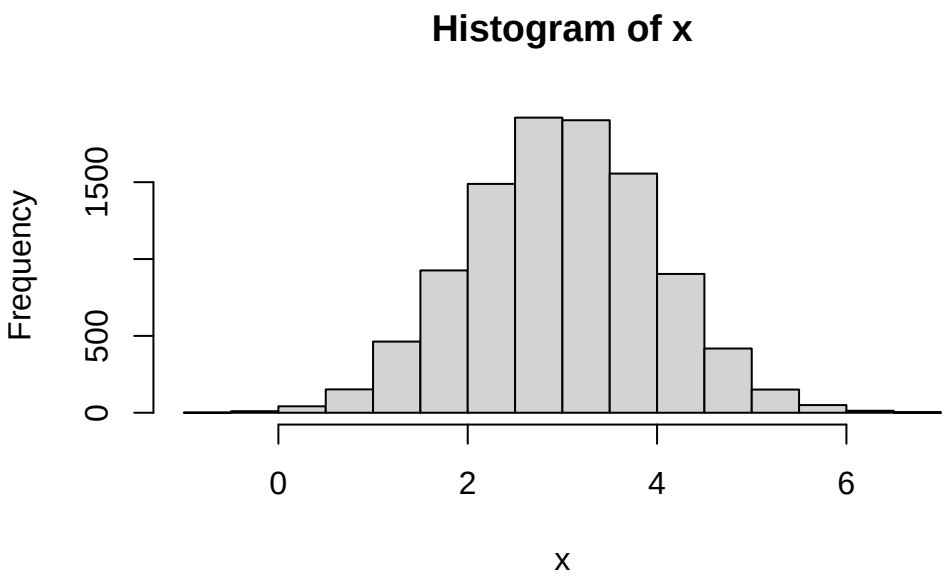
class07 machine learning 1

Jacob Gil

Clustering

with kmeans()

```
x <- rnorm(10000, mean = 3)
hist(x)
```




```
#x
```

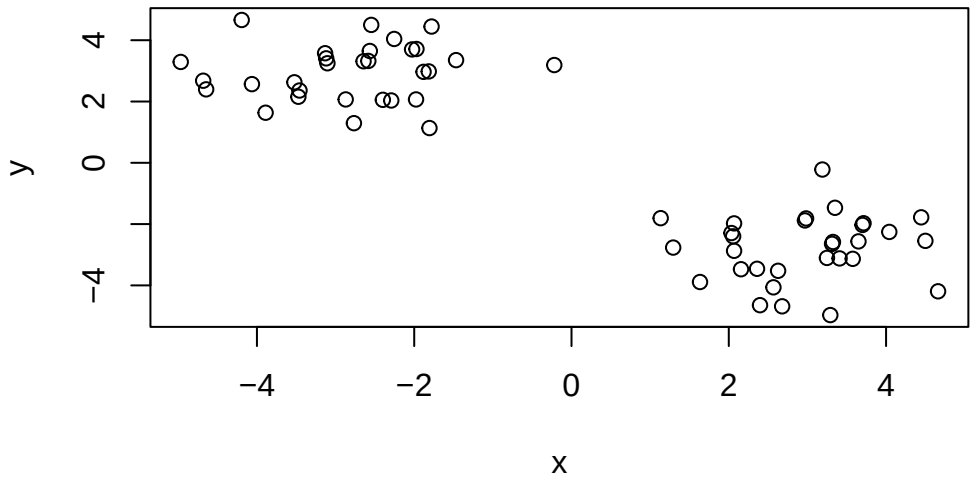
```
tmp <- c(rnorm(30, mean = 3), rnorm(30, -3))
tmp
```

```
 [1]  2.155288  2.982539  3.290923  4.446526  2.067196  4.661107  3.190259
 [8]  4.499575  3.713370  2.396917  1.634199  2.968354  3.313198  1.133783
[15]  3.324566  2.567446  3.700489  1.292953  3.248013  2.359077  3.648830
[22]  2.678407  3.576094  2.055263  2.067462  3.350044  4.040927  3.408771
[29]  2.626796  2.033881 -2.292019 -3.523945 -3.119440 -2.255122 -1.468024
[36] -2.871398 -2.398890 -3.133150 -4.683065 -2.564426 -3.458925 -3.103681
[43] -2.766703 -2.027143 -4.063780 -2.583537 -1.806369 -2.644184 -1.881428
[50] -3.889900 -4.645925 -1.971650 -2.546598 -0.219145 -4.193841 -1.977014
[57] -1.779493 -4.969072 -1.815088 -3.472990
```

```r
x <- cbind(x = tmp, y = rev(tmp))
head(x)
```

```
           x         y
[1,] 2.155288 -3.472990
[2,] 2.982539 -1.815088
[3,] 3.290923 -4.969072
[4,] 4.446526 -1.779493
[5,] 2.067196 -1.977014
[6,] 4.661107 -4.193841
```

```r
plot(x)
```



