Data Mining and Machine Learning

K-Means Clustering

Peter Jančovič

UNIVERSITY OF BIRMINGHAM

Objectives

- To explain the need for *K*-means clustering
- To understand the *K*-means clustering algorithm
- To understand the relationships between:
  - Clustering and statistical modelling using GMMs
  - *K*-means clustering and E-M estimation for GMMs

Clustering so far

- Agglomerative clustering
  - Begin by assuming that every data point is a separate centroid
  - Combine closest centroids until the desired number of clusters is reached
  - See `agglom.c` on the course Canvas page
- Divisive clustering
  - Begin by assuming that there is just one centroid/cluster
  - Split clusters until the desired number of clusters is reached

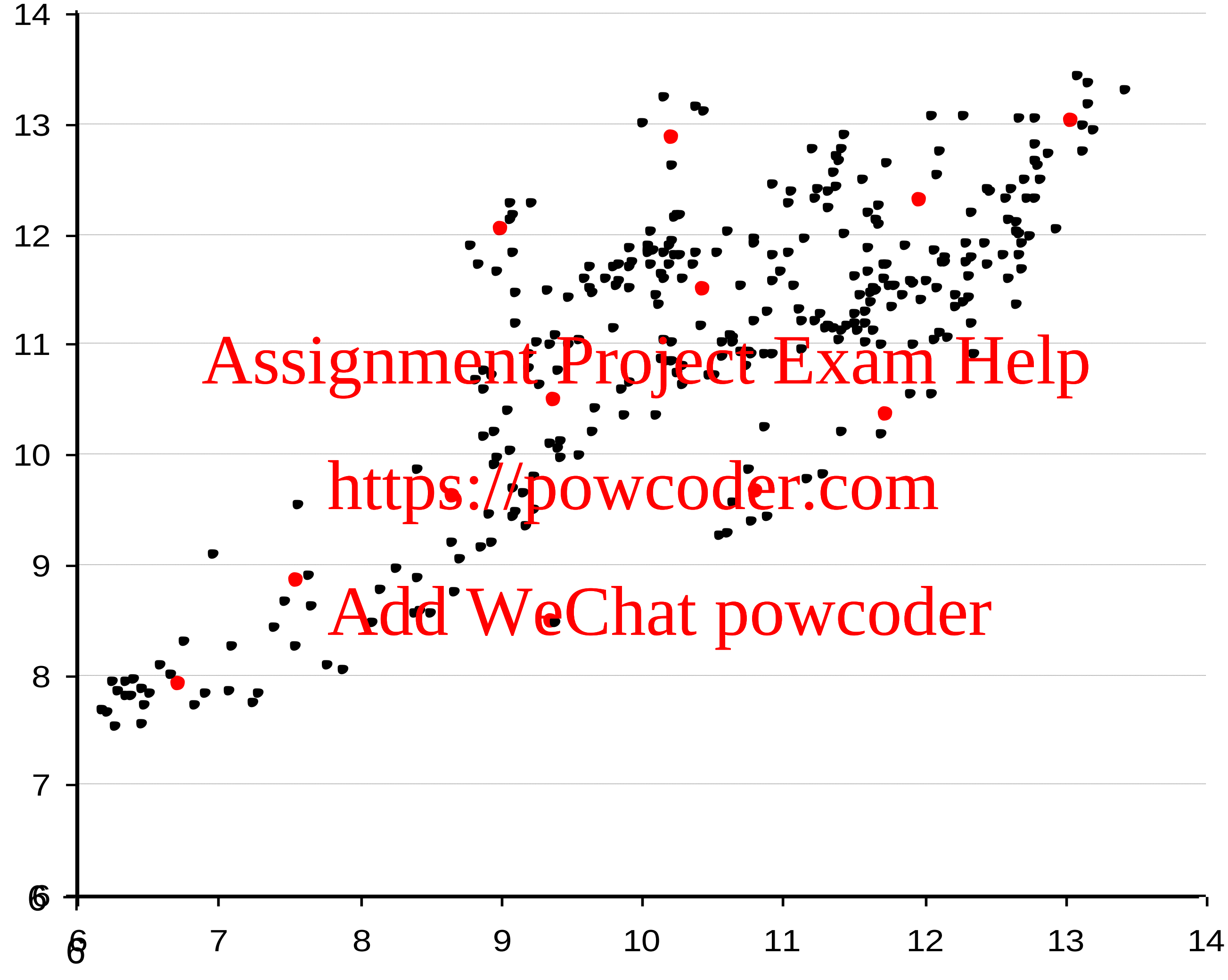
Optimality

- Neither agglomerative clustering nor divisive clustering is optimal
- In other words, the set of centroids which they give is not guaranteed to minimise distortion

Optimality continued

- For example:
  - In agglomerative clustering, a dense cluster of data points will be combined into a single centroid – but to minimise distortion, need several centroids in a region where there are many data points
  - A single 'outlier' may get its own cluster
- Agglomerative clustering provides a useful starting point, but further refinement is needed

12 centroids



K-means Clustering

- Suppose that we have decided how many centroids we need - denote this number by K
- Suppose that we have an initial estimate of suitable positions for our K centroids
- K-means clustering is an iterative procedure for moving these centroids to reduce distortion

Derivation of the K-means clustering algorithm

- Based on direct minimization of distortion
- Given a set of centroids $C^0 = c_1,\ldots,c_K$, and a set of data $Y = y_1,\ldots,y_N$, differentiating $Dist(C^0)$ with respect to the d^{th} component of c_k and setting the result to zero gives:

$$c_k^d = \frac{1}{|Y(k)|} \sum_{y_n \in Y(k)} y_n$$

where $Y(k)$ is the set of data points for which c_k is the closest centroid

Derivation of the K-means clustering algorithm (continued)

- The equation

$$c_k^d = \frac{1}{|Y(k)|} \sum_{y_n \in Y(k)} y_n$$

is not closed – both the LHS and the RHS depend on c_k

Although this equation cannot give a direct solution for c_k^d, it can be used as the basis of an iterative algorithm

