Lab7

Chan-yu Kuo

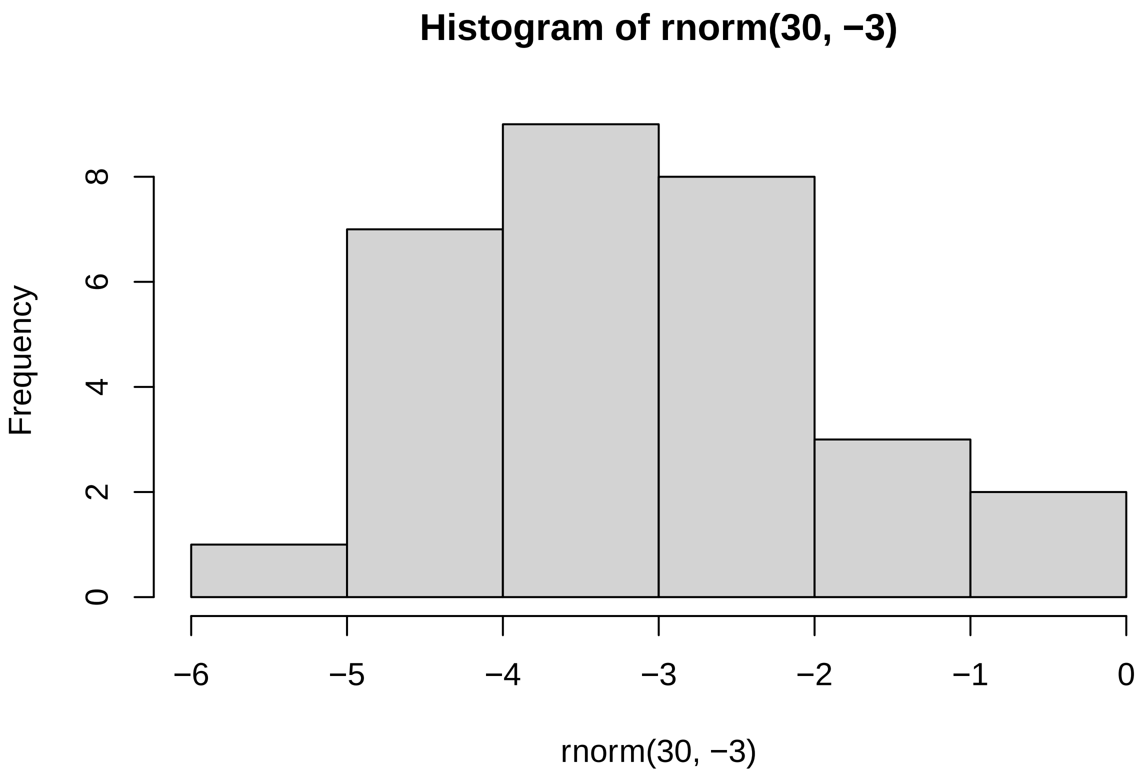
2023-01-31

Explore clustering and dimensionality reduction

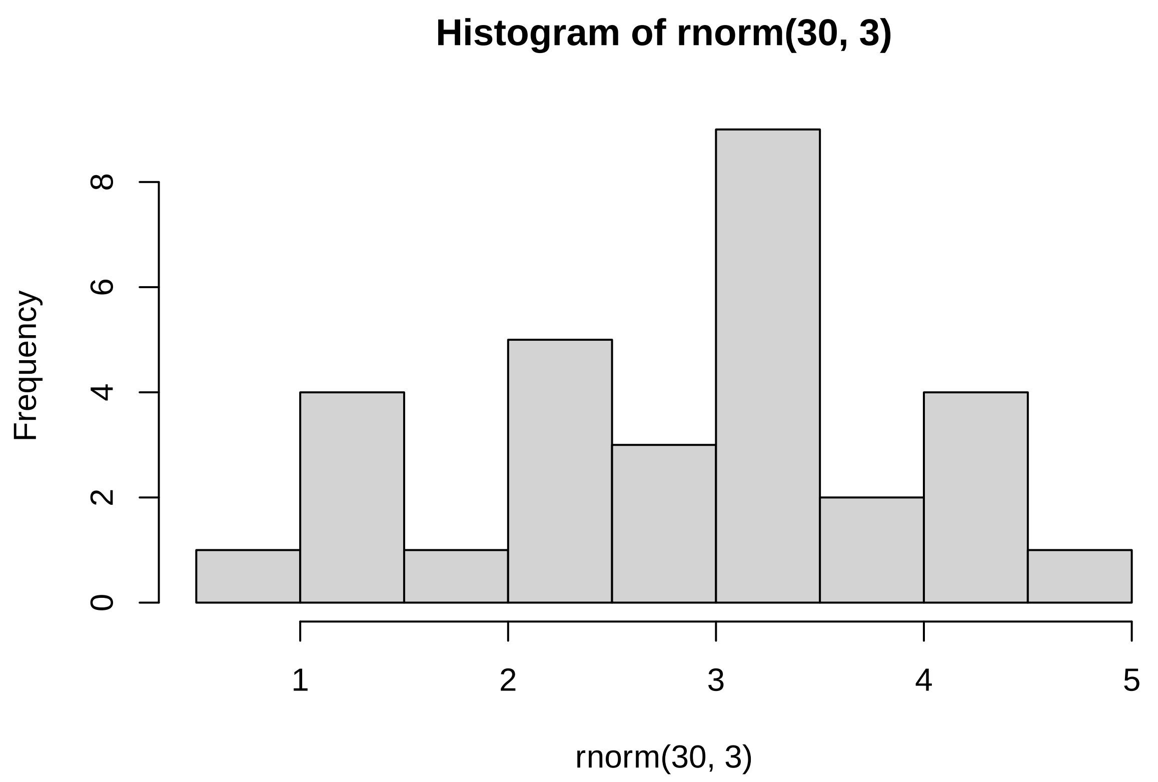
K means number of clusters

```
## K-means

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



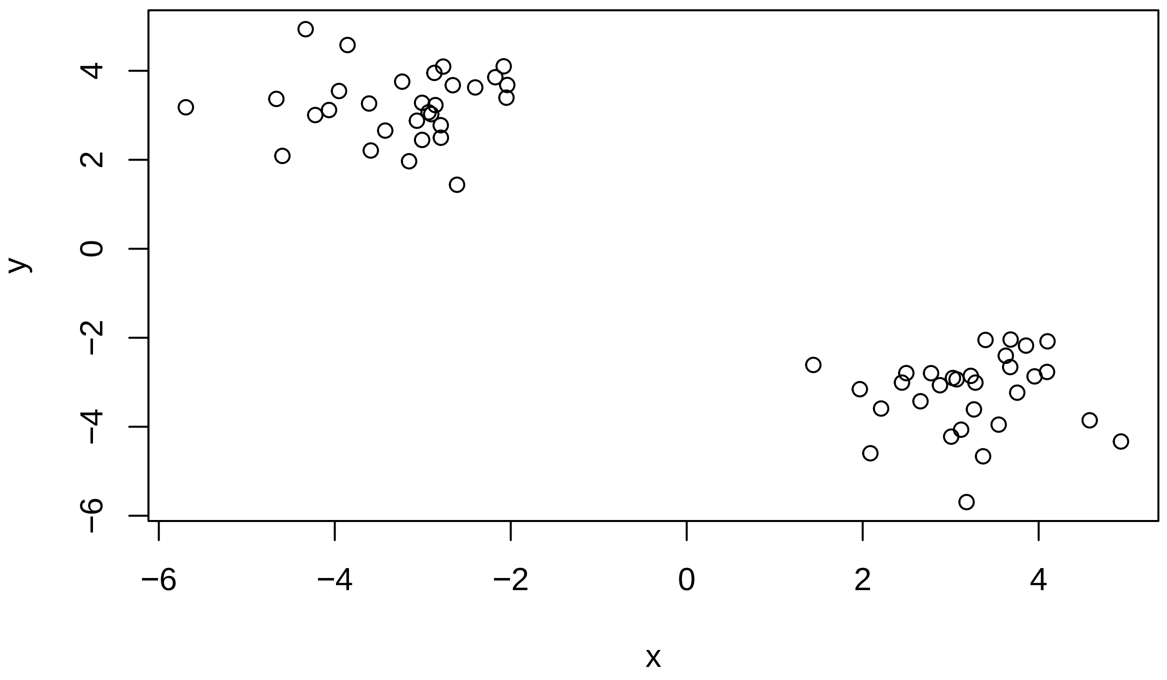
```
hist(rnorm(30,3))
```