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

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

Cluster means:
          x         y
1 -2.804198  2.947742
2  2.947742 -2.804198

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 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] 57.53005 57.53005
 (between_SS / total_SS =  89.6 %)

Available components:

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

Q1. How many points in each cluster

```
k$size
```

```
[1] 30 30
```

Q2. CLuster membership?

```
k$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 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
```

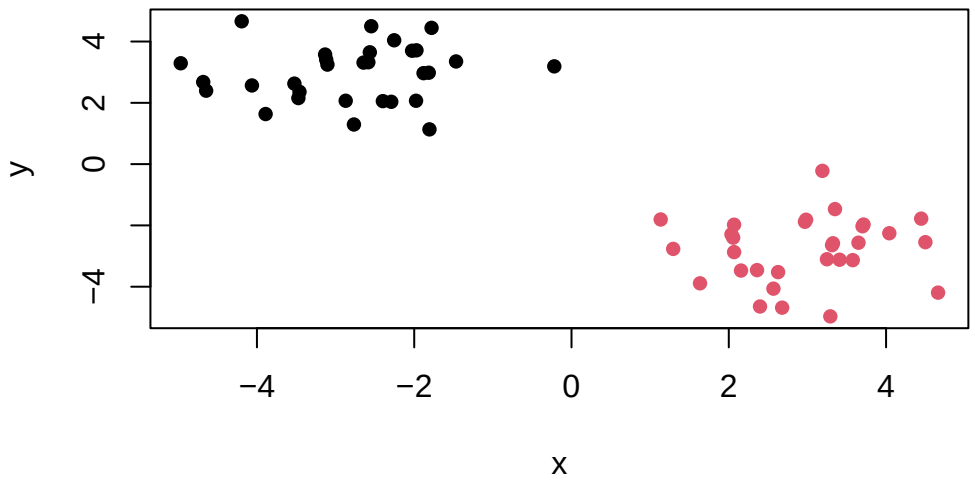
Q3. Cluster centers?

```
k$centers
```

```
          x         y
1 -2.804198  2.947742
2  2.947742 -2.804198
```

Q4. Plot my cluster

```
plot(x, col=k$cluster, pch = 16)
```



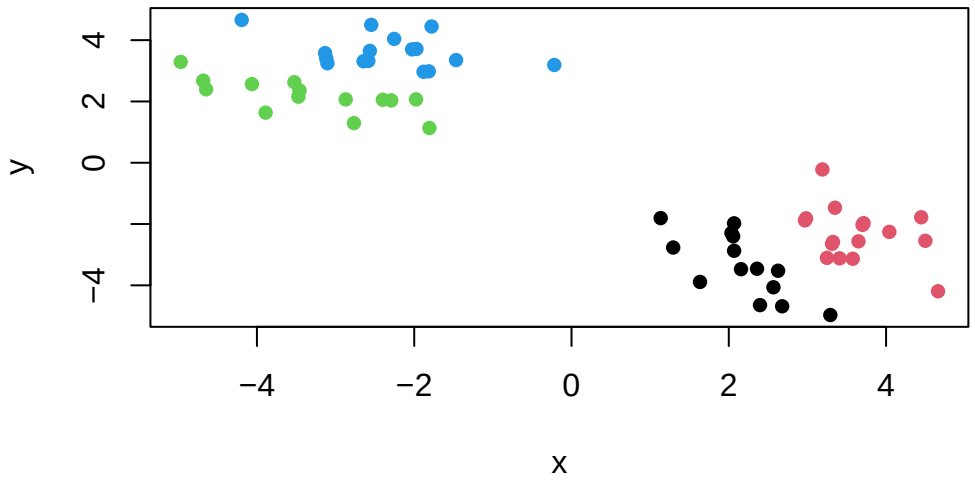
Q5. Cluster the data into 4 groups with kmeans and plot

```
k4 <- kmeans(x, centers = 4, nstart = 20)
k4
```

```
K-means clustering with 4 clusters of sizes 14, 16, 14, 16

Cluster means:
          x         y
1  2.168542 -3.344285
2  3.629541 -2.331622
3 -3.344285  2.168542
4 -2.331622  3.629541
```

```
Clustering vector:
 [1] 1 2 1 2 1 2 2 2 2 1 1 2 2 1 2 1 2 1 2 1 2 1 2 1 1 2 2 2 1 1 3 3 4 4 4 3 3 4
[39] 3 4 3 4 3 4 3 4 3 4 4 3 3 4 4 4 4 3 4 3 4 3

Within cluster sum of squares by cluster:
[1] 17.94857 15.98677 17.94857 15.98677
 (between_SS / total_SS =  93.9 %)

Available components:

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

