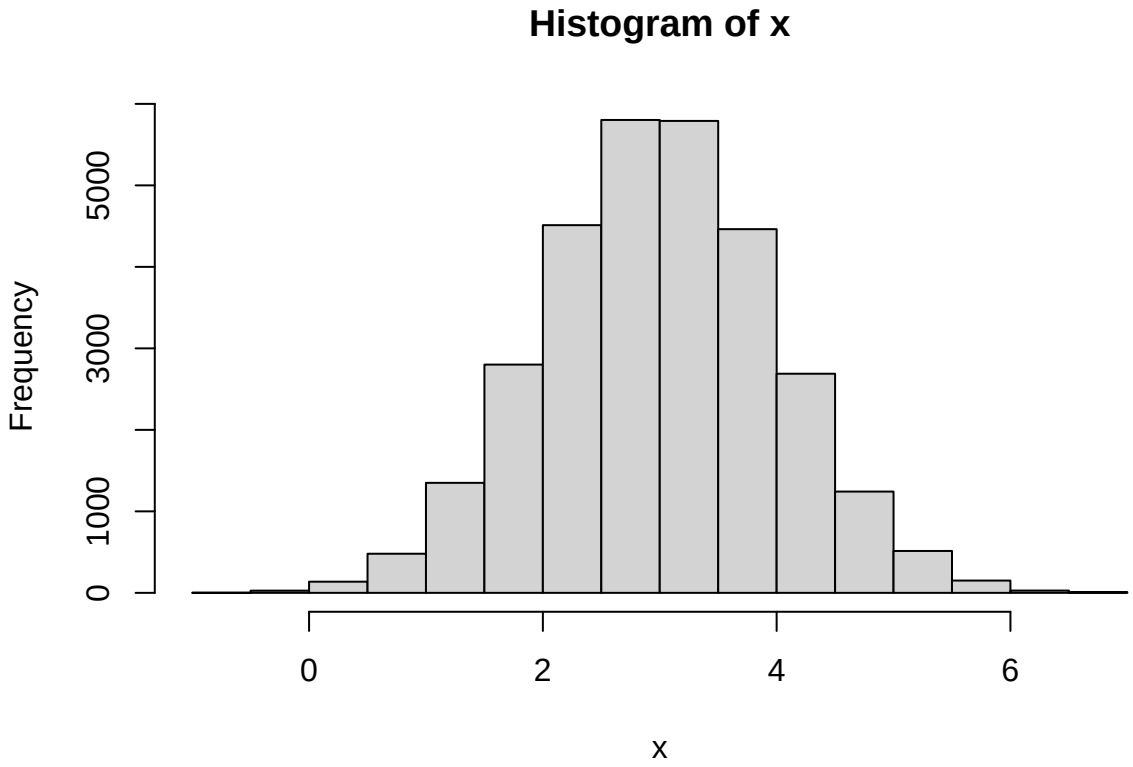
lab07

Zijing

2022-10-19

K-means Clustering

```
x <- rnorm(30000,3)
hist(x)
```




```
rnorm(30,-3)
```

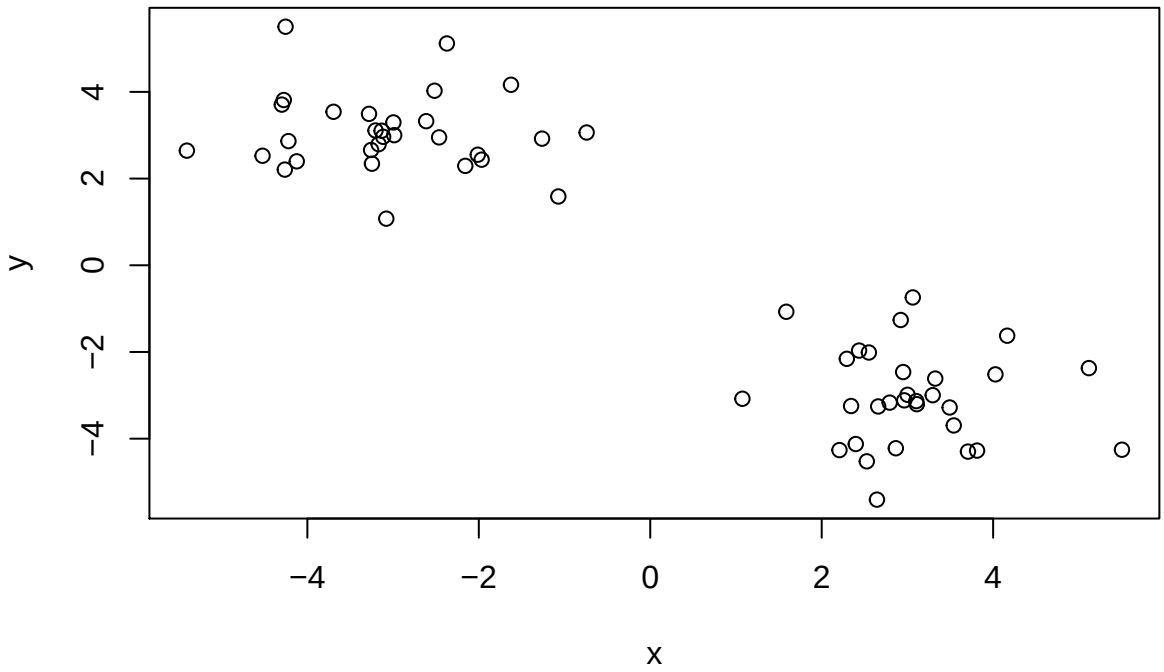
```
##  [1] -3.058790 -2.586772 -2.250747 -3.530271 -2.574675 -1.525708 -1.626995
##  [8] -2.251641 -3.665513 -4.290754 -2.379379 -2.451963 -1.758380 -3.885508
## [15] -2.414995 -2.435145 -3.806971 -3.312111 -1.891516 -3.381180 -1.170601
## [22] -2.747997 -2.535215 -3.815960 -3.818792 -1.183710 -2.139607 -4.198730
## [29] -2.792743 -2.299871
```

```
rnorm(30,3)
```

```
##  [1] 3.5219248 4.2785781 2.9129491 3.4720439 2.0386666 0.9985537 3.0184653
##  [8] 2.2171032 1.8312852 2.1001984 4.1198697 4.4881161 0.7485240 3.7902086
## [15] 3.2403044 1.9740989 3.9682109 2.2234926 3.5571700 2.0980235 2.9102841
```

```
## [22] 3.6296167 1.8541726 2.9973483 2.7472250 3.4858232 2.1330516 3.9389452
## [29] 2.5545049 3.5994275
```

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



The function to do k-means clustering in base R is called `kmeans()`.

