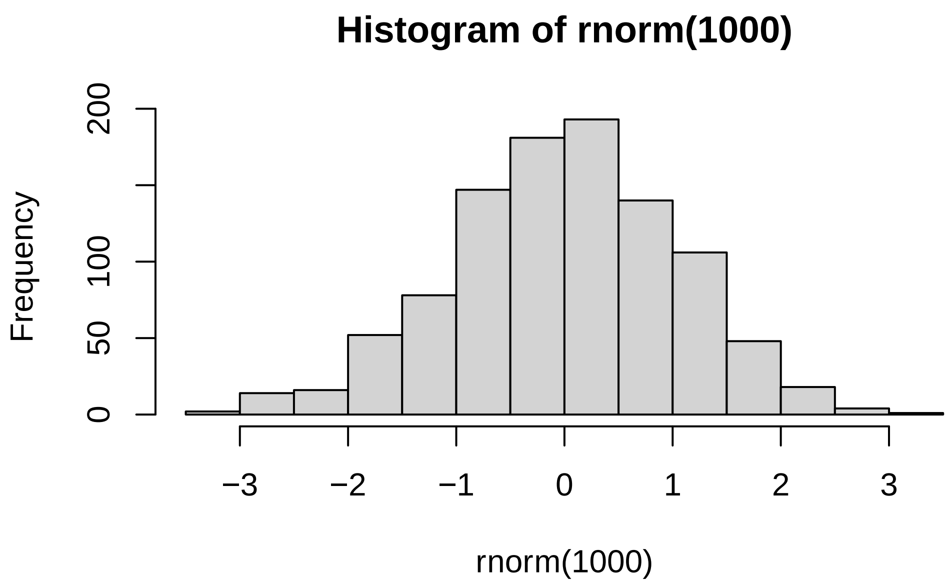
Class 7: Machine Learning

Courtney Anderson (PID:A69038035)

Today, we will begin our exploration of some”classical” means learning approaches. We will start with clustering:

Lets first make up some data cluster where we know what the answer should be.

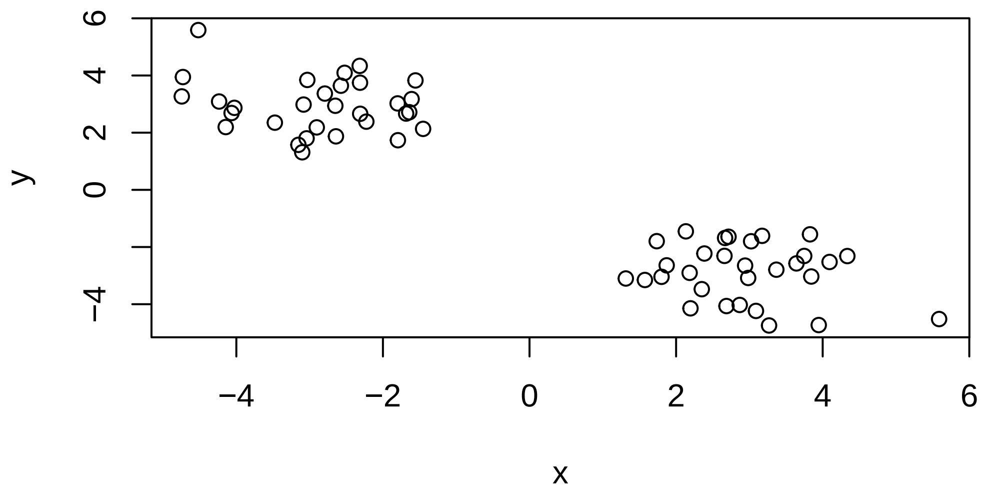
```
hist (rnorm(1000))
```



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

z <- cbind(x,y)
head(z)
```

```
            x         y
[1,] 1.801183 -3.041708
[2,] 2.941477 -2.649415
[3,] 2.184707 -2.902584
[4,] 3.826722 -1.556109
[5,] 3.946130 -4.729025
[6,] 2.715266 -1.640327
```

```
plot(z)
```



The main function in "base" R for K-means clustering is called `kmeans()`.

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

> Q. How big are the clusters?

```
k$size
```

```
[1] 30 30
```

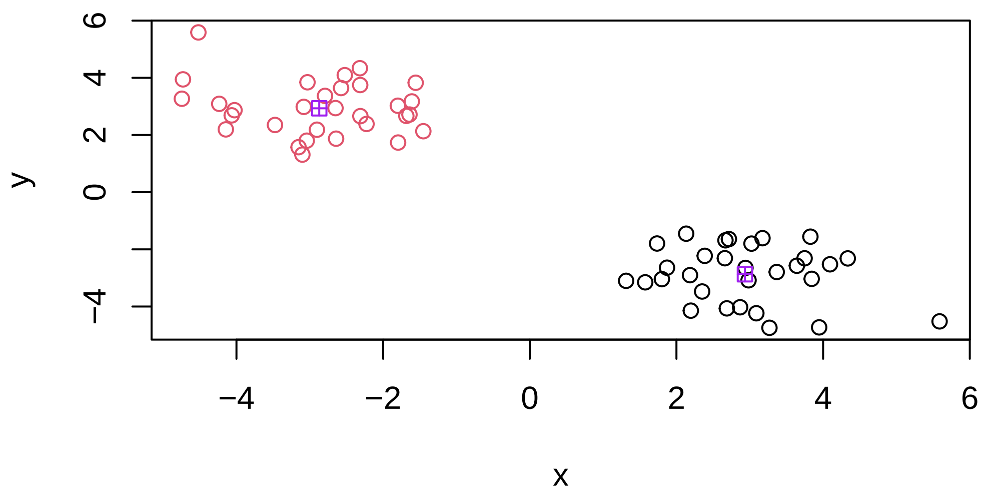
> Q. What clusters fo my data points reside in?

```
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 2 2
```

Q. can you make a plot of our data colored by cluster assigment i.e. Make a result figure

```
plot(z, col = k$cluster)
points(k$centers, col = "purple", pch = 12)
```



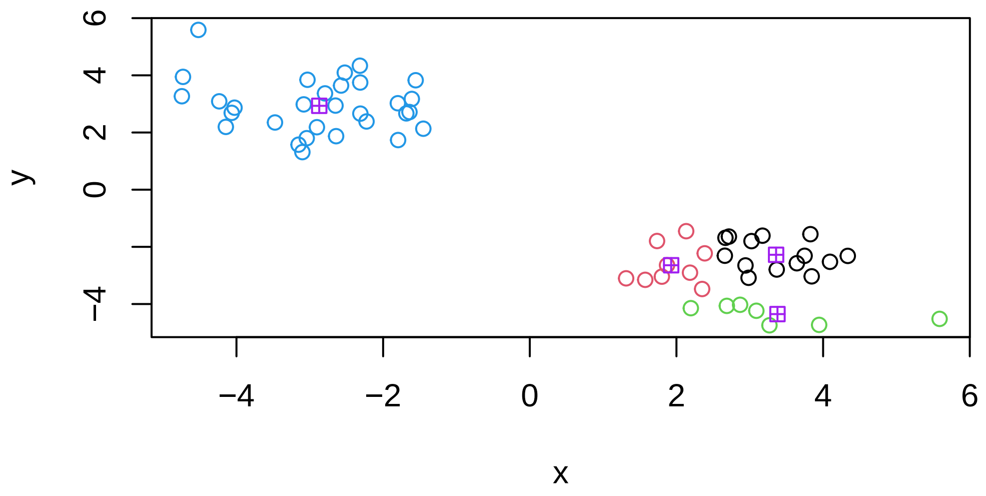
Q. Cluster with K-means into 4 clusters and plot your results above

