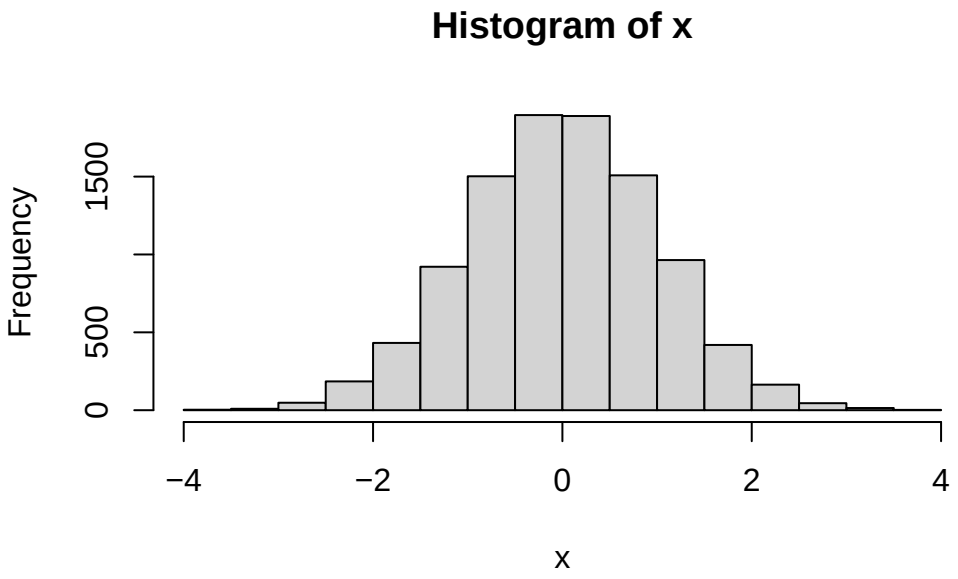
Class 7 Machine Learning 1

Forrest Wang

K-means clustering

First we will test how this method works in R with some made up data.

```
x <- rnorm(10000)
hist(x)
```

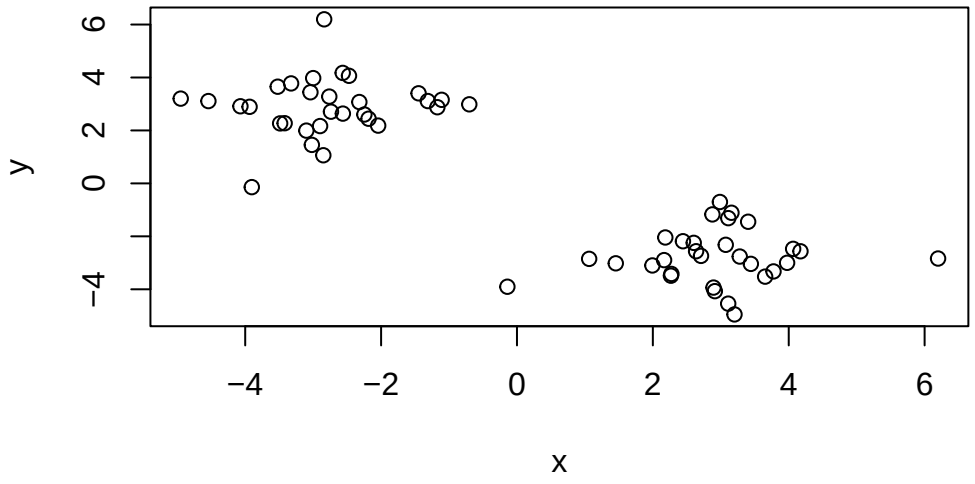


Let's make some numbers centered on -3

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

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

plot (x)
```



Now let's see how 'kmeans()' works with this data^

```r
km <- kmeans(x, centers = 2, nstart=20)
km
```

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

Cluster means:
          x         y
1  2.897744 -2.784478
2 -2.784478  2.897744

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] 64.4462 64.4462
 (between_SS / total_SS =  88.3 %)

Available components:

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

How many points are in each clurster?