K-means clustering - notation

- Suppose there are T data points, denoted by:

$$Y = y_1, y_2, \ldots, y_t, \ldots, y_T$$

- Suppose that the initial K clusters are denoted by:

$$C^0 = c_1^0, c_2^0, \ldots, c_k^0, \ldots, c_K^0$$

- One iteration of K-means clustering will produce a new set of clusters

$$C^1 = c_1^1, c_2^1, \ldots, c_k^1, \ldots, c_K^1$$

Such that

$$Dist\left(C^1\right) \le Dist\left(C^0\right)$$

K-means clustering (1)

- For each data point y_t let $c_{i(t)}$ be the <u>closest centroid</u>
- In other words: $d(y_t, c_{i(t)}) = \min_m d(y_t, c_m)$

- Now, for each centroid c^0_k define:

$$Y_k^0 = \{ y_t : i(t) = k \}$$

- In other words, Y^0_k is the set of data points which are closer to c^0_k than any other centroid

K-means clustering (2)

- Now define a new k^{th} centroid c^1_k by:

$$c_k^1 = \frac{1}{\left|Y_k^0\right|} \sum_{y_t \in Y_k^0} y_t$$

  where $|Y_k^0|$ is the number of samples in Y_k^0
- In other words, c^1_k is the average value of the samples which were closer to c^0_k than to any other centroid

K-means clustering (3)

- Now repeat the same process starting with the new centroids:

$$C^1 = c_1^1, c_2^1, \ldots, c_k^1, \ldots, c_K^1$$

to create a new set of centroids:

$$C^2 = c_1^2, c_2^2, \ldots, c_k^2, \ldots, c_K^2$$

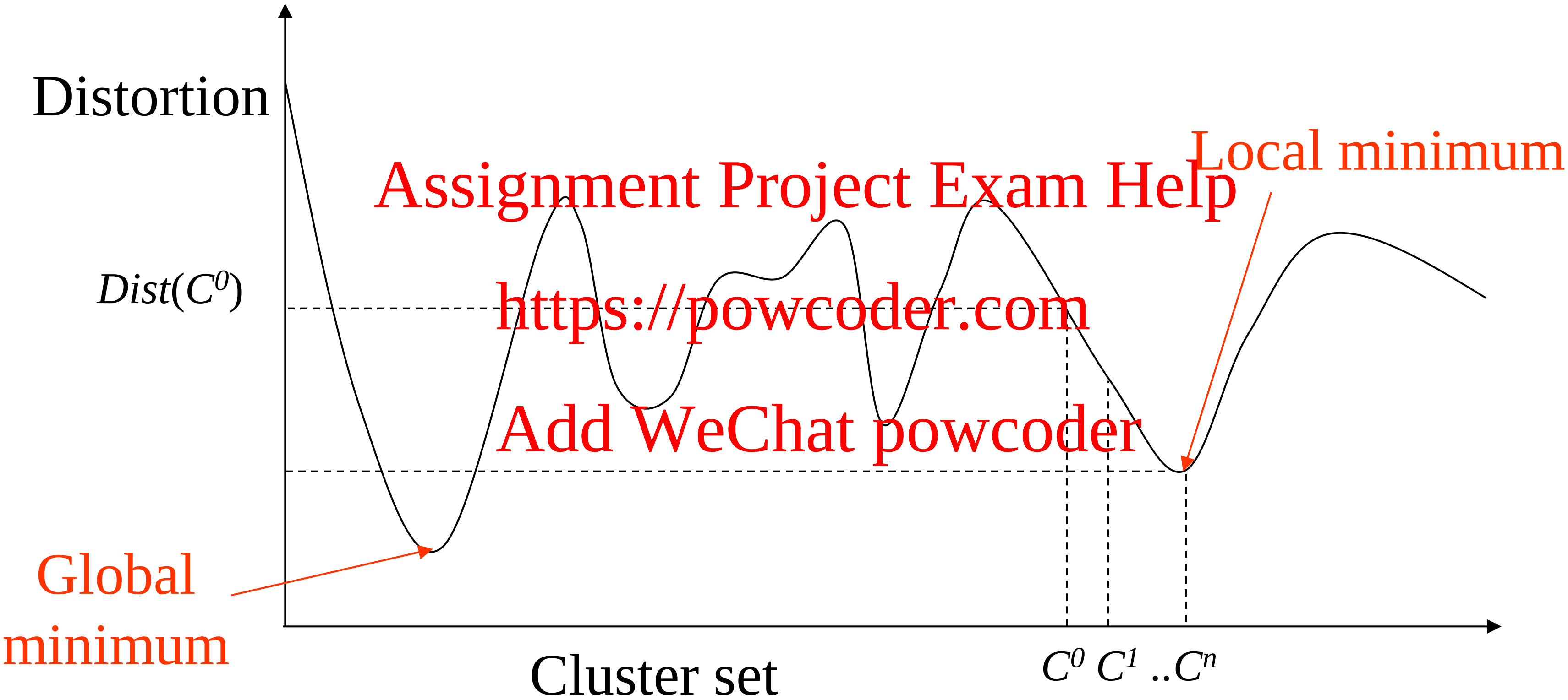
… and so on until the process converges

- Each new set of centroids has smaller distortion than the previous set

Initialisation

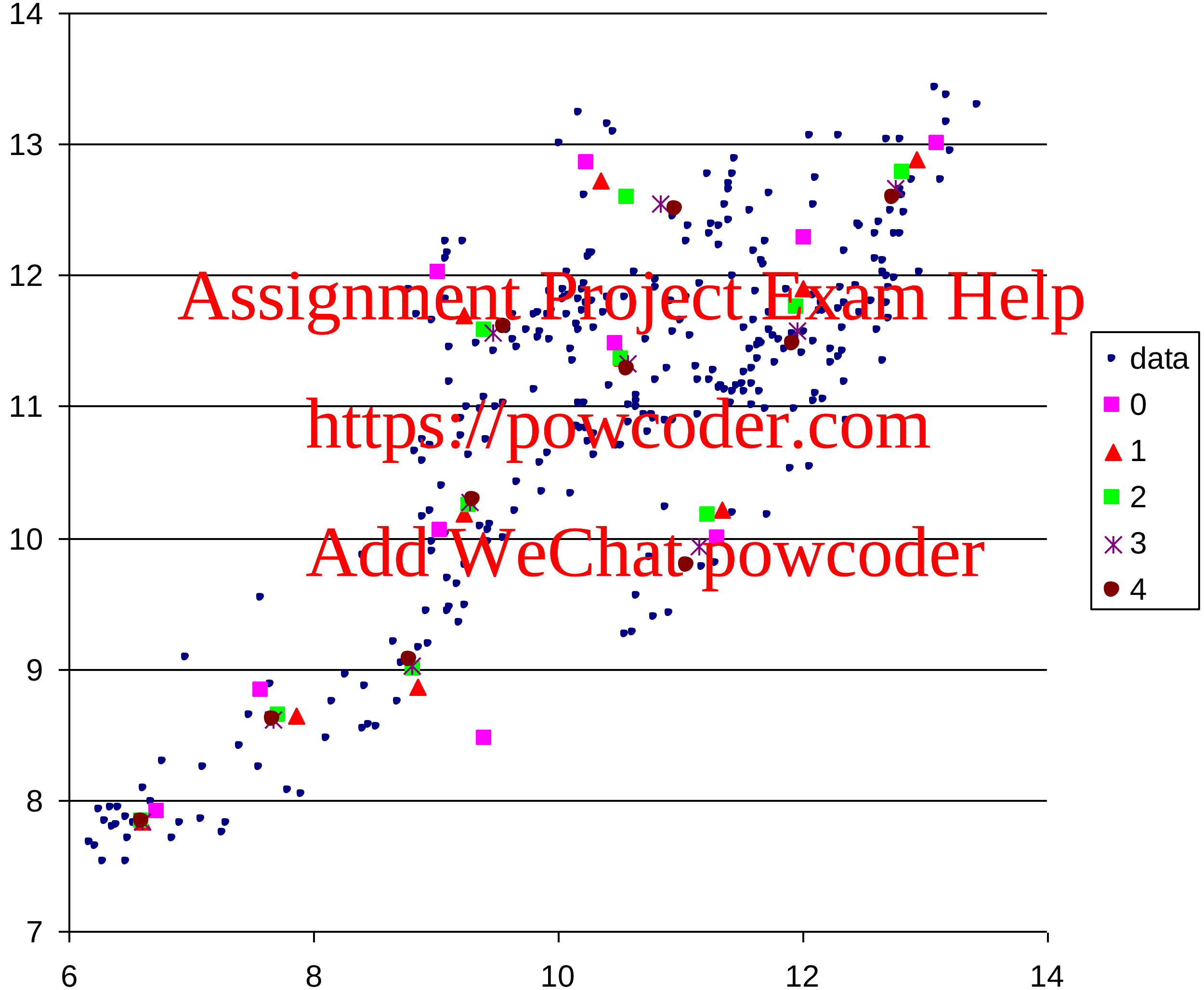
- An outstanding problem is to choose the initial centroid set C^0
- Possibilities include:
  - Choose C^0 randomly
  - Choose C^0 using agglomerative clustering
  - Choose C^0 using divisive clustering
- Choice of C^0 can be important
  - K-means clustering is a "hill-climbing" algorithm
  - Finds a local minimum of the distortion function
  - This local minimum is determined by C^0

Local optimality



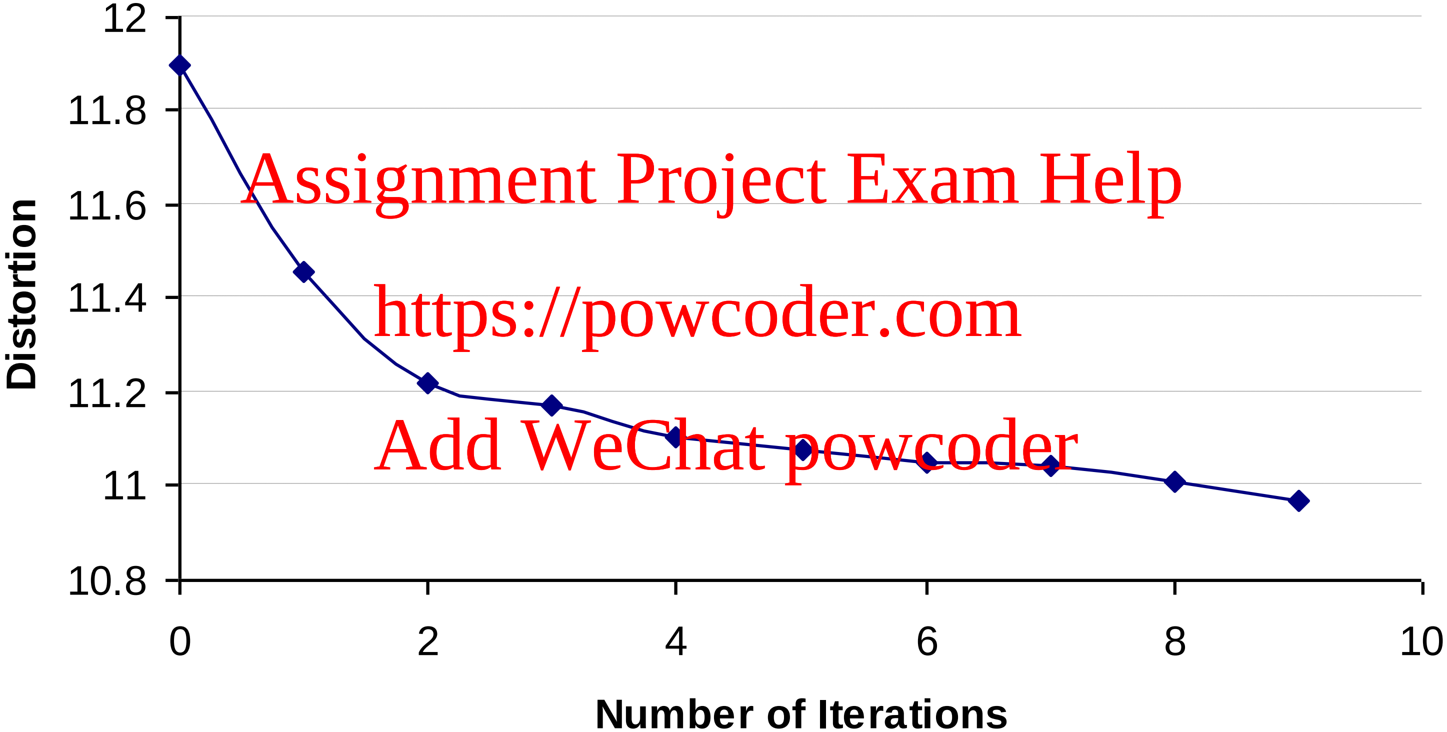
N.B: I've drawn the cluster set space as 1 dimensional for simplicity. In reality it is a very high dimensional space