```
k4 <- kmeans(z,centers = 4)
```

```
k4$cluster
```

```
 [1] 2 1 2 1 3 1 2 3 3 1 3 1 1 1 1 1 2 2 2 2 1 1 1 2 3 1 3 3 2 1 4 4 4 4 4 4 4 4
[39] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
```

```r
plot(z, col=k4$cluster)
points(k4$centers, col="purple", pch=12)
```



Run kmeans with centers (i.e. values of k) equal 1 to 6

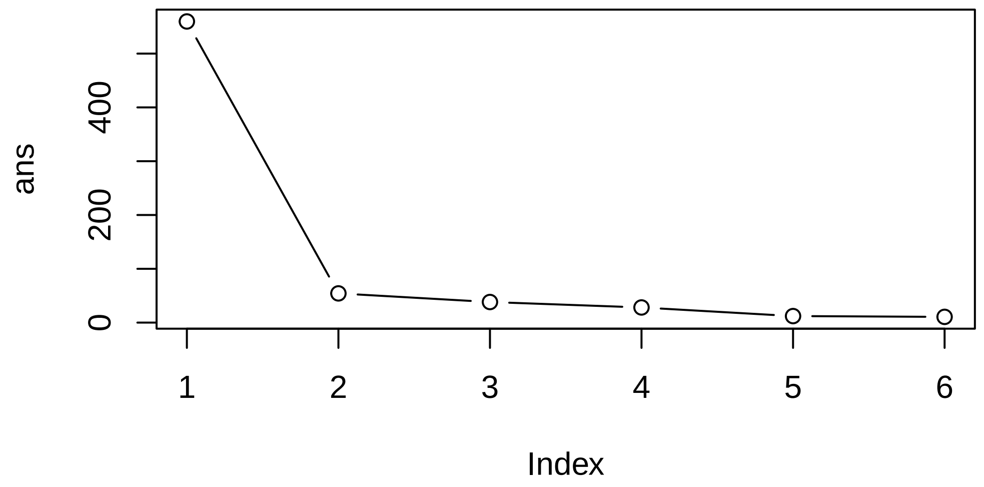
```r
k1 <- kmeans(x,centers=1)$tot.witness
k2 <- kmeans(x,centers=2)$tot.witness
k3 <- kmeans(x,centers=3)$tot.witness
k4 <- kmeans(x,centers=4)$tot.witness
k5 <- kmeans(x,centers=5)$tot.witness
k6 <- kmeans(x,centers=6)$tot.witness

ans <- c(k1, k2, k3, k4, k5, k6)
```

Or use a for loop (Make a scree-plot)

```r
ans <- NULL
for(i in 1:6){
  ans <- c(ans, kmeans(x, centers=i)$tot.withinss)
}
ans
```

```
[1] 559.73038  54.28315  38.27174  28.14872  12.13730  10.66589
```

```
plot(ans, typ="b")
```



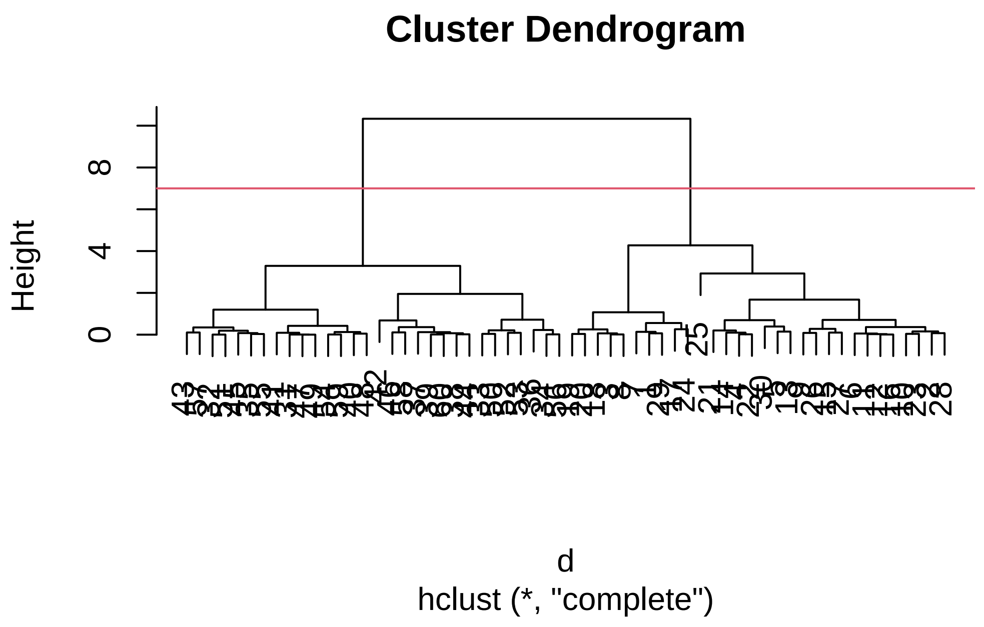
##Hierarchical Clustering

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

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

```

Call:
hclust(d = d)

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

```
plot(hc)
abline(h=7,col=2)
```



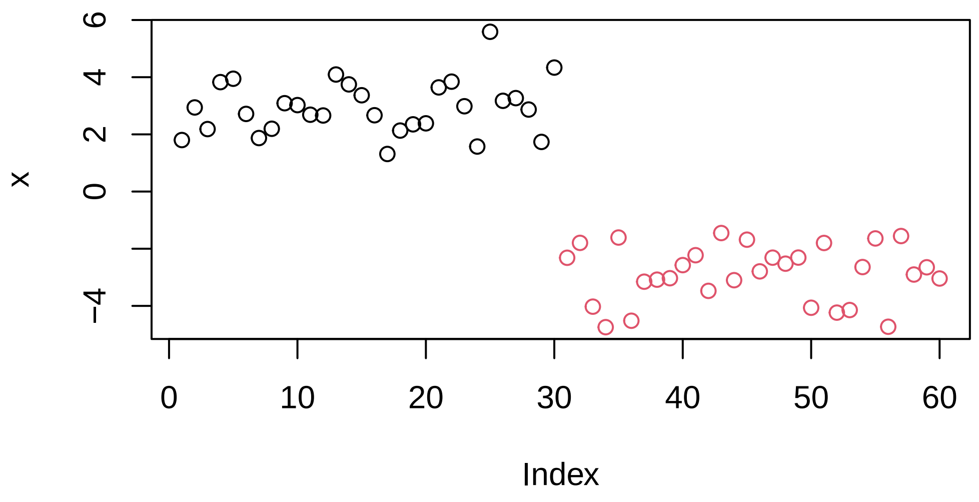
To obtain clusters from our `hclust` object **hc** we "cut" the tree into sub branches. For this we can use the `cutree()` function

```
grps <- cutree(hc, h=7)
grps
```

