Class 7: Machine Learning 1

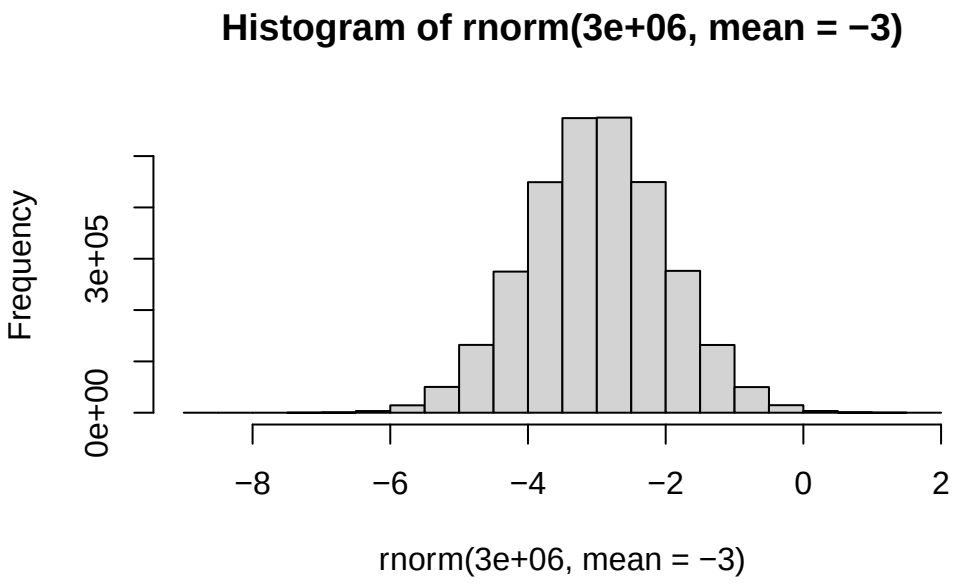
Bianca Barriga

In this class we will explor and get practice with clustering and Principal Componenet Analysis (PCA).

#Clustering with K-means

First we will make up some data to ncluster where we know what the result should be.

```
hist(rnorm(3000000, mean =-3))
```



I want a little vector with two groupings in it:

```
rnorm(30, -3)
```

```
 [1] -2.9342767 -2.9337375 -2.0920734 -3.0931229 -1.5436762 -4.6131963
 [7] -2.5829253 -3.6296857 -0.7480407 -3.9046848 -3.9096887 -4.5715471
[13] -2.9230472 -3.4343926 -3.2659395 -3.4149734 -2.4941712 -2.4197232
[19] -3.2131324 -1.2521018 -3.2926879 -1.9879352 -3.8532400 -3.7775698
[25] -2.8523132 -2.0874349 -4.5711295 -3.3751612 -4.2560691 -0.4771003
```

```
rnorm(30, +3)
```

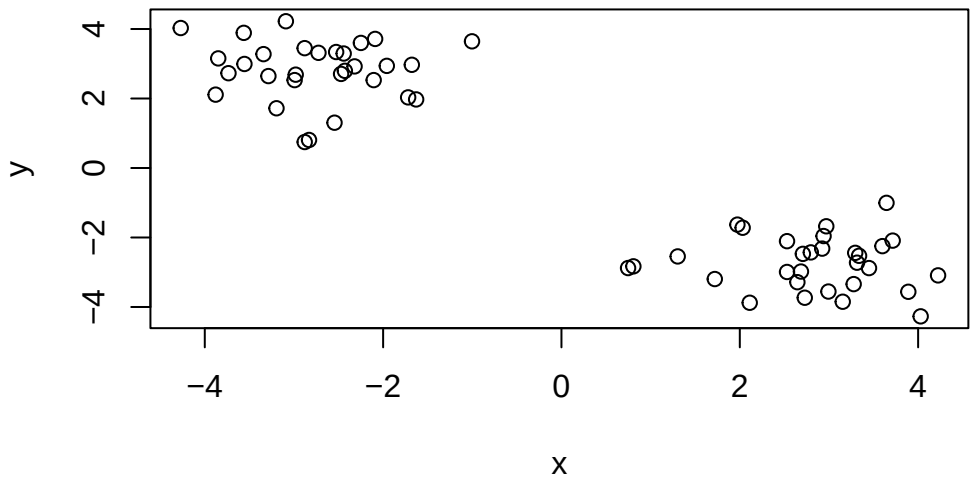
```
 [1] 4.620849 2.348040 2.563708 3.915033 2.709058 3.043555 2.177440 3.106070
 [9] 2.707766 3.596572 2.855885 3.424460 4.260059 3.442906 4.518738 3.193331
[17] 1.317369 2.427352 3.621524 3.400151 4.044312 2.612509 4.101492 2.469969
[25] 4.438793 3.558914 1.839974 2.517312 1.641198 3.539280
```

