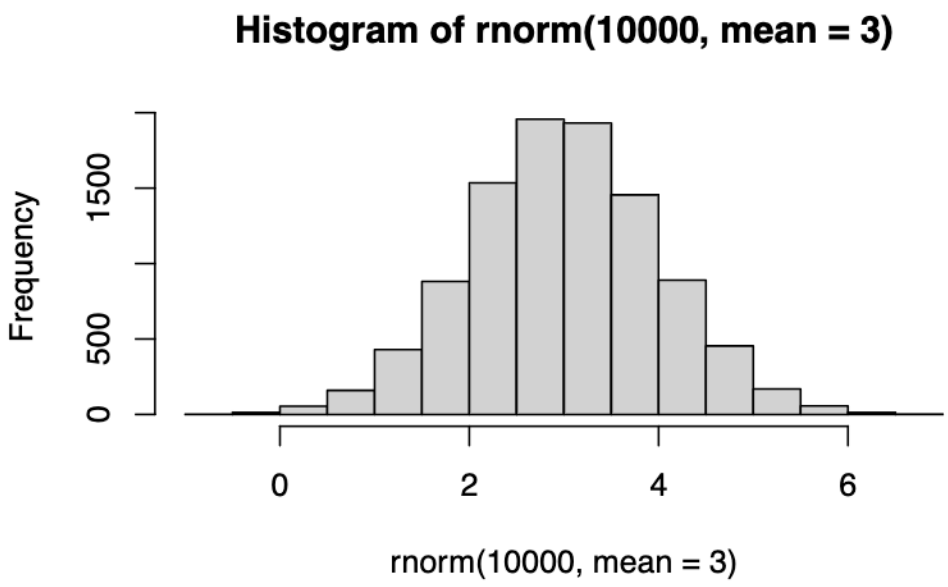
The goal here is to find groups/cluster in your input data.

First I will make up some data with clear groups For this i qill use the 'rnorm()' function:

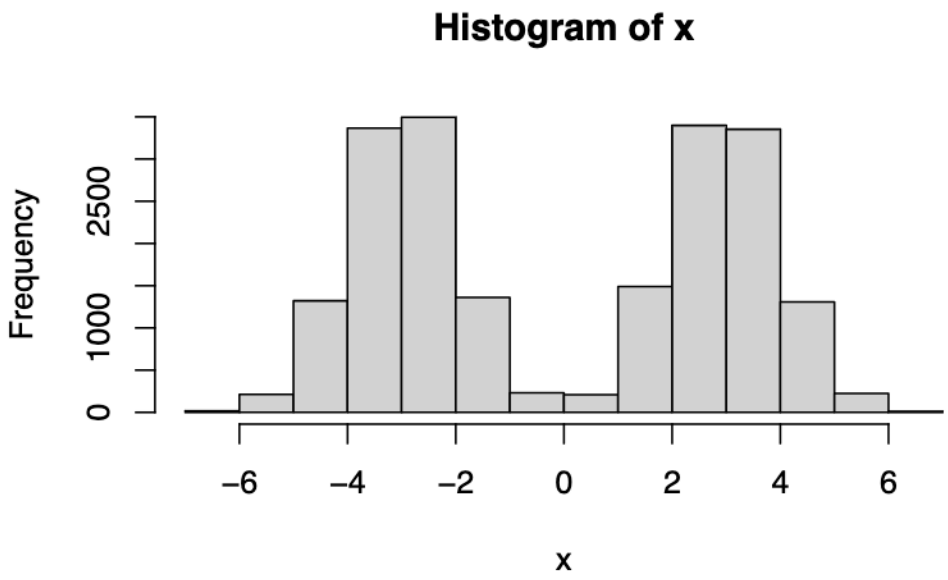
```
rnorm(10)
```

```
 [1]  0.2390210 -0.2193029  1.9033834  0.1759123  0.6524428 -0.5123779
 [7]  0.4448086  1.1708013 -0.7531579 -0.3497160
```

