Class 7: Machine Learning

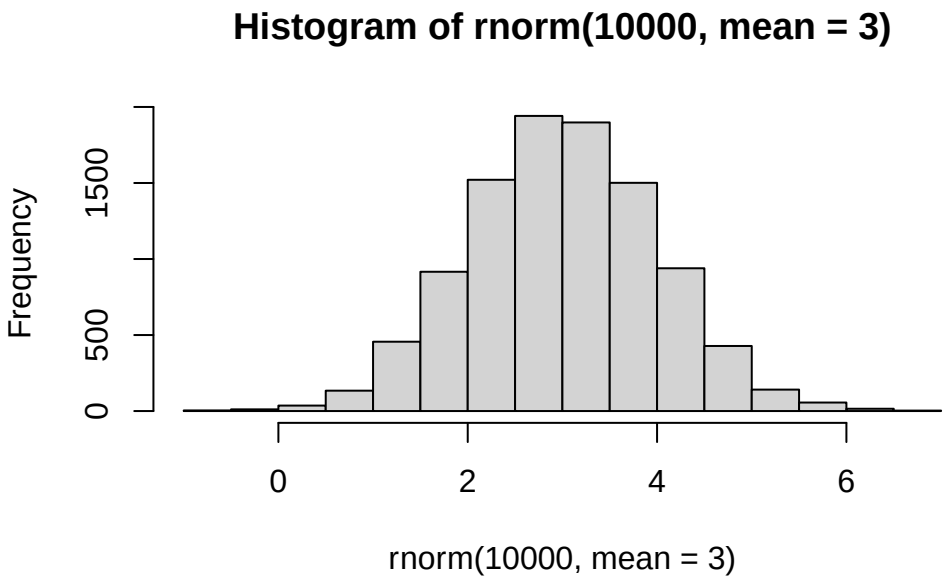
Clarissa Savko (PID:A69028482)

Clustering

We will start with k-means clustering, one of the most prevalent of all clustering methods.

To get started let's make some data up:

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

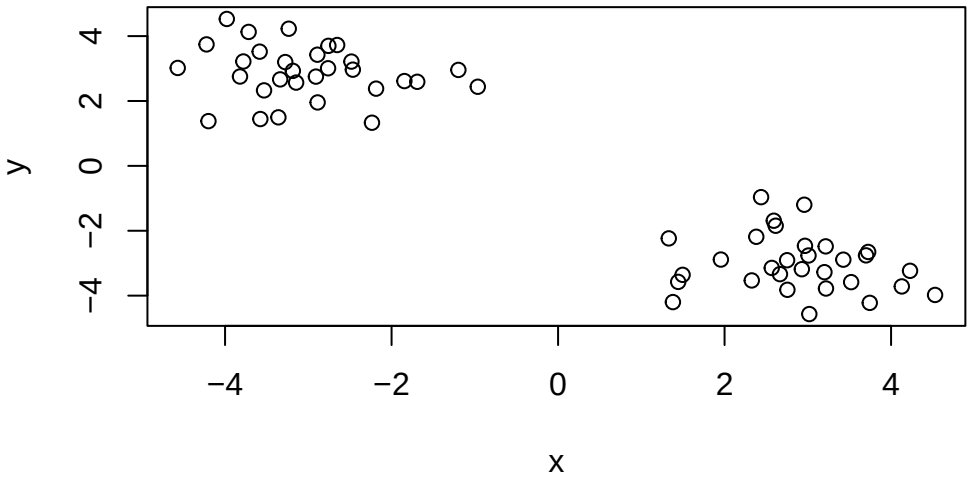



```
tmp <- c(rnorm(30,3), rnorm(30, -3))
x <- cbind(x=tmp, y=rev(tmp))
x
```

```
              x          y
 [1,]  1.4958160 -3.3594089
 [2,]  3.0082580 -2.7627842
 [3,]  3.6993091 -2.7587432
 [4,]  2.9655937 -2.4650073
 [5,]  3.4263601 -2.8910558
 [6,]  2.6138668 -1.8459580
 [7,]  2.7510725 -2.9088524
 [8,]  2.6652445 -3.3380447
 [9,]  4.5285474 -3.9791182
[10,]  3.7441210 -4.2227336
[11,]  2.3266375 -3.5316703
[12,]  1.3297407 -2.2354579
[13,]  2.4385820 -0.9642378
[14,]  2.5913512 -1.6915841
[15,]  2.5669169 -3.1463379
[16,]  2.9574285 -1.1976627
[17,]  3.5205653 -3.5843073
[18,]  2.9283063 -3.1857439
[19,]  1.9556387 -2.8881551
[20,]  1.3793510 -4.1995897
[21,]  2.7545735 -3.8205643
[22,]  1.4431668 -3.5761655
[23,]  3.2145836 -2.4823812
[24,]  3.7249236 -2.6537270
[25,]  3.1996649 -3.2771367
[26,]  3.2181173 -3.7805398
[27,]  4.1287390 -3.7168311
[28,]  2.3806214 -2.1859649
[29,]  3.0178963 -4.5686283
[30,]  4.2276222 -3.2352577
[31,] -3.2352577  4.2276222
[32,] -4.5686283  3.0178963
[33,] -2.1859649  2.3806214
[34,] -3.7168311  4.1287390
[35,] -3.7805398  3.2181173
[36,] -3.2771367  3.1996649
[37,] -2.6537270  3.7249236
[38,] -2.4823812  3.2145836
[39,] -3.5761655  1.4431668
[40,] -3.8205643  2.7545735
[41,] -4.1995897  1.3793510
[42,] -2.8881551  1.9556387
```

```
[43,] -3.1857439  2.9283063
[44,] -3.5843073  3.5205653
[45,] -1.1976627  2.9574285
[46,] -3.1463379  2.5669169
[47,] -1.6915841  2.5913512
[48,] -0.9642378  2.4385820
[49,] -2.2354579  1.3297407
[50,] -3.5316703  2.3266375
[51,] -4.2227336  3.7441210
[52,] -3.9791182  4.5285474
[53,] -3.3380447  2.6652445
[54,] -2.9088524  2.7510725
[55,] -1.8459580  2.6138668
[56,] -2.8910558  3.4263601
[57,] -2.4650073  2.9655937
[58,] -2.7587432  3.6993091
[59,] -2.7627842  3.0082580
[60,] -3.3594089  1.4958160
```

```
plot(x)
```



