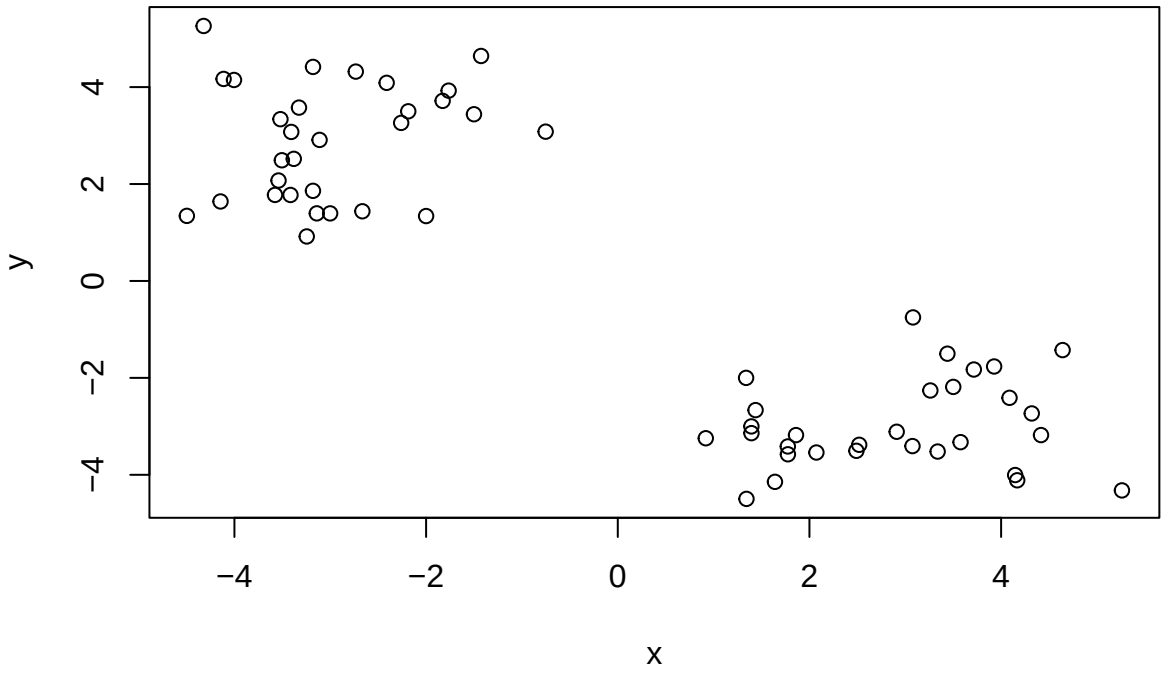
Machine Learning 1

Audrey Ting Zhu (A16898668)

2024-10-27

#First up kmeans() Demo od using kmeans()function in base R. First make up data.

```
tmp<-c(rnorm(30,-3),rnorm(30,3))
x<-cbind(x=tmp,y=rev(tmp))
plot(x)
```



Now we have some made up data in x. Let's see how kmeans works with this data.

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

```
## K-means clustering with 2 clusters of sizes 30, 30
##
## Cluster means:
##           x         y
## 1  2.894131 -2.971369
```

```
## 2 -2.971369  2.894131
##
## 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 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] 67.202 67.202
##  (between_SS / total_SS =  88.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

How many points in each cluster?

