Class 7: Machine Learning 1

Iris Lee (PID: A16297004)

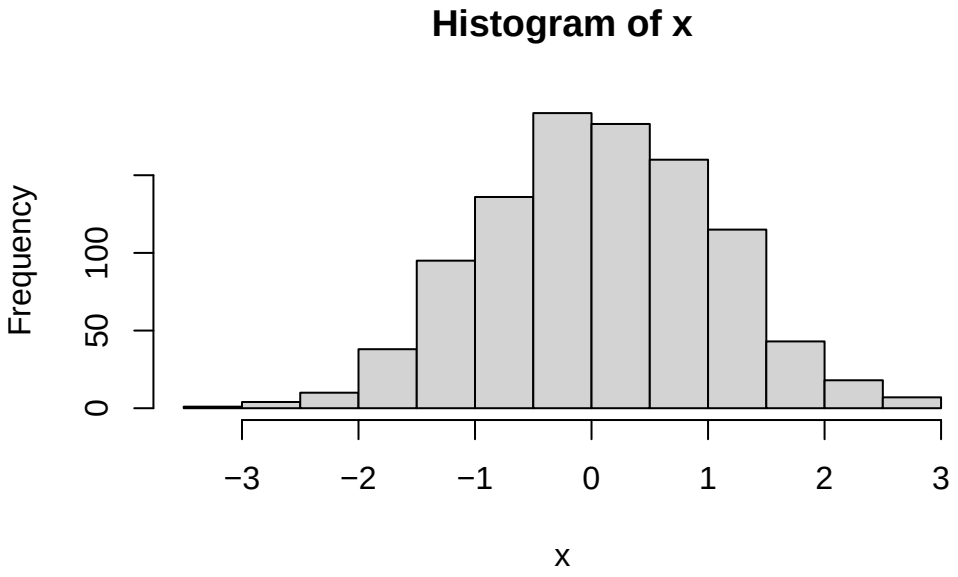
Clustering Methods

The broad goal here is to find groupings (clusters) in your input data.

Kmeans

First, let's make up some data to cluster.

```
x <- rnorm(1000)
hist(x)
```



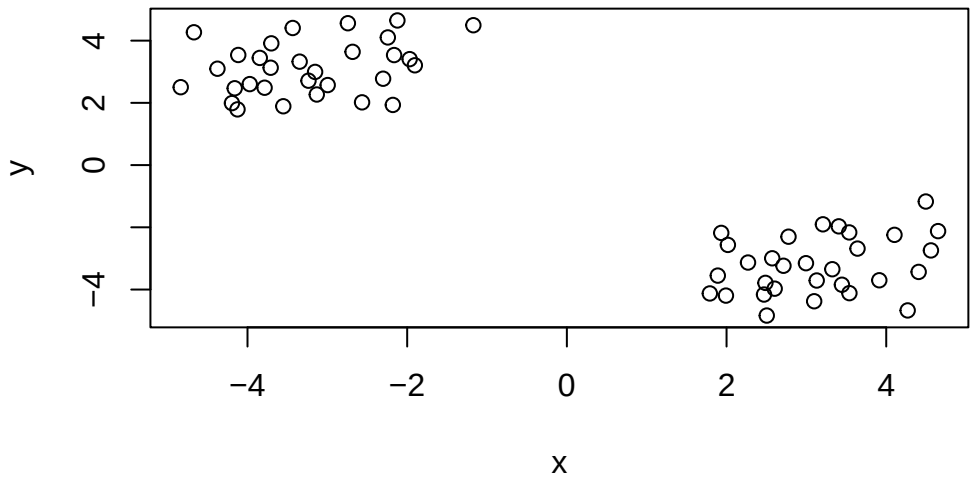
Make a vector of length 60 with 30 points centered at -3 and 30 points centered at +3.

