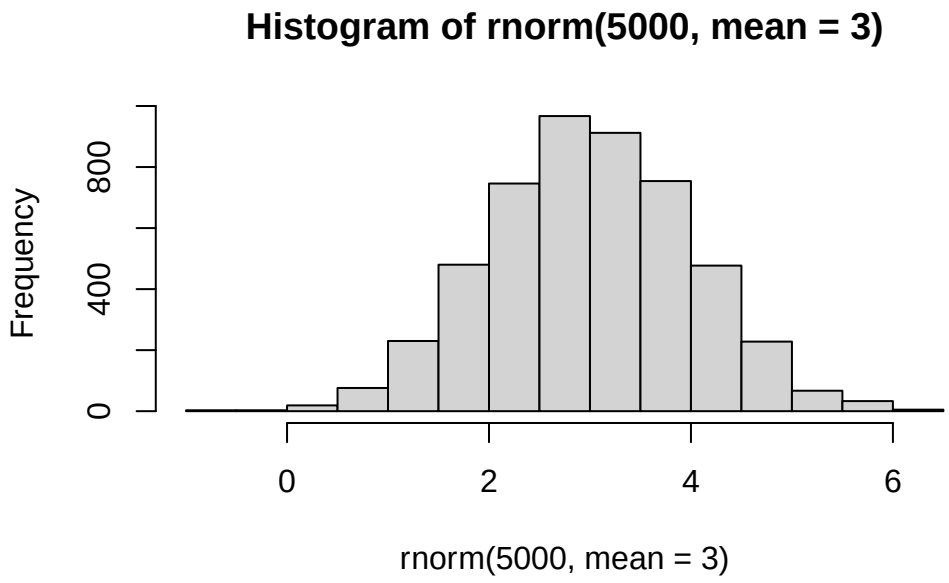
Lab7_inclass

Qianqian Tao

#Clustering First let's make up some data to cluster so we can get a feel for these methods and how to work with them.

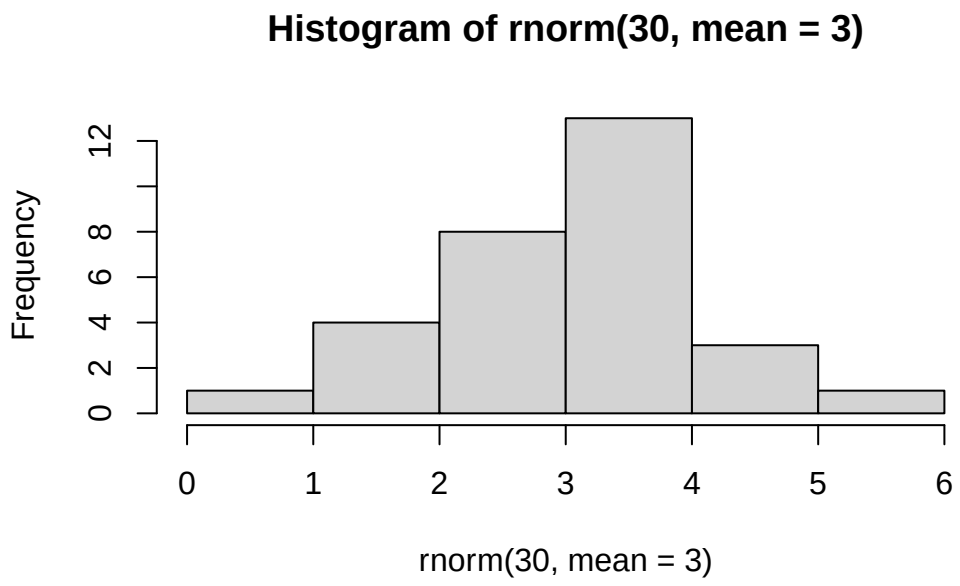
We can use the `rnorm()` function to get random numbers from a normal distribution around a given `mean`.

```
#random number
hist(rnorm(5000, mean=3))
```