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

```
         x        y
1 -2.725085 3.314627
2 -2.320150 2.923761
3 -2.544984 1.303561
4 -2.992493 2.529586
5 -2.880022 3.449035
6 -1.719454 2.032130
```

Lets have a look:

```
plot(x)
```



Have a look at help for kmeans()

```
#input data = x, centers = the number of clusters
km <- kmeans(x, centers= 2 )
km
```

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

Cluster means:
          x         y
1  2.802343 -2.740696
2 -2.740696  2.802343

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] 39.09503 39.09503
 (between_SS / total_SS =  92.2 %)

Available components:
```

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

```
#cluster means is where the centroid of the clusters are
```

It is important to not just run the analysis but to be able to get your important results back in a way that we can do things with them.

Q. How do I find the cluster size

```
km$size
```

```
[1] 30 30
```

Q. How about the cluster centers?

```
km$centers
```

```
          x         y
1  2.802343 -2.740696
2 -2.740696  2.802343
```

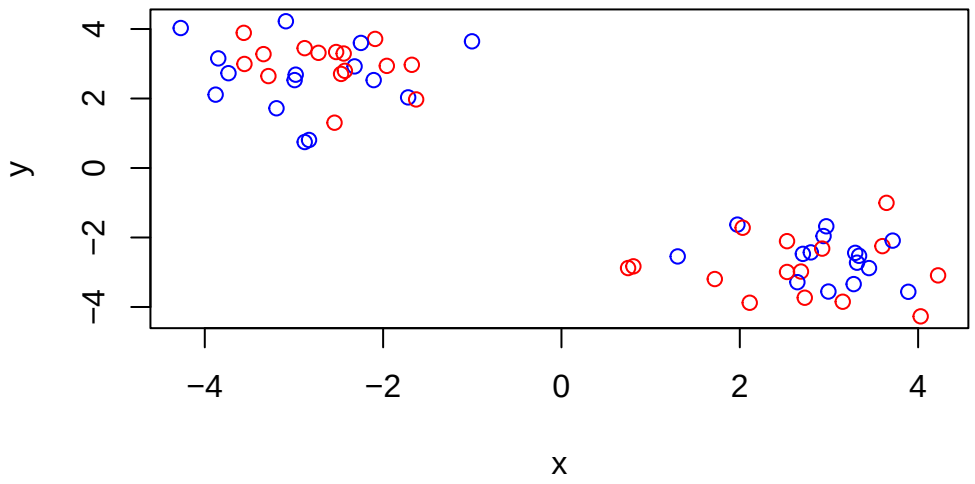
Q. How about the main result - the cluster assignment?

```
km$cluster
```

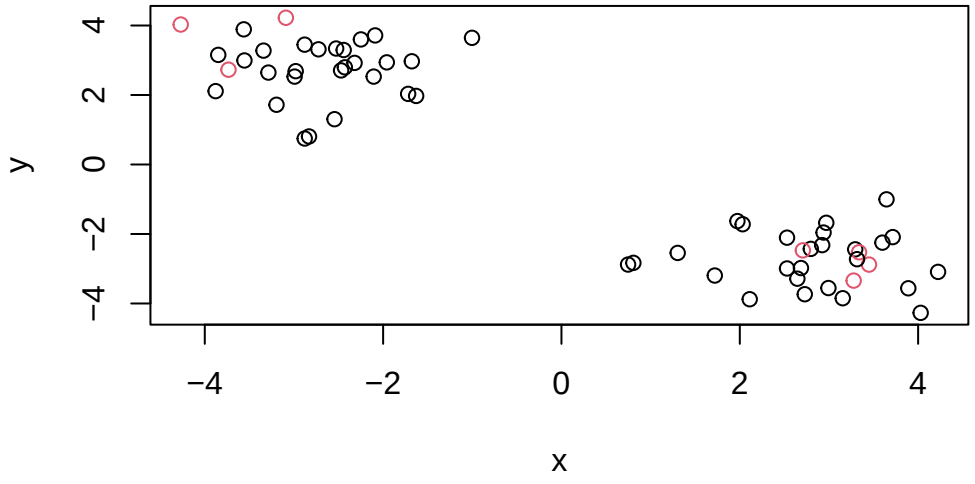
```
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Q. Can we make a summary figure showing the result? - The points colored by cluster assignment and add the cluster centers as a different color?