Example



UNIVERSITY OF BIRMINGHAM

Example - distortion



UNIVERSITY OF BIRMINGHAM

C programs on Canvas

- `agglom.c`
  - Agglomerative clustering


```
agglom dataFile centFile numCent
```


  - Runs agglomerative clustering on the data in `dataFile` until the number of centroids is `numCent`. Writes the centroid (*x*,*y*) coordinates to `centFile`

C programs on Canvas

- `k-means.c`
  - *K*-means clustering


```
k-means dataFile centFile opFile
```

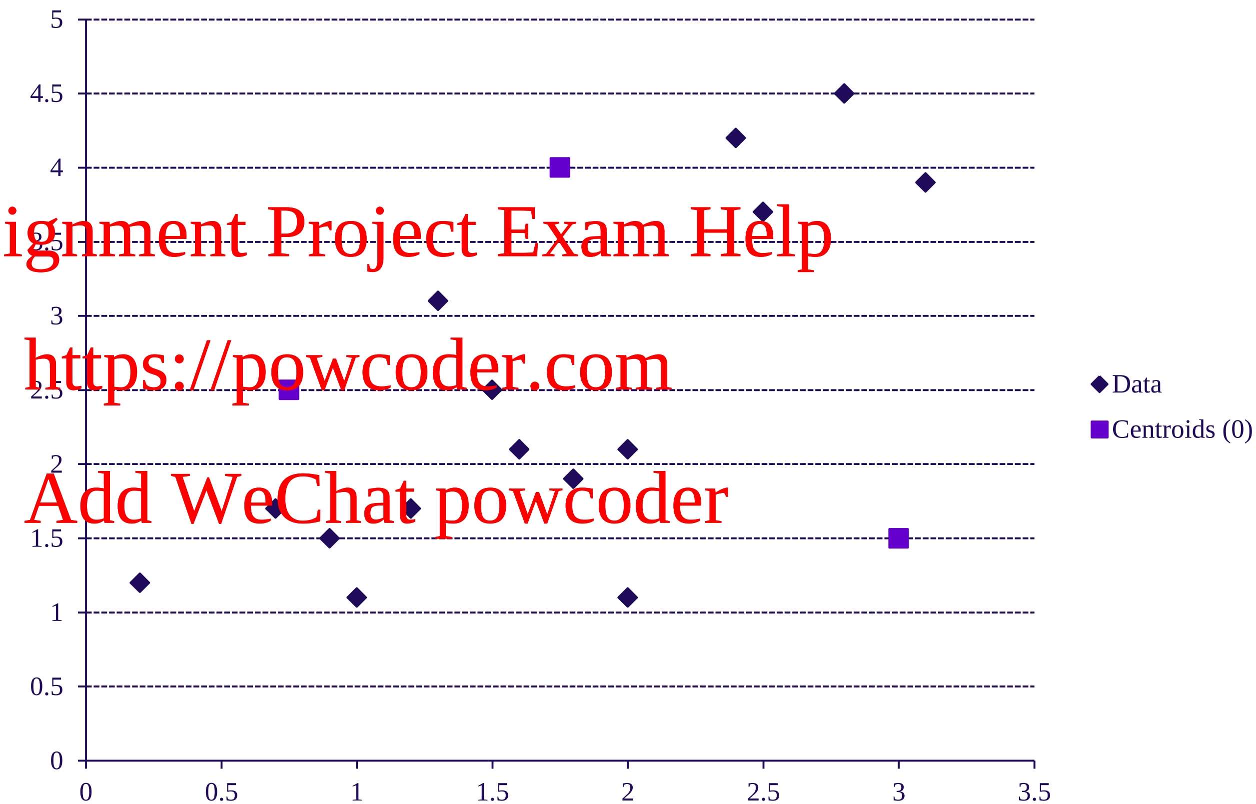

  - Runs 10 iterations of *k*-means clustering on the data in `dataFile` starting with the centroids in `centFile`.
  - After each iteration writes distortion and new centroids to `opFile`

Relationship with GMMs

- The set of centroids in clustering corresponds to the set of means in a GMM
- Measuring distances using Euclidean distance in clustering corresponds to assuming that the GMM variances are all equal to 1
- *k*-means clustering corresponds to the mean estimation part of the E-M algorithm, but:
  - In *k*-means samples are allocated 100% to the closest centroid
  - In E-M samples are shared between GMM components according to posterior probabilities

K-means clustering - example

Data	1.2	1.7
	1	1.1
	1.5	2.5
	2	2.1
	1.3	3.1
	1.8	1.9
	0.9	1.5
	0.2	1.2
	2	1.1
	2.5	3.7
	2.4	4.2
	3.1	3.9
	2.8	4.5
	1.6	2.1
	0.7	1.7
Centroids (0)	0.75	2.5
	3	1.5
	1.75	4



