Class 07 Machine Learning 1

Irene Hsieh (PID: A16197563)

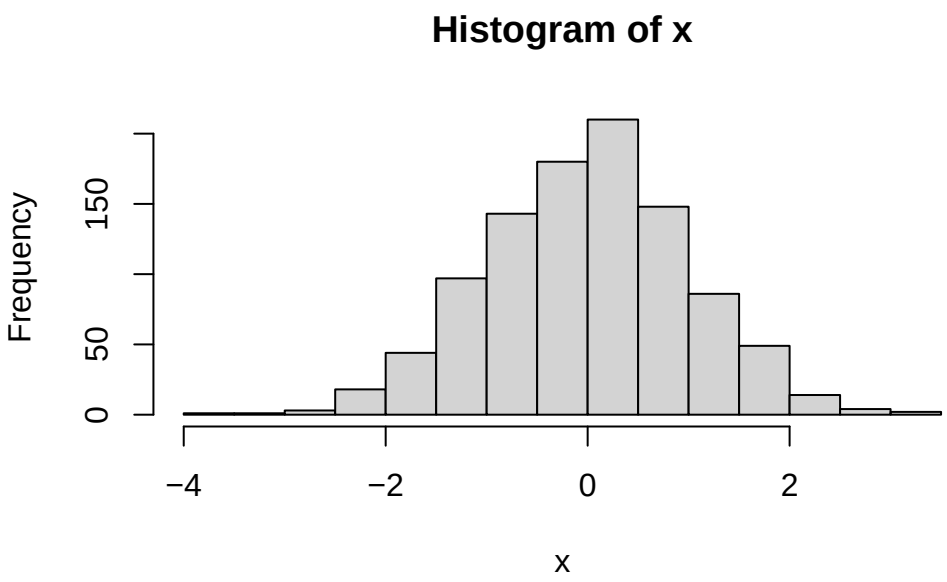
#Clustering Methods

The board goal here is to find groupings (clusters) in your input data

Kmeans

First, let's make up some data to cluster

```
x <- rnorm(1000)
hist(x)
```

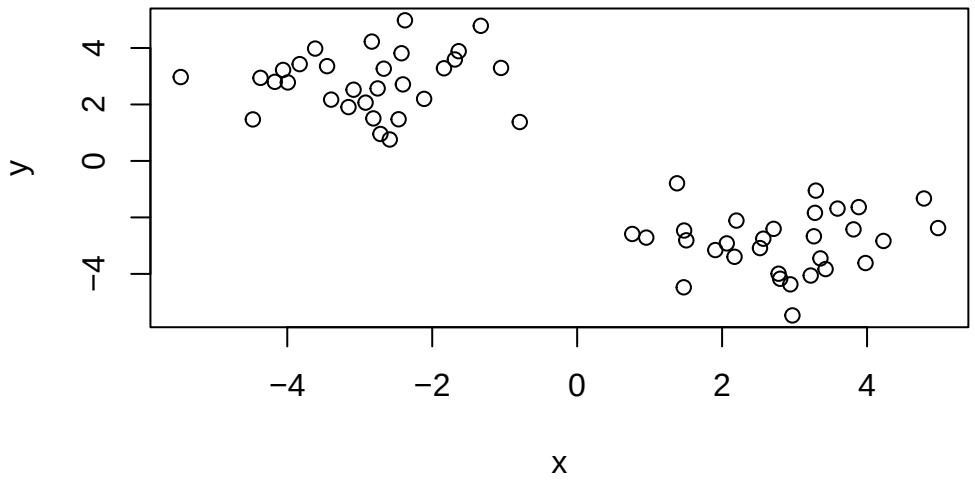


Make a vector of length 60 with 30 points centered at -3 and 30 points centered at +3

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

I will now make a wee x and y dataset with 2 groups of points

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




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

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

Cluster means:
          x         y
1  2.809610 -2.882202
2 -2.882202  2.809610

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] 67.31788 67.31788
 (between_SS / total_SS =  87.8 %)