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

```
 [1] -4.376336 -2.122595 -2.742746 -2.683908 -4.125161 -4.671124 -3.701893
 [8] -2.164027 -3.132214 -3.971938 -3.845993 -3.236149 -4.116009 -4.194790
[15] -3.785874 -3.709892 -1.968941 -2.179738 -4.160514 -3.433814 -3.153487
[22] -4.836367 -3.346877 -2.995577 -1.171389 -2.301511 -1.904066 -2.565231
[29] -2.242395 -3.551980  1.890222  4.103201  2.016624  3.207045  2.775847
[36]  4.495329  2.571951  3.324376  2.503555  2.994397  4.405923  2.469189
[43]  1.933544  3.405481  3.129252  2.486682  1.992026  3.538532  2.712469
[50]  3.444799  2.602686  2.268872  3.536250  3.912993  4.268486  1.790597
[57]  3.640046  4.559664  4.649124  3.097373
```

I will now make a wee x and y dataset with 2 groups of points.

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



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

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

Cluster means:
          x         y
1  3.124218 -3.213084
2 -3.213084  3.124218

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] 46.83414 46.83414
 (between_SS / total_SS =  92.8 %)

Available components:

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

A bit limitation of kmeans is that it does what you ask even if you ask for silly clusters.

Hierarchical Clustering

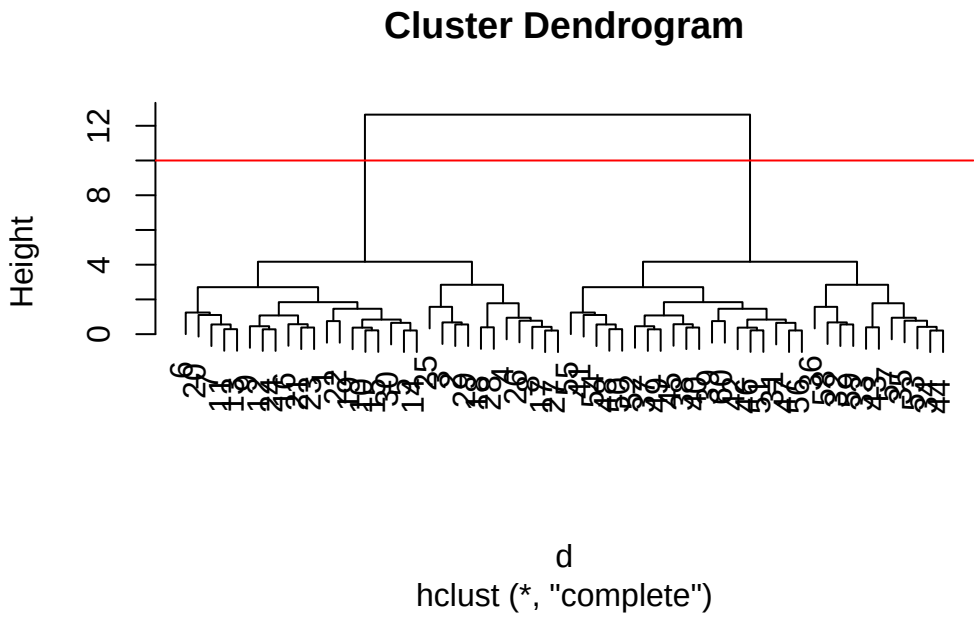
The main base R function for Hierarchical Clustering is `hclust()`. Unlike `kmeans()` you can not just pass it your data as input. You first need to calculate a distance matrix.

