Class 7: 1 Machine Learning

#clustering We will start with k-means clustering, one od the most prevelent of all clustering methods. Fast and could use large data set. Disadvantage is need to tell it value of k (how many clusters).

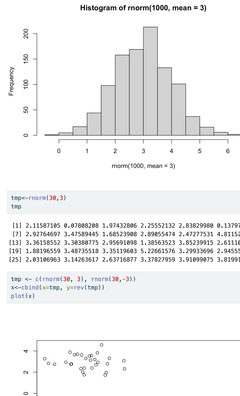
To get started let's make some data up:

```
rnorm(10)
```

```
 [1] -0.6862395 -0.3129806 -0.7094868  0.6377094 -1.1105379  1.2693465
 [7] -0.1155685  0.4072695  1.0197550  0.3252456
```

#rnorm will give you a random numbers. Could get a histogram of values.

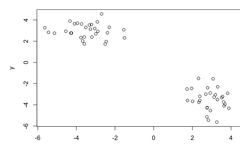
```
hist(rnorm(1000, mean=3))
```




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

```
 [1] 2.11587105 0.07088208 1.97432806 2.25552132 2.83829980 0.13797719
 [7] 2.92764697 3.47589445 1.68523908 2.89053474 2.47277531 4.81152179
[13] 3.96158552 3.30388775 2.05691898 1.38563523 3.85239915 2.61110471
[19] 1.80196559 3.48735518 3.35119083 5.22661576 3.29933696 2.94555853
[25] 2.83186963 3.14263617 2.63716877 3.37827959 3.91899075 3.81991285
```

```
tmp <- c(rnorm(30, 3), rnorm(30,-3))
x<-cbind(x=tmp, y=rev(tmp))
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  2.961298 -3.459674
2 -3.459674  2.961298

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 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

Within cluster sum of squares by cluster:
[1] 42.5357 42.5357
 (between_SS / total_SS =  93.6 %)

Available components:

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

"centers" in k is the number of clusters you want (ie k) >Q1. How many points are in each cluster?

```
k$size
```

```
[1] 30 30
```

> 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 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
```

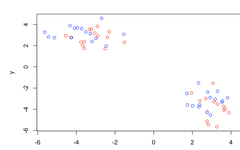
> Q3. cluster centers

```
k$centers
```

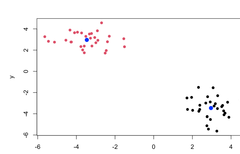
```
          x         y
1  2.961298 -3.459674
2 -3.459674  2.961298
```

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

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

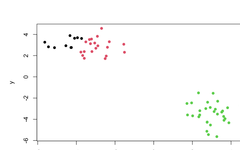



```
plot(x, col=k$cluster, pch=16)
points(k$centers, col="blue", pch=20, cex=2)
```



"pch" is the size of dots >Q5. Run the kmeans again but cluster into 3 groups and plot the results like we did above.

```
k<-kmeans(x, centers=3, nstart=20)
plot(x, col=k$cluster, pch=16)
```



k-means will always return a clustering result - even if there is no clear groupings.

#hierarchical clustering

hierarchical clustering it has an advantage in that it can reveal the structure in your data rather than imposing a structure as k-means will.

The main function in "base" R is called `hclust()`

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

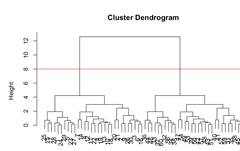
```
hc<-hclust(dist(x))
hc
```

```

Call:
hclust(d = dist(x))

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

```
plot(hc)
abline(h=8, col="red")
```



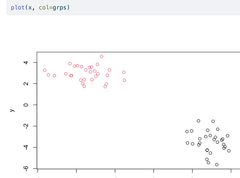
The function to get our clustering/groups from a hclust object is called `cutree()`

```
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 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
```

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

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



#principal component analysis (PCA)

class 07: 2 Machine Learning

Kalodiah Toma (PID: A07606689)