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

```
## K-means clustering with 2 clusters of sizes 30, 30
##
## Cluster means:
##           x         y
## 1 -3.044416  3.049221
## 2  3.049221 -3.044416
##
## 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] 59.30034 59.30034
##  (between_SS / total_SS =  90.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km$size
```

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

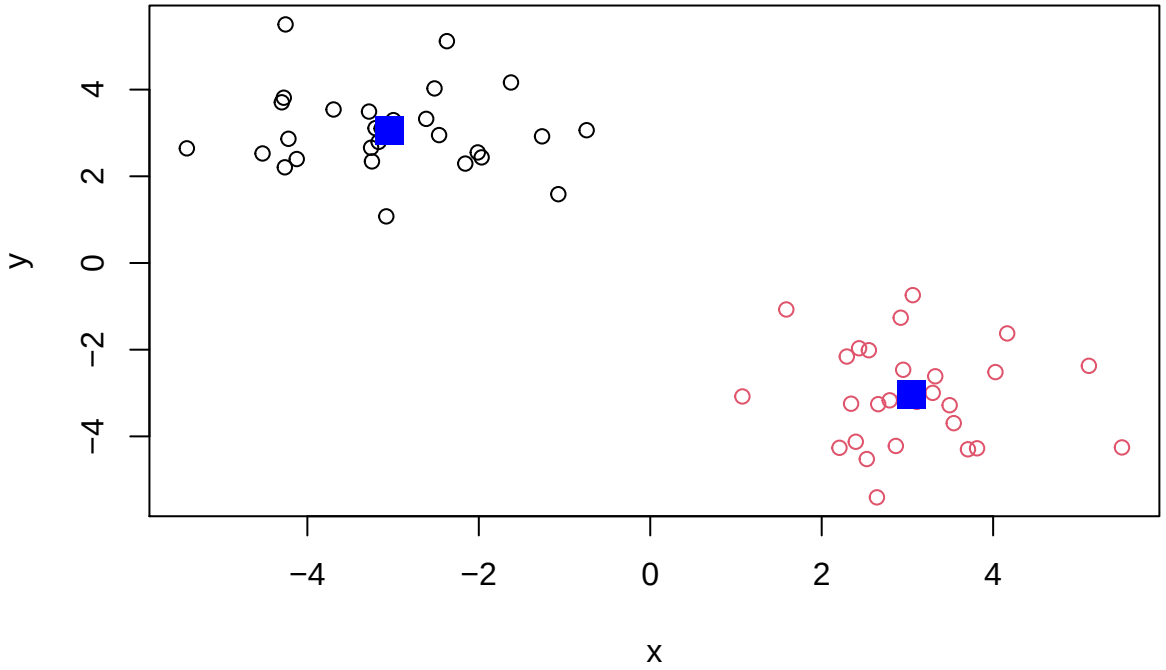
```
km$cluster
```

```
##  [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 2
```

```
km$centers
```

```
##           x         y
## 1 -3.044416  3.049221
## 2  3.049221 -3.044416
```

```
plot(x,col=km$cluster)
points(km$centers[,1],km$center[,2],col="blue",pch=15,cex=2)
```

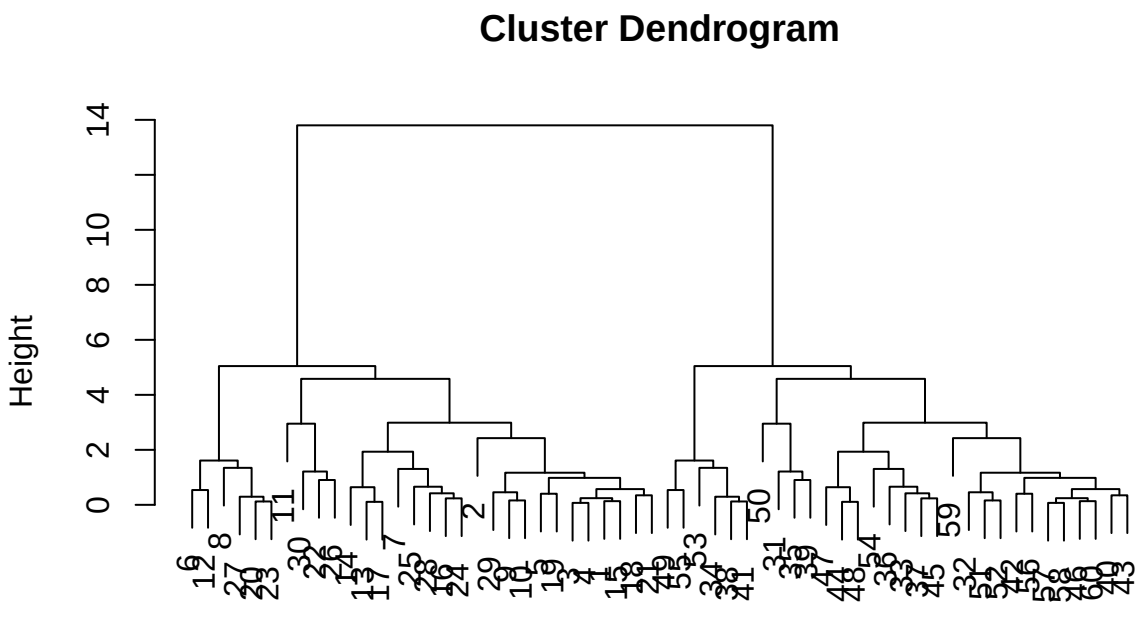


To run hclust() need to input a distance matrix

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

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

```
plot(hc)
```



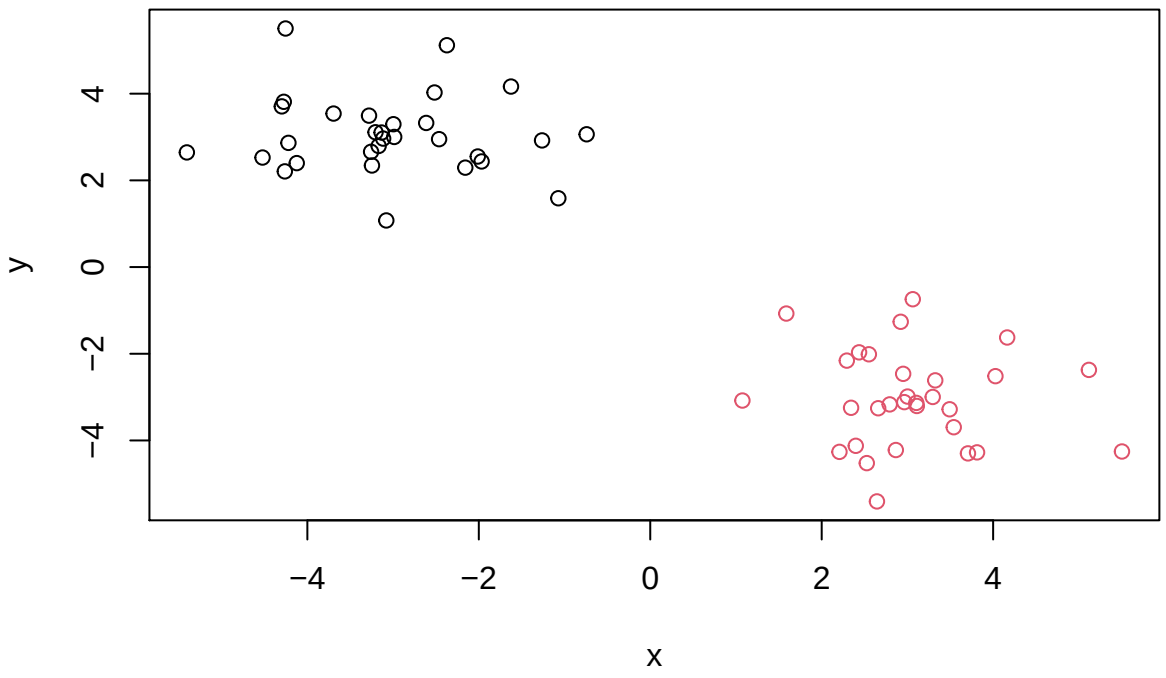
dist(x)
hclust (*, "complete")

```
cutree(hc, h=8)
```

```
##  [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
```

```
grps <- cutree(hc,k=2)
```

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



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

```
## [1] 17  5
```

```
#preview the first 6 rows
#head(x,6)
```

```
#Don't run this cell without running first cell first
rownames(x) <- x[,1]
x <- x[,-1]
head(x,6)
```

```
##                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
```

Q1

```
dim(x)
```

```
## [1] 17  4
```

There are 17 rows and 4 columns.

