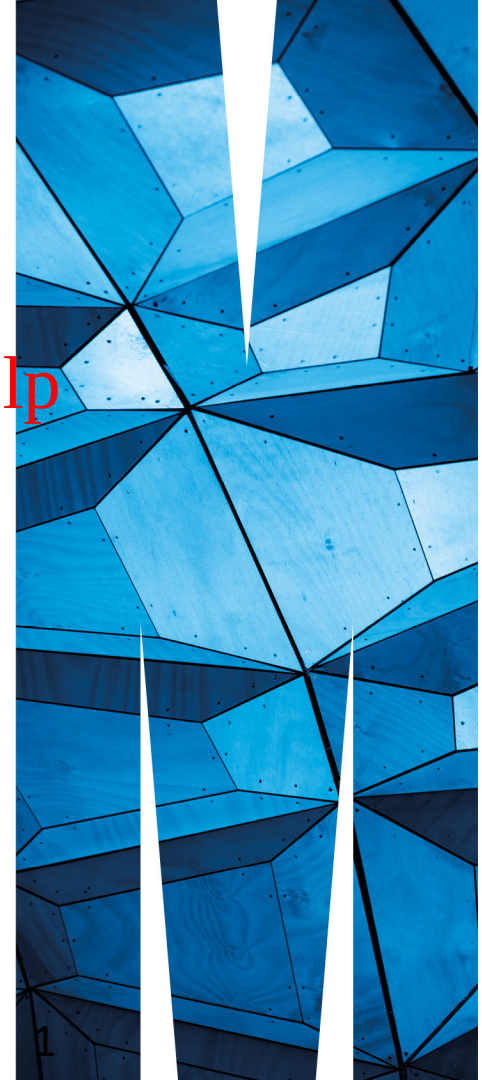
Cluster Analysis II

An Unsupervised Learning Technique

Presented by Klaus Ackermann



Clustering Methods

Partitional clustering

K-Means

1. Random assignment of points in the midst of cluster
2. Repeatedly re-assign points cluster
3. Minimise within-cluster distances
4. Maximise between-cluster distances

Clustering Algorithms

Hierarchical clustering	K-means clustering
• Suitable for small data set (<500 observations) • Easily examine solutions with increasing numbers of clusters • Convenient method if you want to observe how clusters are nested	• Suitable if you know how many clusters you want • Large data set • Good for summarising data with k "average" observations that describe the data with the minimum amount of error

Obtaining clusters: K means method

- Non-hierarchical clustering method
- Require number of clusters be specified before assigning observations
- No single objective method for determining the number of clusters

Obtaining clusters: K means method

Consider the following

- Want cohesion with, diversity between
  - How much improvement does one more cluster provide?
- Single member or very small clusters are not usually useful
- All clusters should be significantly different across the set of clustering variables
- The utility of the clusters ultimately depends on external validation
  - Other variables not used in the analysis
  - External information, e.g. location, geopolitical information

K-Means Clustering

MONASH University

- Given the value of K, the K-means algorithm **randomly** partitions the observations into K clusters
- After all observations have been assigned to a cluster, the resulting cluster *centroids* are calculated
- Using the updated cluster centroids, all observations are reassigned to the cluster with the *closest centroid* (usually Euclidean distance)
- *Continue* until the assignments no longer change or a predetermined maximum number of iterations is reached