Available components:

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

Q: From your result object 'k' how many points are in each cluster?

```
k$size
```

```
[1] 30 30
```

What "component" of your result object details the cluster membership?

```
k$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
```

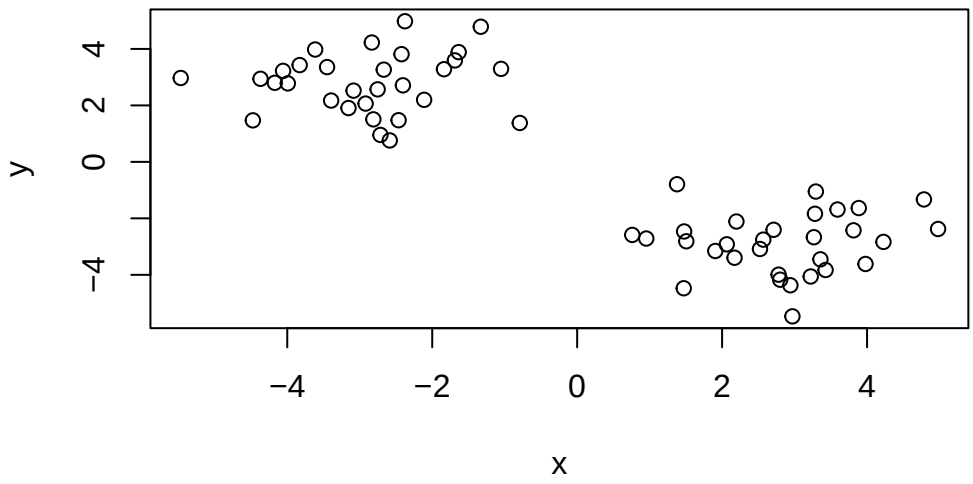
Cluster centers?

```
k$centers
```

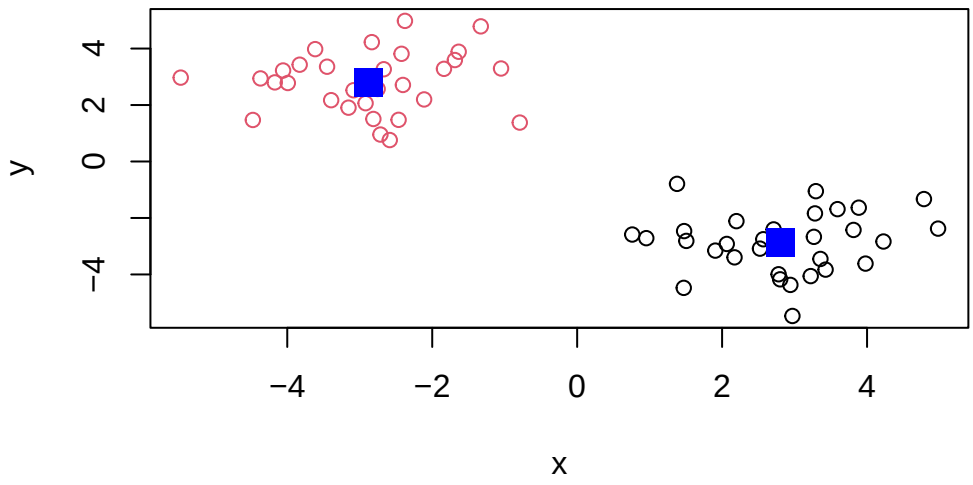
```
          x         y
1  2.809610 -2.882202
2 -2.882202  2.809610
```

