Class 7: Machine Learning I

Youn Soo Na (PID: A1704731)

Today we are going to learn how to apply different machine learning methods, beginning with clustering:

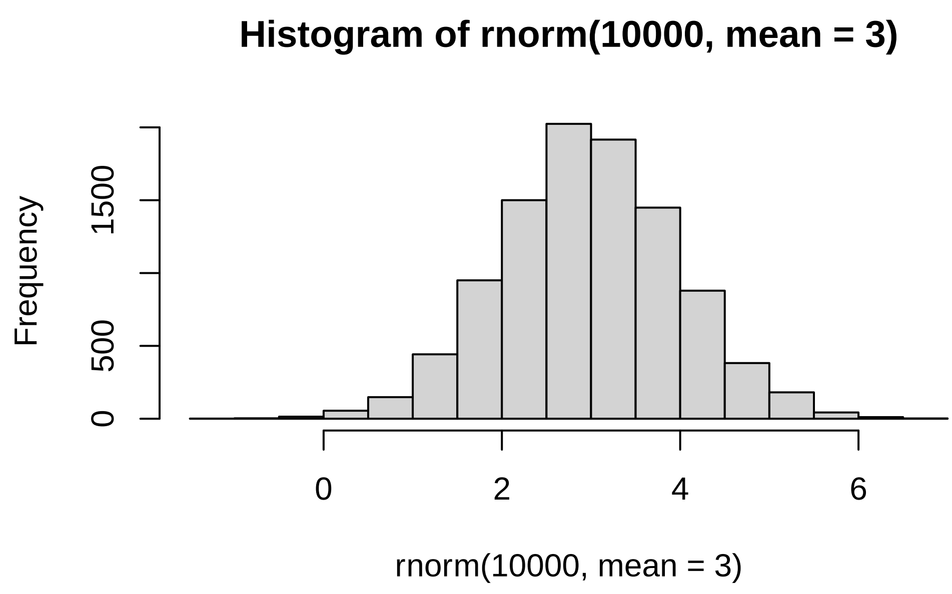
The goal here is to find groups/clusters in your input data.

First, I will make up some data with clear groups. For this I will use the `rnorm()` function:

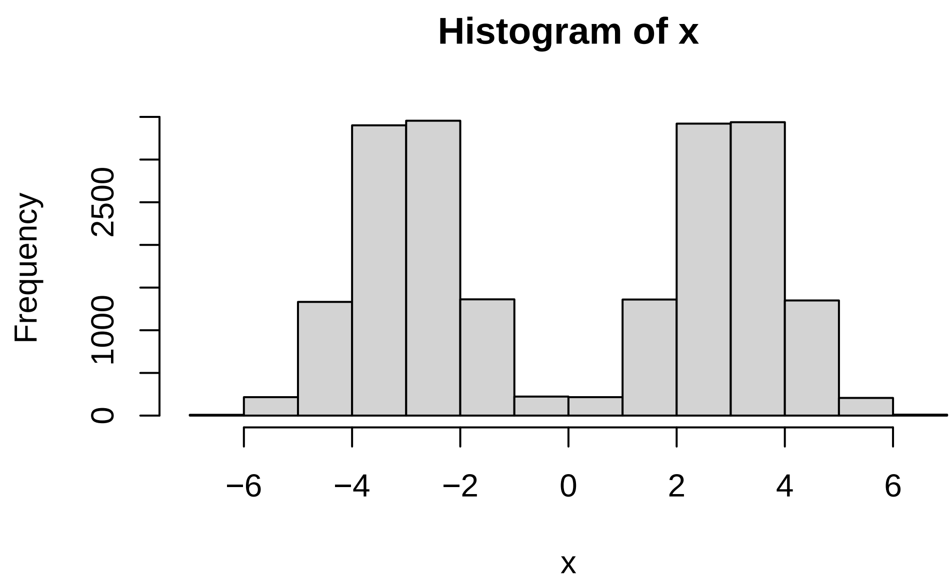
```
rnorm(10)
```

```
 [1] -1.11017313  0.32945160 -0.12018342 -0.32179992  1.77466785  0.32087312
 [7]  0.38968077  0.62894053  0.01826351  0.07473966
```