```r
plot(x, col=k4$cluster, pch = 16)
```



kmeans is popular because it is fast and straightforward. It is limited by knowledge of clusters.

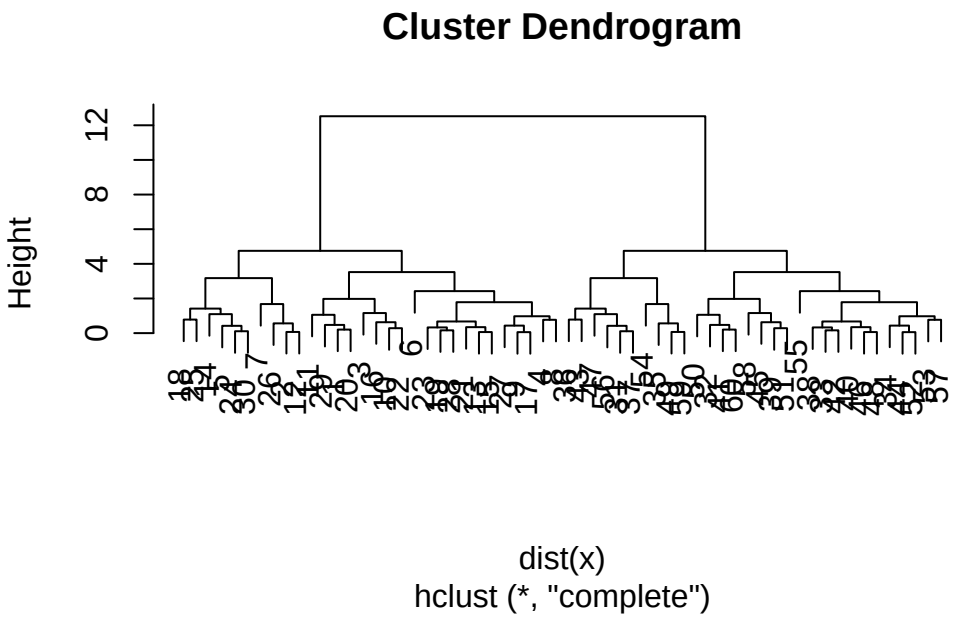
Hierarchical clustering

the main functions is called hclust() that is passed in a distance matri (dist())

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

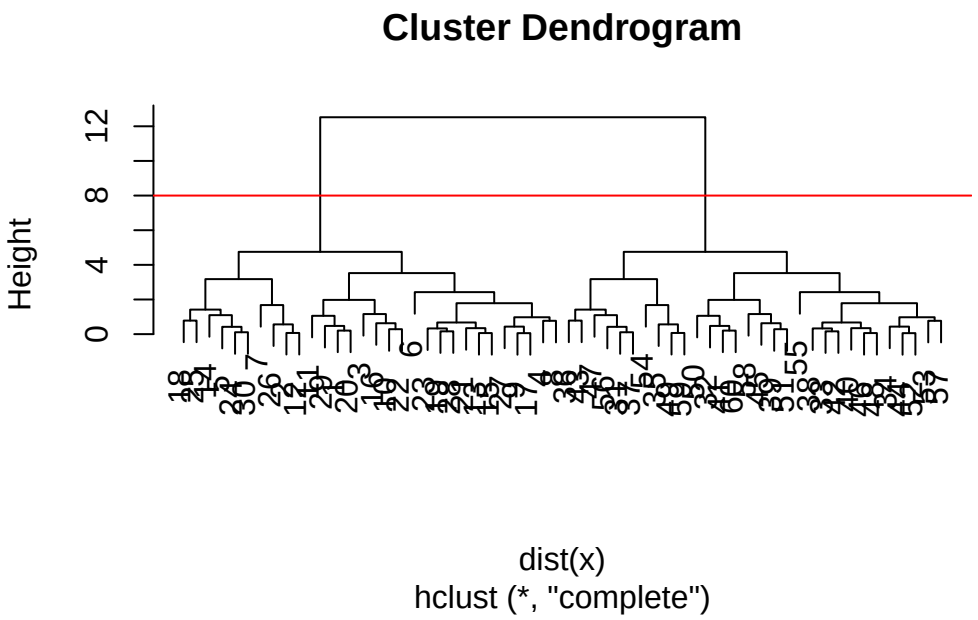
```
Call:
hclust(d = dist(x))