```r
## use to make random number in normal distribution
tmp<-c(rnorm(30,-3),rnorm(30,3))
x<-cbind(x=tmp,y=rev(tmp))
head(x)
```

```
##              x        y
## [1,] -3.854994 4.579709
## [2,] -4.595487 2.087296
## [3,] -3.610092 3.264432
## [4,] -2.768310 4.094820
## [5,] -4.664032 3.368134
## [6,] -2.904158 3.025511
```

```r
plot(x)
```



```
## they are supposed by be +- 3 on both x and y, reread how x and y are set up if not sure. X= -3 ascen
```

```
km<-kmeans(x,centers=2,nstart=10)
## k == 2, have two center, it will find the distance of any given point and classify into either of th
km
```

```
## K-means clustering with 2 clusters of sizes 30, 30
##
## Cluster means:
##           x         y
## 1  3.223273 -3.247195
## 2 -3.247195  3.223273
##
## 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] 39.15405 39.15405
##  (between_SS / total_SS =  94.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

how many points in each cluster:

```
km$size
```

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

component of your result object details cluster assignment: cluster vector cluster center: centers

```
km$cluster
```

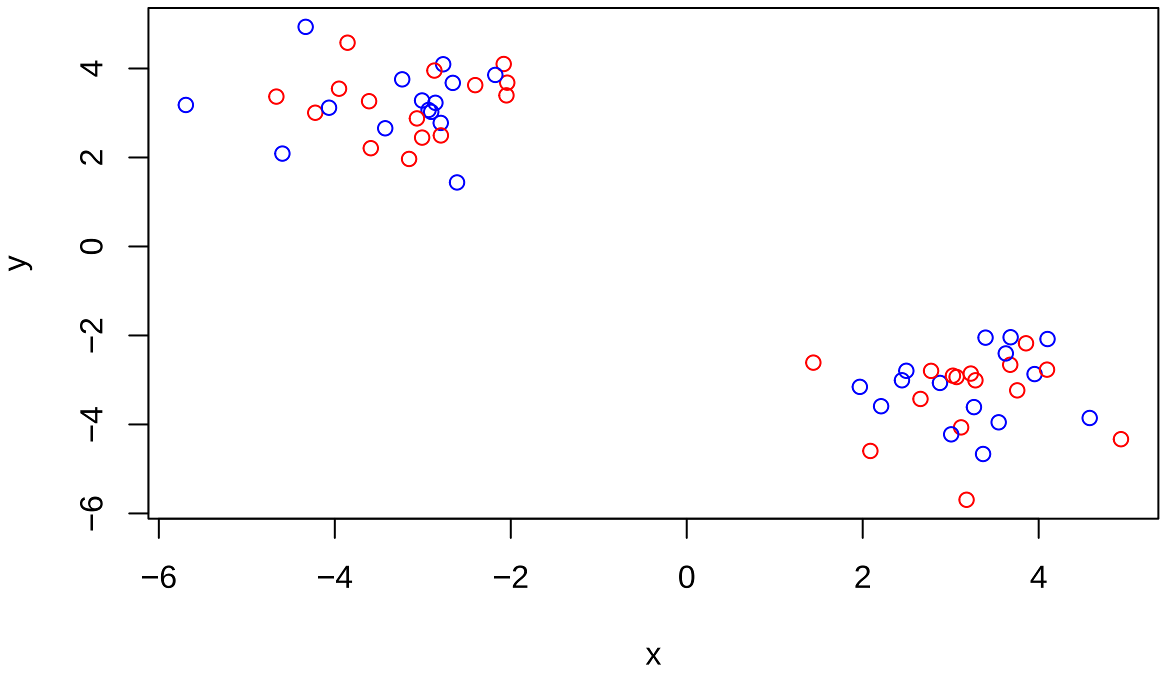
```
##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
km$centers
```