```
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?

```
nrow(x)
```

```
[1] 17
```

```
ncol(x)
```

```
[1] 4
```

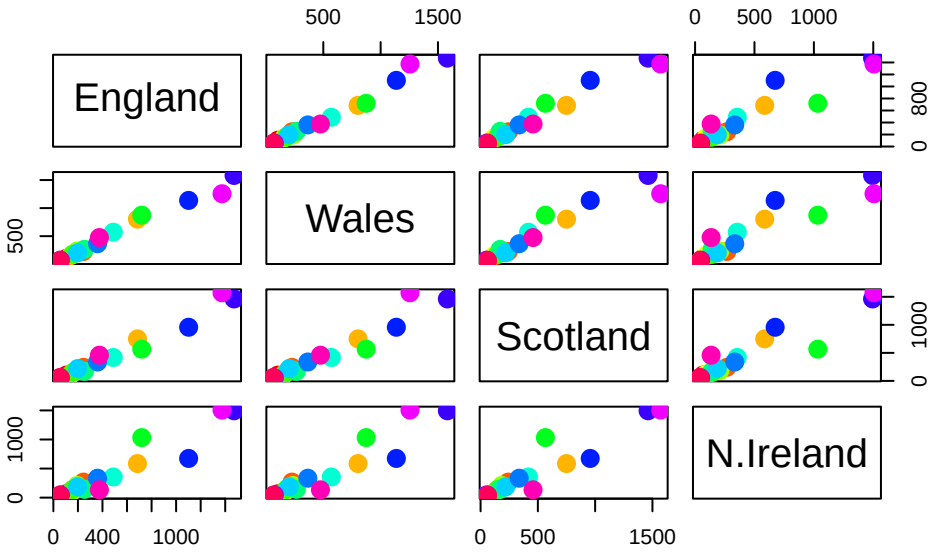
```
head(x)
```

	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

```
tail(x)
```

	England	Wales	Scotland	N.Ireland
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

```
pairs(x, col=rainbow(17), pch=16, cex=2)
```



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? x <- read.csv(url, row.names=1) head(x) This is the preferred method as it it is not self distructive and will be transferred when rendering.

#PCA to the resuce

Help make sense of this data… The main function for PCA is base R is called `prcomp()`

It wants the transpose (with the `t()`) of our food data for analysis

```
dim(x)
```

```
[1] 17  4
```

```
t(x)
```

```
          Cheese Carcass_meat  Other_meat  Fish Fats_and_oils  Sugars
England      105           245         685  147            193    156
Wales        103           227         803  160            235    175
Scotland     103           242         750  122            184    147
N.Ireland     66           267         586   93            209    139
          Fresh_potatoes  Fresh_Veg  Other_Veg  Processed_potatoes
England              720        253        488                 198
Wales                874        265        570                 203
Scotland             566        171        418                 220
N.Ireland           1033        143        355                 187
          Processed_Veg  Fresh_fruit  Cereals  Beverages Soft_drinks
England             360         1102     1472         57         1374
Wales               365         1137     1582         73         1256
Scotland            337          957     1462         53         1572
N.Ireland           334          674     1494         47         1506
          Alcoholic_drinks  Confectionery
England                375             54
Wales                  475             64
Scotland               458             62
N.Ireland              135             41
```

```
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
```