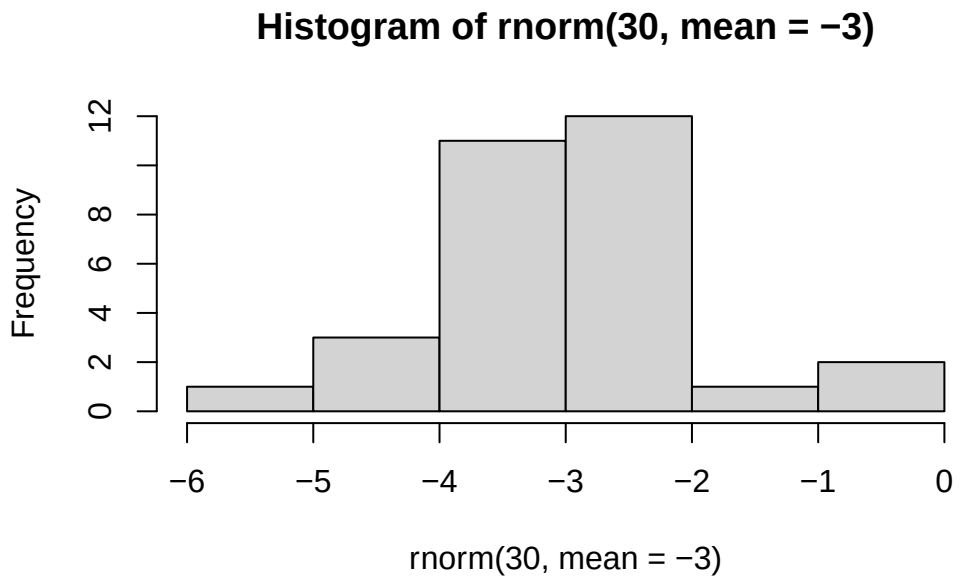


Let's get 30 points with a mean of 3.

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




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




```
a <- c(rnorm(30, mean=3),rnorm(30, mean=-3))
x <- cbind(x=a,y=rev(a))
x
```

```
           x          y
```

```
 [1,]  3.7785110 -2.9280959
 [2,]  3.1090284 -2.6149411
 [3,]  4.3727669 -4.1466839
 [4,]  2.8447949 -4.7616025
 [5,]  2.5888402 -2.5373107
 [6,]  0.6388272 -3.4475599
 [7,]  5.0597991 -4.1419792
 [8,]  3.5449317 -2.5140144
 [9,]  3.6612587 -3.0048985
[10,]  2.0905794 -1.8962746
[11,]  2.3314941 -3.5821153
[12,]  4.6061880 -2.6138389
[13,]  1.5504612 -0.6559492
[14,]  3.8045313 -3.5837767
[15,]  3.2661249 -2.5353615
[16,]  1.5606090 -2.0182963
[17,]  3.3244769 -4.1763087
[18,]  1.6704531 -2.3798862
[19,]  2.2029439 -3.0226257
[20,]  3.7552036 -1.0974689
[21,]  3.3248097 -3.0271464
[22,]  2.3944669 -2.9728121
[23,]  4.7123428 -2.3455399
[24,]  4.0537530 -3.7314000
[25,]  4.8084103 -4.0406517
[26,]  5.9994351 -1.4814832
[27,]  2.6551271 -3.9820101
[28,]  3.5925896 -4.2990141
[29,]  4.1947529 -4.0644288
[30,]  1.7252626 -2.8873737
[31,] -2.8873737  1.7252626
[32,] -4.0644288  4.1947529
[33,] -4.2990141  3.5925896
[34,] -3.9820101  2.6551271
[35,] -1.4814832  5.9994351
[36,] -4.0406517  4.8084103
[37,] -3.7314000  4.0537530
[38,] -2.3455399  4.7123428
[39,] -2.9728121  2.3944669
[40,] -3.0271464  3.3248097
[41,] -1.0974689  3.7552036
[42,] -3.0226257  2.2029439
[43,] -2.3798862  1.6704531
```

```
[44,] -4.1763087  3.3244769
[45,] -2.0182963  1.5606090
[46,] -2.5353615  3.2661249
[47,] -3.5837767  3.8045313
[48,] -0.6559492  1.5504612
[49,] -2.6138389  4.6061880
[50,] -3.5821153  2.3314941
[51,] -1.8962746  2.0905794
[52,] -3.0048985  3.6612587
[53,] -2.5140144  3.5449317
[54,] -4.1419792  5.0597991
[55,] -3.4475599  0.6388272
[56,] -2.5373107  2.5888402
[57,] -4.7616025  2.8447949
[58,] -4.1466839  4.3727669
[59,] -2.6149411  3.1090284
[60,] -2.9280959  3.7785110
```