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

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

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

Cluster means:
          x         y
1  2.873421 -3.015122
2 -3.015122  2.873421

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 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

Within cluster sum of squares by cluster:
[1] 41.91874 41.91874
 (between_SS / total_SS =  92.5 %)

Available components:

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

Q1. How many points are in each cluster

```r
k$size
```

```
[1] 30 30
```

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

```r
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 2
```

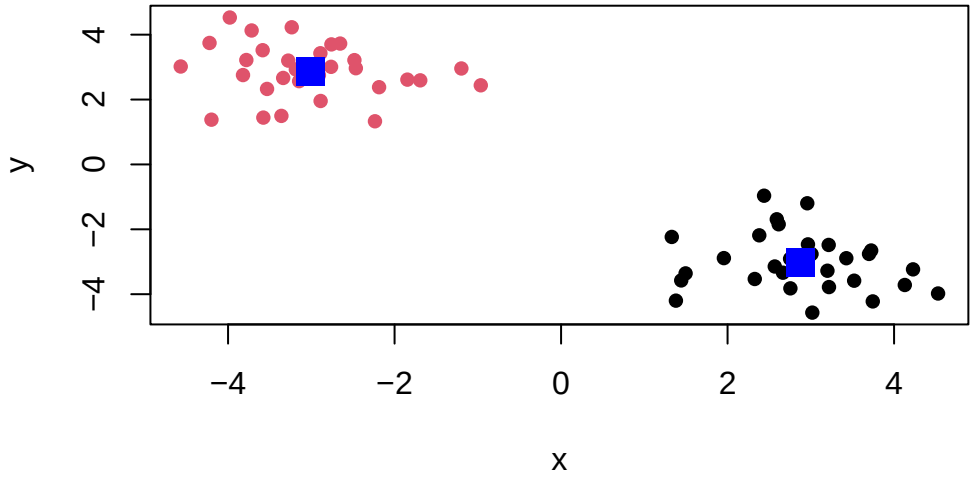
Q3. Cluster centers

```r
k$centers
```

```
          x         y
1  2.873421 -3.015122
2 -3.015122  2.873421
```

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)
```



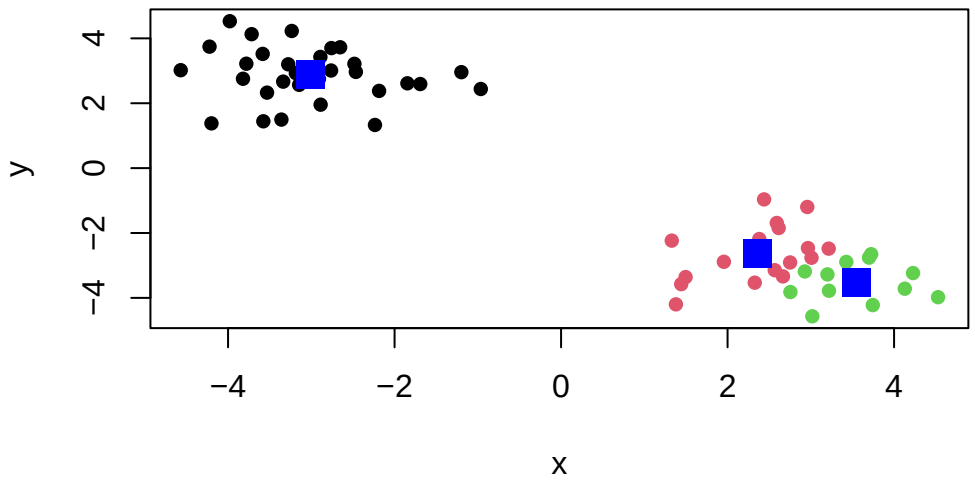
Q5. Run kmeans again except with 3 groups rather than 2 and plot the results.

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

