

Machine Learning & Artificial Intelligence for Data Scientists: Clustering (Part 1)

Fani Deligianni

<https://www.gla.ac.uk/schools/computing/staff/fanideligianni/>

School of Computing Science

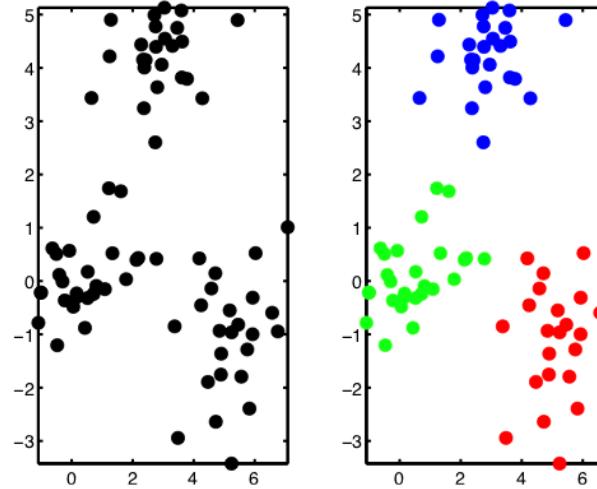
Unsupervised learning

- ▶ Everything we've seen so far has been *supervised*
- ▶ We were given a set of \mathbf{x}_n **and** associated t_n .
- ▶ What if we just have \mathbf{x}_n ?
- ▶ For example:
 - ▶ \mathbf{x}_n is a binary vector indicating products customer n has bought.
 - ▶ Can group customers that buy similar products.
 - ▶ Can group products bought together.
- ▶ Known as **Clustering**
- ▶ And is an example of *unsupervised* learning.
- ▶ We'll also cover *projection* (next week)

Aims

- ▶ Understand what clustering is.
- ▶ Understand the K-means algorithm.
- ▶ Understand the idea of mixture models.

Clustering



- ▶ In this example each object has two attributes:
$$\mathbf{x}_n = [x_{n1}, x_{n2}]^T$$
- ▶ Left: data.
- ▶ Right: data after clustering (points coloured according to cluster membership).

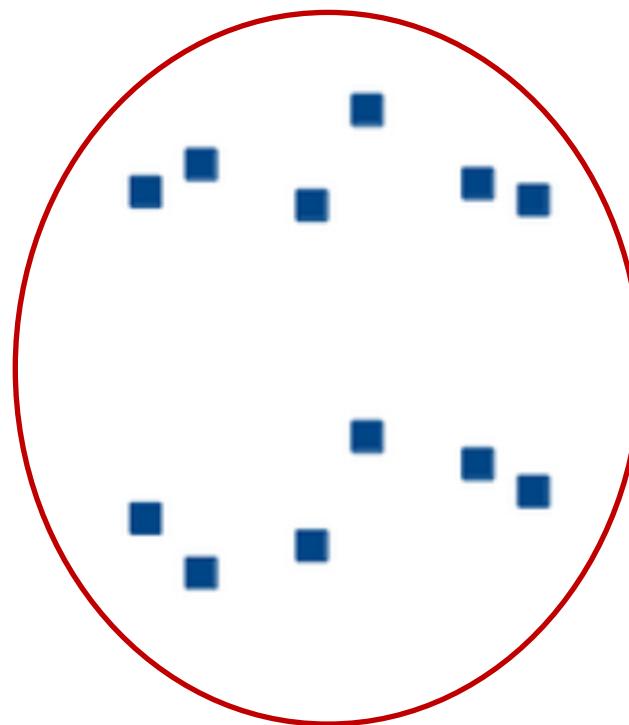
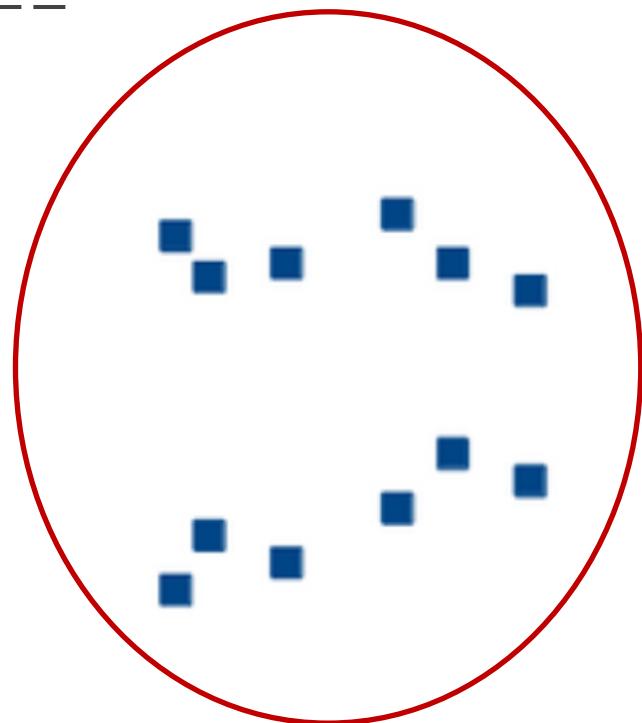
What we'll cover

- 2 algorithms
 - K-means
 - Mixture models
- The two are related
- We'll also see how K-means can be kernelised

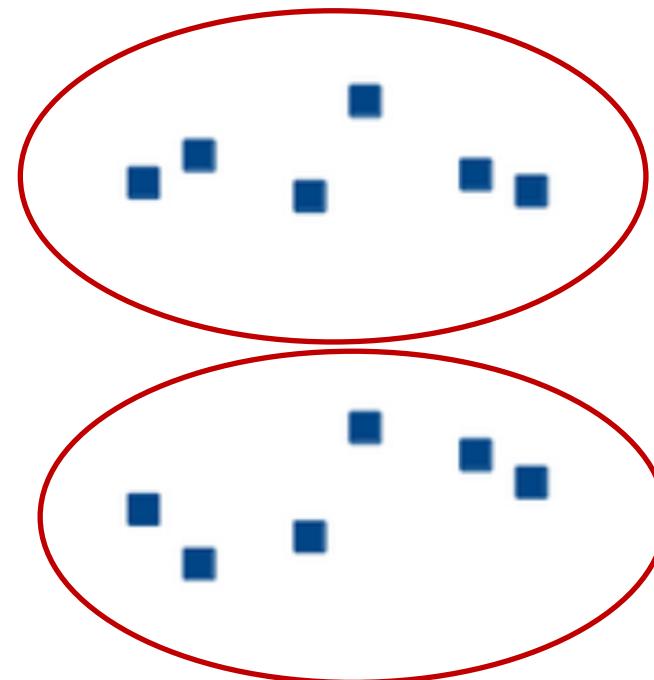
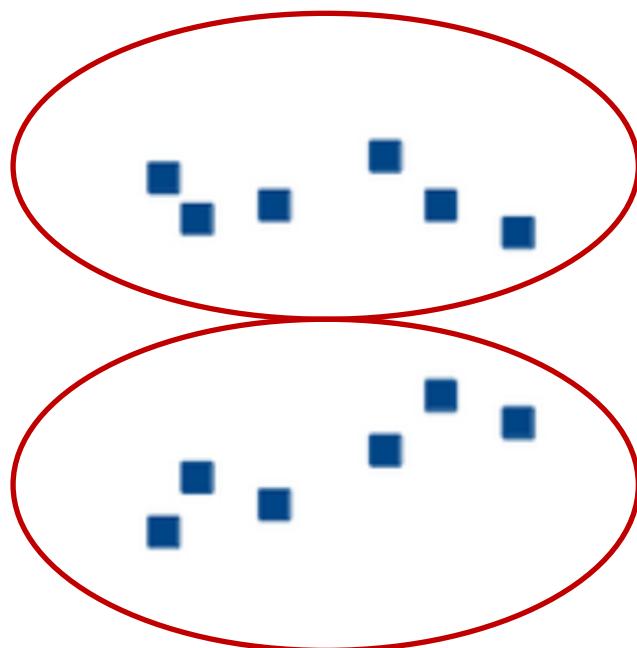
Number of clusters?



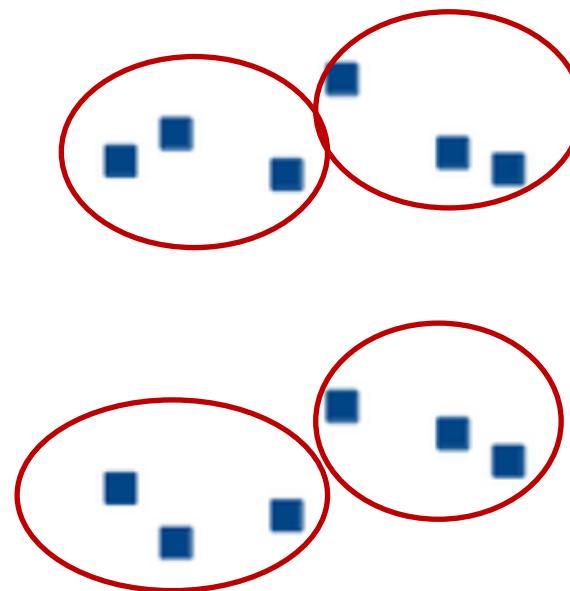
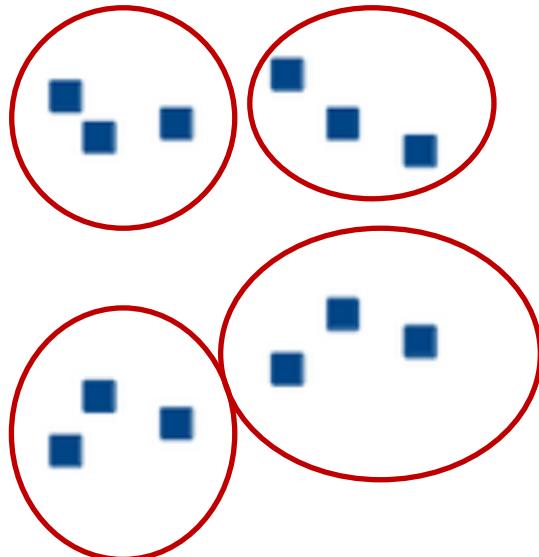
Number of clusters?



Number of clusters?



Number of clusters?



K-means

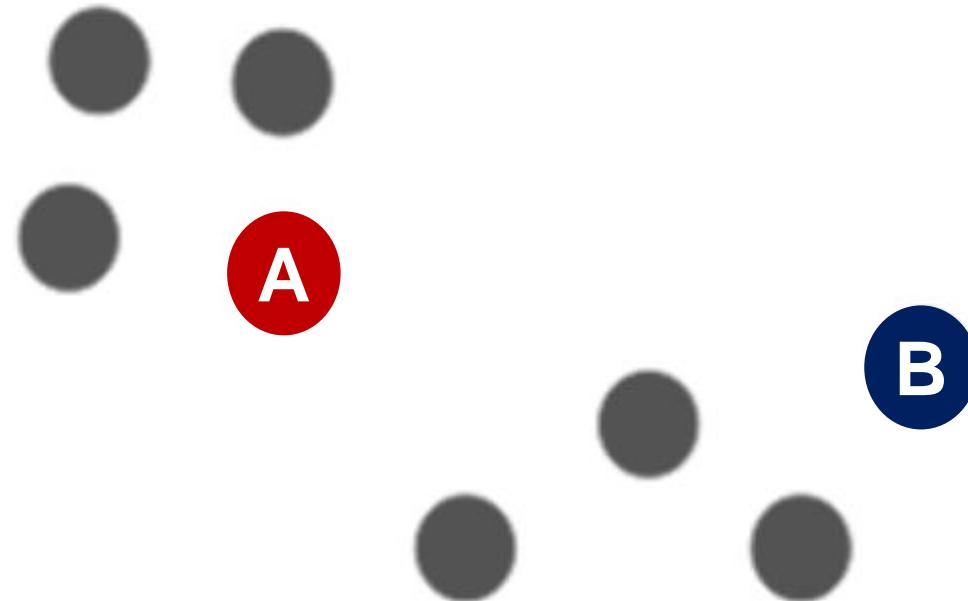
- ▶ Assume that there are K clusters.



Step 1:
Choose number of clusters
2 clusters

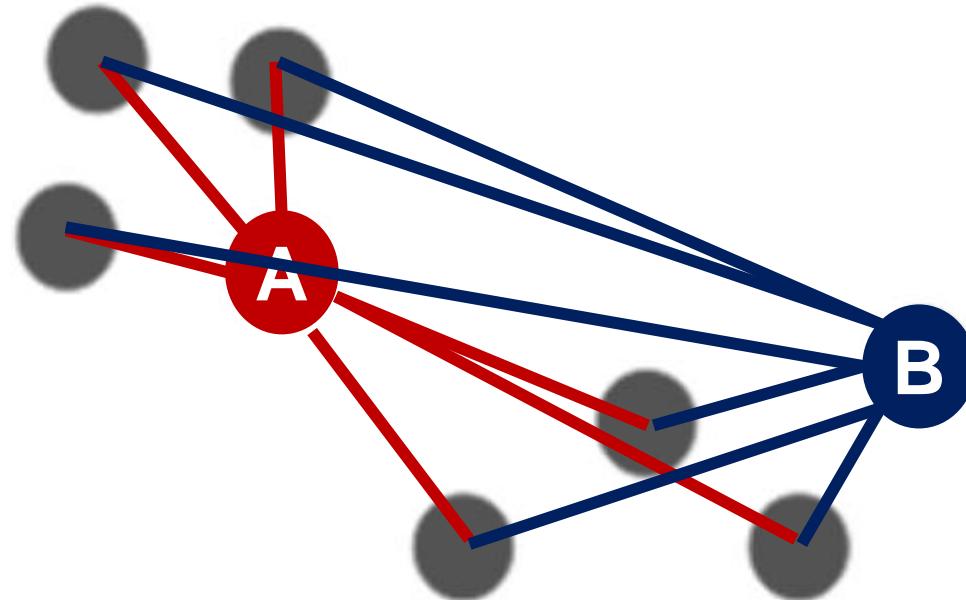
K-means

- ▶ Assume that there are K clusters.



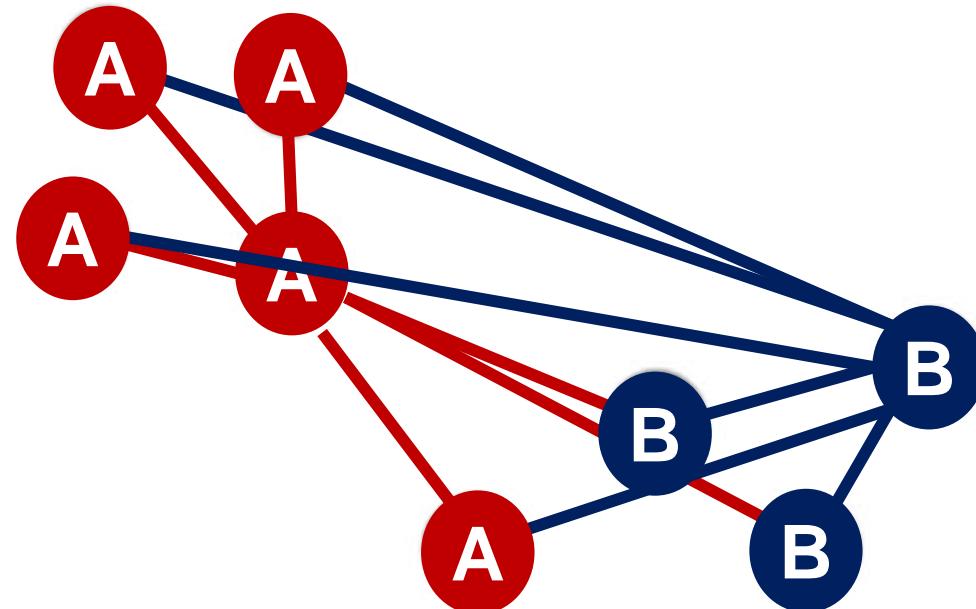
Step 2:
**Select k random
points as centroids**

K-means



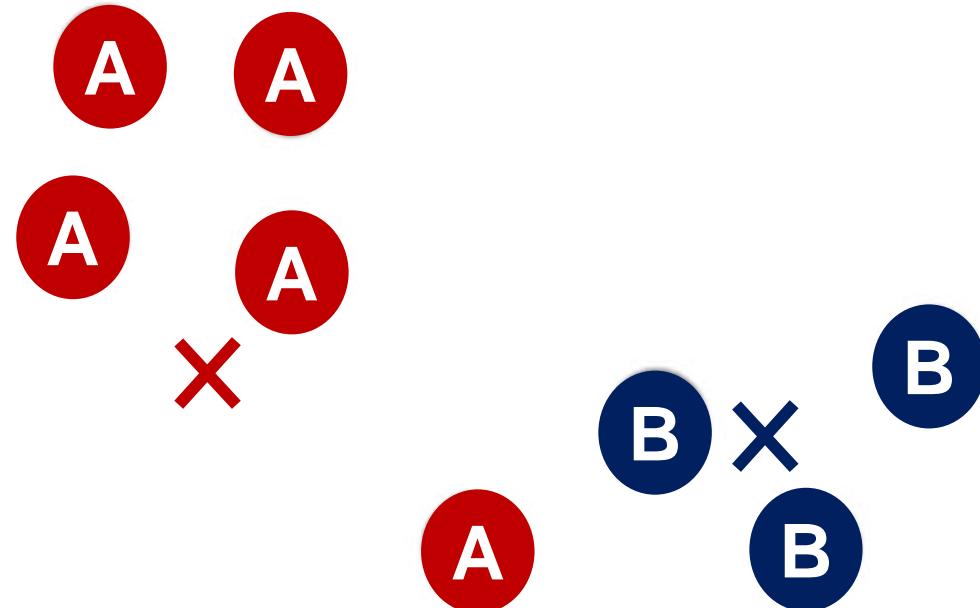
Step 3:
Estimate the distance
between each centroid
and each other point
in the dataset