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

```
Call:
hclust(d = d)

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

Use `plot()` to view results

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



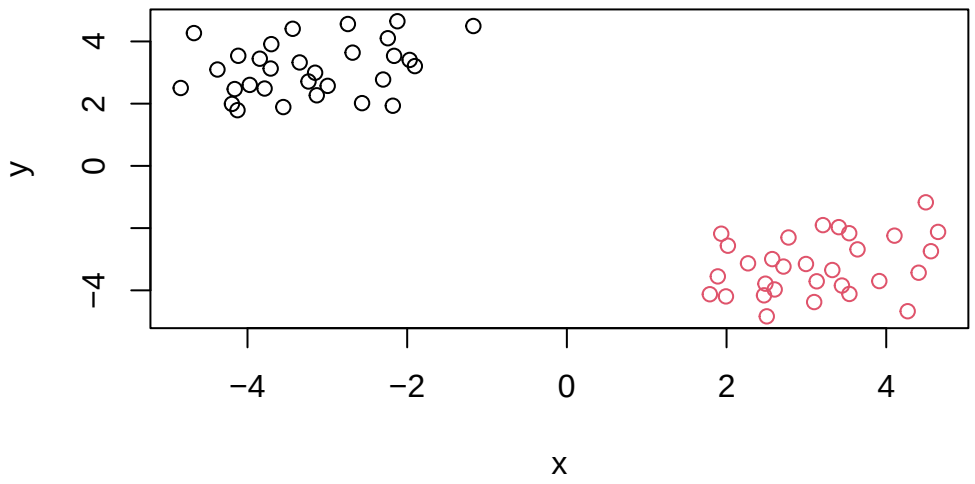
To make the "cut" and get out cluster membership vector we can use the `cutree()` function.

```
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 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
```

Make a plot of our data colored by hclust results.

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



Principal Component Analysis (PCA)

Here we will do Principal Component Analysis (PCA) on some food data from the UK.

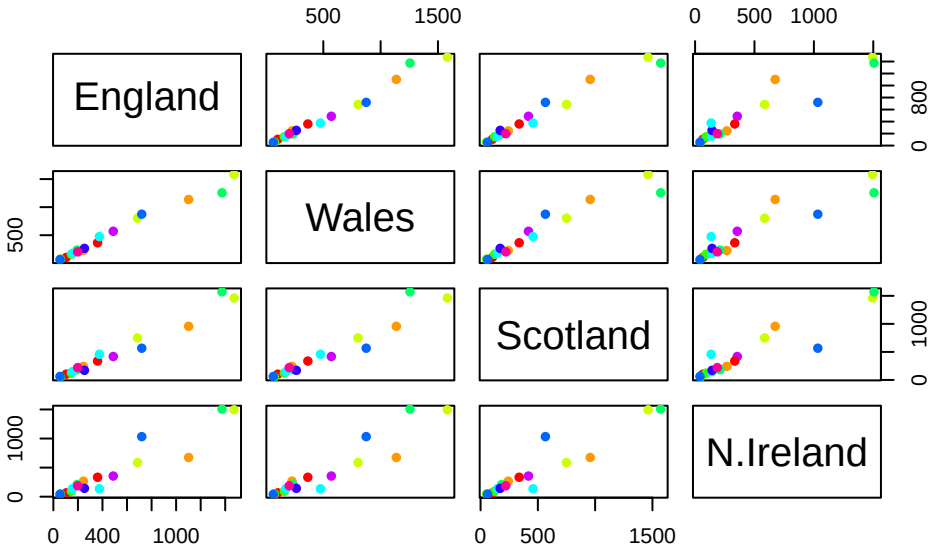
```
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url, row.names=1)
x
```

	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
Fresh_potatoes	720	874	566	1033
Fresh_Veg	253	265	171	143
Other_Veg	488	570	418	355
Processed_potatoes	198	203	220	187
Processed_Veg	360	365	337	334
Fresh_fruit	1102	1137	957	674

```
Cereals                1472   1582      1462       1494
Beverages                57     73        53         47
Soft_drinks            1374   1256      1572       1506
Alcoholic_drinks        375    475       458        135
Confectionery            54     64        62         41
```

```
#rownames(x) <- x[,1]
#x <- x[,-1]
#x
```

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



PCA to the resuce

The main "base" R function for PCA is called `prcomp()`.

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

Q. How much variance is captured in 2 PCs

96.5%

To make our main "PC score plot" (a.k.a "PC1 vs PC2 plot", or "PC plot" or "ordination plot").

