# ML_class8

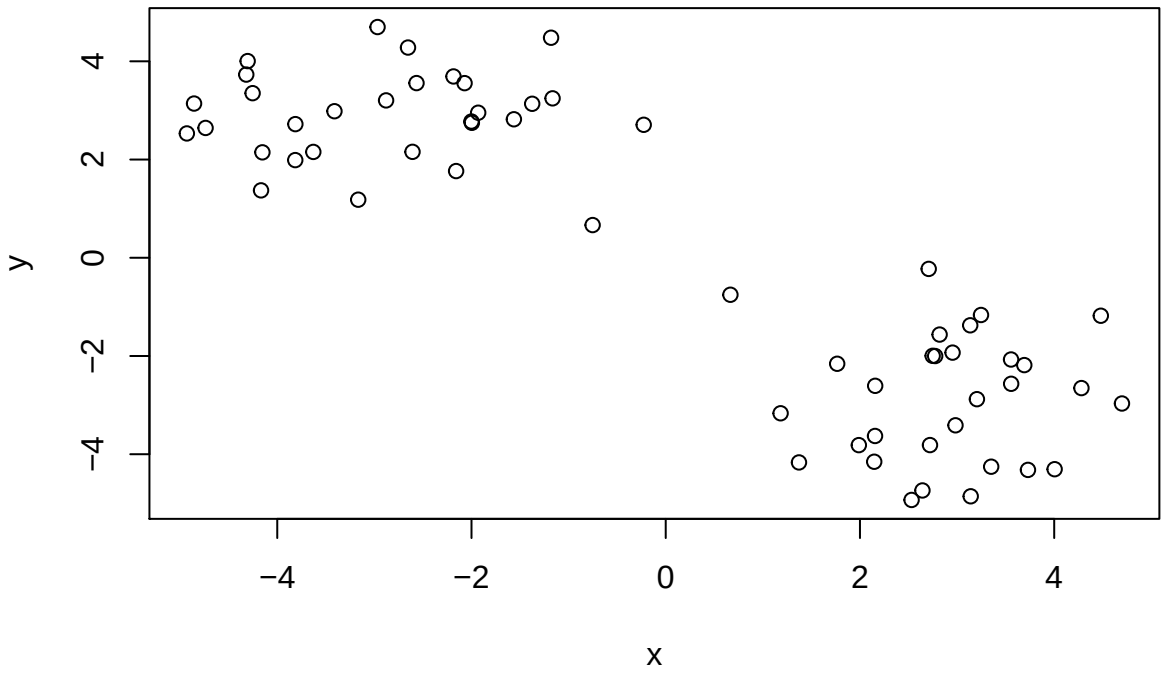
*Phoebe He*

*2/6/2019*

** Clustering**

*K-means clustering*

```
# Generate some example data for clustering
tmp <- c(rnorm(30,-3), rnorm(30,3))
x <- cbind(x=tmp, y=rev(tmp))
plot(x)
```




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

```
## K-means clustering with 2 clusters of sizes 30, 30
##
## Cluster means:
##          x        y
## 1 -2.86123  2.87966
## 2  2.87966 -2.86123
##
## 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 1 1 2 2 2 2 2
## [36] 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
##
## Within cluster sum of squares by cluster:
## [1] 75.16586 75.16586
##  (between_SS / total_SS =  86.8 %)
##
```

```
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Size of Clusters

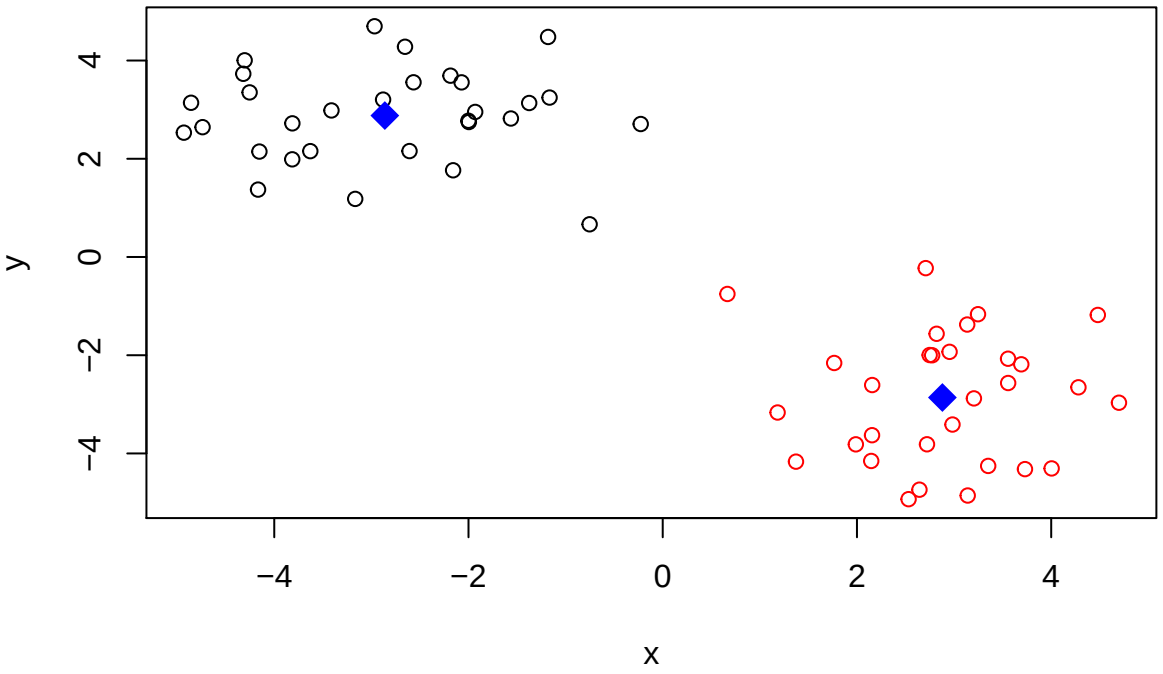
```
km$size
```

```
## [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
## [36] 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 2 2
```

