Machine Learning 1

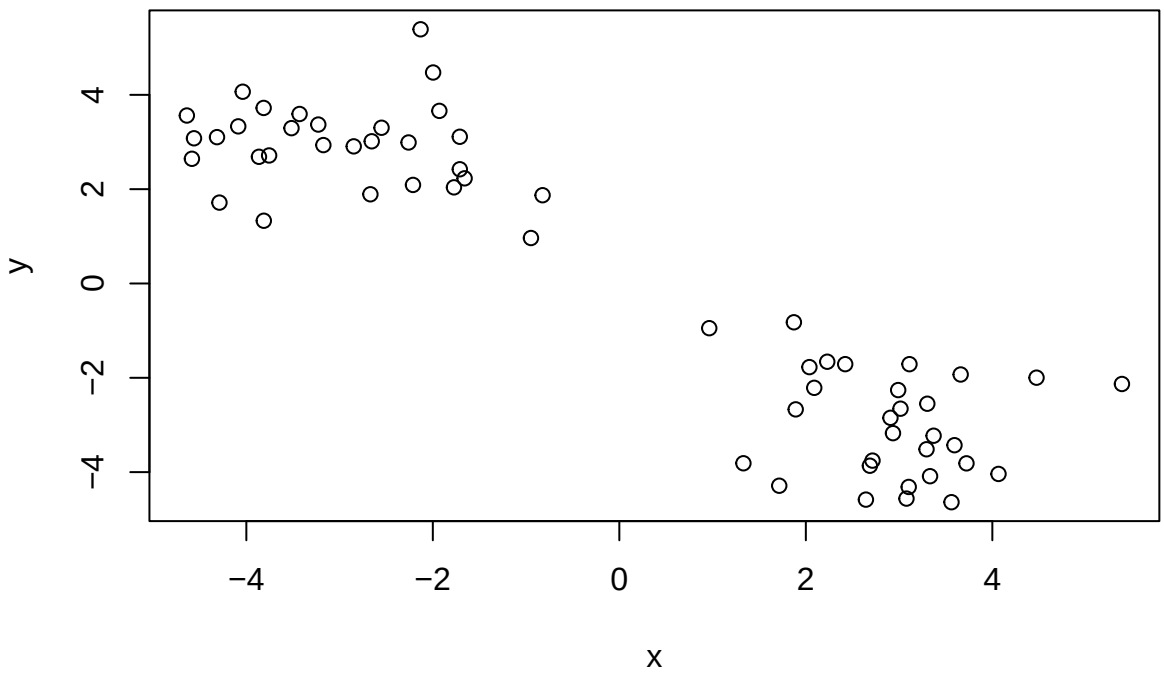
Katie Chau

2/14/2022

First up kmeans()

Demo of using kmeans() function in base R. First make up some data with a known structure.

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



Now we have some made up data in 'x' let's see how kmeans works with this data.

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

```
## K-means clustering with 2 clusters of sizes 30, 30
##
## Cluster means:
##           x         y
## 1 -2.966186  2.916812
## 2  2.916812 -2.966186
##
## 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 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
##
## Within cluster sum of squares by cluster:
## [1] 61.49557 61.49557
##  (between_SS / total_SS =  89.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Q. How many points are in each cluster?

```
k$size
```

```
## [1] 30 30
```

Q. How do we get to the cluster membership/assignment?

```
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 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
```

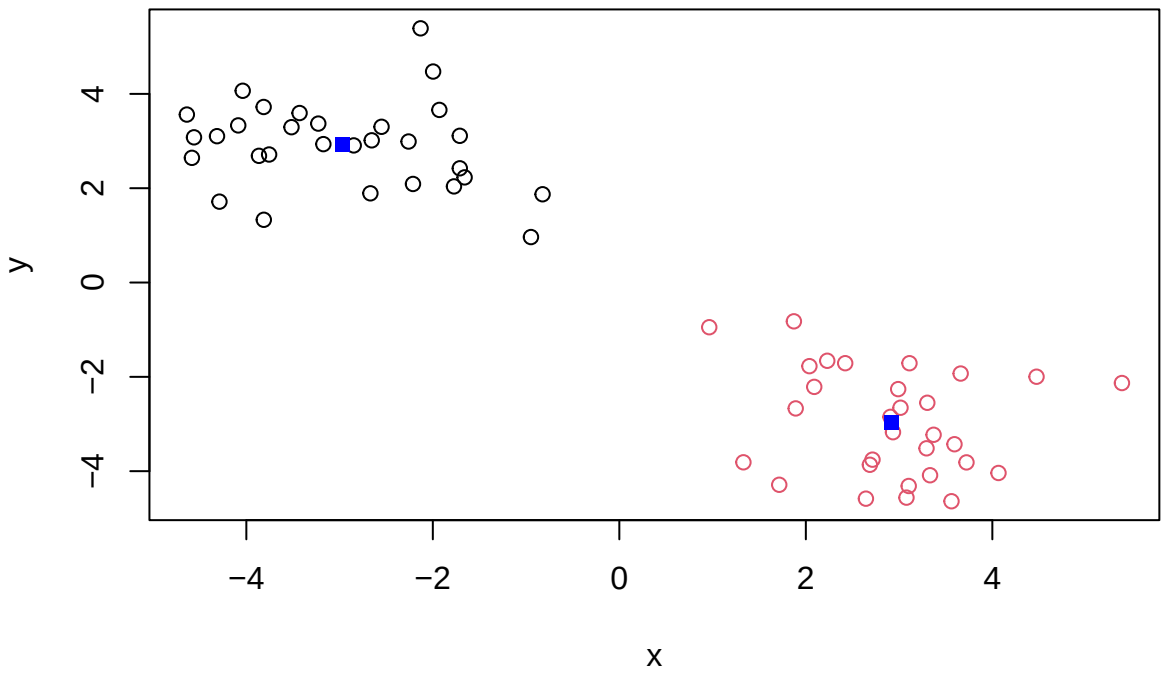
Q. What about cluster centers?

```
k$centers
```

```
##           x         y
## 1 -2.966186  2.916812
## 2  2.916812 -2.966186
```

Now we've got to main results, let's use thme to plot our data with the kmeans result

```
plot(x,col=k$cluster)
points(k$centers,col="blue", pch=15)
```



Now for Hierarchial Clustering

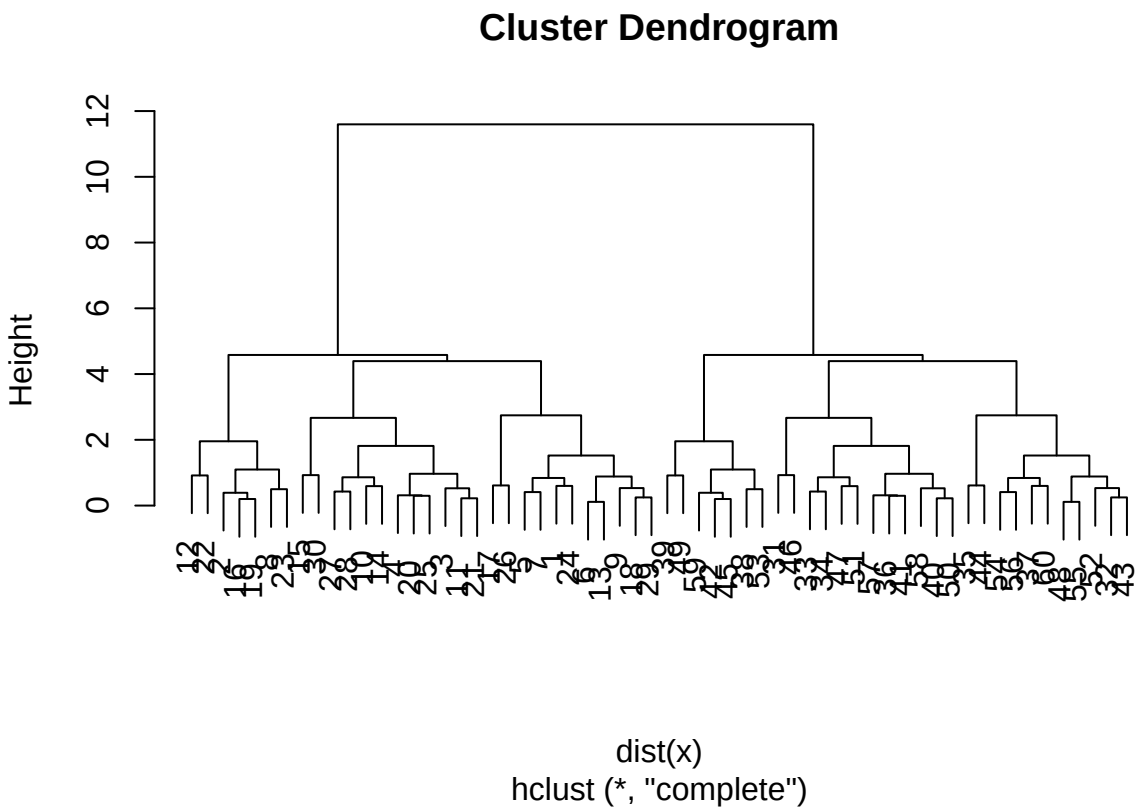
We will cluster the same data 'x' with the 'hclust()'. In this case 'hclust()' requires a distance matrix as input

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