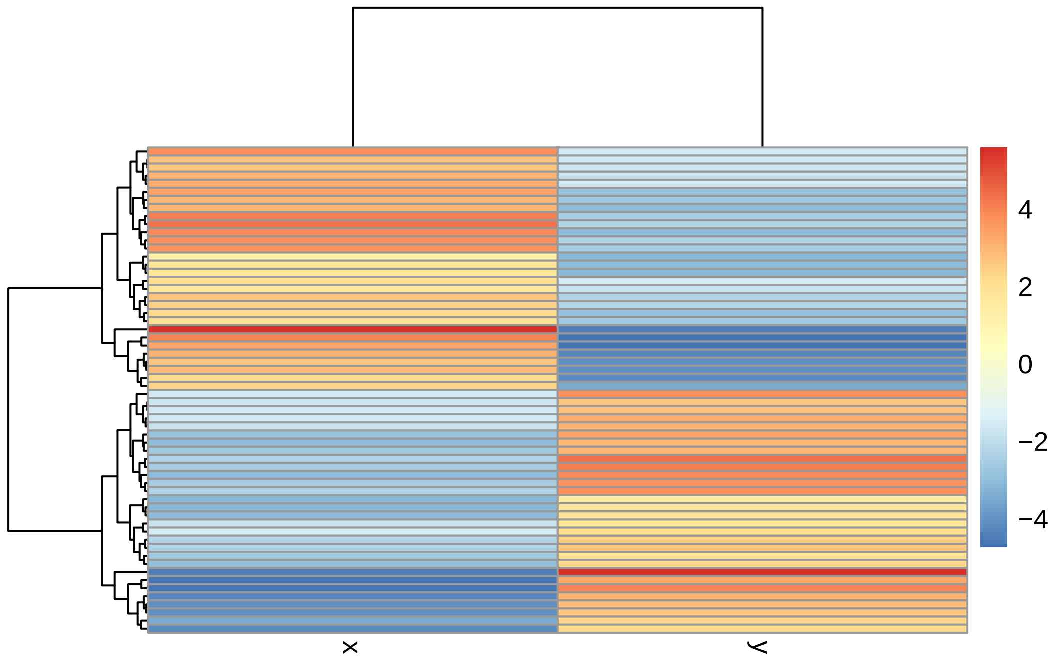
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

Results figure

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



```
library (pheatmap)
pheatmap(z)
```



Prinicipal Component Analysis (PCA)

PCA is a dimensional reduction, to take all things measuring and projects them on PC axes. PC1 is the "best fit" of the data, that maximizes the data spread/variance in data. It captures the most variance.PC2 captures the rest of the variance. PC looks to see the left/right and up/down variance easier to see. Think of a big funnel, and putting all your datta in it.

> Q1. How many rows and columns are in your new data frame named x? What R functions could you use to answer this questions?

There are 17 rows and 5 columns. You could use nrow and ncol.

```r
x <- read.csv(file = "UK_foods.csv")
```

```r
nrow(x)
```

```
[1] 17
```

```r
ncol(x)
```

```
[1] 5
```

Preview the first 6 rows

```r
head(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
```

Fix the row names

```r
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
```

I fixed the rownames again (I was having trouble with my file)

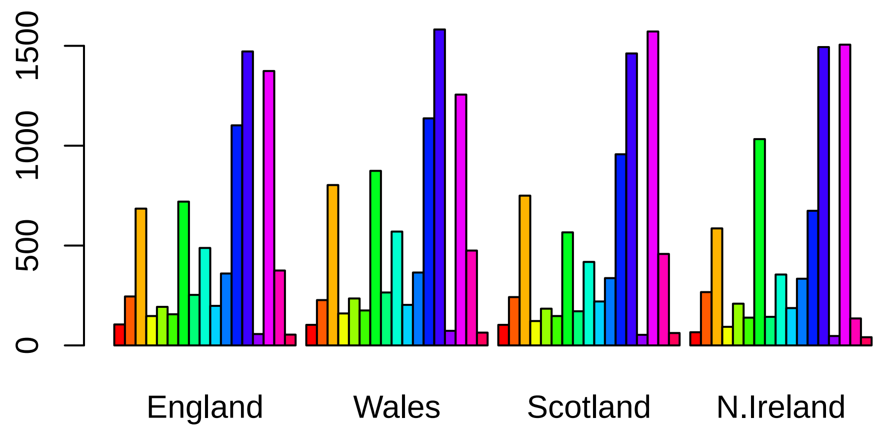
```
x <- read.csv("UK_foods.csv", row.names=1)
head(x)
```

```
               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
```

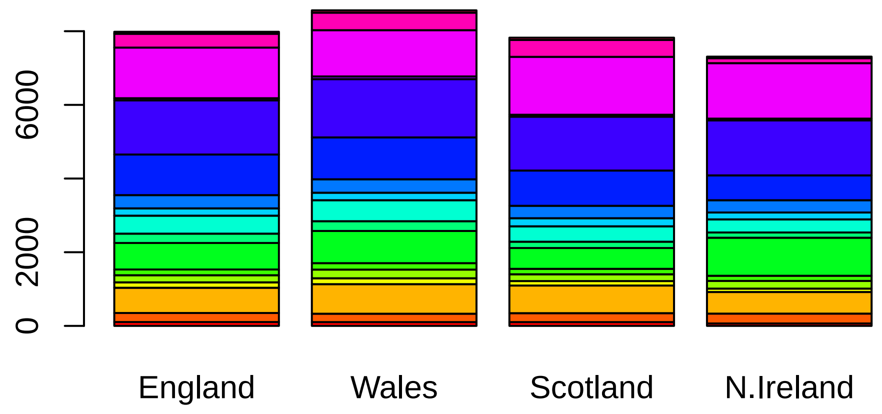
> Q2. Which approach to solving the 'row-names problem' mentioned above do you prefer and why? Is one approach more robust than another under certain circumstances?