```
K-means clustering with 3 clusters of sizes 30, 17, 13

Cluster means:
          x         y
1 -3.015122  2.873421
2  2.357875 -2.634074
```

```
3  3.547596 -3.513414

Clustering vector:
 [1] 2 2 3 2 3 2 2 2 3 3 2 2 2 2 2 2 3 3 2 2 3 2 2 3 3 3 3 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 1 1

Within cluster sum of squares by cluster:
[1] 41.918740 18.419880  7.375618
 (between_SS / total_SS =  94.0 %)

Available components:

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

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



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

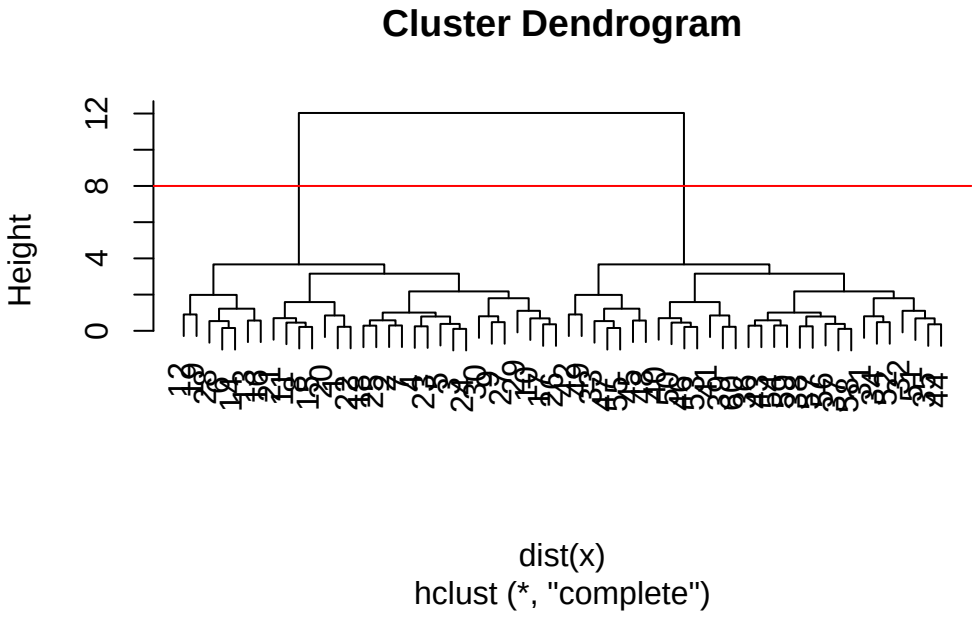
The main function for this in "base" R is called `hclust()`

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")
```

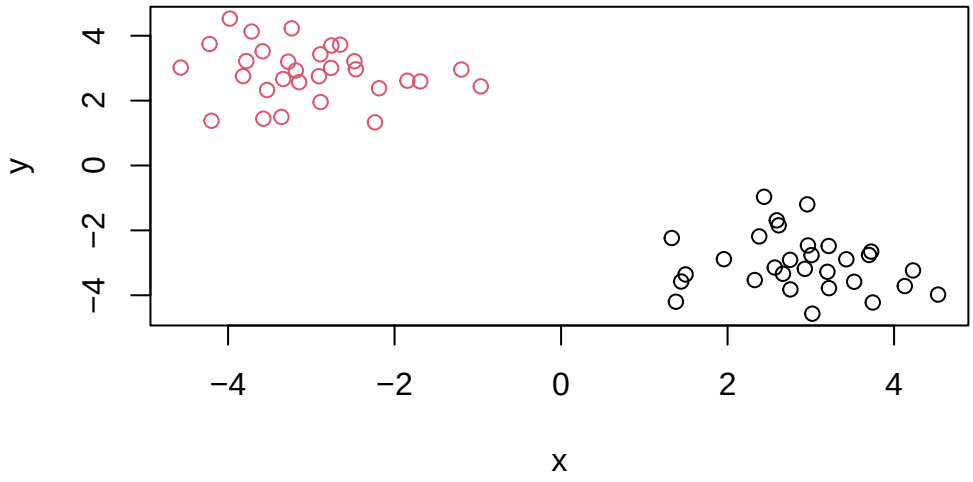


The function to get our clusters/groups from a hclust object is called `cutree()`

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

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

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



Principal Component Analysis (PCA)

We will work on data from the UK about the strange stuff folks there eat. It has 17 different foods for 4 countries.

```
url <- "https://tinyurl.com/UK-foods"
data <- read.csv(url)
head(data)
```

```
              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
```

This risks losing data if you run the code multiple times and lose columns.