First iteration of *k*-means

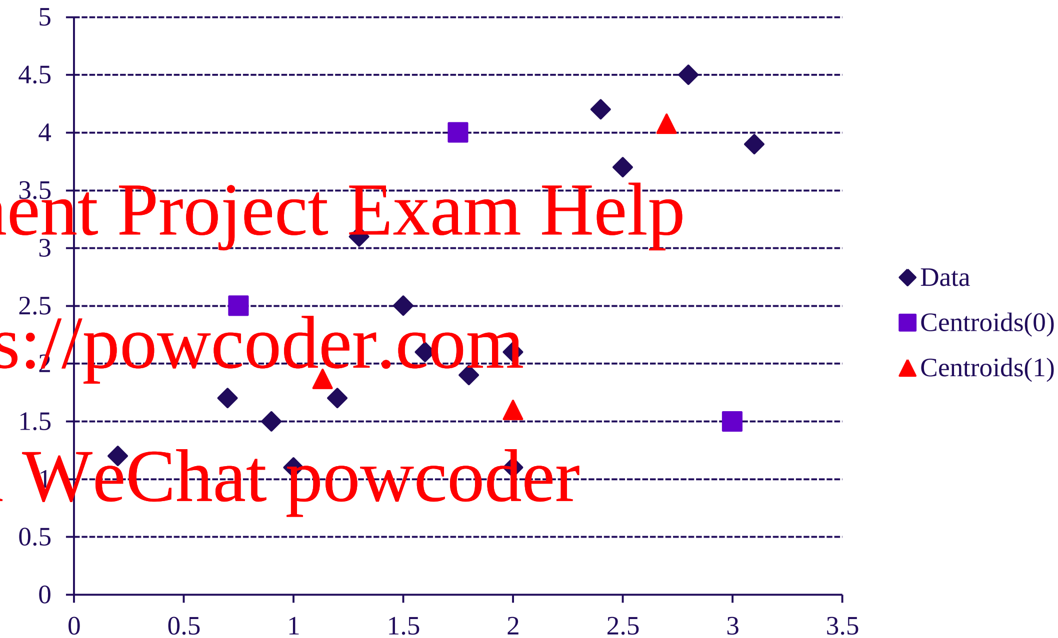
			Distance to centroids			Closest centroid		
			d(x(n),c(1))	d(x(n),c(2))	d(x(n),c(3))	c(1)	c(2)	c(3)
Data	1.2	1.7	0.92	1.81	2.36	1		
	1	1.1	1.42	2.04	3.00	1		
	1.5	2.5	0.75	1.80	1.52	1		
	2	2.1	1.31	1.17	1.92		1	
	1.3	3.1	0.81	2.33	1.01	1		
	1.8	1.9	1.21	1.26	2.10	1		
	0.9	1.5	1.01	2.10	2.64	1		
	0.2	1.2	1.41	2.82	3.20	1		
	2	1.1	1.88	1.08	2.91		1	
	2.5	3.7	2.12	2.26	0.81			1
	2.4	4.2	2.37	2.77	0.68			1
	3.1	3.9	2.74	2.40	1.35			1
	2.8	4.5	2.86	3.01	1.16			1
	1.6	2.1	0.94	1.52	1.91	1		
	0.7	1.7	0.80	2.31	2.53	1		
					Totals	9	2	4
Centroids (0)	0.75	2.5						
	3	1.5						
	1.75	4		**Distortion(0)**	**15.52**			

First iteration of *k*-means

			Distance to centroids			Closest centroid			c(1)		c(2)		c(3)	
			d(x(n),c(1))	**d(x(n),c(2))**	**d(x(n),c(3))**	**c(1)**	**c(2)**	**c(3)**	**x**	**y**	**x**	**y**	**x**	**y**
Data	1.2	1.7	0.92	1.81	2.36	1			1.20	1.70				
	1	1.1	1.42	2.04	3.00	1			1.00	1.10				
	1.5	2.5	0.75	1.80	1.52	1			1.50	2.50				
	2	2.1	1.31	1.17	1.92		1				2.00	2.10		
	1.3	3.1	0.81	2.33	1.01	1			1.30	3.10				
	1.8	1.9	1.21	1.26	2.10	1			1.80	1.90				
	0.9	1.5	1.01	2.10	2.64	1			0.90	1.50				
	0.2	1.2	1.41	2.82	3.20	1			0.20	1.20				
	2	1.1	1.88	1.08	2.91		1				2.00	1.10		
	2.5	3.7	2.12	2.26	0.81			1					2.50	3.70
	2.4	4.2	2.37	2.77	0.68			1					2.40	4.20
	3.1	3.9	2.74	2.40	1.35			1					3.10	3.90
	2.8	4.5	2.86	3.01	1.16			1					2.80	4.50
	1.6	2.1	0.94	1.52	1.91	1			1.60	2.10				
	0.7	1.7	0.80	2.31	2.53	1			0.70	1.70				
					Totals	9	2	4	10.2	16.8	4	3.2	10.8	16.3
Centroids (0)	0.75	2.5												
	3	1.5												
	1.75	4			**Dist'n(0)**	**15.52**								

First iteration of *k*-means

Data	1.2	1.7
	1	1.1
	1.5	2.5
	2	2.1
	1.3	3.1
	1.8	1.9
	0.9	1.5
	0.2	1.2
	2	1.1
	2.5	3.7
	2.4	4.2
	3.1	3.9
	2.8	4.5
	1.6	2.1
	0.7	1.7
Centroids (0)	0.75	2.5
	3	1.5
	1.75	4
Centroids (1)	1.133333	1.866667
	2	1.6
	2.7	4.075