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




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

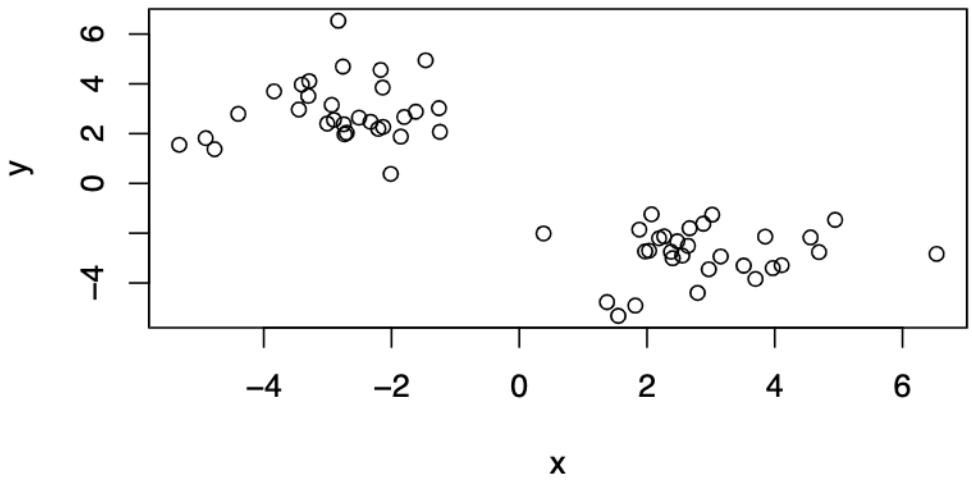



```
n <- 30
x <- c(rnorm(n, -3), rnorm(n, +3))
y<- rev(x)

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

```
             x        y
[1,] -2.734450 1.970921
[2,] -3.303594 3.513470
[3,] -1.465798 4.944847
[4,] -2.935672 3.152970
[5,] -2.328262 2.473638
[6,] -5.322928 1.549587
```

```
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

```
km <- kmeans(z, center = 2)
km
```

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

Cluster means:
          x         y
1  2.910983 -2.801661
2 -2.801661  2.910983

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] 76.97413 76.97413
 (between_SS / total_SS =  86.4 %)

Available components:

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

Results in kmean object 'km'

```
attributes(km)
```

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

$class
[1] "kmeans"
```

Cluster center?

```
km$size
```

```
[1] 30 30
```

Cluster assignment/membership?

```
km$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 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
```

Cluster center?

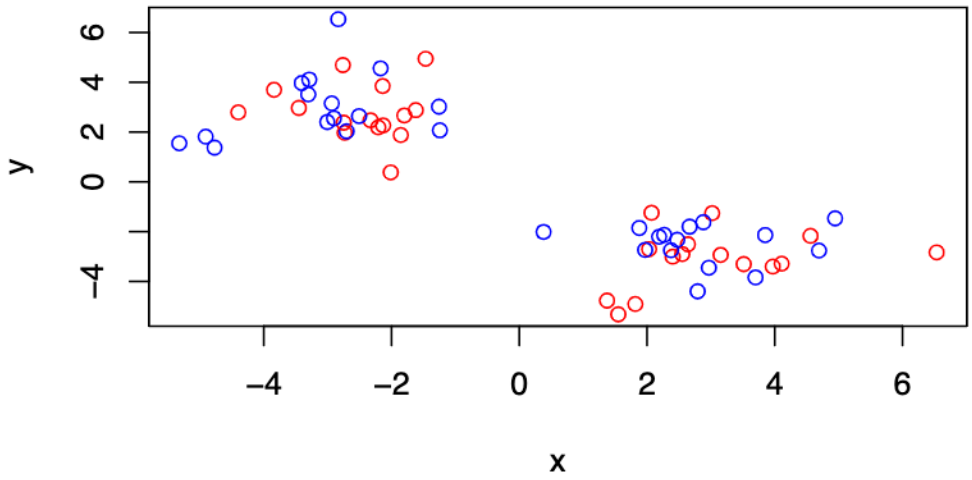
```
km$centers
```

