lab07: Hands on with Principal Component Analysis (PCA)

Ebony Michelle Argaez (PID: A59026556)

Clustering

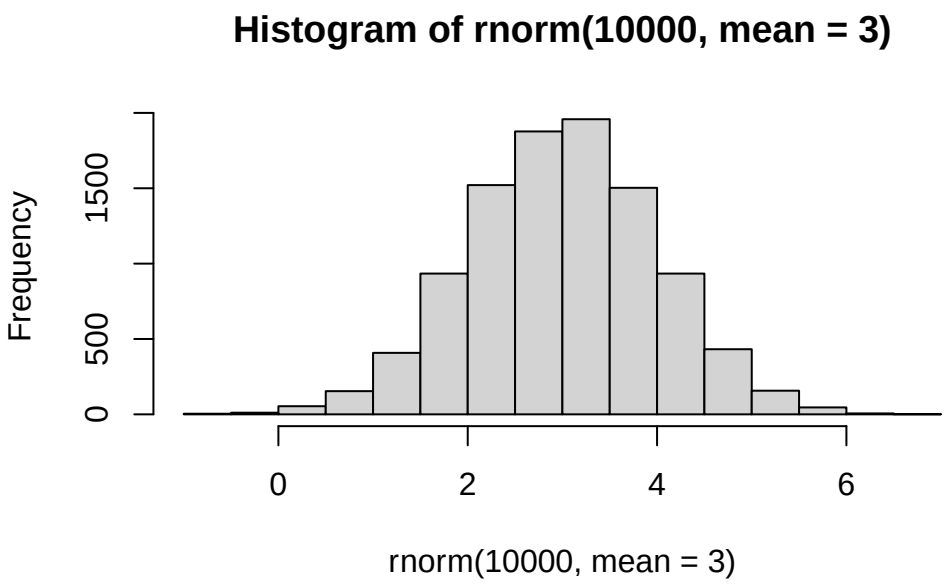
k-means clustering (most prevalent clustering method)

Making some data

```
#gives you back random numbers from normal distribution
rnorm(10)
```

```
 [1]  0.61288885  1.22768991  0.02108464  0.96713745 -1.49066103 -0.62622933
 [7] -0.77791910  2.37957922  0.18226112  0.22152086
```

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




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

```
 [1] 2.3992117 3.5197008 3.4209151 2.7656477 4.4005923 2.0616343 0.6399232
 [8] 1.7934666 4.0755605 2.8756041 2.1786729 1.1300094 2.0976403 3.4262205
[15] 3.7015588 1.0126038 2.7316575 2.4954209 3.1561099 2.3073350 2.4726484
[22] 2.5945809 4.9167594 1.9802965 2.5757080 4.7735508 4.1420288 3.9091999
[29] 3.7033618 2.8526297
```

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

```
 [1]  2.7107545  2.8535362  4.6026258  4.6857699  3.4420935  2.9624156
 [7]  4.4884551  3.1389403  2.5255344  3.2470068  3.5430109  0.4924854
[13]  4.5105906  0.8223359  2.6711999  1.8491466  0.6913675  4.3434423
[19]  2.2138942  3.4056185  2.3824626  4.0417533  3.4027024  5.0161647
[25]  2.7631460  2.5772230  2.7205936  3.5454887  2.6173444  1.3444401
[31] -1.4127059 -4.0640276 -3.5789770 -1.7722844 -3.9277814 -3.9884789
[37] -2.8915479 -2.4425860 -2.7009613 -3.4265713 -2.8844977 -4.6519962
[43] -1.2847049 -3.2275874 -3.4062184 -1.5825985 -4.2797222 -3.0104441
```