I prefer the second method just because its quicker and requires less typing. (more)

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



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



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

Just change the beside to equal False or F

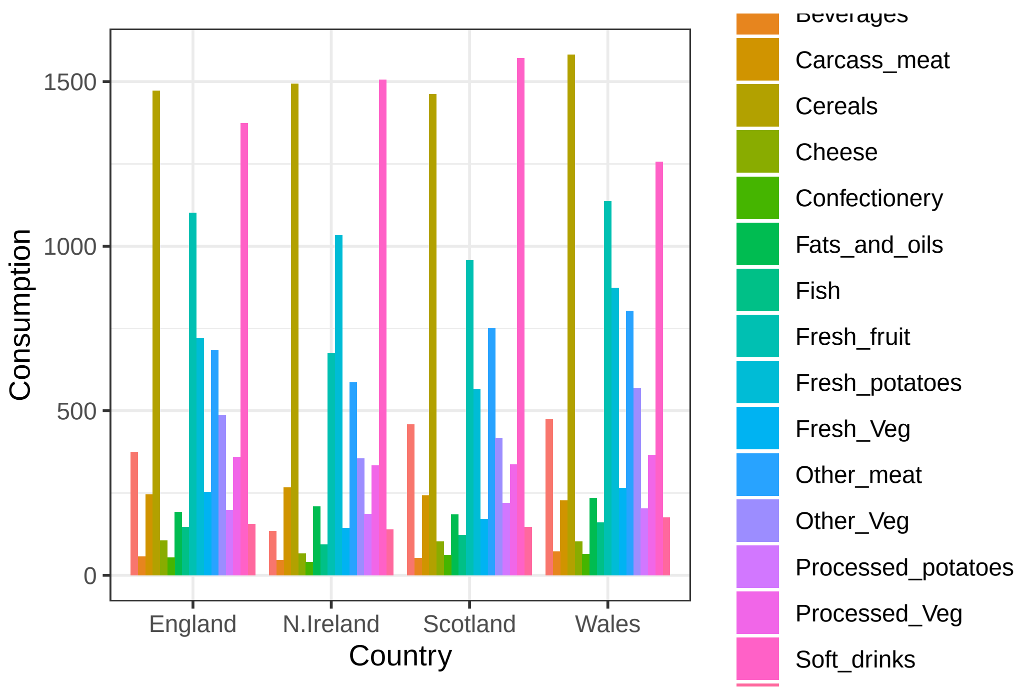
```
library(tidyr)
x_long <- x |>

tibble::rownames_to_column("Food") |>
pivot_longer(cols = -Food,
names_to = "Country",
values_to = "Consumption")

dim(x_long)
```

```
[1] 68  3
```

```
head(x_long)
```

```
# A tibble: 6 x 3
  Food               Country   Consumption
  <chr>              <chr>           <int>
1 "Cheese"           England           105
2 "Cheese"           Wales             103
3 "Cheese"           Scotland          103
4 "Cheese"           N.Ireland          66
5 "Carcass_meat "    England           245
6 "Carcass_meat "    Wales             227
```

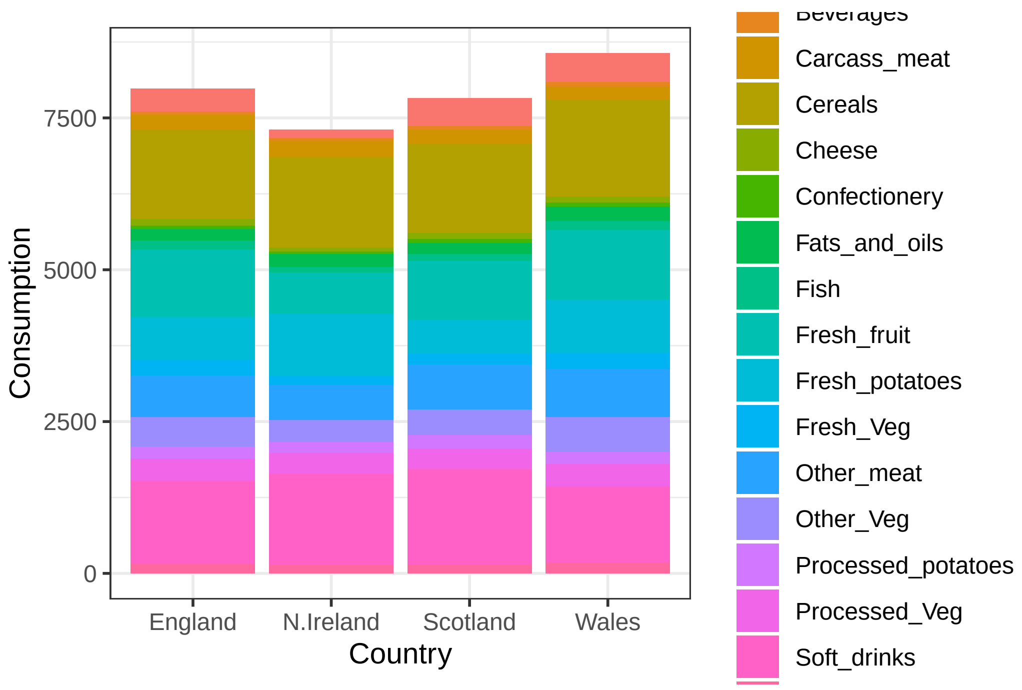
```
library(ggplot2)
ggplot(x_long) +
aes(Country, Consumption, fill = Food) +
geom_col(position = "dodge") +
theme_bw()
```



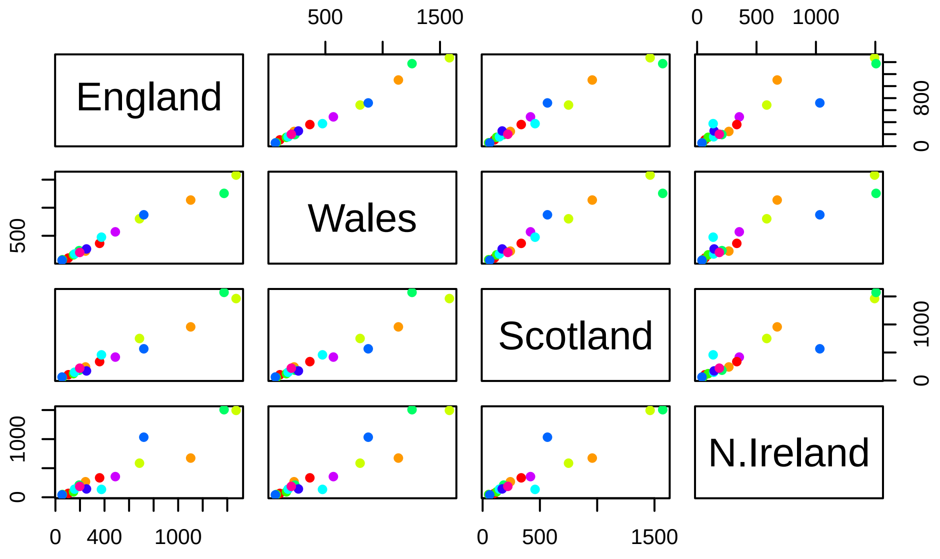
> Q4: Changing what optional argument in the above barplot() function results in the following plot?