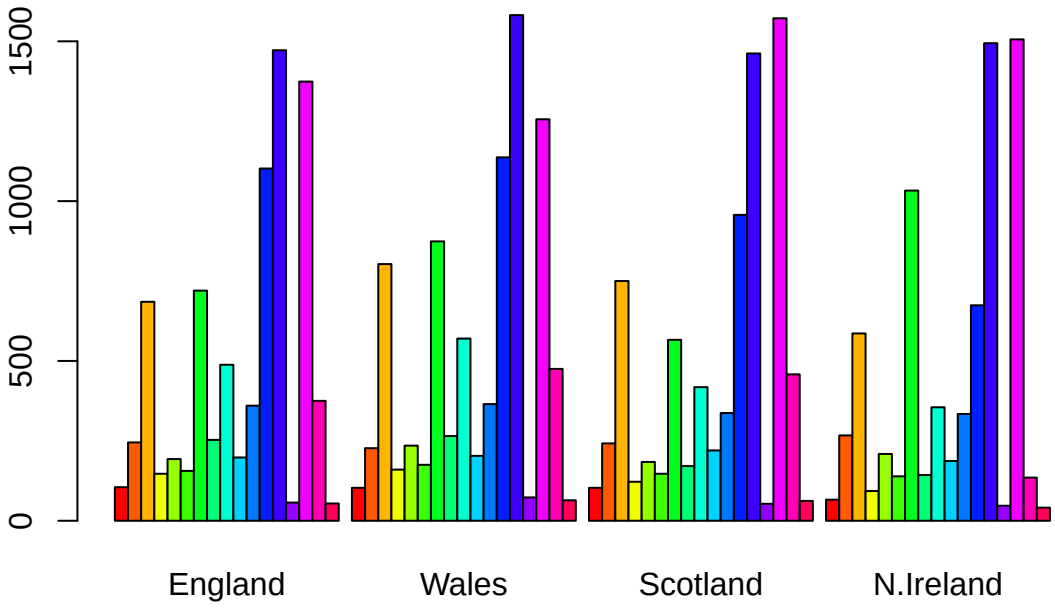
```
x <- read.csv(url,row.names = 1)
head(x)
```

```
##                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

The second method better. the previous method can cause the columns of x to be deleted one by one if the cell with x[,-1] is ran more than once.

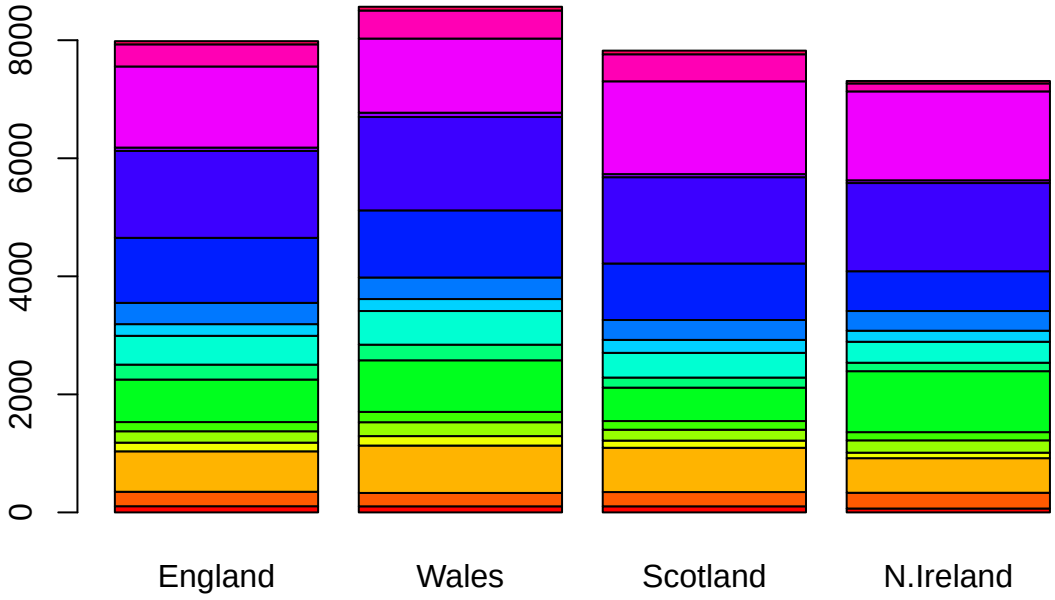
```
barplot(as.matrix(x),beside = T, col=rainbow(nrow(x)))
```



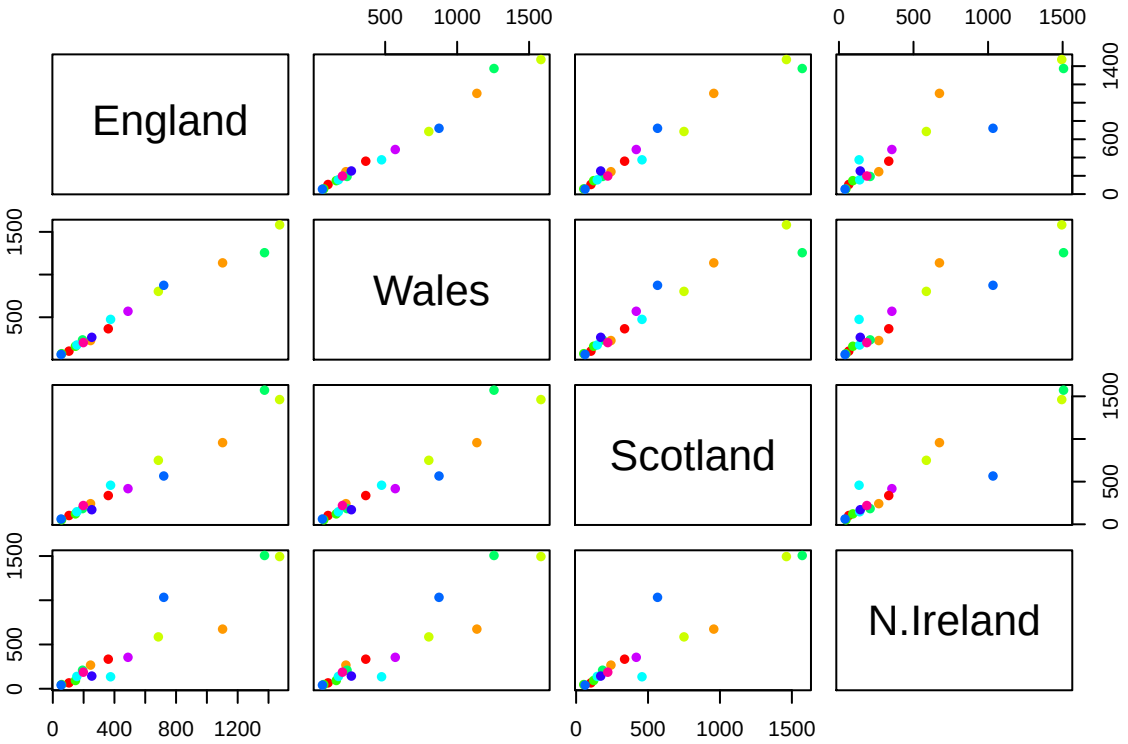
Q3

Changing beside to False changes the graph

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




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



Q5 A given point lies on the diagonal if the consumption of the corresponding food is the same in the two countries being compared.

Q6

N. Ireland is consuming much more fresh potatoes and less alcoholic drinks than the other three countries.

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