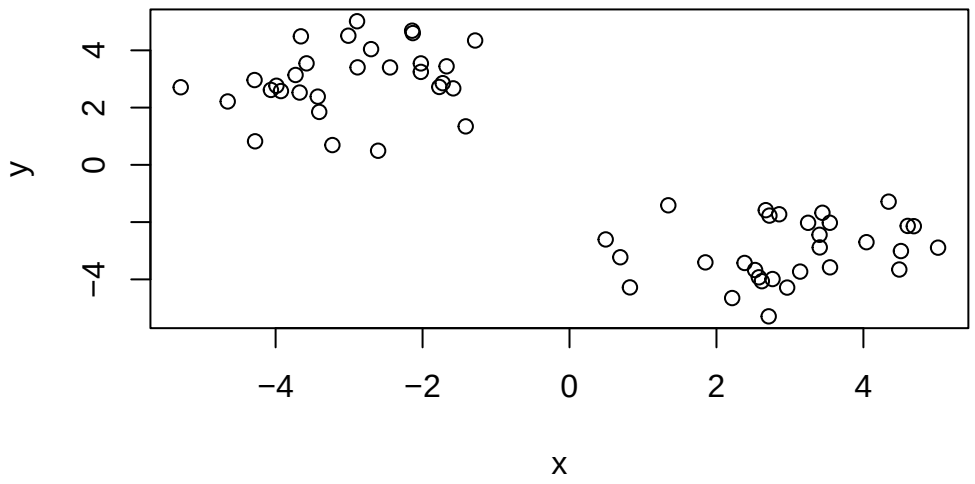
```
[49] -2.6050761 -2.0230707 -2.0234442 -3.6730460 -3.7285095 -3.6559281
[55] -4.2865206 -1.6733247 -2.1418524 -2.1329285 -1.7260047 -5.2914976
```

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

```
              x          y
 [1,]  2.7107545 -5.2914976
 [2,]  2.8535362 -1.7260047
 [3,]  4.6026258 -2.1329285
 [4,]  4.6857699 -2.1418524
 [5,]  3.4420935 -1.6733247
 [6,]  2.9624156 -4.2865206
 [7,]  4.4884551 -3.6559281
 [8,]  3.1389403 -3.7285095
 [9,]  2.5255344 -3.6730460
[10,]  3.2470068 -2.0234442
[11,]  3.5430109 -2.0230707
[12,]  0.4924854 -2.6050761
[13,]  4.5105906 -3.0104441
[14,]  0.8223359 -4.2797222
[15,]  2.6711999 -1.5825985
[16,]  1.8491466 -3.4062184
[17,]  0.6913675 -3.2275874
[18,]  4.3434423 -1.2847049
[19,]  2.2138942 -4.6519962
[20,]  3.4056185 -2.8844977
[21,]  2.3824626 -3.4265713
[22,]  4.0417533 -2.7009613
[23,]  3.4027024 -2.4425860
[24,]  5.0161647 -2.8915479
[25,]  2.7631460 -3.9884789
[26,]  2.5772230 -3.9277814
[27,]  2.7205936 -1.7722844
[28,]  3.5454887 -3.5789770
[29,]  2.6173444 -4.0640276
[30,]  1.3444401 -1.4127059
[31,] -1.4127059  1.3444401
[32,] -4.0640276  2.6173444
[33,] -3.5789770  3.5454887
[34,] -1.7722844  2.7205936
[35,] -3.9277814  2.5772230
```

```
[36,] -3.9884789  2.7631460
[37,] -2.8915479  5.0161647
[38,] -2.4425860  3.4027024
[39,] -2.7009613  4.0417533
[40,] -3.4265713  2.3824626
[41,] -2.8844977  3.4056185
[42,] -4.6519962  2.2138942
[43,] -1.2847049  4.3434423
[44,] -3.2275874  0.6913675
[45,] -3.4062184  1.8491466
[46,] -1.5825985  2.6711999
[47,] -4.2797222  0.8223359
[48,] -3.0104441  4.5105906
[49,] -2.6050761  0.4924854
[50,] -2.0230707  3.5430109
[51,] -2.0234442  3.2470068
[52,] -3.6730460  2.5255344
[53,] -3.7285095  3.1389403
[54,] -3.6559281  4.4884551
[55,] -4.2865206  2.9624156
[56,] -1.6733247  3.4420935
[57,] -2.1418524  4.6857699
[58,] -2.1329285  4.6026258
[59,] -1.7260047  2.8535362
[60,] -5.2914976  2.7107545
```

```
plot(x)
```



The main function in R for k-means clustering is called 'kmeans()'.

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

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

Cluster means:
          x         y
1  2.987051 -2.983163
2 -2.983163  2.987051

Clustering vector:
 [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

Within cluster sum of squares by cluster:
[1] 73.09696 73.09696
 (between_SS / total_SS =  88.0 %)

Available components:
```

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

Q1. How many points are in each cluster?

```
k$size
```

```
[1] 30 30
```

30 in each cluster

Q2. The clustering result i.e. membership vector?