```
rownames(data) <- data[,1]
data <- data[,-1]
head(data)
```

```
              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
```

```
data <- read.csv(url,row.names=1)
head(data)
```

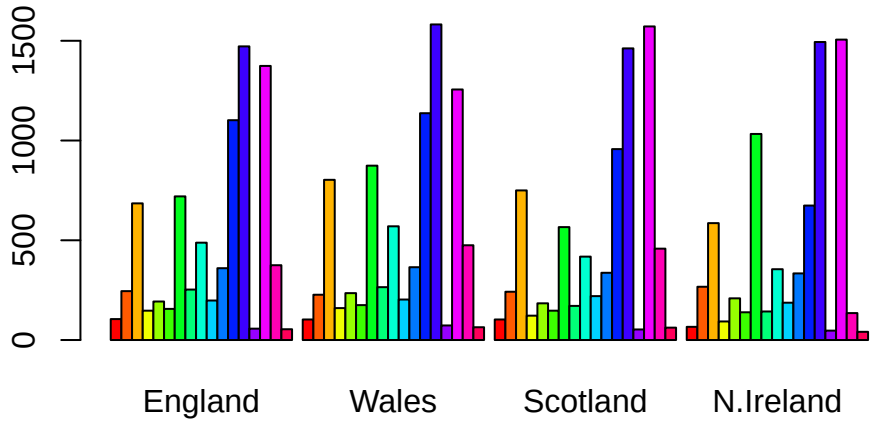
```
              England Wales Scotland N.Ireland
Cheese            105   103      103        66
Carcass_meat      245   227      242       267
Other_meat        685   803      750       586
Fish              147   160      122        93
Fats_and_oils     193   235      184       209
Sugars        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

```
dim(data)
```

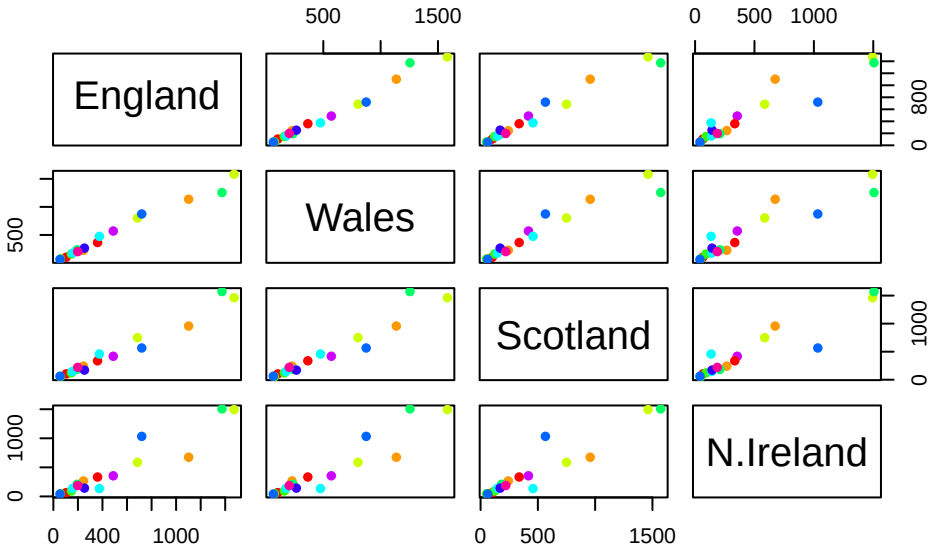
```
[1] 17  4
```

17 rows and 4 columns.

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




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



Most of the points for the graphs comparing to North Ireland are in the bottom left corner and then there are just two points in the top right corner.

PCA to the rescue

Help me make sense of this data. The main function for PCA in base R is called `prcomp`

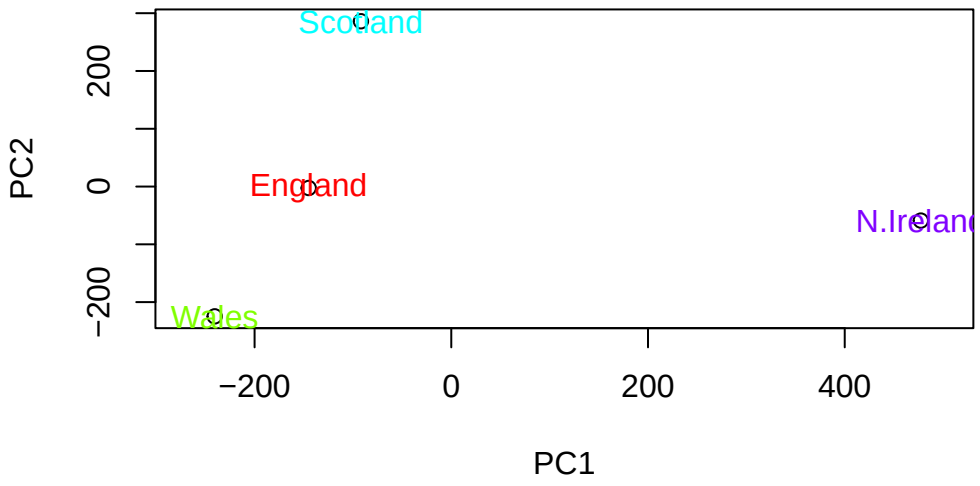
It wants the transpose (`t()`) of our food data for analysis.

```
pca <- prcomp(t(data))
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
```

