

# Class 07: Machine Learning 1

Helena Khalil #A16313711

We will start today's lab with clustering methods, in particular so-called K-means. The main function for this in R is `kmeans()`.

Let's try it on some made up data where we know what the answer should be

```
x<- rnorm(10000, mean=3)
hist(x)
```



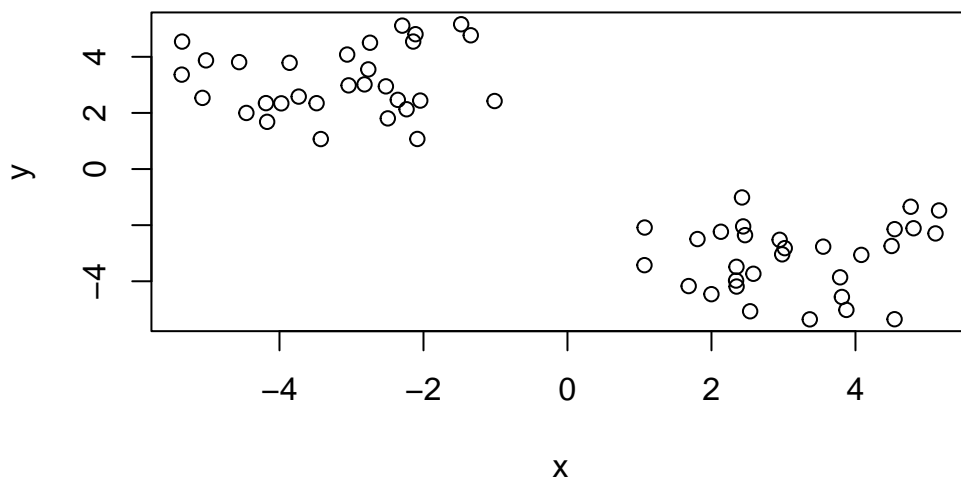
60 points

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

```
      x      y
[1,] 2.581773 -3.732943
[2,] 3.365479 -5.357727
[3,] 2.423953 -1.012311
[4,] 2.536880 -5.069415
[5,] 2.466016 -2.356520
[6,] 2.347141 -3.484514
```

We can pass this to the base R `plot()` function for a quick.

```
plot(x)
```



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

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

Cluster means:

	x	y
1	-3.173481	3.136637
2	3.136637	-3.173481

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

Within cluster sum of squares by cluster:

```
[1] 83.57406 83.57406
(between_SS / total_SS = 87.7 %)
```

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
```

Q2. 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 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

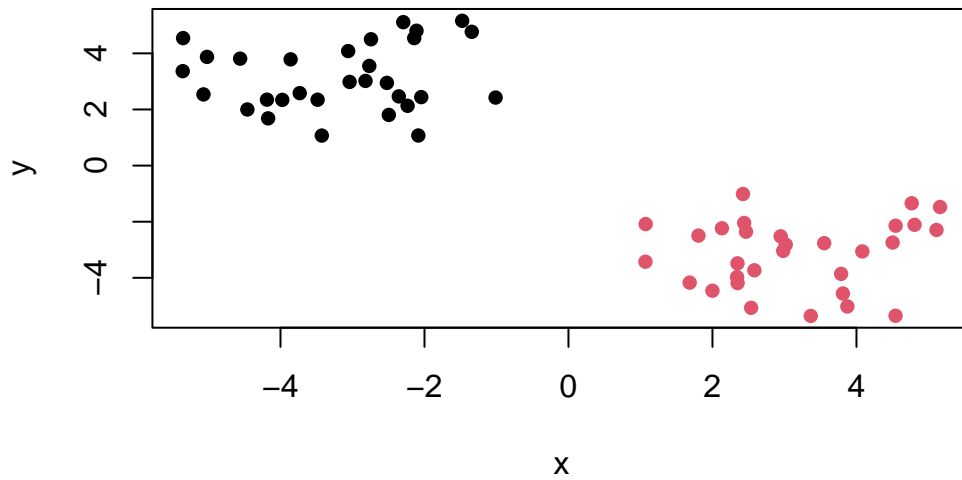
Q3. Cluster centers?