K-Means Clustering

MONASH University

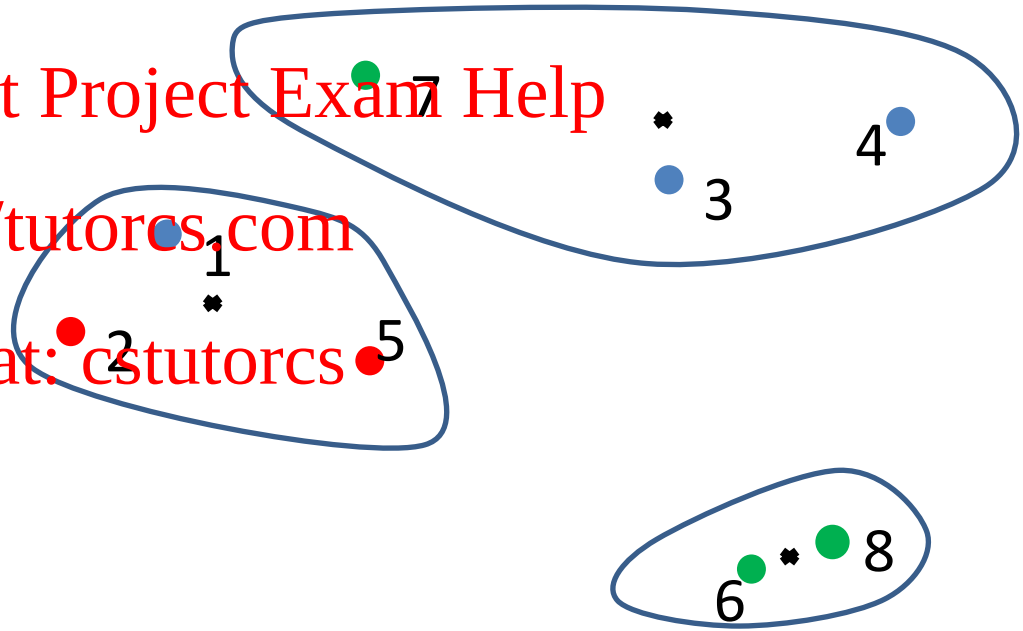
- Number of K clusters needs to be specified, usually advisable to try several values of K

- Initial K clusters are chosen randomly and the results do depend strongly on this particular starting point – procedure should be repeated a number of times (>10) with different starting points

- The “best” result is chosen

Recall: Centroid

- A centroid is the average observation of a cluster
- The average value for each variable across the observations in the cluster
- Centroid does not have to be an actual observation



Recall: KTC Example

- Example: continuous variables in KTC data
- These are: Age, Income, Children (columns 2, 4 and 6

ID	Age	Female	Income	Married	Children	CarLoan	Mortgage
1	48	1	17546	0	1	0	0
2	40	0	30085	1	3	1	1
3	51	1	16575	1	0	1	0
4	23	1	20375	1	3	0	0

- Use Euclidean distance: the variables will need standardising so that each of the three variables is on a similar scale. This is easily done in R.

Within Sum of Squares

- Use $\|\ldots\|$ to represent Euclidean distance in *n* dimensional space, where *n* is the number of variables
- Suppose we have *k* clusters, call them $S_1, \quad , S_k$
- We are trying to select the clusters in such a way as to minimise the "within sum of squares"

$$\sum_{j=1}^{k} \sum_{x \in S_j} \left\| x - \bar{m}_j \right\|^2$$

- Where x are the members of the clusters, and $\bar{m}_j$ is the centroid of cluster *j*

Total_SS and Between_SS

- Total_SS = sum of squared distances of each data point to the global sample mean
- Obtain between_SS:

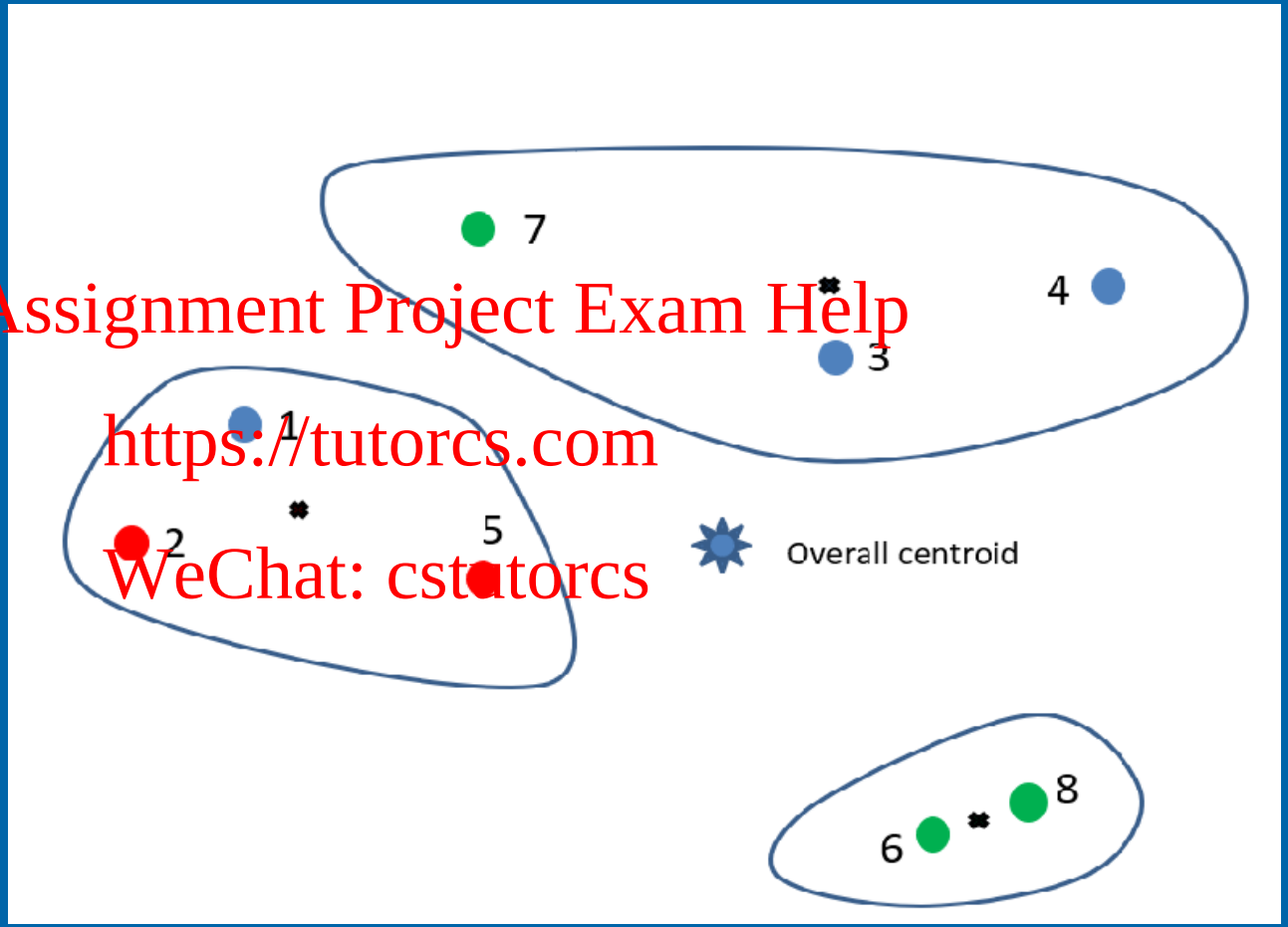
For each cluster, calculate the squared distance from the centroid of the cluster to the global sample mean

Multiply by the number of points in the cluster

Add up the results for each cluster

If there were no discernible pattern of clustering, the 3 means of the 3 groups would be close to the global mean, and between_SS would be a very small fraction of total_SS

MONASH University



Outcome Measure

- Since we want little variation within clusters and a lot of variation between clusters
  - The ratio *betweenss/totss* is a possible measure of how good the clustering is that K means has found

MONASH University

Example: Step by Step


```
ktc.df <- read.table("KTC.csv", header=TRUE, sep =",")
head(ktc.df)

ktcNumeric.df<- ktc.df %>%
    select(ID,Age,Income,Children) %>%
    column_to_rownames(var = 'ID') %>%
    scale
head(ktcNumeric.df)
km <- kmeans(ktcNumeric.df, centers=3, nstart=25)
km
```

	Age	Income	Children
1	0.1559026	-0.7637480	0.06169096
2	-0.4574848	0.1512843	1.91241969
3	0.3859229	-0.8346067	-0.86367341
4	-1.7609332	-0.5573020	1.91241969
5	0.8459635	1.6466130	-0.86367341
6	0.8459635	0.7193939	0.98705533

ID	Age	Female	Income	Married	Children	CarLoan	Mortgage
1	48	1	17546	0	1	0	0
2	40	0	30085	1	3	1	1
3	51	1	16575	1	0	1	0
4	23	1	20375	1	3	0	0
5	57	1	50576	1	0	0	0
6	57	1	37870	1	2	0	0