Plot of our clustering result?

```
plot(x)
```

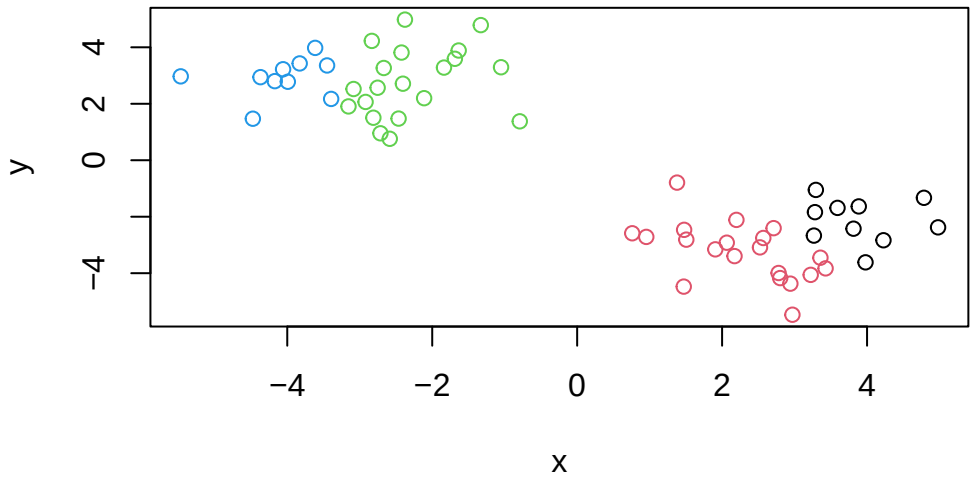



```
plot(x, col =k$cluster)
points(k$centers, col = "blue", pch=15, cex =2)
```



We can cluster into 4 grps

```
#kmeans
k4<- kmeans(x, centers = 4)
#plot results
plot(x, col = k4$cluster)
```



A big limitation of kmeans is that it does what you ask even if you ask for silly clusters

Hierarchical Clustering

The main base R function for Hierarchical Clustering is `hclust()`. Unlike `kmeans()` you can not just pass it your data as input. You first need calculate a distance matrix.