```
##
## Call:
## hclust(d = dist(x))
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 60
```

Let's plot our hclus result

```
plot(hc)
```



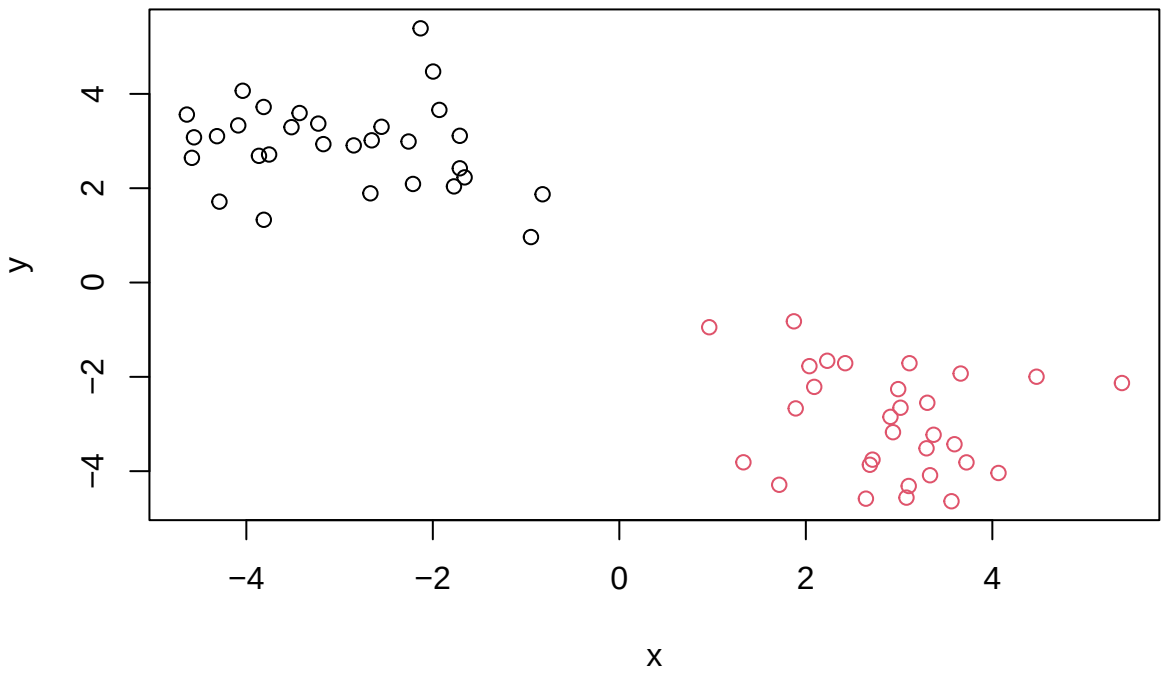
To get our cluster membershit vector we need to "cut" the tree with 'cutree()'

```
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 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
```

Now plot our data with hclust() results

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



Principal Component Analysis (PCA)

PCA of UK Food Data

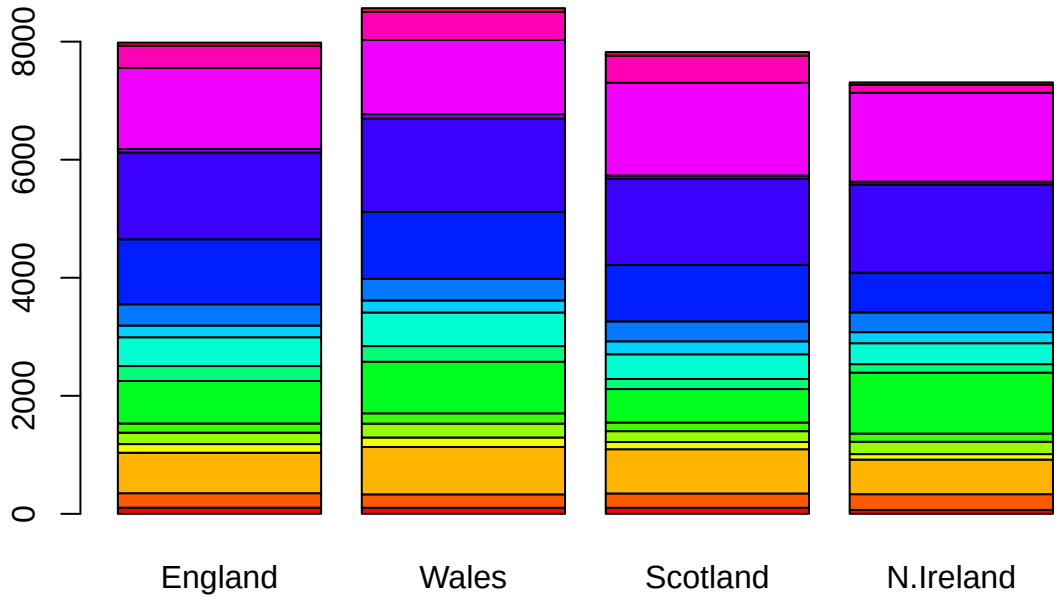
Read data from website and try a few visualization

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

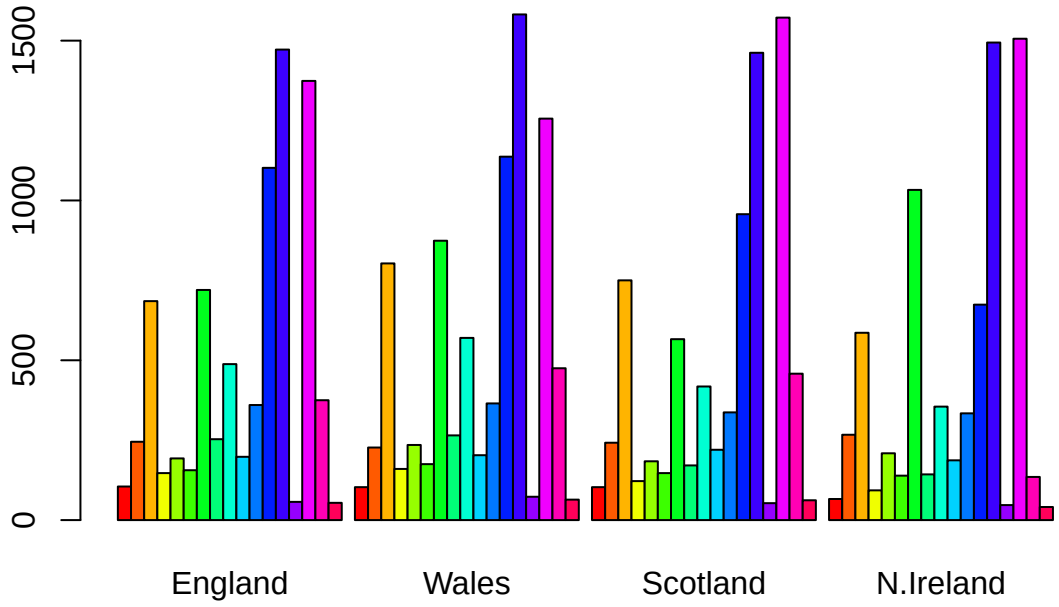
```
##                     England Wales Scotland N.Ireland
## Cheese                  105   103      103        66
## Carcass_meat            245   227      242       267
## Other_meat              685   803      750       586
## Fish                    147   160      122        93
## Fats_and_oils           193   235      184       209
## Sugars                  156   175      147       139
## Fresh_potatoes          720   874      566      1033
## Fresh_Veg               253   265      171       143
## Other_Veg               488   570      418       355
## Processed_potatoes      198   203      220       187
## Processed_Veg           360   365      337       334
## 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
```

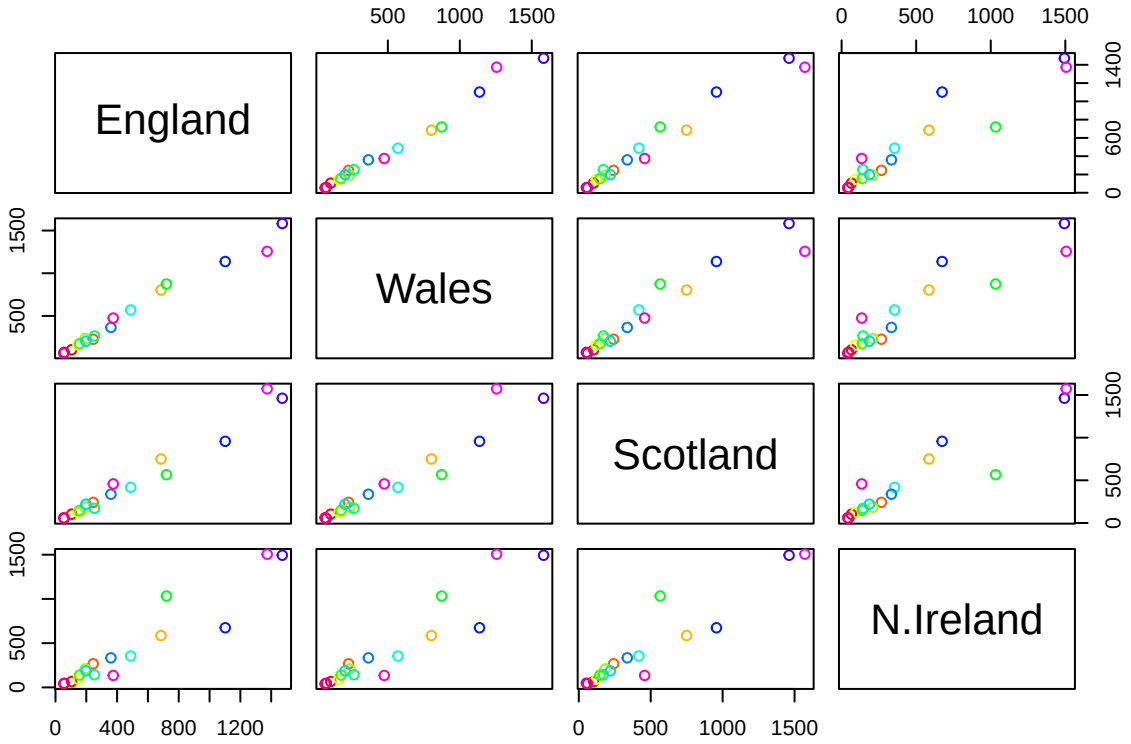
```
cols <-rainbow(nrow(x))
barplot(as.matrix(x),col=cols)
```