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

```
plot(pca$x[,1], pca$x[,2],xlab="PC1", ylab="PC2", xlim=c(-270,500))
text(pca$x[,1], pca$x[,2], colnames(data), col=rainbow(4))
```



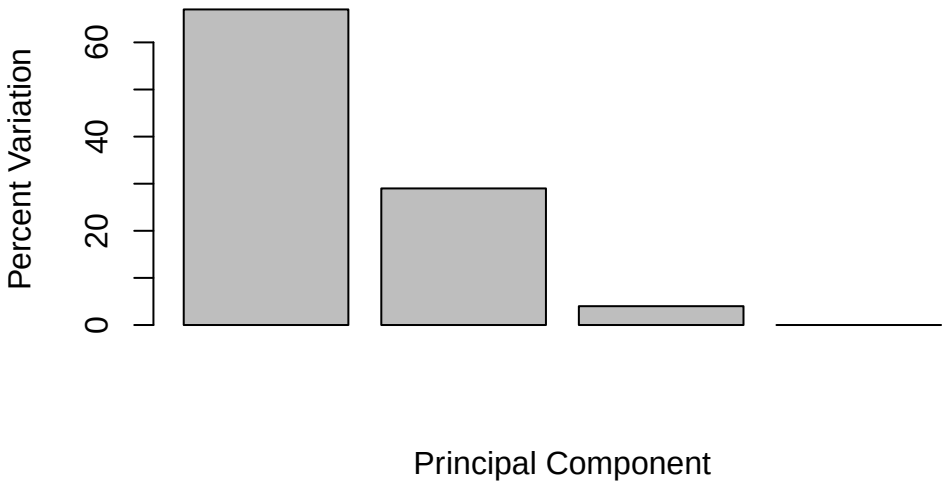

```
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

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

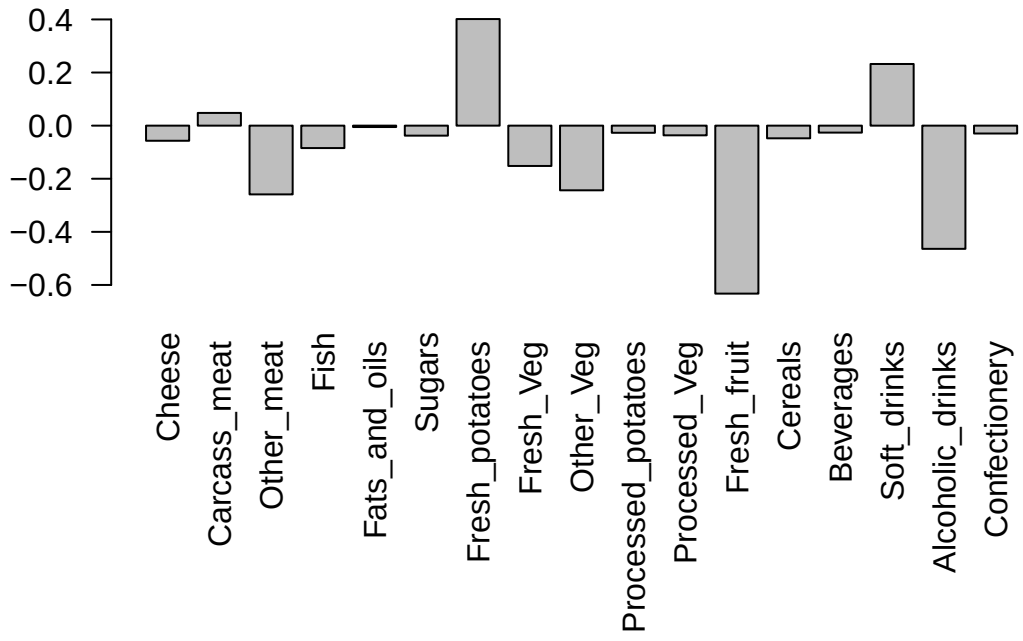
```
z <- summary(pca)
z$importance
```