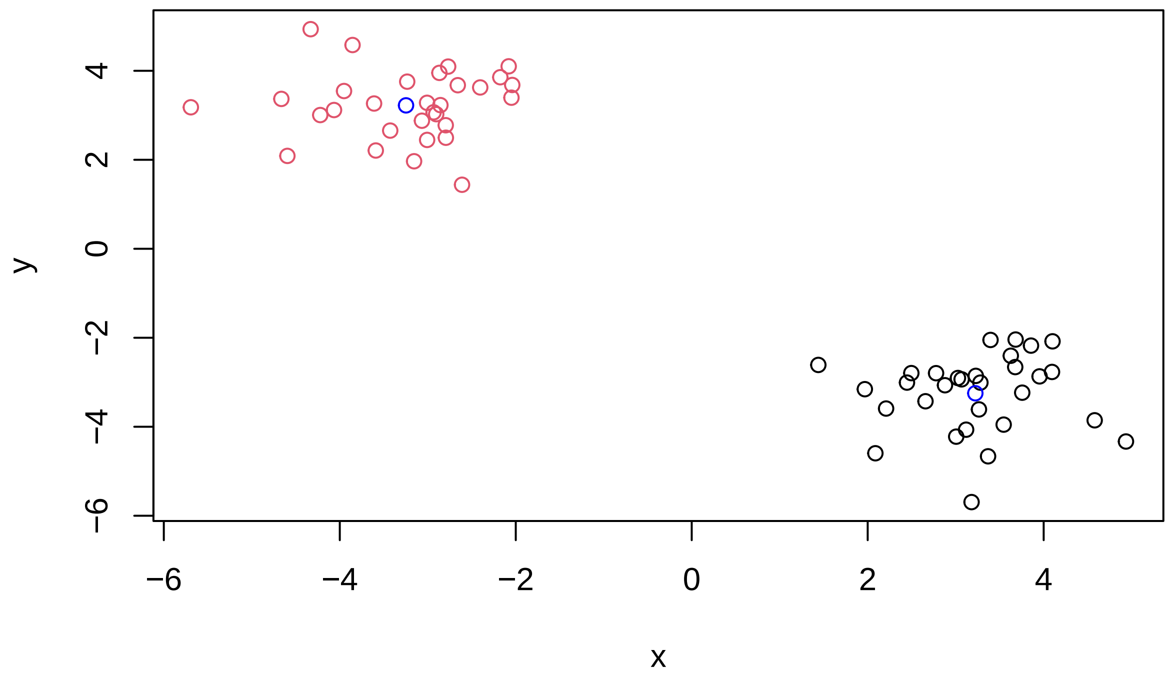
```
##           x         y
## 1  3.223273 -3.247195
## 2 -3.247195  3.223273
```

plot x colored by the kmeans cluster assignment and add cluster centers as blue points

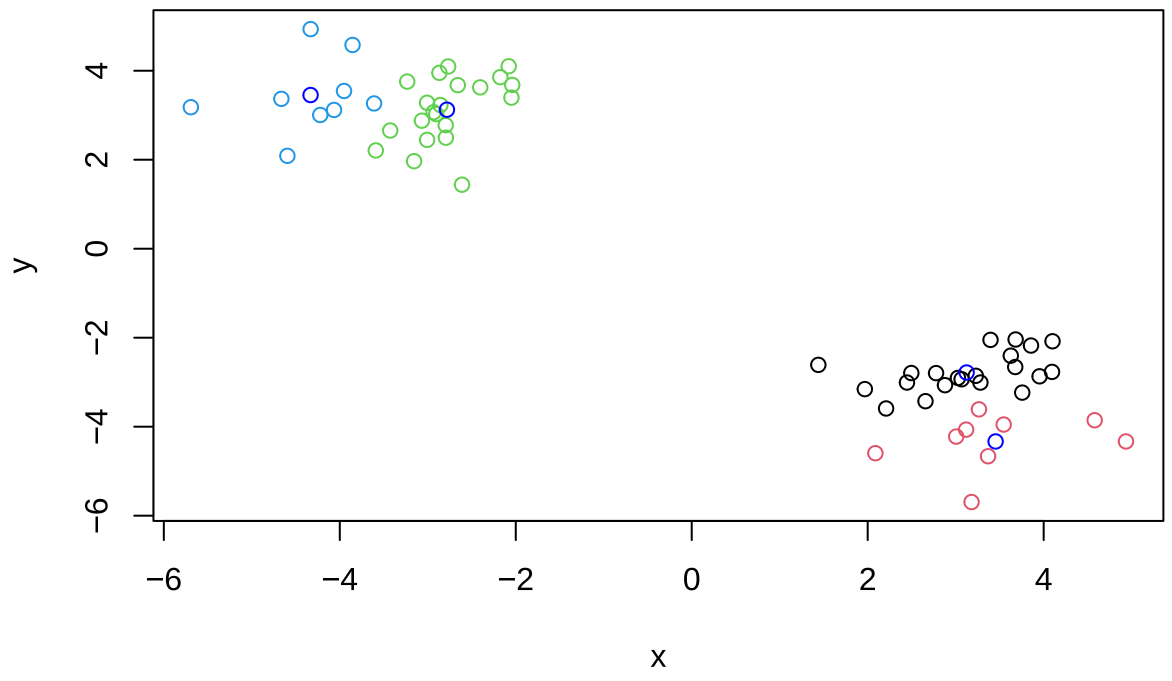
```
plot(x,col=c("red","blue"))## raw data visualize
```



```
plot(x,col=km$cluster)## km$cluster gives vector 1 and 2, indicating different cluster, in vector, from
points(km$centers,col="blue")## add points at the centers
```



```
km<-kmeans(x,centers=4,nstart=20)
plot(x,col=km$cluster)
points(km$centers,col="blue")
```