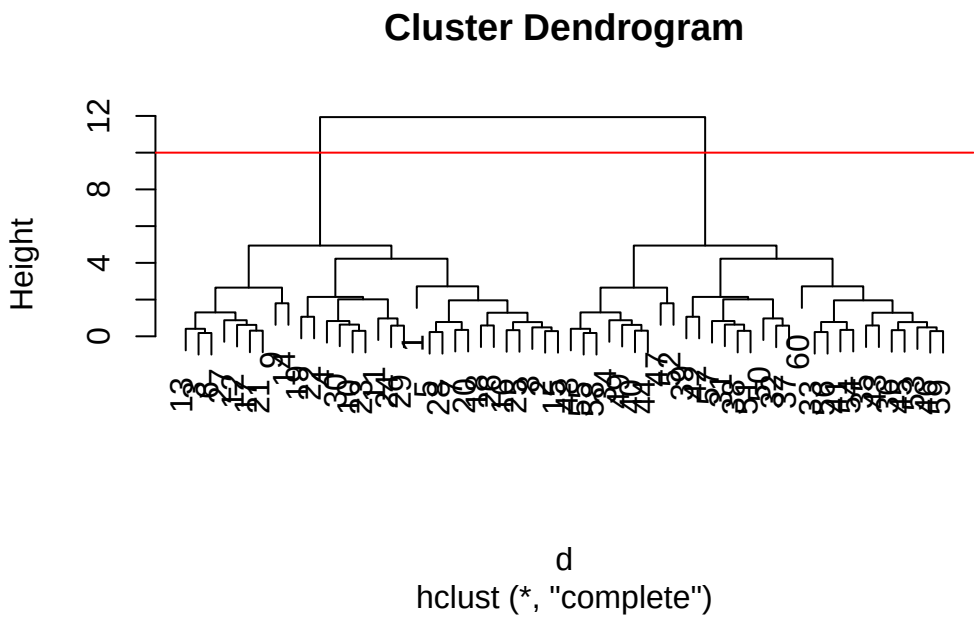
```
d <- dist(x)
hc<- hclust(d)
hc
```

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

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

Use `plot()` to view results

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



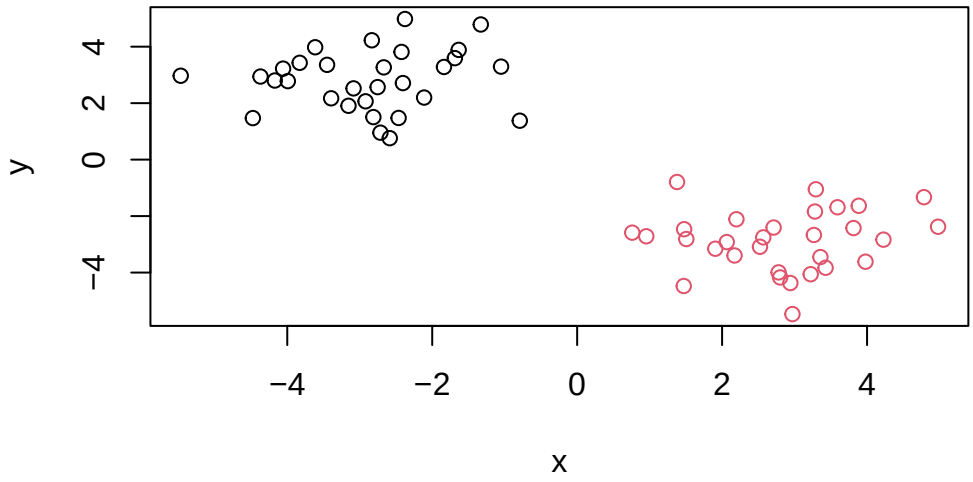
To make the "cut" and get our cluster membership vector we can use the `cutree()` function.

```
grps <- cutree(hc, h=10)
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
```

Make a plot of our data colored by hclust results

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



Principal Component Analysis (PCA)

Here we will do Principal Component Analysis (PCA) on some food data from the UK

```
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

```
#rownames(x) <- x[,1]
#x <- x[,-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?

```
dim(x)
```

```
[1] 17  4
```

17 rows, 4 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?

This is more prefered because this does not overide the function and does not eliminate the data in the end. This method is more robust since it does not replicate the reduction process

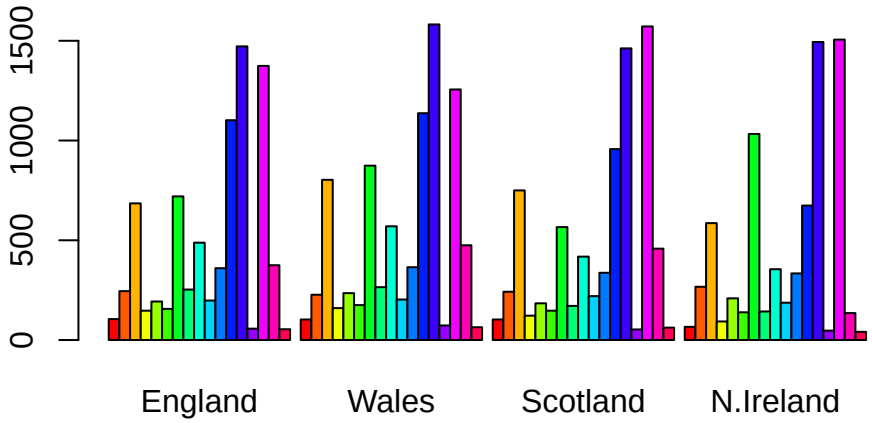
```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url, row.names =1)
head(x)
```

```
              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
```