```
                             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
```

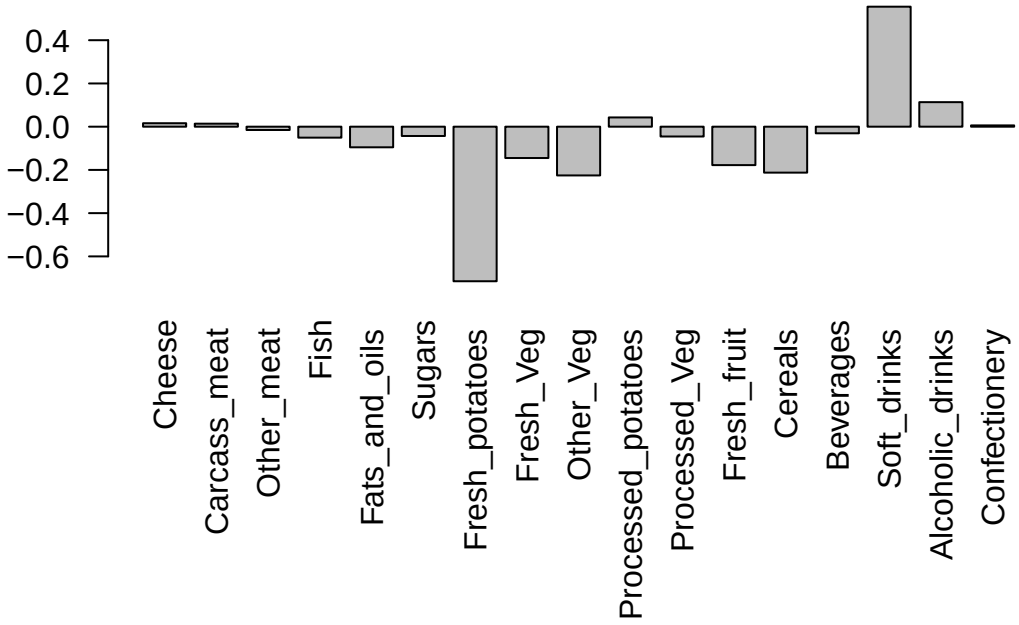
```
barplot(v, xlab="Principal Component", ylab="Percent Variation")
```




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



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

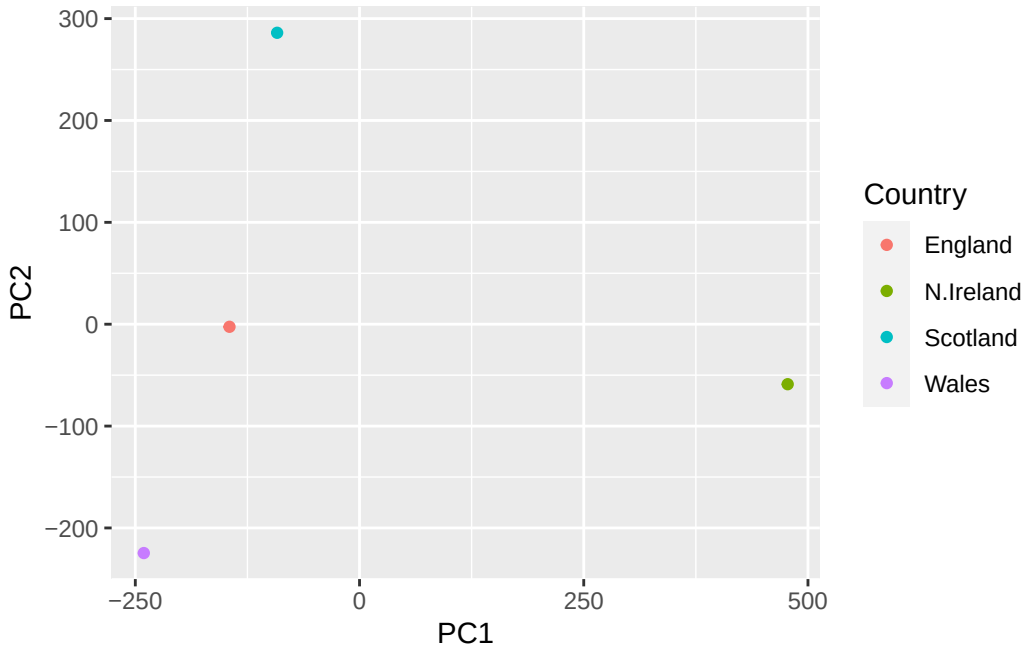