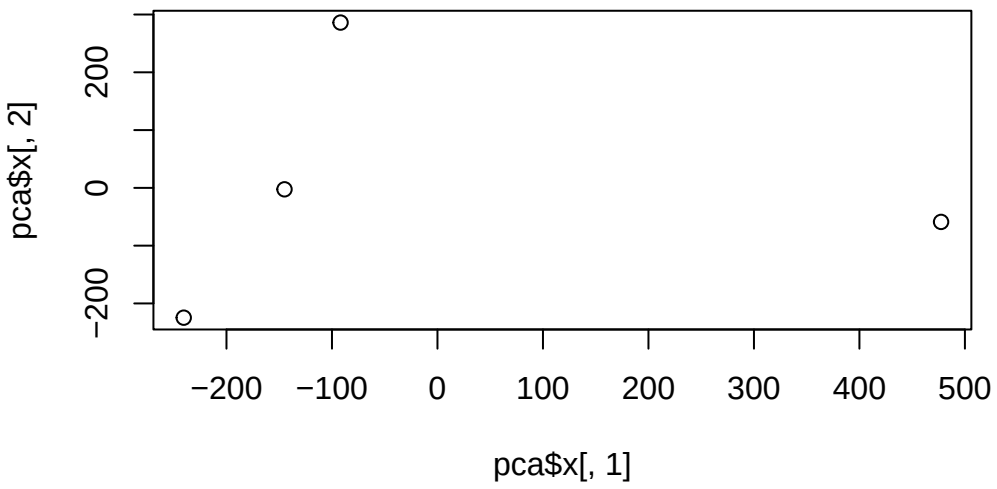
PC1 captures over 67% of data PC2 captures nearly 30% of data so if make a plot of PC1 AND PC1 have 97% of data. Do not need all 17 PC.

One of the main results that folks look for is called the "score plot" aka PC plot, PC1 vs PC2 plot….

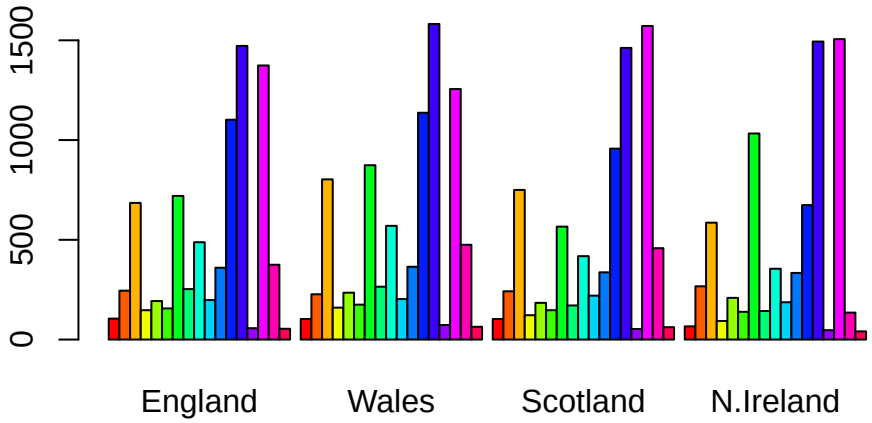
```
pca$x
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -4.894696e-14
Wales     -240.52915 -224.646925 -56.475555  5.700024e-13
Scotland   -91.86934  286.081786 -44.415495 -7.460785e-13
N.Ireland  477.39164  -58.901862  -4.877895  2.321303e-13
```