Change the geom_col(position = “dodge”) to just geom_col().

```
ggplot(x_long) +
aes(Country, Consumption, fill = Food) +
geom_col() +
theme_bw()
```



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

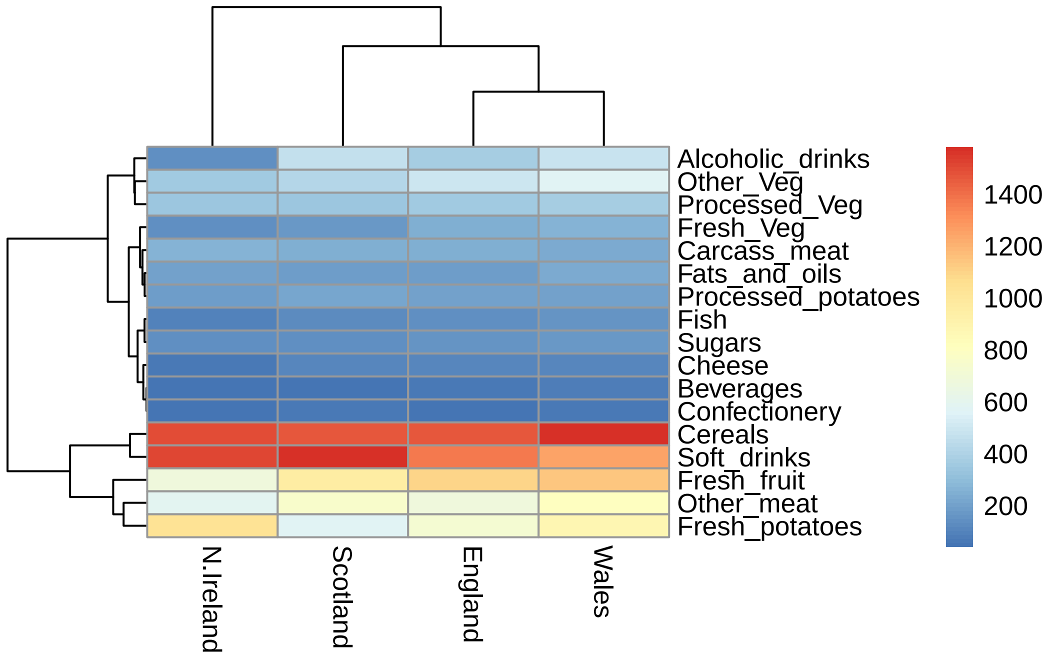


Q5: Generating all pairwise plots may help somewhat. Can you make sense of the

> following code and resulting figure? What does it mean if a given point lies on the diagonal for a given plot?

The points on the plots are specific types of food. If a point lies on the diagonal then that means that the food level is the same in both countries. If one point lies outside of the diagonal then that means that the food is being consumed at different levels between the two countries.

```r
pheatmap( as.matrix(x) )
```



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

Ireland appears to consume more potatoes than the other countries. They also consume less fresh fruit and other meats compared to other countries. Finally they have high consumption of soft drinks and cereals (but these values are more comparable).

##PCA to the rescue

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

The main function in "base" R for PCA is called `prcomp()`. As we want to do PCA on the food data, for the different countries, we will want the foods in the columns.

```r
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
```

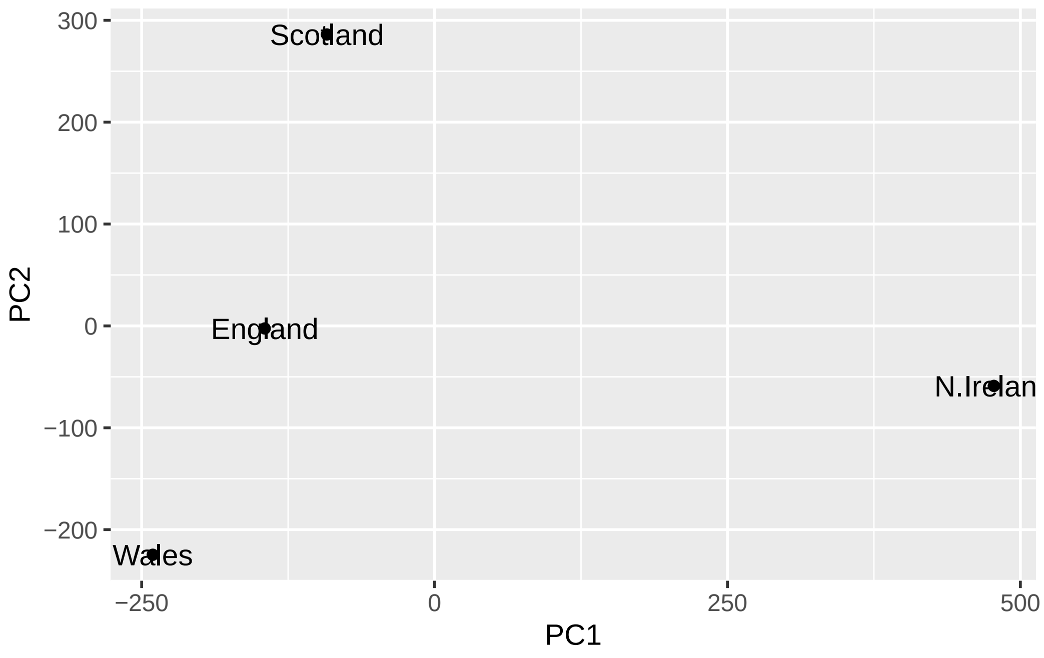
```r
pca <- prcomp( t(x) )
summary(pca)
```

```
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
```

```r
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
```

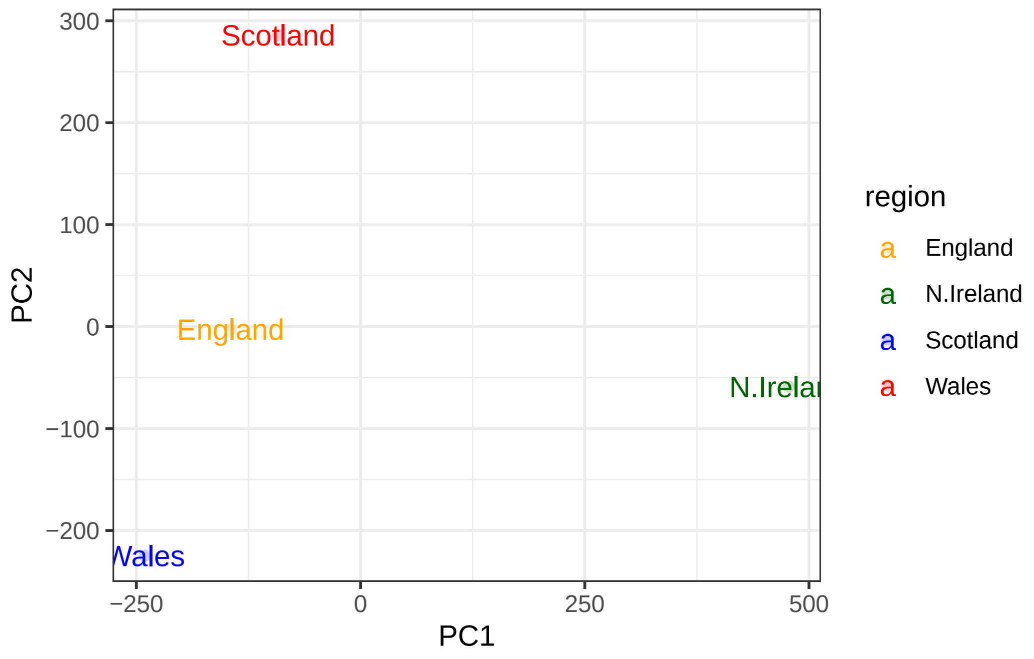
```r
library(ggplot2)

ggplot(pca$x) +
  aes(PC1, PC2, label = rownames(pca$x)) +
  geom_point() +
  geom_text()
```