```
barplot(as.matrix(x),col=cols, beside = TRUE)
```




```
pairs(x,col=cols)
```



PCA to the rescue! The main base R PCA function is called 'prcomp()'and we will need to give it the transporse of our input data.

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

```
## Standard deviations (1, .., p=4):
## [1] 3.241502e+02 2.127478e+02 7.387622e+01 4.188568e-14
##
## Rotation (n x k) = (17 x 4):
##                              PC1          PC2         PC3          PC4
## Cheese               -0.056955380 -0.016012850 -0.02394295 -0.691718038
## Carcass_meat          0.047927628 -0.013915823 -0.06367111  0.635384915
## Other_meat           -0.258916658  0.015331138  0.55384854  0.198175921
## Fish                 -0.084414983  0.050754947 -0.03906481 -0.015824630
## Fats_and_oils        -0.005193623  0.095388656  0.12522257  0.052347444
## Sugars               -0.037620983  0.043021699  0.03605745  0.014481347
## Fresh_potatoes        0.401402060  0.715017078  0.20668248 -0.151706089
## Fresh_Veg            -0.151849942  0.144900268 -0.21382237  0.056182433
## Other_Veg            -0.243593729  0.225450923  0.05332841 -0.080722623
## Processed_potatoes   -0.026886233 -0.042850761  0.07364902 -0.022618707
## Processed_Veg        -0.036488269  0.045451802 -0.05289191  0.009235001
## Fresh_fruit          -0.632640898  0.177740743 -0.40012865 -0.021899087
## Cereals              -0.047702858  0.212599678  0.35884921  0.084667257
## Beverages            -0.026187756  0.030560542  0.04135860 -0.011880823
## Soft_drinks           0.232244140 -0.555124311  0.16942648 -0.144367046
```

```
## Alcoholic_drinks    -0.463968168 -0.113536523  0.49858320 -0.115797605
## Confectionery       -0.029650201 -0.005949921  0.05232164 -0.003695024
```

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

There is a nice summary of how well PCA is doing

```
summary(pca)
```

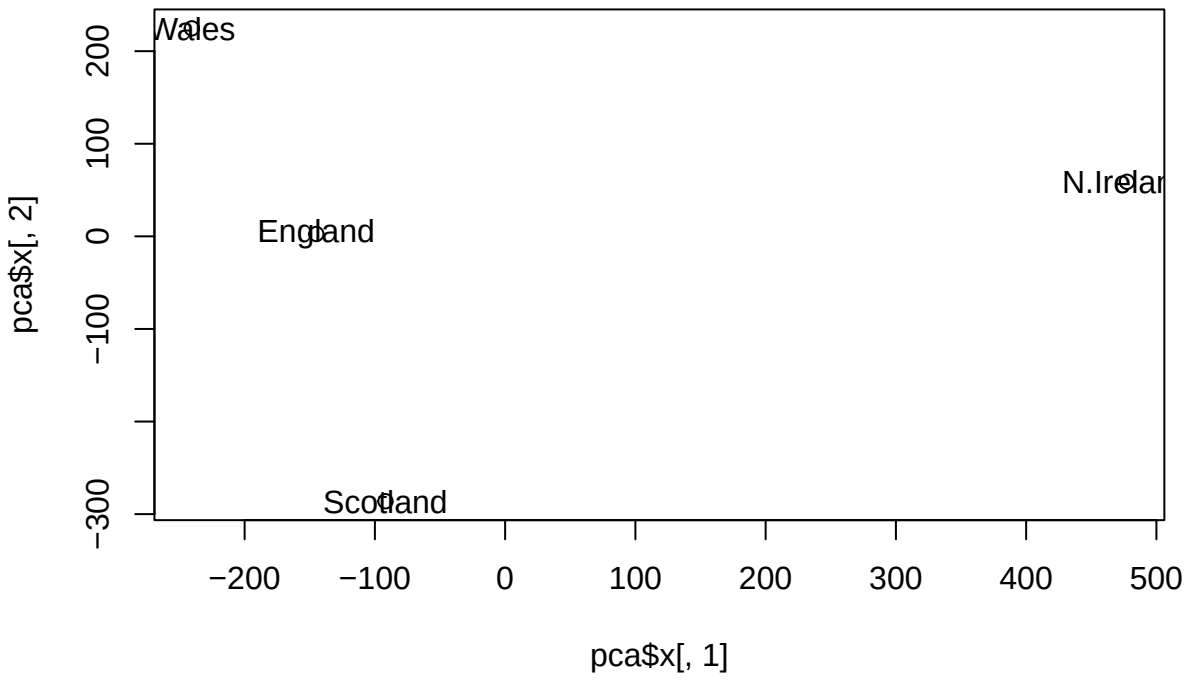
```
## Importance of components:
##                             PC1      PC2      PC3       PC4
## Standard deviation     324.1502 212.7478 73.87622 4.189e-14
## Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
## Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

```
attributes (pca)
```