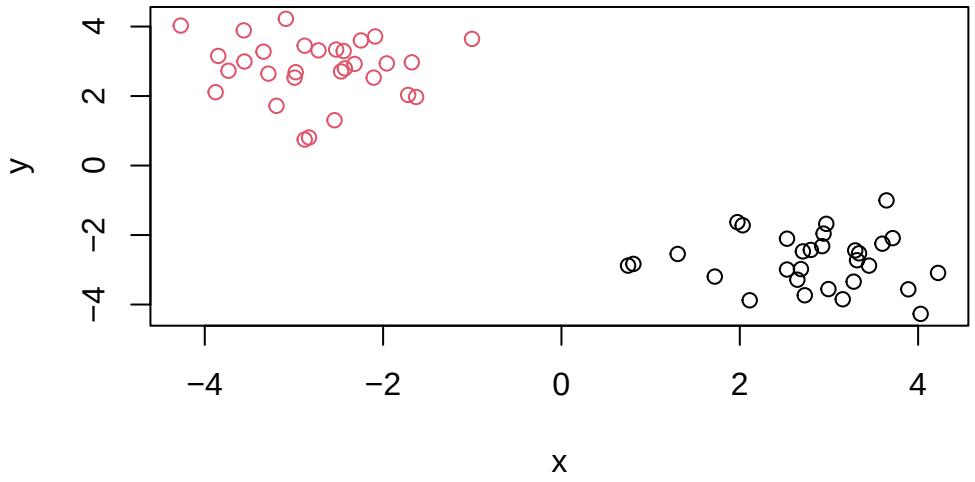
```
plot(x, col=c("red", "blue"))
```




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

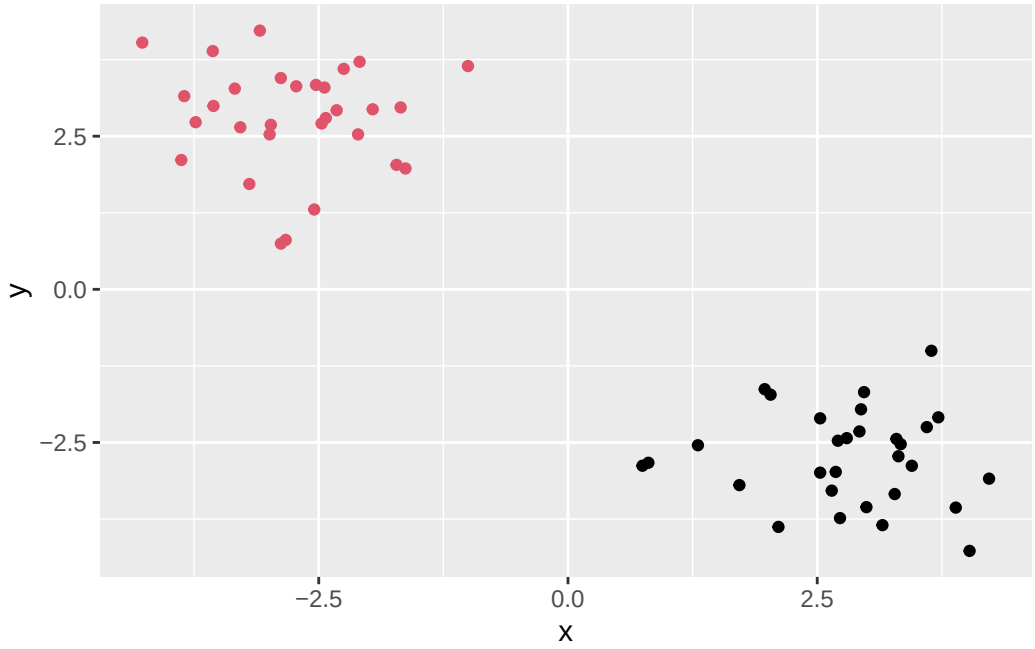


```
plot(x, col=km$cluster)
```



Lets plot the data with ggplot. I need 3 things: librar

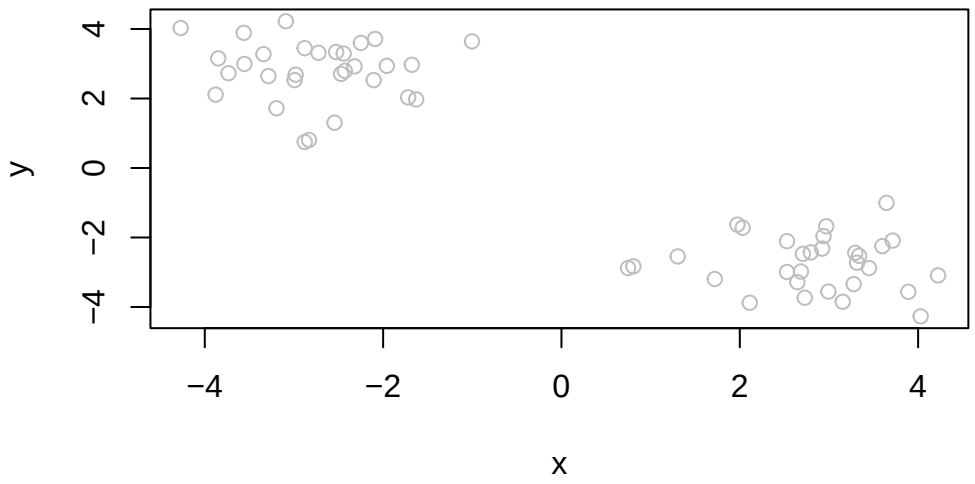
```
library(ggplot2)
ggplot(x) +
  aes(x,y) +
  geom_point(col=km$cluster)
```




```
#Make up a color vector
mycols <- rep("gray", 60)
mycols
```

```
 [1] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
[11] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
[21] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
[31] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
[41] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
[51] "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray" "gray"
```