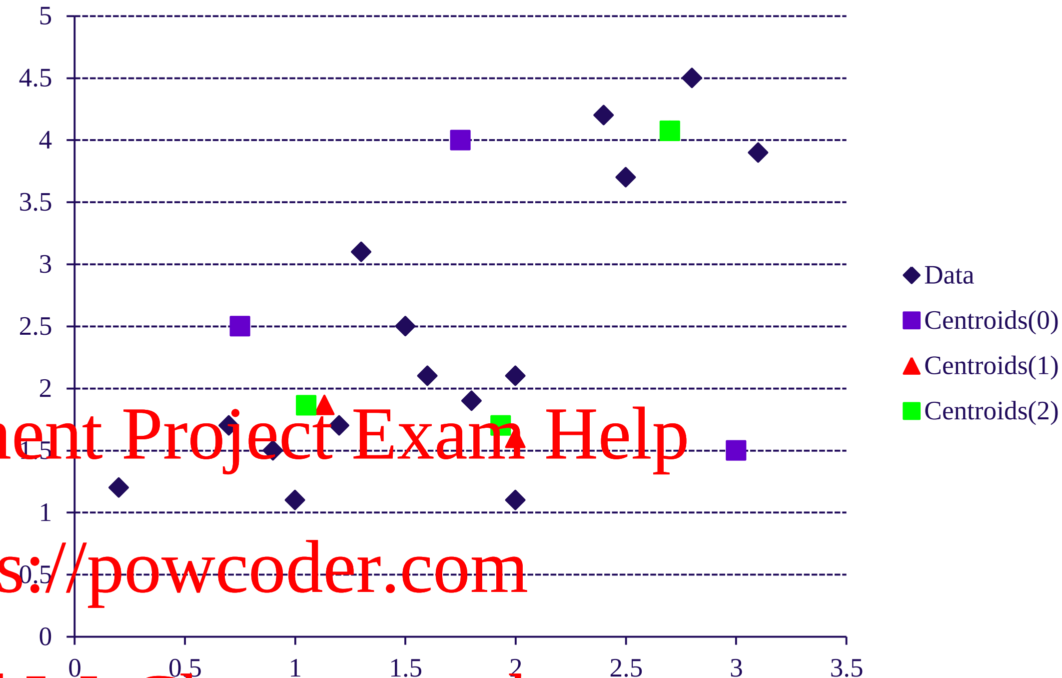
UNIVERSITY OF BIRMINGHAM

Second iteration of *k*-means

			Distance to centroids			Closest centroid		
			d(x(n),c(1))	d(x(n),c(2))	d(x(n),c(3))	c(1)	c(2)	c(3)
Data	1.2	1.7	0.18	0.81	2.81	1		
	1	1.1	0.78	1.12	3.43	1		
	1.5	2.5	0.73	1.03	1.98	1		
	2	2.1	0.90	0.50	2.10		1	
	1.3	3.1	1.24	1.66	1.71	1		
	1.8	1.9	0.67	0.36	2.35		1	
	0.9	1.5	0.43	1.10	3.14	1		
	0.2	1.2	1.15	1.84	3.81	1		
	2	1.1	1.16	0.50	3.06		1	
	2.5	3.7	2.29	2.16	0.43			1
	2.4	4.2	2.65	2.63	0.33			1
	3.1	3.9	2.83	2.55	0.44			1
	2.8	4.5	3.12	3.01	0.44			1
	1.6	2.1	0.52	0.64	2.26	1		
	0.7	1.7	0.46	1.30	3.10	1		
						8	3	4
Centroids(1)	1.133333	1.866667						
	2	1.6						
	2.7	4.075						

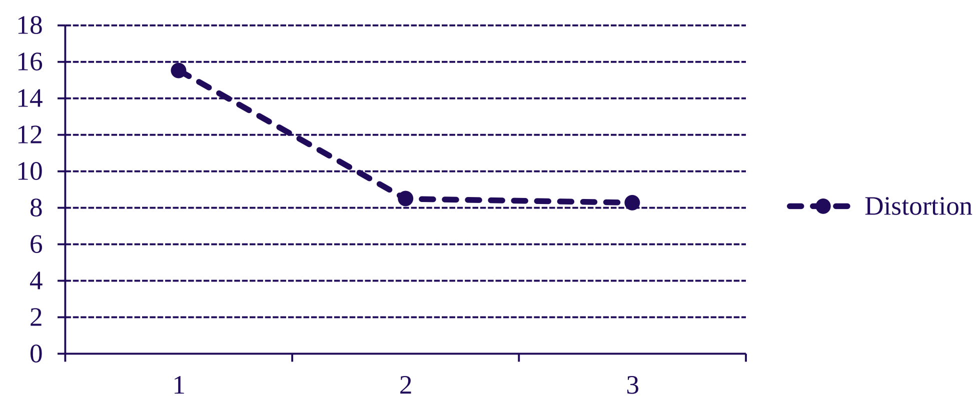
Second iteration of *k*-means

Data	1.2	1.7
	1	1.1
	1.5	2.5
	2	2.1
	1.3	3.1
	1.8	1.9
	0.9	1.5
	0.2	1.2
	2	1.1
	2.5	3.7
	2.4	4.2
	3.1	3.9
	2.8	4.5
	1.6	2.1
	0.7	1.7