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



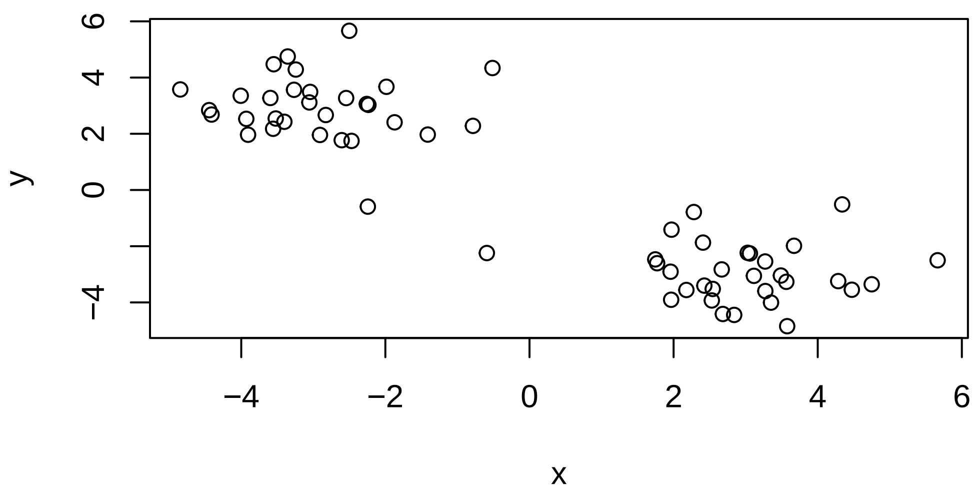
```r
n <- 10000
x <- c(rnorm(n,-3), rnorm(n, +3))
hist(x)
```




```r
n <- 30
x <- c(rnorm(n,-3), rnorm(n, +3))
y <- rev(x)
z <- cbind(x,y)
head(z)
```

```
             x        y
[1,] -4.445269 2.840753
[2,] -3.521256 2.543306
[3,] -3.267614 3.564238
[4,] -3.054634 3.114294
[5,] -4.411724 2.683313
[6,] -3.557461 2.177102
```

```r
plot(z)
```



Use the **kmeans()** function setting k to 2 and nstart=20

Inspect/print the results

Q. How many points are in each cluster?

Q. What 'component' of your result object details - cluster size? - cluster assignment/membership? - cluster center?

```
km <- kmeans(z, centers = 2)
km
```

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

Cluster means:
          x         y
1  2.942853 -2.943101
2 -2.943101  2.942853

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] 69.3442 69.3442
 (between_SS / total_SS =  88.2 %)

Available components:

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

Results in kemans object `km`

```
attributes(km)
```

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

$class
[1] "kmeans"
```

cluster size?

```
km$size
```

```
[1] 30 30
```

cluster assignment/membership?

```
km$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 1 1
```

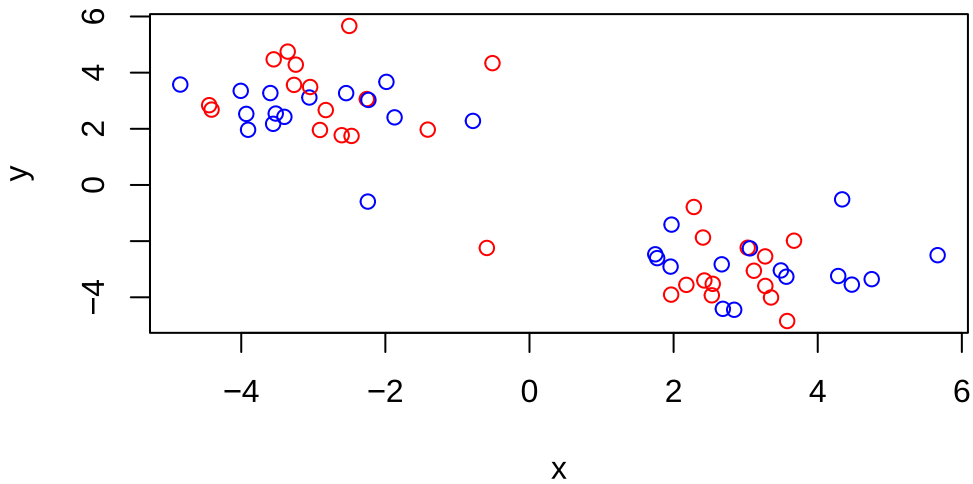
cluster center?