```
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
```

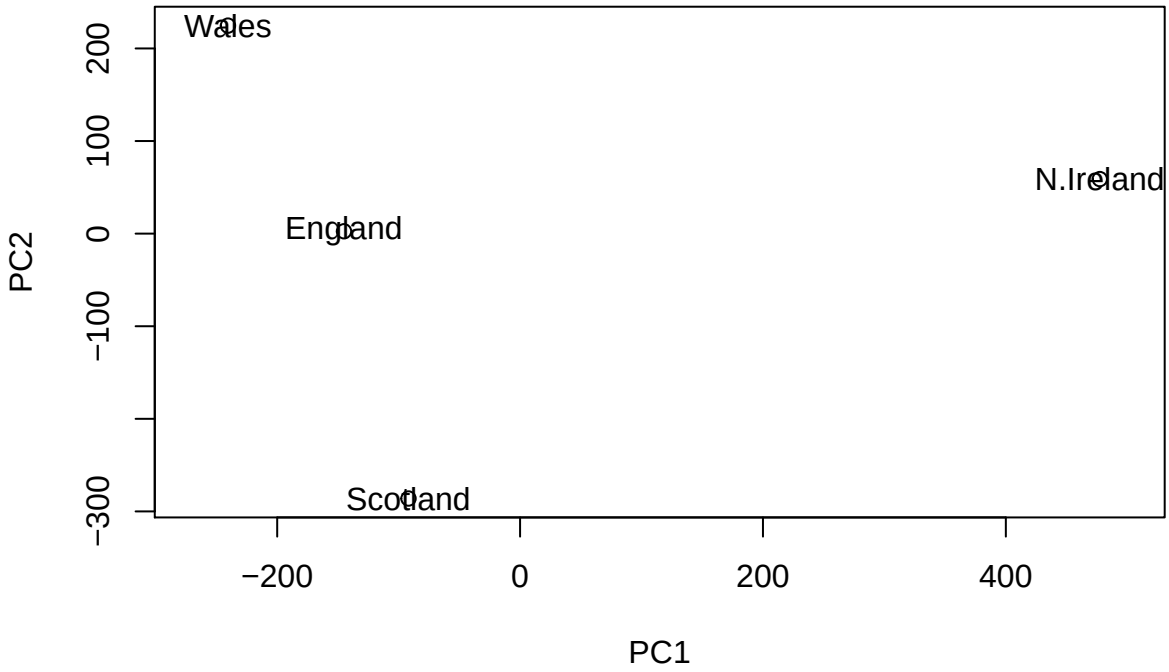
```
## Other_Veg           -0.243593729  0.225450923  0.05332841 -0.080722623
## Processed_potatoes  -0.026886233 -0.042850761  0.07364902 -0.022618707
## Processed_Veg       -0.036488269  0.045451802 -0.05289191  0.009235001
## Fresh_fruit         -0.632640898  0.177740743 -0.40012865 -0.021899087
## 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$x
```

```
##                  PC1         PC2         PC3           PC4
## England   -144.99315    2.532999 -105.768945  2.842865e-14
## Wales     -240.52915  224.646925   56.475555  7.804382e-13
## Scotland   -91.86934 -286.081786   44.415495 -9.614462e-13
## N.Ireland  477.39164   58.901862    4.877895  1.448078e-13
```

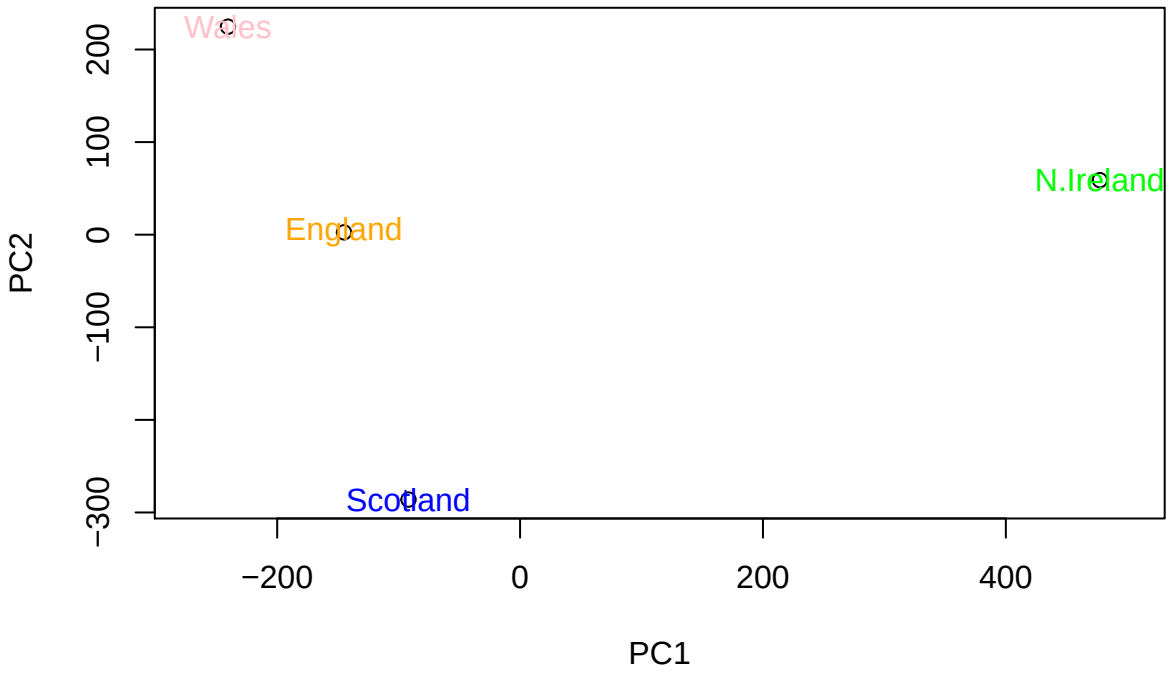
Q7

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



Q8

```
plot(pca$x[,1],pca$x[,2],xlab="PC1", ylab="PC2", xlim=c(-270,500))
text(pca$x[,1],pca$x[,2],colnames(x),col=c("orange","pink","blue","green"))
```



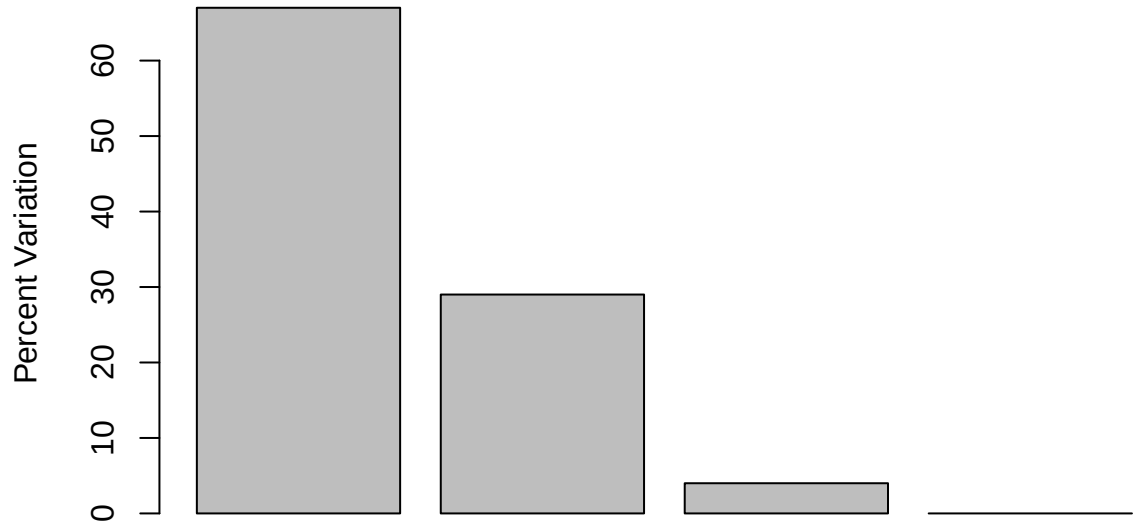