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

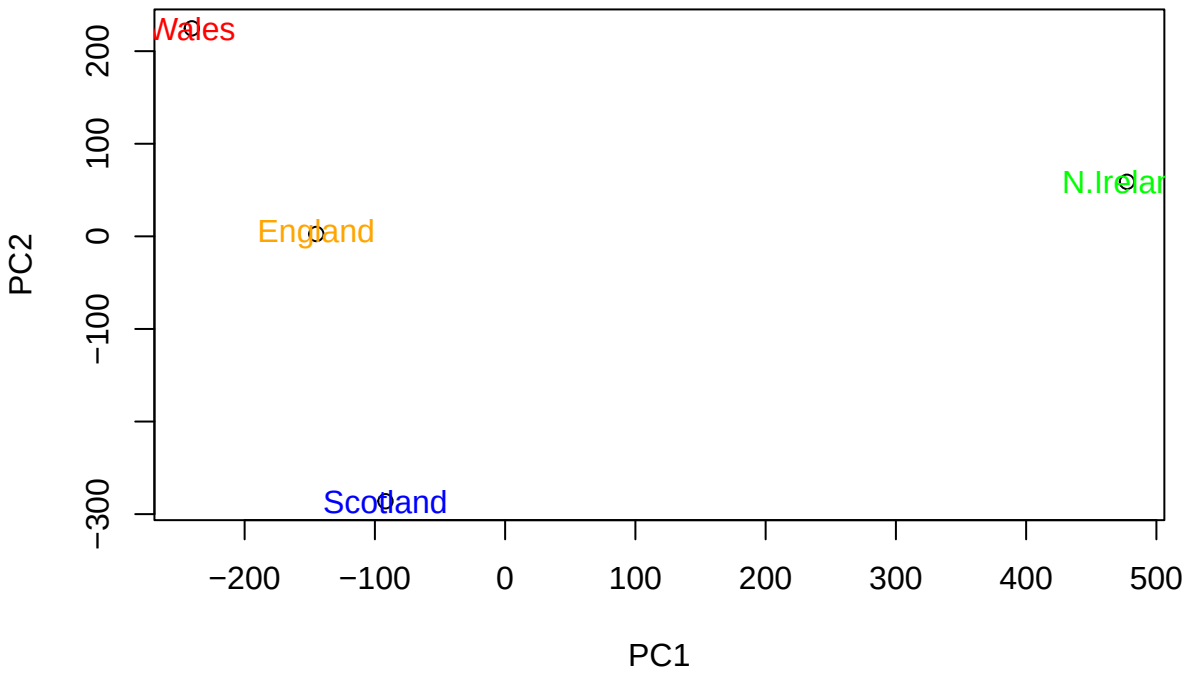
To make our new PCA plot (aka PCA score plot) we access 'pca$x'

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



Color up the plot

```
country_cols <- c("orange", "red", "blue", "green")
plot(pca$x[,1],pca$x[,2],xlab="PC1", ylab="PC2")
text(pca$x[,1],pca$x[,2],colnames(x), col=country_cols)
```



Calculate how much variation in the original data each PC accounts for

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

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

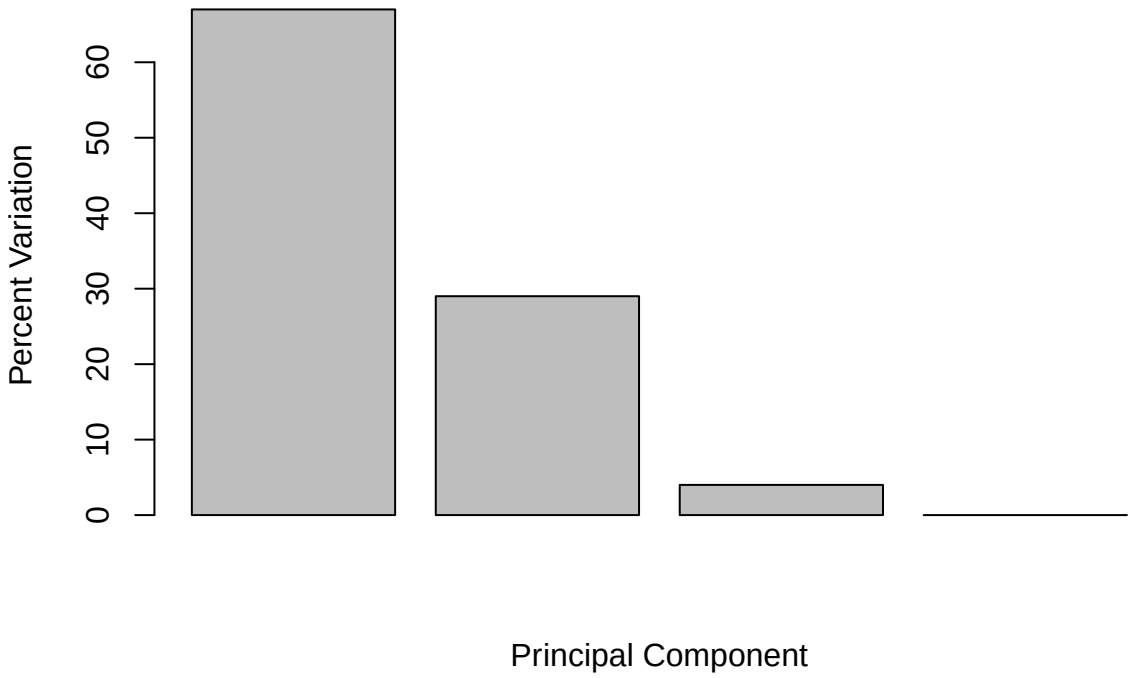
Or second row

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

```
##                             PC1       PC2      PC3          PC4
## Standard deviation     324.15019 212.74780 73.87622 4.188568e-14
## Proportion of Variance   0.67444   0.29052  0.03503 0.000000e+00
## Cumulative Proportion    0.67444   0.96497  1.00000 1.000000e+00
```

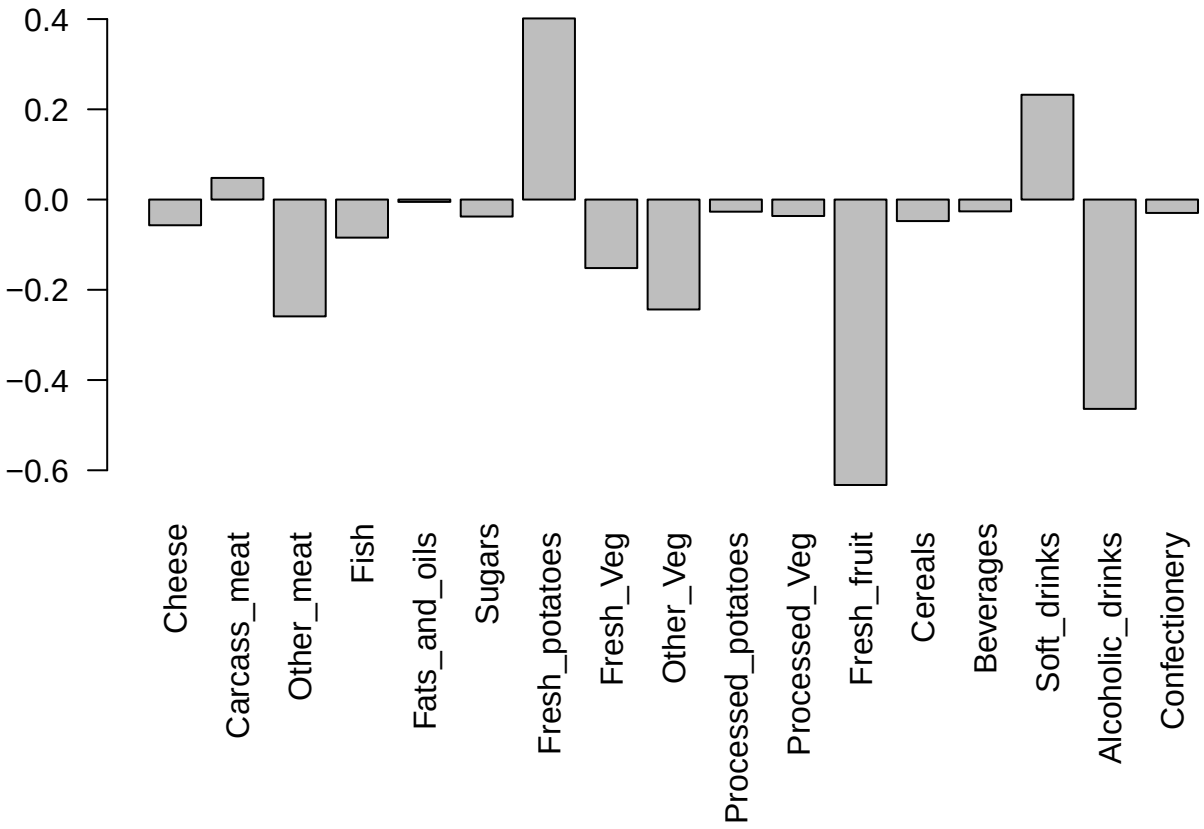
Plot of variances(eigenvalues) with respect to the principal component number (eigenvector number)