```
k$cluster
```

```
 [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
```

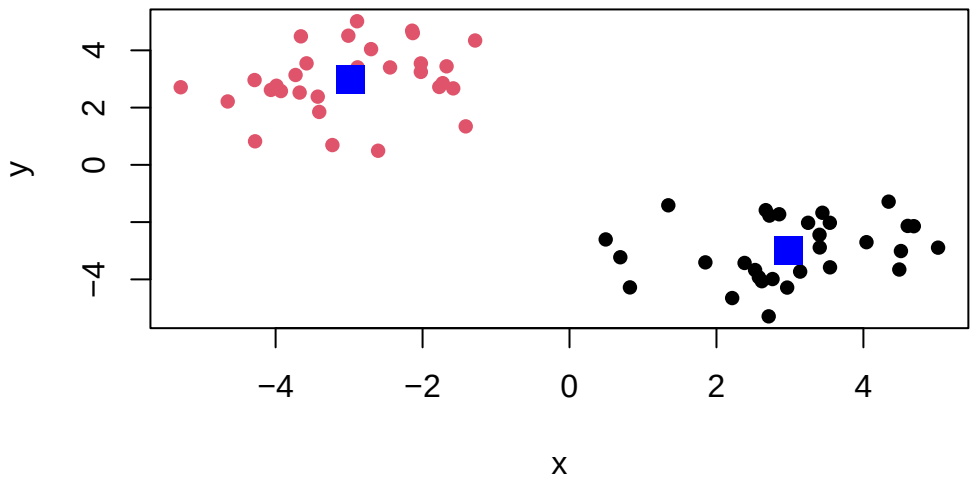
Q3. Cluster centers

```
k$centers
```

```
          x         y
1  2.987051 -2.983163
2 -2.983163  2.987051
```

Q4. Make a plot of our data colored by clustering results with optionally the cluster centers shown.

```
plot(x, col=k$cluster, pch=16)
points(k$centers, col="blue", pch=15, cex=2)
```



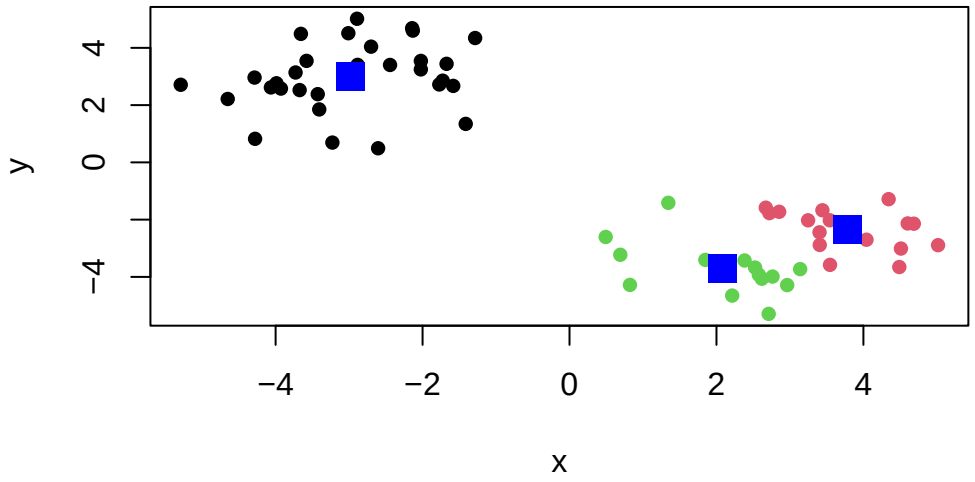
Q5. Run kmeans again but cluster into 3 groups and plot the results like we did above

```
k3 <- kmeans(x, centers=3, nstart=20)
k3
```

```
K-means clustering with 3 clusters of sizes 30, 16, 14

Cluster means:
          x         y
1 -2.983163  2.987051
2  3.782503 -2.345322
3  2.077964 -3.712124

Clustering vector:
 [1] 3 2 2 2 2 3 2 3 3 2 2 3 2 3 2 3 3 2 3 2 3 2 2 2 3 3 2 2 3 3 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] 73.09696 16.01996 21.43410
 (between_SS / total_SS =  90.9 %)