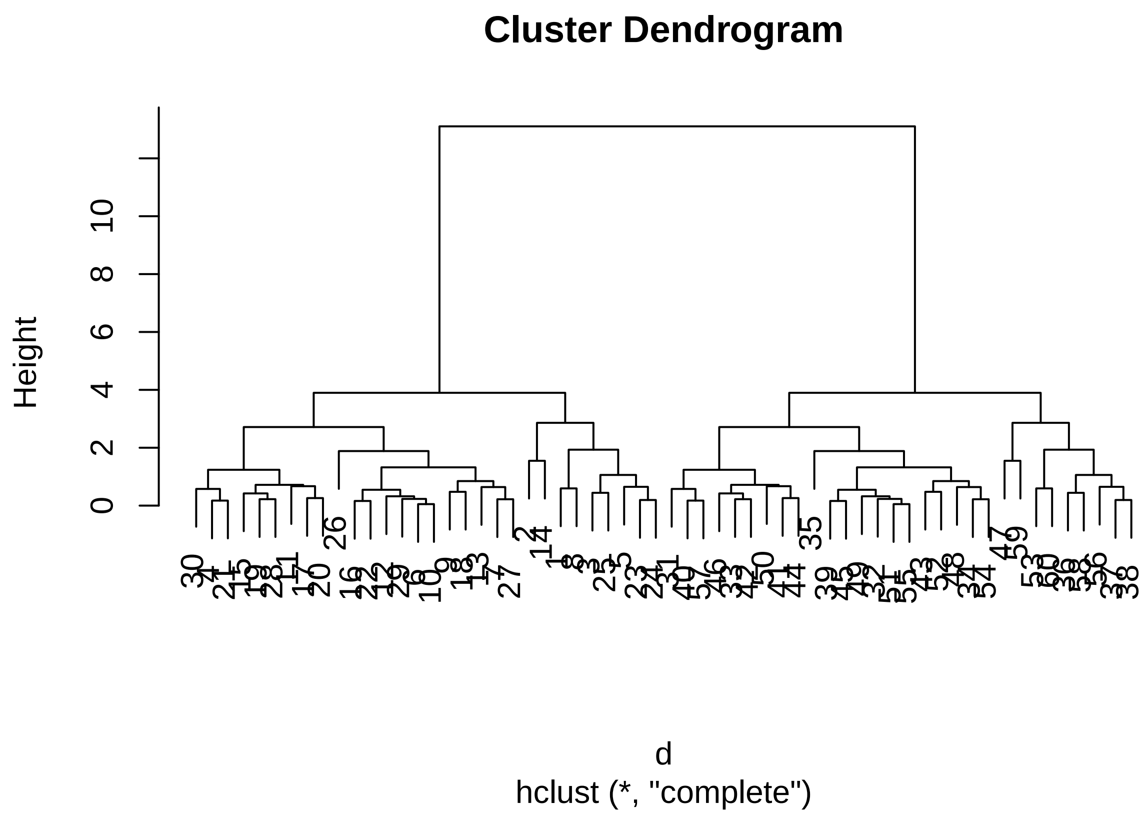
Hierarchical Clustering This is another very useful and widely employed clustering method.

```
d<-dist(x)## d is the distance, if having two points, the distance between the point will be calculated
hc<-hclust(d)
hc
```

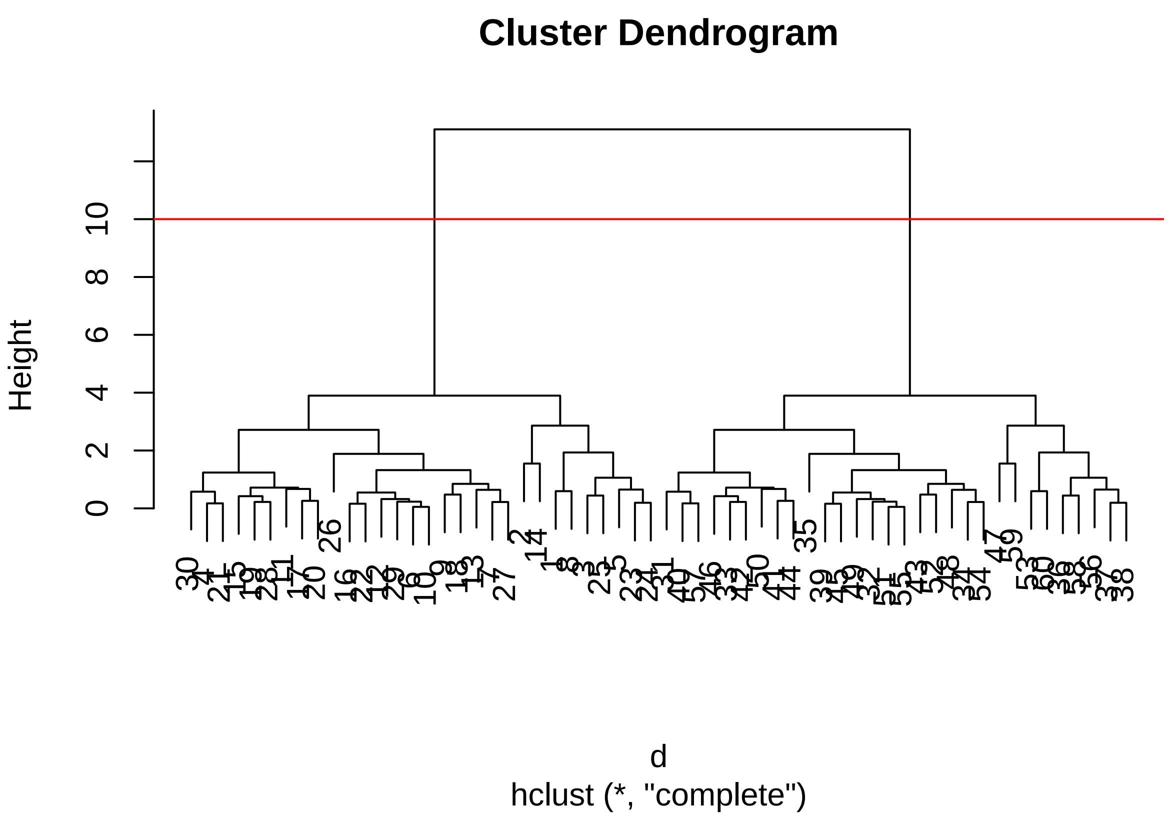
```
##
## Call:
```

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

```
plot(hc)## What is this height?
```