```
          x         y
1  2.910983 -2.801661
2 -2.801661  2.910983
```

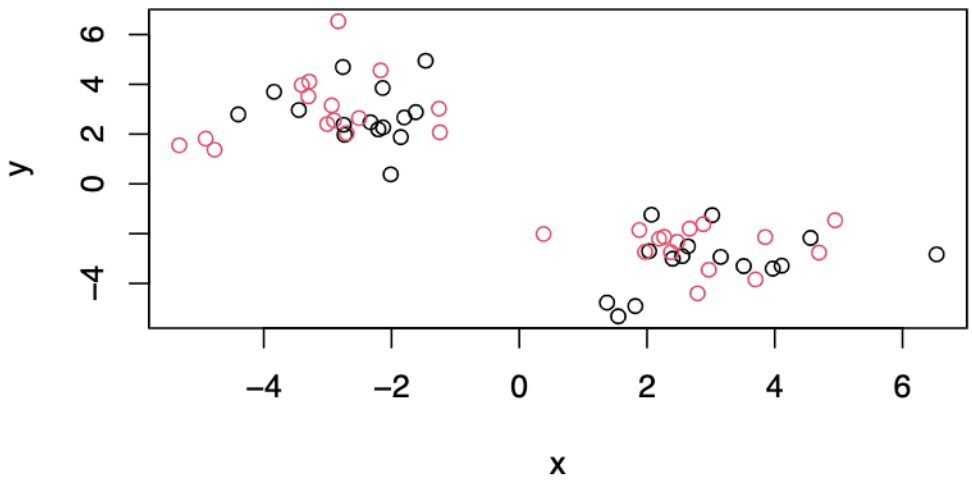
Q. Plot x colored by the kmeans cluster assignment and add cluster centers as blue point?

R will recycle the shorter color vector to be the same length as the longer (number if data) in z

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

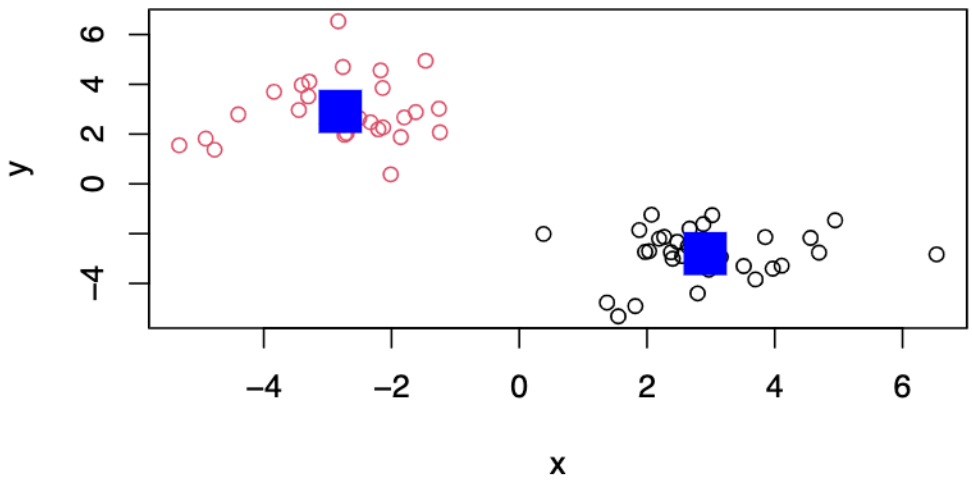



```
plot(z, col=c(1,2) )
```



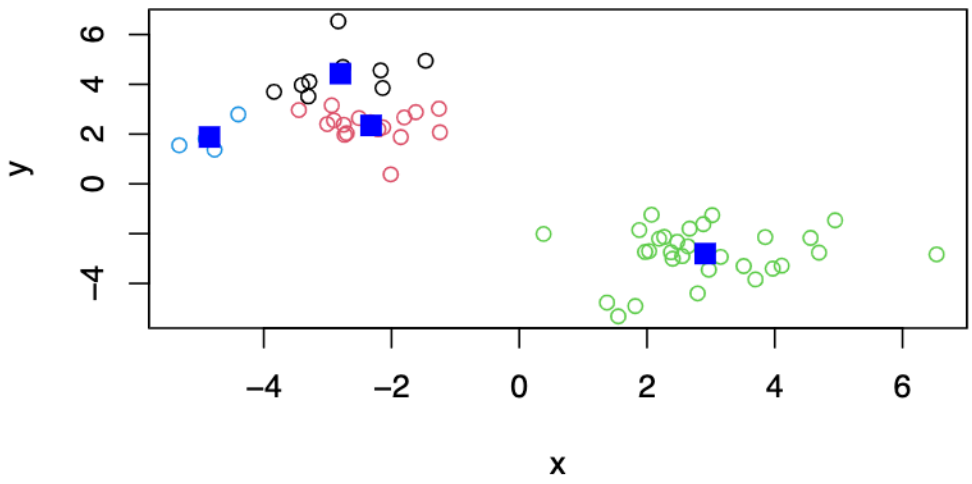
we can use tge 'points()' function to add new points to an existing plot... like the scluster centers

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



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