```
k$centers
```

	x	y
1	-3.173481	3.136637
2	3.136637	-3.173481

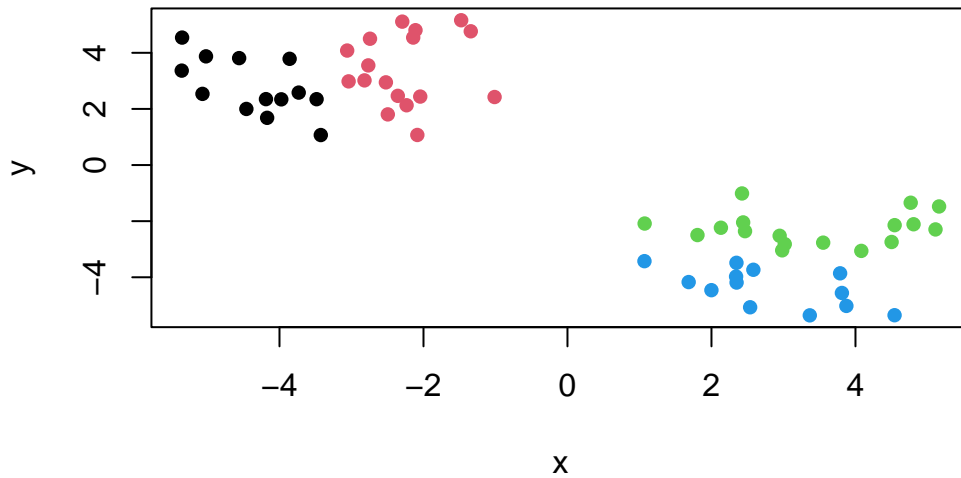
Q4. Plot my clustering results

```
plot(x, col=k$cluster, pch=16)
```



Q5. Cluster the data again with `kmeans()` into 4 groups and plot the results.

```
k4 <- kmeans(x, centers=4, nstart=20)  
plot(x, col=k4$cluster, pch=16)
```



K-means is very popular mostly because it is fast and relatively straightforward to run and understand. It has a big limitation in that you need to tell it how many groups (k, or centers) you want. #Hierarchical clustering

The main function in base R is called `hclust()`. You have to pass it in a “distance matrix” not just your input data.

You can generate a distance matrix with the `dist()` function.

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

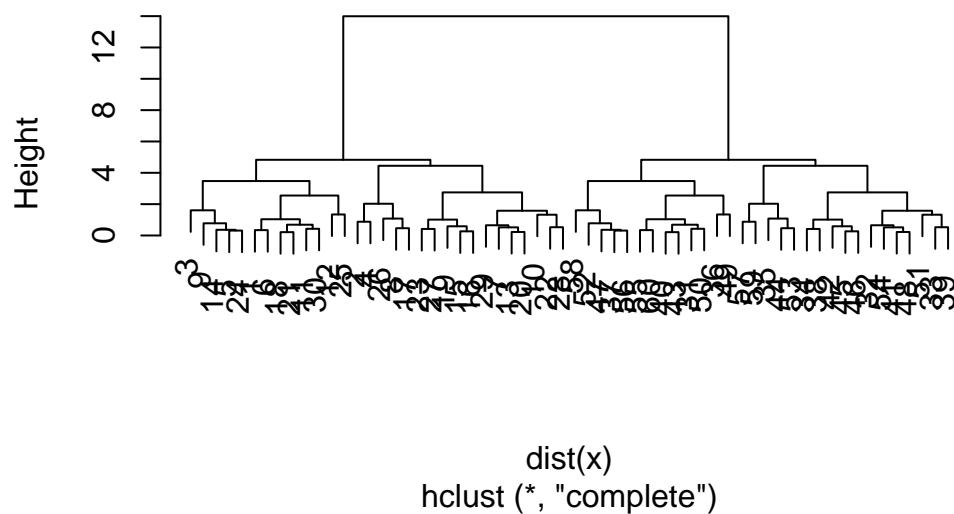
Call:

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

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