```
k$size
```

```
## [1] 30 30
```

How do we get to 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 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
```

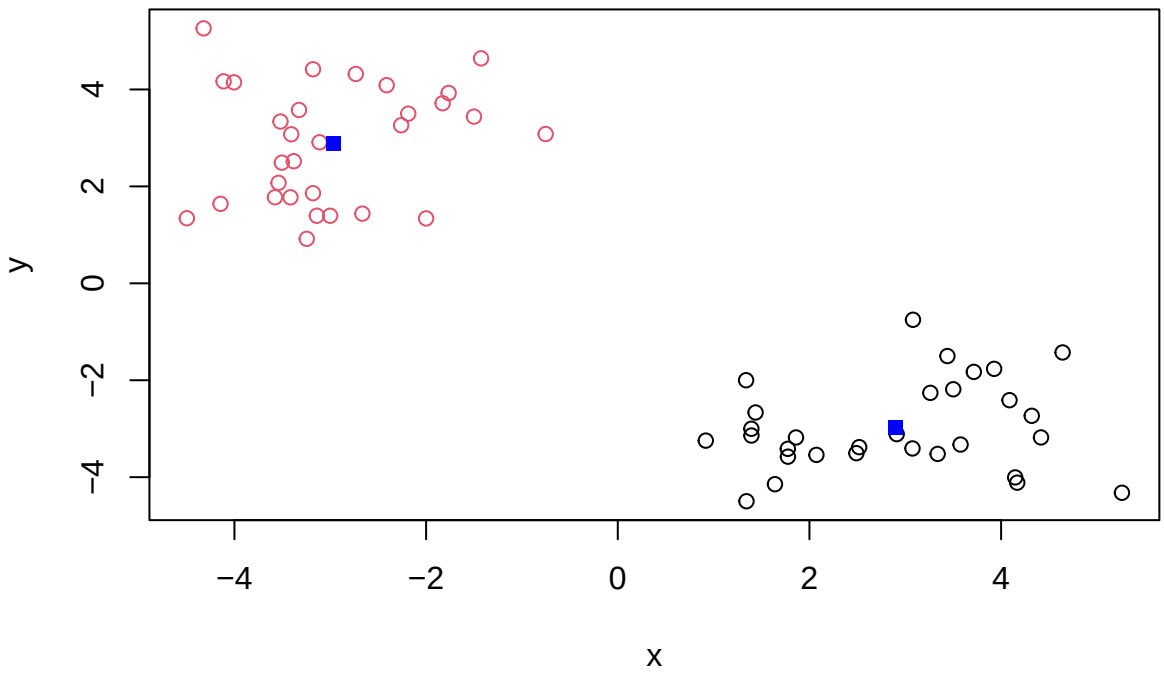
What about cluster centers?

```
k$centers
```

```
##           x         y
## 1  2.894131 -2.971369
## 2 -2.971369  2.894131
```

Now we have main results, we use them to plot our data.

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



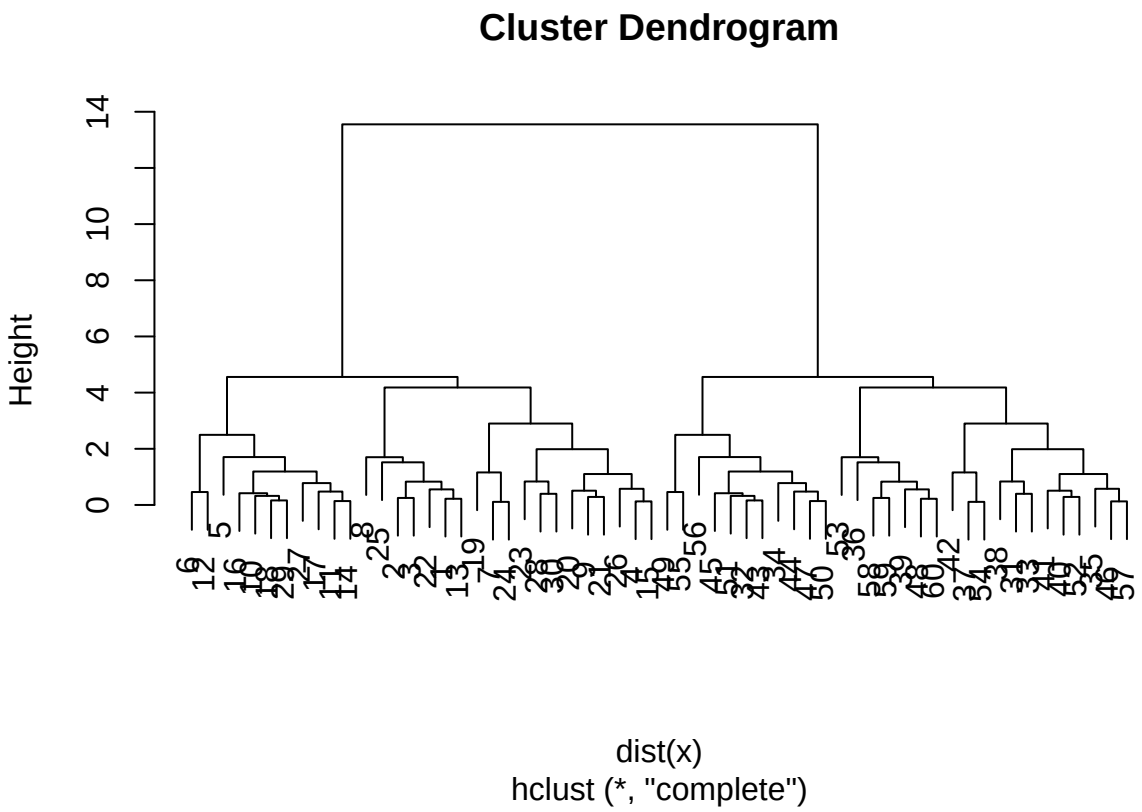
##Now for hclust(), we will cluster the same data "x" with the hclust(). This requires a distance matrix.