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

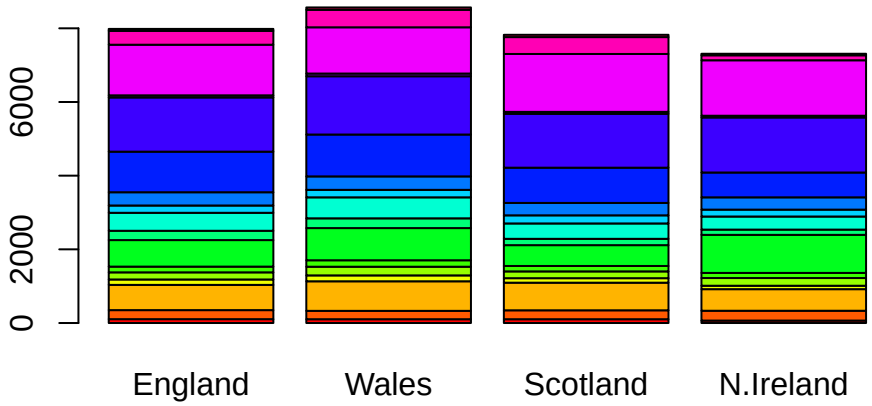


```
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```



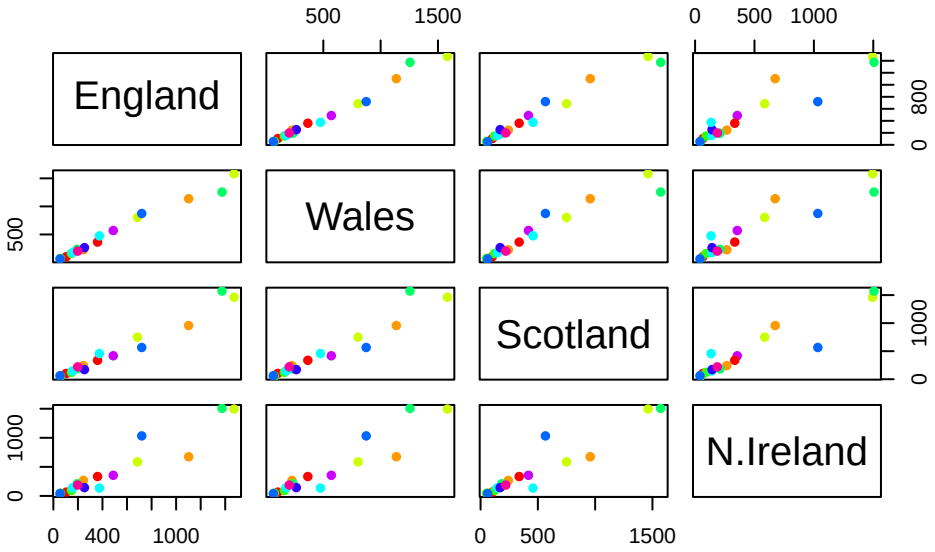
Q3: Changing what optional argument in the above barplot() function results in the following plot? leaving beside out or setting it to false sets the "value" of beside as "0)

```
barplot(as.matrix(x), beside = 0, 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?

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



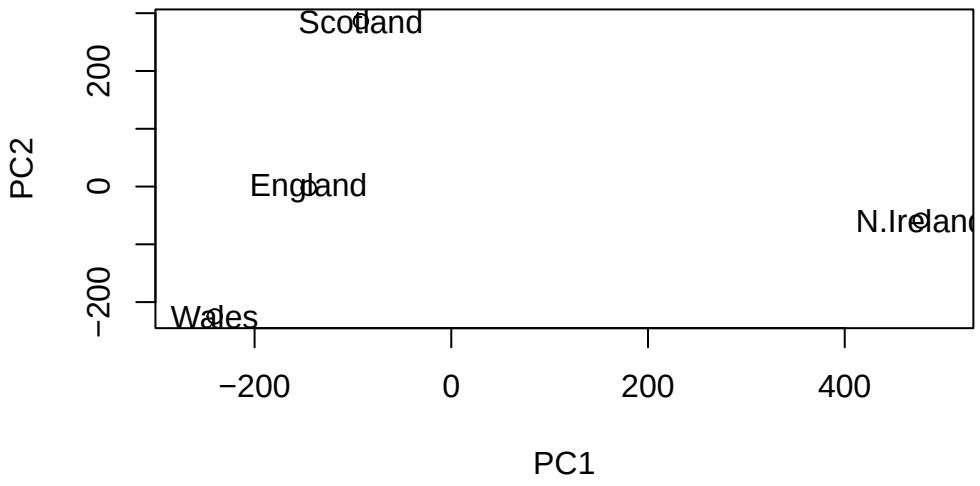
Q6. What is the main differences between N. Ireland and the other countries of the UK in terms of this data-set? There is one variable (in blue) that is most different and higher in N.Ireland. It is difficult to know what this point is from this type of graph. This is likely alcoholic drinks.

```
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
```