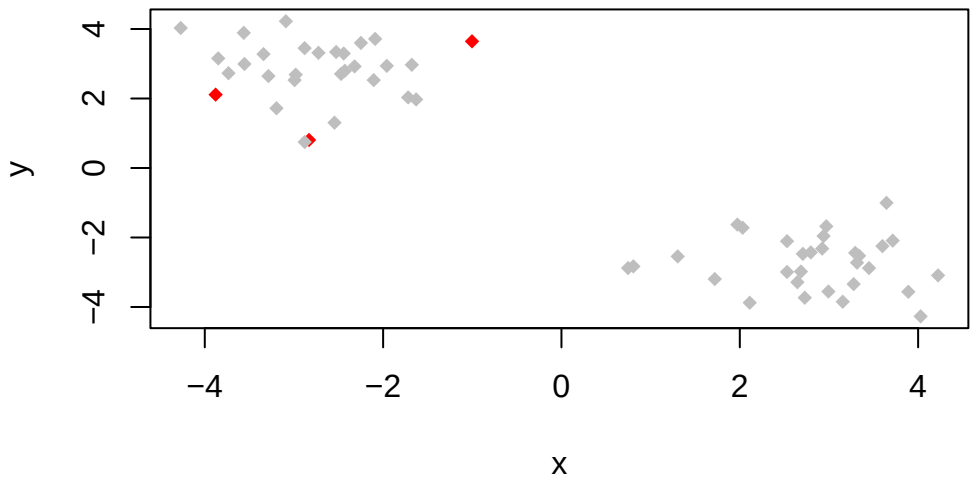
```
plot(x, col=mycols)
```



Let's highlight points 10, 12, and 20.

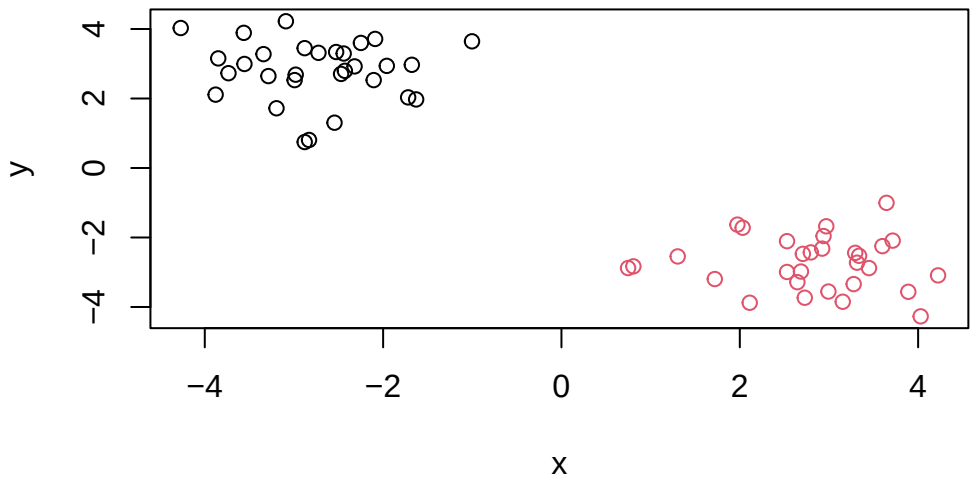
```
mycols[c(10, 12, 20)] <- "red"

plot(x, col=mycols, pch = 18)
```



Play with different numbers of centers

```
km <- kmeans(x,centers=2)
plot(x, col=km$cluster)
```




```
km$tot.withinss
```

```
[1] 78.19006
```