```
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

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

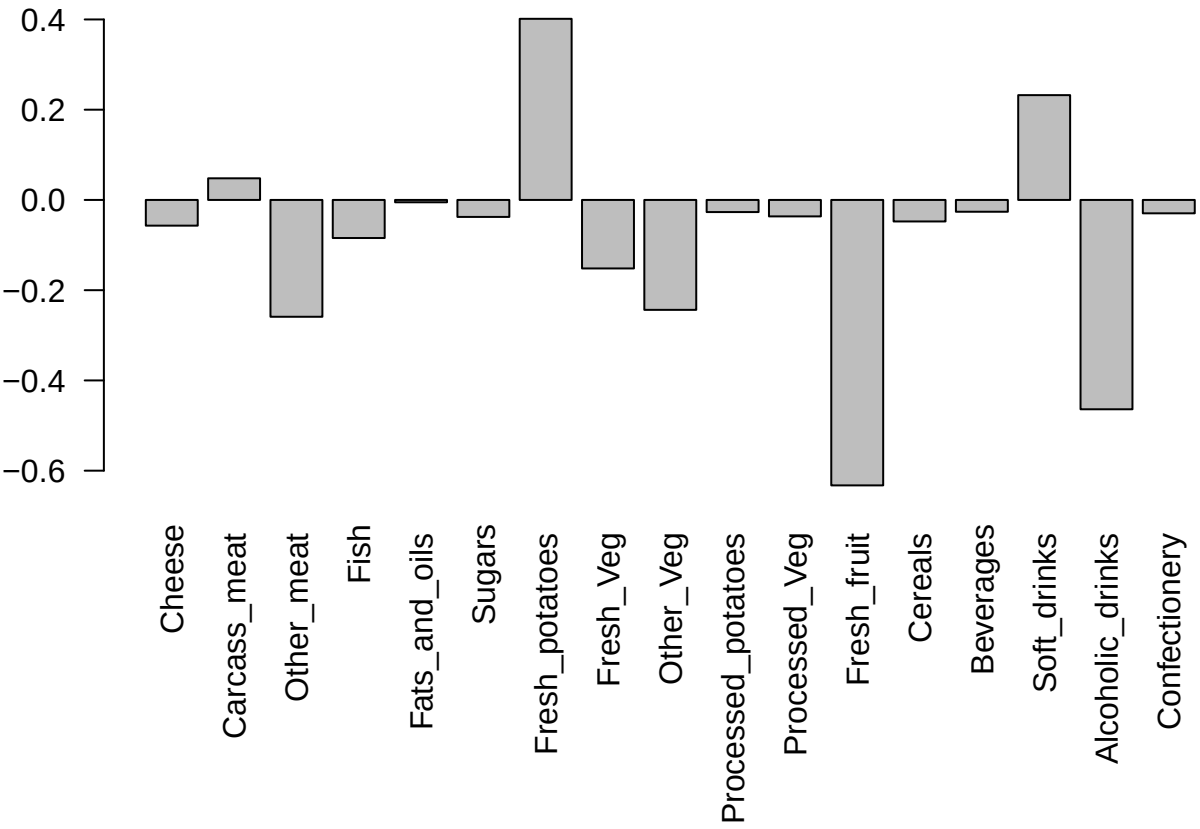
```
z <- summary(pca)
z$importance
```

```
##                             PC1       PC2      PC3          PC4
## Standard deviation     324.15019 212.74780 73.87622 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
```

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



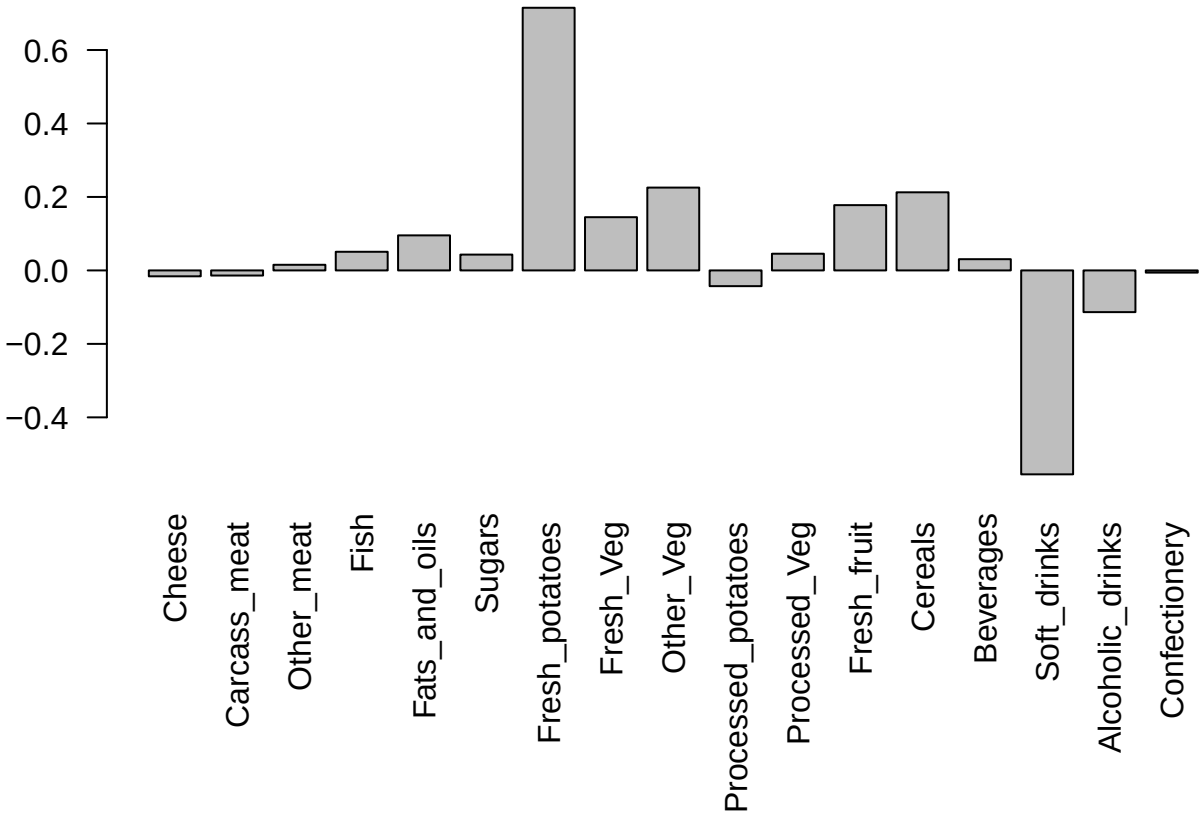

```
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,1], las=2 )
```



#

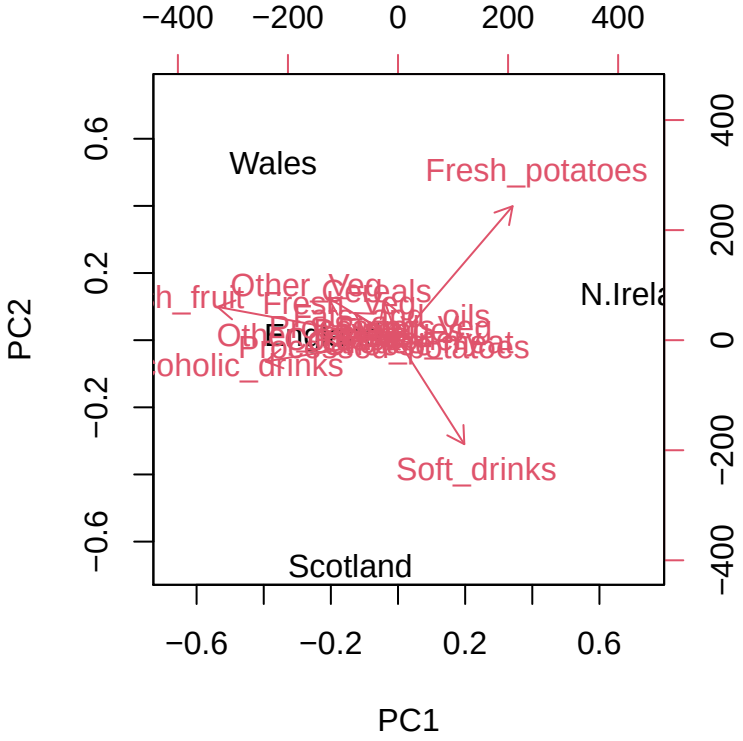
Q9

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



Fresh potatoes and soft drinks stands out. The PC2 tells us that fresh potatoes and soft drinks are what send Scotland and Wales to the bottom and top of the plot in PC2 vs. PC1 plot.

```
biplot(pca)
```



```
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
```