K-means



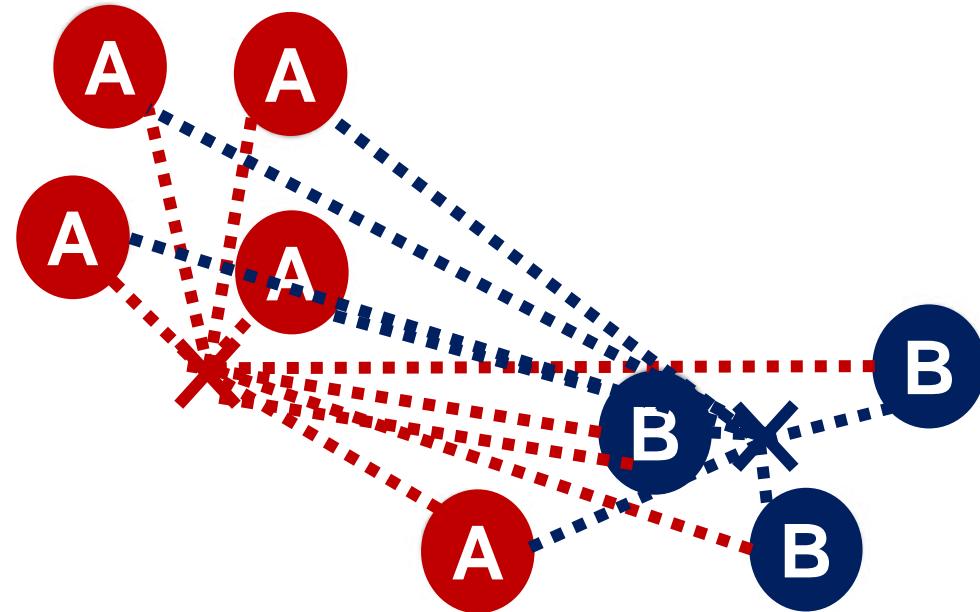
Step 4:
Assign all the points
to the closest cluster

K-means



Step 5:
Recompute the
centroids of newly
formed clusters &
Repeat steps 3,4

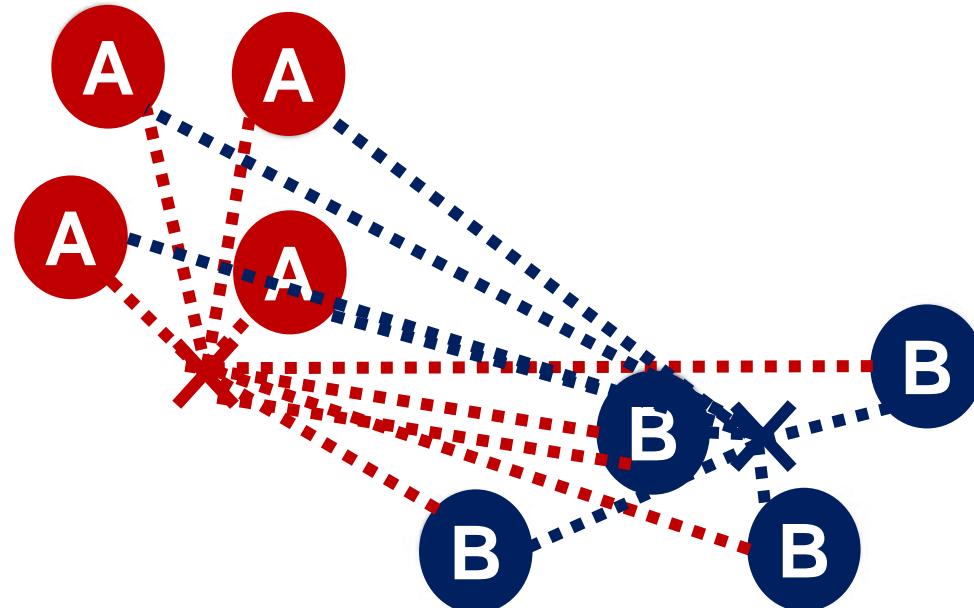
K-means



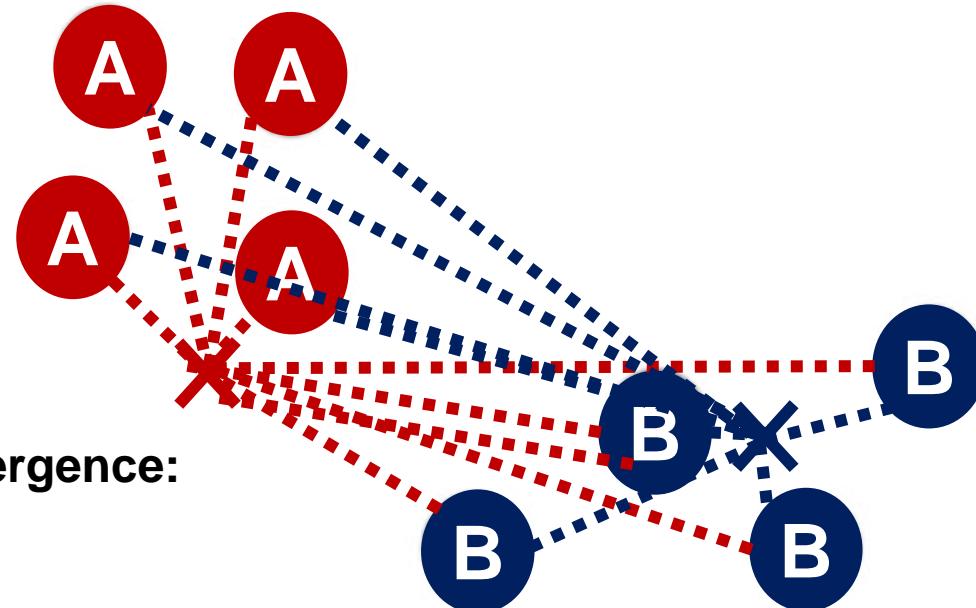
Repeat step 3:
Estimate the distance
between each centroid
and each other point
in the dataset

K-means

Repeat step 4:
Assign all the points
to the closest cluster



K-means



Repeat process up to convergence:

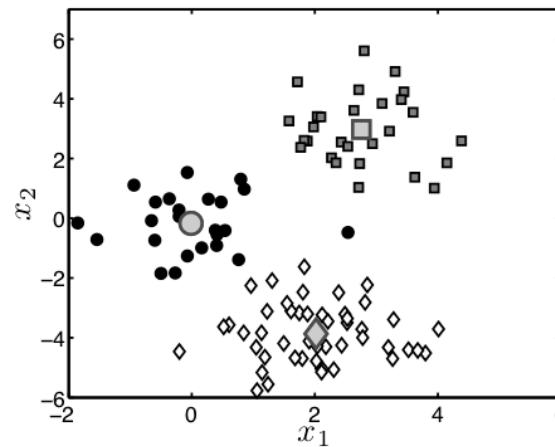
- **No changes in group memberships**
- **Maximum number of iterations have reached**

K-means

- ▶ Assume that there are K clusters.
- ▶ Each cluster is defined by a position in the input space:

$$\boldsymbol{\mu}_k = [\mu_{k1}, \mu_{k2}]^\top$$

- ▶ Each \mathbf{x}_n is assigned to its closest cluster:



- ▶ Distance is normally Euclidean distance:

$$d_{nk} = (\mathbf{x}_n - \boldsymbol{\mu}_k)^\top (\mathbf{x}_n - \boldsymbol{\mu}_k)$$

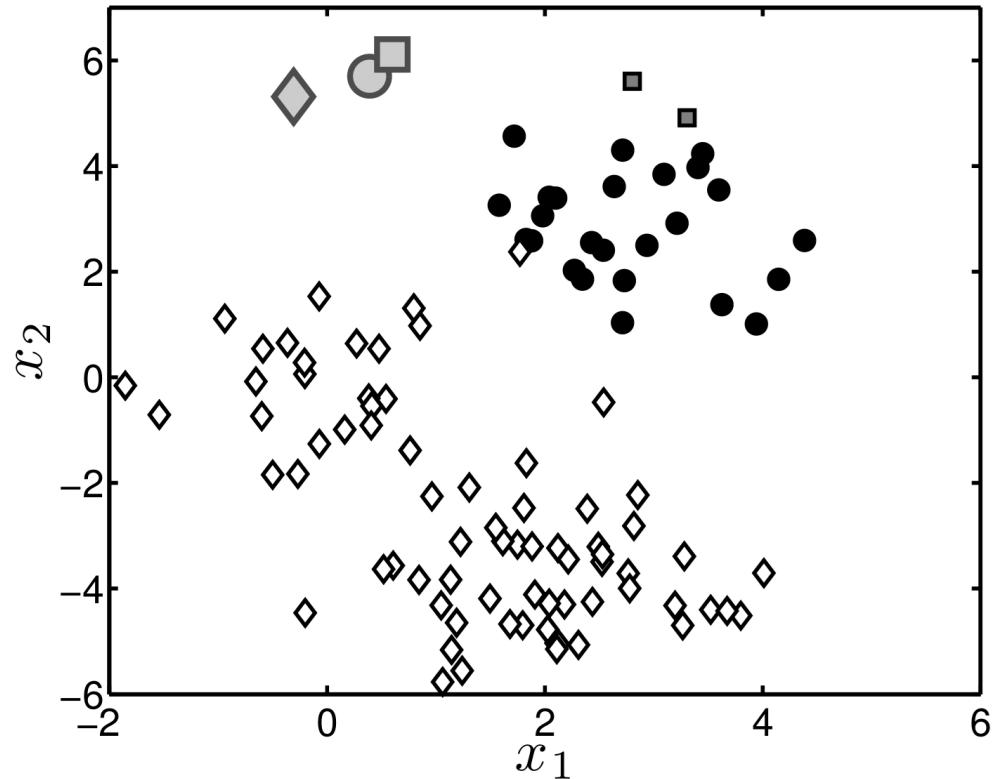
How do we find the cluster means

- ▶ No analytical solution – we can't write down μ_k as a function of \mathbf{X} .
- ▶ Use an iterative algorithm:
 1. Guess $\mu_1, \mu_2, \dots, \mu_K$
 2. Assign each \mathbf{x}_n to its closest μ_k
 3. $z_{nk} = 1$ if \mathbf{x}_n assigned to μ_k (0 otherwise)
 4. Update μ_k to average of \mathbf{x}_n s assigned to μ_k :

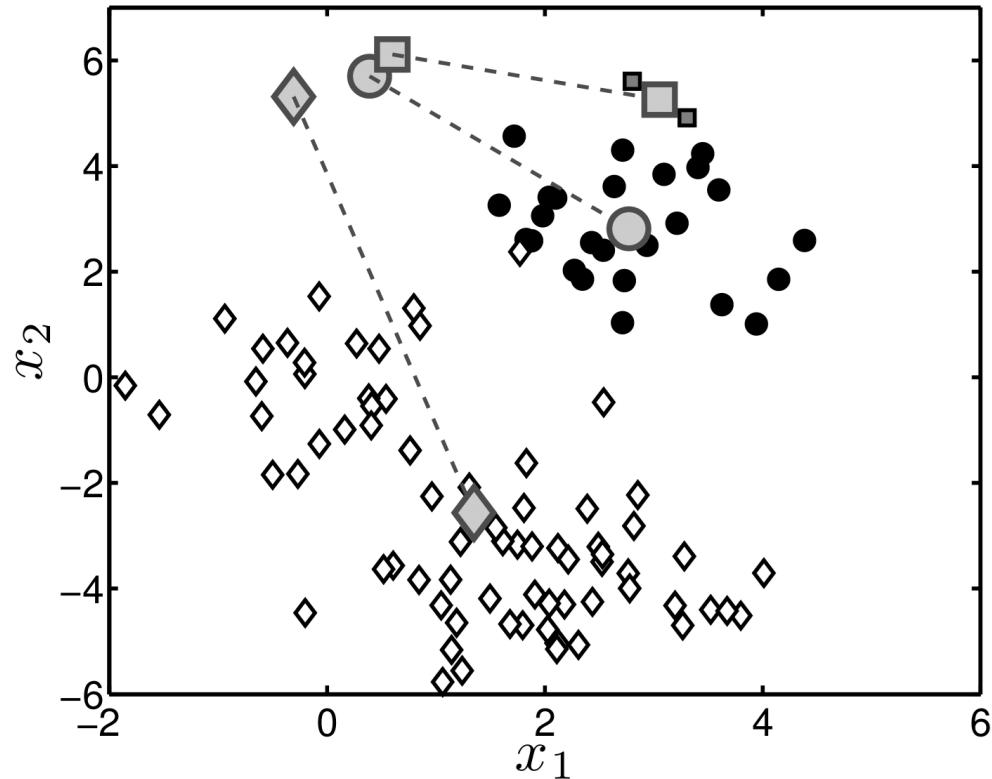
$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^N z_{nk} \mathbf{x}_n}{\sum_{n=1}^N z_{nk}}$$

- 5. Return to 2 until assignments do not change.
- ▶ Algorithm will converge....it will reach a point where the assignments don't change.