KTC Using K-Means

- *Centres* is the value of K, that is the number of clusters

- *nstart* is the number of times you run through the procedure
  - A recommended value is 25

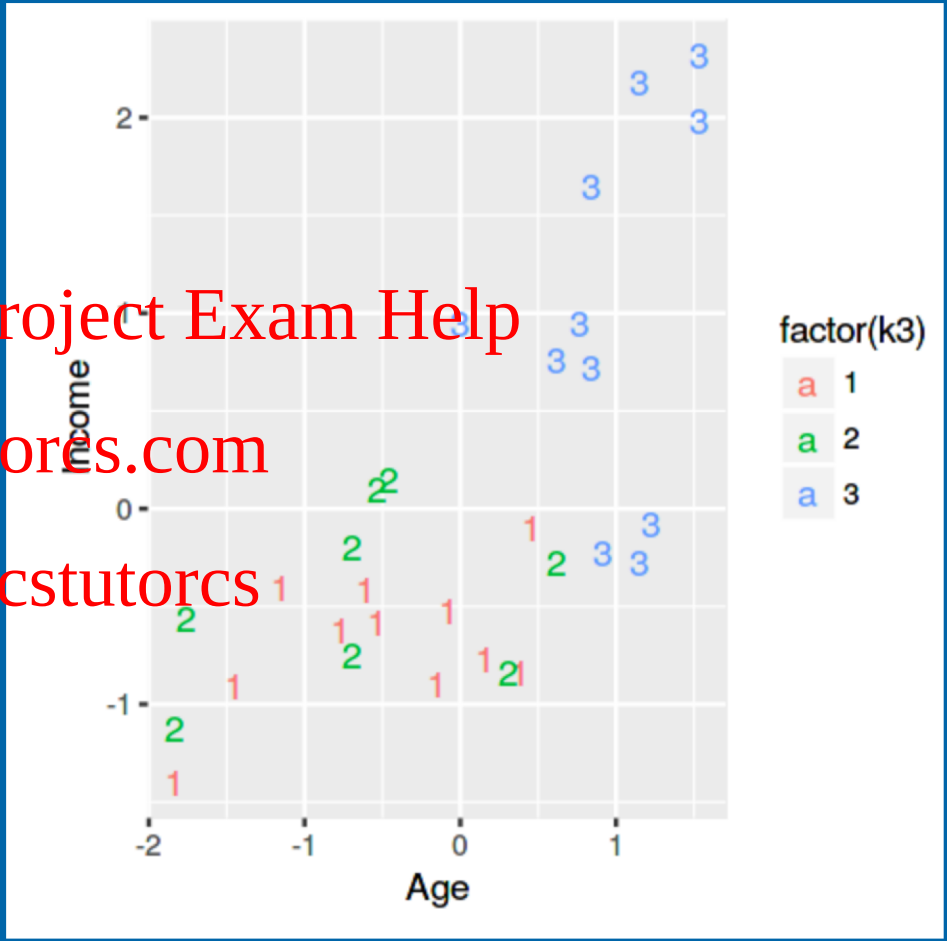
K-Means Output

```
K-means clustering with 3 clusters of sizes 11, 8, 11

Cluster means:
          Age      Income   Children
1 -0.5062770 -0.6752445 -0.6114013
2 -0.6300000 -0.4368752  1.3340670
3  0.9644588  0.9939719 -0.3589292

Clustering vector:
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
 1  2  1  2  3  3  1  3  2  2  3  1  1  3  1  1  2  3  3  1  3  2  3  1  2  3
27 28 29 30
 1  1  2  3

Within cluster sum of squares by cluster:
[1]  8.683382  8.282644 16.616678
 (between_SS / total_SS =  61.4 %)

Available components:

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

MONASH University

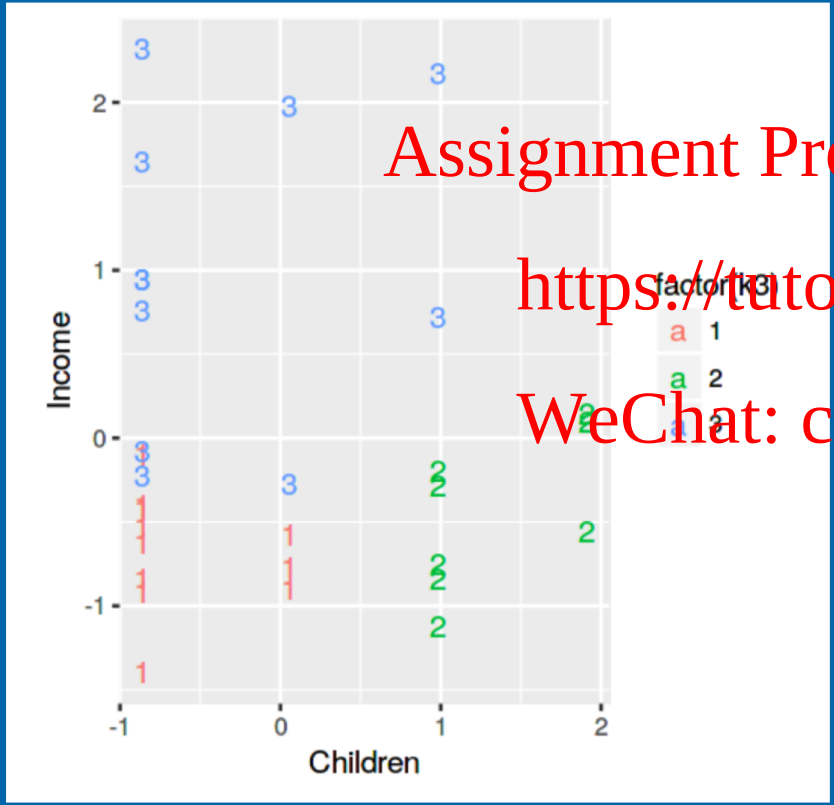
2-Dimension Graph

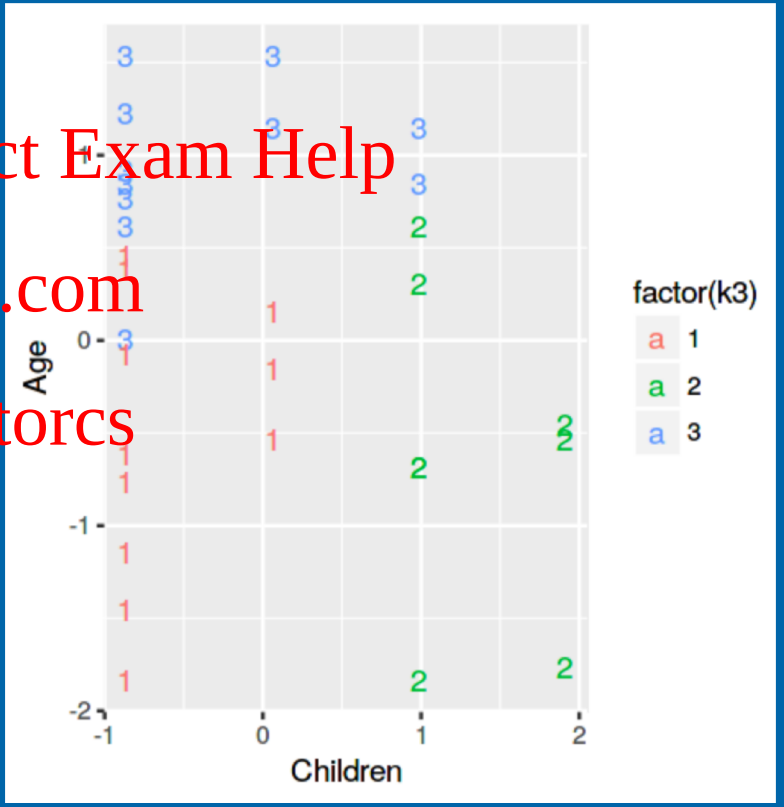