##K-means clustering

Very popular clustering method

```
km <- kmeans(x, 2)
km
```

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

Cluster means:
          x         y
1  3.240759 -3.016362
2 -3.016362  3.240759

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 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] 72.41529 72.41529
 (between_SS / total_SS =  89.0 %)

Available components:
```

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

```
#What component is cluster size?
km$size
```

```
[1] 30 30
```

```
#What component is cluster membership/assignment
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 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
```

```
#what component details cluster centers?
km$centers
```

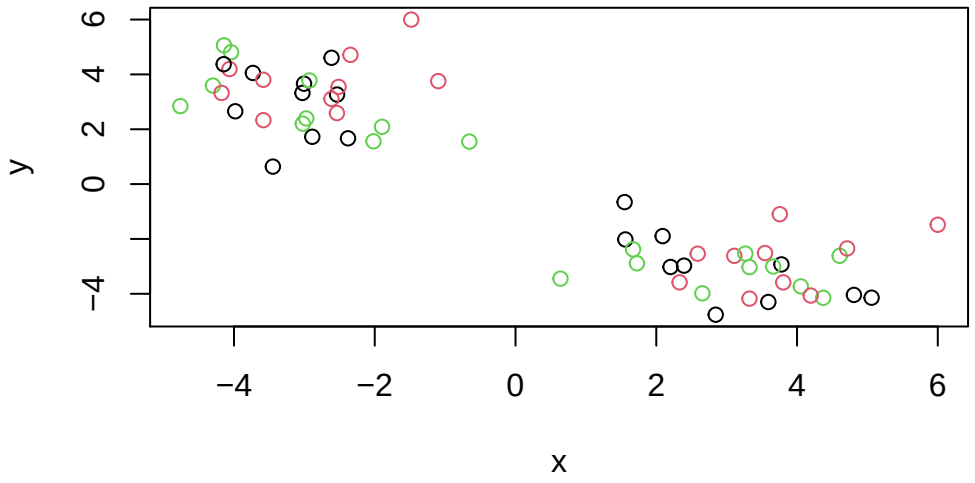
```
          x         y
1  3.240759 -3.016362
2 -3.016362  3.240759
```

```
#how spread the points are
km$tot.withinss
```

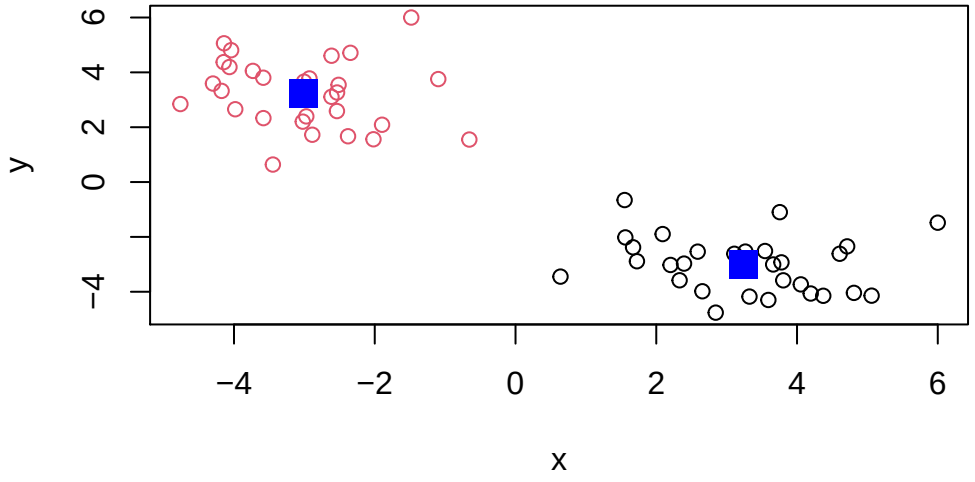
```
[1] 144.8306
```

Q. Plot x and add centers in different colors