```
pca$rotation
```

	PC1	PC2	PC3	PC4
Cheese	-0.056955380	0.016012850	0.02394295	-0.694538519
Carcass_meat	0.047927628	0.013915823	0.06367111	0.489884628
Other_meat	-0.258916658	-0.015331138	-0.55384854	0.279023718
Fish	-0.084414983	-0.050754947	0.03906481	-0.008483145
Fats_and_oils	-0.005193623	-0.095388656	-0.12522257	0.076097502
Sugars	-0.037620983	-0.043021699	-0.03605745	0.034101334
Fresh_potatoes	0.401402060	-0.715017078	-0.20668248	-0.090972715
Fresh_Veg	-0.151849942	-0.144900268	0.21382237	-0.039901917
Other_Veg	-0.243593729	-0.225450923	-0.05332841	0.016719075
Processed_potatoes	-0.026886233	0.042850761	-0.07364902	0.030125166
Processed_Veg	-0.036488269	-0.045451802	0.05289191	-0.013969507
Fresh_fruit	-0.632640898	-0.177740743	0.40012865	0.184072217
Cereals	-0.047702858	-0.212599678	-0.35884921	0.191926714
Beverages	-0.026187756	-0.030560542	-0.04135860	0.004831876
Soft_drinks	0.232244140	0.555124311	-0.16942648	0.103508492

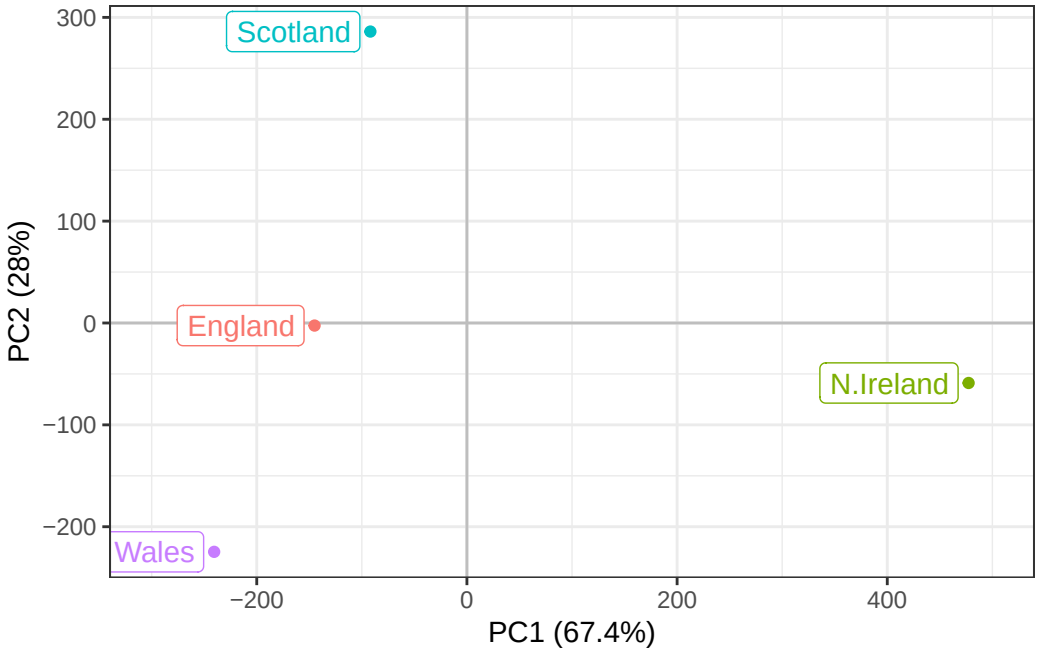
```
Alcoholic_drinks    -0.463968168  0.113536523 -0.49858320 -0.316290619
Confectionery       -0.029650201  0.005949921 -0.05232164  0.001847469
```

```
library(ggplot2)
df <- as.data.frame(pca$x)
df_lab <- tibble::rownames_to_column(df, "Country")

# Our first basic plot
ggplot(df_lab) +
  aes(PC1, PC2, col=Country) +
  geom_point()
```

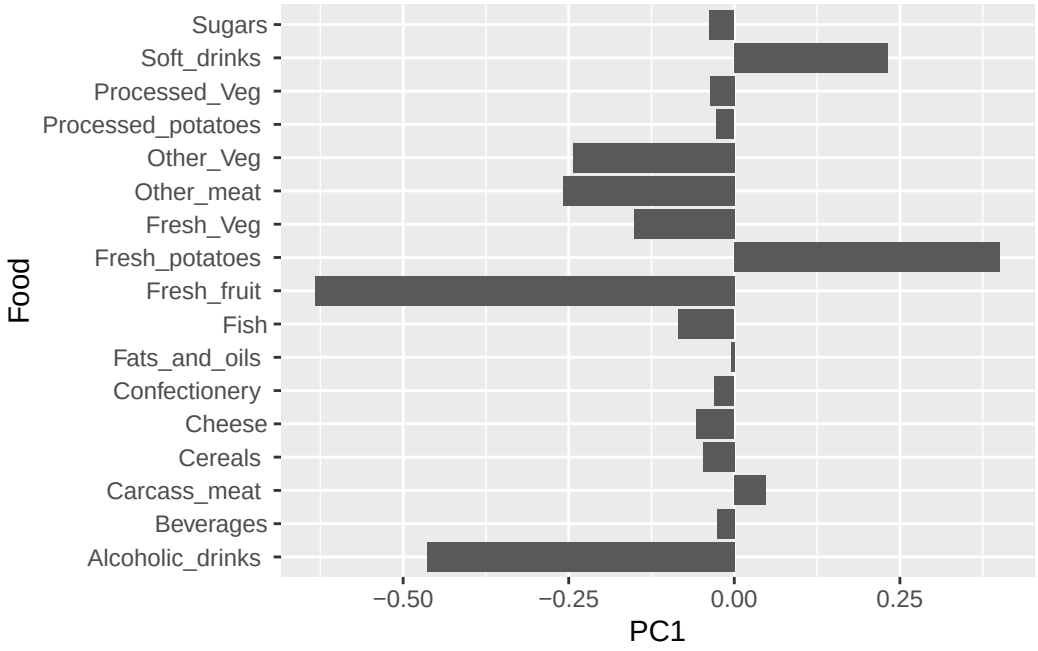