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

Change the beside to False

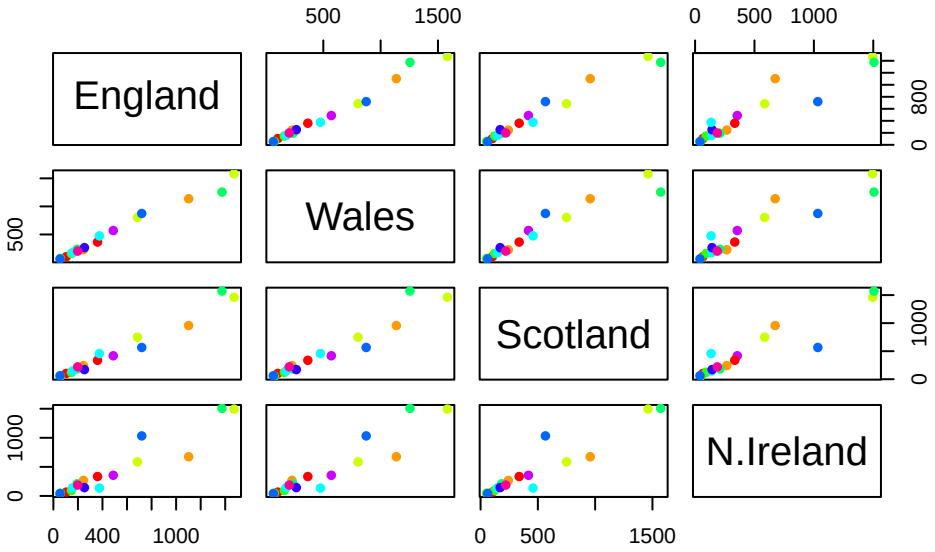
```
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```



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 given plot?

You can read the graph in half which the y and x indicates different countries that represent in the diagrams. For example, England will be compared on the left to other countries one by one when it goes down (the left column is all England). In terms of point, it demonstrated the comparasion between two countries of a diet

```
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 difference between Ireland and other countries is that the color dots are separated differently on the right of each graphs. Which demonstrated different trend of the diets than other countries. (dark blue and orange dataset is differently distributed)

PCA to the rescue

The main "base" R function for 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 2.921e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

Q. How much variance is captured in 2 PCs

96.5%

To make our main "PC score plot" or "PC1 vs PC2 plot" or "PC plot" or "ordination plot".

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

```
attributes(pca)
```

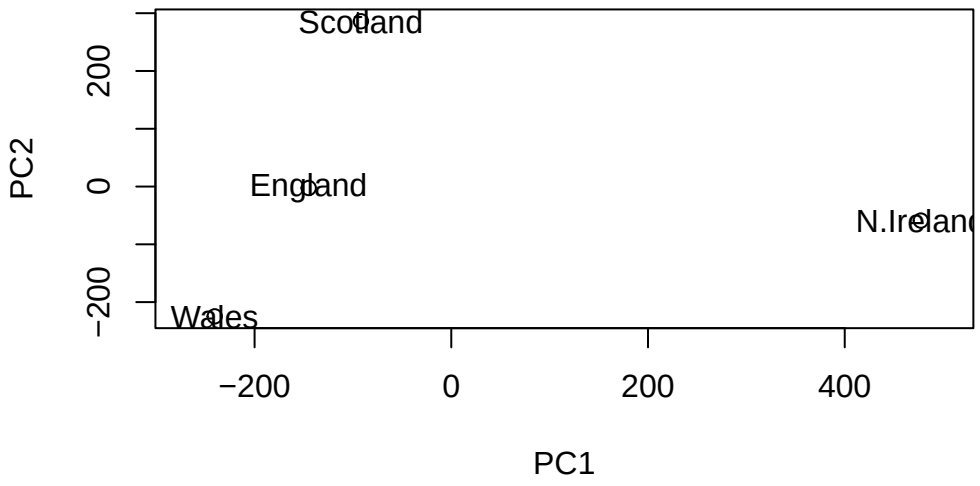
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

We are after the `pca$x` result component to make our main PCA plot.

```
pca$x
```