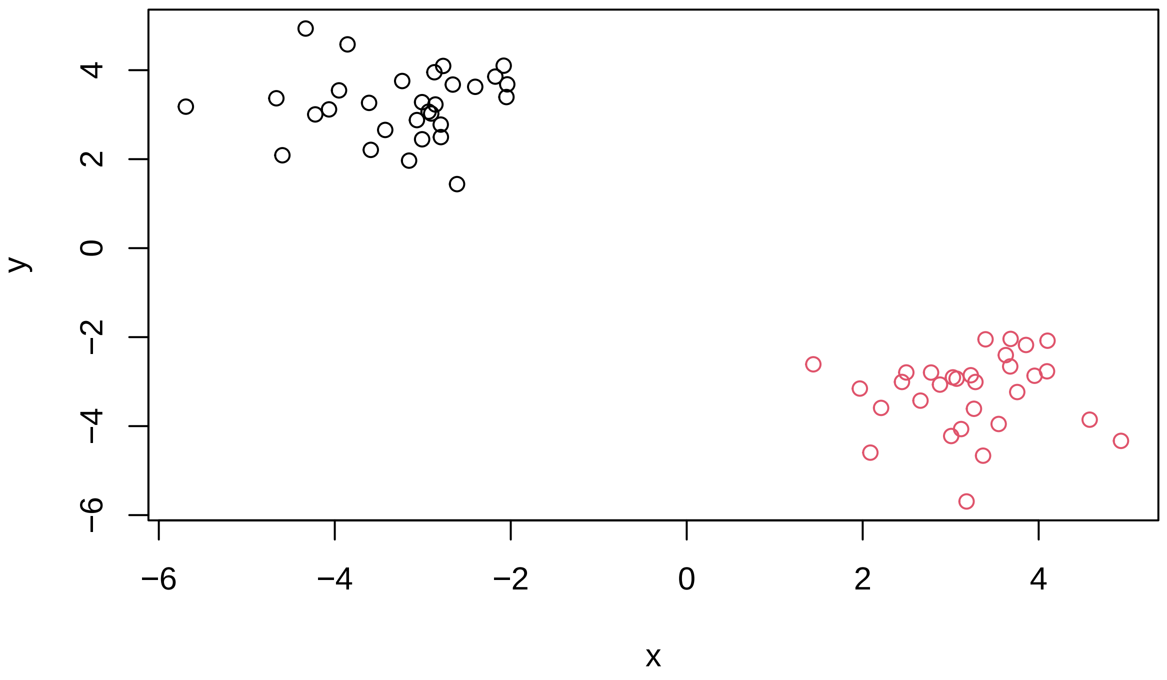


```
plot(hc)
abline(h=10,col="red")
```



Cut the tree to yield sub-tree. Put all the members within the tree into new membership

```
h<-cutree(hc,h=10)
plot(x,col=h)
```



use k= to cutree rather than h= height of cutting with cutree. k= will give the number of cluster you want.

```
newh<-cutree(hc,k=4)
newh
```

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

```
## PCA
### principle component, new low dimensional axis closest to the observations
### we always have pc1, then depend on the needs, we can have pc2, pc3 etc.
```

```
#Q1 17 rows and 5 columns, include one name column
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)
dim(x)
```

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

```
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
```

```
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
```