```
plot(x,col=km$cluster)
points(km$centers,col='blue',pch=18,cex=2)
```



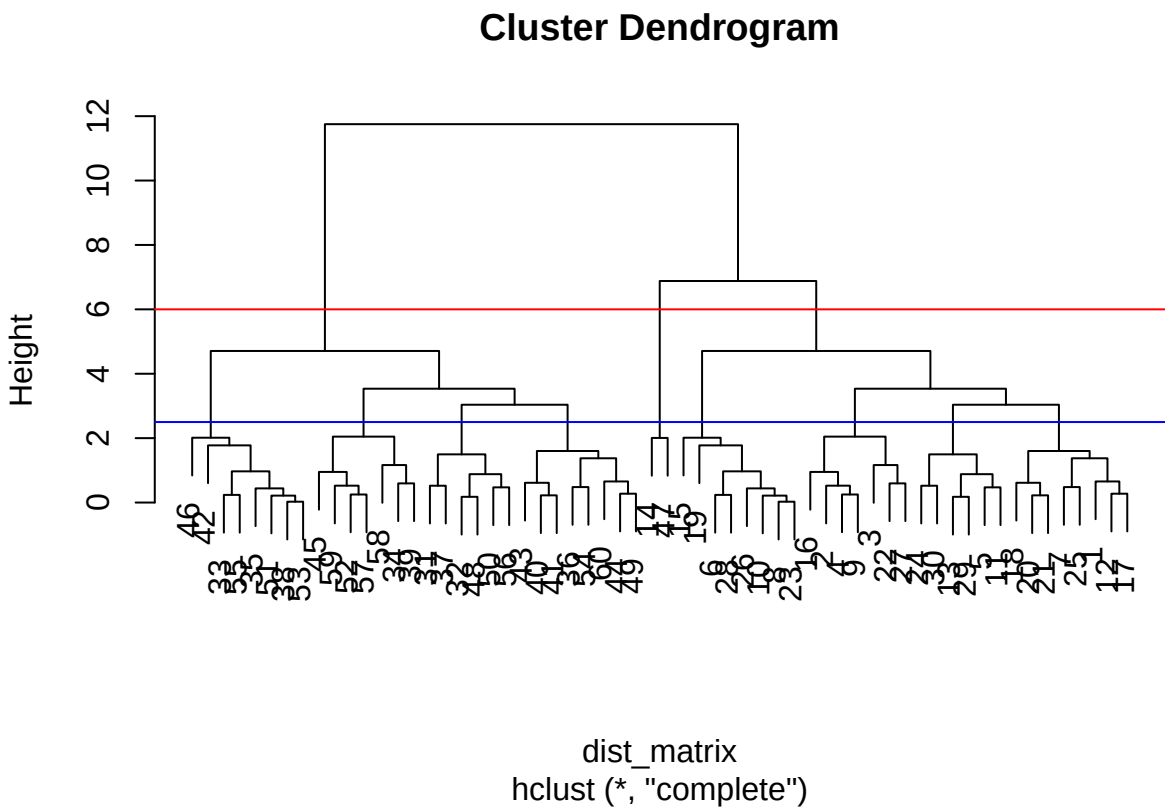

*Hierarchical clustering*

```
# First we need to calculate point (dis)similarity
#   as the Euclidean distance between observations
dist_matrix <- dist(x)
# The hclust() function returns a hierarchical
#  clustering model
hc <- hclust(d = dist_matrix)
# the print method is not so useful here
hc
```

```
##
## Call:
## hclust(d = dist_matrix)
##
## Cluster method   : complete
```

```
## Distance         : euclidean
## Number of objects: 60
```

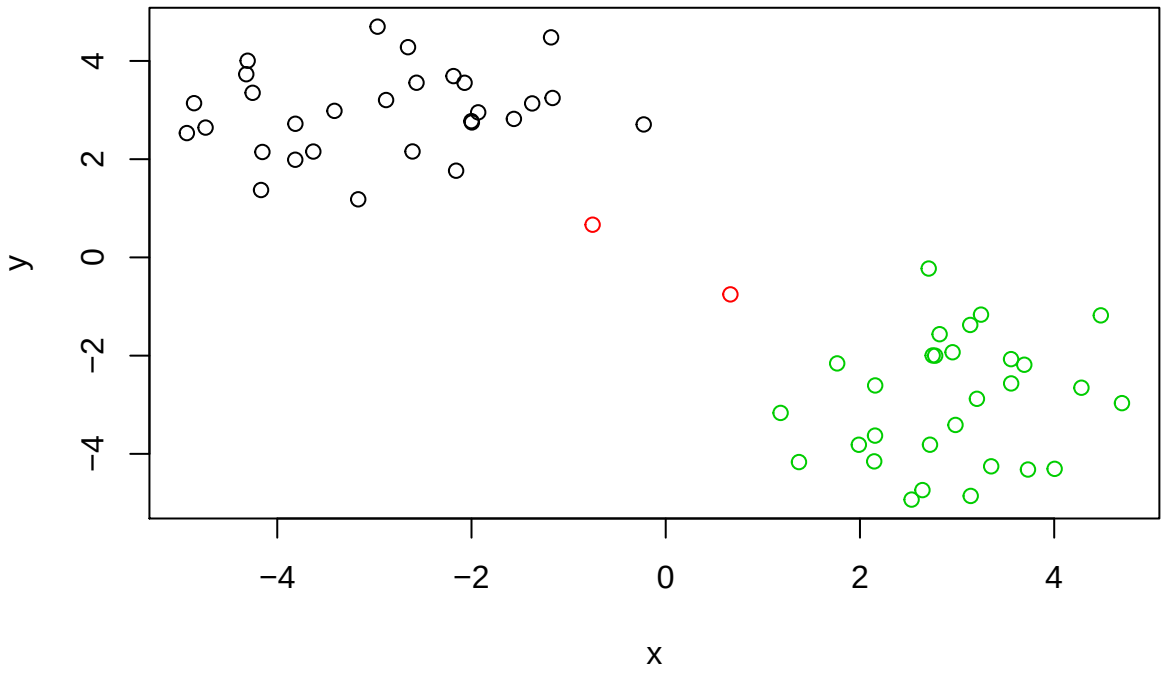
```
plot(hc)
abline(h=6,col='red')
abline(h=2.5,col='blue')
```




```
grp2 <- cutree(hc,h=6)
grp6 <- cutree(hc,h=2.5)
table(grp6)
```

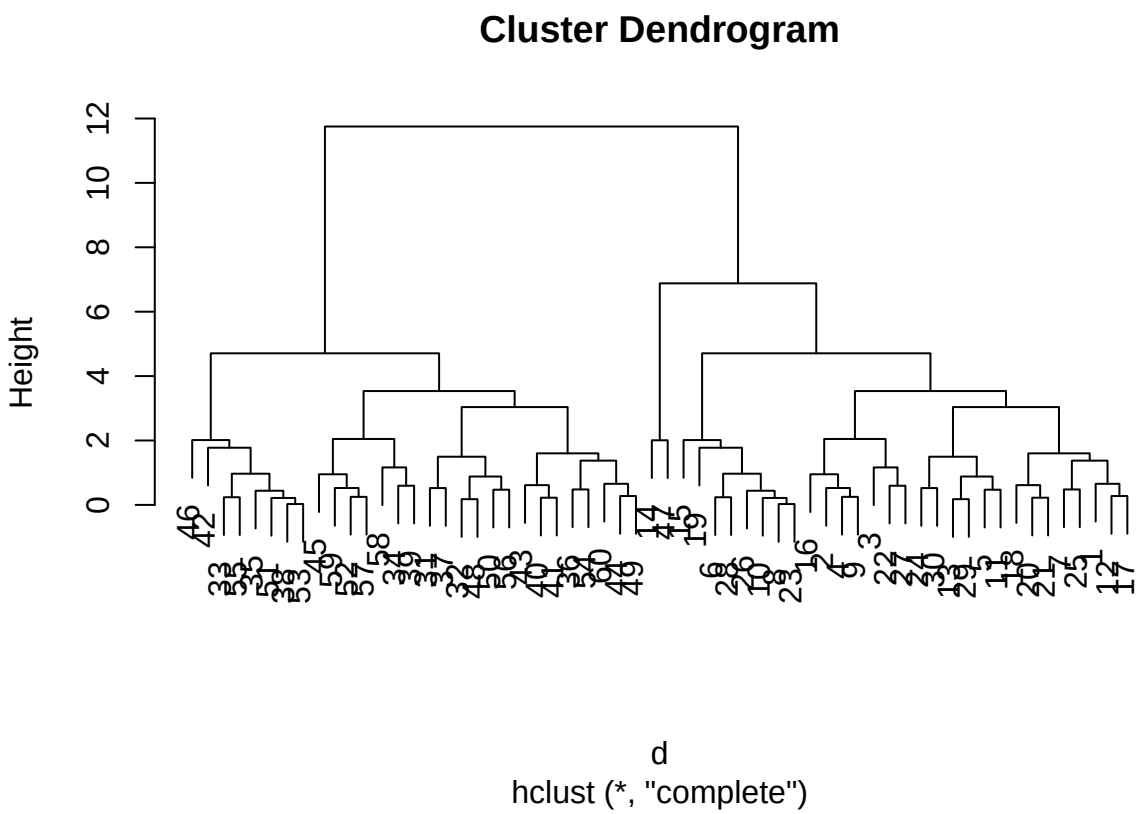
```
## grp6
## 1 2 3 4 5 6 7 8 9
## 8 7 6 8 2 6 8 7 8
```