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.

```r
library(ggplot2)

# Example: add a grouping column (one color per region)
regions <- data.frame(
  name = rownames(pca$x),
  region = c("England", "Scotland", "Wales", "N.Ireland")
)

# Combine with PCA results
df <- cbind(pca$x, regions)

# Plot with custom colors
```

```r
ggplot(df, aes(PC1, PC2, label = name, color = region)) +
  geom_point(alpha=0.2) +
  geom_text() +
  scale_color_manual(values = c(
    "England" = "orange",
    "Scotland" = "blue",
    "Wales" = "red",
    "N.Ireland" = "darkgreen"
  )) +
  theme_bw() +
  labs(x = "PC1", y = "PC2")
```



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

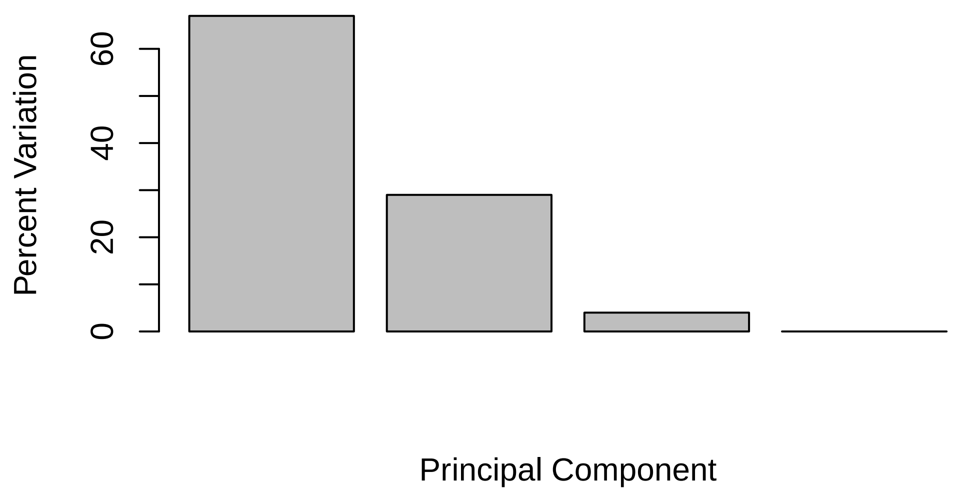
```
[1] 67 29  4  0
```

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

```
                             PC1       PC2      PC3          PC4
```

```
Standard deviation     324.15019 212.74780 73.87622 2.921348e-14
Proportion of Variance   0.67444   0.29052  0.03503 0.000000e+00
Cumulative Proportion    0.67444   0.96497  1.00000 1.000000e+00
```

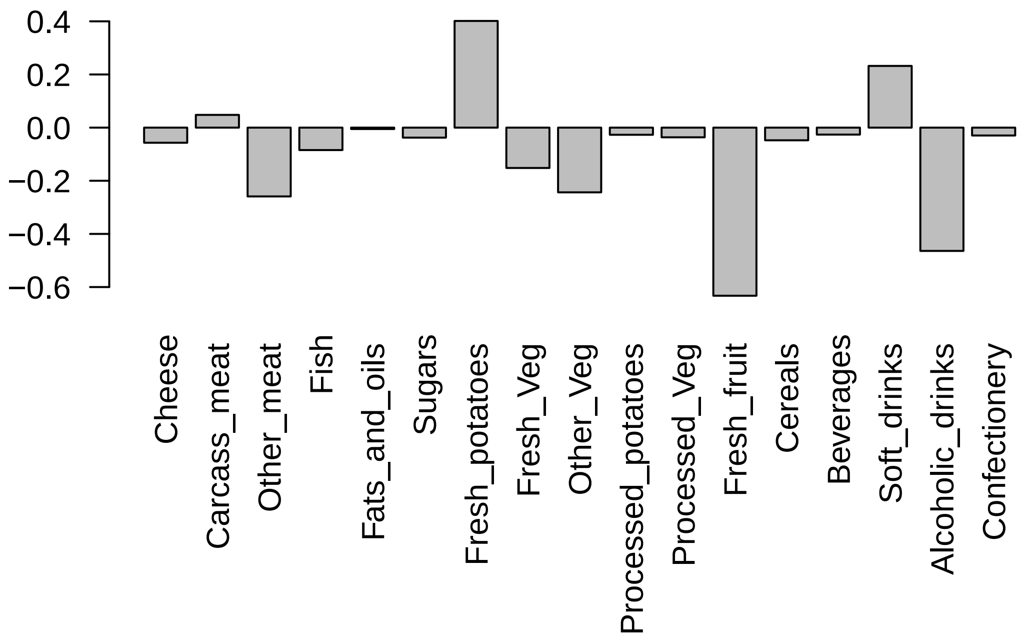
```
barplot(v, xlab="Principal Component", ylab="Percent Variation")
```



##Digging Deeper(variable loadings)

Lets focus on PC1 as it accounts for $>$ 90% of variance

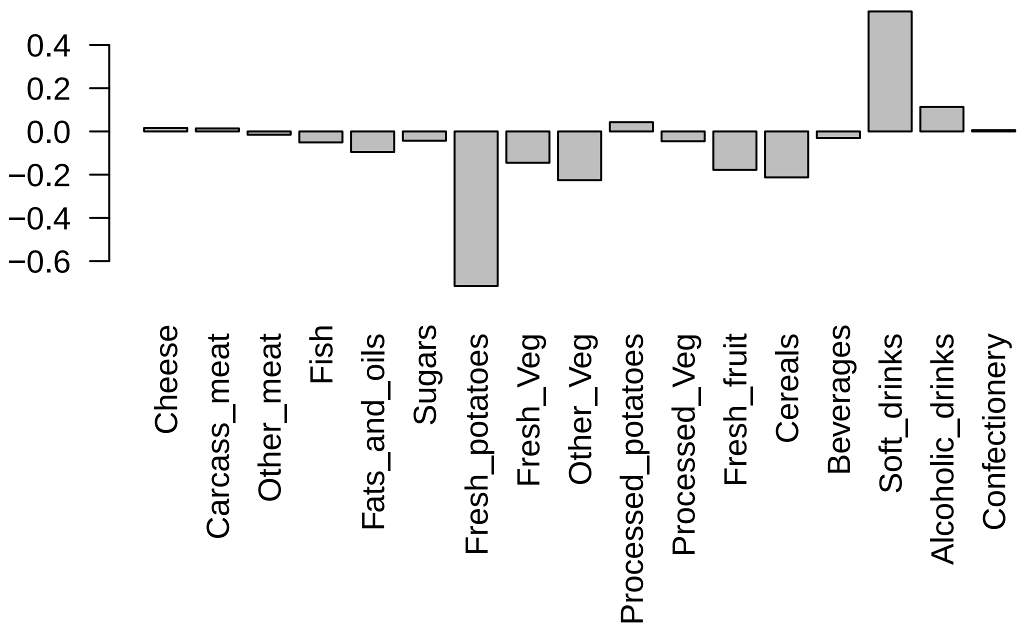
```
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,1], las=2 )
```



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

Fresh potatoes and soft drinks are featured prominently. PC2 mainly tells us the secondary differences in the data.