```
km$centers
```

```
          x         y
1  2.942853 -2.943101
2 -2.943101  2.942853
```

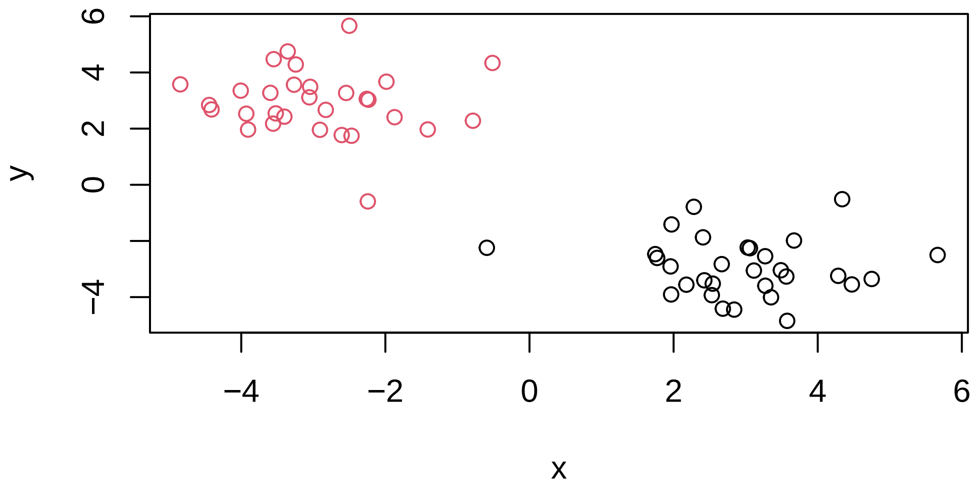
Q. Plot x colored by the kmeans cluster assignment and add cluster centers as blue points.

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



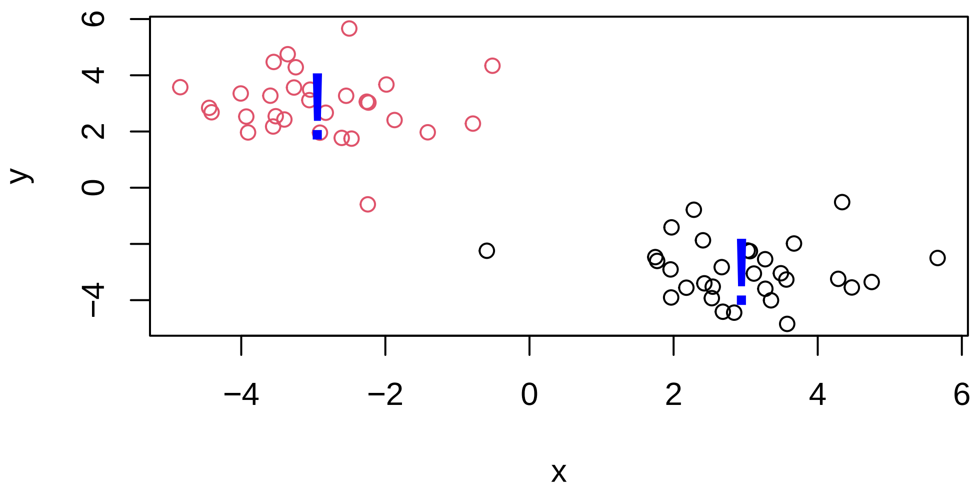
R will re-cycle the shorter color vector to be the same length as the longer (number of data poitns) in z.

```
plot(z, col = km$cluster)
```



We can use the `points()` function to add new points to an existing plot. . .

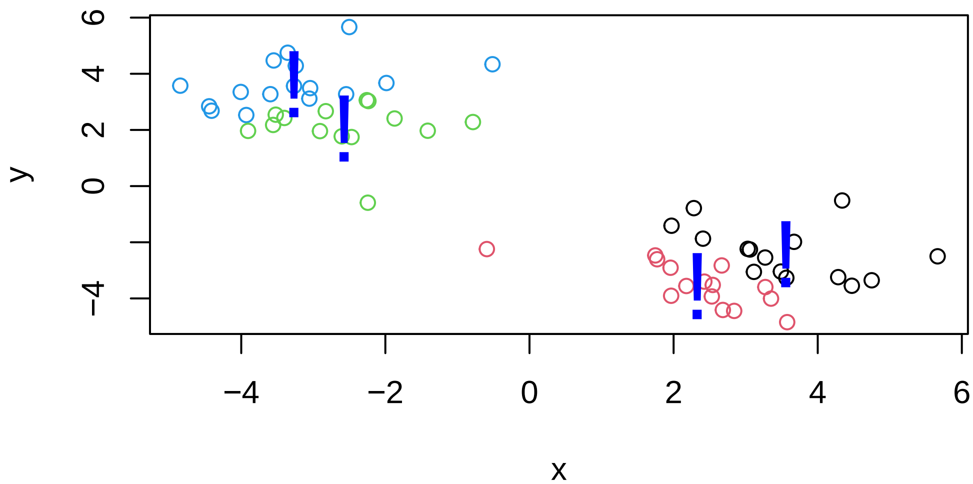
```
plot(z, col = km$cluster)
points(km$centers, col="blue", pch=33, cex = 3)
```



```
# max pch is 127
```

Q. Can you run kmeans and ask for 4 clusters please and plot the results like we have done above?