```
km$size
```

```
[1] 30 30
```

What 'component' of your result object details - 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
```

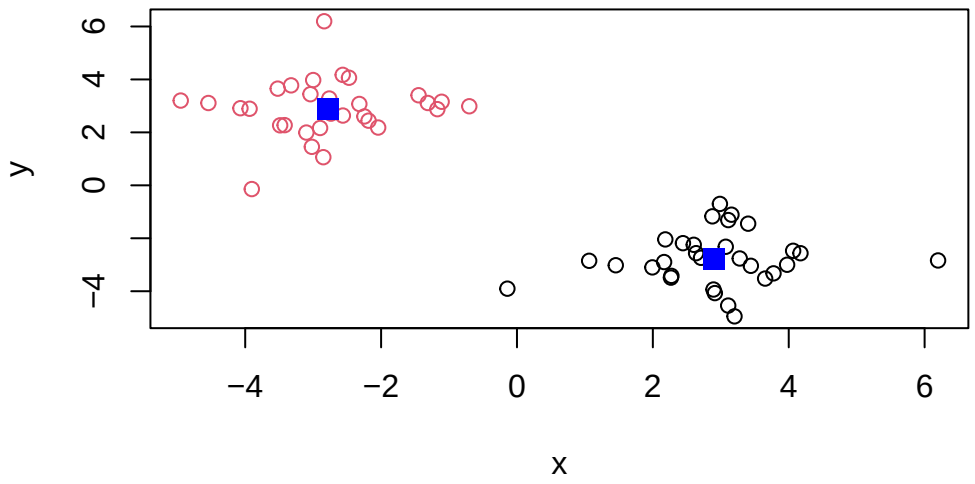
What 'component' of your result object details - cluster center

```
km$centers
```

```
          x         y
1  2.897744 -2.784478
2 -2.784478  2.897744
```

Plot x colored bt kmeans cluster assignment & add clusters as blue points

```
plot(x, col=km$cluster)
points(km$centers, col="blue", pch=15, cex=1.5)
```



#Hierarchical Clustering

The 'hclust()' function in R performs hierarchical clustering.

The 'hclus()' function requires an input distance matrix, which I can get from the 'dist()' function

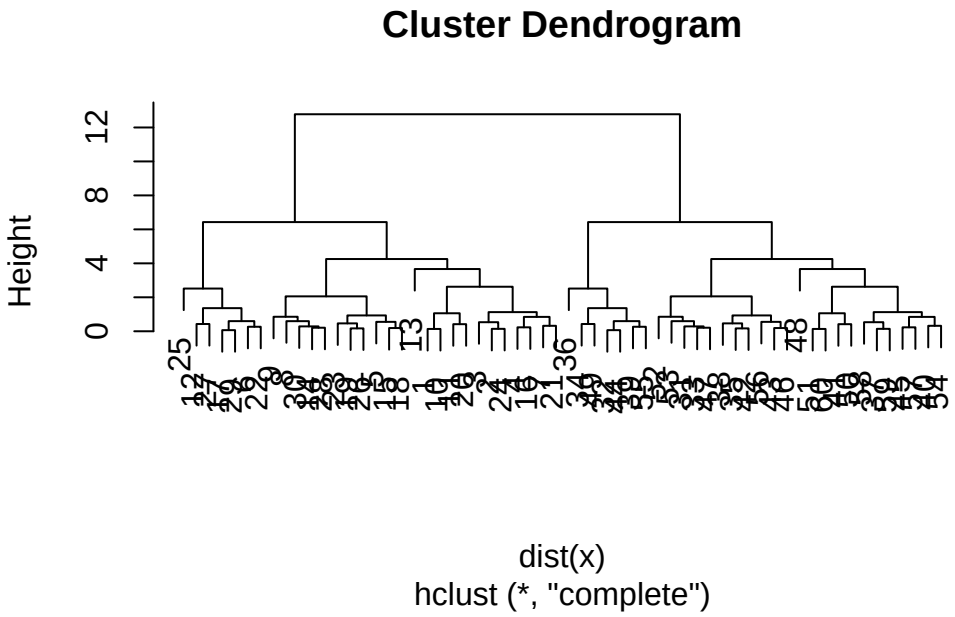
```
hc <- hclust( dist (x) )
hc
```

```
Call:
hclust(d = dist(x))

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

There is a plot() method for hclust objects:

```
plot(hc)
```



Now to get my cluster membership vector, I need to "cut" the tree to yield sreperate "branches" with the "leaves" on each branch being out clusters. TO do this we use the 'cutree()' function.

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

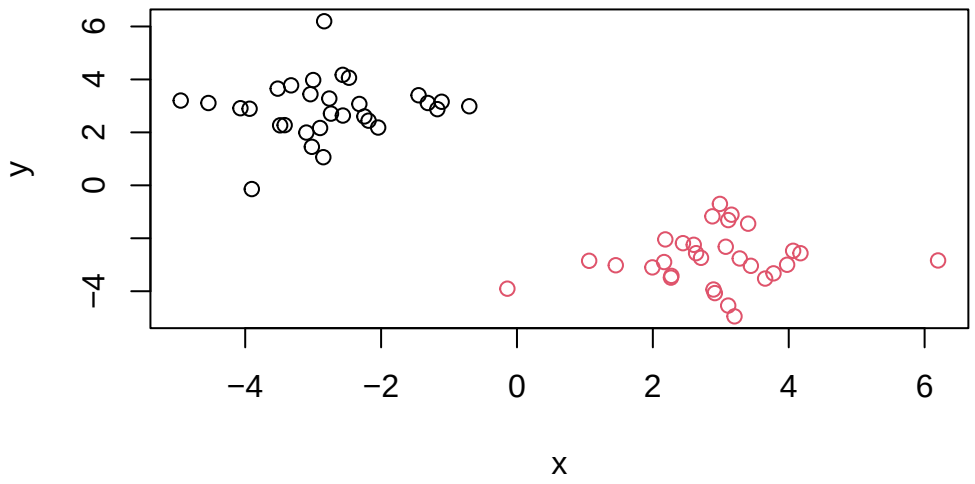
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

Use 'cutree()' with a k=2 (more useful)

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

A plot of our data colored by our hclust

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



#Principal Component Analysis (PCA) Data:

```
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
```

Q1:How many rows and columns are in your new data frame named x? What R functions could you use to answer this questions?