Available components:
```

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

```
plot(x, col=k3$cluster, pch=16)
points(k3$centers, col="blue", pch=15, cex=2)
```



hierarhical clustering

It reveals the structure in your data rather than imposing a structure as k-means will.

the main function is a "base" R is called 'hclust()'.

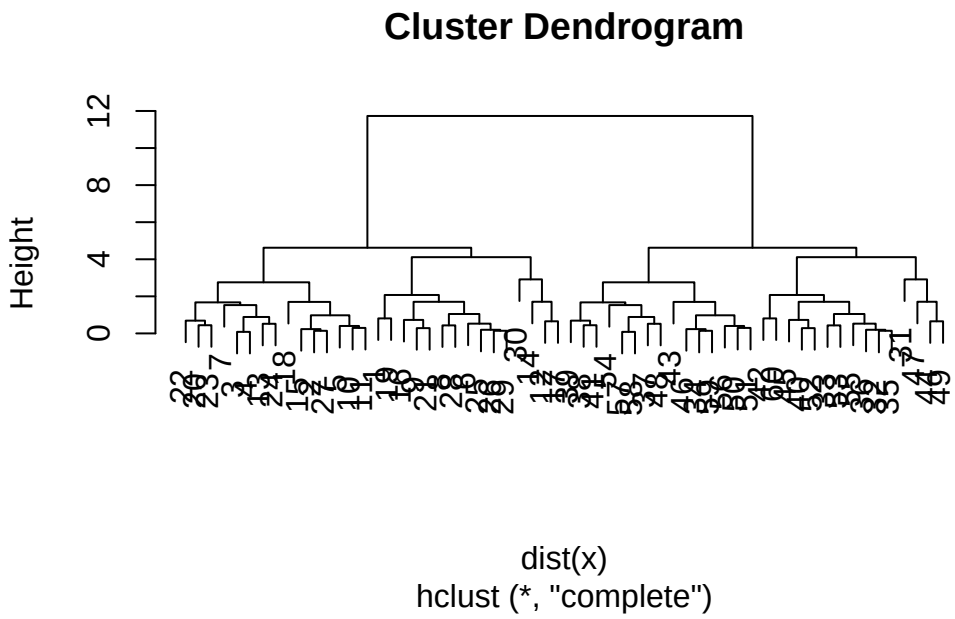
It requires a distance matrix as input, not the raw data itself.

```
#dist(x)
```

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

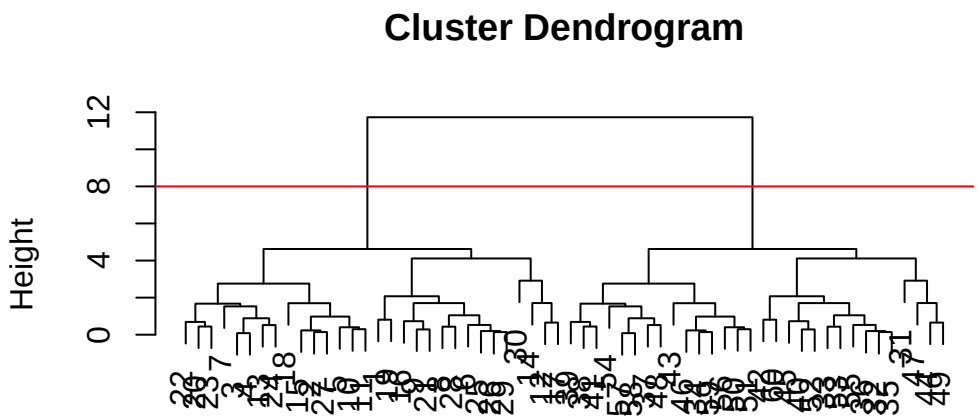
```
Call:
hclust(d = dist(x))

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

```r
#plotting dendrogram
plot(hc)
```



The function to get our clusters/groups, from a hclust objext is called 'cutree()'

```r
plot(hc)
#draws line, your barrier
abline(h=8, col="red")
```