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



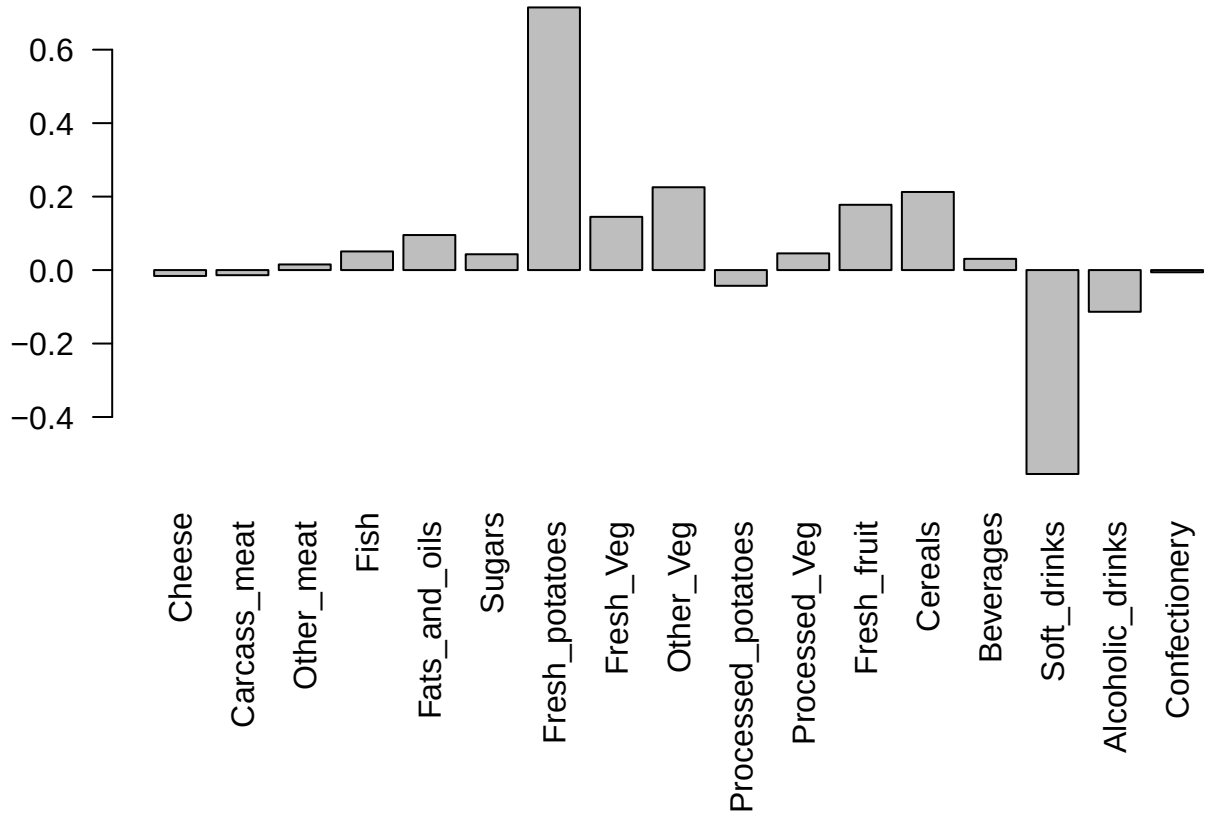
Consider the influence of each original variable upon the principal components (loading scores)

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



Q. Generate a similar 'loadings plot' for PC2. What two food groups feature prominately and what does PC2 mainly tell us about?

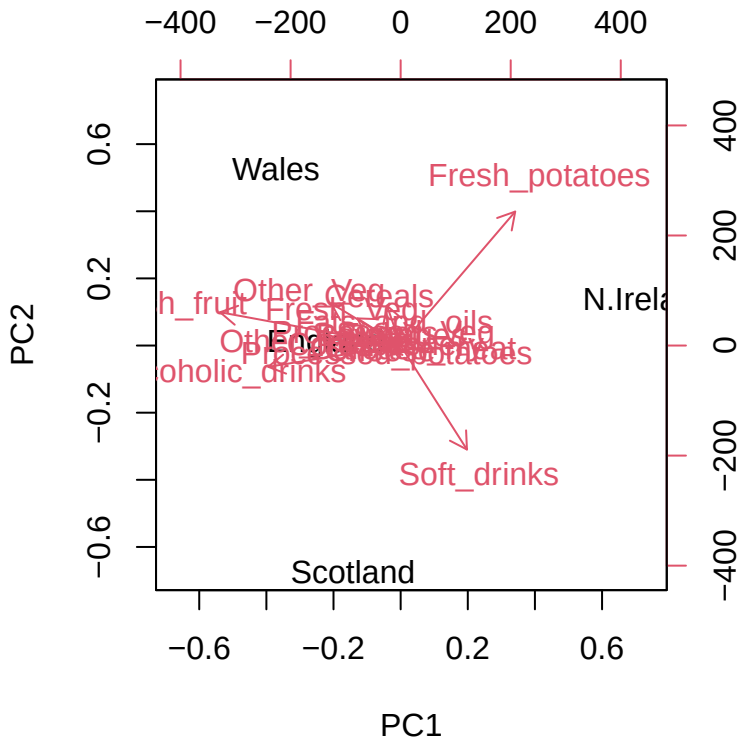
```
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,2], las=2 )
```



Fresh_potatoes and Soft_drinks are the two prominent groups. PC2 tells us that Soft_drinks push the other countries to the left negative side of the plot while Fresh_potatoes pushes N.Ireland to the right positive side of the plot.

Inbuilt biplot()

```
biplot(pca)
```



##PCA of RNA-Seq data

Read in data from website

```
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
```

Q How many genes are in this data set?

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

There is a nice summary of how well PCA is doing

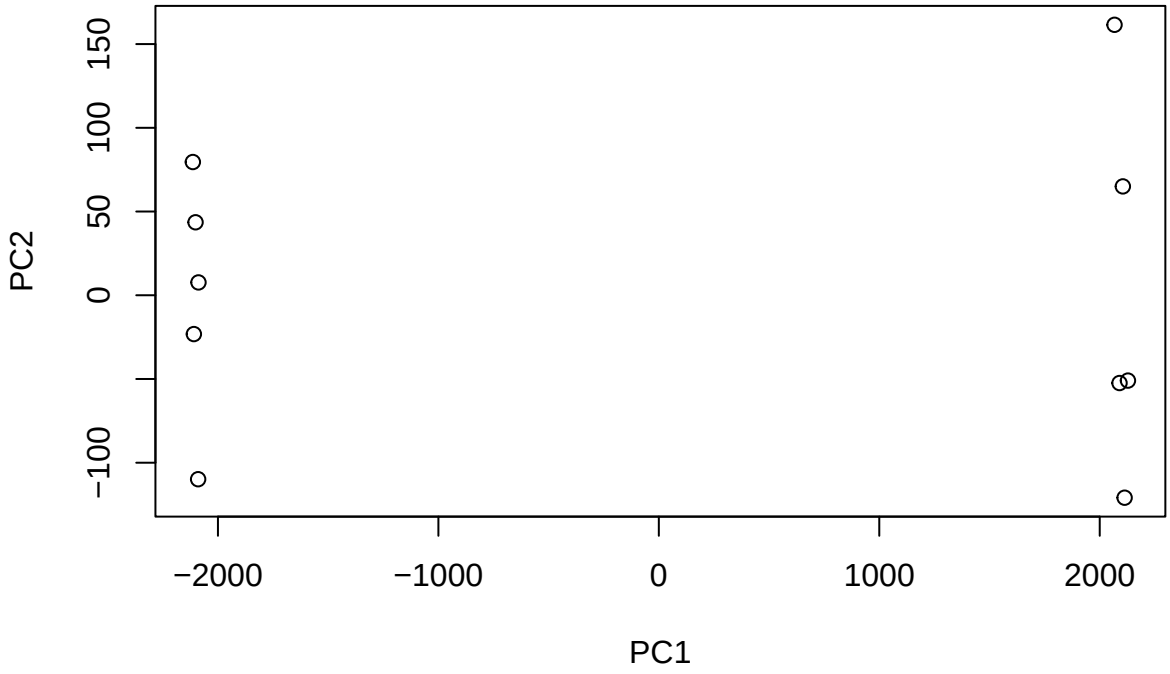
```
summary(pca)
```

```
## Importance of components:
```