```
plot(hc)
```

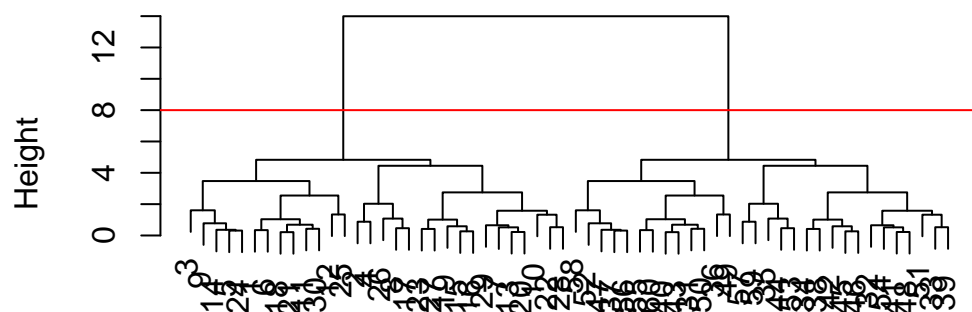
## Cluster Dendrogram



To find the clusters (cluster membership vector) from a 'hclust()' result we can “cut” the tree at a certain height that we like.

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

## Cluster Dendrogram



dist(x)  
hclust (\*, "complete")

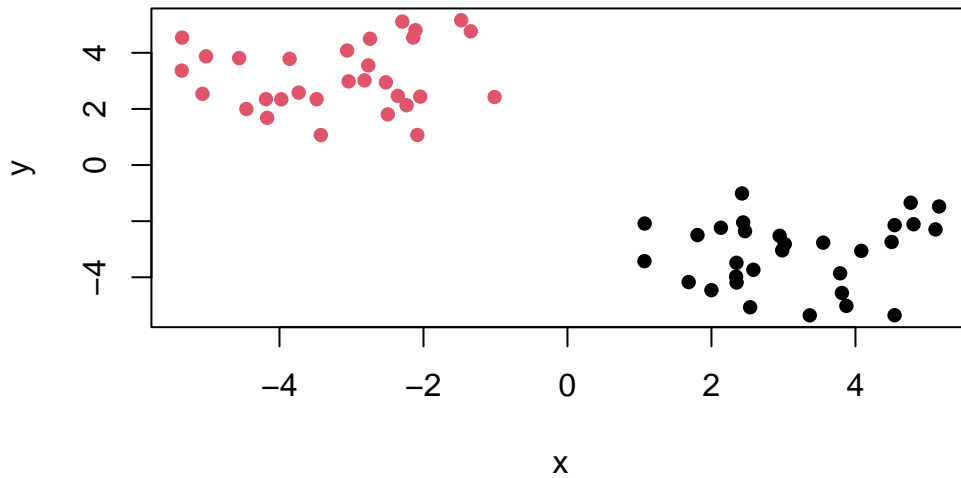
```
grps <- cutree(hc, h=8)
```

```
table(grps)
```

```
grps
 1  2
30 30
```

Q6. Plot our hclust results.

```
plot(x,col=grps,pch=16)
```



#Principal Component Analysis

##PCA of UK food data

Read data showing the consumption

Let's see how PCA can help us but first we can try conventional analysis.

```
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?

```
dim(x)
```

```
[1] 17  5
```

```
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

I need to fix that first column...

```
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

```
dim(x)
```

```
[1] 17  4
```

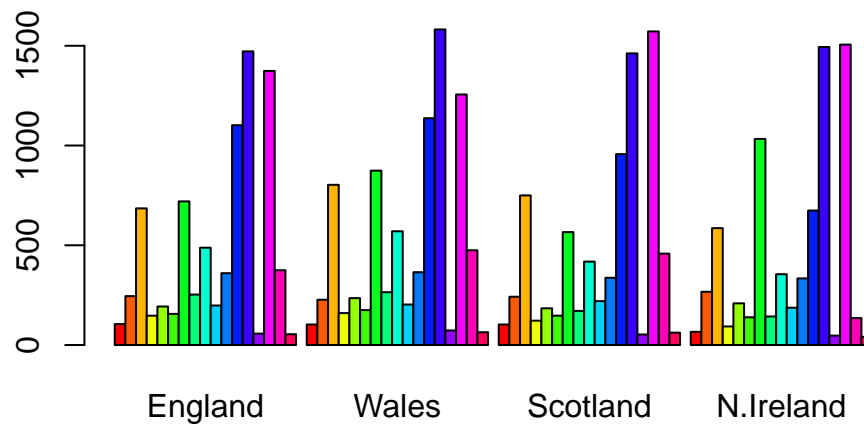
```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)
x <- read.csv(url, row.names=1)
head(x)
```

	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?

The first approach take away a column each time which if you play it multiple times it will start deleting data.