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



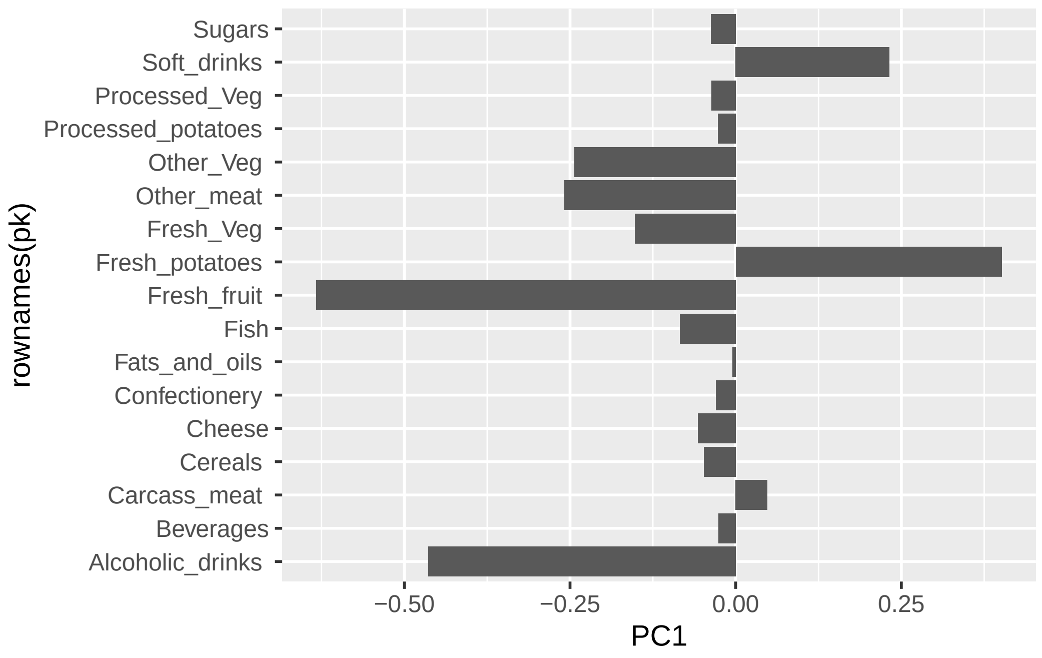
Another major result of PCA is the so-called "variable loadings" or `$rotation` that tells us how the origional variables (foods) contribute to PCs (i.e. our new axis).

```
pk <- pca$rotation
pk
```

```
                            PC1          PC2         PC3          PC4
Cheese             -0.056955380  0.016012850  0.02394295 -0.409382587
Carcass_meat        0.047927628  0.013915823  0.06367111  0.729481922
Other_meat         -0.258916658 -0.015331138 -0.55384854  0.331001134
Fish               -0.084414983 -0.050754947  0.03906481  0.022375878
Fats_and_oils      -0.005193623 -0.095388656 -0.12522257  0.034512161
Sugars             -0.037620983 -0.043021699 -0.03605745  0.024943337
Fresh_potatoes      0.401402060 -0.715017078 -0.20668248  0.021396007
Fresh_Veg          -0.151849942 -0.144900268  0.21382237  0.001606882
Other_Veg          -0.243593729 -0.225450923 -0.05332841  0.031153231
Processed_potatoes -0.026886233  0.042850761 -0.07364902 -0.017379680
Processed_Veg      -0.036488269 -0.045451802  0.05289191  0.021250980
Fresh_fruit        -0.632640898 -0.177740743  0.40012865  0.227657348
Cereals            -0.047702858 -0.212599678 -0.35884921  0.100043319
Beverages          -0.026187756 -0.030560542 -0.04135860 -0.018382072
Soft_drinks         0.232244140  0.555124311 -0.16942648  0.222319484
Alcoholic_drinks   -0.463968168  0.113536523 -0.49858320 -0.273126013
```

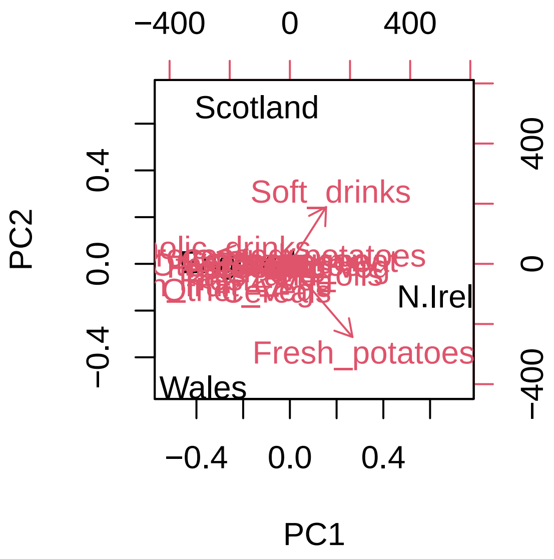
```
Confectionery       -0.029650201  0.005949921 -0.05232164  0.001890737
```

```
ggplot(pk) + aes(PC1, rownames(pk)) + geom_col()
```



##Biplots

```
biplot(pca)
```



##PCA or RNA seq data

```r
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
```

```r
nrow(rna.data)
```

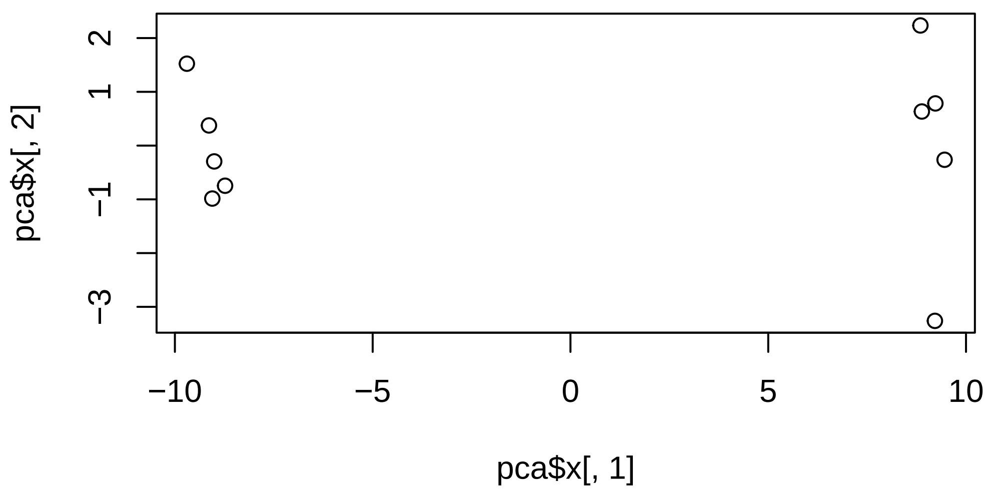
```
[1] 100
```

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

There are 100 genes.