```
km4 <- kmeans(z, center = 4)
plot(z, col = km4$cluster)
points(km4$centers, col ="blue", pch = 15, cex = 1.5)
```



Hierarchiacal Clustering

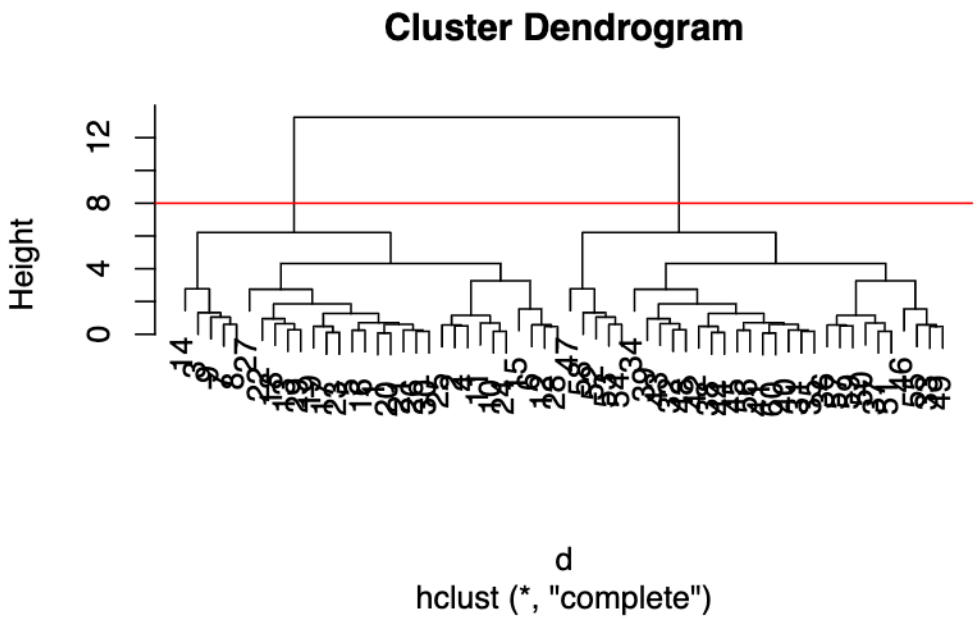
Lets take our same data 'z' and see how hclust works Fist we need a distance matrix of our data to be clustered

```
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")
```



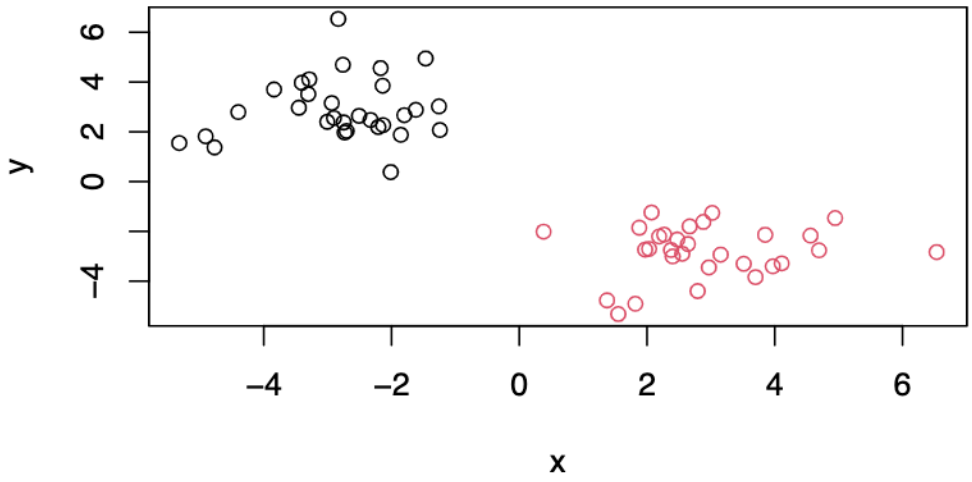
I can et nmy cluster membership vectot by “cutting te tree” the ‘cuttree()’ function like so

```
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
```

Can you plot ‘z’ again colored by our hclust results:

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



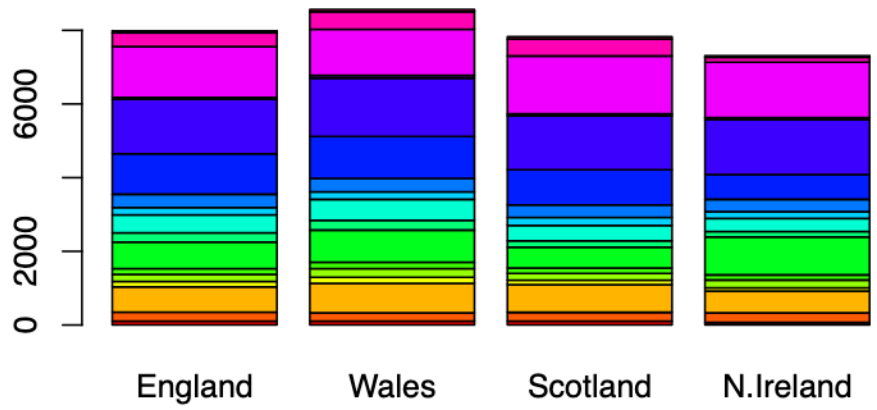
1. PCA of UK food data

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)
```