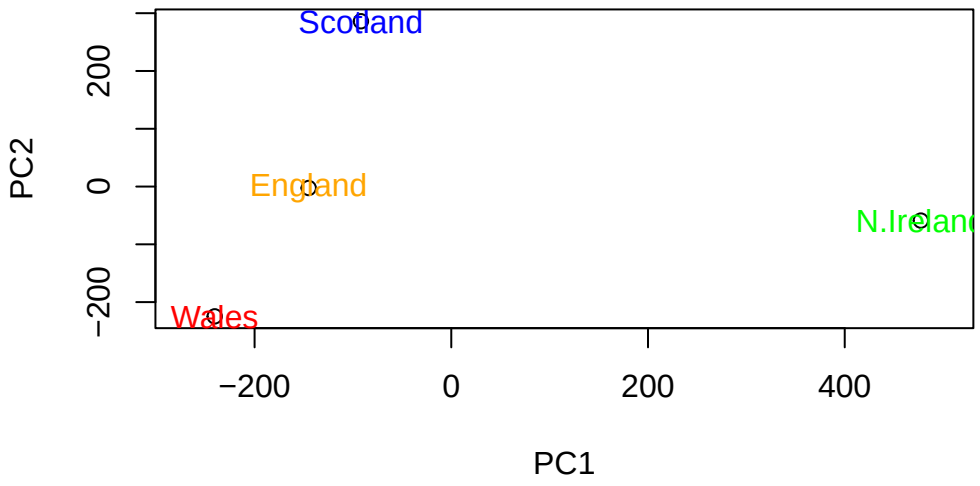
Q7. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points.

```
plot(pca$x[,1],pca$x[,2], 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(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("orange","red", "blue", "green"))
```



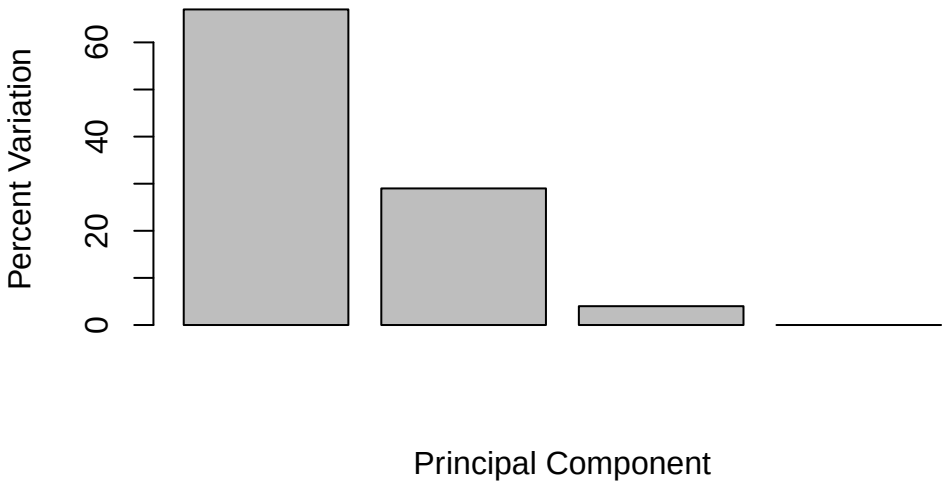

```
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

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

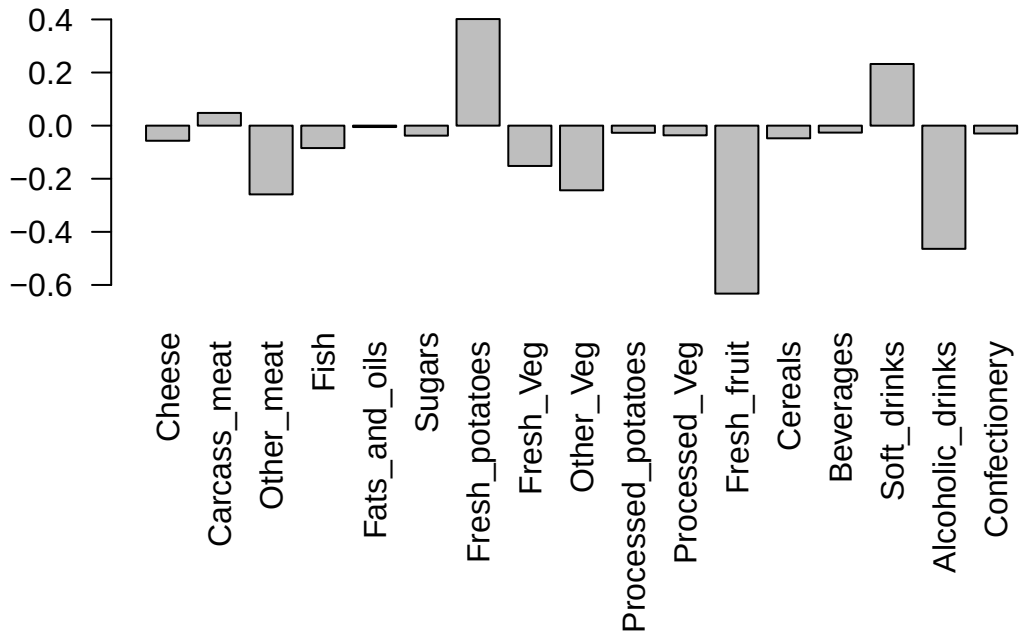
```
z <- summary(pca)
z$importance
```