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

```
##
## Call:
## hclust(d = dist(x))
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 60
```

Let's plot

```
plot(hc)
```



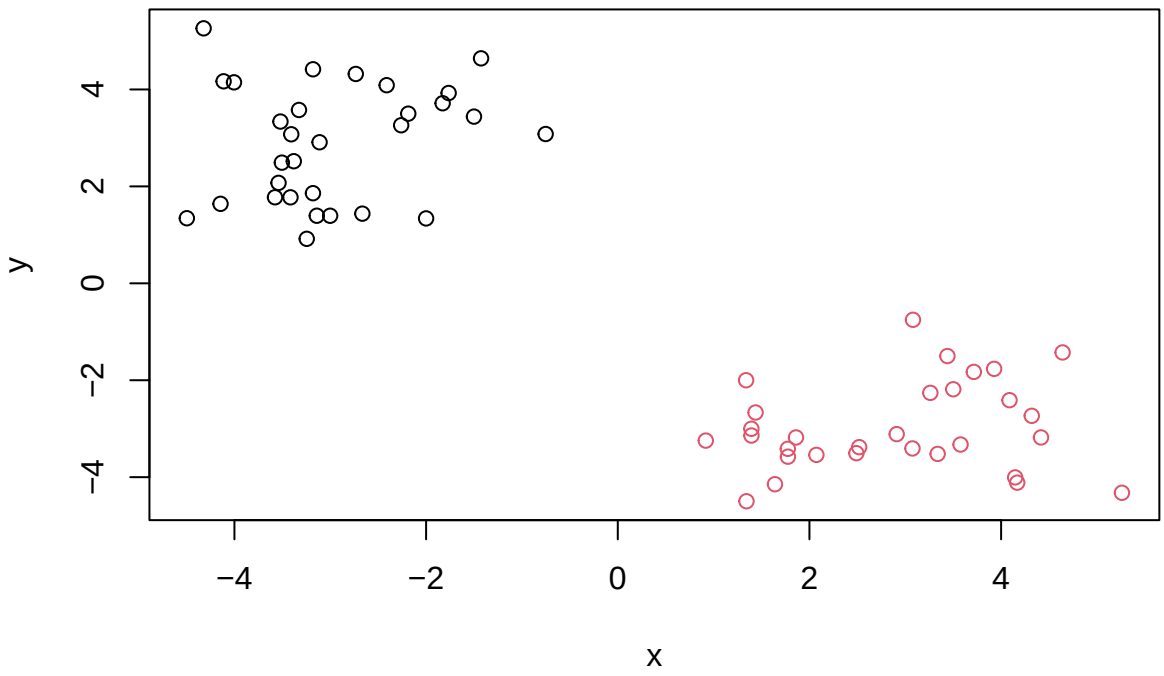
Get cluster membership, cut with cutree()

```
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 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
```