```
plot(x,col=grp2)
```

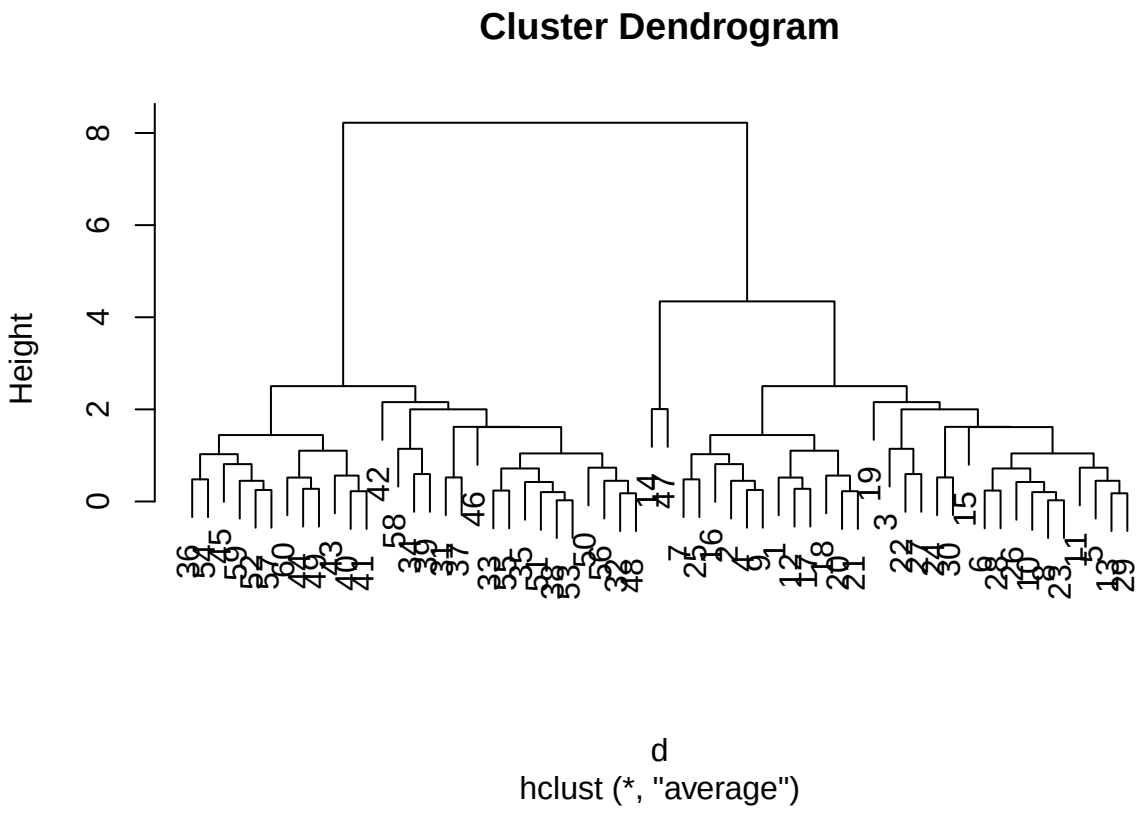



#using different linkage methods in R

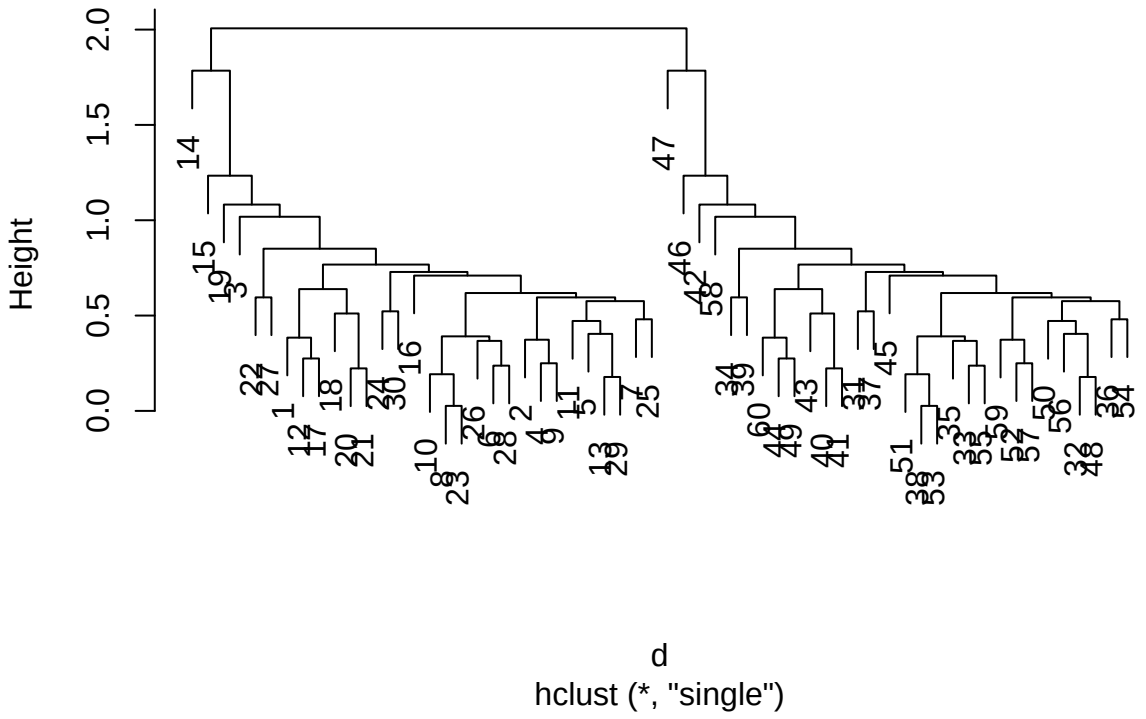
```
d <- dist_matrix