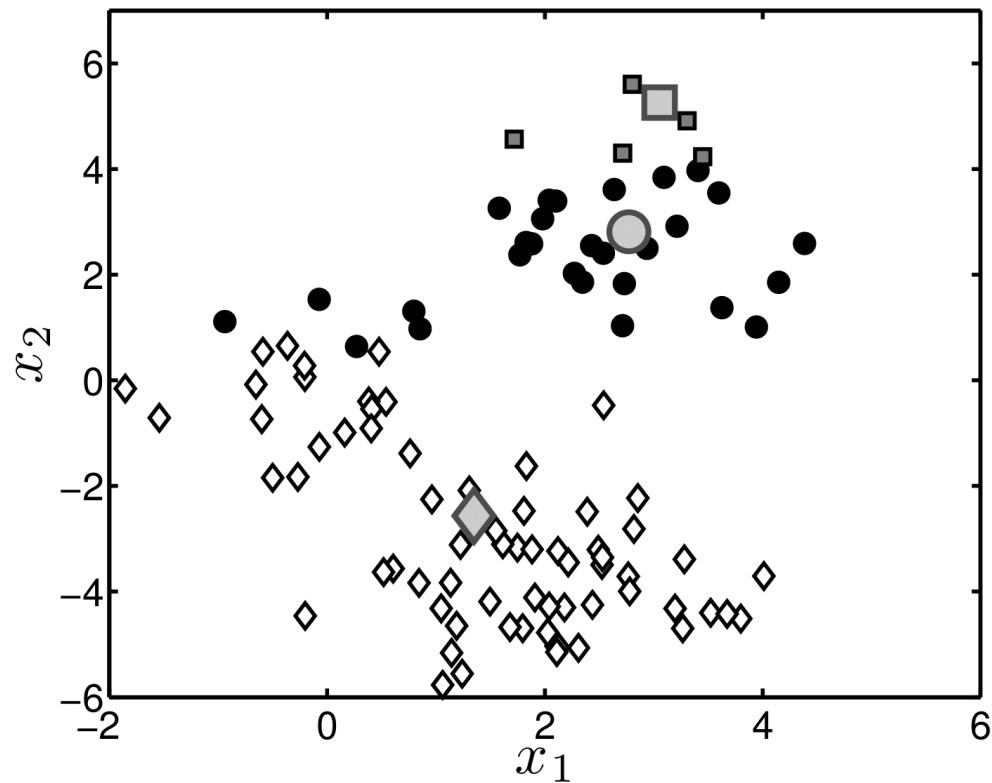
K-means - example



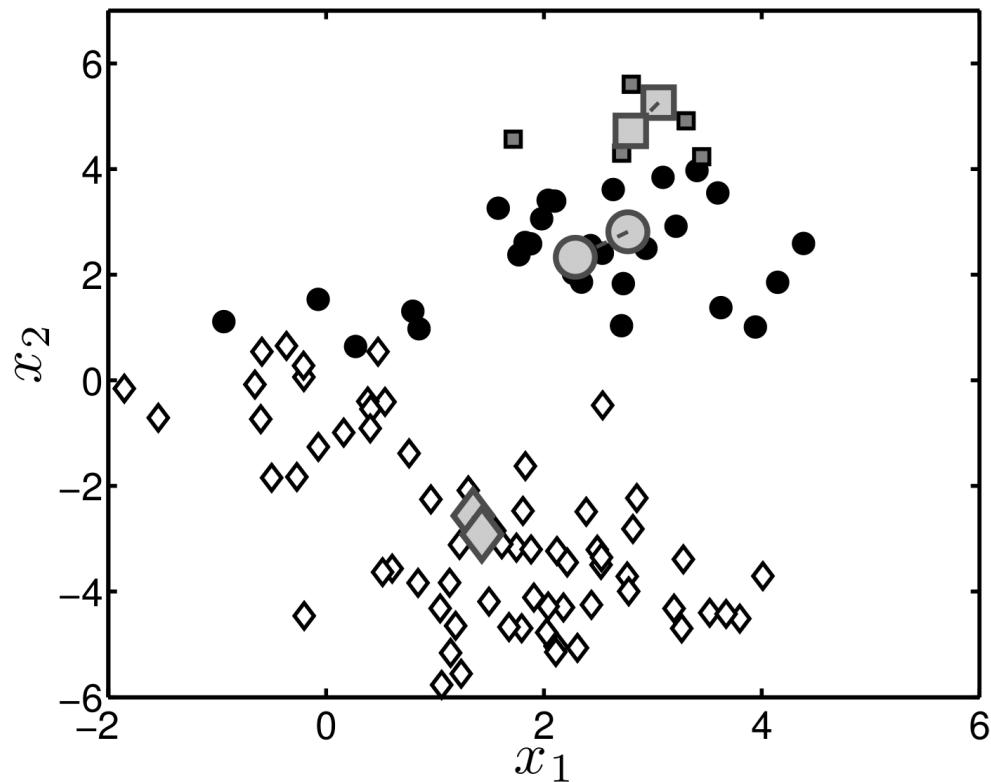
K-means - example



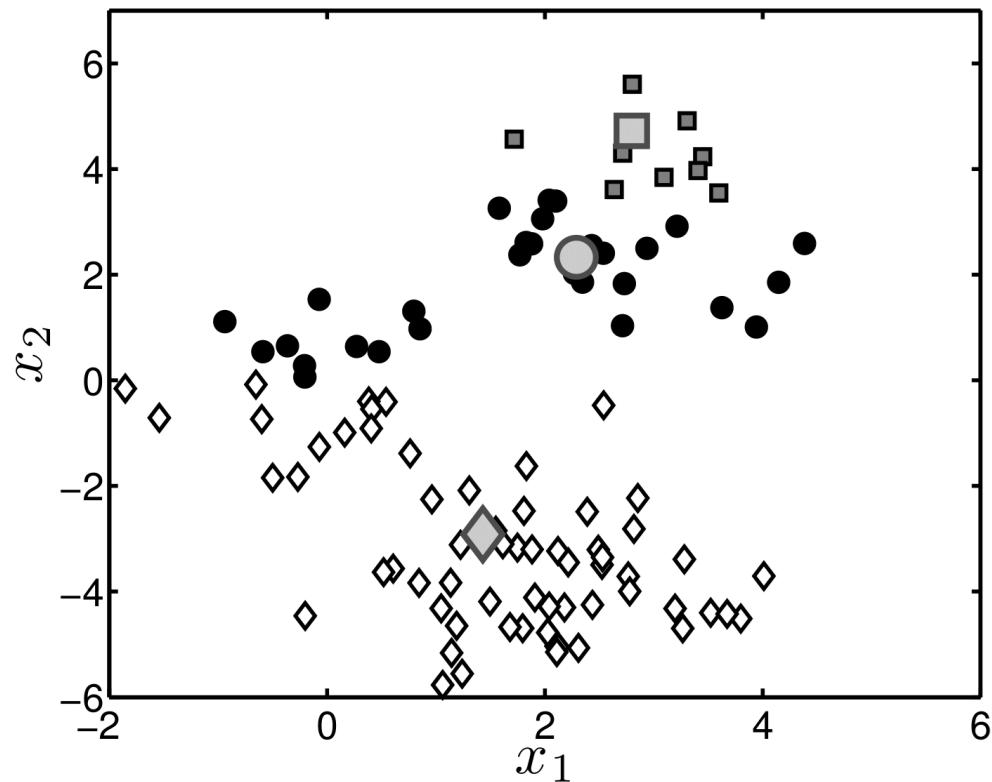
K-means - example



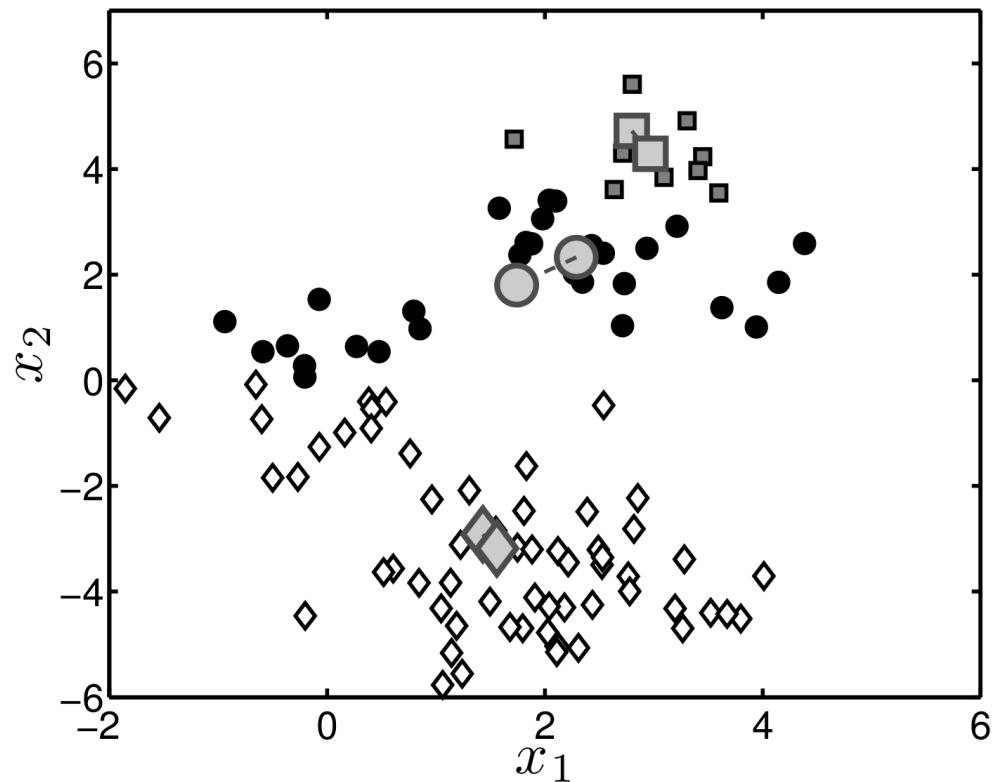
K-means - example



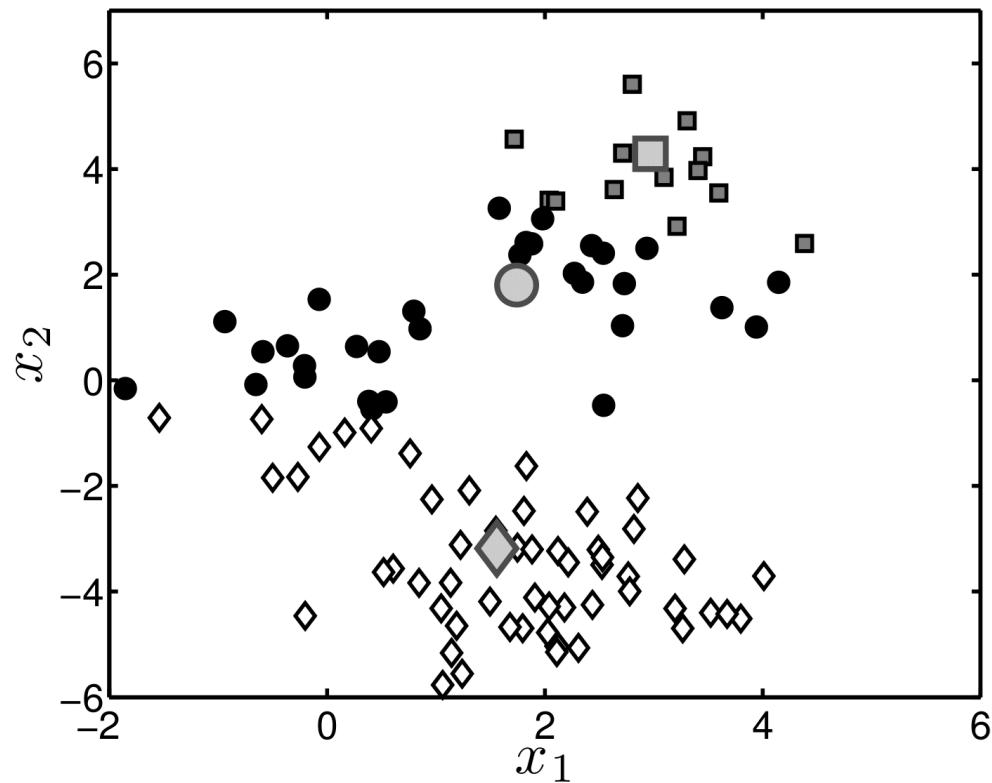
K-means - example



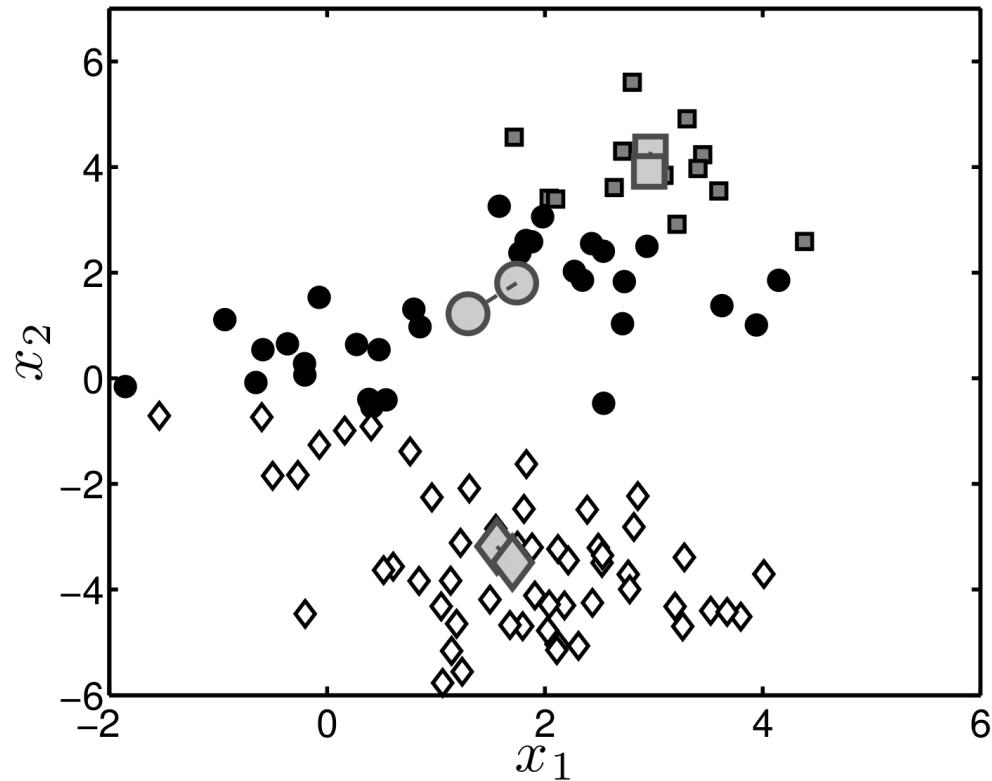
K-means - example



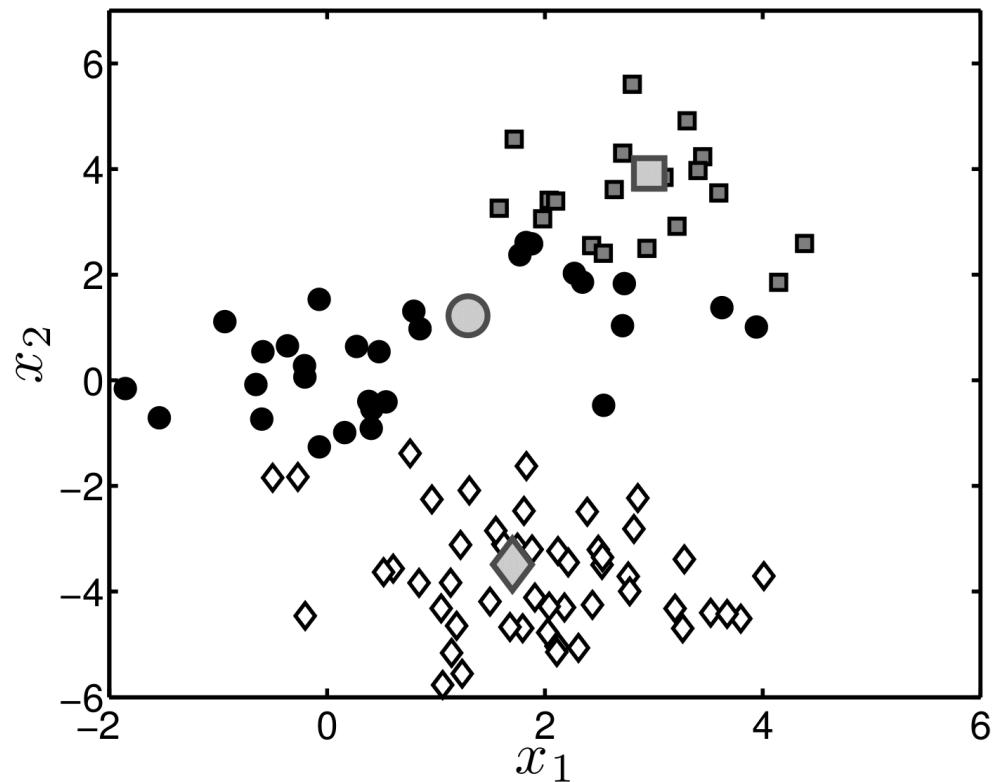
K-means - example



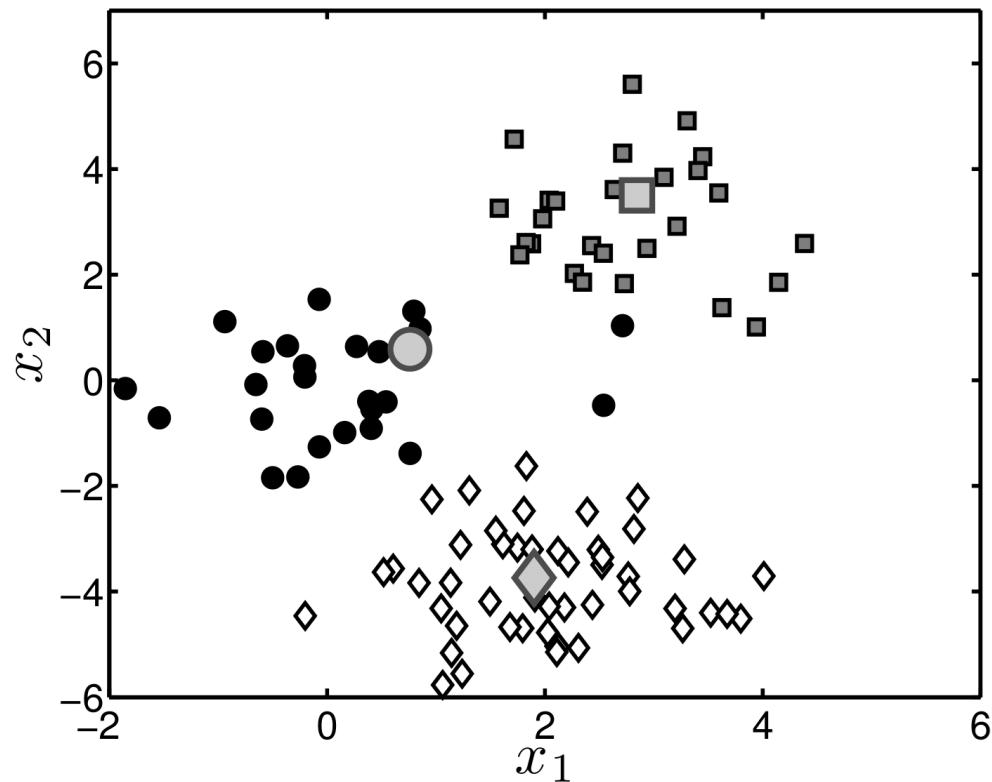
K-means - example



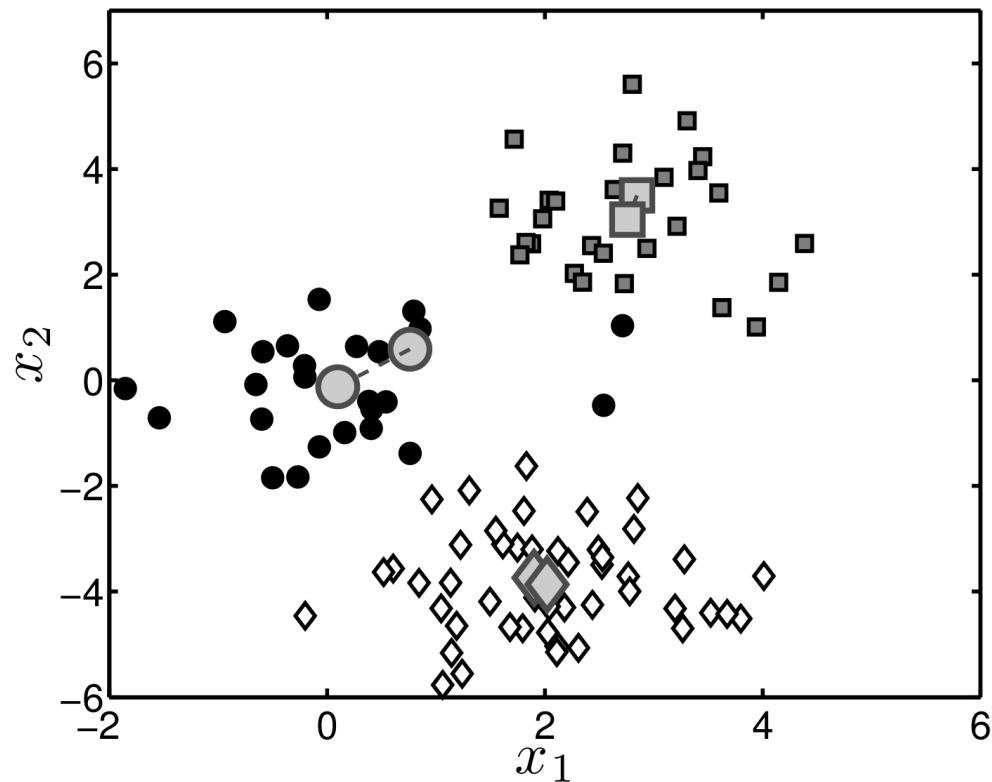