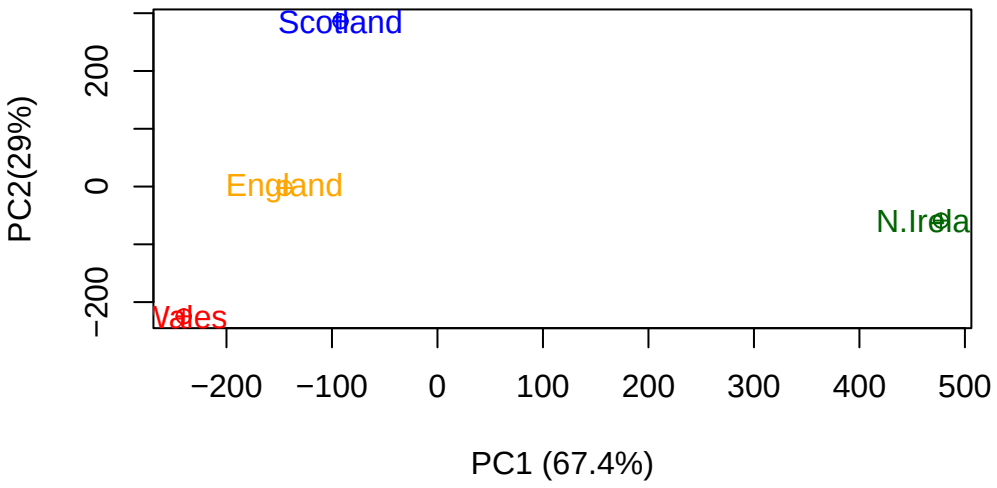
```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -9.152022e-15
Wales     -240.52915 -224.646925 -56.475555  5.560040e-13
Scotland   -91.86934  286.081786 -44.415495 -6.638419e-13
N.Ireland  477.39164  -58.901862  -4.877895  1.329771e-13
```

```
# Plot PC1 vs PC2
plot(pca$x[,1], pca$x[,2], 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.

```
mycols <- c("orange", "red", "blue", "darkgreen")
plot(pca$x[,1], pca$x[,2], col = mycols, pch = 10, xlab="PC1 (67.4%)", ylab="PC2(29%)")
text(pca$x[,1], pca$x[,2], colnames(x),col = mycols )
```



Another important result from PCA is how original variables (in this case the foods) contribute to the PCs.

This is contained in the `pca$rotation` object - folks often call this the "loadings" or "contributions" to the PCs.

```
pca$rotation
```

	PC1	PC2	PC3	PC4
Cheese	-0.056955380	0.016012850	0.02394295	-0.409382587
Carcass_meat	0.047927628	0.013915823	0.06367111	0.729481922
Other_meat	-0.258916658	-0.015331138	-0.55384854	0.331001134
Fish	-0.084414983	-0.050754947	0.03906481	0.022375878
Fats_and_oils	-0.005193623	-0.095388656	-0.12522257	0.034512161
Sugars	-0.037620983	-0.043021699	-0.03605745	0.024943337
Fresh_potatoes	0.401402060	-0.715017078	-0.20668248	0.021396007
Fresh_Veg	-0.151849942	-0.144900268	0.21382237	0.001606882
Other_Veg	-0.243593729	-0.225450923	-0.05332841	0.031153231
Processed_potatoes	-0.026886233	0.042850761	-0.07364902	-0.017379680
Processed_Veg	-0.036488269	-0.045451802	0.05289191	0.021250980
Fresh_fruit	-0.632640898	-0.177740743	0.40012865	0.227657348
Cereals	-0.047702858	-0.212599678	-0.35884921	0.100043319

```
Beverages            -0.026187756 -0.030560542 -0.04135860 -0.018382072
Soft_drinks           0.232244140  0.555124311 -0.16942648  0.222319484
Alcoholic_drinks     -0.463968168  0.113536523 -0.49858320 -0.273126013
Confectionery        -0.029650201  0.005949921 -0.05232164  0.001890737
```

We can make a plot along PC1

```
library(ggplot2)

contrib <- as.data.frame(pca$rotation)

ggplot(contrib)+
  aes(PC1, rownames(contrib))+
  geom_col()
```