```
mycols <- c(1,2,3)
plot(x, col=mycols)
```




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

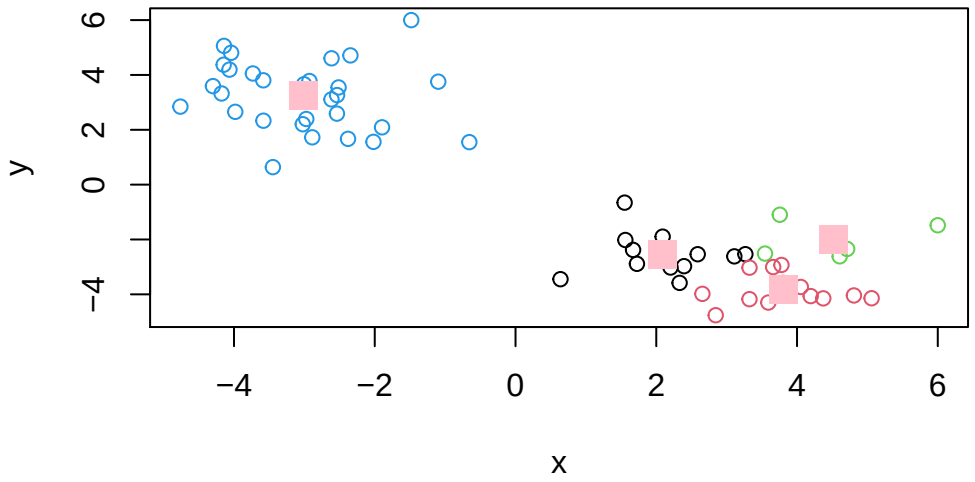


Q. Let's cluster into 3 groups on the same **x** data and make a plot.

```
km_2 <- kmeans(x,4)
plot(x,col=km_2$cluster)
km_2$tot.withinss
```

```
[1] 100.1913
```

```
points(km_2$centers, col="pink", pch=15, cex=2)
```



Hierachical Clustering

We can use the `hclust()` function for Hierachical Clustering. Unlike `kmeans()`, where we could just pass in our data as input, we need to give `hclust()` a "distance matrix".

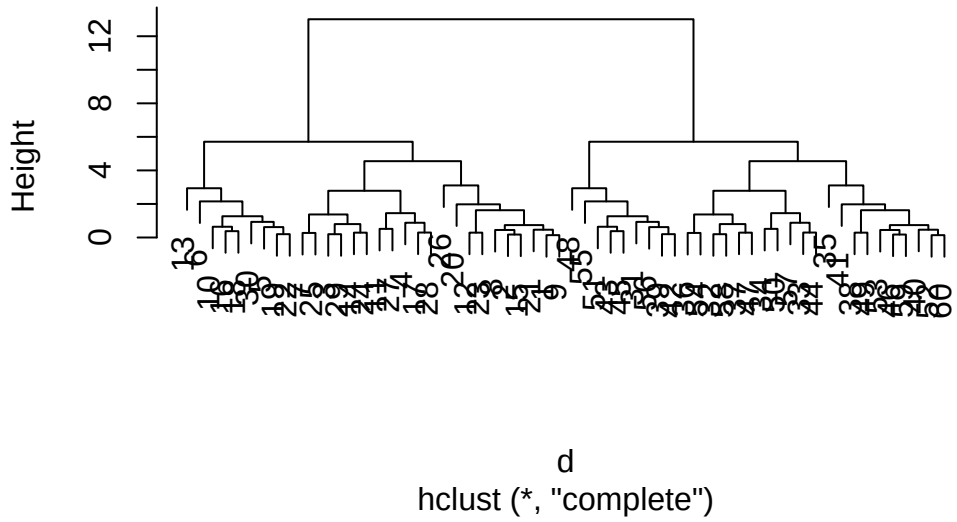
We will use the `dist()` function to start with.

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

```
Call:
hclust(d = d)