K-means - example



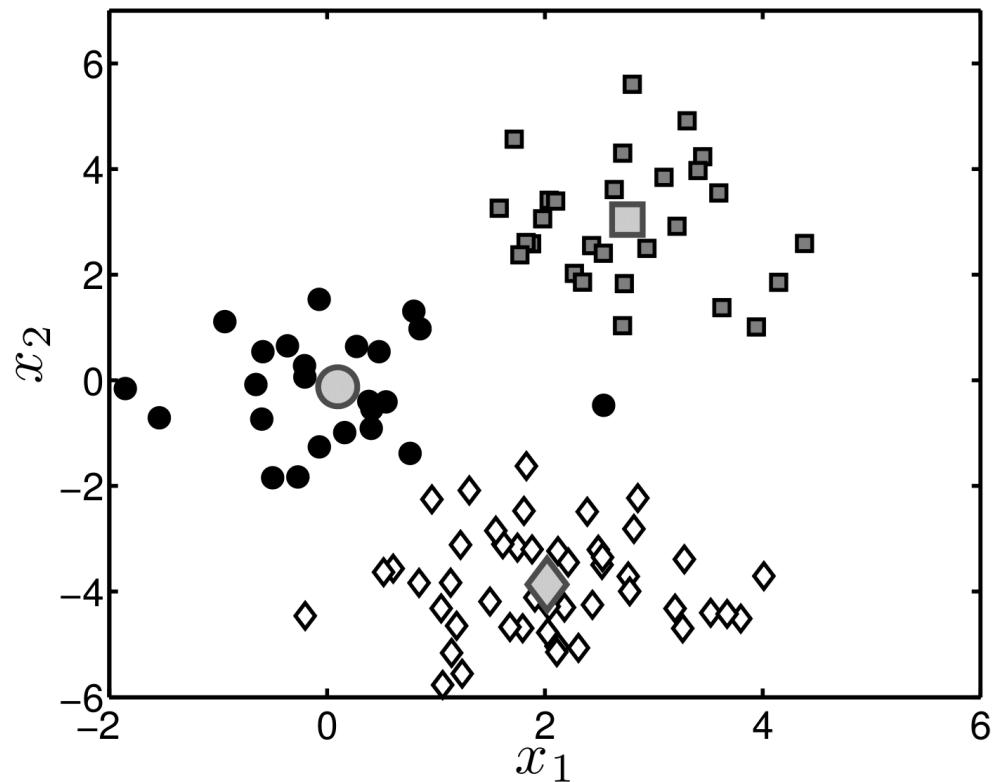
K-means - example



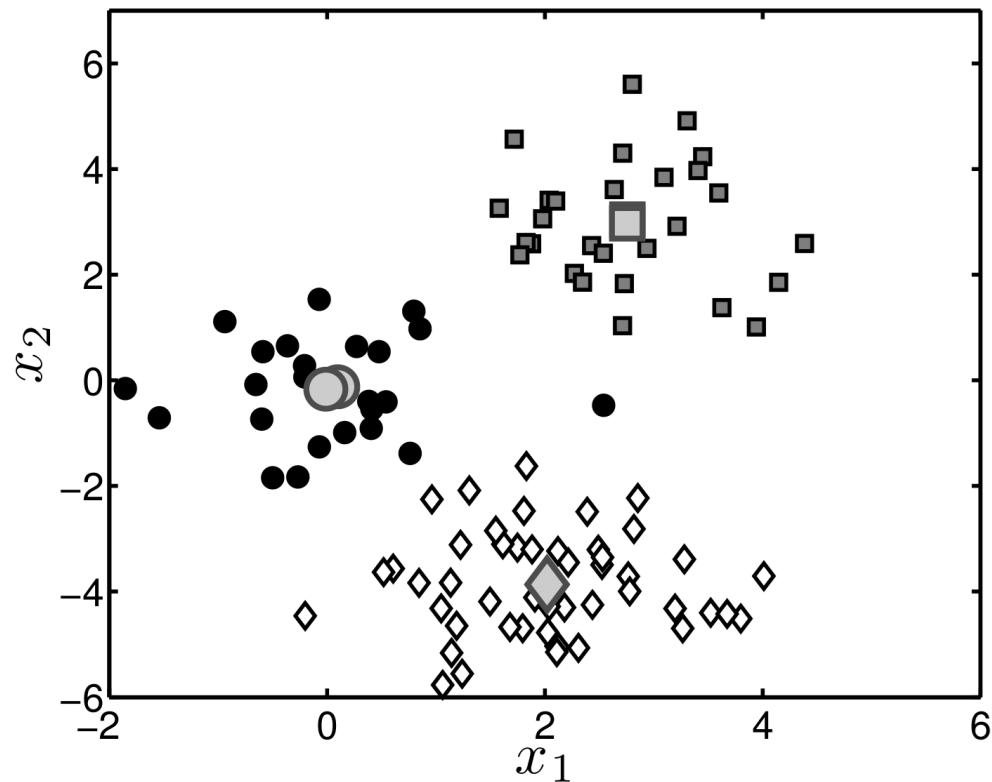
K-means - example



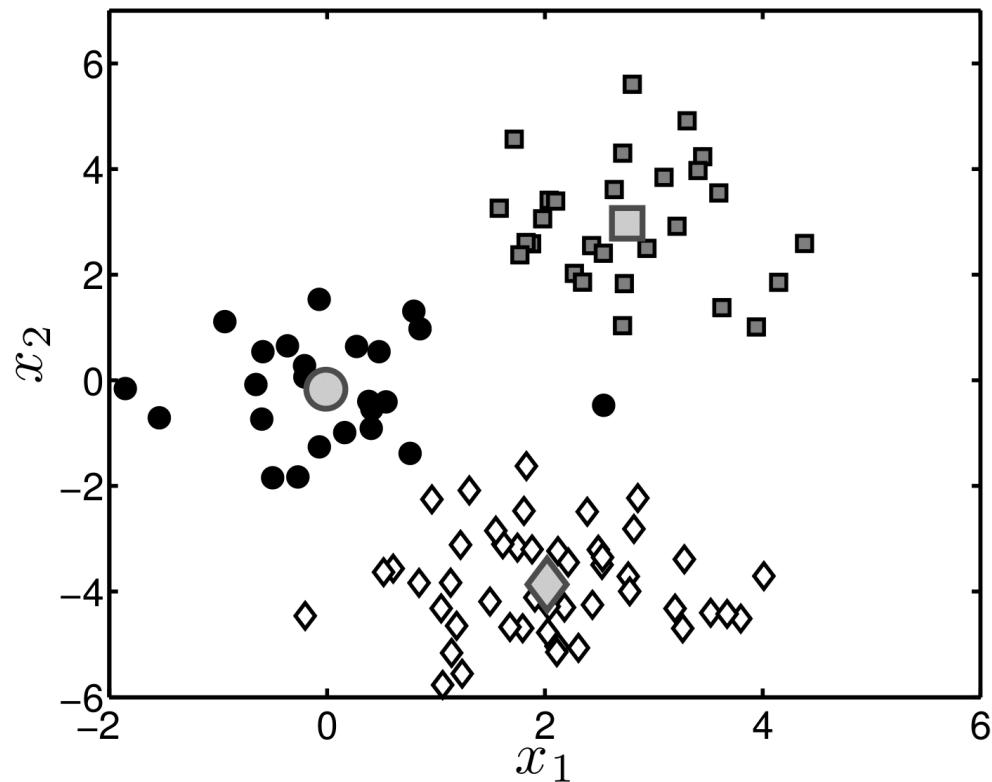
K-means - example



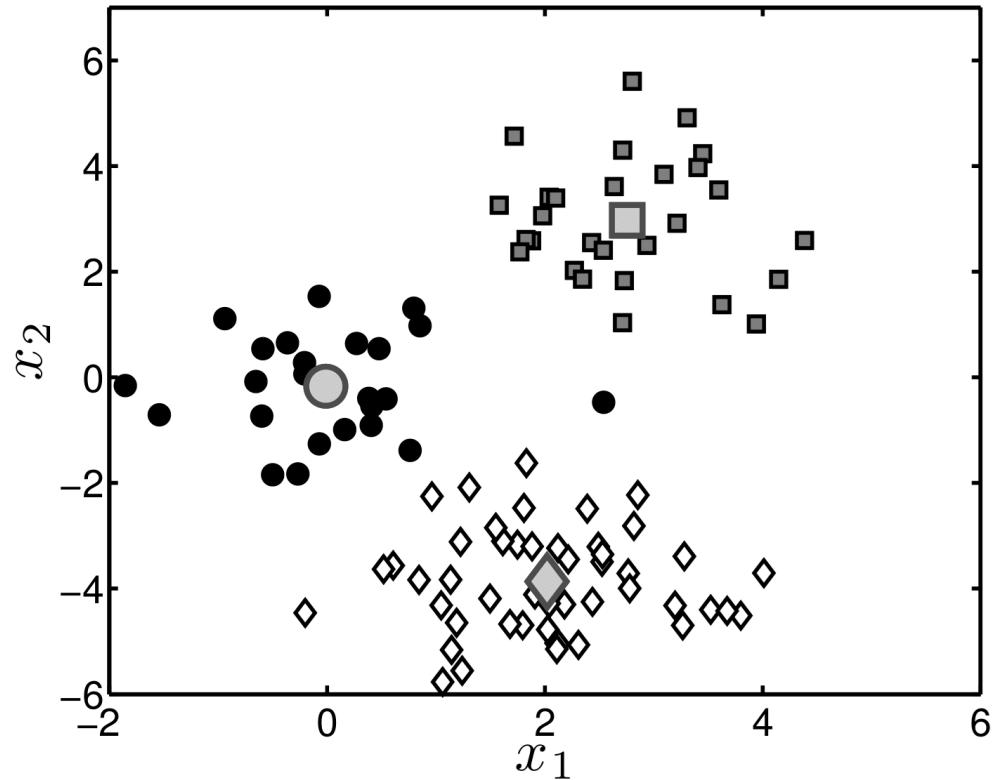
K-means - example



K-means - example

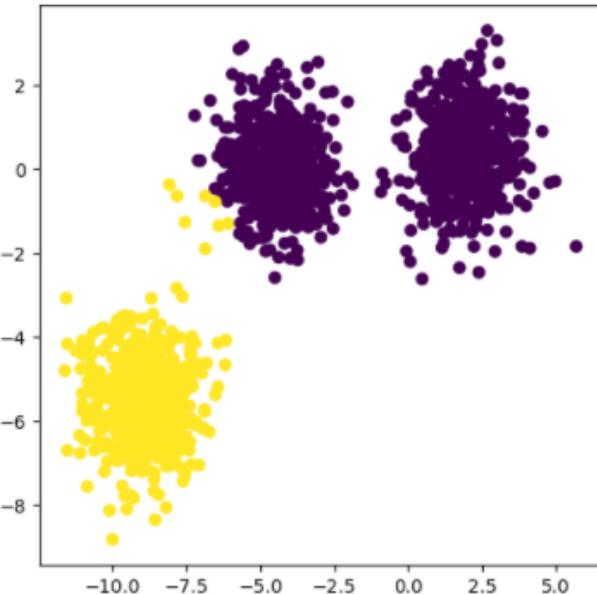


K-means - example - converged



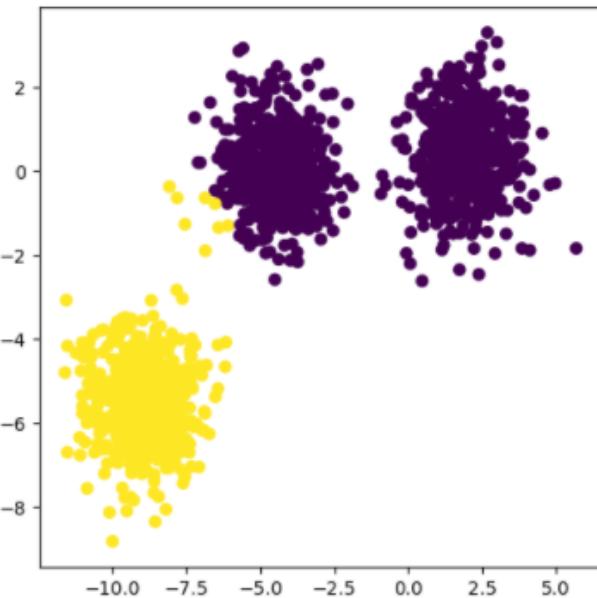
When does K-means break?

Incorrect Number of Blobs

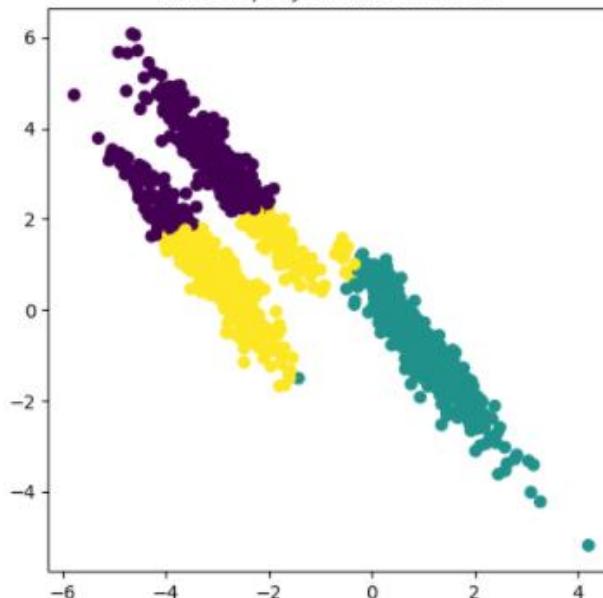


When does K-means break?