```
dim(x)
```

```
[1] 17  4
```

```
nrow(x)
```

```
[1] 17
```

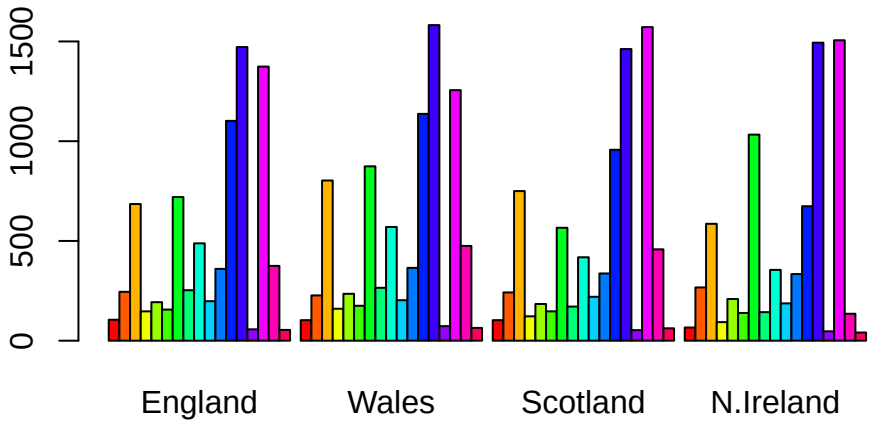
A1: There are 17 rows and 5 columns

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? A2: Insert inputs to edit rows inside the initial URL csv command to manipulate data directly as its pulled into R.

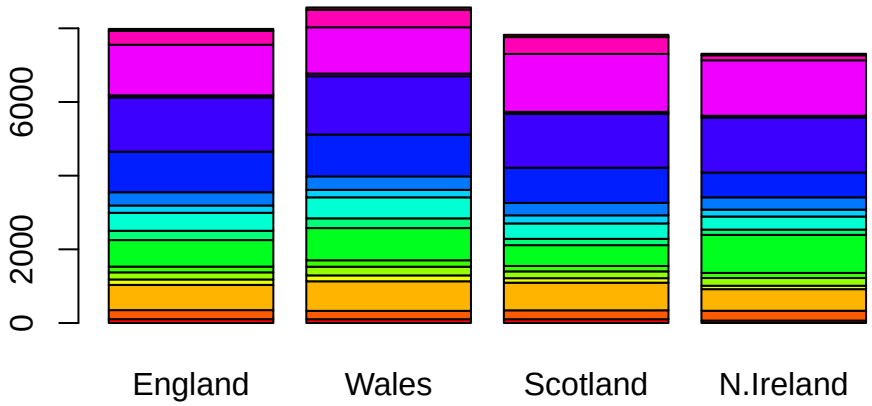
Spotting Diff/Trends:

Bar Plot

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




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

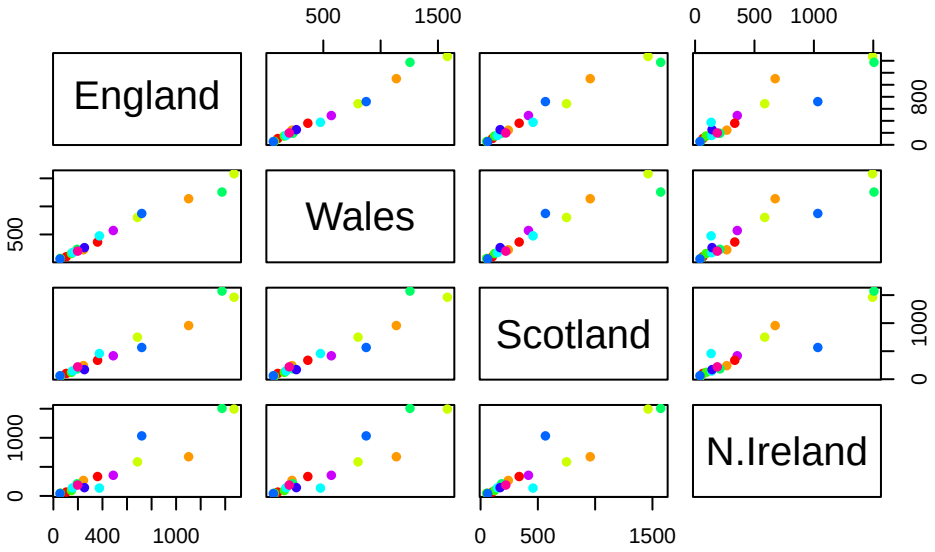


Q3: Changing what optional argument in the above barplot() function results in the following plot? A3: Changing beside from "beside=T" to "beside=FALSE" will change a bar plot to a cumulative plot that stacks all the data of each country which isn't that helpful.

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? A5:The pairwise plot allows 1 on 1 comparisons of each country. If a given point lies on a diagonal for a given plot, a diagonal spread of points represents an average trend ("as x increases, so does y"), this means that said point is expected

Pairs 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? A6: Northern Ireland points are much more right, especially the blue point, "maybe more alcohol" says Prof.B.

While this is kind of useful it takes work to dig into the details here to find out what is the difference in these countries

PCA to the rescue

Principal Component Analysis (PCA for short) can be a big help in these cases where we have lots of things that are being measured in a dataset (lots of dimensions).

The main PCA function in base R is called 'prcomp()'

The 'prcomp()' function wants as input the transpose of our food matrix/table/data.frame.

```
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
```

The above result shows PCA captures 67% of the total variance in the original data in one PC and 96.5% in two PCs.

```
attributes(x)
```

```
$names
[1] "England"   "Wales"     "Scotland"  "N.Ireland"

$class
[1] "data.frame"

$row.names
 [1] "Cheese"              "Carcass_meat "       "Other_meat "
 [4] "Fish"                "Fats_and_oils "      "Sugars"
 [7] "Fresh_potatoes "     "Fresh_Veg "          "Other_Veg "
[10] "Processed_potatoes " "Processed_Veg "      "Fresh_fruit "
[13] "Cereals "            "Beverages"           "Soft_drinks "
[16] "Alcoholic_drinks "   "Confectionery "
```

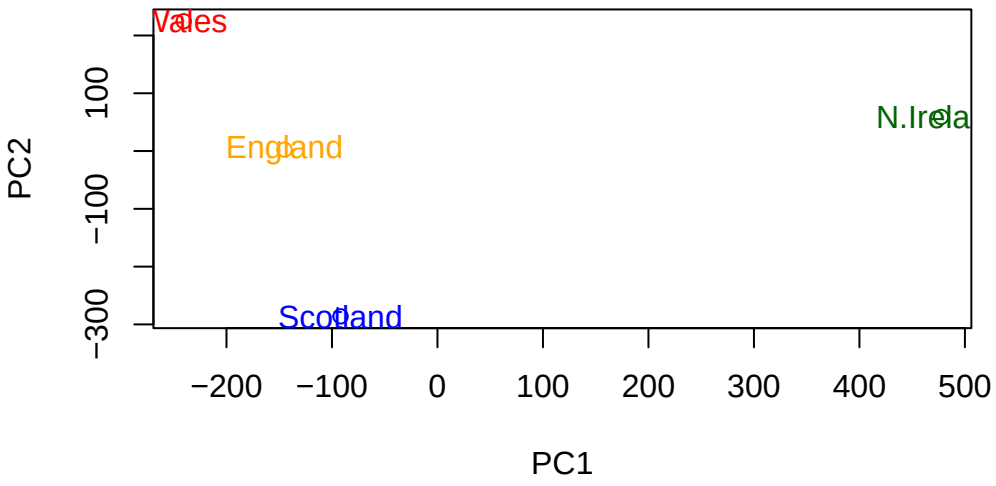
```
head(pca$x)
```