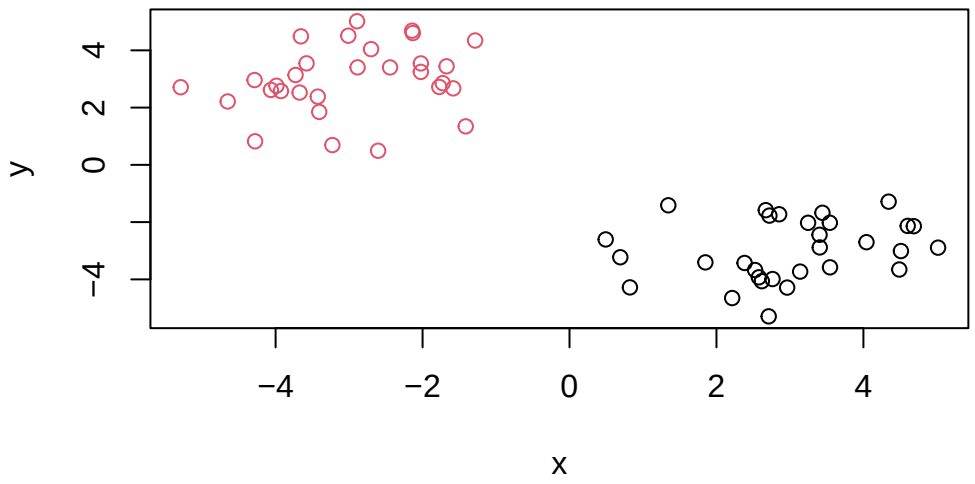



```
#will give you the # of cluster
grps <- cutree(hc, k=2)
grps
```

```
 [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
```

Q. Plot our hclust results in terms of our data colored by cluster membership

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



##Principal Component Analysis (PCA)

Principals components are new low dimensional axis (or surfaces) closest to the observations.

Data has maximum variance along PC1 (then PC2 etc).

Data Import

```
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?

```
#determining rows and columns; use any of the below
dim(x)
```

```
[1] 17  5
```

```
nrow(x)
```

```
[1] 17
```

```
ncol(x)
```

```
[1] 5
```

17 rows and 5 columns!

checking your data

```
#preview first 6 rows
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
```

```
#reorganizing dataframe
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
```

```
#checking rows and columns of dataframe
dim(x)
```

```
[1] 17  4
```

```
#Alt. approach to setting correct row names from 'read.csv()'
x <- read.csv(url, row.names=1)
head(x)
```

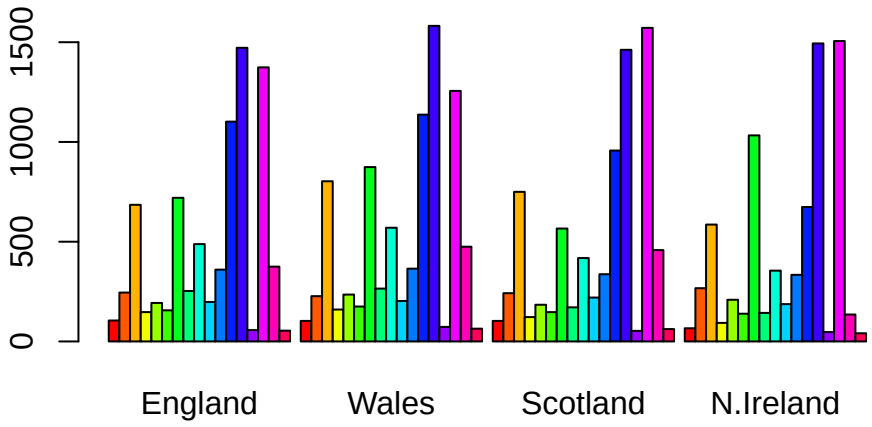
```
              England Wales Scotland N.Ireland
Cheese            105   103      103        66
Carcass_meat      245   227      242       267
Other_meat        685   803      750       586
Fish              147   160      122        93
Fats_and_oils     193   235      184       209
Sugars        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

> 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?

I like the second approach of using 'row.names =' because you're using the original data and less likely to delete data.

