````
---
title: "class07"
format: pdf
editor: visual
---

## 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 `rnorm()` function to get random numbers from a normal distribution around a given `mean`.

```{r}
hist( rnorm(5000, mean=3))
```
Let's get 30 points with a mean of 3. ANother 30 with a mean of -3. Then make a matrix whose plot has two clusters at two opposite corners in the Euclidean space.

```{r}
tmp <- c(rnorm(30, mean=3), rnorm(30, mean=-3))
x <- cbind(tmp, rev(tmp))
plot(x)
```

## K-means clustering

Very popular clusterring method that we can use with the `kmeans()` function in base R

```{r}
km <- kmeans(x, centers = 2)
km
```

```{r}
km$size
```

```{r}
km$cluster
```

```{r}
km$centers
```
````

```{r}
plot(x, col=km$cluster)
points(km$centers, col="blue", pch=15, cex=3)
```

> Q. Let's cluster into 3 groups or same `x` data and make a plot

```{r}
km <- kmeans(x, centers = 4)
plot(x, col=km$cluster)
```

Hierarchial Clustering

We can use the `hclust()` function for Hierarchial 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.

```{r}
d <- dist(x)
hc <- hclust(d)
hc
```

```{r}
plot(hc)
```

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

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

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

```{r}
cutree(hc, k=2)
```
```{r}
plot(x, col=grps)
```

Principal Component Analysis (PCA)

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

> Q1. 17 rows, 4 columns

```{r}
dim(x)
```

> Q2. I prefer the row.names=1 approach because it is simpler and more convenient with larger data sets.

Exploratory analysis

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

```

> Q3. Setting beside=FALSE in the barplot() function results in the other type of graph.

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

```

>Q5. If the point lies on the diagonal for a given plot it means the points are equal.

The main PCA function in base R is called `prcomp()` it expects the transpose of our data

>Q6. Since the blue point is below the diagonal, Northern island has higher value than Scotland, England, and Wales.

```{r}
pca <- prcomp( t(x) )
summary(pca)
```

```{r}
pca$x
```

```{r}
plot(pca$x[,1], pca$x[,2],
     col=c("orange", "red", "blue", "darkgreen"), pch=16)
```

```{r}
# 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", "darkgreen"))
```