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

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

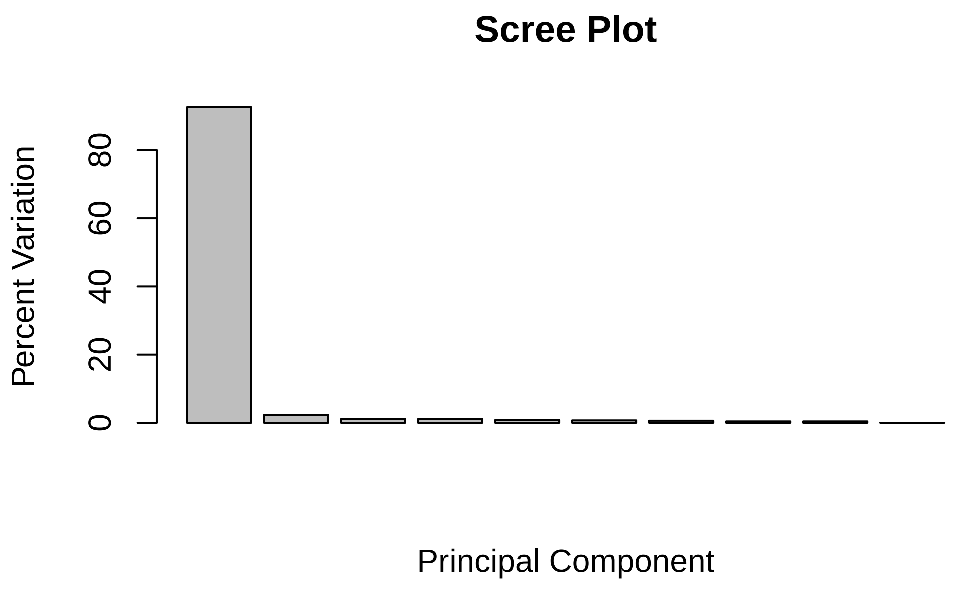


```
pca.var <- pca$sdev^2
```

```
pca.var.per <- round(pca.var/sum(pca.var)*100, 1)
pca.var.per
```

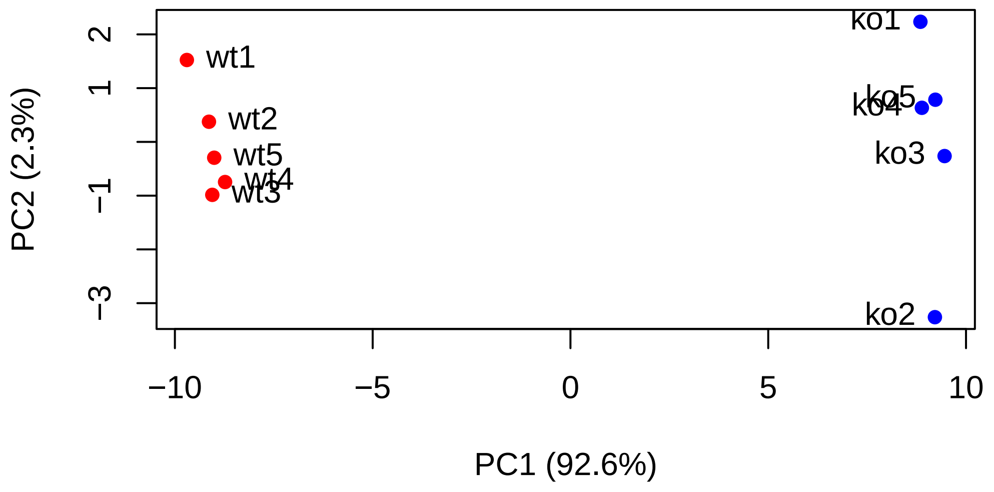
```
 [1] 92.6  2.3  1.1  1.1  0.8  0.7  0.6  0.4  0.4  0.0
```

```
barplot(pca.var.per, main="Scree Plot",
        xlab="Principal Component", ylab="Percent Variation")
```



A vector of colors for wt and ko samples

```
colvec <- colnames(rna.data)
colvec[grep("wt", colvec)] <- "red"
colvec[grep("ko", colvec)] <- "blue"

plot(pca$x[,1], pca$x[,2], col=colvec, pch=16,
     xlab=paste0("PC1 (", pca.var.per[1], "%)"),
     ylab=paste0("PC2 (", pca.var.per[2], "%)"))

text(pca$x[,1], pca$x[,2], labels = colnames(rna.data), pos=c(rep(4,5), rep(2,5)))
```



Another way to color by sample type

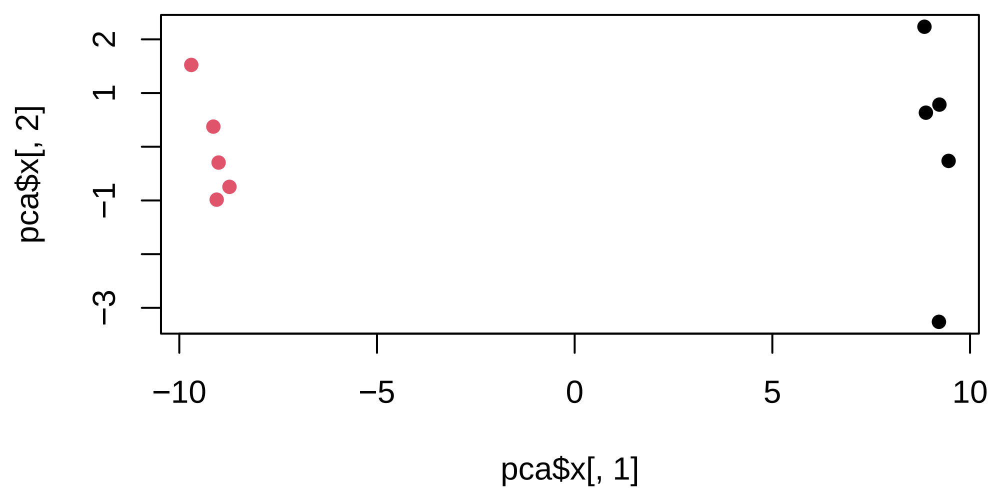
Extract the first 2 characters of the sample name

```
sample.type <- substr(colnames(rna.data),1,2)
sample.type
```

```
 [1] "wt" "wt" "wt" "wt" "wt" "ko" "ko" "ko" "ko" "ko"
```

now use this as a factor input to color our plot

```
plot(pca$x[,1], pca$x[,2], col=as.factor(sample.type), pch=16)
```



Find the top 10 measurements (genes) that contribute

most to PC1 in either direction (+ or -)

```
loading_scores <- pca$rotation[,1]
gene_scores <- abs(loading_scores)
gene_score_ranked <- sort(gene_scores, decreasing=TRUE)
```

Show the top ten genes

```
top_10_genes <- names(gene_score_ranked[1:10])
top_10_genes
```

```
 [1] "gene100" "gene66"  "gene45"  "gene68"  "gene98"  "gene60"  "gene21"
 [8] "gene56"  "gene10"  "gene90"
```