Incorrect Number of Blobs

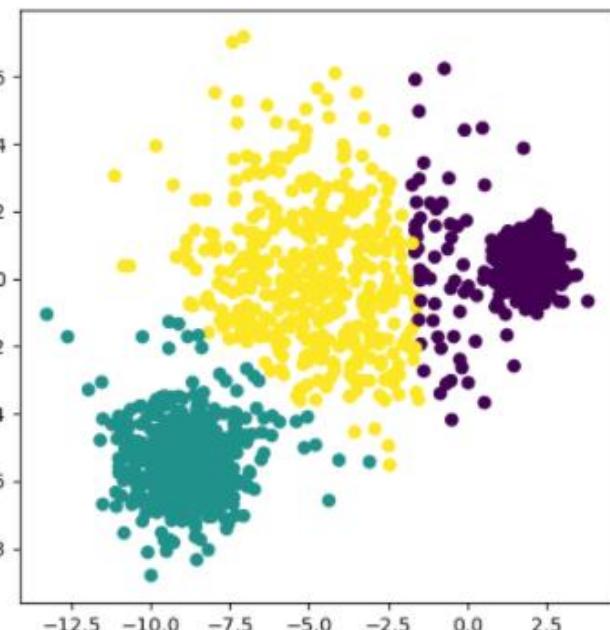
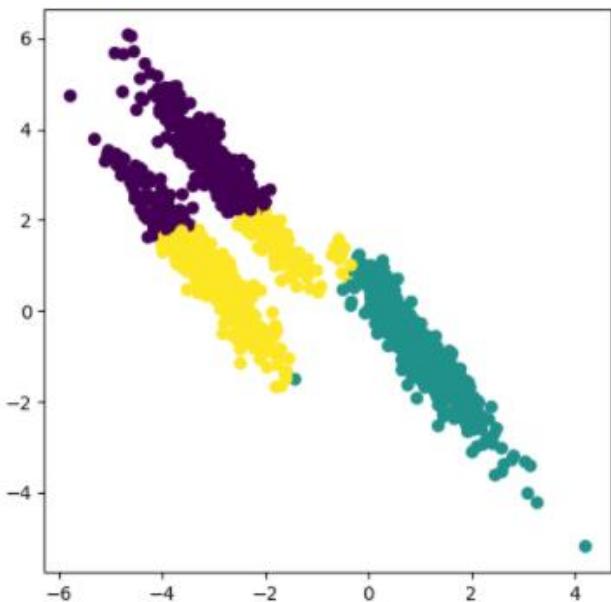
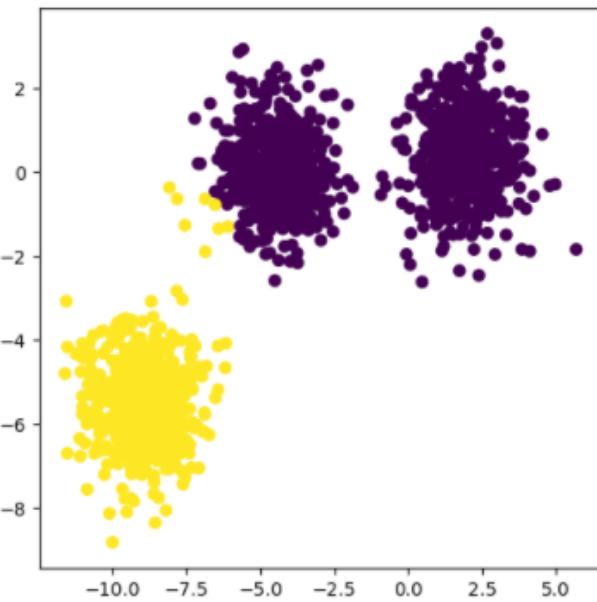


Anisotropically Distributed Blobs

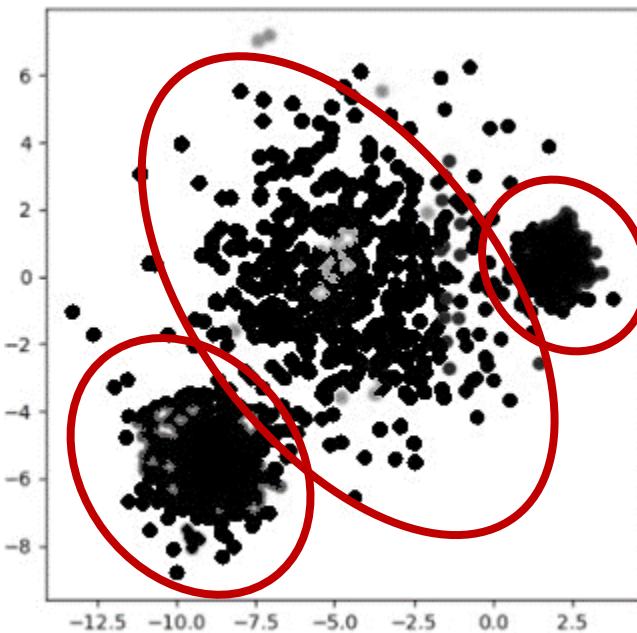
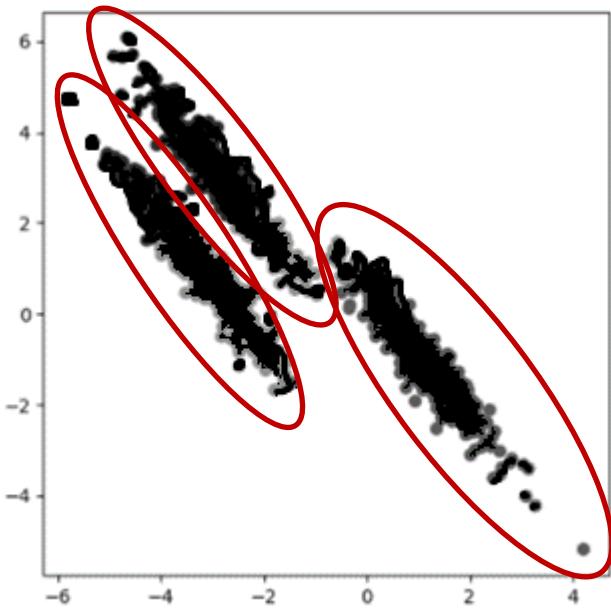


When does K-means break?

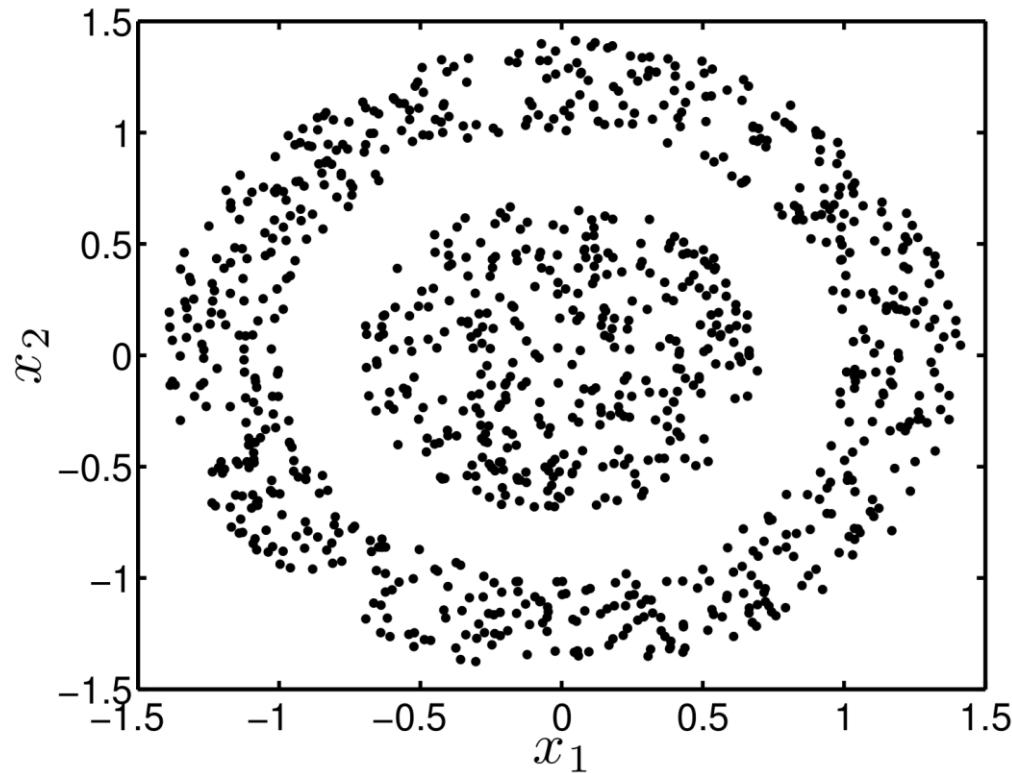
Incorrect Number of Blobs



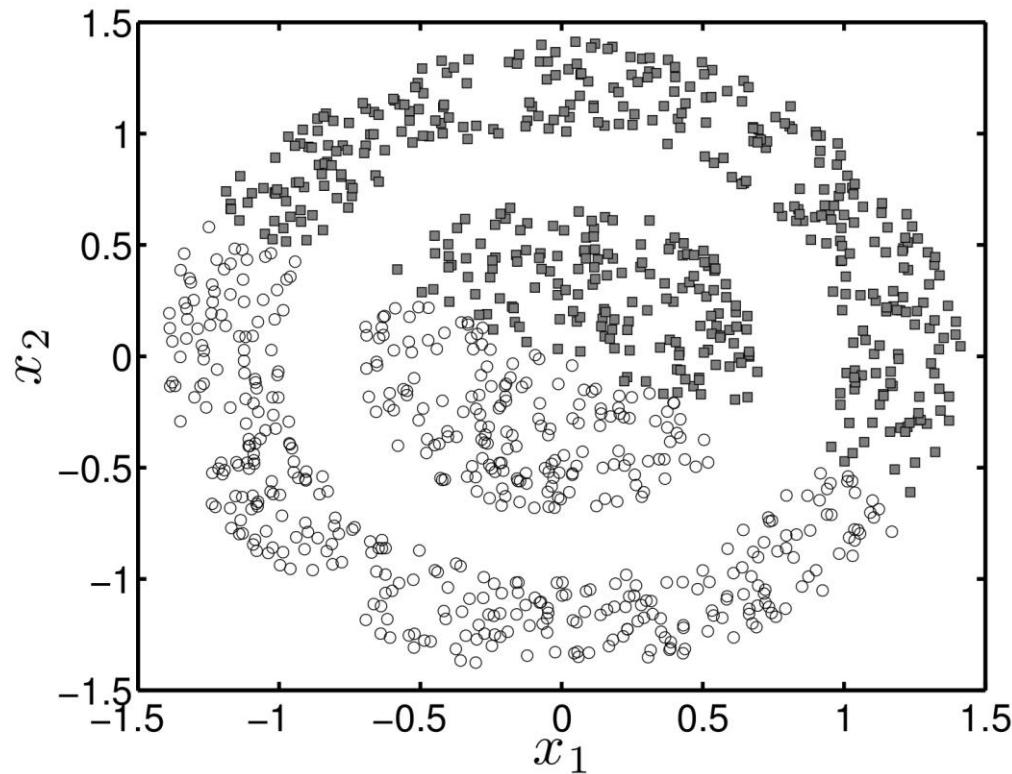
— — —



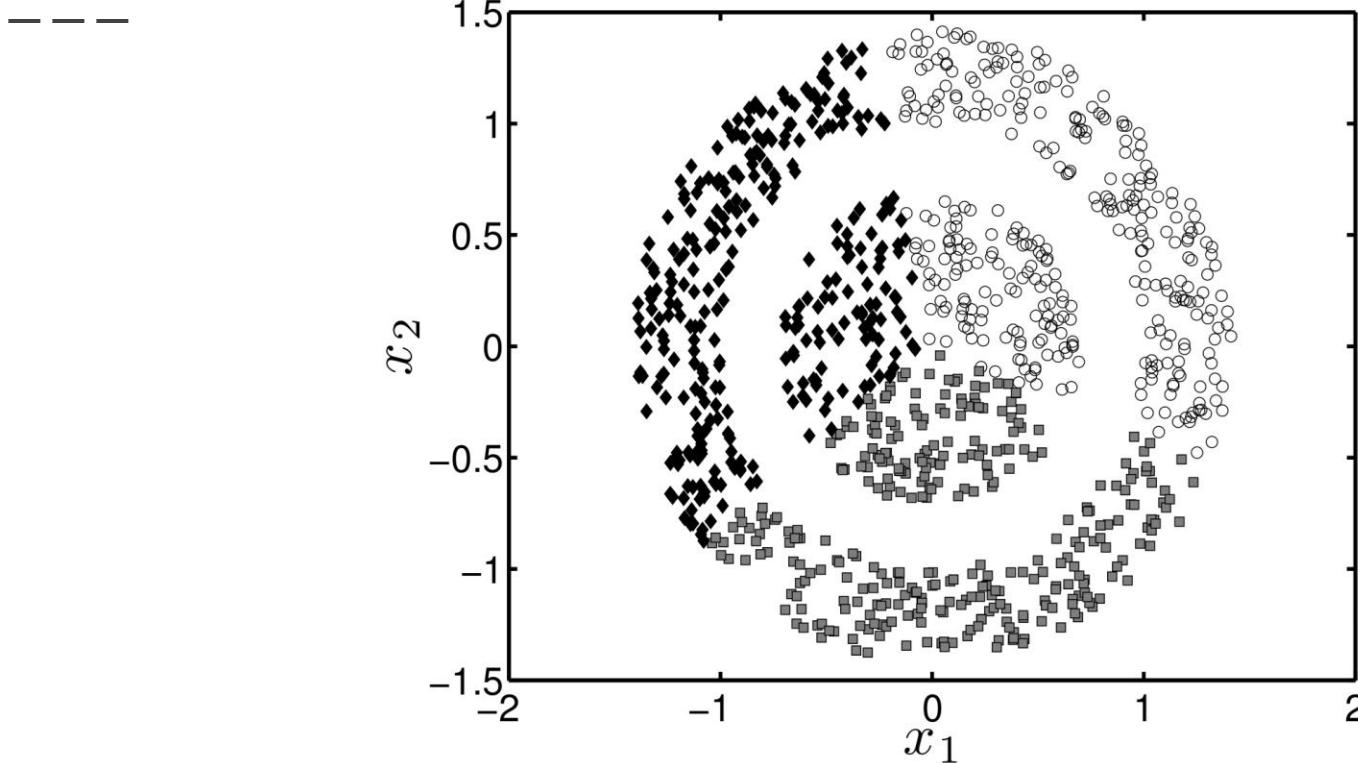
When does K-means break?



K-means with 2 clusters

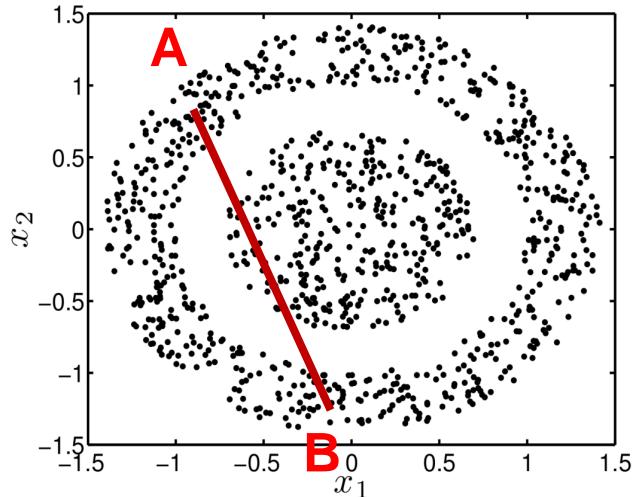


K-means with 3 clusters



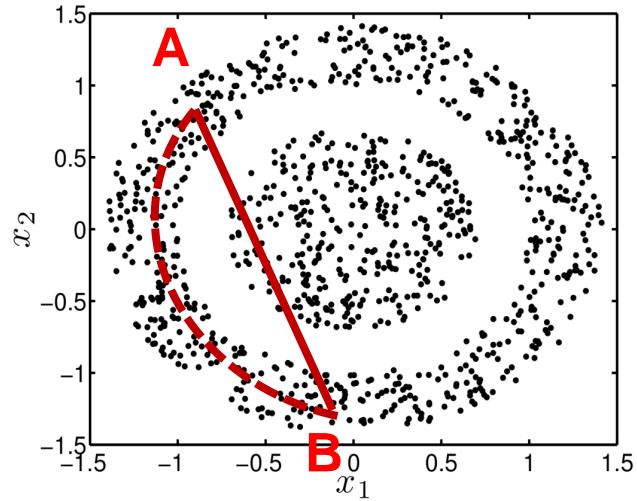
K-means – What is the problem?

- Euclidean distance is not appropriate for non-linearly separable clusters



K-means – What is the problem?

- Euclidean distance is not always appropriate for non-linearly separable clusters



Kernelising K-means

$$d_{nk} = (\mathbf{x}_n - \boldsymbol{\mu}_k)^\top (\mathbf{x}_n - \boldsymbol{\mu}_k) \quad \boldsymbol{\mu}_k = \frac{\sum_{n=1}^N z_{nk} \mathbf{x}_n}{\sum_{n=1}^N z_{nk}}$$

- ▶ Multiply out:

$$\mathbf{x}_n^\top \mathbf{x}_n - 2N_k^{-1} \sum_{m=1}^N z_{mk} \mathbf{x}_m^\top \mathbf{x}_n + N_k^{-2} \sum_{m,l} z_{mk} z_{lk} \mathbf{x}_m^\top \mathbf{x}_l$$

- ▶ Kernel substitution:

$$k(\mathbf{x}_n, \mathbf{x}_n) - 2N_k^{-1} \sum_{m=1}^N z_{mk} k(\mathbf{x}_n, \mathbf{x}_m) + N_k^{-2} \sum_{m,l=1}^N z_{mk} z_{lk} k(\mathbf{x}_m, \mathbf{x}_l)$$

Kernel K-means

- ▶ Algorithm:
 1. Choose a kernel and any necessary parameters.
 2. Start with random assignments z_{nk} .
 3. For each \mathbf{x}_n , assign it to the nearest 'center' where distance is defined as:
$$k(\mathbf{x}_n, \mathbf{x}_n) - 2N_k^{-1} \sum_{m=1}^N z_{mk} k(\mathbf{x}_n, \mathbf{x}_m) + N_k^{-2} \sum_{m,l=1}^N z_{mk} z_{lk} k(\mathbf{x}_m, \mathbf{x}_l)$$
 4. If assignments have changed, return to 3.
- ▶ Note – no μ_k . This would be $N_k^{-1} \sum_n z_{nk} \phi(\mathbf{x}_n)$ but we don't know $\phi(\mathbf{x}_n)$ for kernels. We only know $\phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$ (last week)...

