class07

Nikan Ostovan

intro to machine learning

in this class we will explore clutstring and dimensionality reduction methods.

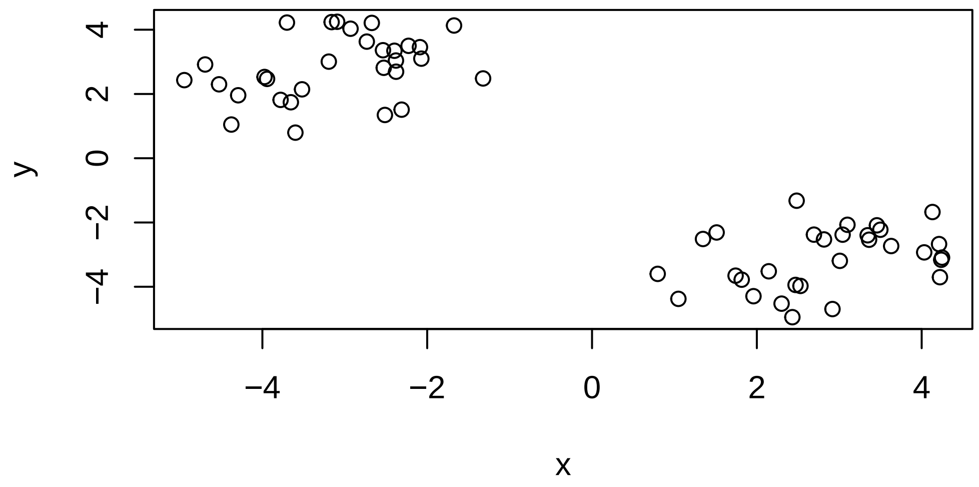
##k-means number of clusters we want.

make up some input data where we know the answer shoulbe be

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

```
             x        y
[1,] -2.527481 2.814751
[2,] -2.672223 4.211217
[3,] -2.398008 3.344538
[4,] -2.377739 2.692792
[5,] -3.194182 3.007006
[6,] -3.519000 2.144454
```

```
plot(x)
```



ucsd the kmeans function with k =2 and nstart = 20

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

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

Cluster means:
          x         y
1 -3.107253  2.817367
2  2.817367 -3.107253

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 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

Within cluster sum of squares by cluster:
[1] 54.25498 54.25498
 (between_SS / total_SS =  90.7 %)

Available components:
```

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

how many points in each cluster

```r
km$size
```

```
[1] 30 30
```

what omponent of your result objects details cluseter assignment/membership. -and cluster center

```r
km$cluster
```

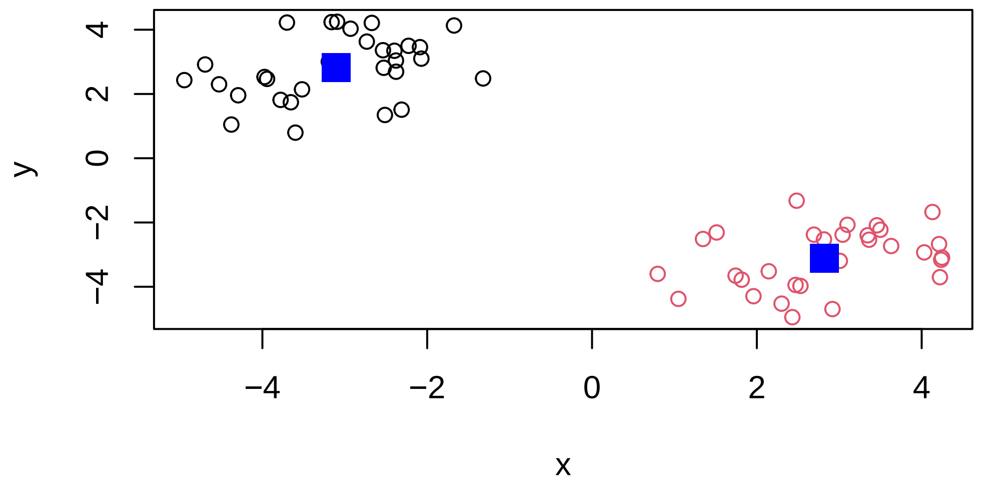
```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
[39] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```r
km$centers
```

```
          x         y
1 -3.107253  2.817367
2  2.817367 -3.107253
```

plot x colored b the kmeans cluster assinemnt and add centers as blue points

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



this is anoter very useful and widely employed clustring method whih has the advantage over kmeans in that it can help reveal the something of the true grouping in your data.

hclusts function wants a distance matrix as input. and we get this from the dist function