```
ggplot(df_lab) +
  aes(PC1, PC2, col=Country, label=Country) +
  geom_hline(yintercept = 0, col="gray") +
  geom_vline(xintercept = 0, col="gray") +
  geom_point(show.legend = FALSE) +
  geom_label(hjust=1, nudge_x = -10, show.legend = FALSE) +
  expand_limits(x = c(-300,500)) +
  xlab("PC1 (67.4%)") +
  ylab("PC2 (28%)") +
```

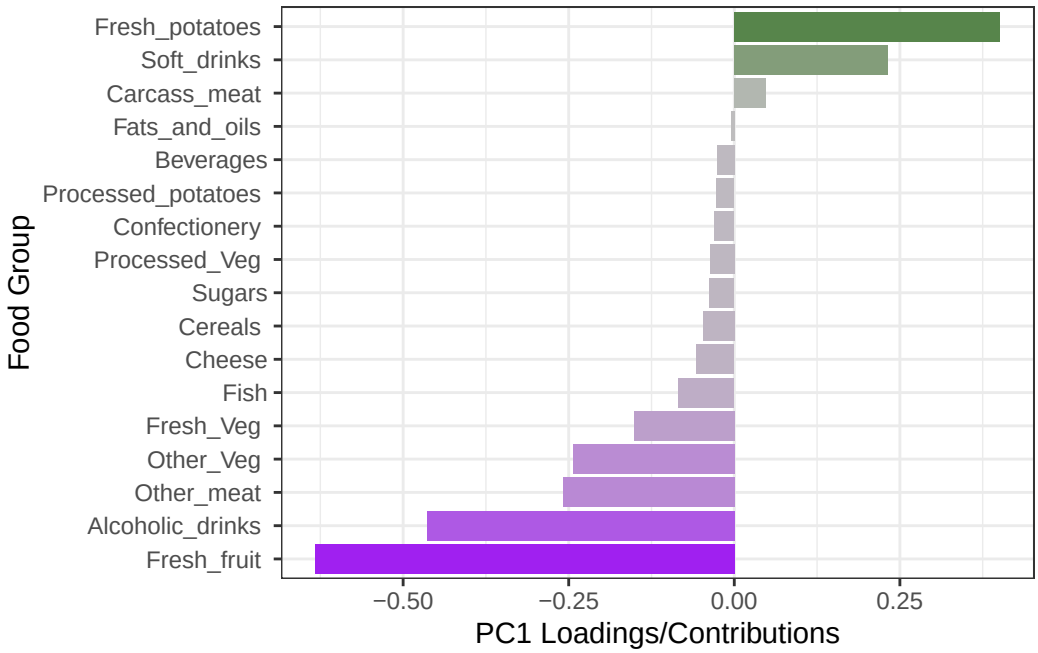
```
  theme_bw()
```



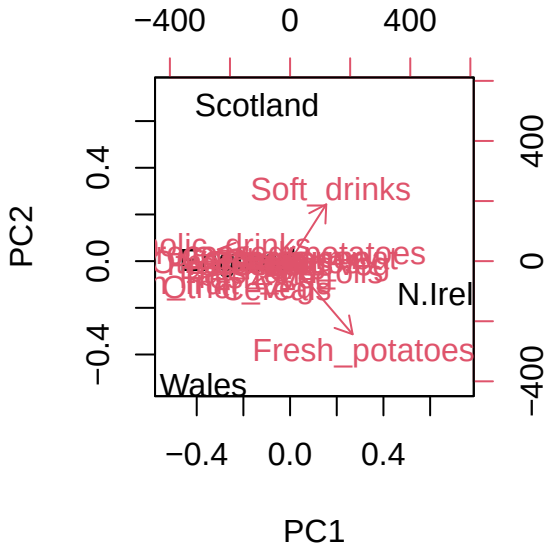

```
ld <- as.data.frame(pca$rotation)
ld_lab <- tibble::rownames_to_column(ld, "Food")

ggplot(ld_lab) +
  aes(PC1, Food) +
  geom_col()
```




```
ggplot(ld_lab) +
  aes(PC1, reorder(Food, PC1), bg=PC1) +
  geom_col() +
  xlab("PC1 Loadings/Contributions") +
  ylab("Food Group") +
  scale_fill_gradient2(low="purple", mid="gray", high="darkgreen", guide=NULL) +
  theme_bw()
```


Fresh_potatoes
Soft_drinks
Carcass_meat
Fats_and_oils
Beverages
Processed_potatoes
Confectionery
Processed_Veg
Sugars
Cereals
Cheese
Fish
Fresh_Veg
Other_Veg
Other_meat
Alcoholic_drinks
Fresh_fruit
Food Group
−0.50
−0.25
0.00
0.25
PC1 Loadings/Contributions

```
biplot(pca)
```


−400
0
400
Scotland
Soft_drinks
N.Irel
Fresh_potatoes
Wales
0.4
0.0
−0.4
PC2
400
0
−400
−0.4
0.0
0.4
PC1

```
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
```

```
dim(rna.data)
```

```
[1] 100  10
```

100 genes, 10 samples