```
attributes(pca)
```

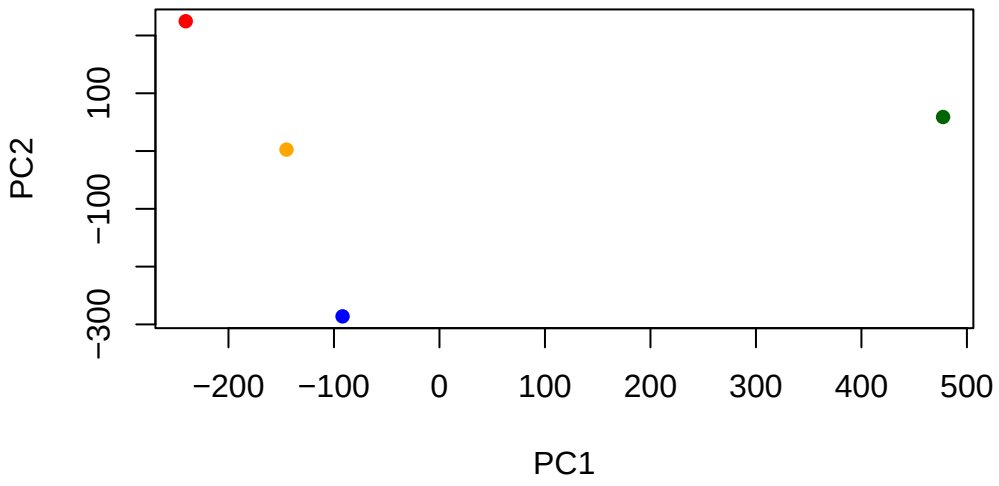
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

We are after the `pca$x` result component.

```
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
```

```
mycols <- c("orange", "red", "blue", "darkgreen")
plot(pca$x[,1], pca$x[,2], col=mycols, pch=16, xlab="PC1", ylab="PC2")
```



Another important result from PCA is how the original variables (in this case the foods) contribute to the PCs.

This is contained in the `pca$rotation` object - folks often call this the "loadings" or "contributions" to the PCs.

```
pca$rotation
```

	PC1	PC2	PC3	PC4
Cheese	-0.056955380	-0.016012850	-0.02394295	-0.691718038
Carcass_meat	0.047927628	-0.013915823	-0.06367111	0.635384915
Other_meat	-0.258916658	0.015331138	0.55384854	0.198175921
Fish	-0.084414983	0.050754947	-0.03906481	-0.015824630
Fats_and_oils	-0.005193623	0.095388656	0.12522257	0.052347444
Sugars	-0.037620983	0.043021699	0.03605745	0.014481347
Fresh_potatoes	0.401402060	0.715017078	0.20668248	-0.151706089
Fresh_Veg	-0.151849942	0.144900268	-0.21382237	0.056182433
Other_Veg	-0.243593729	0.225450923	0.05332841	-0.080722623
Processed_potatoes	-0.026886233	-0.042850761	0.07364902	-0.022618707
Processed_Veg	-0.036488269	0.045451802	-0.05289191	0.009235001
Fresh_fruit	-0.632640898	0.177740743	-0.40012865	-0.021899087
Cereals	-0.047702858	0.212599678	0.35884921	0.084667257

```
Beverages           -0.026187756  0.030560542  0.04135860 -0.011880823
Soft_drinks          0.232244140 -0.555124311  0.16942648 -0.144367046
Alcoholic_drinks    -0.463968168 -0.113536523  0.49858320 -0.115797605
Confectionery       -0.029650201 -0.005949921  0.05232164 -0.003695024
```

We can make a plot along PC1.

```
library(ggplot2)
contrib <- as.data.frame(pca$rotation)
ggplot(contrib) +
  aes(PC1, rownames(contrib)) +
  geom_col()
```