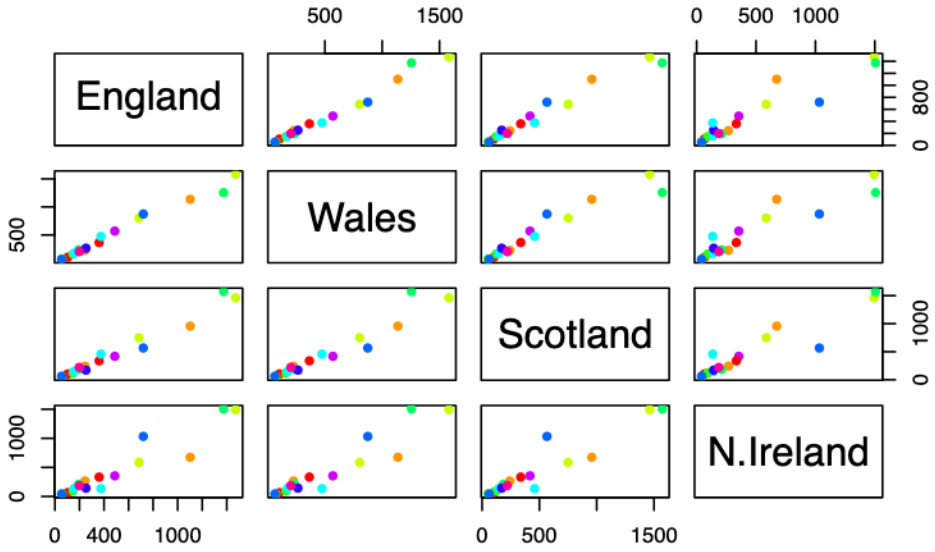
```
              England Wales Scotland N.Ireland
Cheese            105   103      103        66
Carcass_meat      245   227      242       267
Other_meat        685   803      750       586
Fish              147   160      122        93
Fats_and_oils     193   235      184       209
Sugars        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

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



Pairs plot can be useful for small data sets

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



IT gets hard to see structure and trends in even this small data set, how will we ever do this when we have big datasets with 1,000s or 10s of thousands of things we are measuring...

Lets see how PCA deals with this dataset, So main function in base R to PCA is called 'prcomp()'

```
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
```

Lets see what is inside this 'pca' object that we created from running 'pca'

```
attributes(pca)
```

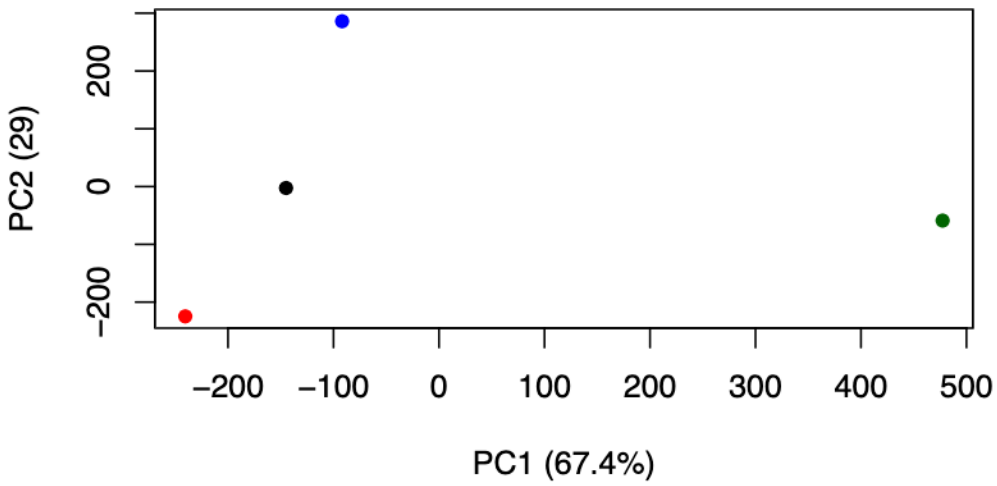
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

```
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
```

```
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 )
```