Q10

```
gene <- nrow(rna.data)
sample <- ncol(rna.data)
gene
```

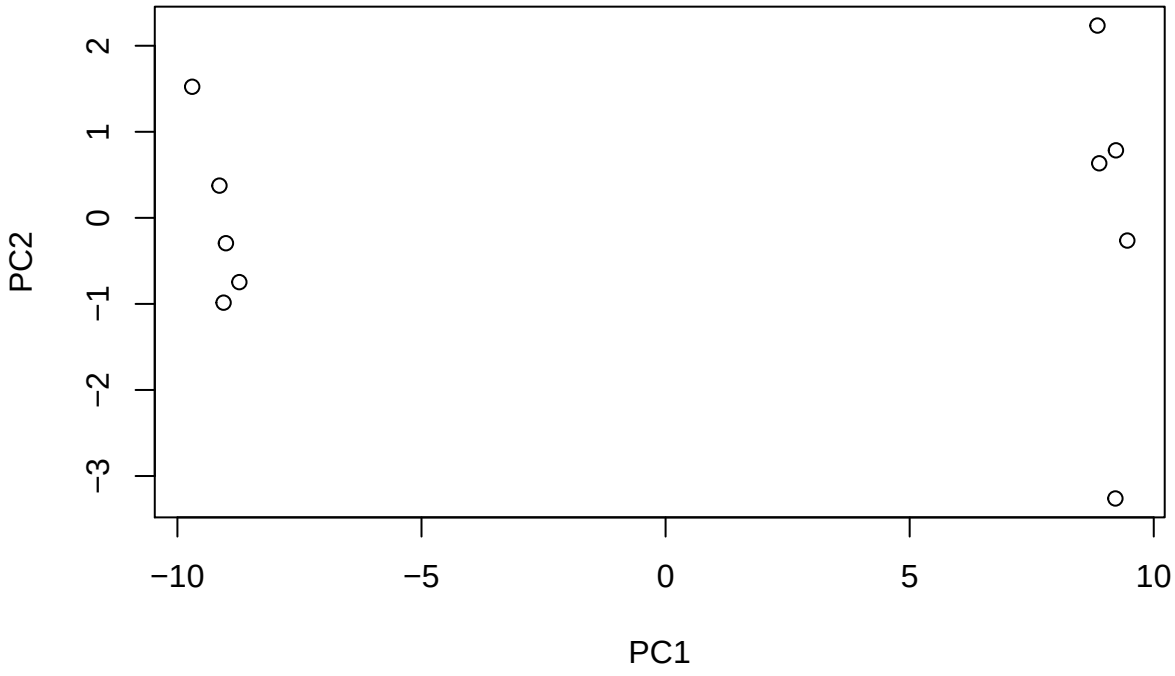
```
## [1] 100
```

```
sample
```

```
## [1] 10
```

There are 100 genes and 10 samples in the data.

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

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

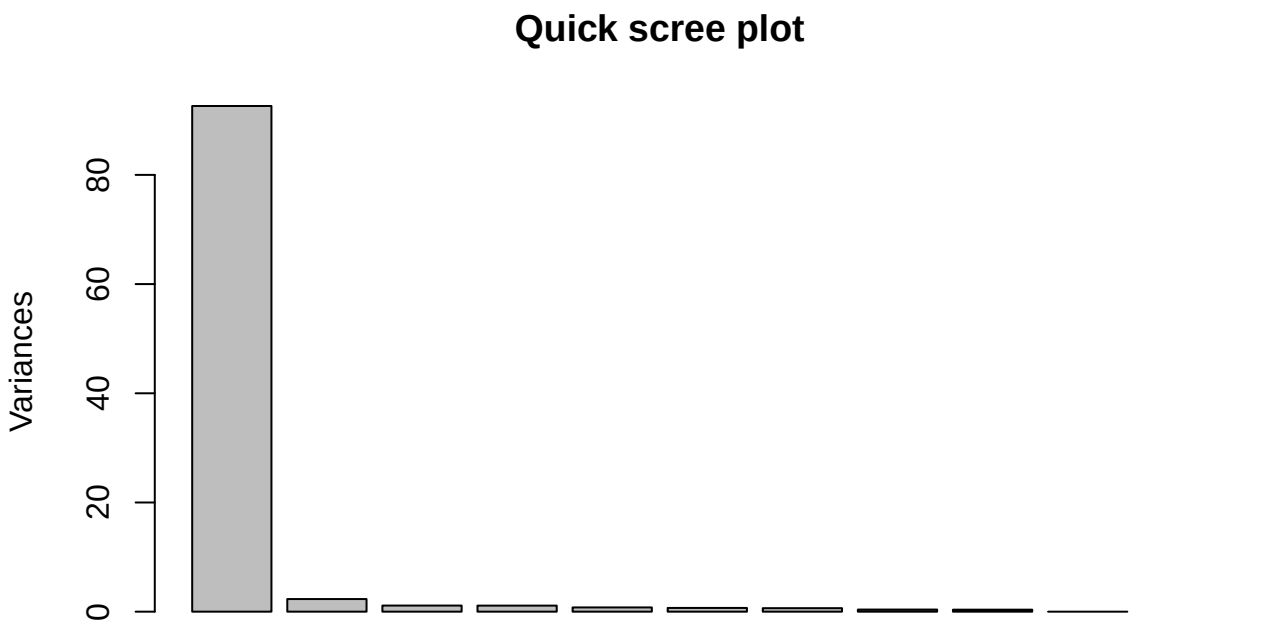



```
summary(pca)
```

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

```
##                             PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     9.6237 1.5198 1.05787 1.05203 0.88062 0.82545 0.80111
## Proportion of Variance 0.9262 0.0231 0.01119 0.01107 0.00775 0.00681 0.00642
## Cumulative Proportion  0.9262 0.9493 0.96045 0.97152 0.97928 0.98609 0.99251
##                            PC8     PC9      PC10
## Standard deviation     0.62065 0.60342 3.348e-15
## Proportion of Variance 0.00385 0.00364 0.000e+00
## Cumulative Proportion  0.99636 1.00000 1.000e+00
```