hc.complete <- hclust(d, method="complete")
plot(hc.complete)
```



```r
hc.average  <- hclust(d, method="average")
plot(hc.average)
```

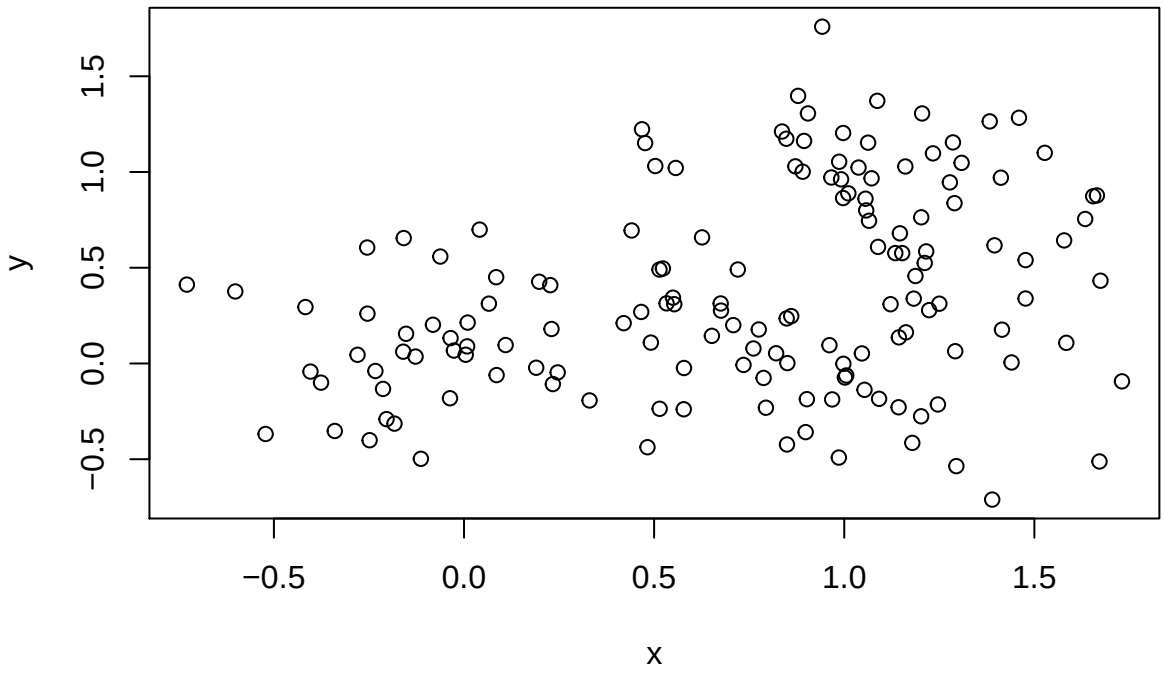



```r
hc.single   <- hclust(d, method="single")
plot(hc.single)
```

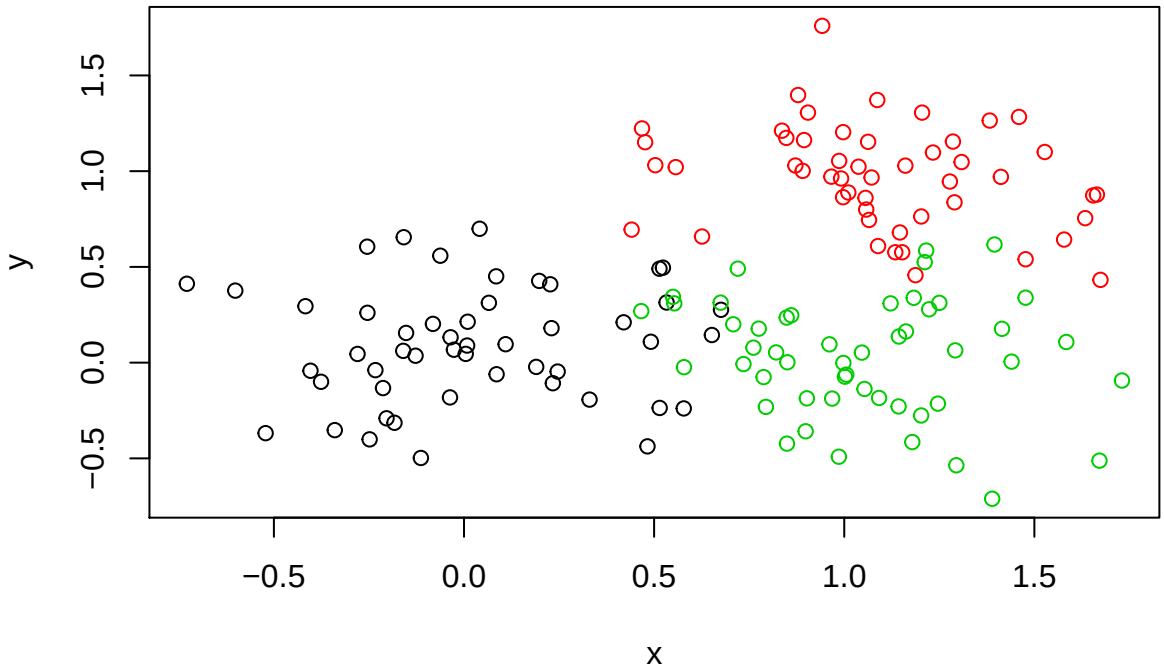



made up overlapping data a bit more rell

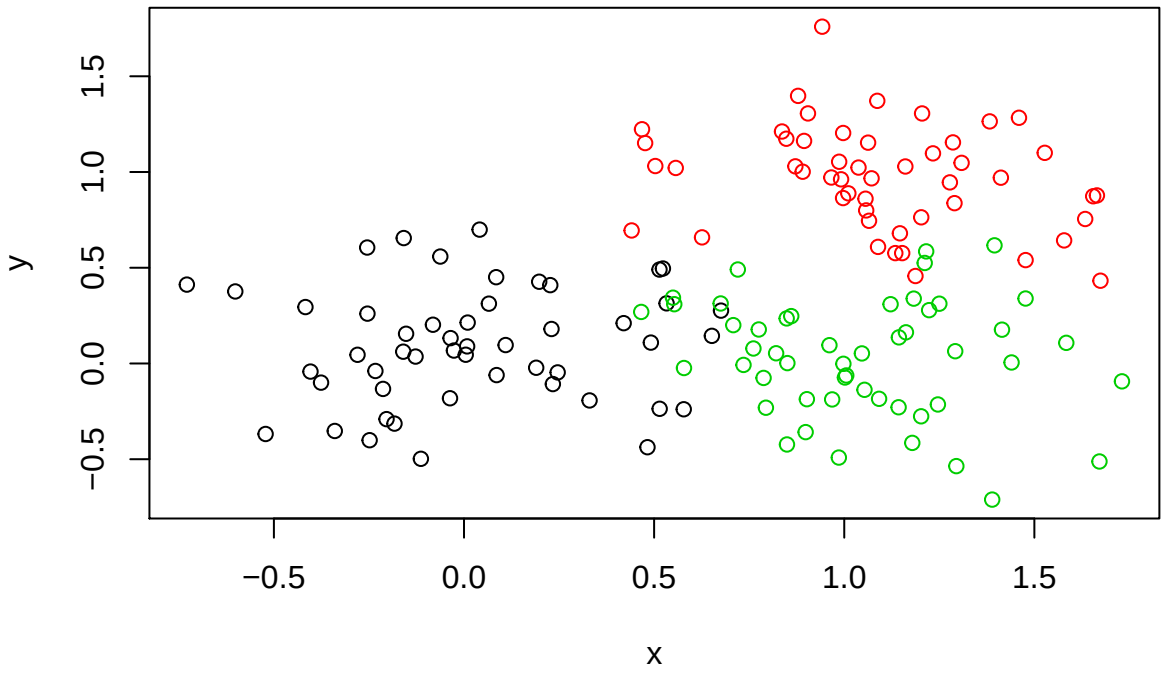
```
# Step 1. Generate some example data for clustering
a <- rbind(
  matrix(rnorm(100, mean=0, sd = 0.3), ncol = 2),   # c1
  matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2), # c2
  matrix(c(rnorm(50, mean = 1, sd = 0.3),           # c3
           rnorm(50, mean = 0, sd = 0.3)), ncol = 2))
colnames(a) <- c("x", "y")
# Step 2. Plot the data without clustering
plot(a)
```



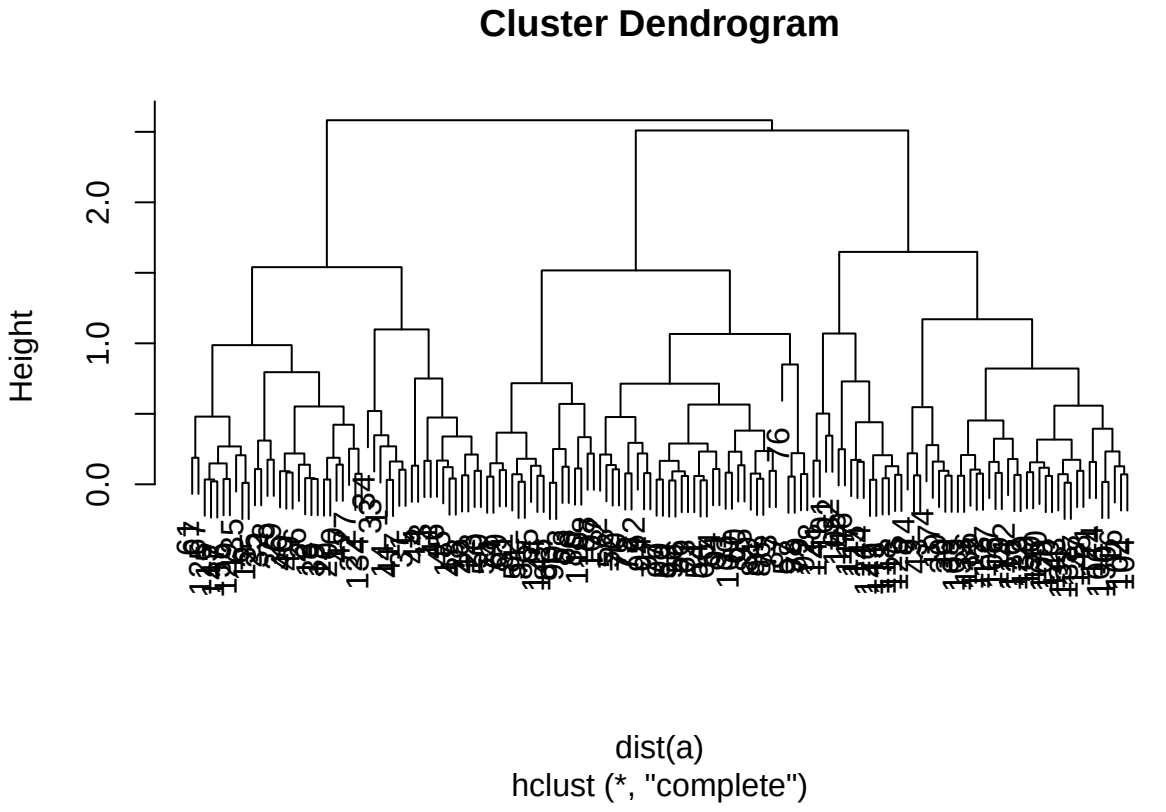