Now plot hclust() results

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



Principal Component Analysis(PCA)

##PCA of UK food data

Read data from website and try a few visualizations

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)
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
## 7      Fresh_potatoes      720   874      566      1033
## 8           Fresh_Veg      253   265      171       143
## 9           Other_Veg      488   570      418       355
## 10 Processed_potatoes      198   203      220       187
## 11      Processed_Veg      360   365      337       334
## 12        Fresh_fruit     1102  1137      957       674
## 13            Cereals     1472  1582     1462      1494
## 14          Beverages       57    73       53        47
## 15        Soft_drinks     1374  1256     1572      1506
## 16   Alcoholic_drinks      375   475      458       135
```

```
## 17      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?

ANS: 17 rows and 5 columns, I can use the dim or nrow and ncol functions.

```
## Complete the following code to find out how many rows and columns are in x?
dim(x)
```

```
## [1] 17  5
```

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?

Ans: I prefer using row.names=1. x <- x[,-1] on repeat deletes the first column again and again into the dataset.

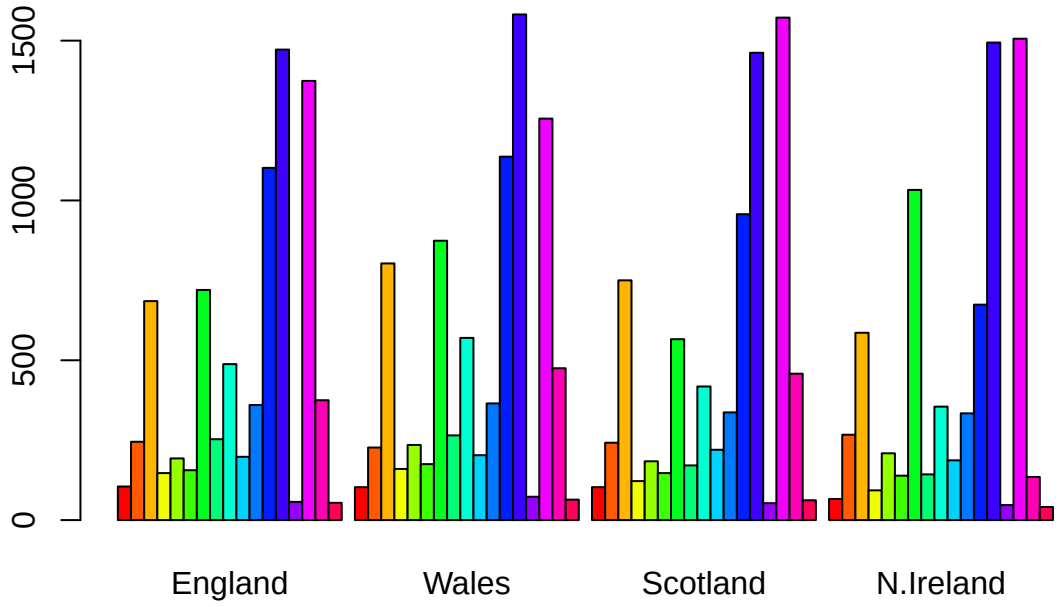
```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url, row.names=1)
x
```

```
##                     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
```