```
d = dist(x)
hc = hclust(d)
hc
```

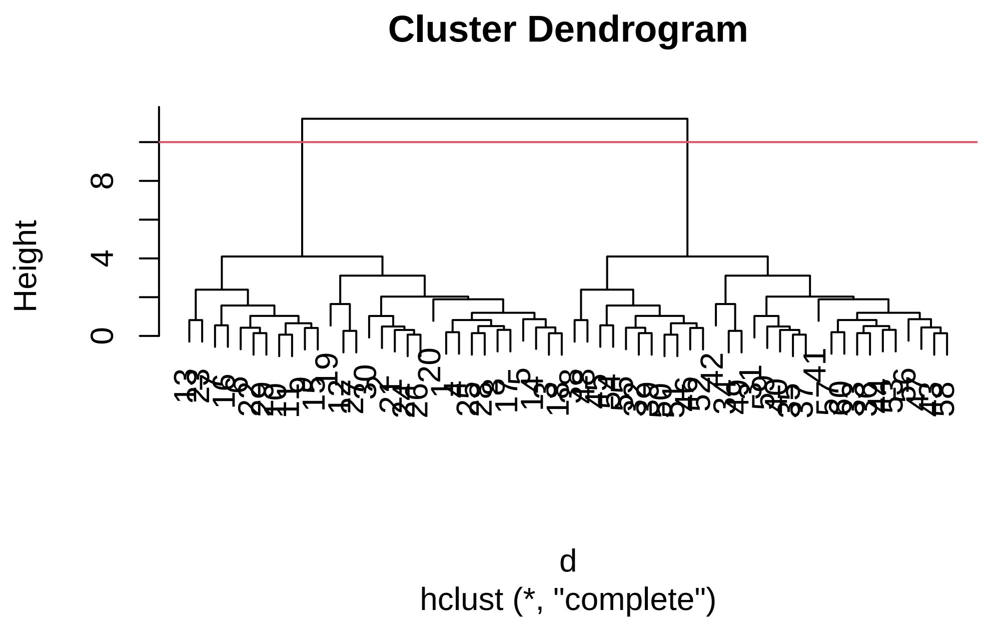
```

Call:
hclust(d = d)

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

there is a plot moethod for hclust results

```
plot(hc)
abline(h=10, col = 2)
```

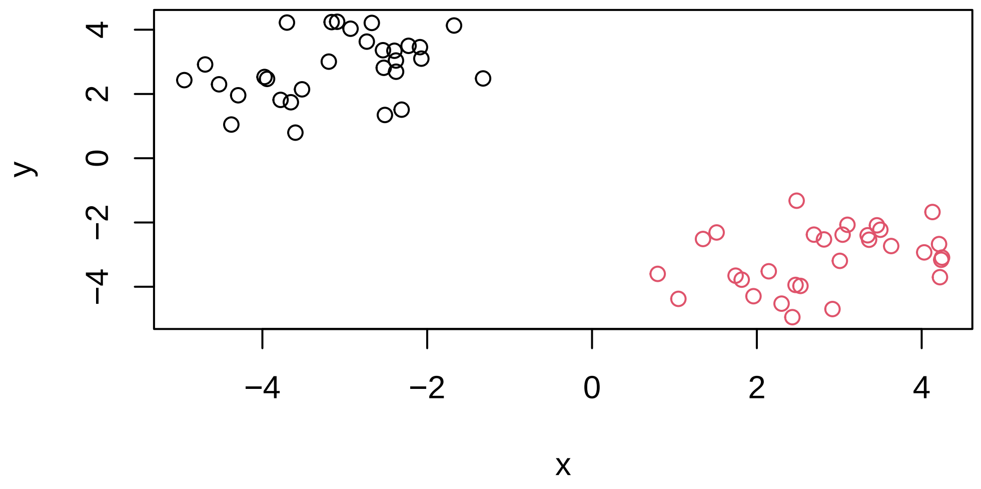


to get my cluster membership beoter i need to cut my tree to yield subtrees or branches with all the members of a given cluster resising on the same cut branch.using the cutree function

```
grps = cutree(hc, h = 10)
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 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
```

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



```r
cutree(hc, k = 4)
```

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

it is often helpful to use the k= argument to cutree rather than the h= height of cutting. this will cue the tree to yield the number of clusters you want.

#principal component analysis (PCA)

the R function for PCA is callded prcomp()

PCA of UK food data, the 17d data set import data

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

```r
dim(x)
```

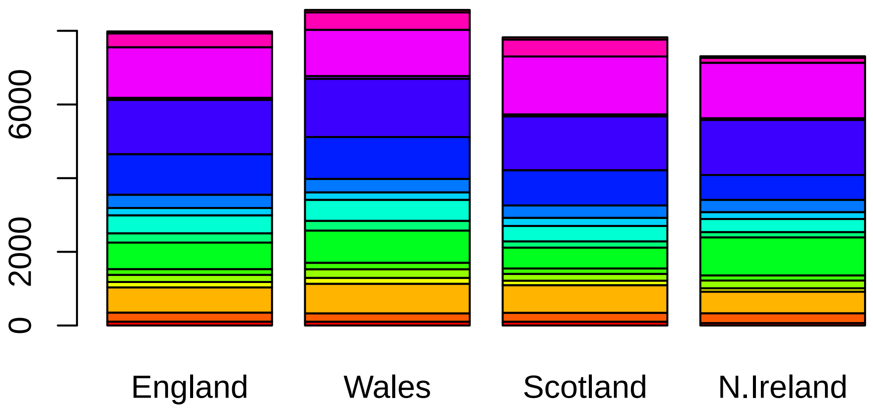
```
[1] 17  5
```

```r
rownames(x) <- x[,1]
x <- x[,-1]
head(x)
```