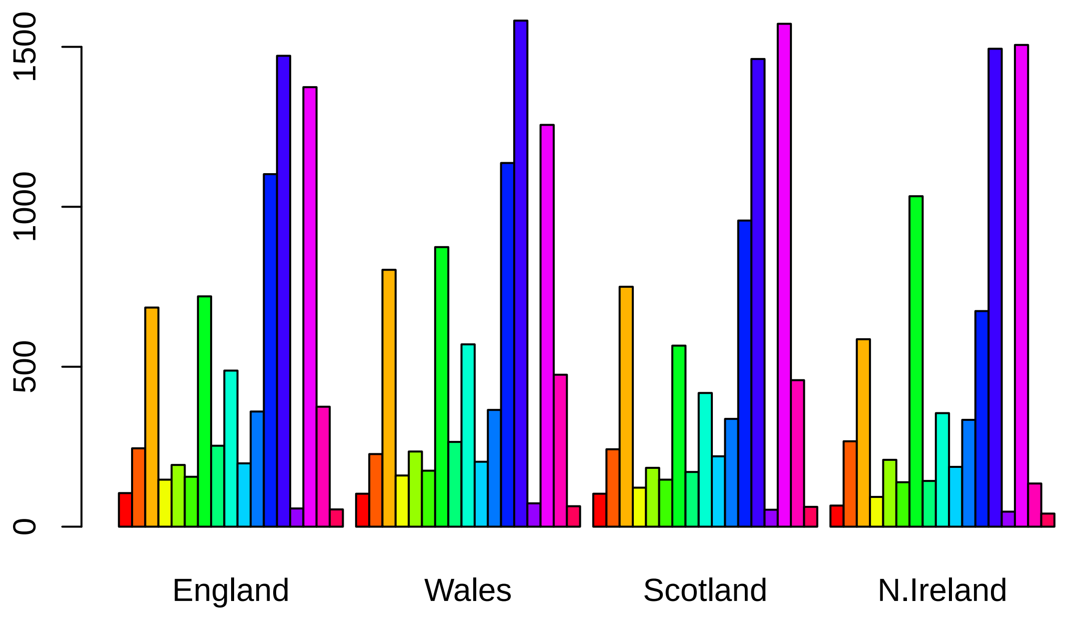
```
##Q2
dim(x)
```

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

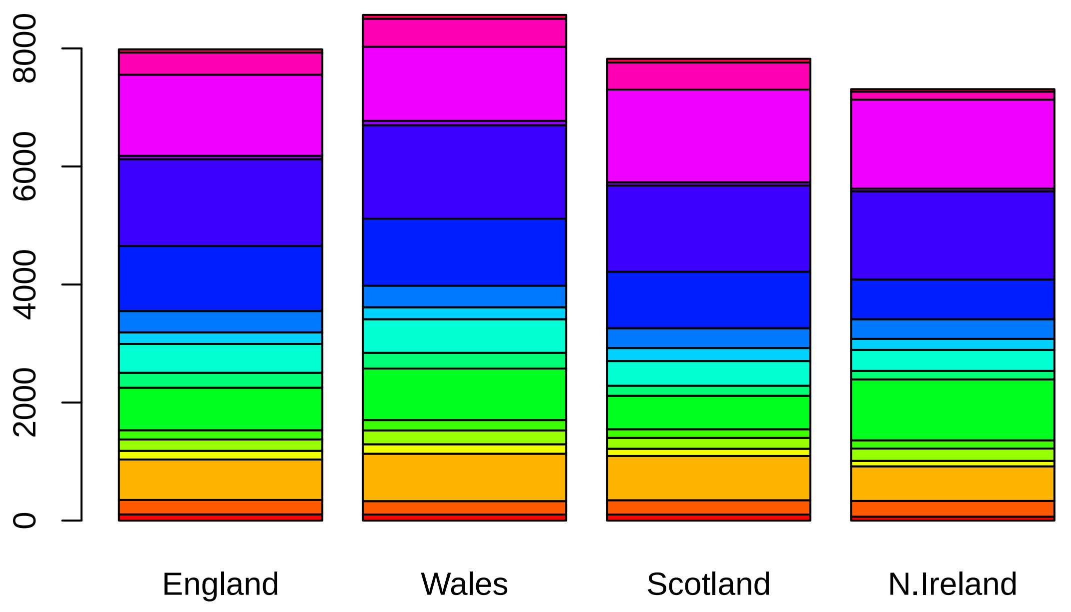
```
x <- read.csv(url, row.names=1) ## this helps reduce steps we need to assign names. This is faster and
head(x)
```

```
##               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)))
```



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

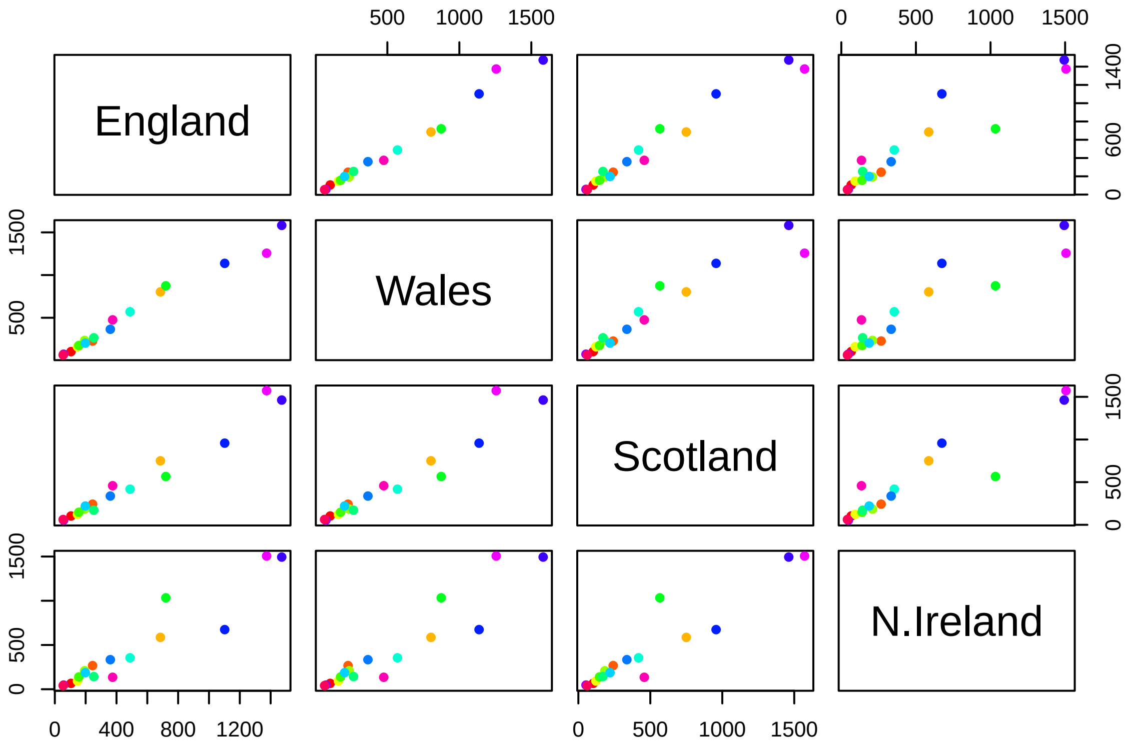


```
## q3, change the beside from true to false will result in this plot
```

```
str(x)
```

```
## 'data.frame':    17 obs. of  4 variables:
##  $ England  : int  105 245 685 147 193 156 720 253 488 198 ...
##  $ Wales    : int  103 227 803 160 235 175 874 265 570 203 ...
##  $ Scotland : int  103 242 750 122 184 147 566 171 418 220 ...
##  $ N.Ireland: int  66 267 586 93 209 139 1033 143 355 187 ...
```

```
pairs(x, col=rainbow(17), pch=16)
```



```
##Q5 x is the inputm col is the color of each point ( in rows), pch specify the color.
## pairs essentially gather all the value of a row from the 4 column, and compare each of them by displ

## a diagnole line means the comparsion between two contry is small (they are similar)
## N Ireland have highest difference in cost , but can't tell the detail on what the color represent if
cbind(row.names(x),rainbow(17))
```

```
##       [,1]                     [,2]
##  [1,] "Cheese"                 "#FF0000"
##  [2,] "Carcass_meat "          "#FF5A00"
##  [3,] "Other_meat "            "#FFB400"
##  [4,] "Fish"                   "#F0FF00"
##  [5,] "Fats_and_oils "         "#96FF00"
##  [6,] "Sugars"                 "#3CFF00"
##  [7,] "Fresh_potatoes "        "#00FF1E"
##  [8,] "Fresh_Veg "             "#00FF78"
##  [9,] "Other_Veg "             "#00FFD2"
## [10,] "Processed_potatoes "    "#00D2FF"
## [11,] "Processed_Veg "         "#0078FF"
## [12,] "Fresh_fruit "           "#001EFF"
## [13,] "Cereals "               "#3C00FF"
## [14,] "Beverages"              "#9600FF"
## [15,] "Soft_drinks "           "#F000FF"
## [16,] "Alcoholic_drinks "      "#FF00B4"
## [17,] "Confectionery "         "#FF005A"
```