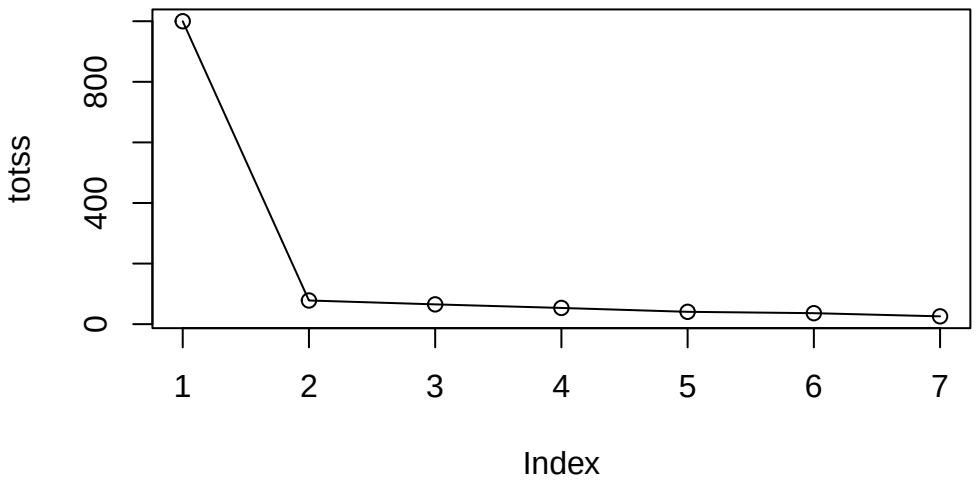
```
#can use to generate a SCREE plot
```

What we want to do is try out different numbers of K from 1 to 7. We can write a `for` loop to do this for us to store the `tot.withinss` each time.

```
#create empty vector
totss <- NULL
#designate k
k <- 1:7

for(i in k) {
  #cat(i, "\n")
 totss <-  c(totss,kmeans(x, centers =i)$tot.withinss)
}
```

```
plot(totss, typ='o')
```



#Hierarchical Clustering

We can't just give the `hclust()` function teh data `x` like we did for `kmeans()`. Each cluster is a lot more flexible so we can give it identities.

We need to first calculate a "distance matrix". The `dist()` function by default will calculate euclidean distance.

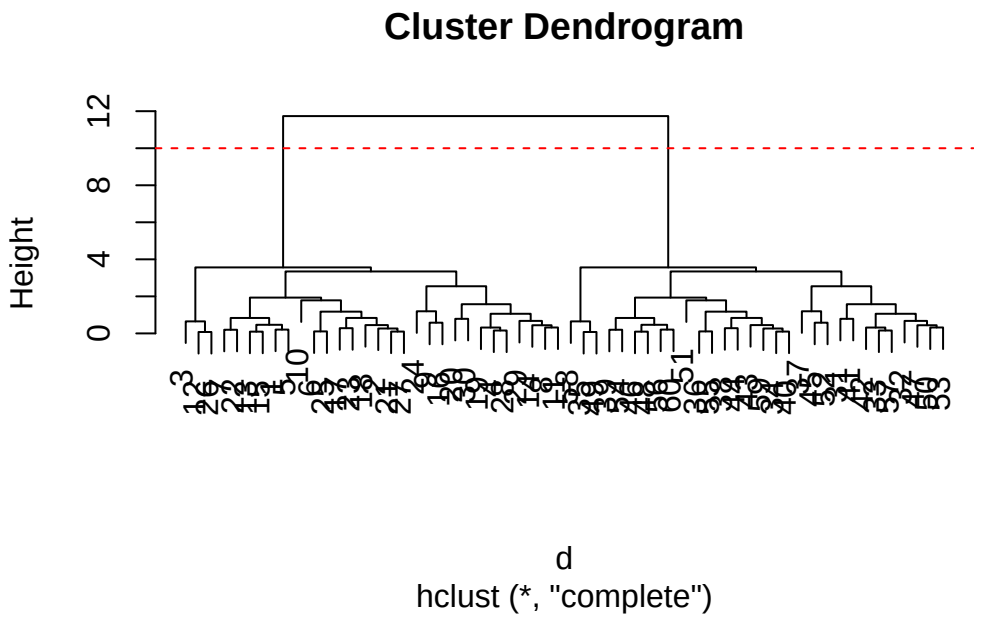
```
d <- dist(x)
hc <- hclust(d)
hc
```

```
Call:
hclust(d = d)

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

The print out from `hclust` is not very helpful, but the plot method is very useful. Height is important for relatedness.

```
plot(hc)
abline(h=10, col="red", lty=2)
```



To get my all important cluster membership vector out of a hclust ovject I can use the `cutree()`