```
# Step 3. Generate colors for known clusters
#         (just so we can compare to hclust results)
col <- as.factor( rep(c("c1","c2","c3"), each=50) )
plot(a, col=col)
points(km$centers,col='blue',pch=18,cex=2)
```



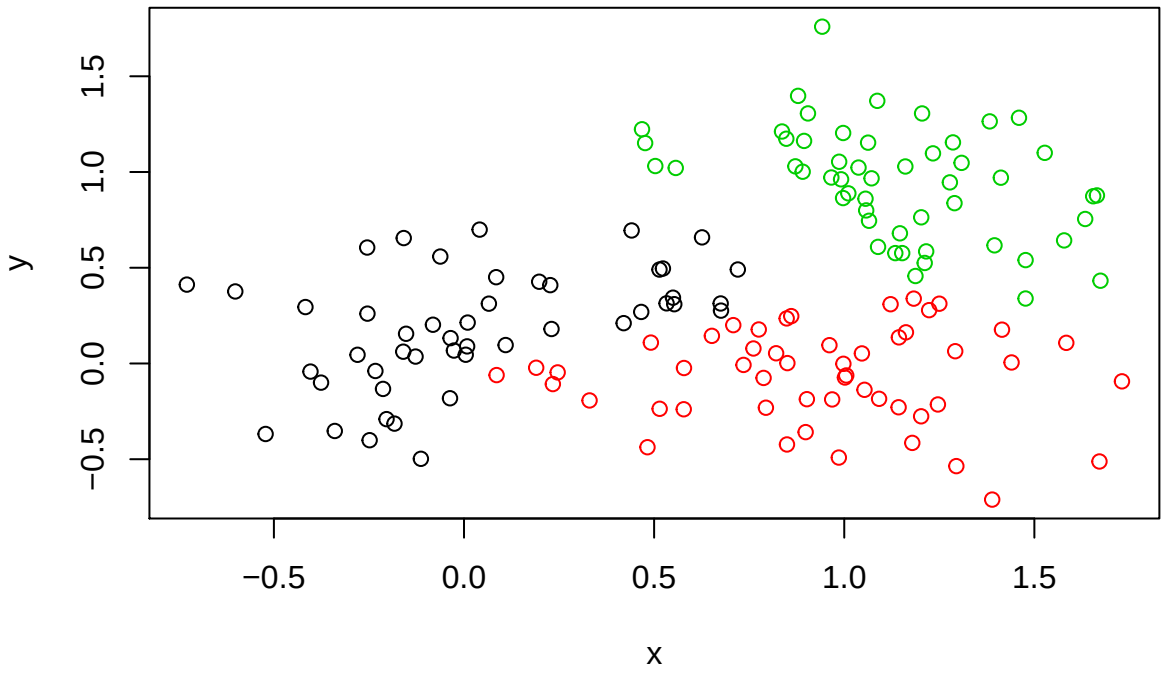

```
plot(a, col=col)
```




```
a.hc<- hclust(dist(a))
plot(a.hc)
```




```
grps3 <- cutree(a.hc,3)
plot(a,col=grps3)
```




*Principal Component Analysis (PCA)*

```
mydata <- read.csv("https://tinyurl.com/expression-CSV",
row.names=1)
head(mydata)
```

```
##        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
```

*NOTE* prcomp() expects the samples to be rows and genes to be columns so we need to first transpose the matrix with the t() function!