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

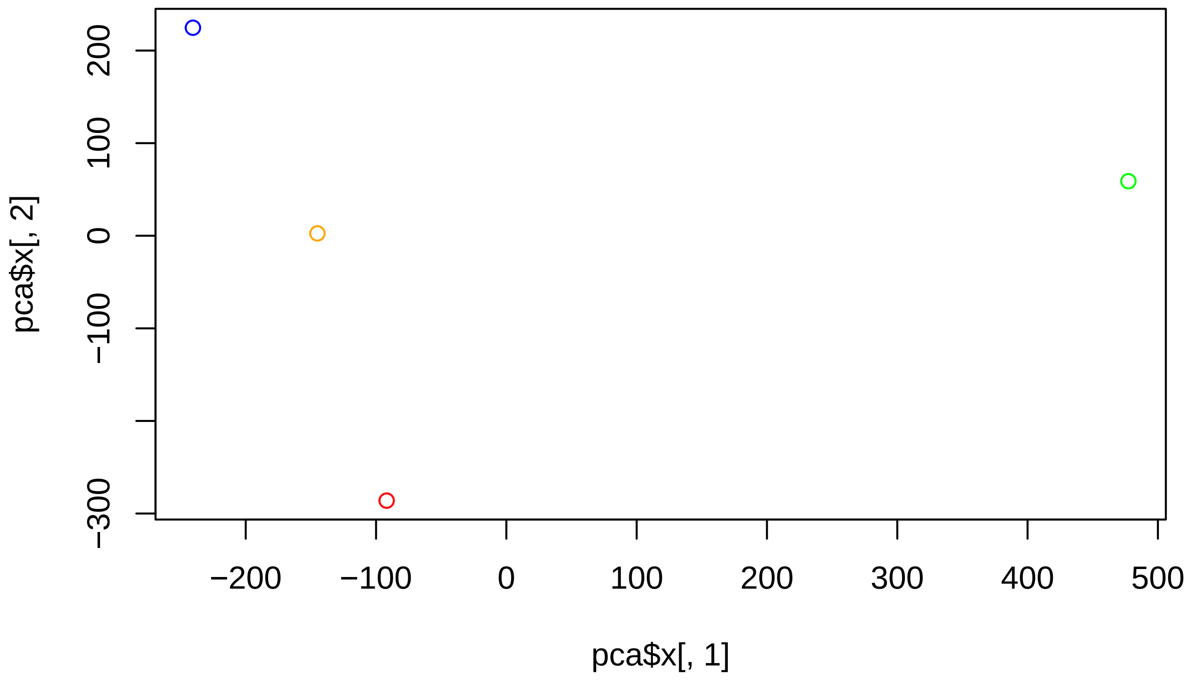
```
## Importance of components:
##                              PC1       PC2       PC3        PC4
## Standard deviation     324.1502 212.7478 73.87622 5.552e-14
## Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
## Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

```
## proportion of variance, 67.44 % of data point will be shown based on the PC.
## Cumulative proportion: add up from the proportion of variation.
```

```
## PCA plot (a.k.a) Score plot PC1 VS PC2
 pca$x
```

```
##                 PC1         PC2         PC3           PC4
## England   -144.99315    2.532999 -105.768945  1.042460e-14
## Wales     -240.52915  224.646925   56.475555  9.556806e-13
## Scotland   -91.86934 -286.081786   44.415495 -1.257152e-12
## N.Ireland  477.39164   58.901862    4.877895  2.872787e-13
```

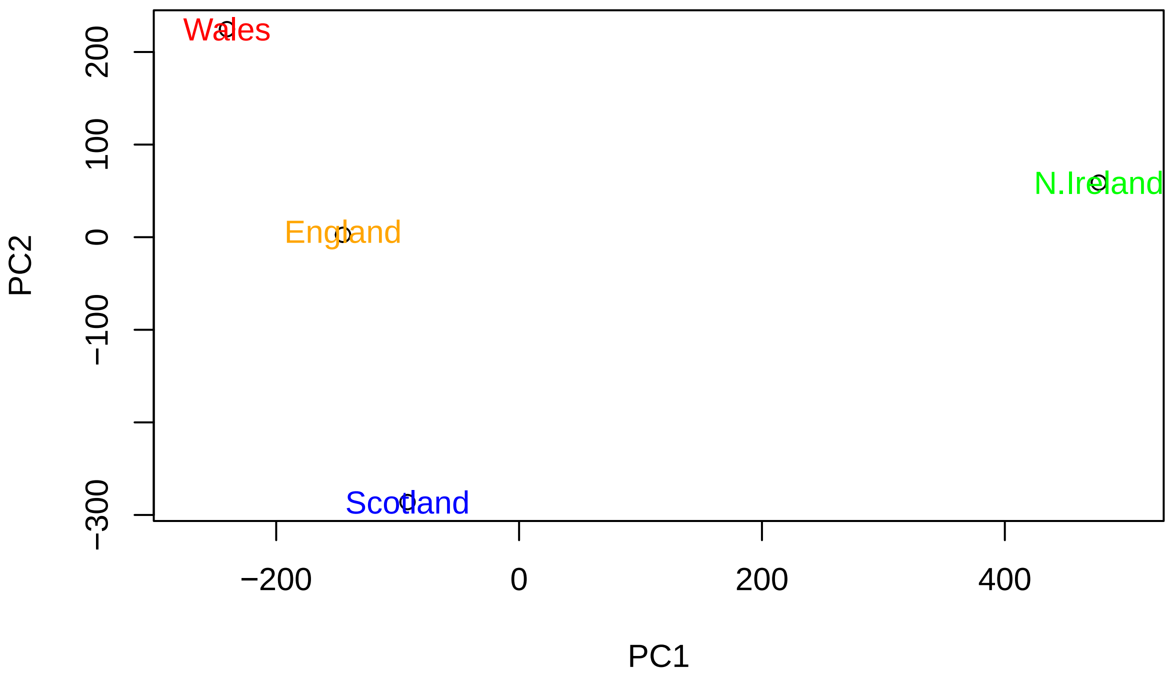
```
plot(pca$x[,1],pca$x[,2],col=c("orange","blue","red","green"))
```