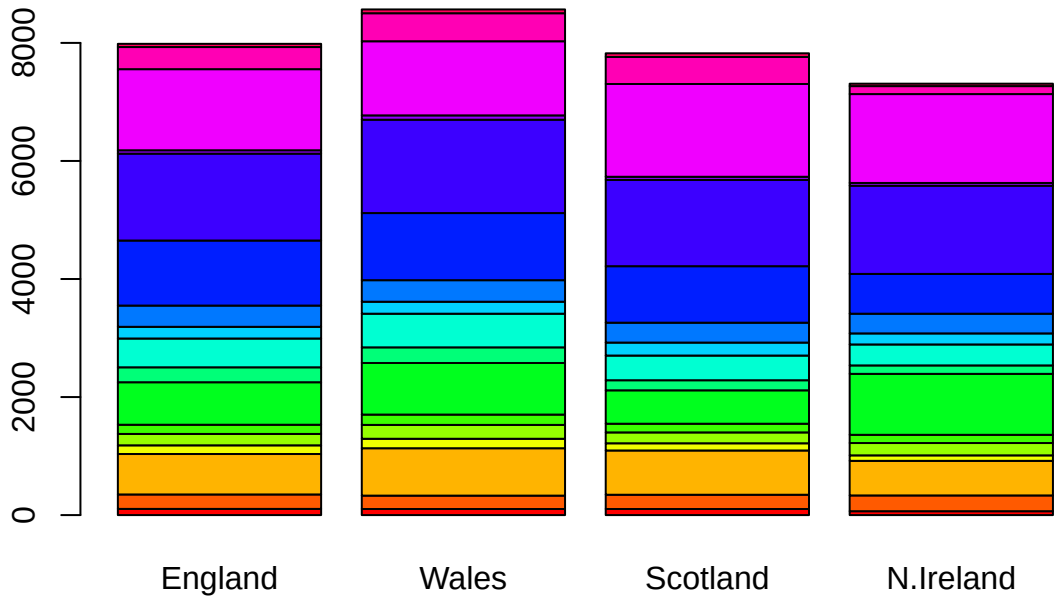
Q3: Changing what optional argument in the above barplot() function results in the following plot?

ANS:delete the beside = TRUE

```
cols<-rainbow(nrow(x))
barplot(as.matrix(x),col=cols,beside=TRUE)
```



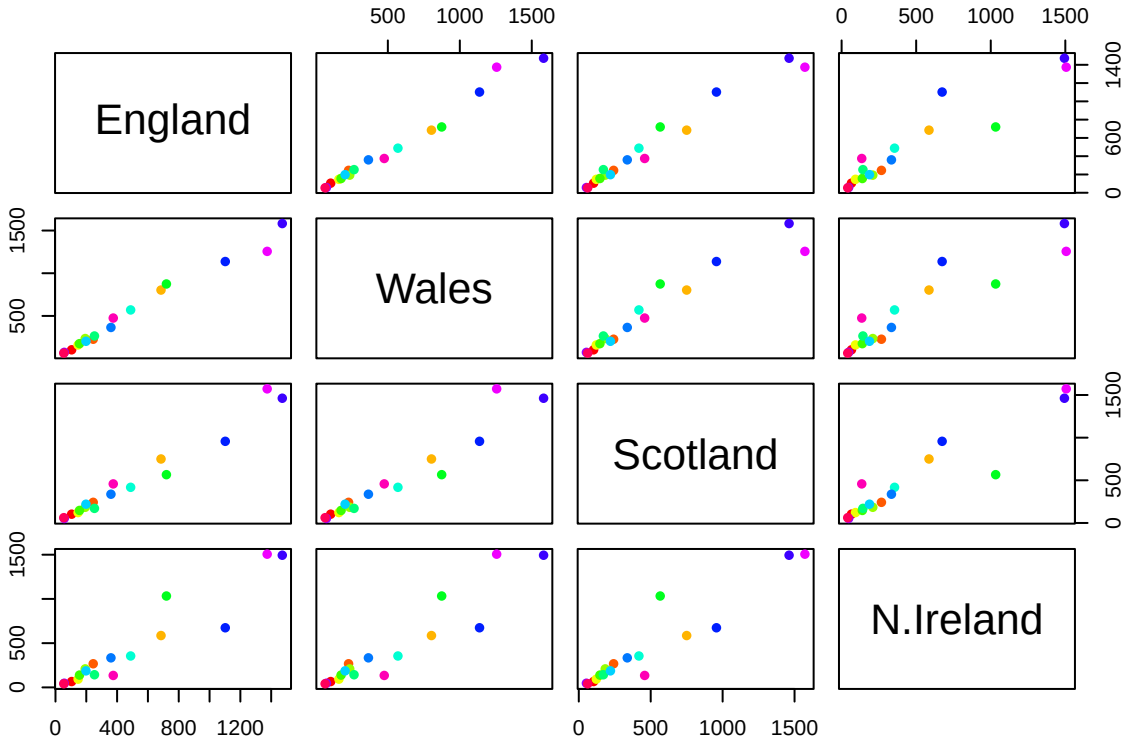

```
cols<-rainbow(nrow(x))
barplot(as.matrix(x),col=cols)
```