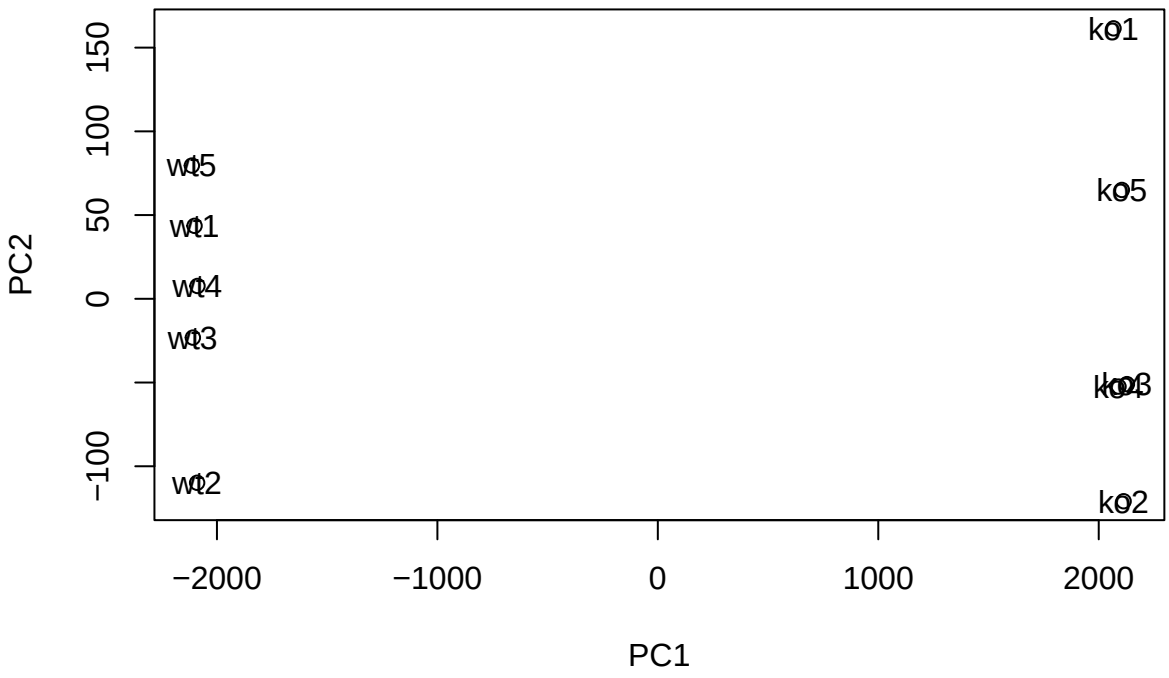
```
##                               PC1     PC2      PC3      PC4      PC5      PC6
## Standard deviation     2214.2633 88.9209 84.33908 77.74094 69.66341 67.78516
## Proportion of Variance    0.9917  0.0016  0.00144  0.00122  0.00098  0.00093
## Cumulative Proportion     0.9917  0.9933  0.99471  0.99593  0.99691  0.99784
##                               PC7      PC8      PC9      PC10
## Standard deviation     65.29428 59.90981 53.20803 3.142e-13
## Proportion of Variance  0.00086  0.00073  0.00057 0.000e+00
## Cumulative Proportion   0.99870  0.99943  1.00000 1.000e+00
```

Do our PCA plot of this RNA-seq data

```
plot(pca$x[,1],pca$x[,2], xlab="PC1", ylab="PC2")
```

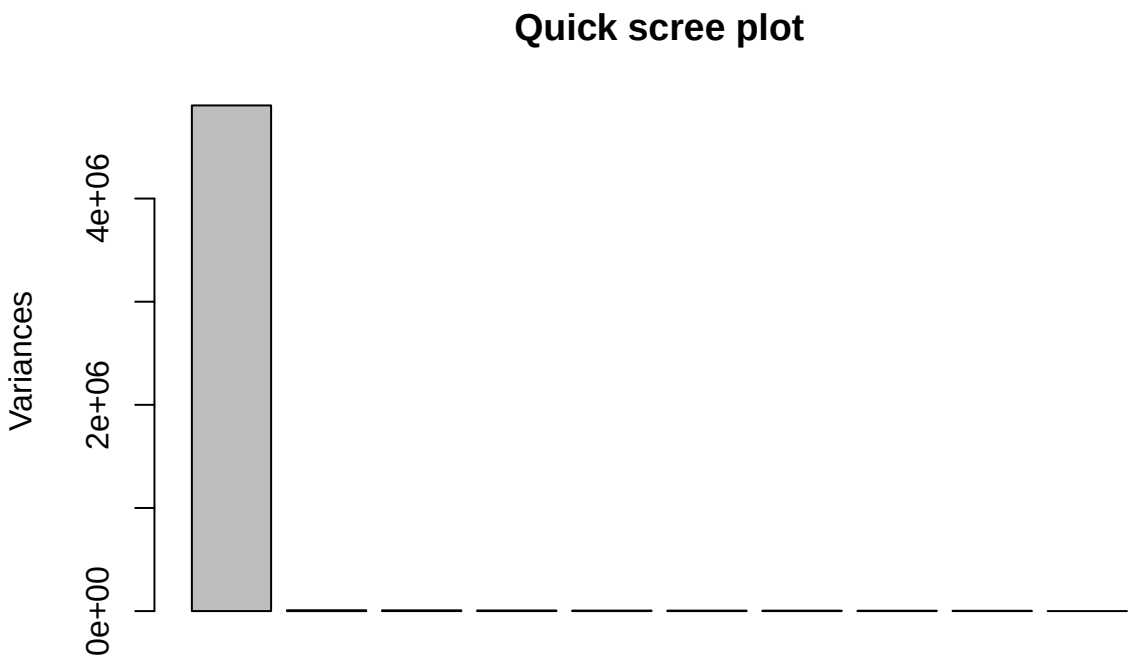



```
plot(pca$x[,1],pca$x[,2], xlab="PC1", ylab="PC2")
text(pca$x[,1],pca$x[,2],colnames(rna.data))
```



Barplot summary of Proportion of Variance for each PC

```
plot(pca, main="Quick scree plot")
```