```
pca <- prcomp(t(mydata),scale=TRUE)
summary(pca)
```

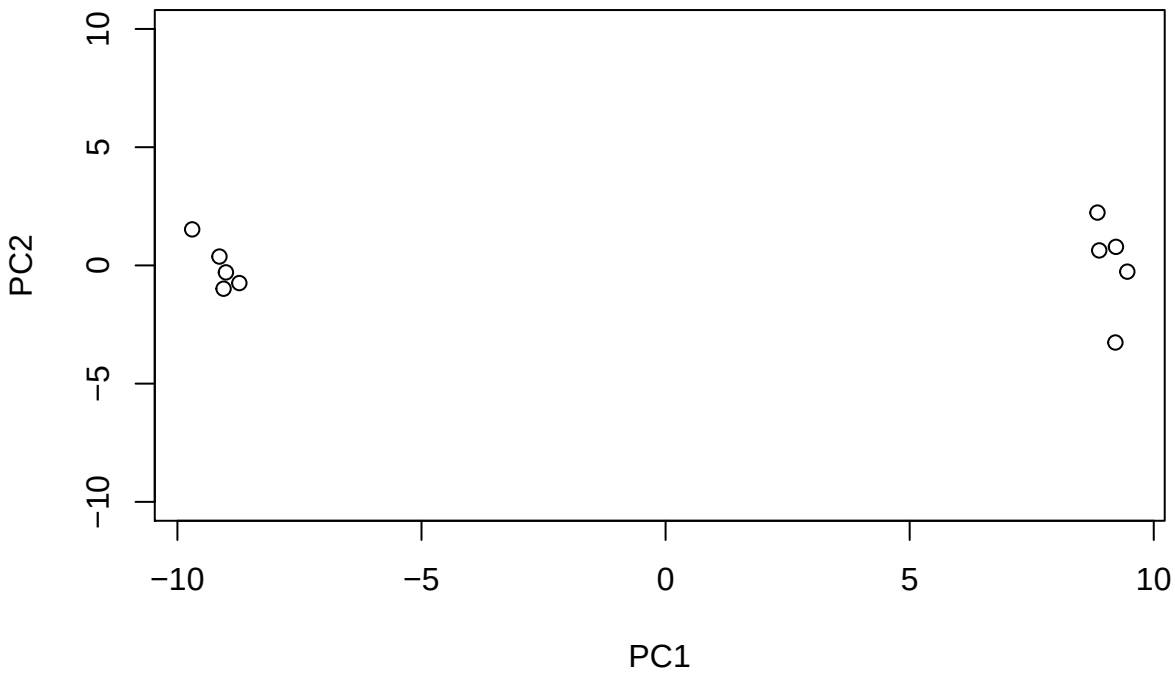
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     9.6237 1.5198 1.05787 1.05203 0.88062 0.82545
## Proportion of Variance 0.9262 0.0231 0.01119 0.01107 0.00775 0.00681
## Cumulative Proportion  0.9262 0.9493 0.96045 0.97152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

square of the standard deviation is the variation

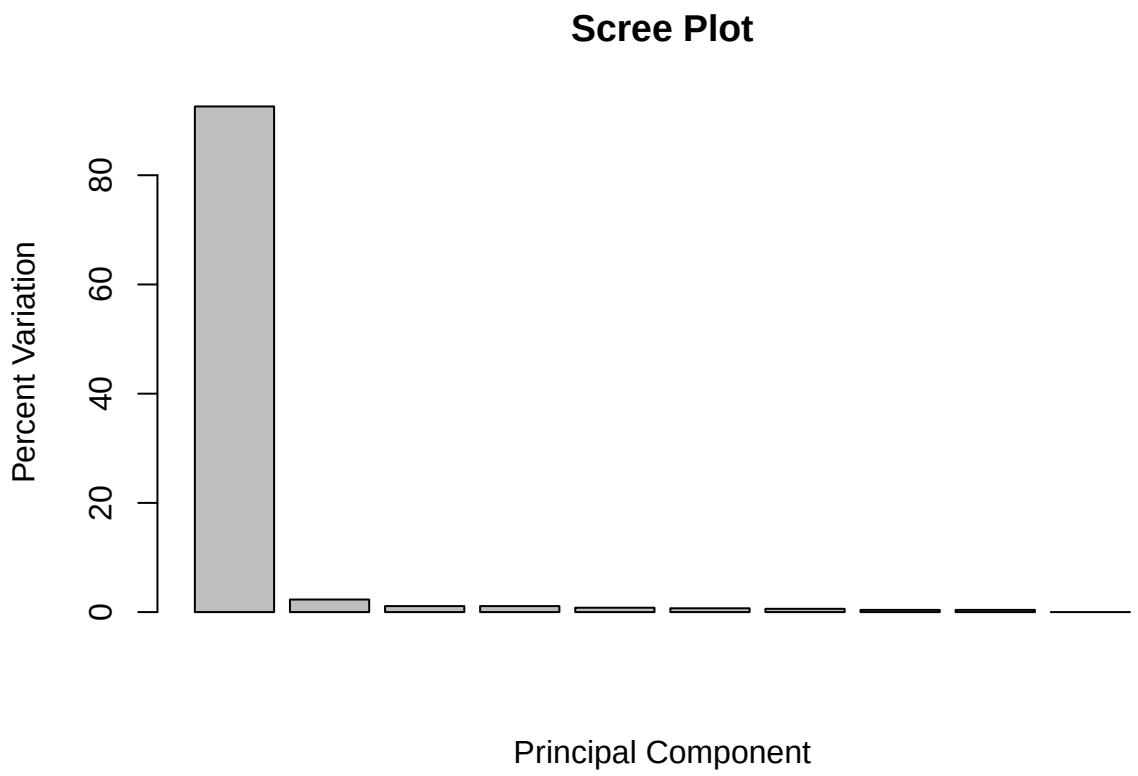
```
dim(pca$x)
```

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