```
                             PC1       PC2      PC3           PC4
Standard deviation     324.15019 212.74780 73.87622 3.175833e-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")
```

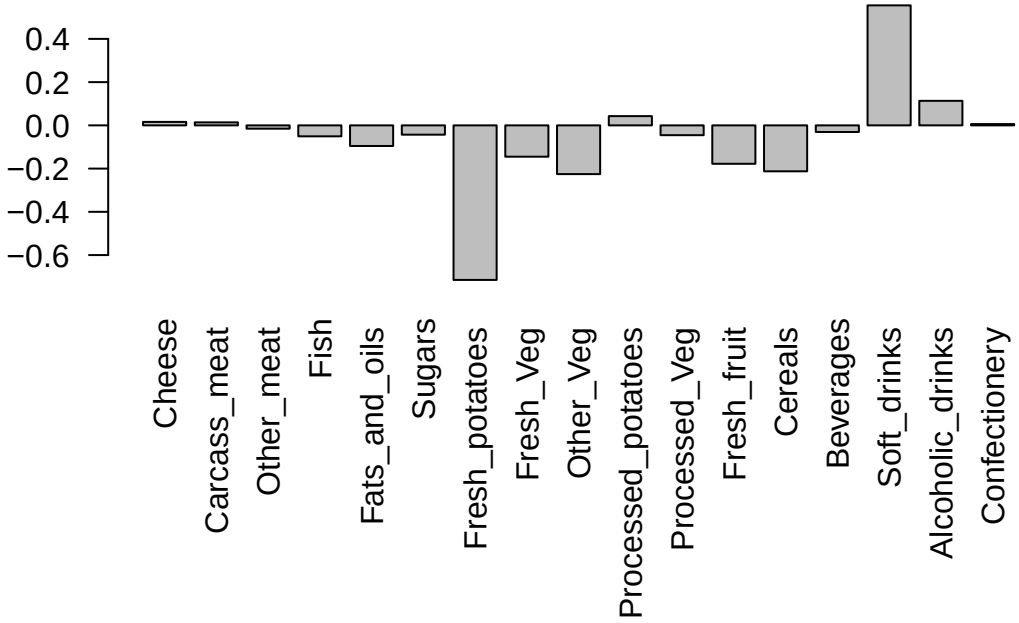



```
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,1], las=2 )
```



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

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



PC2 mainly tells us the variance between Wales and Scotland. The two food groups that feature prominantely are "fresh potatoes" and "soft-drinks".