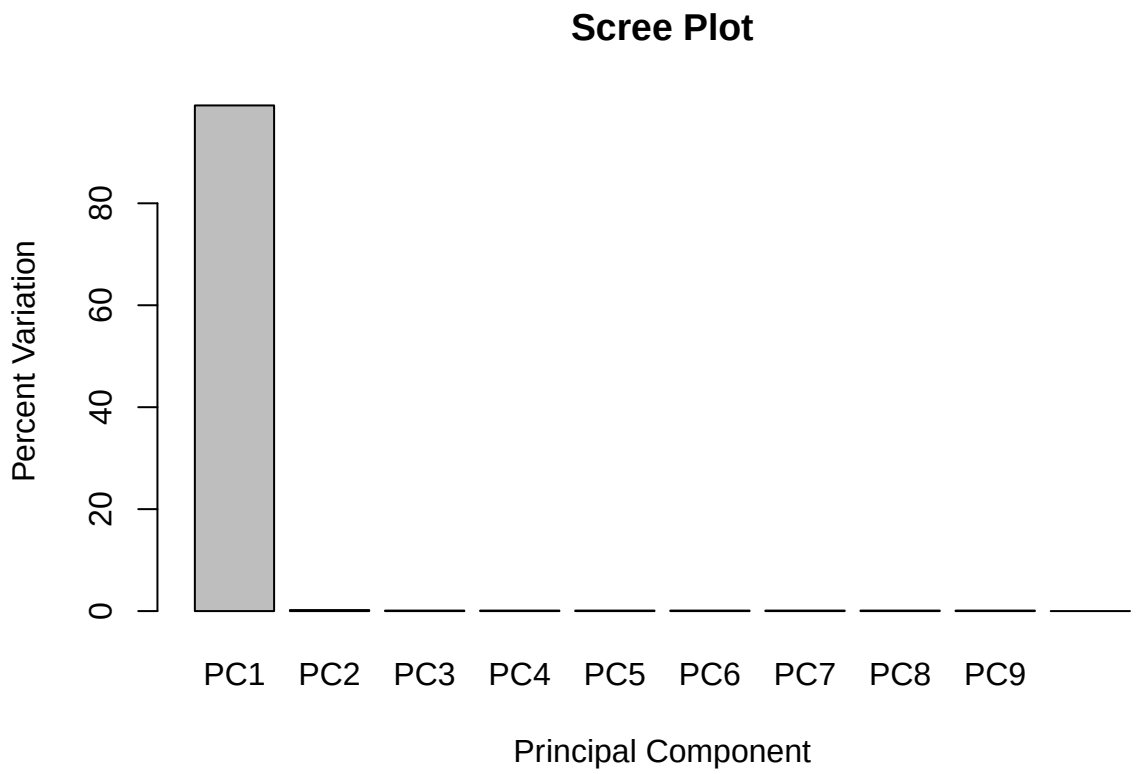


Make a scree plot ourselves

```
## Variance captured per PC
pca.var <- pca$sdev^2

## Percent variance is often more informative to look at
pca.var.per <- round(pca.var/sum(pca.var)*100, 1)
pca.var.per
```

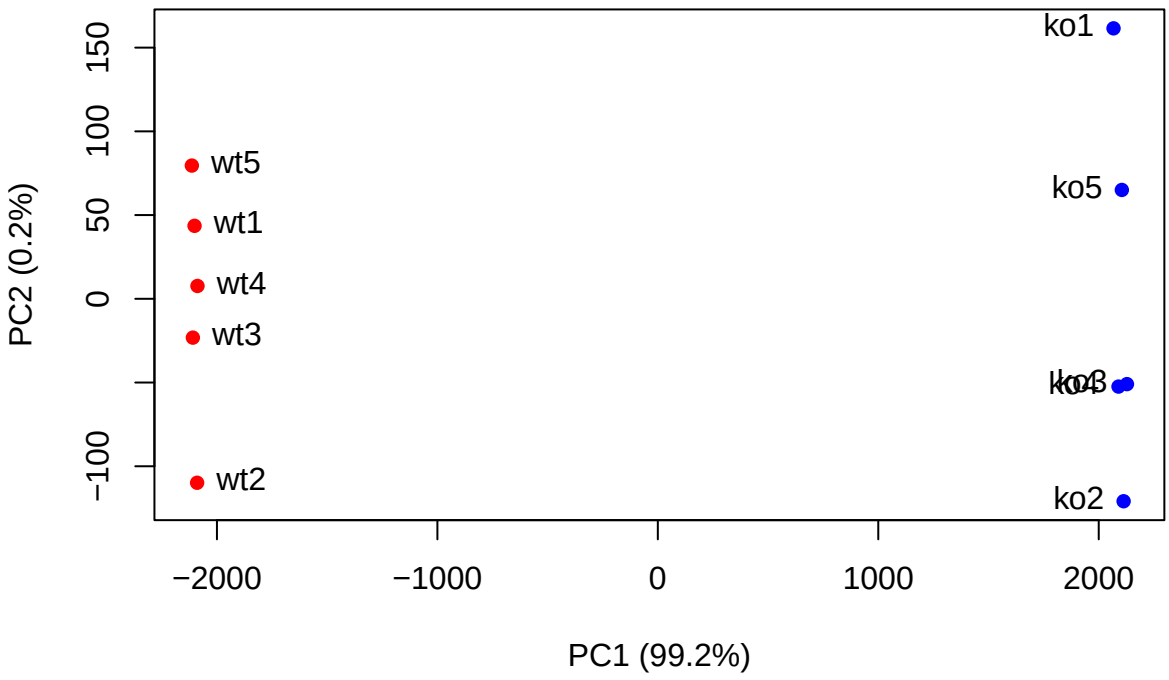
```
##  [1] 99.2  0.2  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.0
```

```
barplot(pca.var.per, main="Scree Plot",
        names.arg = paste0("PC", 1:10),
        xlab="Principal Component", ylab="Percent Variation")
```



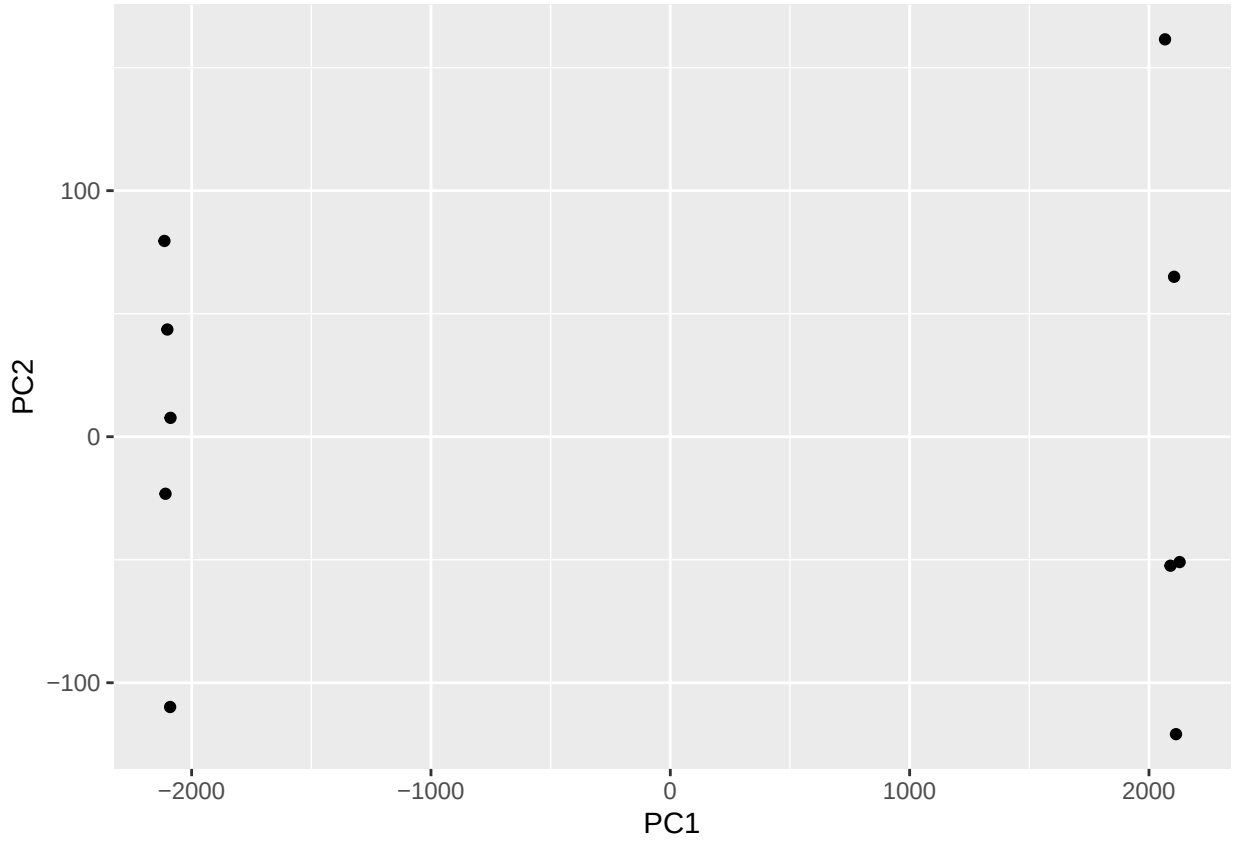
Make the PCA plot more attractive

```
## 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)))
```