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

```
plot(hc)
```



I can now "cut" my tree with the `cutree()` to yield a cluster membership vector

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

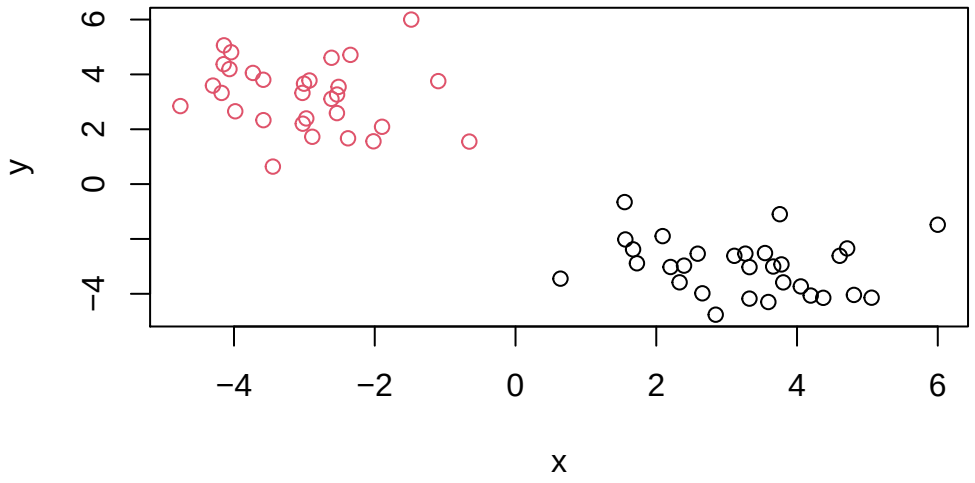
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

You can also tell `cutree()` to cut where it yield "k" groups.

```
grps2 <- cutree(hc, k=2)
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 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
```

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

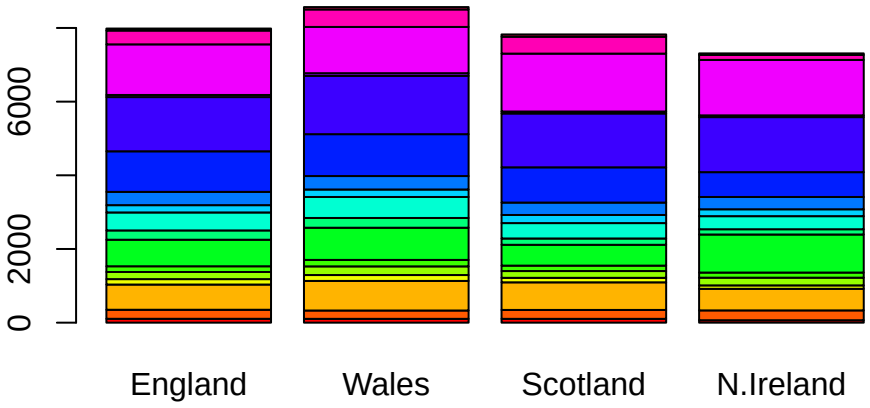


Principal Component Analysis

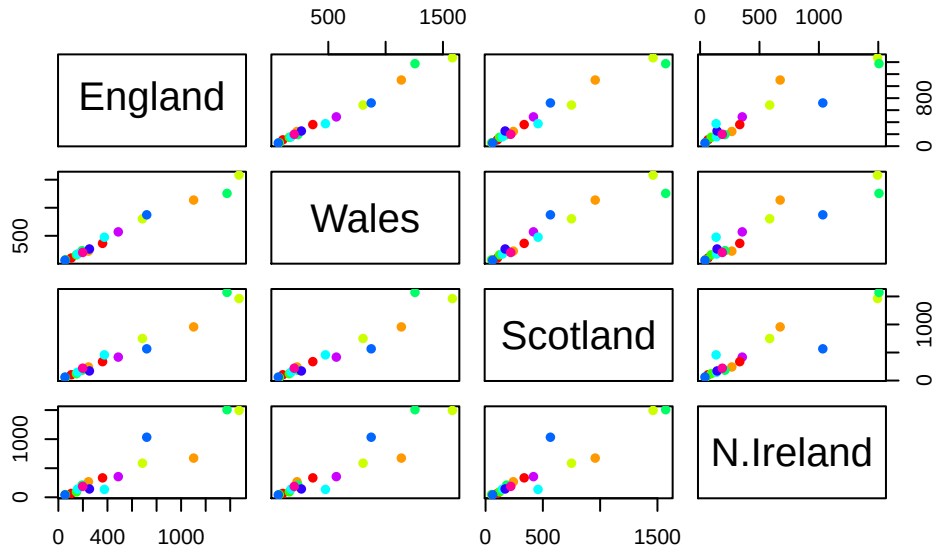
```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url,row.names = 1)
dim(x)
```

```
[1] 17  4
```

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




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




```
#if the value are exactly the same, the line will be straight in the pair-wise graph betwe
#if above the diagnol, more in the y axis; if below, more in x-axis
```

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

```
#proportion of variance: include how much portion of variance
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"
```

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