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?

ANS: "pairs(x)" makes the pairwise plots between each two different countries. If the point lies on the diagnol, it means that the two countries consume around the same amount for that specific food. If the point shifts to the top, it means the country on the y axis eats more of that food. If it shifts to the right, the country on the x axis consumes more of that food.

```
pairs(x, col=cols, pch=16)
```



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

ANS: North Ireland does not have a good diagnol. They eat foods at different distribution compared to other countries.

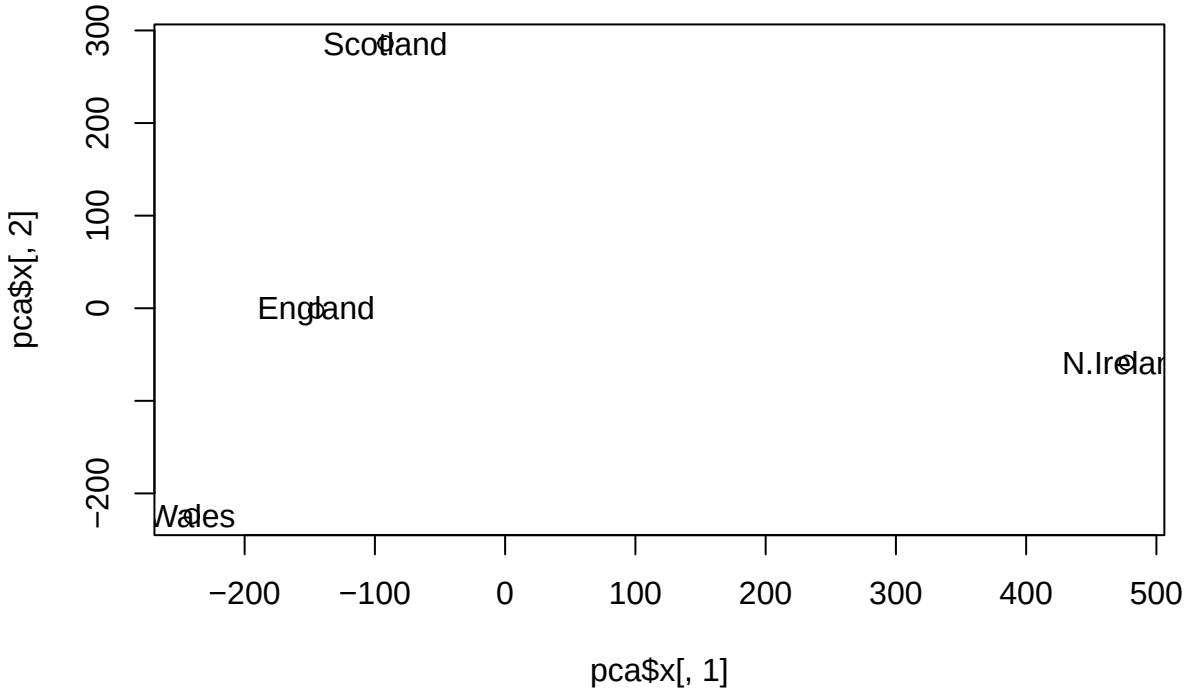
We need PCA for better visualizaton. The main base R PCA function is called "prcomp()". We need to fist transpose our input data.

```
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
```