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

```
plot(hc)
```



to find clusters use cutree

```
plot(hc)
abline(h=8, col = "red")
```

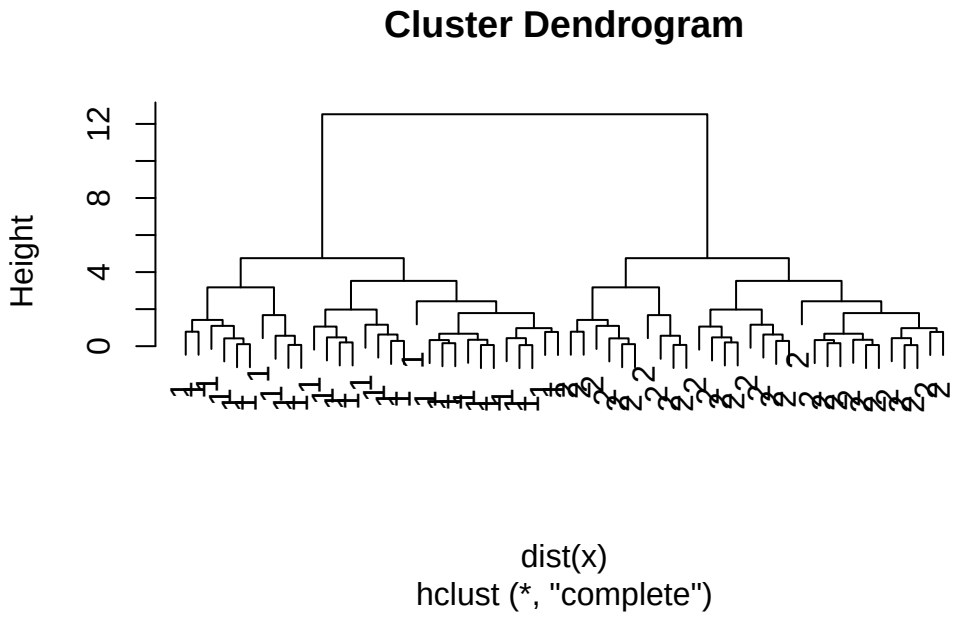



```
grps <- cutree(hc, h=8)

table(grps)
```

```
grps
 1  2
30 30
```

Q6. Plot hclust result

```
plot(hc, grps, pch = 16)
```



#Lab

```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url)
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
7	Fresh_potatoes	720	874	566	1033
8	Fresh_Veg	253	265	171	143
9	Other_Veg	488	570	418	355
10	Processed_potatoes	198	203	220	187
11	Processed_Veg	360	365	337	334
12	Fresh_fruit	1102	1137	957	674
13	Cereals	1472	1582	1462	1494
14	Beverages	57	73	53	47
15	Soft_drinks	1374	1256	1572	1506

```
16   Alcoholic_drinks      375   475      458       135
17      Confectionery       54    64       62        41
```

```
dim(x)
```

```
[1] 17  5
```

```
# Note how the minus indexing works
#rownames(x) <- x[,1]
#x <- x[,-1]
#head(x)