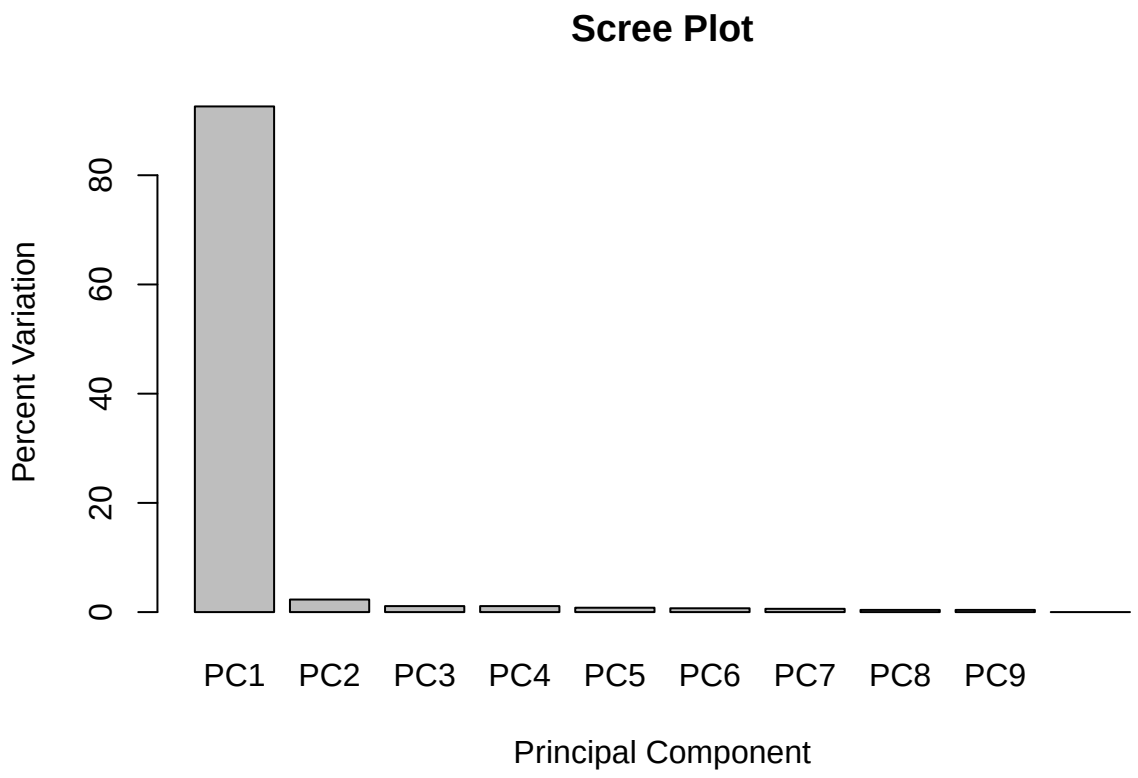
```
plot(pca, main="Quick scree plot")
```



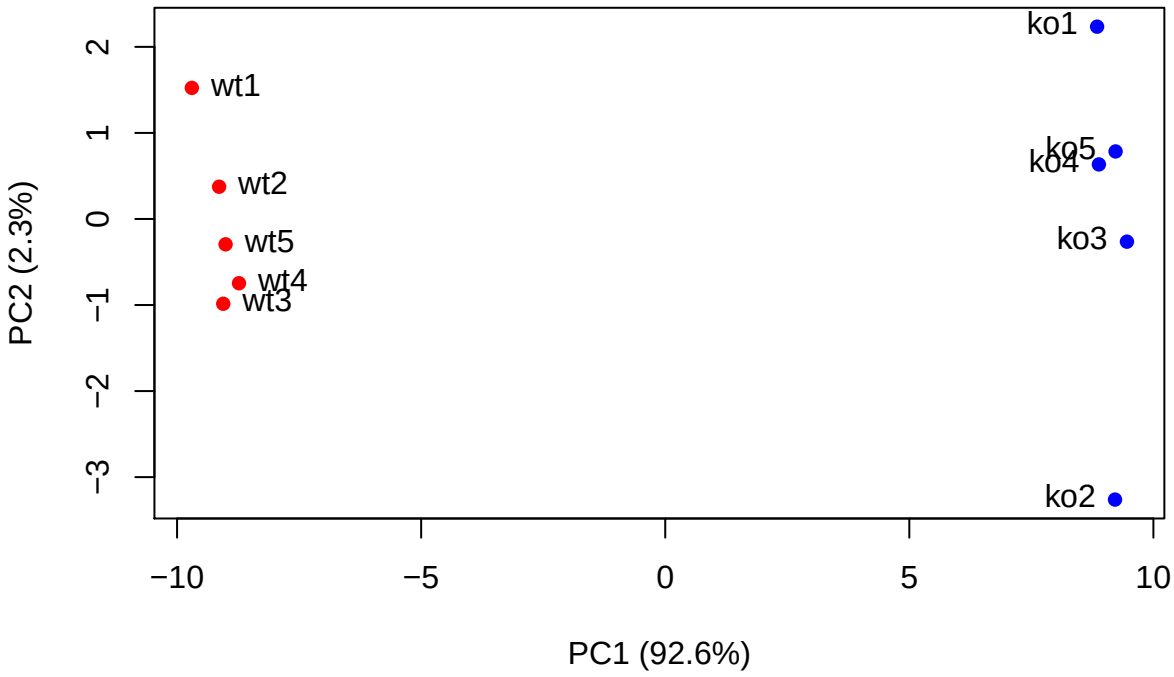

```
pca.var <- pca$sdev^2

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
```

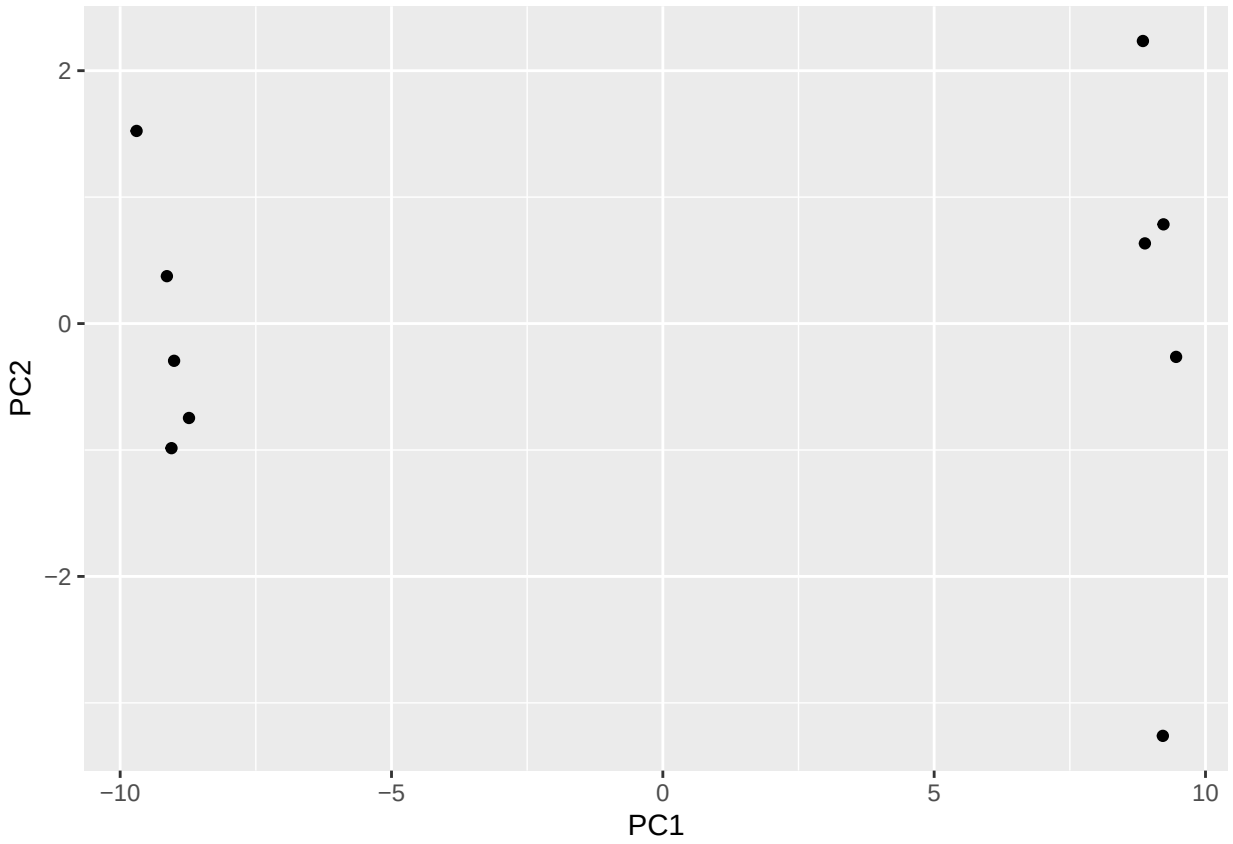
```
barplot(pca.var.per, main="Scree Plot",
        names.arg = paste0("PC", 1:10),
        xlab="Principal Component", ylab="Percent Variation")
```