```r
plot(pca$x[,1],pca$x[,2],
     xlab="PC1",ylab="PC2",
     ylim=c(-10,10))
```

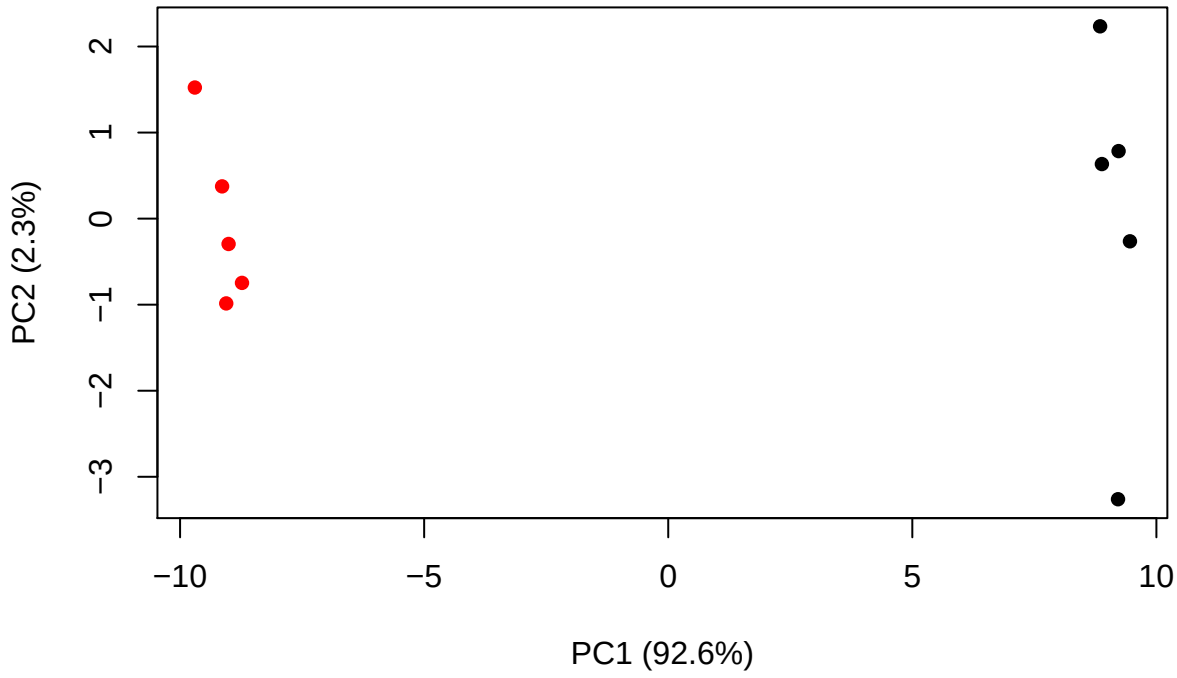



```r
#percent variance is often more informative to look at
pca.var <- pca$sdev^2
pca.var.per <- round(pca.var/sum(pca.var)*100, 1)
barplot(pca.var.per, main="Scree Plot",
        xlab="Principal Component", ylab="Percent Variation")
```




Make our PCA plot nice

```
## A vector of colors for wt and ko samples
colvec <- as.factor( substr( colnames(mydata), 1, 2) )
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], "%)"))
```




**PCA exercise**

```
ukf<- read.csv('/Users/phoebehe/Desktop/BGGN213/class08 ML/UK_foods.csv')
dim(ukf)
```

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

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

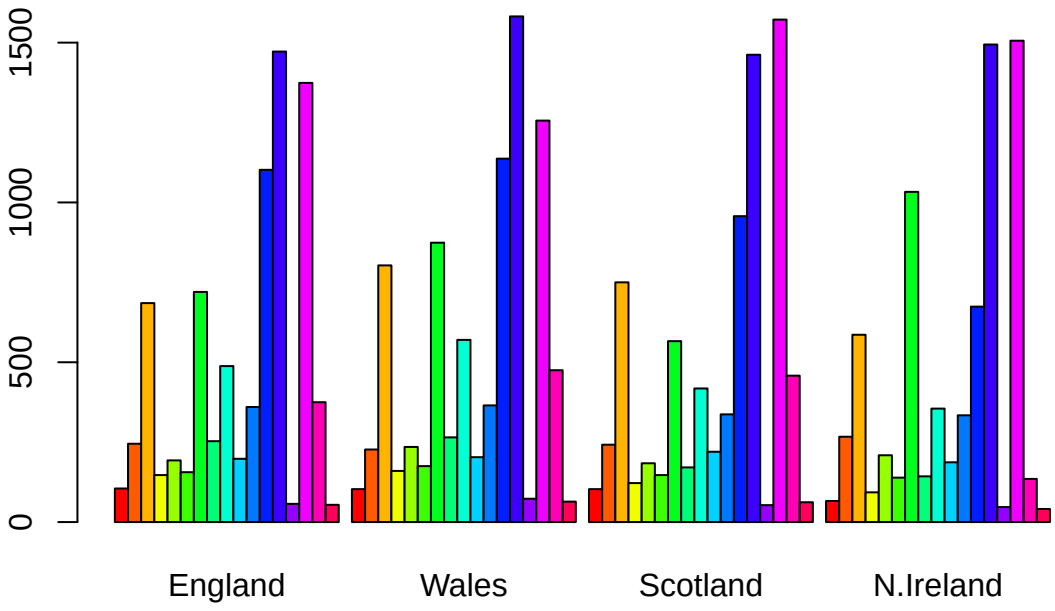
```
##                England Wales Scotland N.Ireland
## Cheese             105   103      103        66
## Carcass_meat       245   227      242       267
## Other_meat         685   803      750       586
## Fish               147   160      122        93
## Fats_and_oils      193   235      184       209
## Sugars             156   175      147       139
```