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



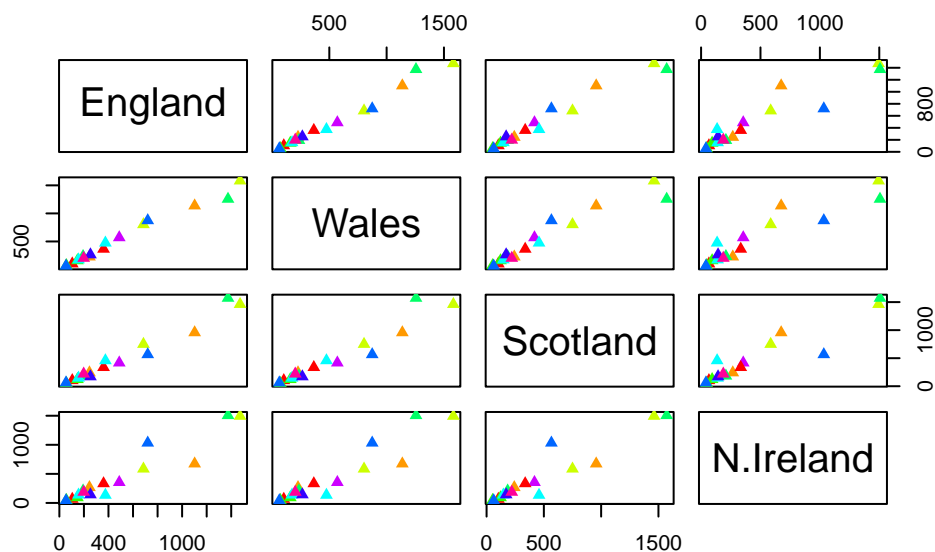
Q3: Changing what optional argument in the above `barplot()` function results in the following plot?

Changing the `beside` factor to `false` makes it a thicker bar chart

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?

The diagonal of the plot shows similarities between the countries and the green dot is the outlier.

```
pairs(x, col=rainbow(10), pch=17)
```



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

It is hard to tell but Ireland looks different than the other countries, its like the outlier in the situation

##Principal Component Analysis (PCA)

PCA can help us make sense of these types of datasets. Let's see how it works.

The main function in "base" R is called `prcomp()`. In this case we want to first take the transpose of our input `x` so the columns are the food types and the countries are the rows.

```
head(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:

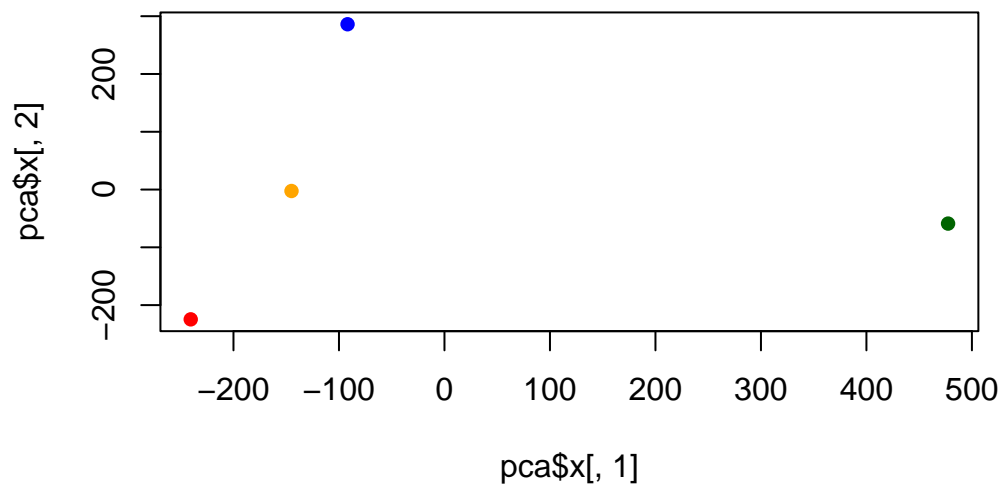
	PC1	PC2	PC3	PC4
Standard deviation	324.1502	212.7478	73.87622	3.176e-14
Proportion of Variance	0.6744	0.2905	0.03503	0.000e+00
Cumulative Proportion	0.6744	0.9650	1.00000	1.000e+00

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