```
cutree(hc, h=10)
```

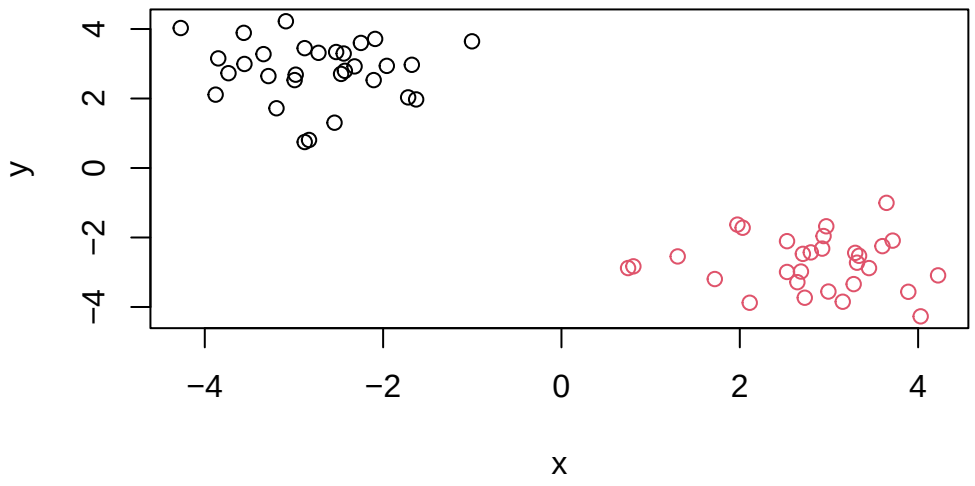
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

You can also set a `k=` argument for `cutree()`.

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

Let's make a figure of our hclust results.

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



#Principal Component Analysis (PCA) PCA projects the features onto a principal componenet. The motivation is to reduce the features dimensionality while only losing a small amount of information. Principal components are new low dimensional axis (or surfaces) closest to the observations.

PCA objectives -reduce dimensionality -visualize multidimensional data -choose the most useful variables -identify groups of objects (e.g. genes.samples) -identify outliers

The main base R function to do PCA is called `prcomp()`.

First I need to import the data.

```
url <- "http://tinyurl.com/UK-foods"
x <- read.csv(url)
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
```

The row names are showing up as data points. We need to change this to row names and not the first column.

```
rownames(x) <-x[,1]
x <- x[,-1]
head(x)
```

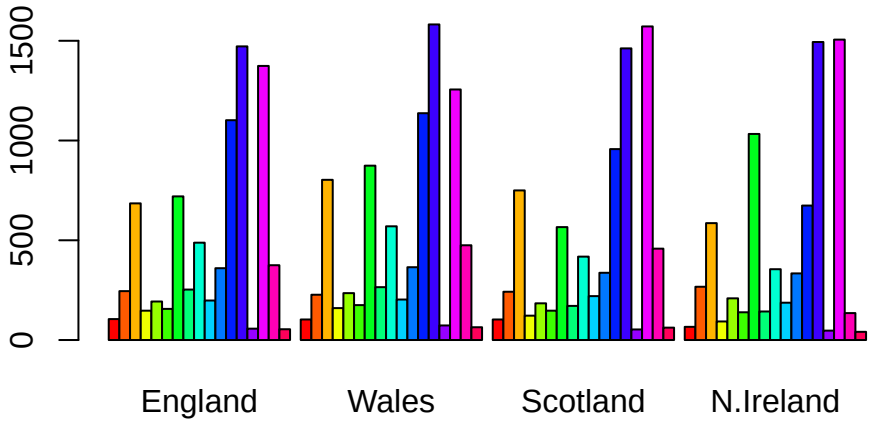
```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars  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

A better way to do this is, because if you run the code above multiple times it will continue to replace the row names. We can correct the row names when we upload the file.

```
x <-read.csv(url, row.names = 1)
head(x)
```

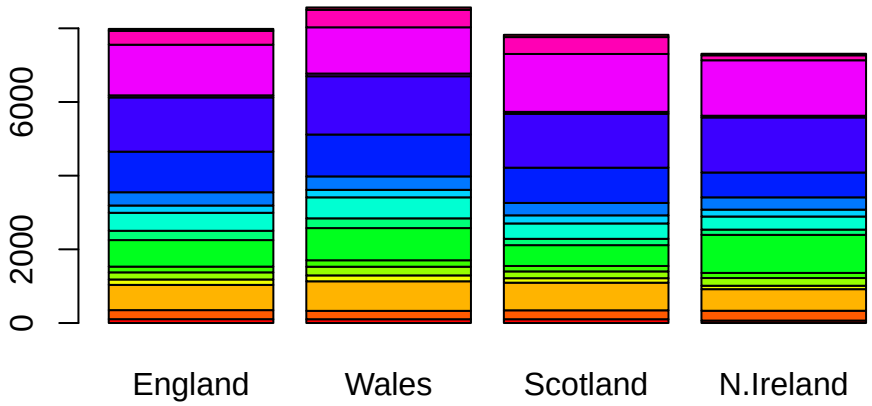
```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars  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

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



Q3. Changing what optional argument in the above `barplot()` function reuslts in the following plot?

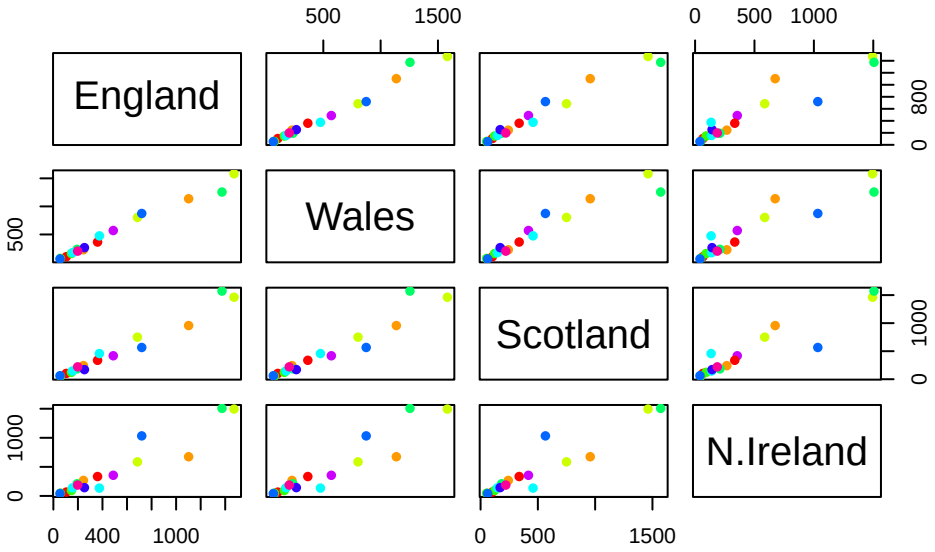
```
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```