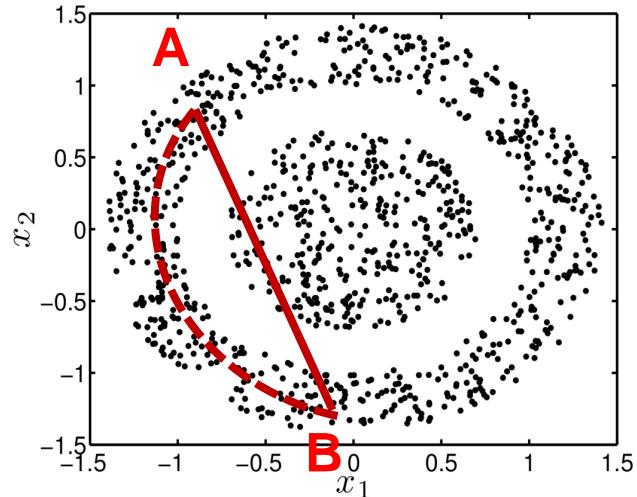
Kernels

- Polynomial:

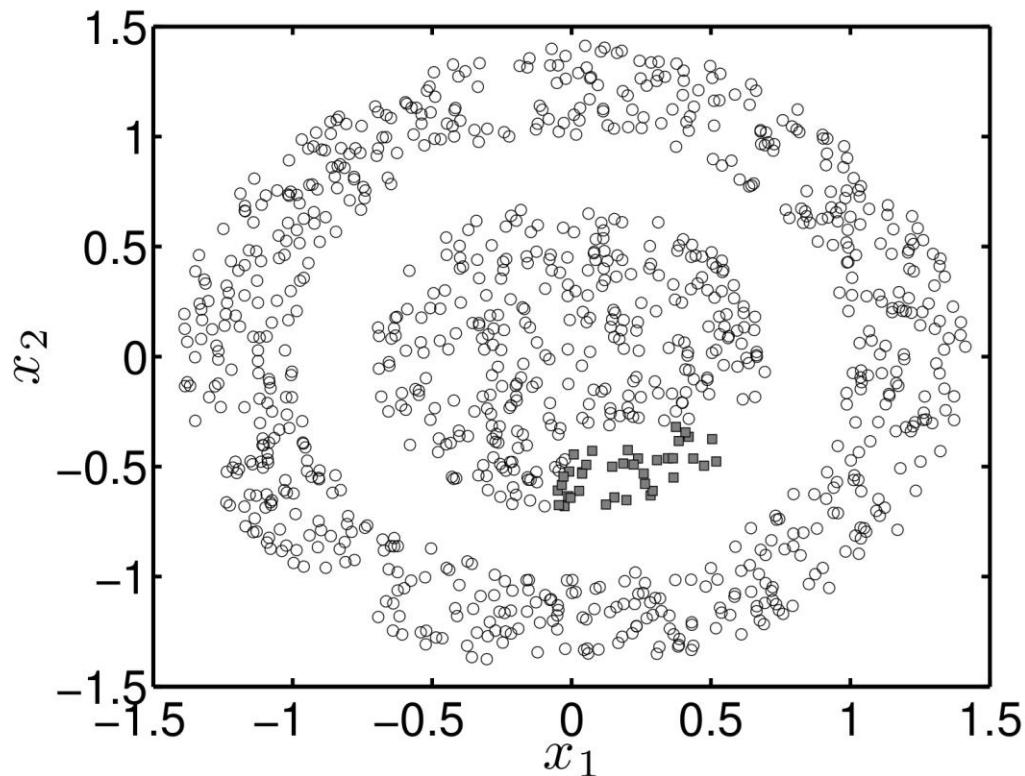
$$\kappa(x, y) = (x^T y)^p$$

- Gaussian:

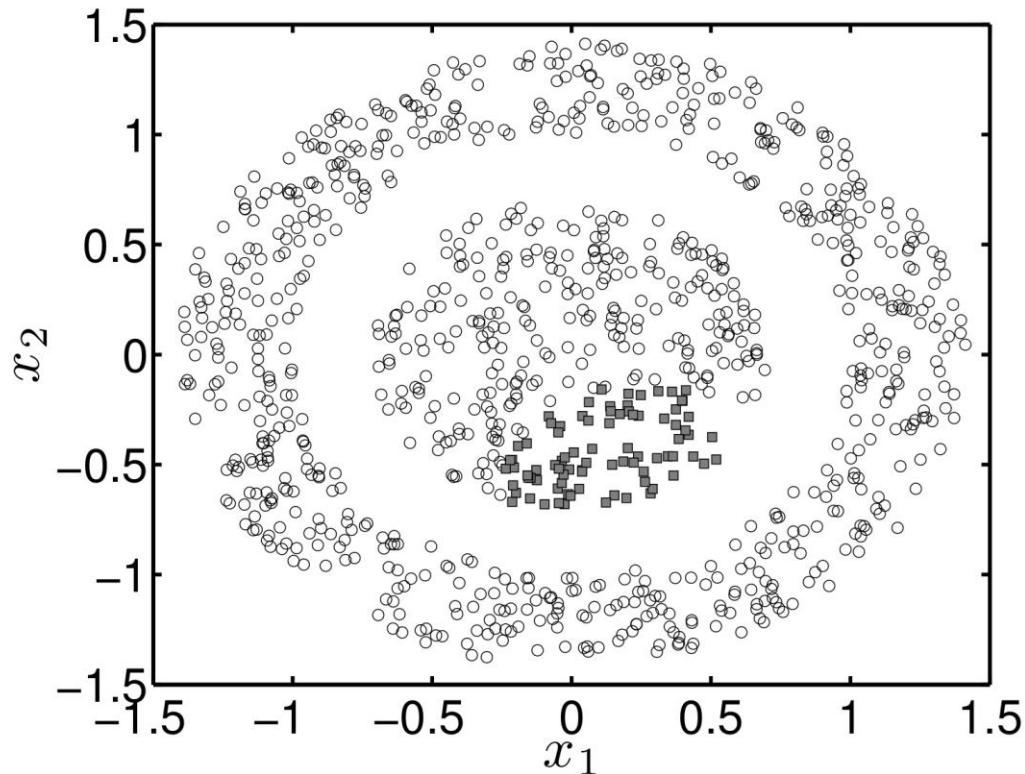
$$\kappa(x, y) = \exp(-\lambda \|x - y\|^2)$$