#' instead of ^^ use
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
```

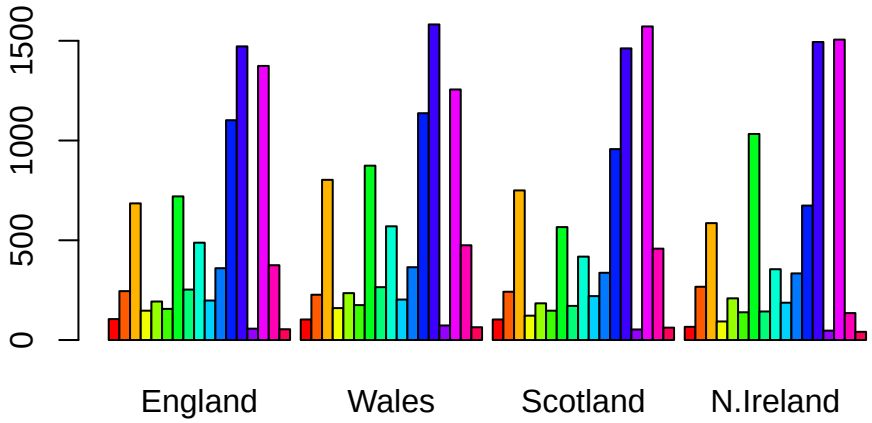
```
dim(x)
```

```
[1] 17  4
```

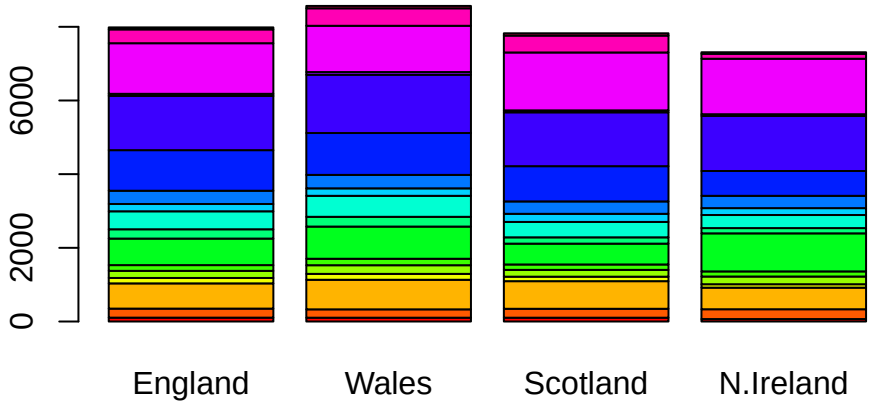
Q2.

the row.names=1 approach is better because it is harder to mess up the data in the matrix (ex. if you ran x <- x[,-1] multiple times it would start deleting your data)

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



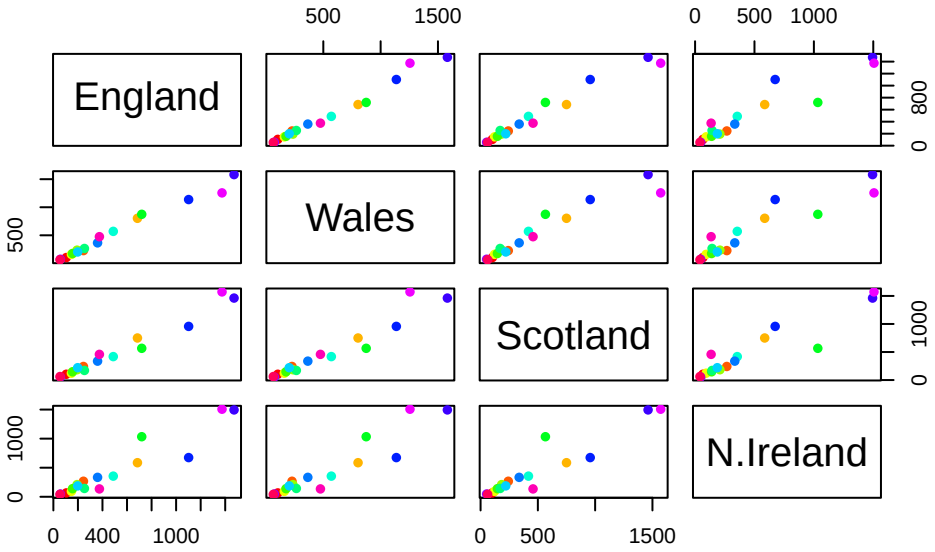

```
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```



Q3.

changing the barplot beside value to false

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



Q5.

the diagonal halves are identical. The y axis relates to the country that column and the x axis relates to the country that is on that row.

Q6.