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 give plot?

If the given point lies on the diagonal, it means that the points they are the same or very similar.

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



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

The `prcomp()` function expects the observations to be rows and the variables to be columns therefore we need to first transpose our `data.frame` matrix with the `t()` transpose function.

```
# Use the prcomp() PCA function
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
```

```
attributes(pca)
```

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

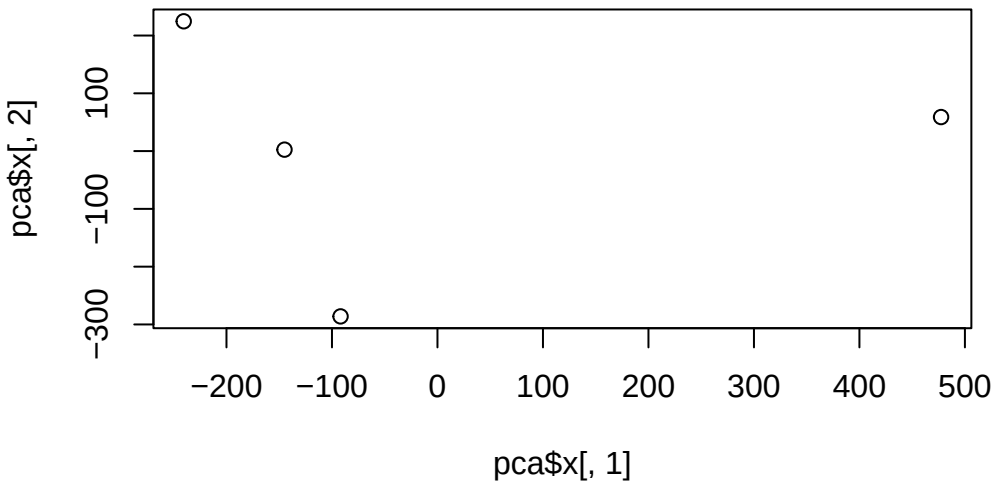
```
$class
[1] "prcomp"
```

```
pca$x
```

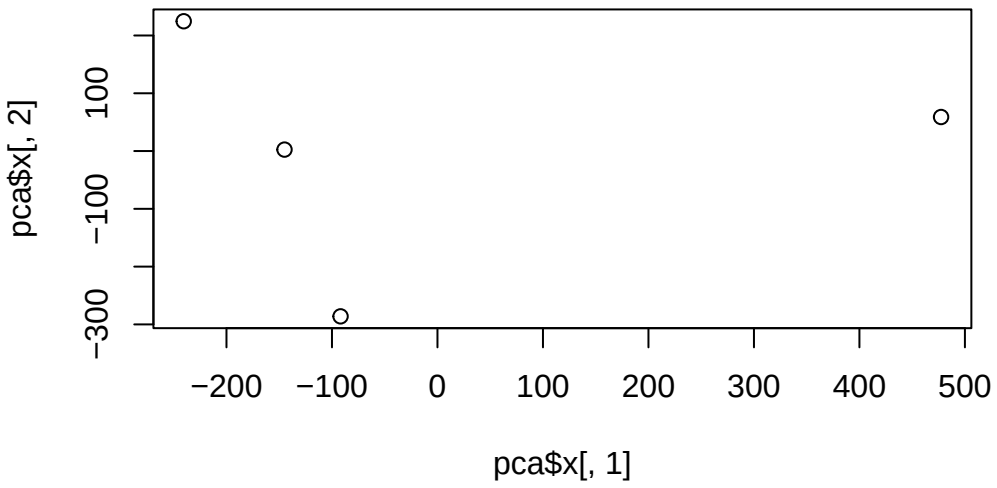
```
                PC1         PC2         PC3           PC4
England   -144.99315    2.532999 -105.768945  2.842865e-14
Wales     -240.52915  224.646925   56.475555  7.804382e-13
Scotland   -91.86934 -286.081786   44.415495 -9.614462e-13
N.Ireland  477.39164   58.901862    4.877895  1.448078e-13
```