```
km4 <- kmeans(z, centers = 4)
plot(z, col = km4$cluster)
points(km4$centers, col="blue", pch=33, cex = 3)
```



Hierarchical Clustering

Let's take our same made-up data `z` and see how hclust works.

First we need a distance matrix of our data to be clustered.

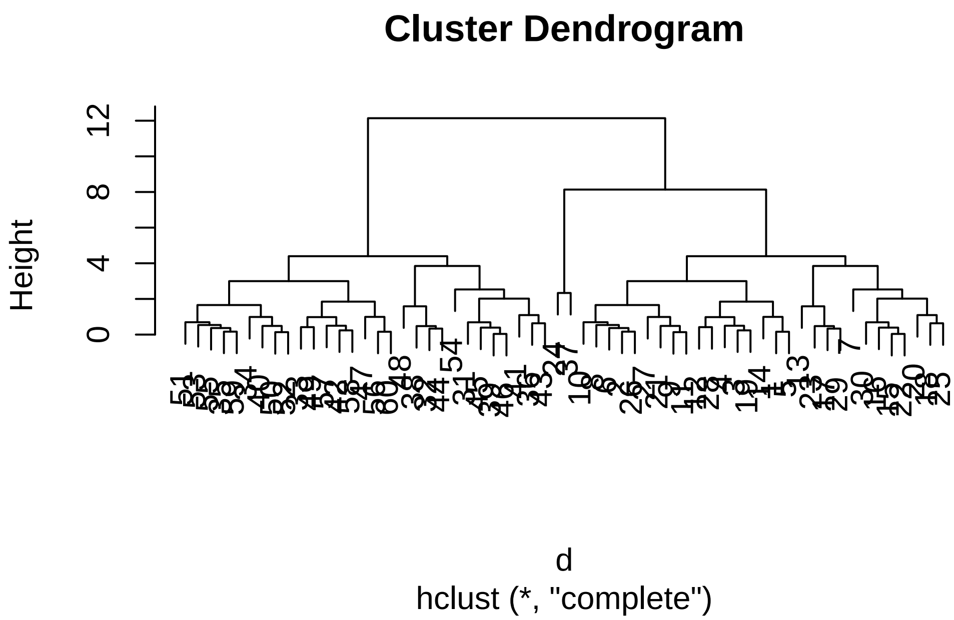
```
d <- dist(z)
hc <- hclust(d)
hc
```

```
Call:
```

```
hclust(d = d)

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

```r
plot(hc)
```



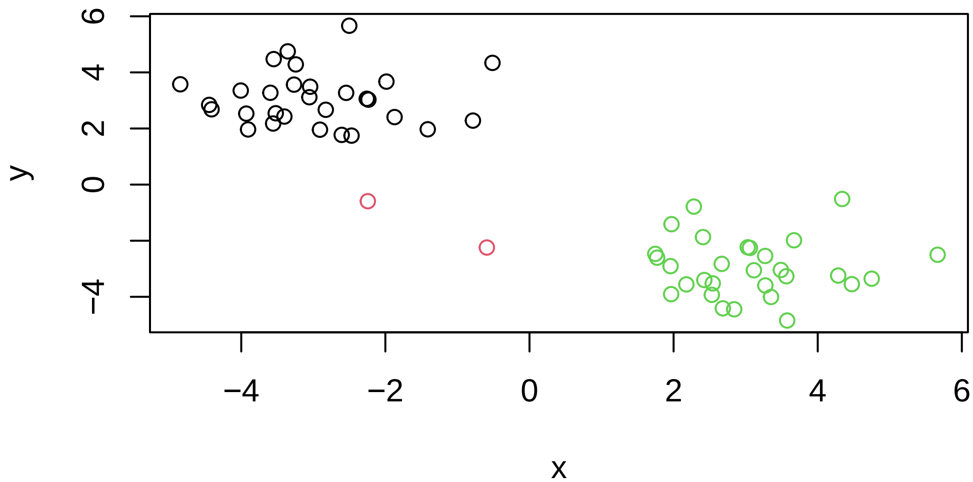
I can get my cluster membership vector by "cutting the tree" with the `cutree()` function like so

```r
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 2 1 1 1 1 1 1 3 3 3 3 3 3 2 3
[39] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

Can you plot `z` colored by out hclust results?