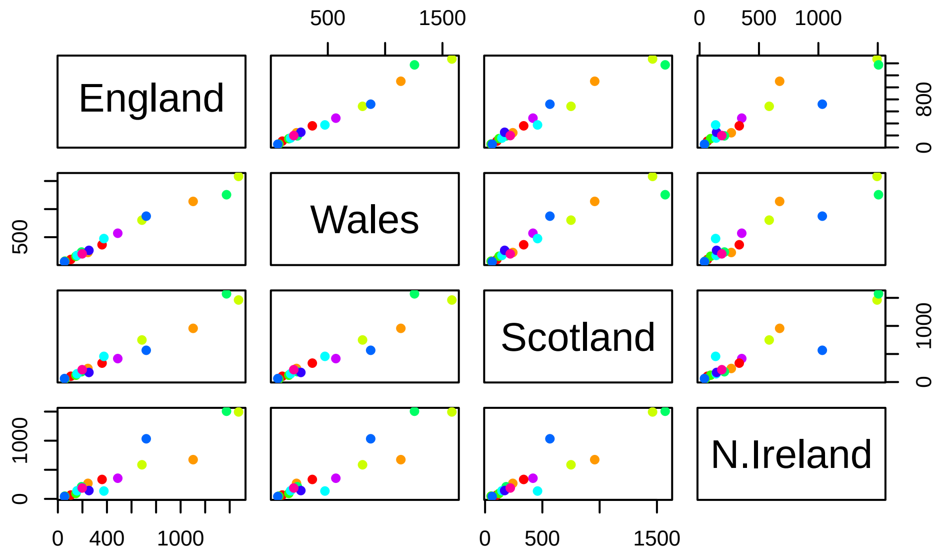
```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars  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

Q2: the one with the optional argument seems more robust and prone to mistakes Q3: beside argument will change that

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



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



Q5: if are on the diagonal they are correlated but if they are not they are skewed between the different countries.

Q6: plots with irelend vs all other countries are not as diagonal, as a result N. Ireland is different thant the other countries

```
pca = prcomp(t(x))
summary(pca)
```

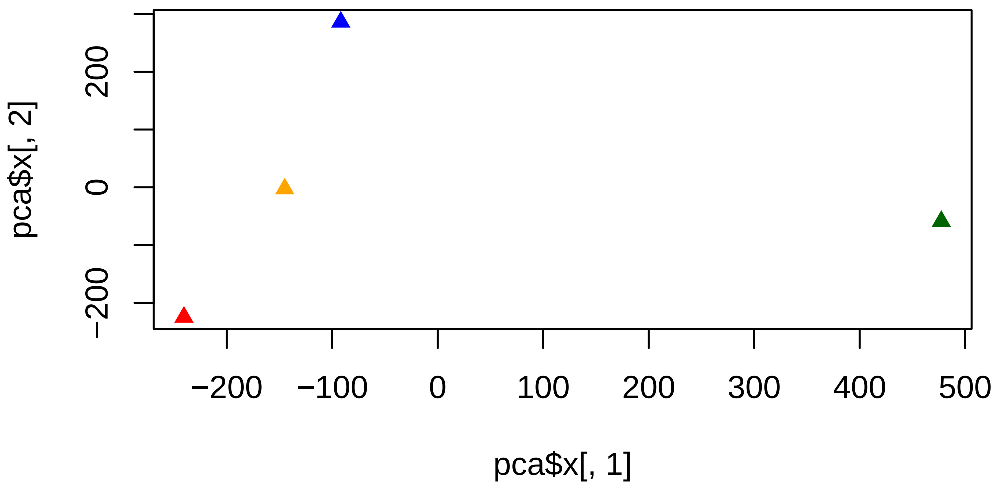
```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 3.176e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

A PCA plot aka score plot, PC1 vs PC2 plot, etc…

```
pca$x
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -4.894696e-14
Wales     -240.52915 -224.646925 -56.475555  5.700024e-13
Scotland   -91.86934  286.081786 -44.415495 -7.460785e-13
N.Ireland  477.39164  -58.901862  -4.877895  2.321303e-13
```

```
plot(pca$x[,1], pca$x[,2], col = (c("orange", "red", "blue", "darkgreen")), pch = 17)
```



pca$x[, 2]
200
0
−200
−200
−100
0
100
200
300
400
500
pca$x[, 1]