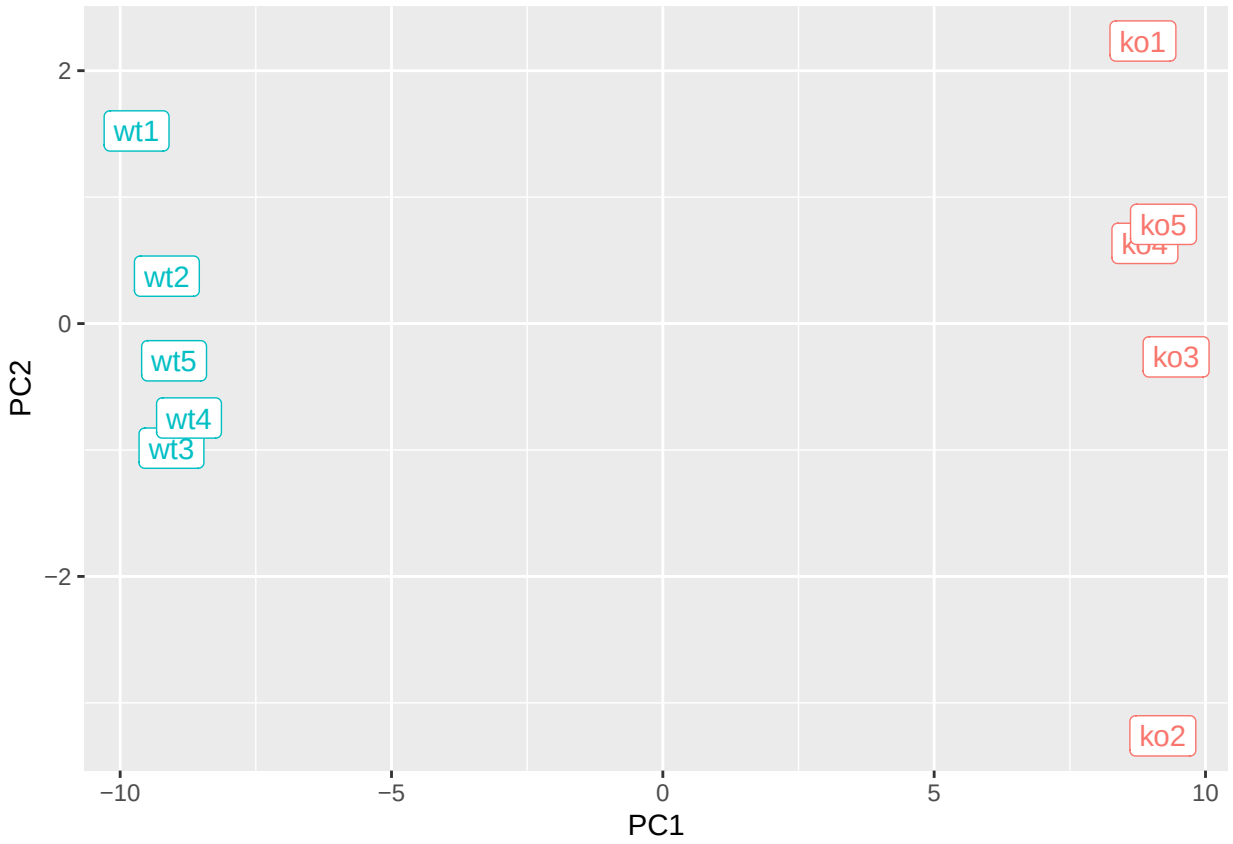

```
colvec <- colnames(rna.data)
colvec[grep("wt", colvec)] <- "red"
colvec[grep("ko", colvec)] <- "blue"

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)))
```



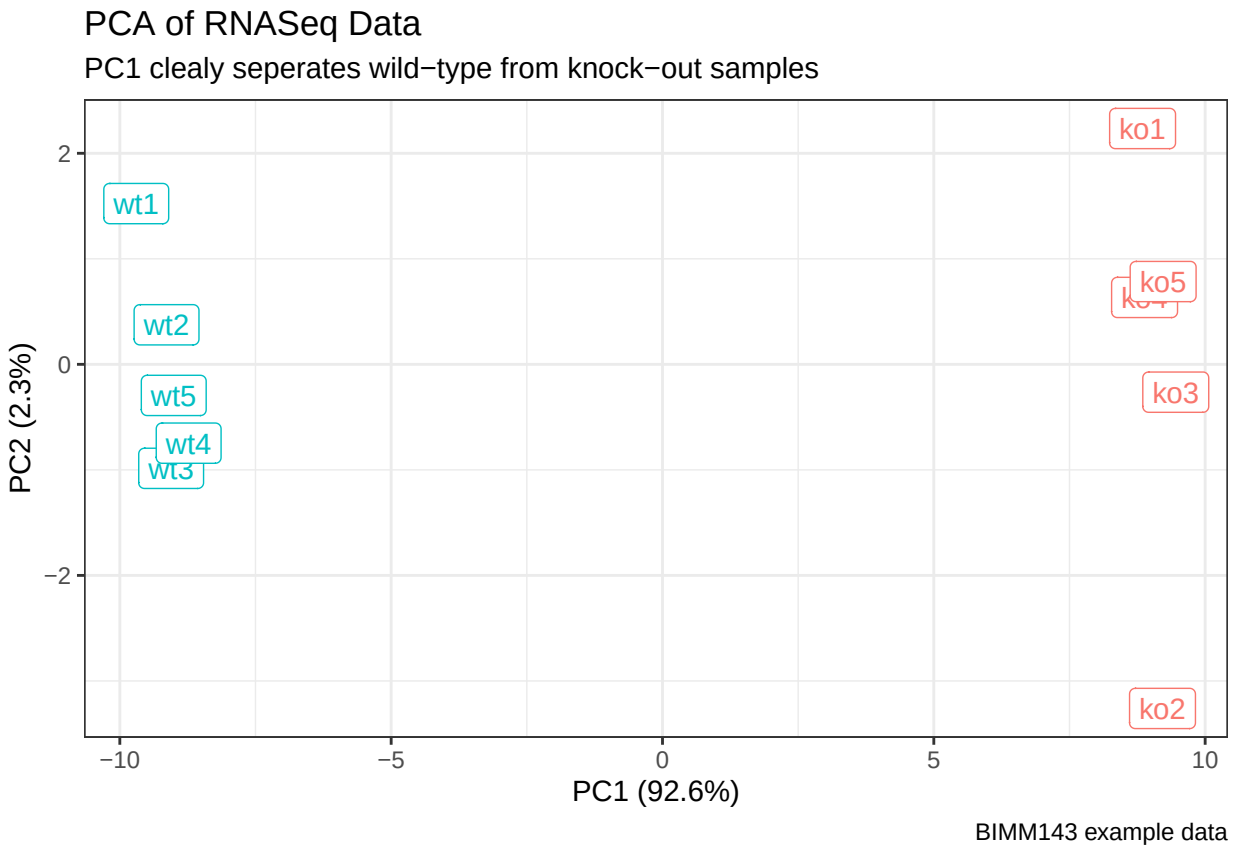

```
library(ggplot2)

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

ggplot(df) +
  aes(PC1, PC2) +
  geom_point()
```




```
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()
```




```
loading_scores <- pca$rotation[,1]

gene_scores <- abs(loading_scores)
gene_score_ranked <- sort(gene_scores, decreasing=TRUE)

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

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

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.