```
dim(ukf)
```

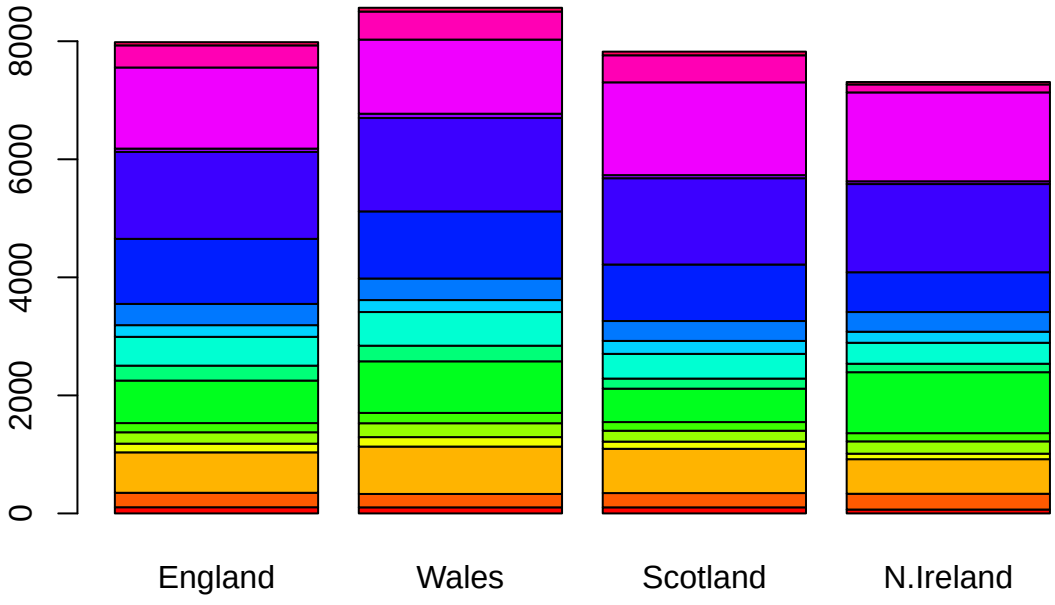
```
## [1] 17  4
```

```
#assign row names from the first col of the data upon reading.
ukf <- read.csv('/Users/phoebehe/Desktop/BGGN213/class08 ML/UK_foods.csv',row.names=1)
```

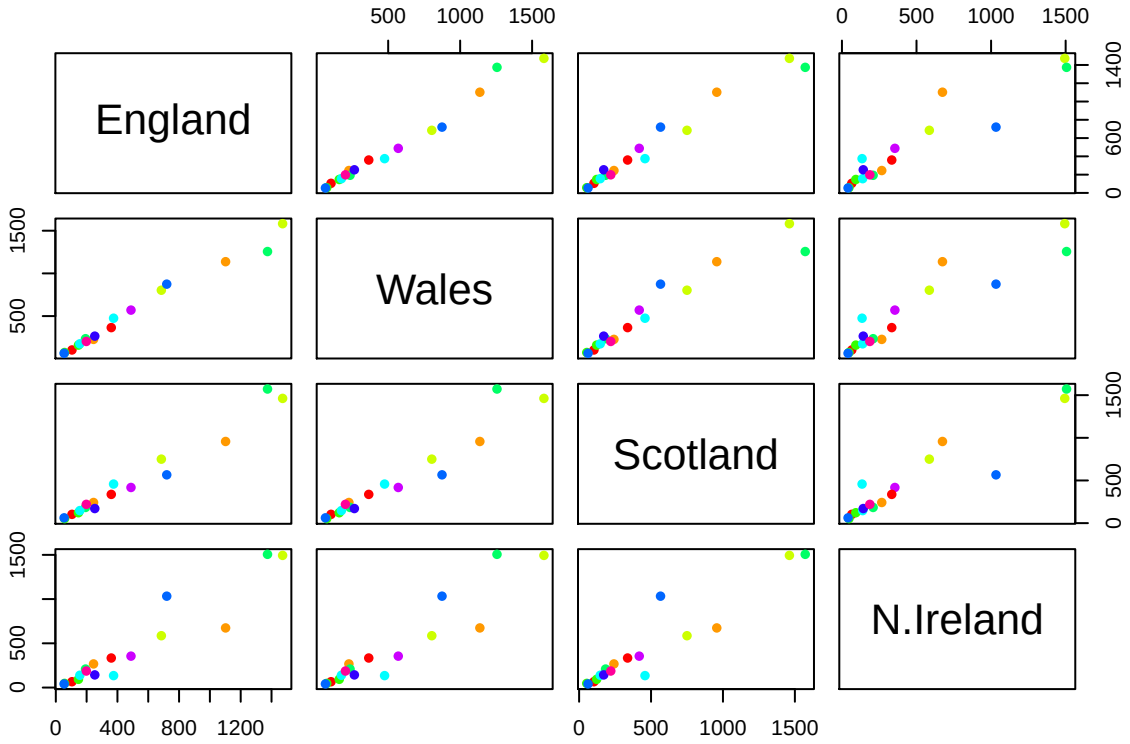
```
barplot(as.matrix(ukf), beside=T, col=rainbow(nrow(ukf)))
```




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



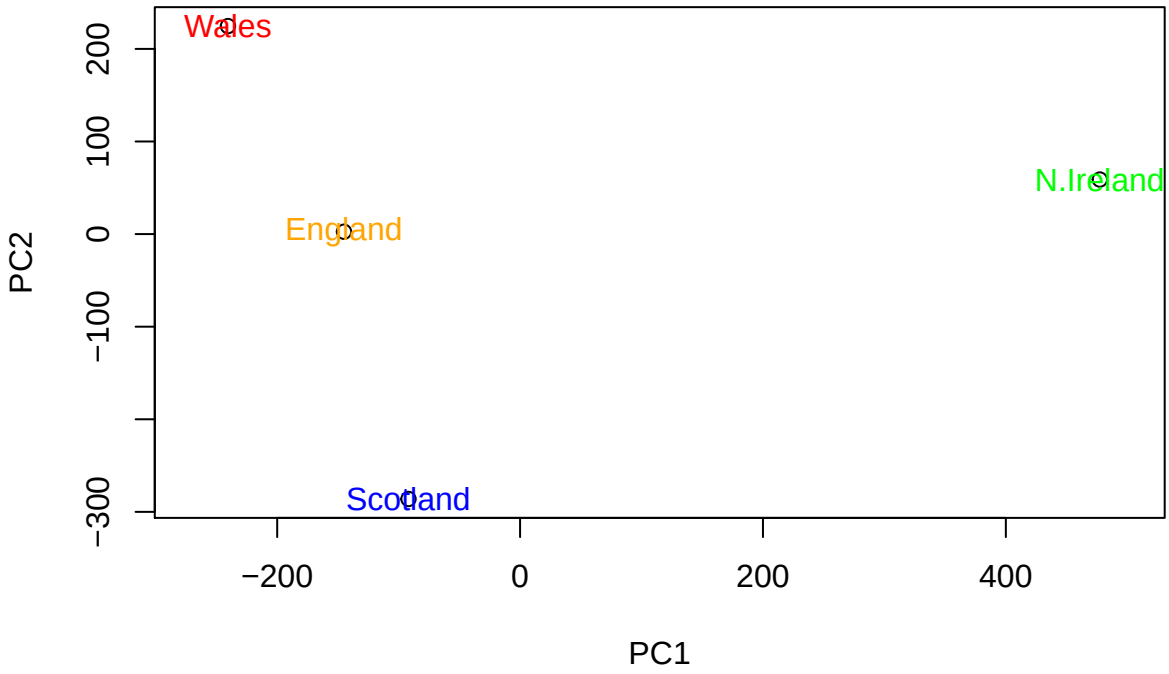

```
pairs(ukf, col=rainbow(10), pch=16)
```




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

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



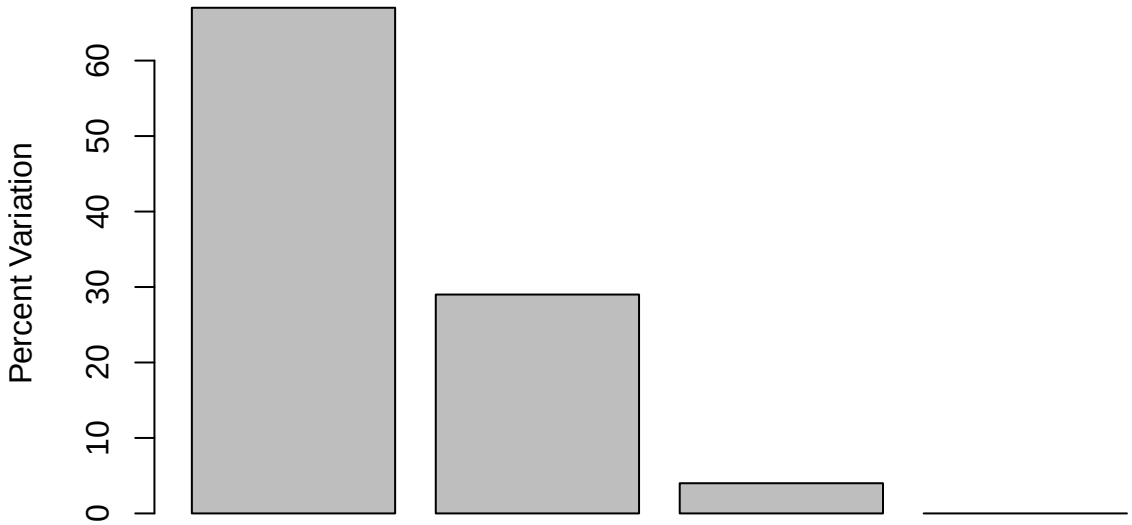

```
v <- round( pca$sdev^2/sum(pca$sdev^2) * 100 )
v
```

```
## [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")
```




Principal Component

```
## 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 )
```