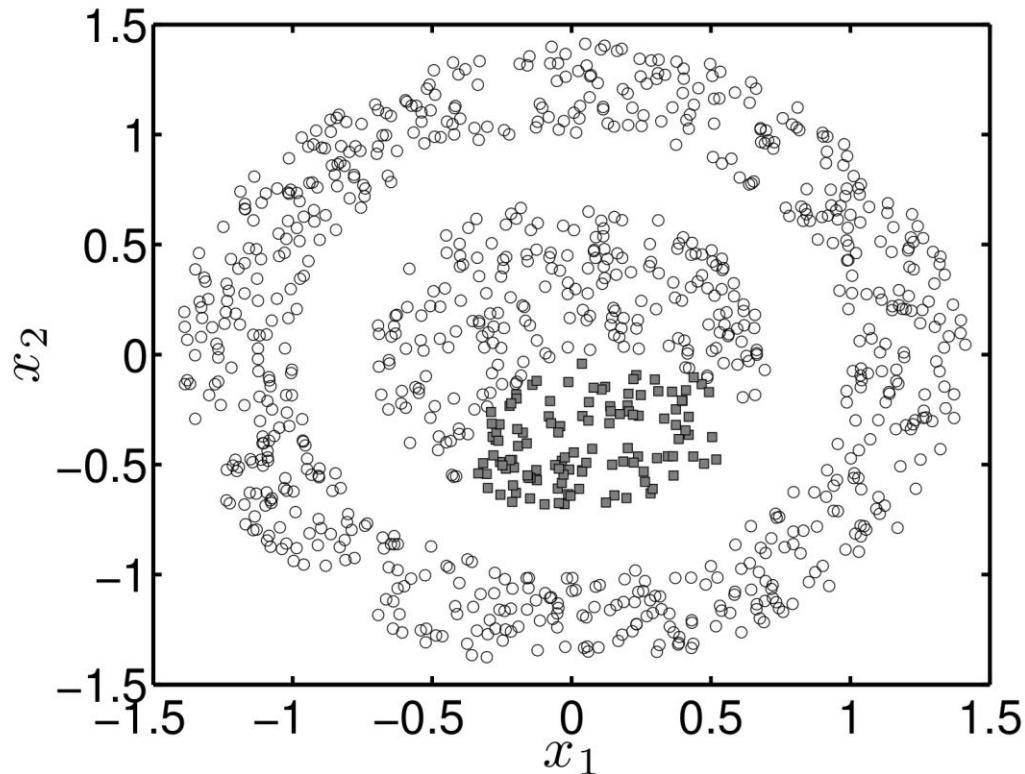
Kernel K-means - example



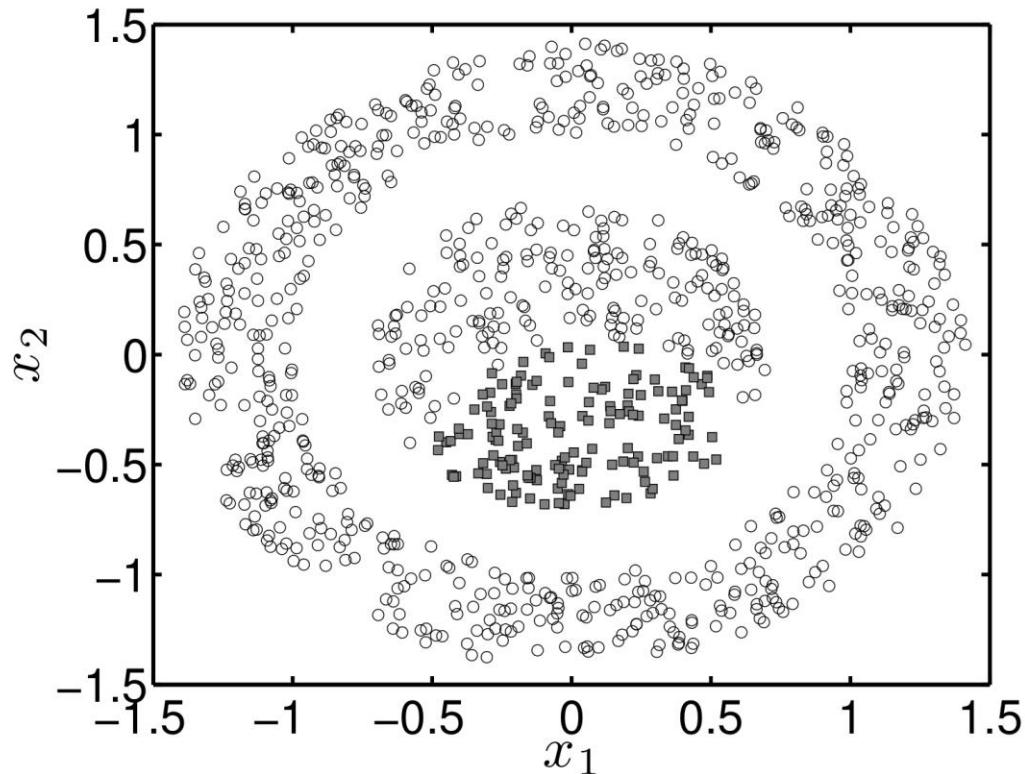
Kernel K-means - example



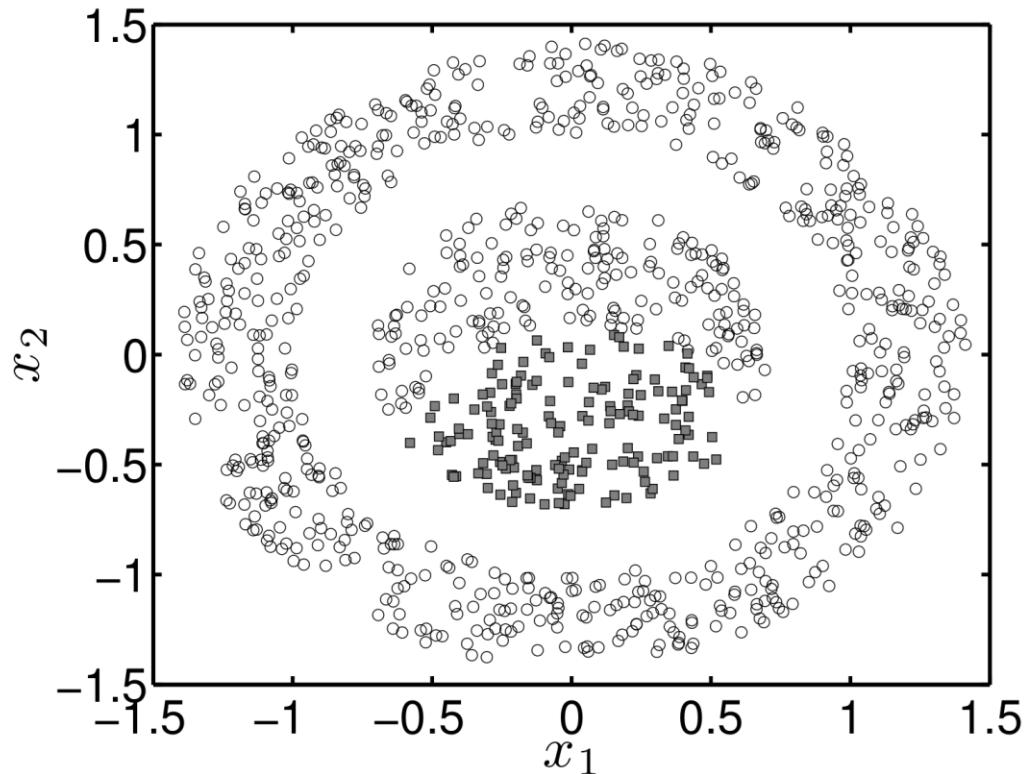
Kernel K-means - example



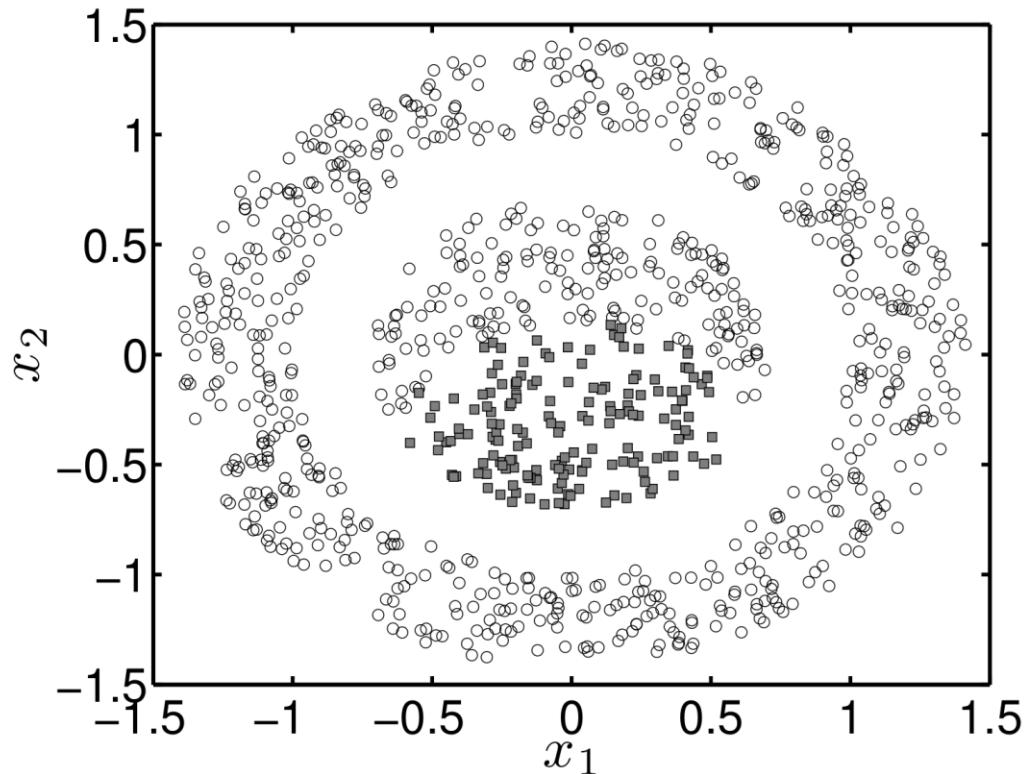
Kernel K-means - example



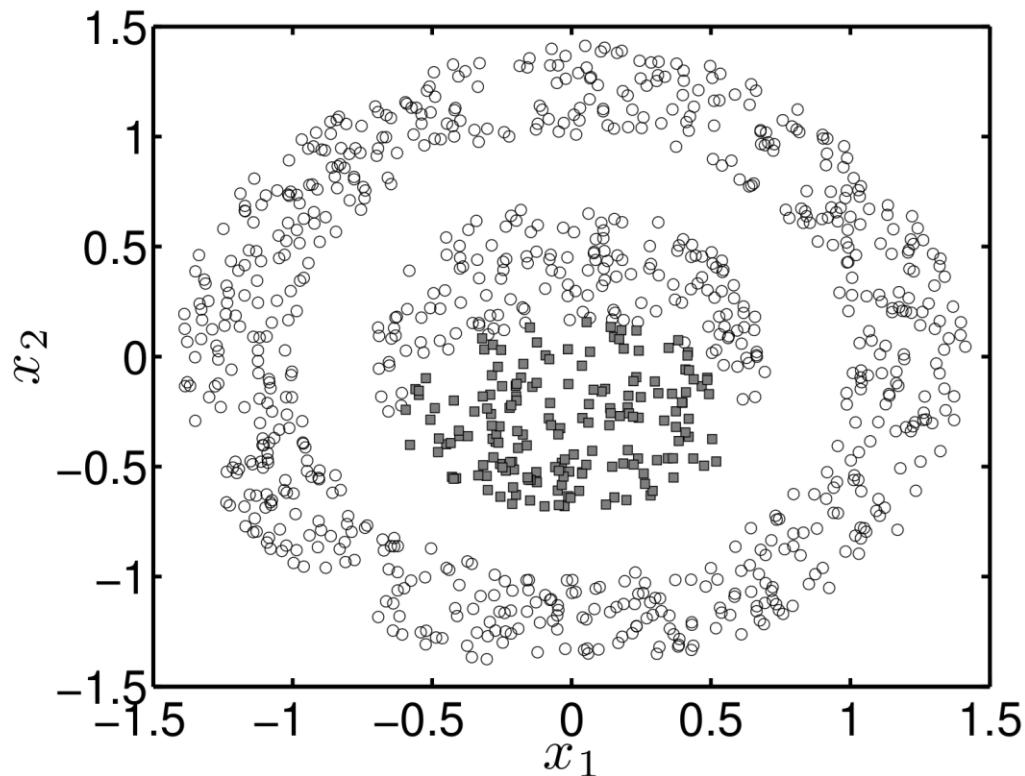
Kernel K-means - example



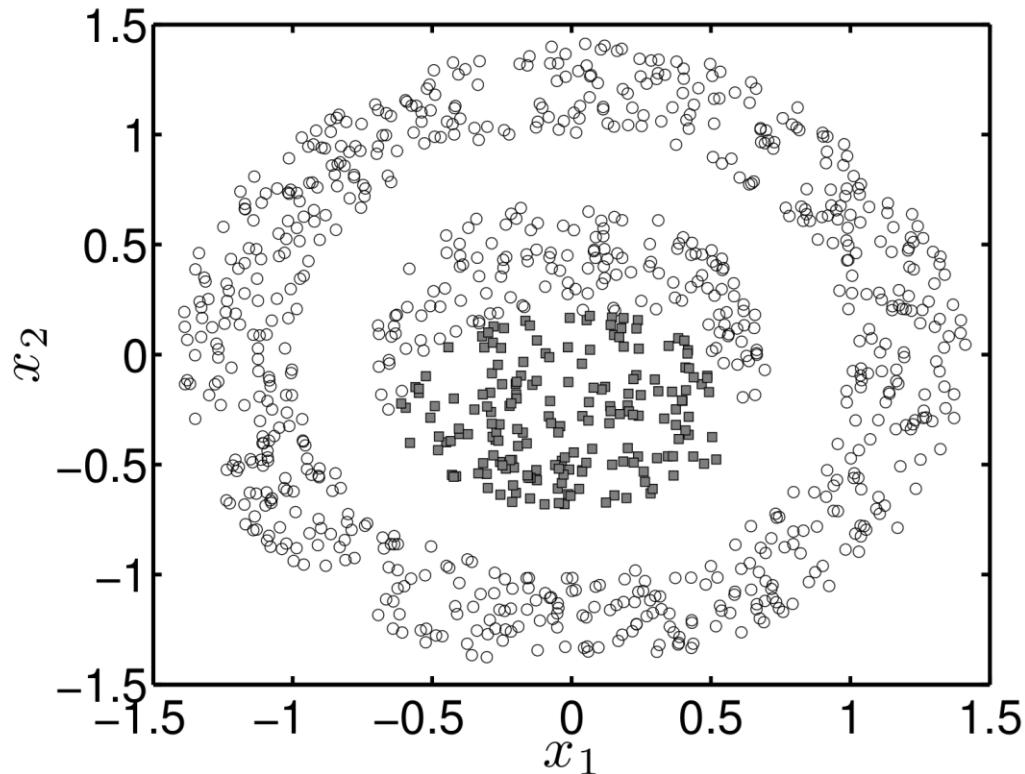
Kernel K-means - example



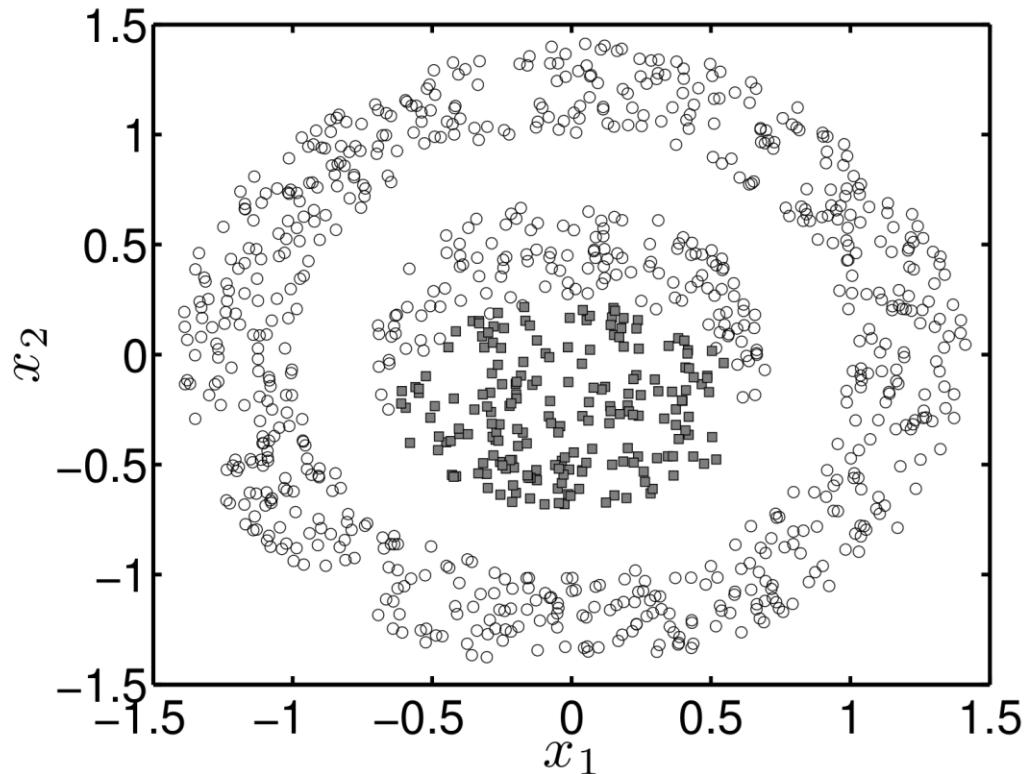
Kernel K-means - example



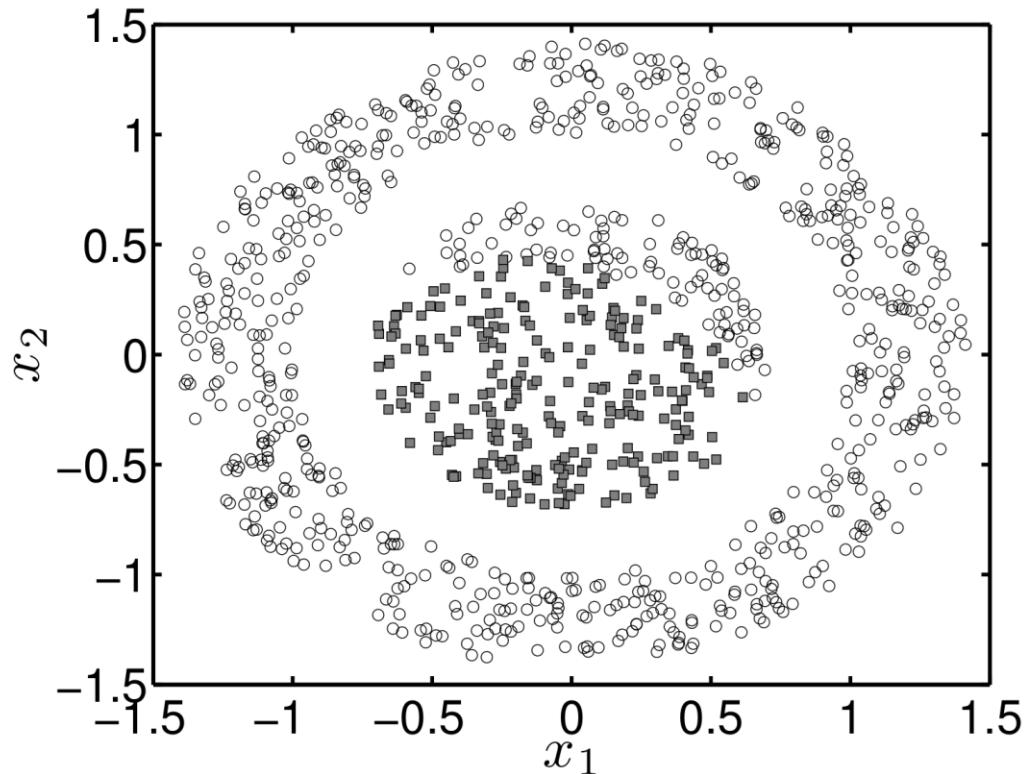
Kernel K-means - example



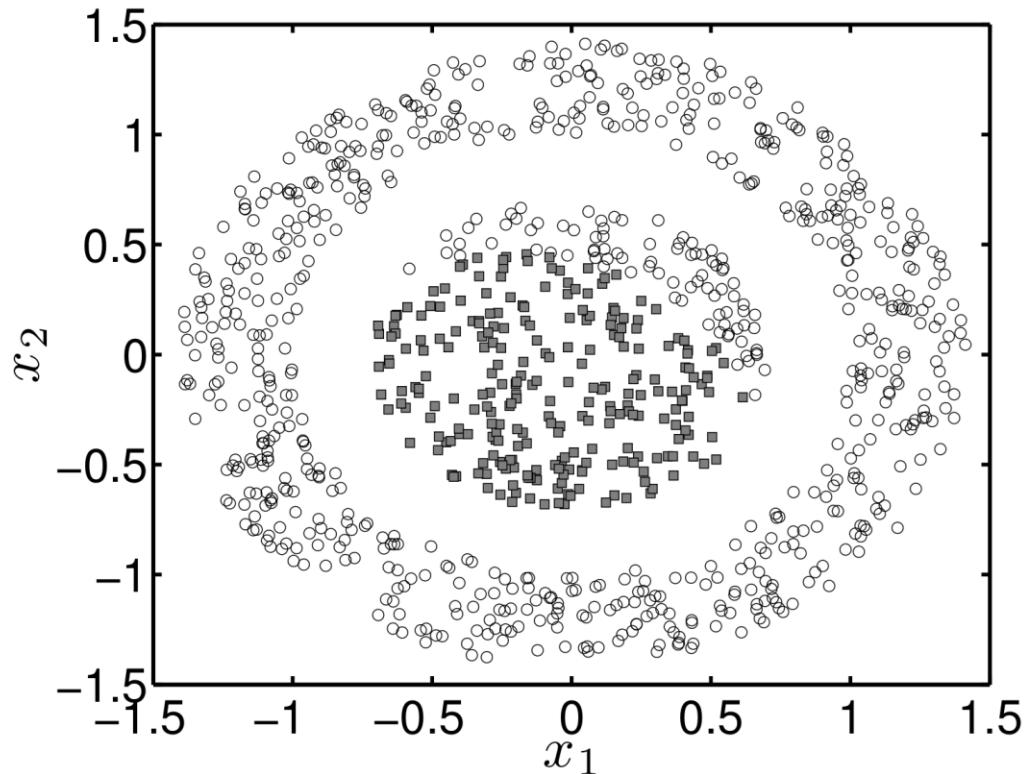
Kernel K-means - example



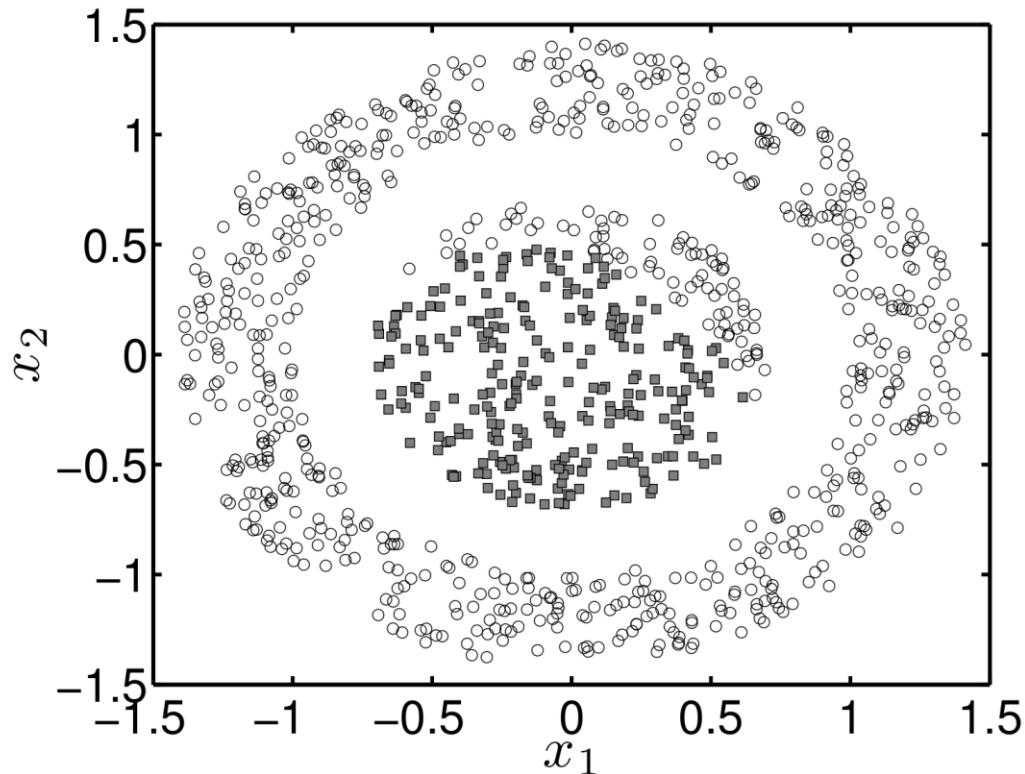
Kernel K-means - example



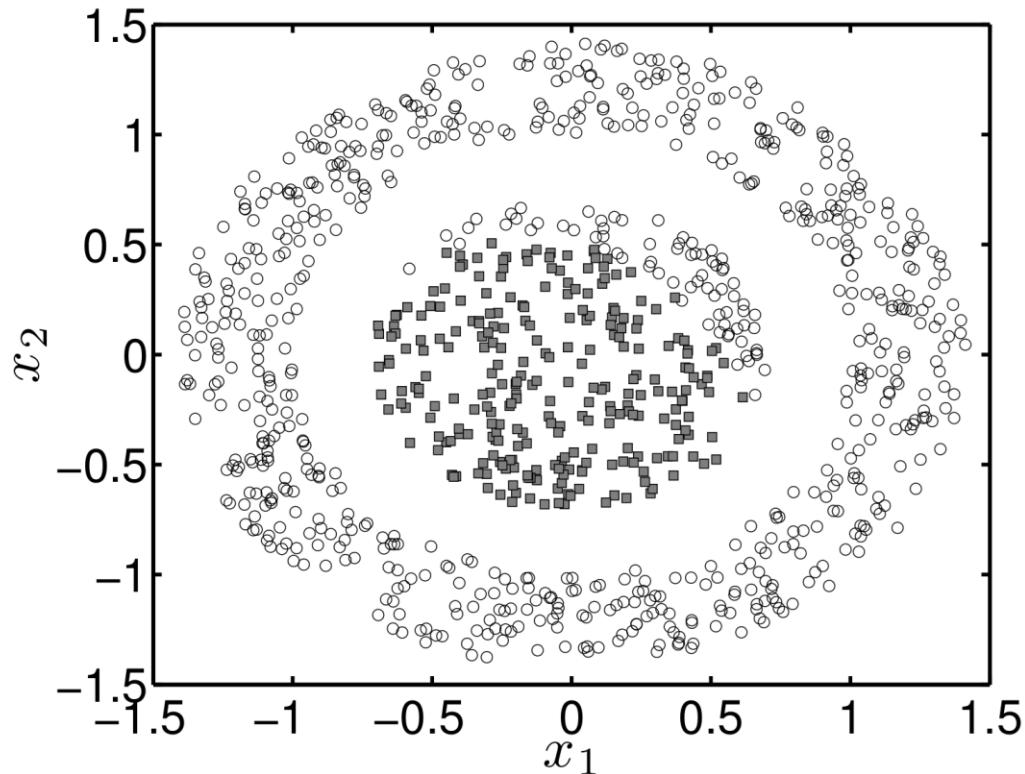
Kernel K-means - example



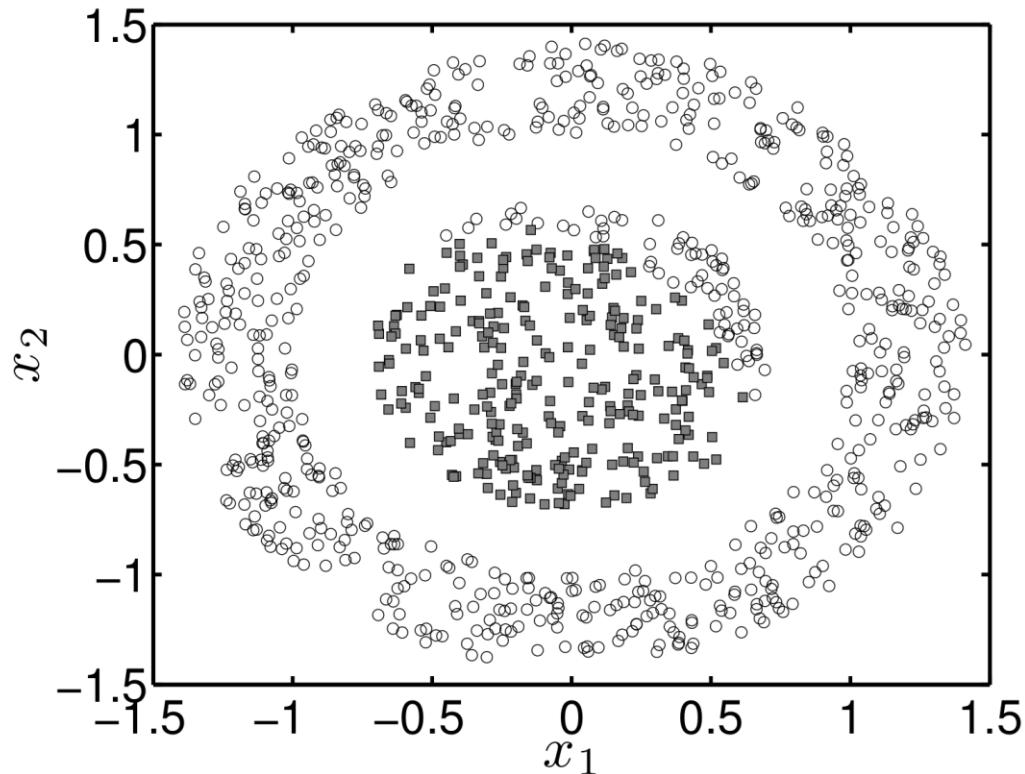
Kernel K-means - example



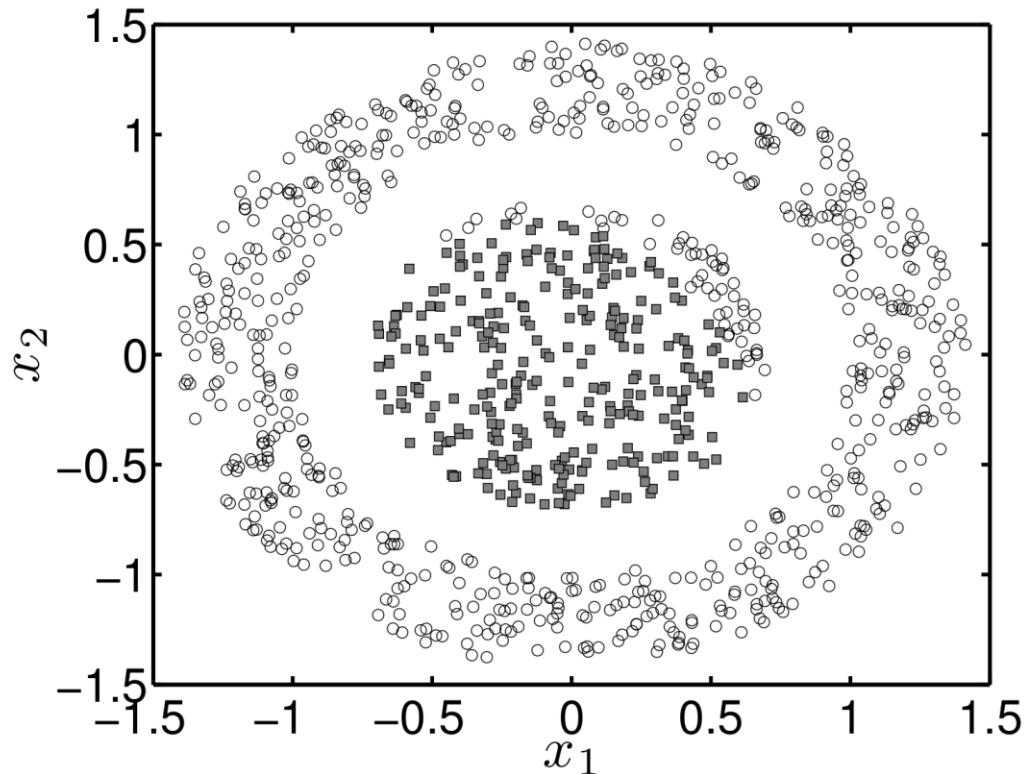
Kernel K-means - example



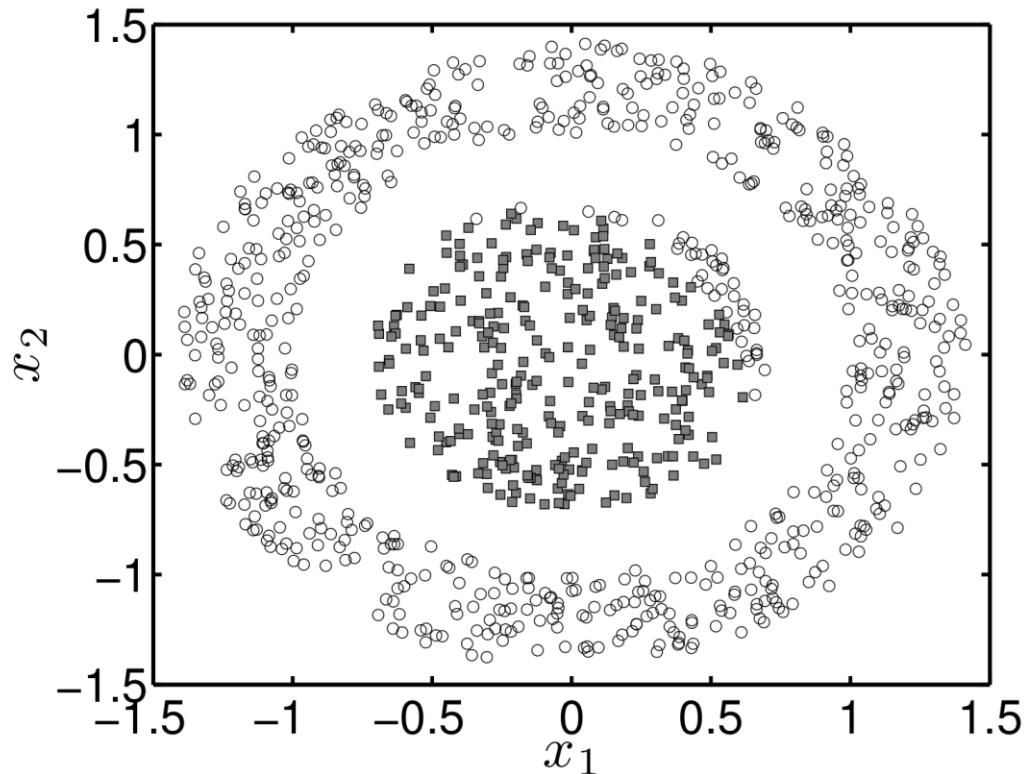
Kernel K-means - example