Centroids(1)	1.133333	1.866667
	2	1.6
	2.7	4.075

Centroids (2)	1.05	1.8625
	1.933333	1.7
	2.7	4.075

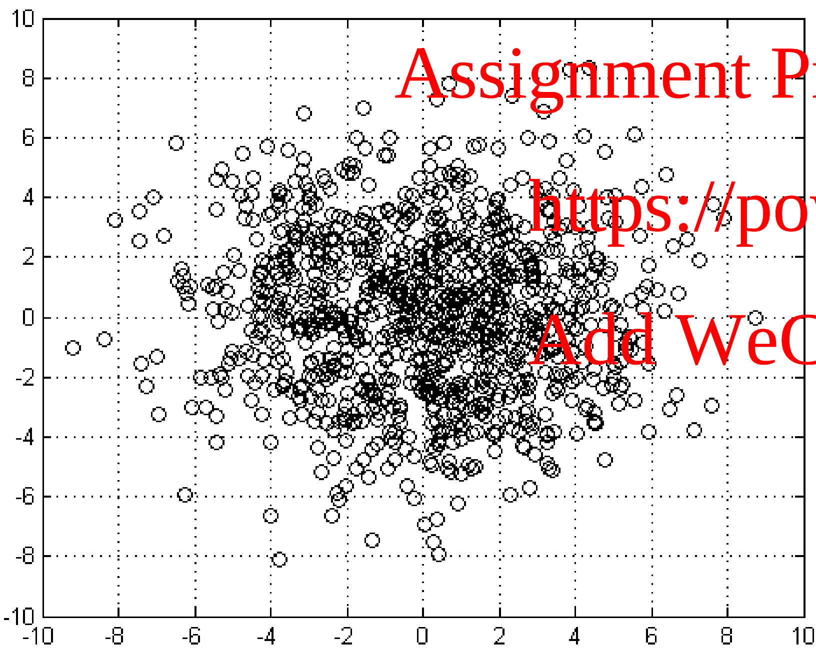


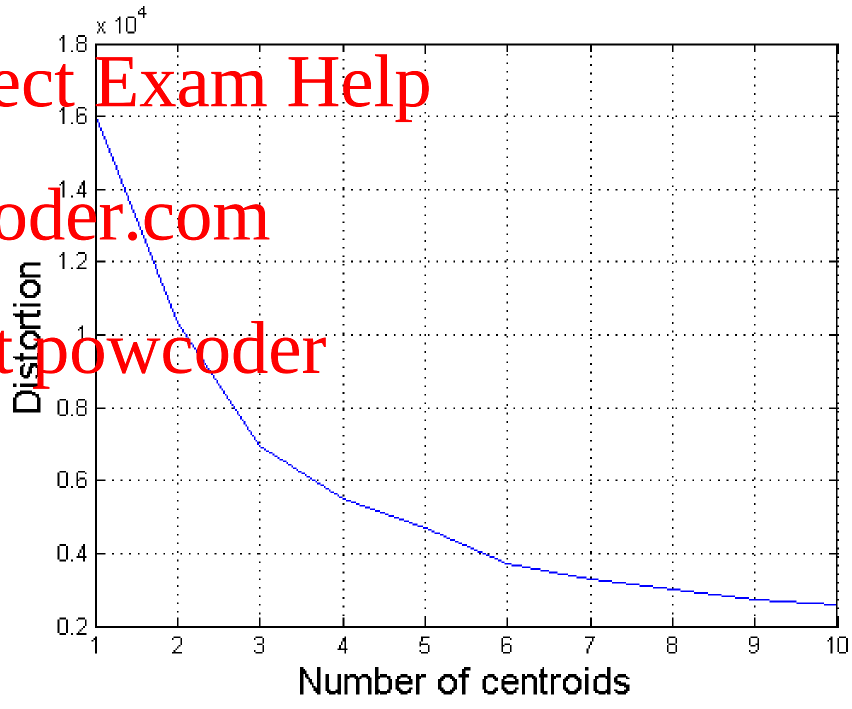
Examples

- Three example 2-dimensional datasets
- For each data set, and for *k*=1,...,10:
  - Create *k* centroids using agglomerative clustering
  - Run k-means algorithm for 15 iterations for these initial centroid values
  - Plot distortion after 15 iterations of *k*-means as a function of number of centroids

Example 1

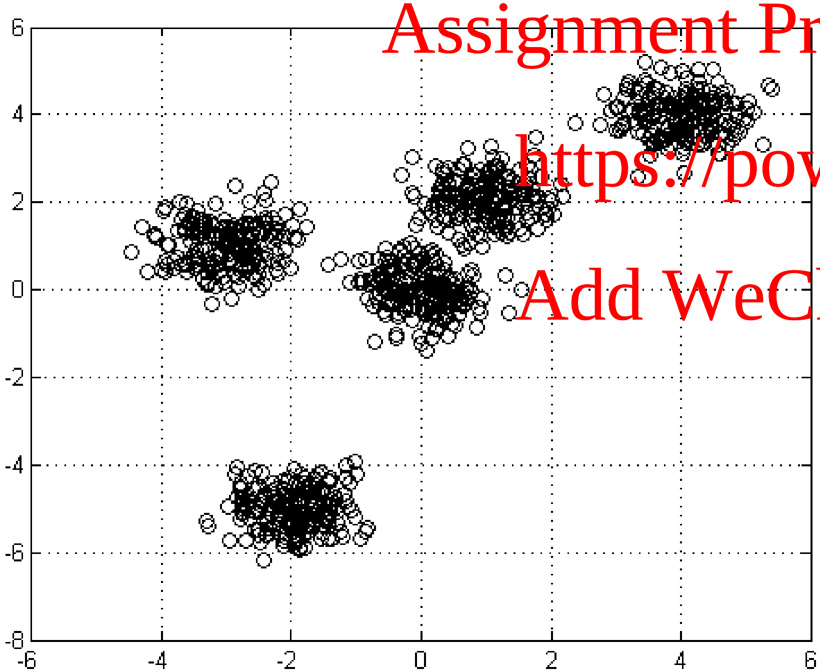
- Gaussian distributed data: single 2D Gaussian, centre (0,0), variance 16 in *x* and *y* directions