```r
plot(z, col=grps)
```



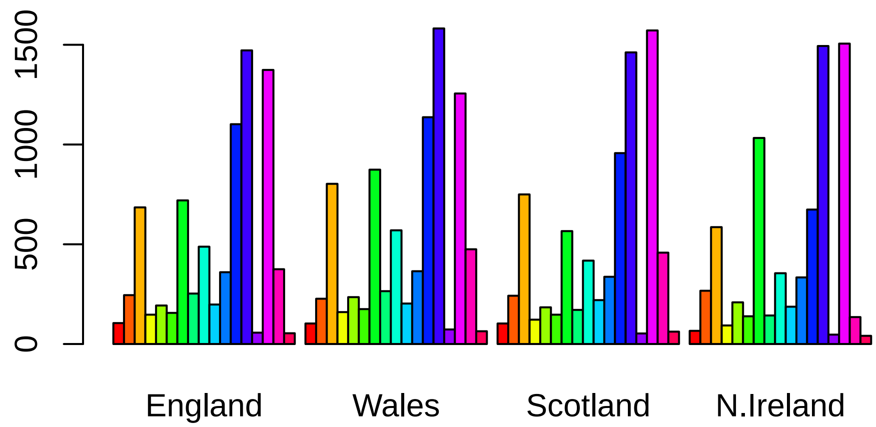
PCA of UK Food Data

Read data from the UK on food consumption in different parts of the UK

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

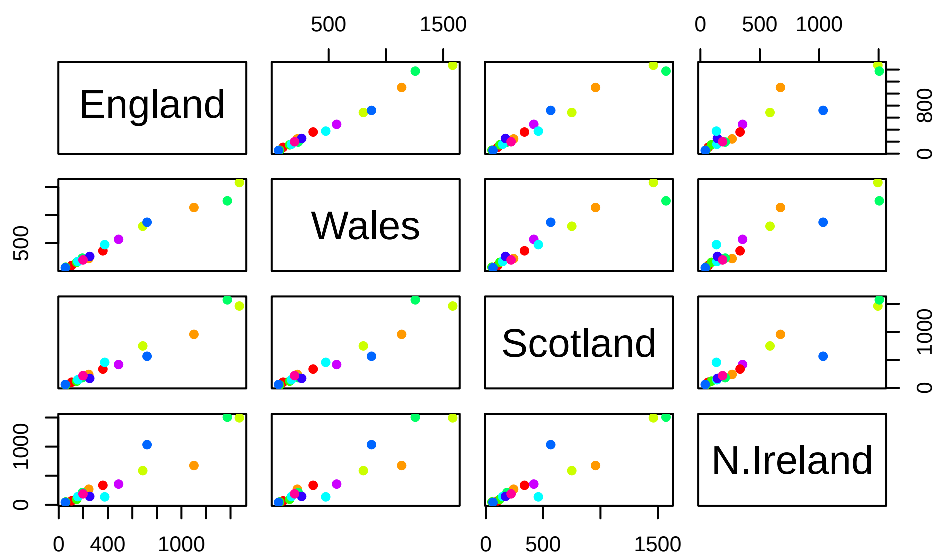
```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars  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

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



A so-called “Pairs” plot can be useful for small datasets like this

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



It's hard to see structure and trends in even this small data-set. How will we ever do this when we have big data-sets with 1,000s or 10s of thousands of things we are measuring. . .