```
                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.

Let's plot our main results

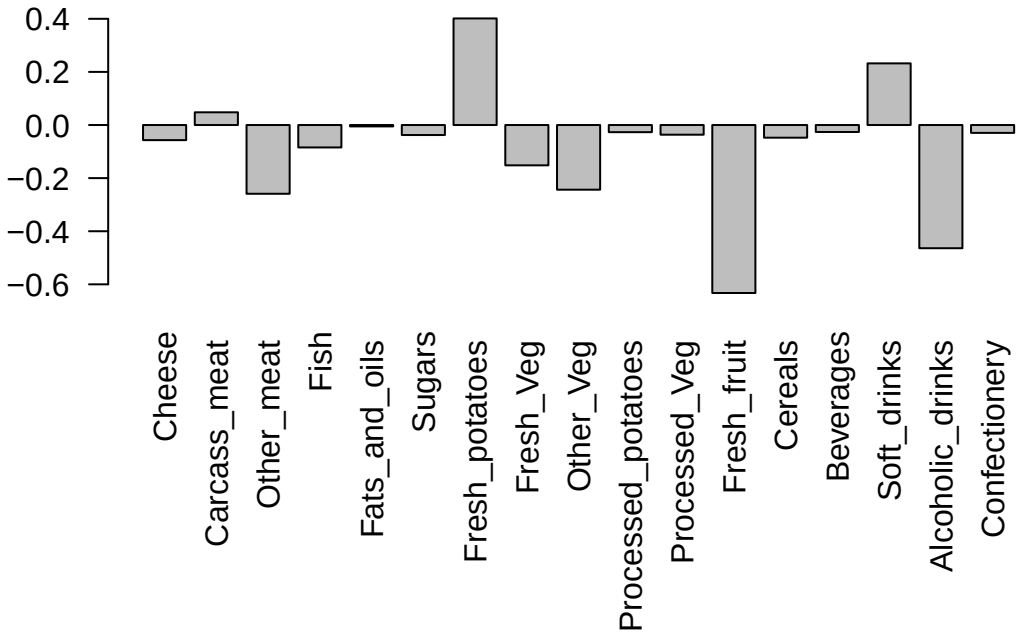
```
plot(pca$x[,1], pca$x[,2], xlab="PC1", ylab="PC2", col=c("orange", "red", "blue", "darkgre
text(pca$x[,1], pca$x[,2], colnames(x), col=c("orange", "red", "blue", "darkgreen"))
```



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. A8: Done^

DIGGING DEEPER (Variable Loading)

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



Q9:Generate a similar 'loadings plot' for PC2. What two food groups feature prominantely and what does PC2 maninly tell us about? A9: The two prominent food groups highlighted by the plot above are soft drinks and fresh potatoes

#2. PCA of RNA-seq

```
url2 <- "https://tinyurl.com/expression-CSV"
rna.data <- read.csv(url2, row.names=1)
head(rna.data)
```

```
       wt1 wt2  wt3  wt4 wt5 ko1 ko2 ko3 ko4 ko5
gene1  439 458  408  429 420  90  88  86  90  93
gene2  219 200  204  210 187 427 423 434 433 426
gene3 1006 989 1030 1017 973 252 237 238 226 210
gene4  783 792  829  856 760 849 856 835 885 894
gene5  181 249  204  244 225 277 305 272 270 279
gene6  460 502  491  491 493 612 594 577 618 638
```

Q10: How many genes and samples are in this data set? Q10: There are 6 genes and 10 samples in the data set.