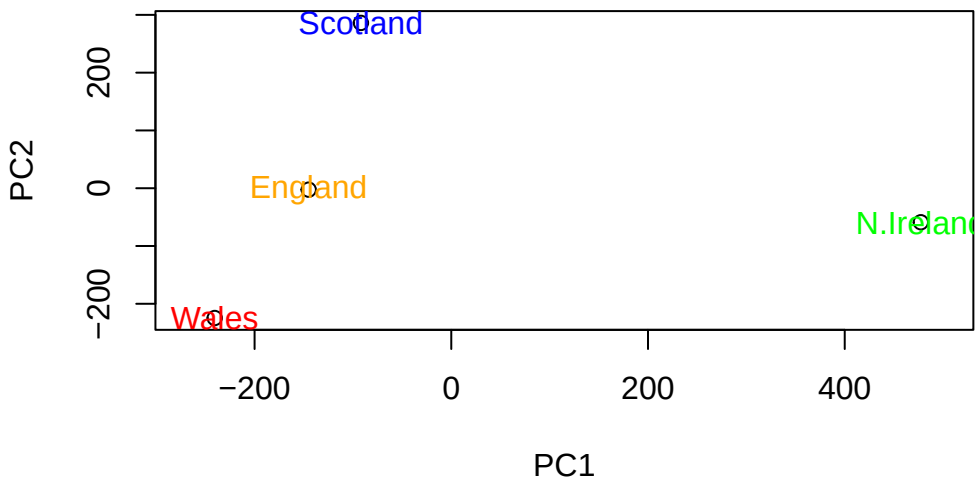
the green and blue dots in the middle of the graphs, and the pink dot on the bottom lefts are the furthest from the axis.

```
# Use the prcomp() PCA function
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
```

```
# Plot PC1 vs PC2
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"))
```

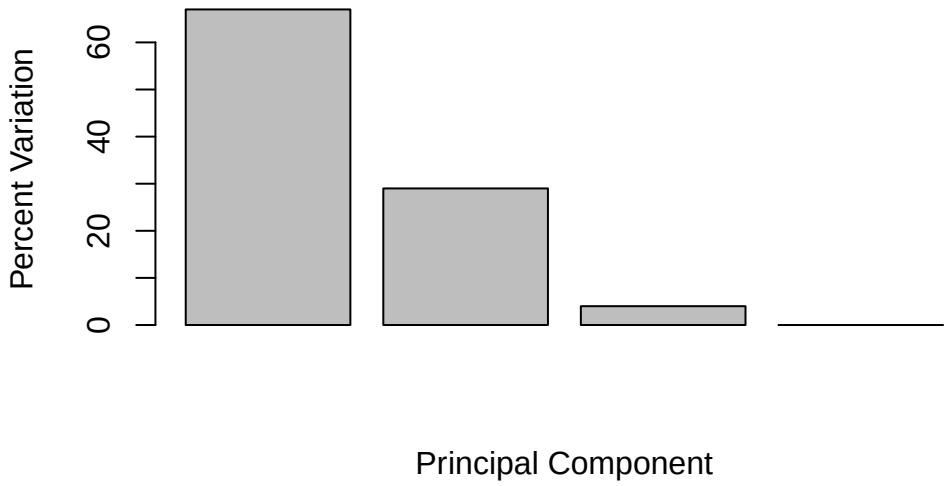


the loadings tell us how much he original variables conribute to the new variables (the PCs)