Make my PC1 vs PC2 plot, "score plot", "PCA plot", etc.

```
plot(pca$x[,1], pca$x[,2])
```

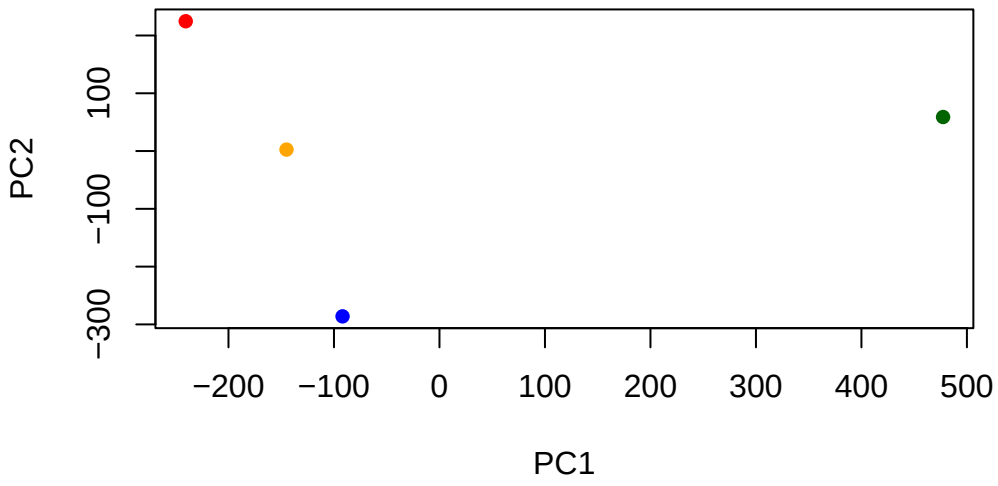



```
plot(pca$x[,1], pca$x[,2])
```

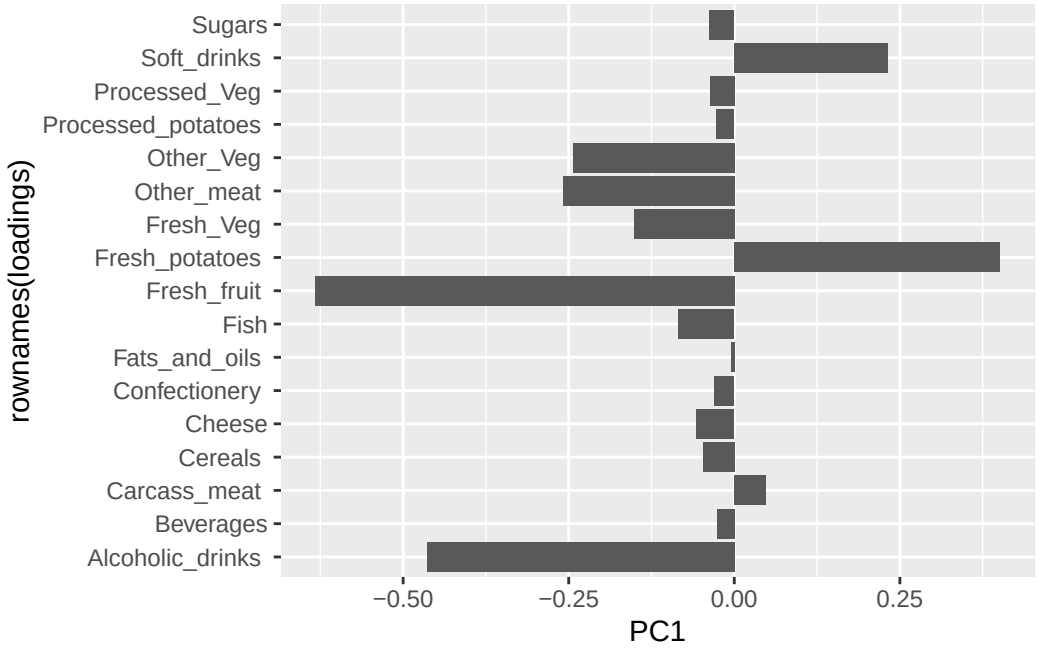


Color the points by country.

```
mycols = c("orange", "red", "blue", "darkgreen")
plot(pca$x[,1], pca$x[,2], col=mycols, pch=16, xlab = "PC1", ylab="PC2")
```




```
loadings <- as.data.frame(pca$rotation)
ggplot(loadings) +
  aes(PC1, rownames(loadings)) +
  geom_col()
```


rownames(loadings)
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
−0.50
−0.25
0.00
0.25
PC1