spotting major differences and trends

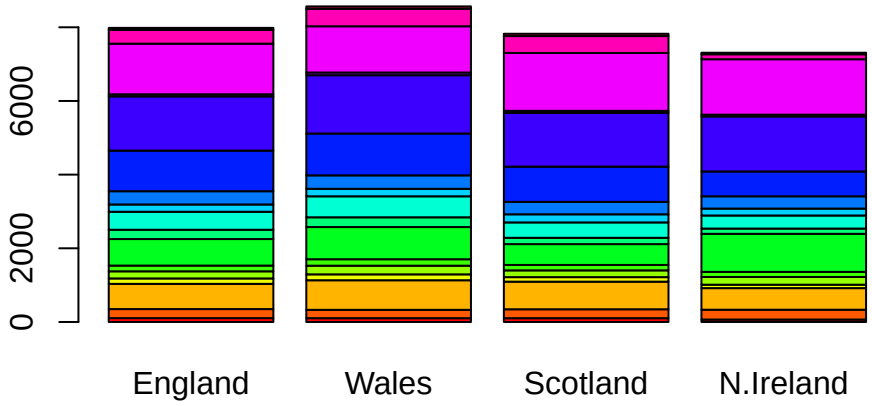
```
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```



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

Changing orientation of barplot

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

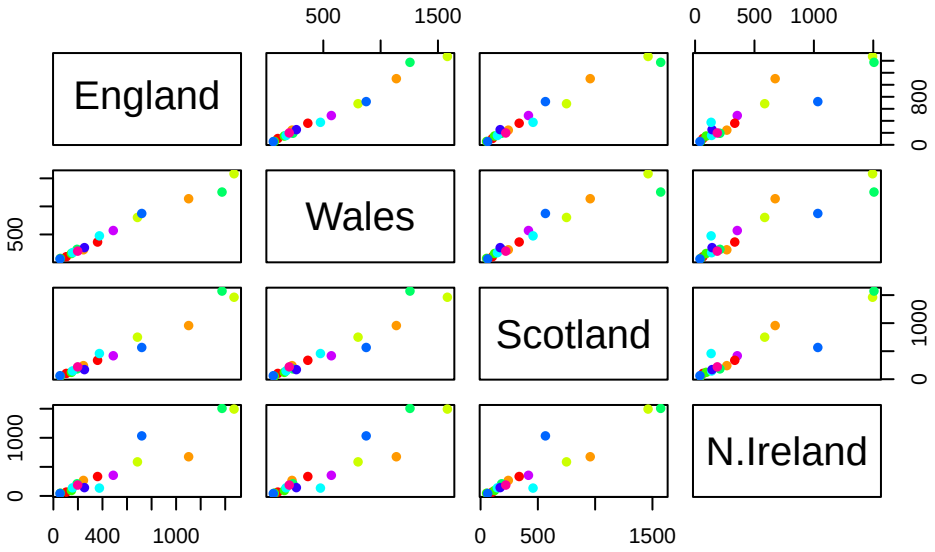


Q5: Generating all pairwise plots may help somewhat. Can you make sense of the following code and resulting figure? What does it mean if a given point lies on the diagonal for a given plot?

If the points are more linear, it means there's a correlation and the more similar two countries are the more diagonal the line is. If a point is outside the diagonal line there is a difference in what is being measured between the two countries.

Generating pairwise plots

```
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?

N. Ireland and the other UK countries are different because the correlation for N. Ireland is not as linear as the other countries.

##PCA to the rescue

'prcomp()' 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) )
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
```

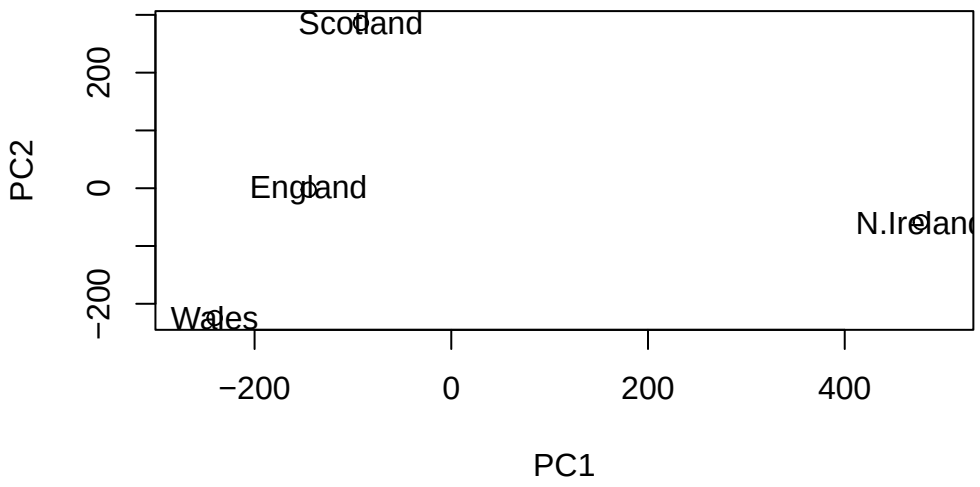
```
summary(pca)
```