```
pca$x
```

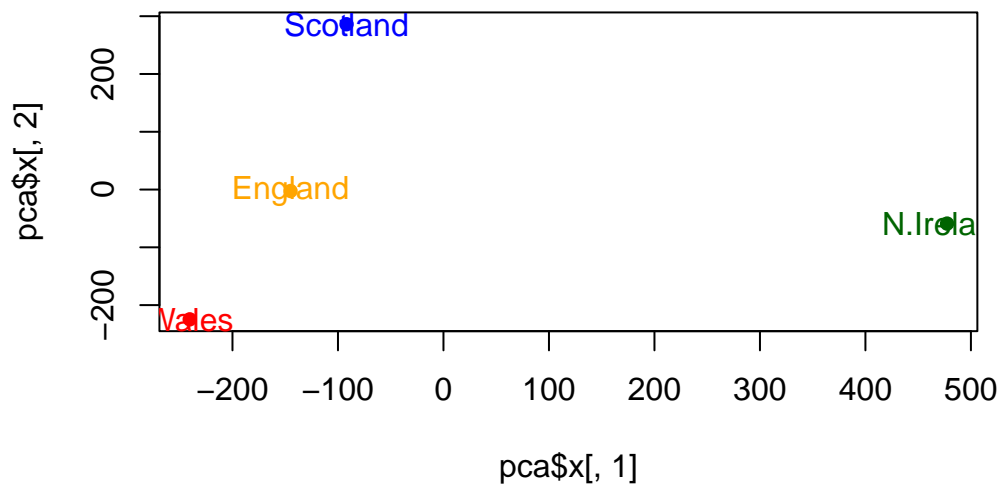
	PC1	PC2	PC3	PC4
England	-144.99315	-2.532999	105.768945	-4.894696e-14
Wales	-240.52915	-224.646925	-56.475555	5.700024e-13
Scotland	-91.86934	286.081786	-44.415495	-7.460785e-13
N.Ireland	477.39164	-58.901862	-4.877895	2.321303e-13

```
plot(pca$x[,1], pca$x[,2], col=c("orange", "red", "blue", "darkgreen"), pch=16)
```



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], col=c("orange", "red", "blue", "darkgreen"), pch=16)
text(pca$x[,1], pca$x[,2], colnames(x), col=c("orange", "red", "blue", "darkgreen"))
```

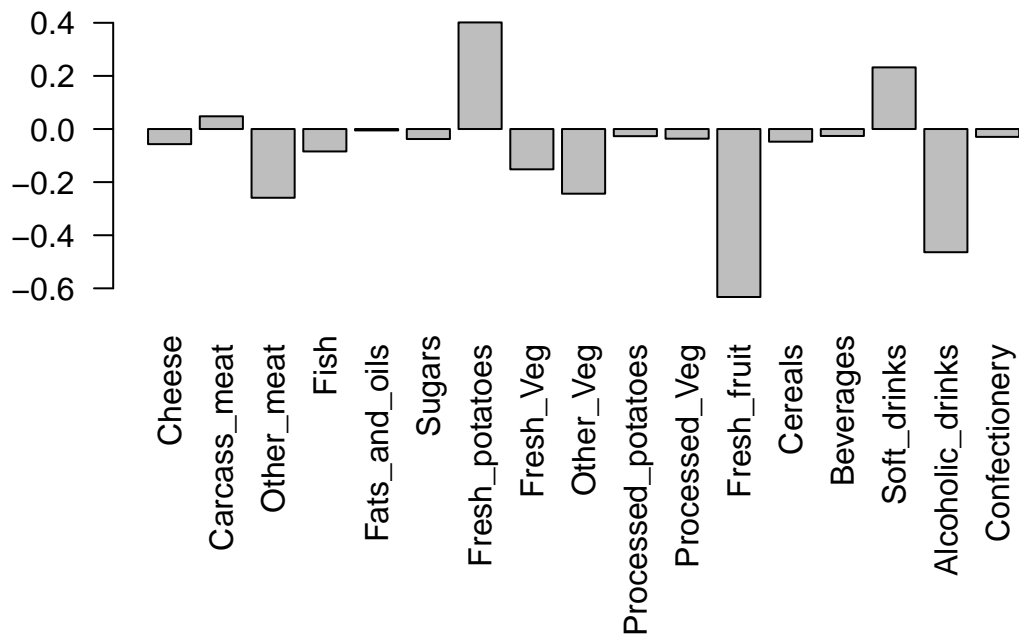


The “loading tell us how much the original variables (in our case the foods) contribute to the new variables.

```
head(pca$rotation)
```

	PC1	PC2	PC3	PC4
Cheese	-0.056955380	0.01601285	0.02394295	-0.694538519
Carcass_meat	0.047927628	0.01391582	0.06367111	0.489884628
Other_meat	-0.258916658	-0.01533114	-0.55384854	0.279023718
Fish	-0.084414983	-0.05075495	0.03906481	-0.008483145
Fats_and_oils	-0.005193623	-0.09538866	-0.12522257	0.076097502
Sugars	-0.037620983	-0.04302170	-0.03605745	0.034101334

```
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 prominently and what does PC2 mainly tell us about?

Fresh potatoes and soft drinks, it tells us about the food groups that differ significantly with Ireland vs the other countries. It accounts for the variants that it didn't account for in PC1

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