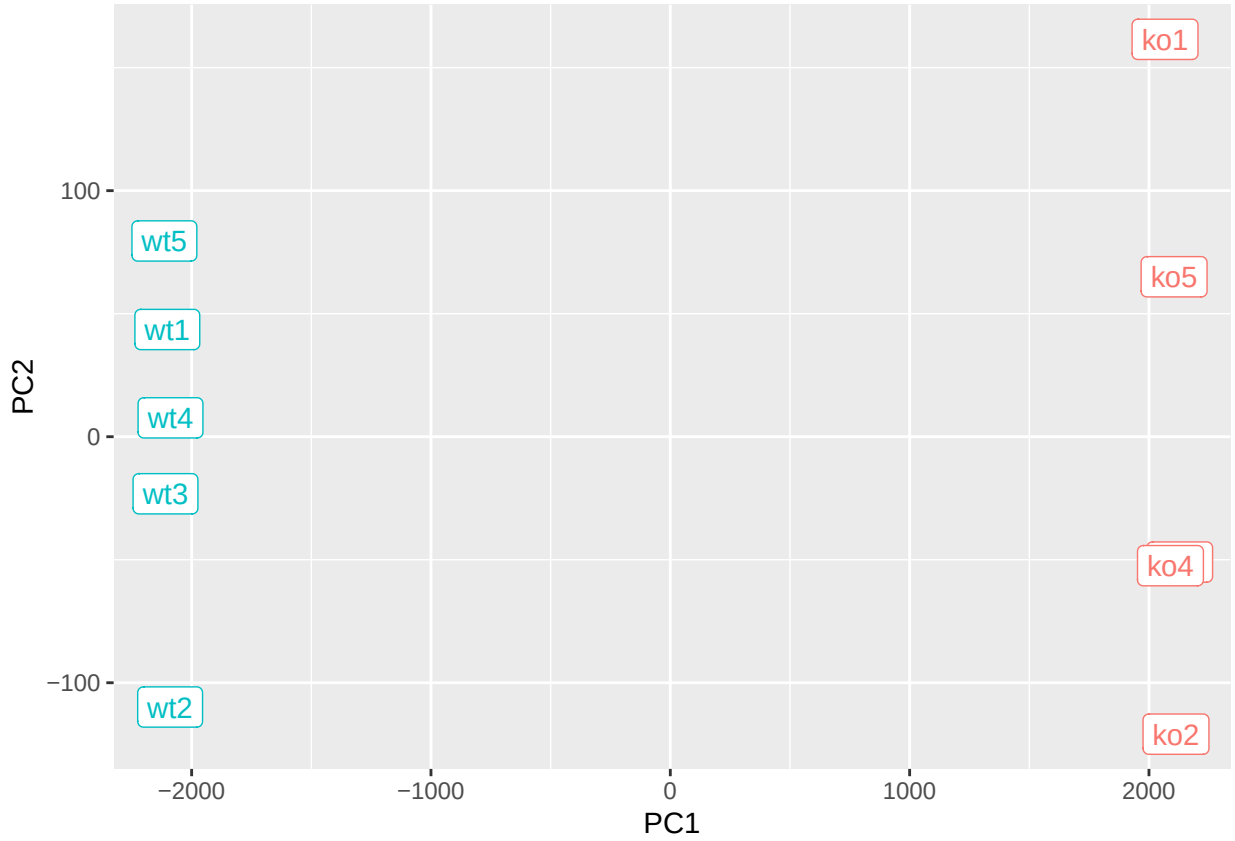
Using ggplot

```
library(ggplot2)

df <- as.data.frame(pca$x)

# Our first basic plot
ggplot(df) +
  aes(PC1, PC2) +
  geom_point()
```

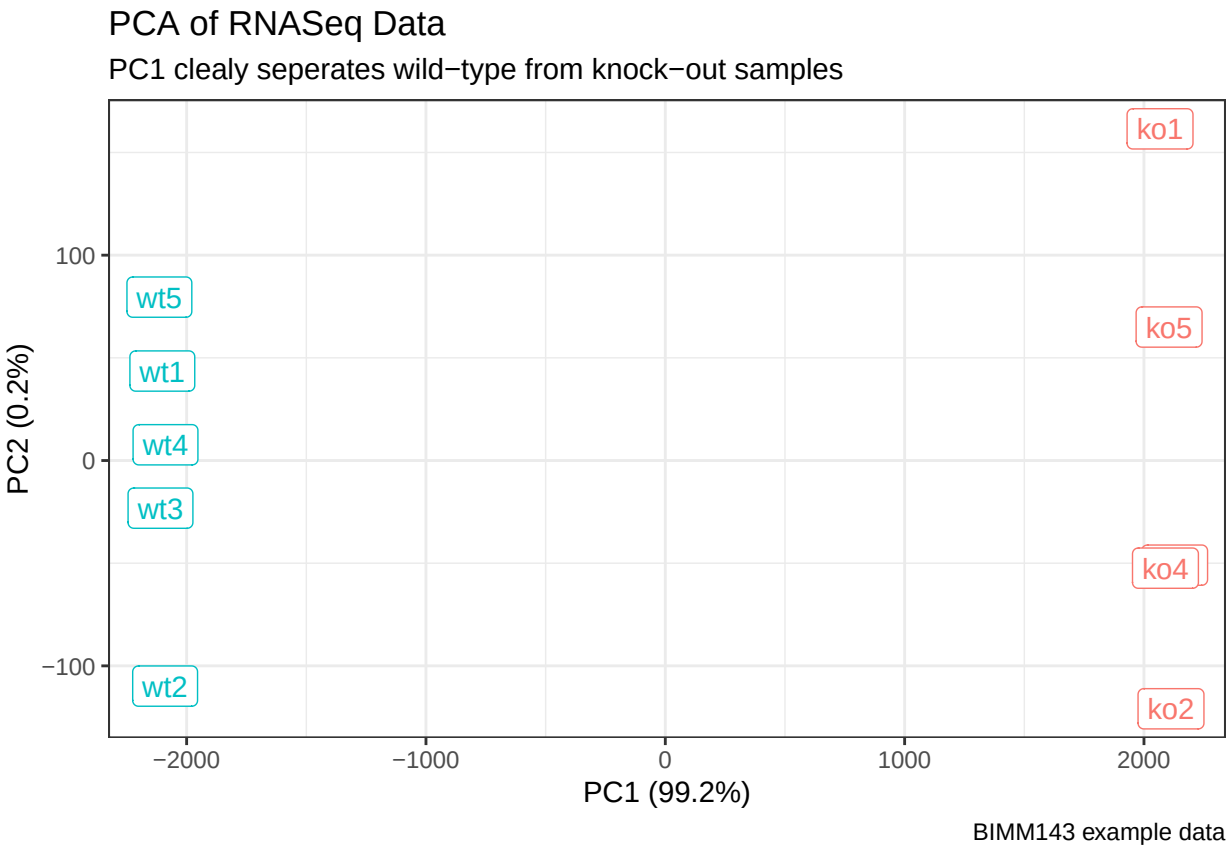


Add a condition specific color and labels for aesthetics

```
df$samples <- colnames(rna.data)
df$condition <- substr(colnames(rna.data),1,2)

p <- ggplot(df) +
        aes(PC1, PC2, label=samples, col=condition) +
        geom_label(show.legend = FALSE)
p
```




```
p + labs(title="PCA of RNASeq Data",
       subtitle = "PC1 clealy seperates wild-type from knock-out samples",
       x=paste0("PC1 (", pca.var.per[1], "%)"),
       y=paste0("PC2 (", pca.var.per[2], "%)"),
       caption="BIMM143 example data") +
     theme_bw()
```


PCA of RNASeq Data
PC1 clealy seperates wild−type from knock−out samples
ko1
wt5
ko5
wt1
wt4
wt3
ko4
wt2
ko2
100
0
−100
PC2 (0.2%)
−2000
−1000
0
1000
2000
PC1 (99.2%)
BIMM143 example data