```
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
```

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

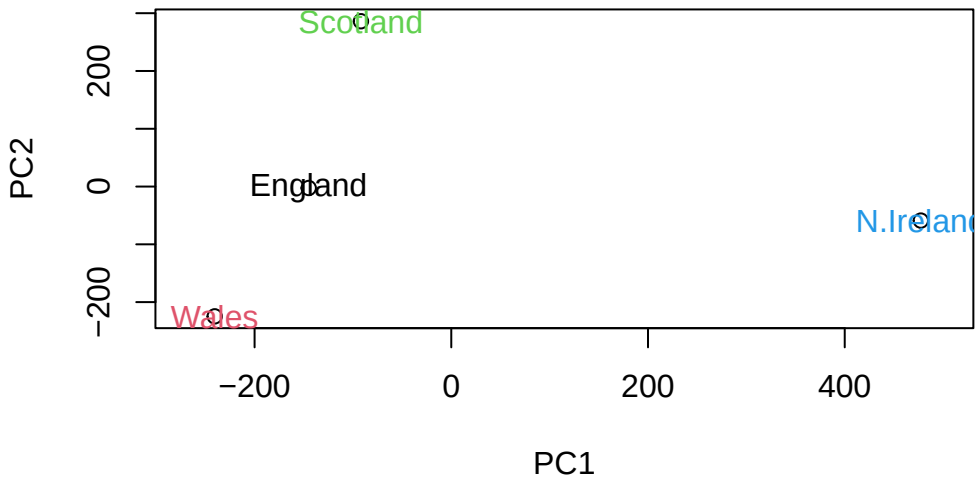
Generating plot of PC1 vs PC2

```
# 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.

```
# 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), col=c(1,2,3,4))
```



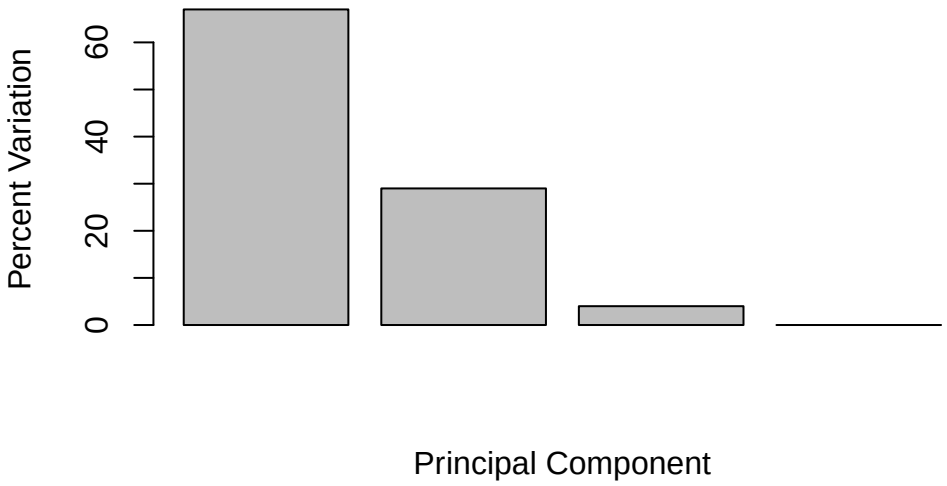

```
#standard deviation to calculate variation in the original data each PC accounts for
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