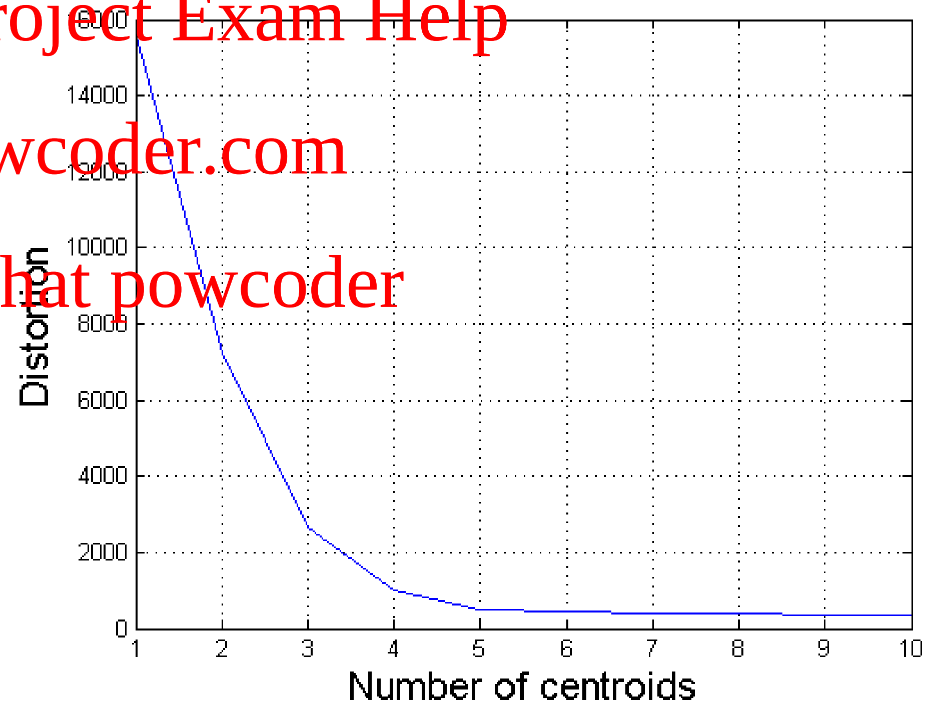




Example 2

- Five 2D Gaussians, centres (0,0), (1,2), (4,4), (-2,-5) and (-3,1), variance 0.5 in *x* and *y* directions

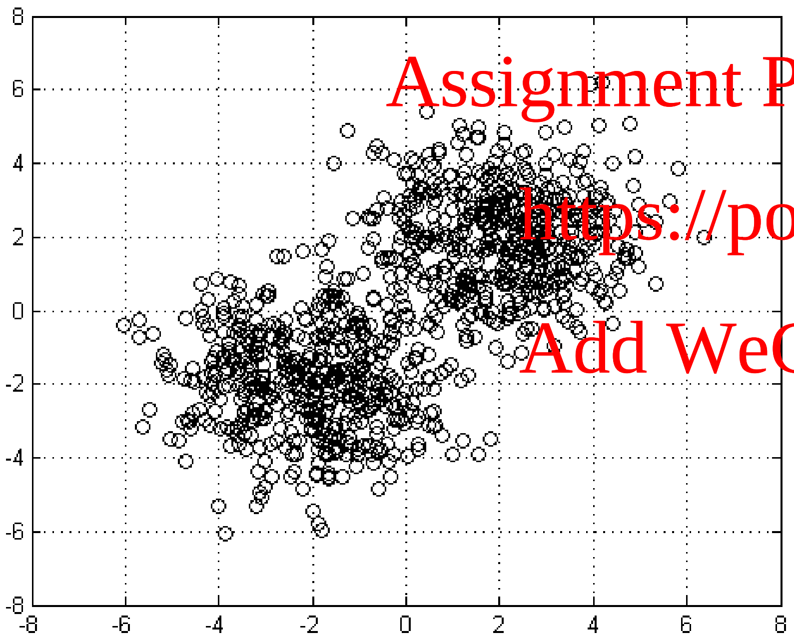


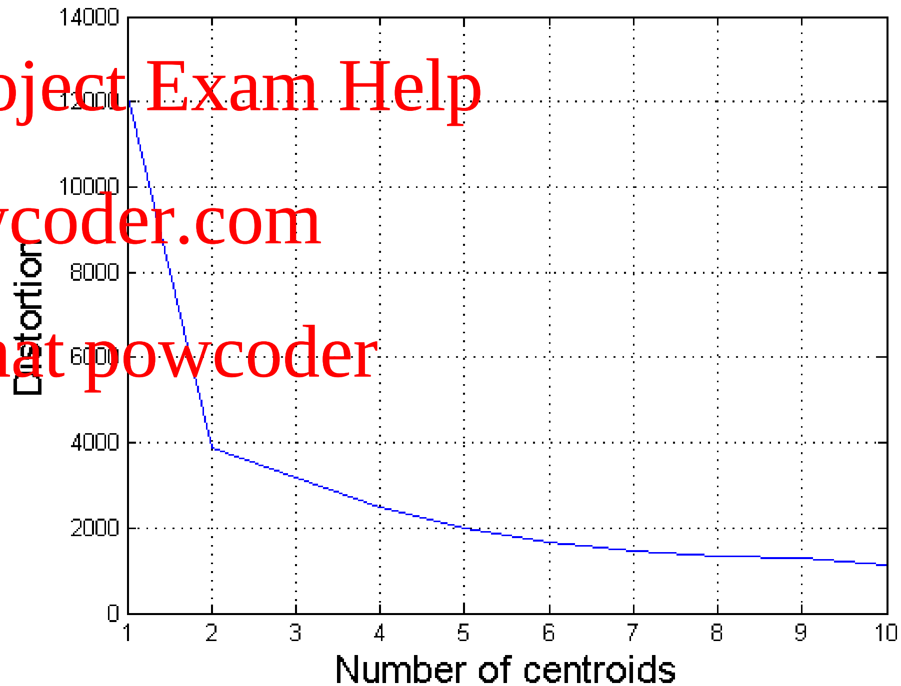


UNIVERSITY OF BIRMINGHAM

Example 3

- Two 2D Gaussians, centres (2,2), (-2,-2), variance 4 in *x* and *y* directions





Summary

- The need for *k*-means clustering
- The *k*-means clustering algorithm
- Example of *k*-means clustering
- Choosing *k* empirically