```
ktcNumeric.df %>%
    as_tibble() %>%
    mutate(k3=km$cluster) %>%
    ggplot(data = .,
       aes(x      = Age,
           y      = Income,
           colour = factor(k3),
           label  = k3)) +
        geom_text()
```



MONASH University

The Other Possibilities





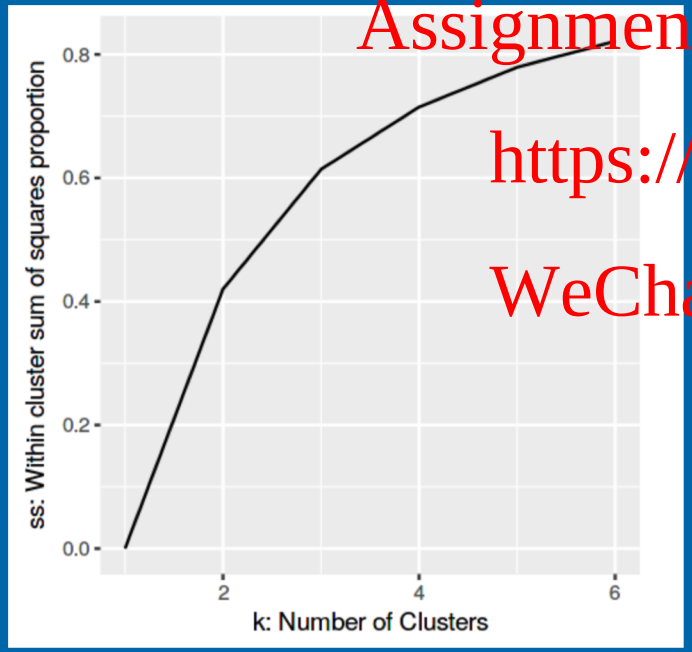
Choosing K

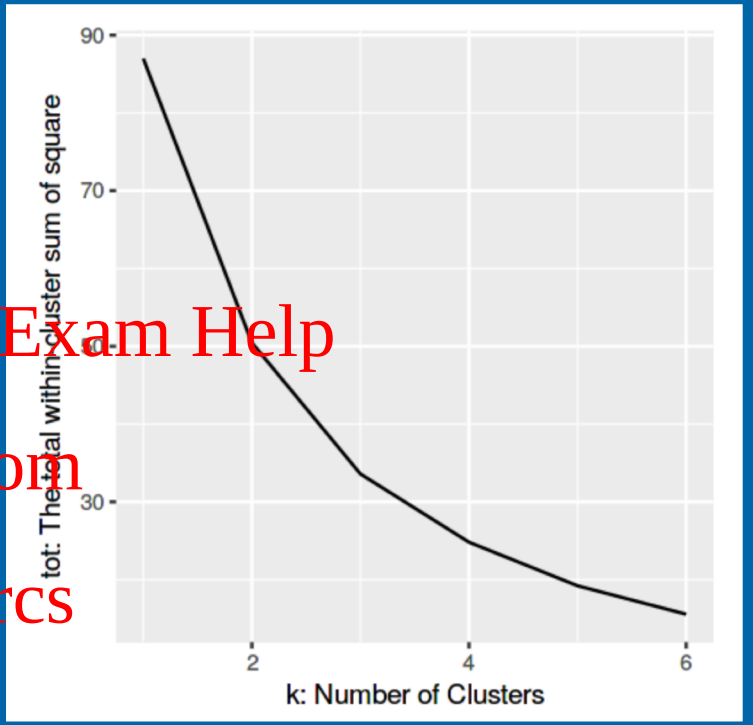
- The choice of k needs to be varied. Rather than trying out possible values, write code to do this:


```
n <- 6
ss.df <- data.frame(k = numeric(n),
                    ss = numeric(n))
for(i in 1:n){
    km          <- kmeans(ktcNumeric.df, centers=i, nstart=25) # calculate the clusters
    ss.df$k[i]  <- i # store the index of how many clusters
    ss.df$ss[i] <- km$betweenss/km$totss # store the calculated
}
```

MONASH University

Plot of Within Cluster Sum of Squares





The *total within-cluster sum of square* measures the compactness (i.e goodness) of the clustering and we want it to be as small as possible

MONASH University

MONASH BUSINESS SCHOOL

GROUP OF EIGHT AUSTRALIA

AACSB ACCREDITED

EFMD EQUIS ACCREDITED

ASSOCIATION OF MBAS ACCREDITED