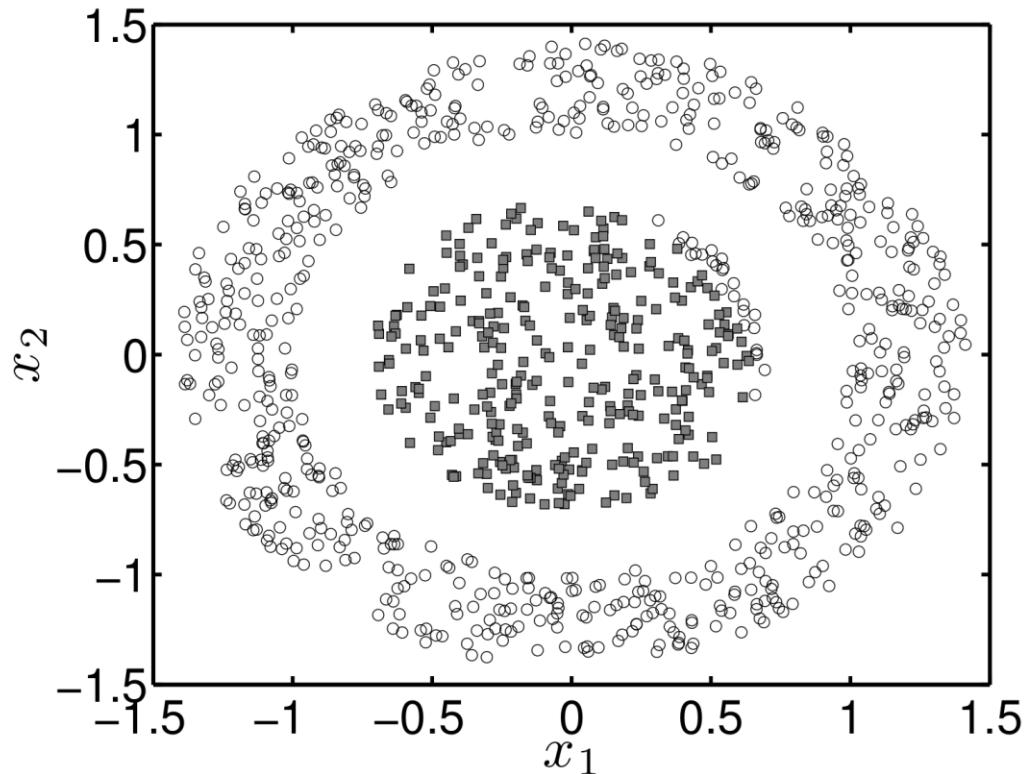
Kernel K-means - example



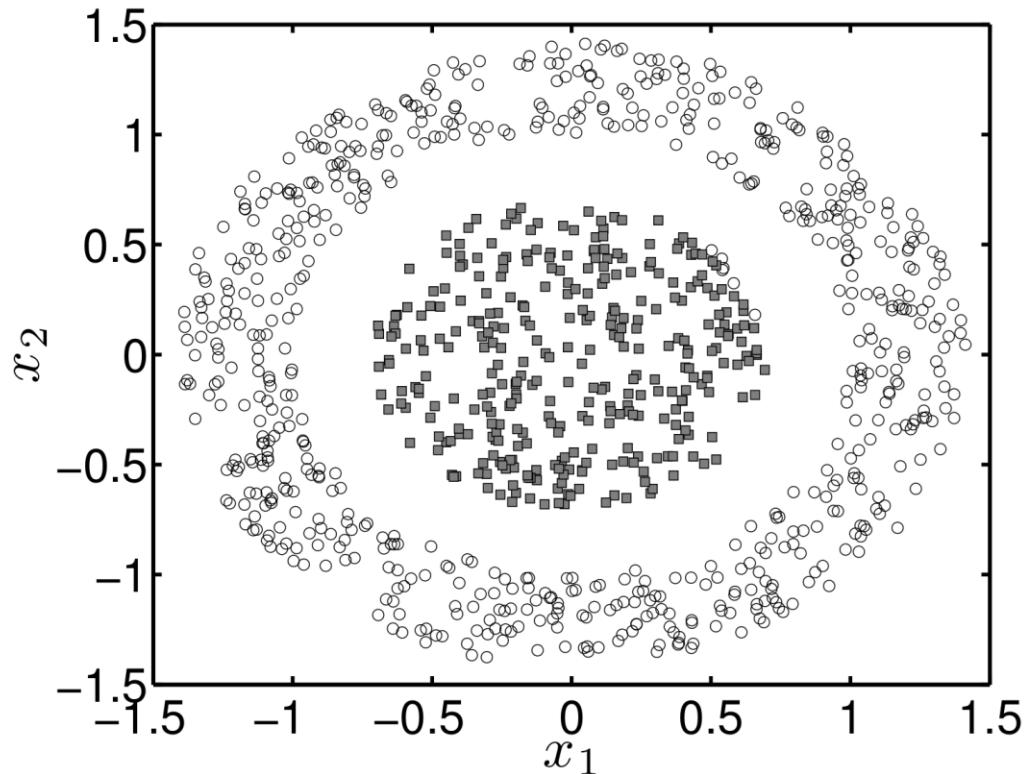
Kernel K-means - example



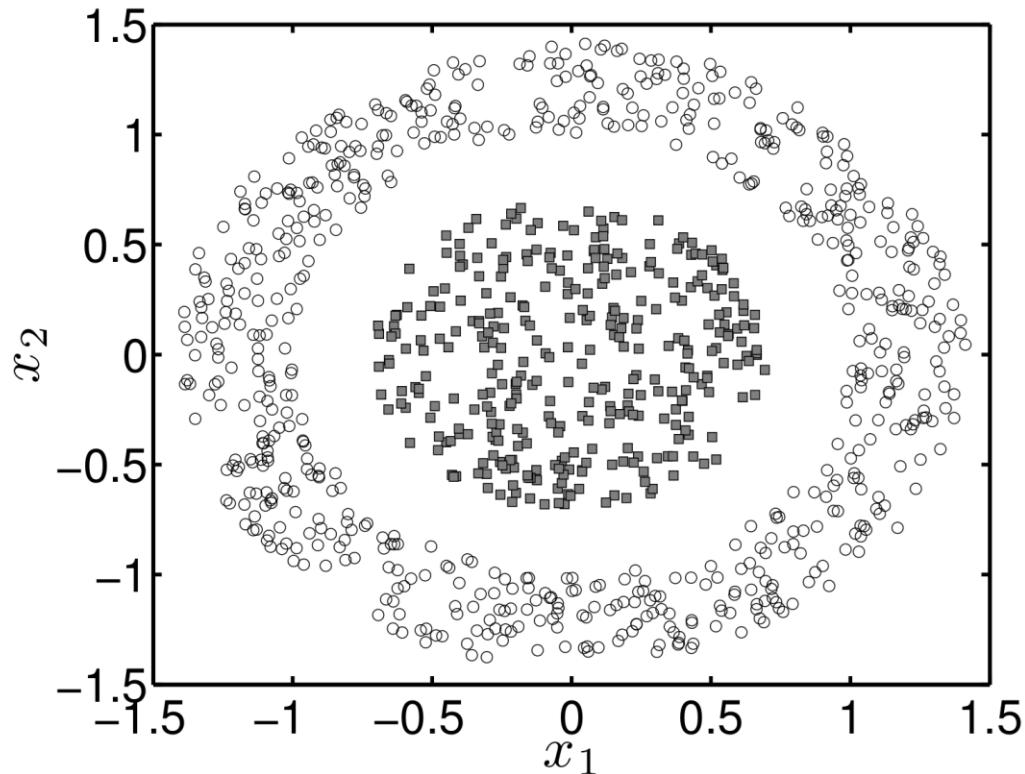
Kernel K-means - example



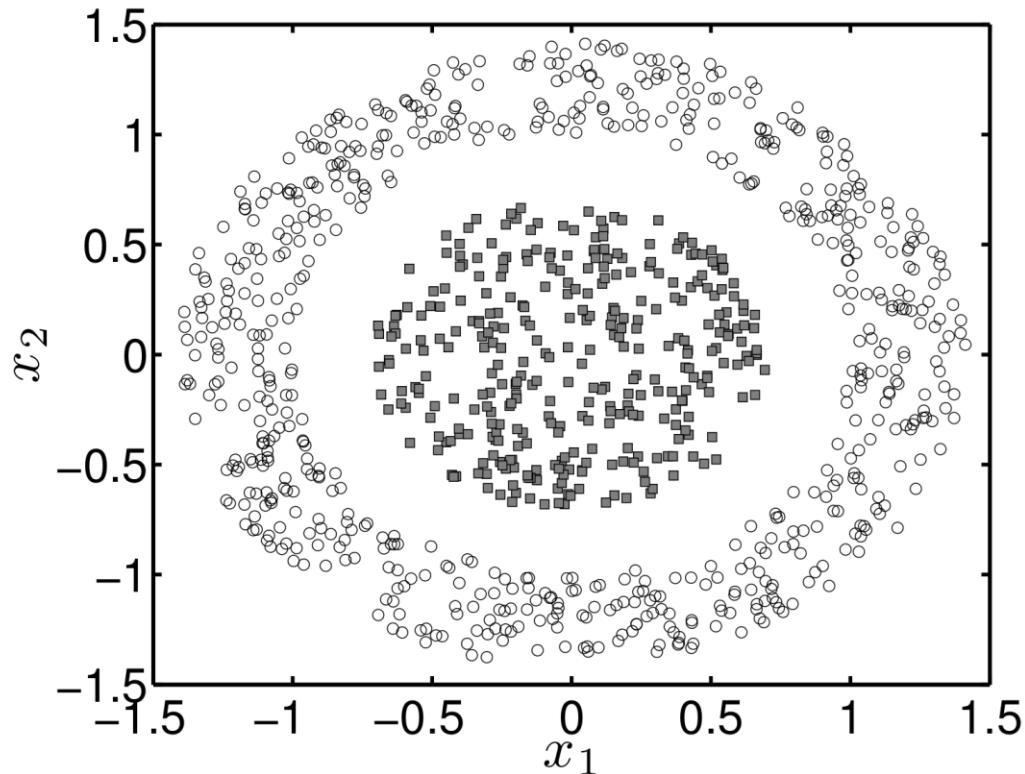
Kernel K-means - example



Kernel K-means - example



Kernel K-means - example



Kernel K-means

- ▶ Makes simple K-means algorithm more flexible.
- ▶ But, have to now set additional parameters.
- ▶ Very sensitive to initial conditions – lots of local optima.

K-means - summary

- ▶ Simple (and effective) clustering strategy.
- ▶ Converges to (local) minima of:

$$\sum_n \sum_k z_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k)^T (\mathbf{x}_n - \boldsymbol{\mu}_k)$$

- ▶ Sensitive to initialisation.
- ▶ How do we choose K ?
 - ▶ Tricky: Quantity above always decreases as K increases.
 - ▶ Can use CV if we have a measure of ‘goodness’.
 - ▶ For clustering these will be application specific.