```
## green represent N ireland, shows further away
```

z

```
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","red","blue","green"))
```



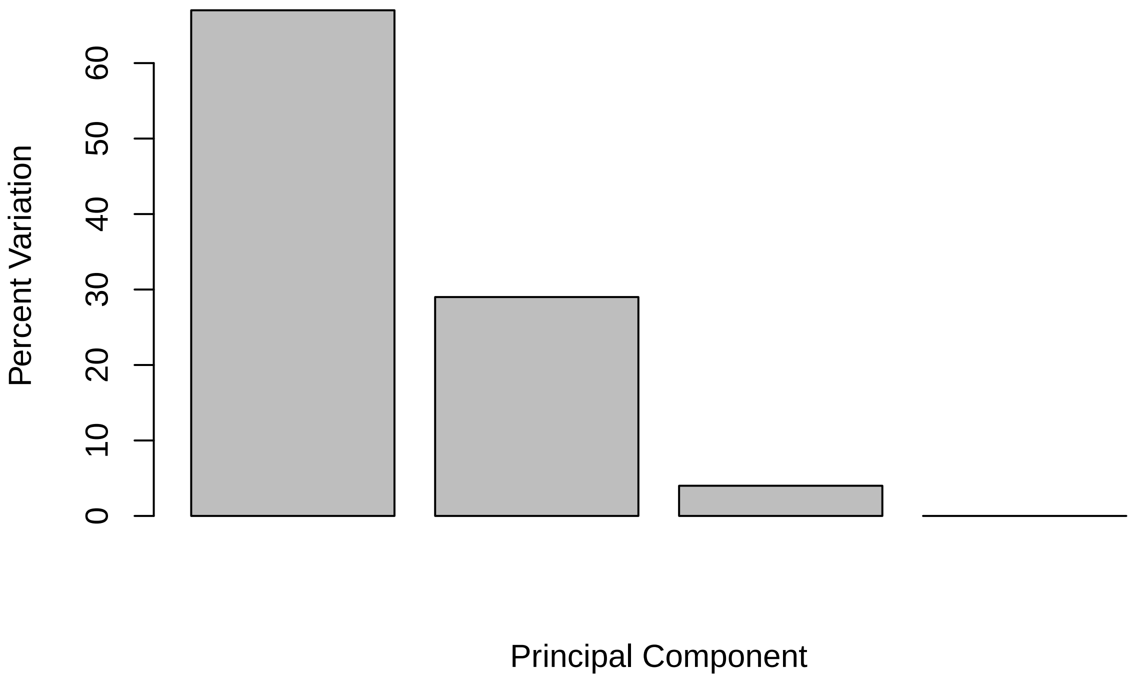
```
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

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

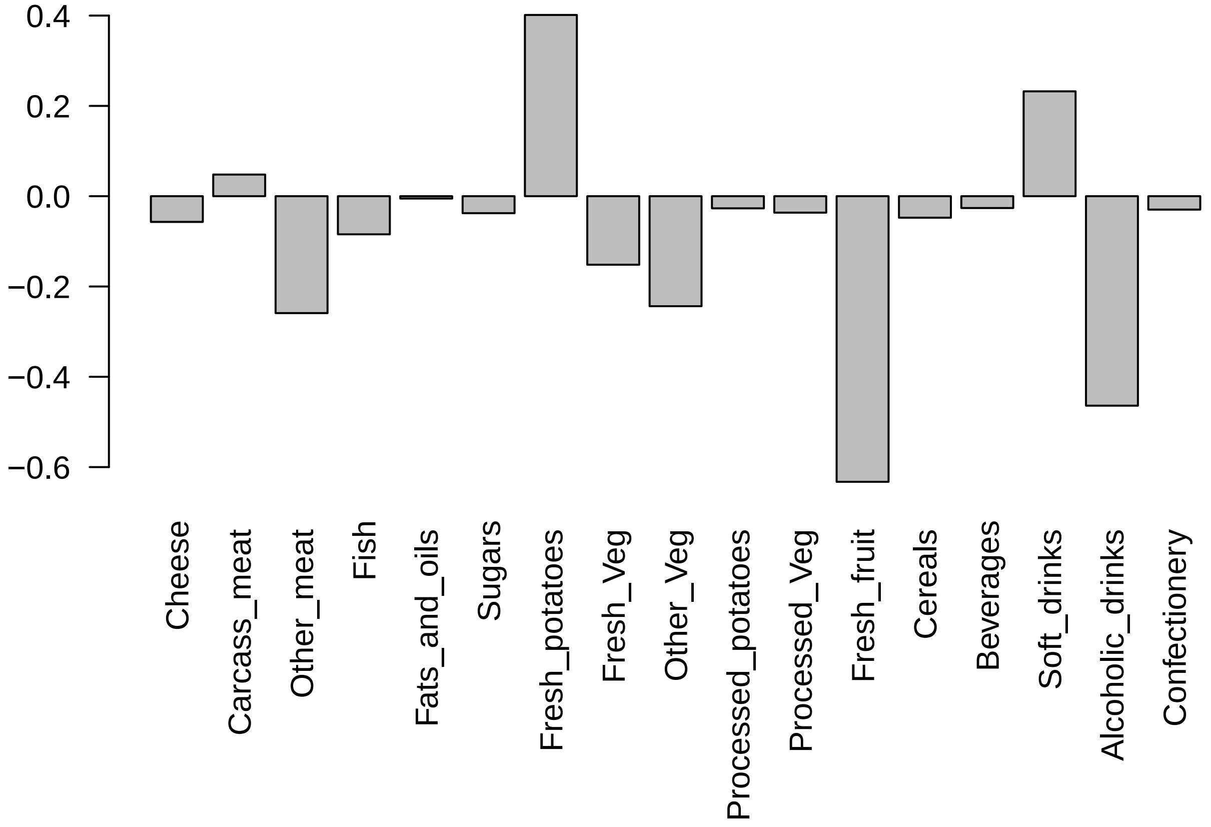
```
## or the second row here...
z <- summary(pca)
z$importance
```

```
##                              PC1       PC2      PC3          PC4
## Standard deviation     324.15019 212.74780 73.87622 5.551558e-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")
```



```
## 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 )
```


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