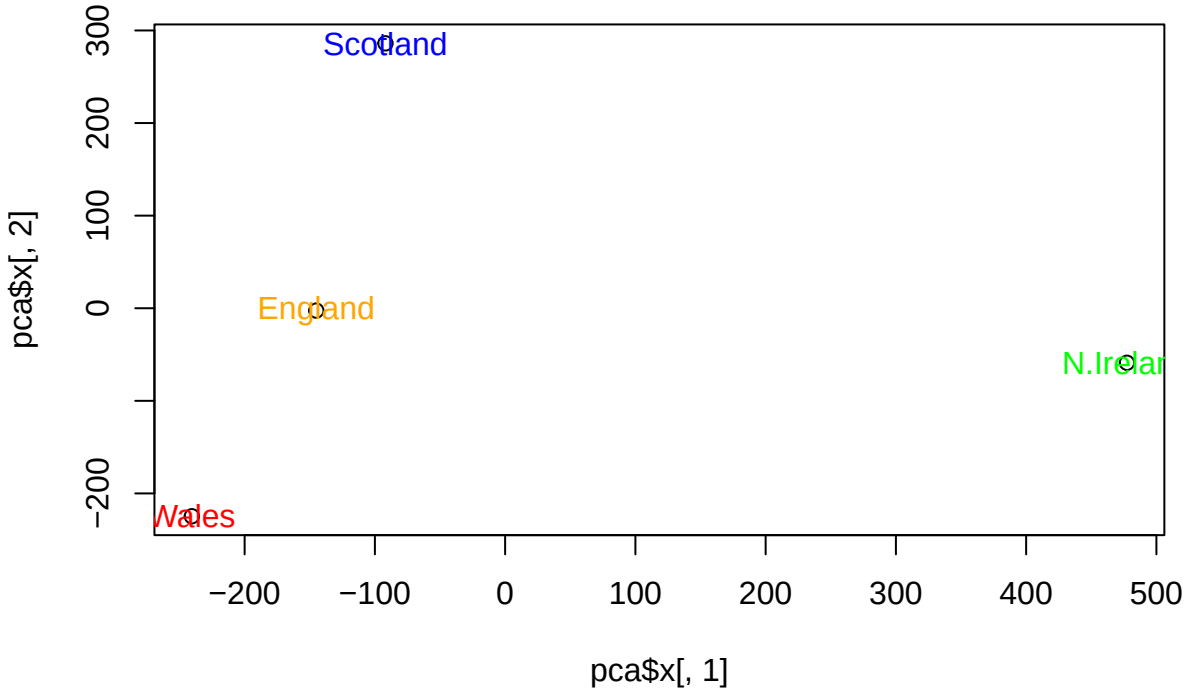
Q7. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points.