```
[1] 67 29  4  0
```

```
z <- summary(pca)
z$importance
```

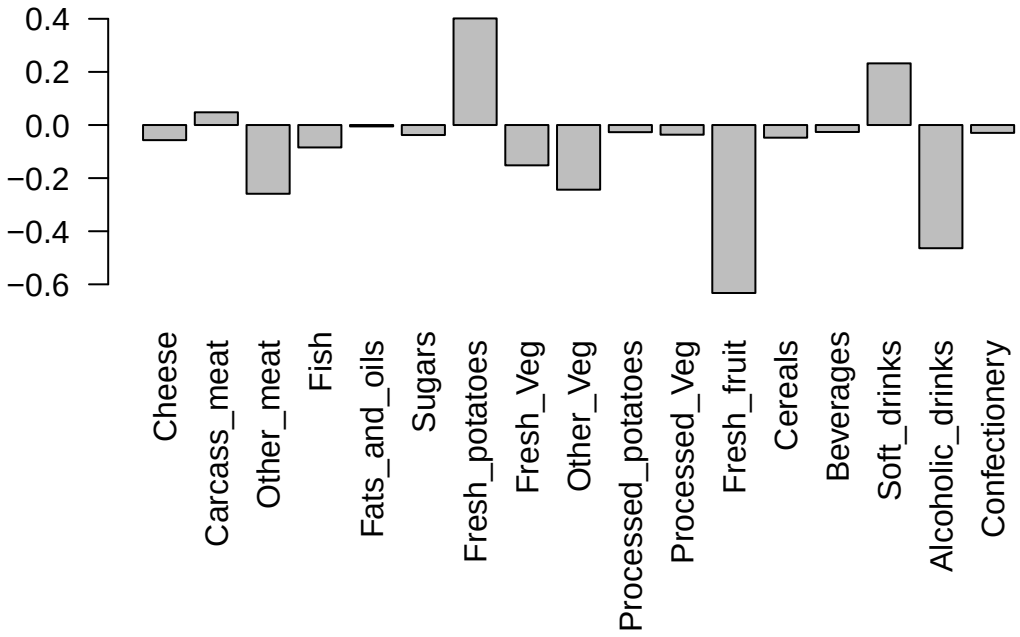
```
                             PC1       PC2      PC3           PC4
Standard deviation     324.15019 212.74780 73.87622 2.921348e-14
Proportion of Variance   0.67444   0.29052  0.03503 0.000000e+00
Cumulative Proportion    0.67444   0.96497  1.00000 1.000000e+00
```

```
barplot(v, xlab="Principal Component", ylab="Percent Variation")
```



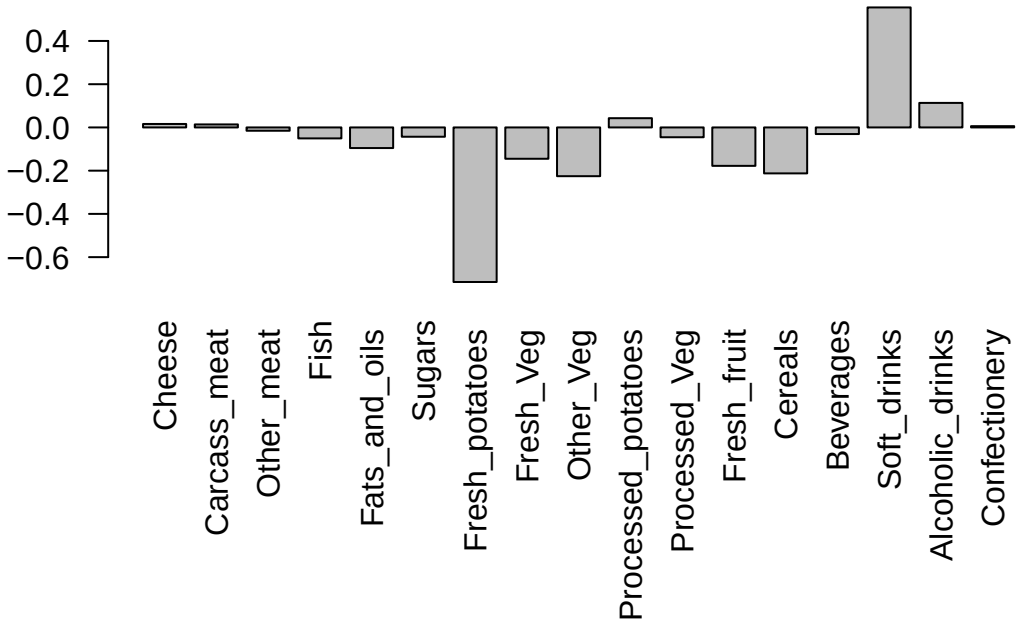
Digging Deeper (variable loadings)

```
## Lets focus on PC1 as it accounts for > 90% of variance
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?

```
## making PC2
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,2], las=2 )
```



Fresh potatoes and Soft drinks contributes to the variance in PC2 between in each country. Scotland drinks more soft drinks (the most famous soda in scotland is iron brew, story by Barry) and wales eat more fresh potatoes.