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

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

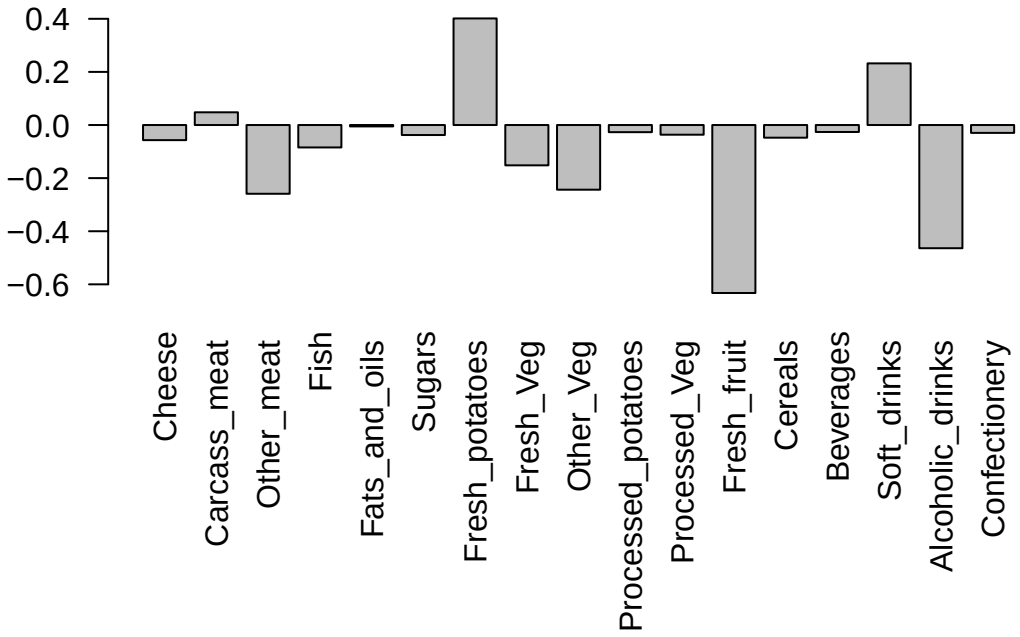
```
barplot(v, xlab="Principal Component", ylab="Percent Variation")
```




```
head(pca$rotation)
```

```
                    PC1         PC2         PC3         PC4
Cheese       -0.056955380  0.01601285  0.02394295 -0.40938259
Carcass_meat  0.047927628  0.01391582  0.06367111  0.72948192
Other_meat   -0.258916658 -0.01533114 -0.55384854  0.33100113
Fish         -0.084414983 -0.05075495  0.03906481  0.02237588
Fats_and_oils -0.005193623 -0.09538866 -0.12522257  0.03451216
Sugars       -0.037620983 -0.04302170 -0.03605745  0.02494334
```

```
## 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 )
```

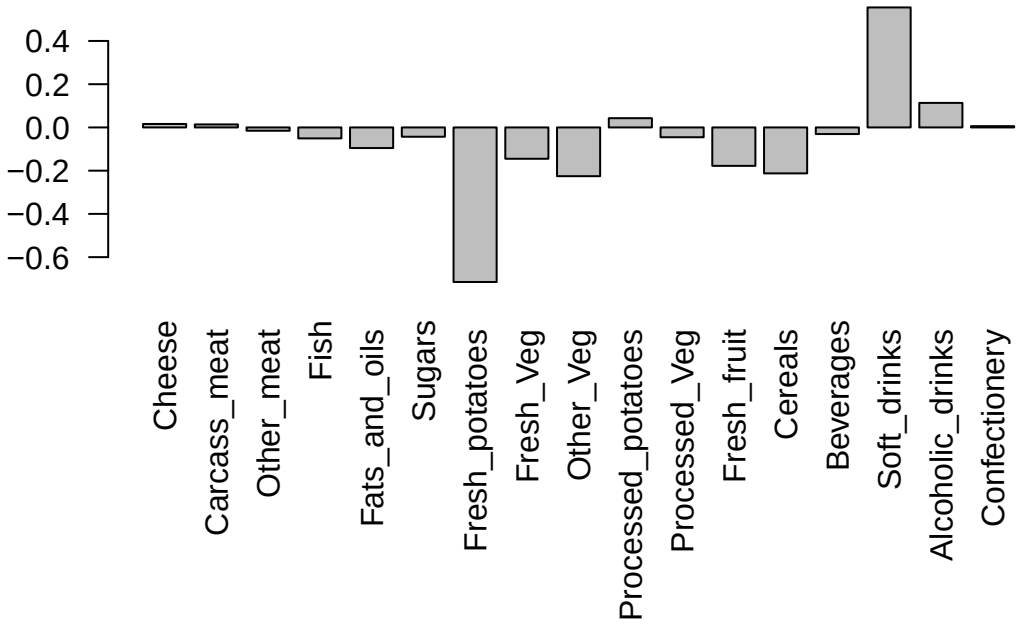


Q9.

```
head(pca$rotation)
```

```
                    PC1         PC2         PC3         PC4
Cheese       -0.056955380  0.01601285  0.02394295 -0.40938259
Carcass_meat  0.047927628  0.01391582  0.06367111  0.72948192
Other_meat   -0.258916658 -0.01533114 -0.55384854  0.33100113
Fish         -0.084414983 -0.05075495  0.03906481  0.02237588
Fats_and_oils -0.005193623 -0.09538866 -0.12522257  0.03451216
Sugars       -0.037620983 -0.04302170 -0.03605745  0.02494334
```

```
## Lets focus on PC1 as it accounts for > 90% of variance
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,2], las=2 )
```


0.4
0.2
0.0
−0.2
−0.4
−0.6
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