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

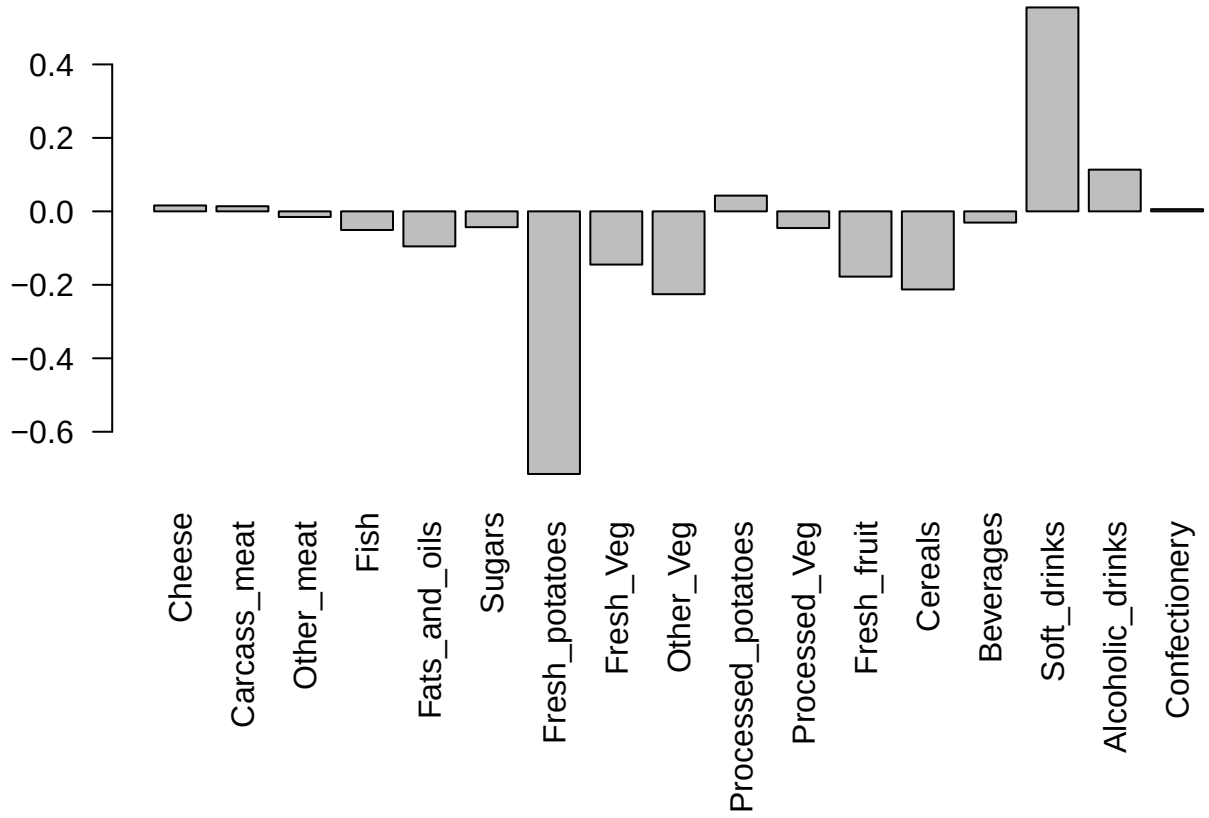
```
country_cols<-c("orange","red","blue","green")
plot(pca$x[,1],pca$x[,2])
text(pca$x[,1],pca$x[,2], colnames(x),col=country_cols)
```



Q9: Generate a similar 'loadings plot' for PC2. What two food groups feature prominantely and what does PC2 maninly tell us about?

ANS:Here we see observations (foods) with the largest positive loading scores (soft_drinks) that effectively "push" Scotland to the top.cWe also see observations/foods with high negative scores(fresh_potatoes) that push wales to the bottom.

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



Q10: How many genes and samples are in this data set?

ANS: 11 samples and 6 genes

```
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
```

```
pca<-prcomp(t(rna.data),scale=TRUE)
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     9.6237 1.5198 1.05787 1.05203 0.88062 0.82545 0.80111
## Proportion of Variance 0.9262 0.0231 0.01119 0.01107 0.00775 0.00681 0.00642
## Cumulative Proportion  0.9262 0.9493 0.96045 0.97152 0.97928 0.98609 0.99251
##                           PC8     PC9      PC10
## Standard deviation     0.62065 0.60342 3.457e-15
## Proportion of Variance 0.00385 0.00364 0.000e+00
## Cumulative Proportion  0.99636 1.00000 1.000e+00
```

```
plot(pca$x[,1],pca$x[,2],xlab="PC1", ylab="PC2")
text(pca$x[,1],pca$x[,2],colnames(rna.data))
```