PCA to the rescue

Let's see how PCA deals with this dataset. So main function in base R to do PCA is called `prcomp()`

```
#transpose t()
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
```

Let's see what is inside this `pca` object that we created from running `prcomp()`

```
attributes(pca)
```

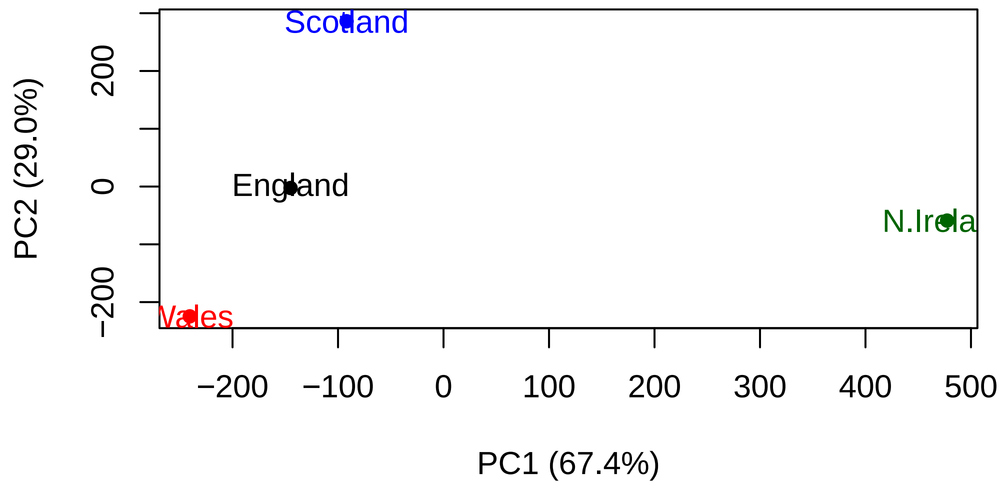
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

```
pca$x
```

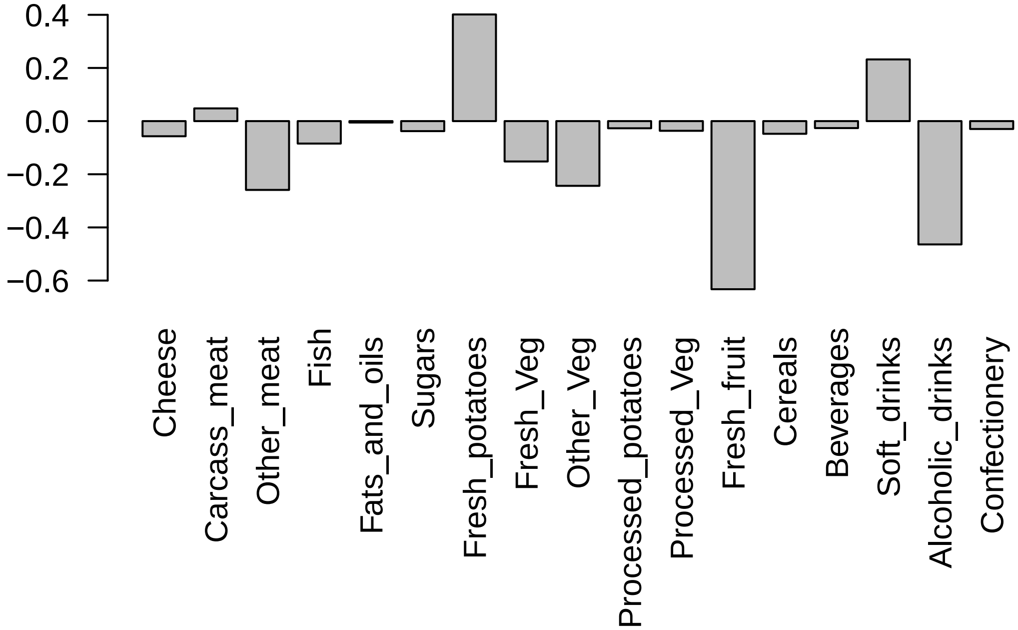
```
                 PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -9.152022e-15
Wales     -240.52915 -224.646925 -56.475555  5.560040e-13
Scotland   -91.86934  286.081786 -44.415495 -6.638419e-13
N.Ireland  477.39164  -58.901862  -4.877895  1.329771e-13
```

```
plot(pca$x[,1], pca$x[,2], col = c("black", "red", "blue", "darkgreen"), pch=16,
     xlab="PC1 (67.4%)", ylab = "PC2 (29.0%)")
text(pca$x[,1], pca$x[,2], labels=c("England", "Wales", "Scotland", "N.Ireland"), col=c("bla
```



Variable Loadings Plot

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


0.4
0.2
0.0
−0.2
−0.4
−0.6
Cheese
Carcass_meat
Other_meat
Fish
Fats_and_oils
Sugars
Fresh_potatoes
Fresh_Veg
Other_Veg
Processed_potatoes
Processed_Veg
Fresh_fruit
Cereals
Beverages
Soft_drinks
Alcoholic_drinks
Confectionery