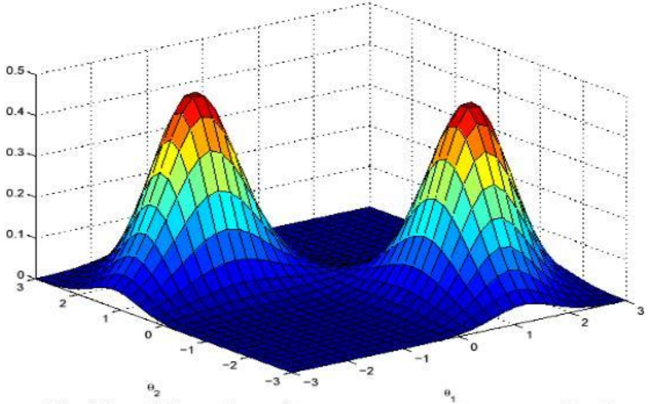
Week 8 Discussion: The EM Algorithm and Kernels



Likelihood function of two component means Θ_1, Θ_2

The Expectation Maximization Algorithm (EM)

$$L(\boldsymbol{\theta}; \mathbf{X}) = p(\mathbf{X}|\boldsymbol{\theta})$$

Finding the MLE through using a 2-step iterative method, repeat until convergence

E-step

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \operatorname{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}}\left[\log L(\boldsymbol{\theta}; \mathbf{X}, \mathbf{Z})\right]$$

M-step

$$\boldsymbol{\theta}^{(t+1)} = \underset{\boldsymbol{\theta}}{\arg\max}\, Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$$

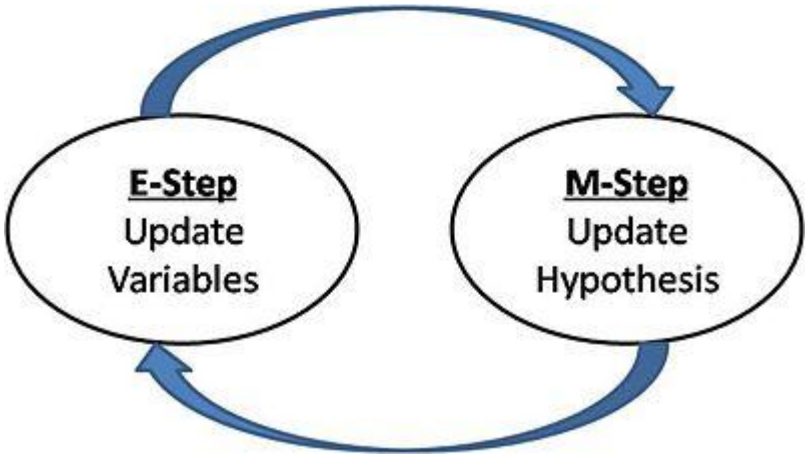
The Expectation Maximization Algorithm (EM)

- Finding the maximum likelihood estimates of parameters of a model (i.e., a Gaussian) using an iterative method
- Structured similarly to k-means (alternate between compute and update, E-Step vs. M-Step)
- Commonly used on GMM (Gaussian mixture models/mixtures of Gaussians)

In simple words:

- **E-step:** maximizing the expectation of the log likelihood function using data and **fixed** parameters
- **M-step:** updating parameters to maximize the expectation computed in the E-step

Continue until convergence



Expectation Maximization for MDG

1) Initialize $\vec{u}_c, \Sigma_c, \pi_c$ for $c=1 \ldots k$

2) E-step: Using the current parameters, evaluate

$$\gamma(z_{ic}) = \frac{\pi_c \cdot N(\vec{x} \mid \vec{u}_c, \Sigma_c)}{\sum^{k} \pi_j \cdot N(\vec{x} \mid \vec{u}_j, \Sigma_j)} \quad \text{for } i=1 \ldots n, \; c=1 \ldots k$$

for every point, you compute the probability and normalize

- use these to solve parameters

3) M-step: update parameters

$$\vec{u}_c^{\,new} = \frac{1}{n_c} \sum^{n} \gamma(z_{ic}) \vec{x}_i \qquad \Sigma_c^{new} = \frac{1}{n_c} \sum^{n} \gamma(z_{ic}) (\vec{x}_i - \vec{u}_c^{\,new})(x_i - u_c^{new})^T$$

$$\pi_c^{new} = \frac{n_c}{n} \qquad n_c = \sum_{i=1}^{n} \gamma(z_{ic})$$

4) evaluate the loglikelihood and check for convergence

if not, go back to step 2.

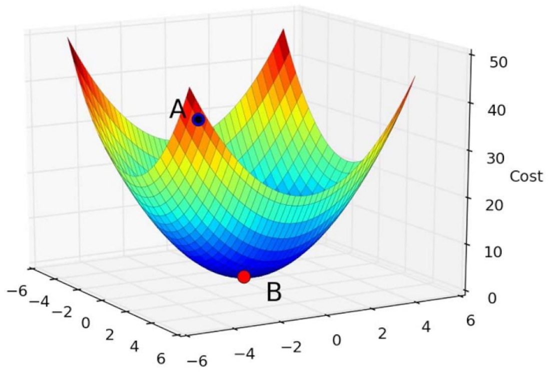
Notes credit: Seung Hee Lee (thank you!)

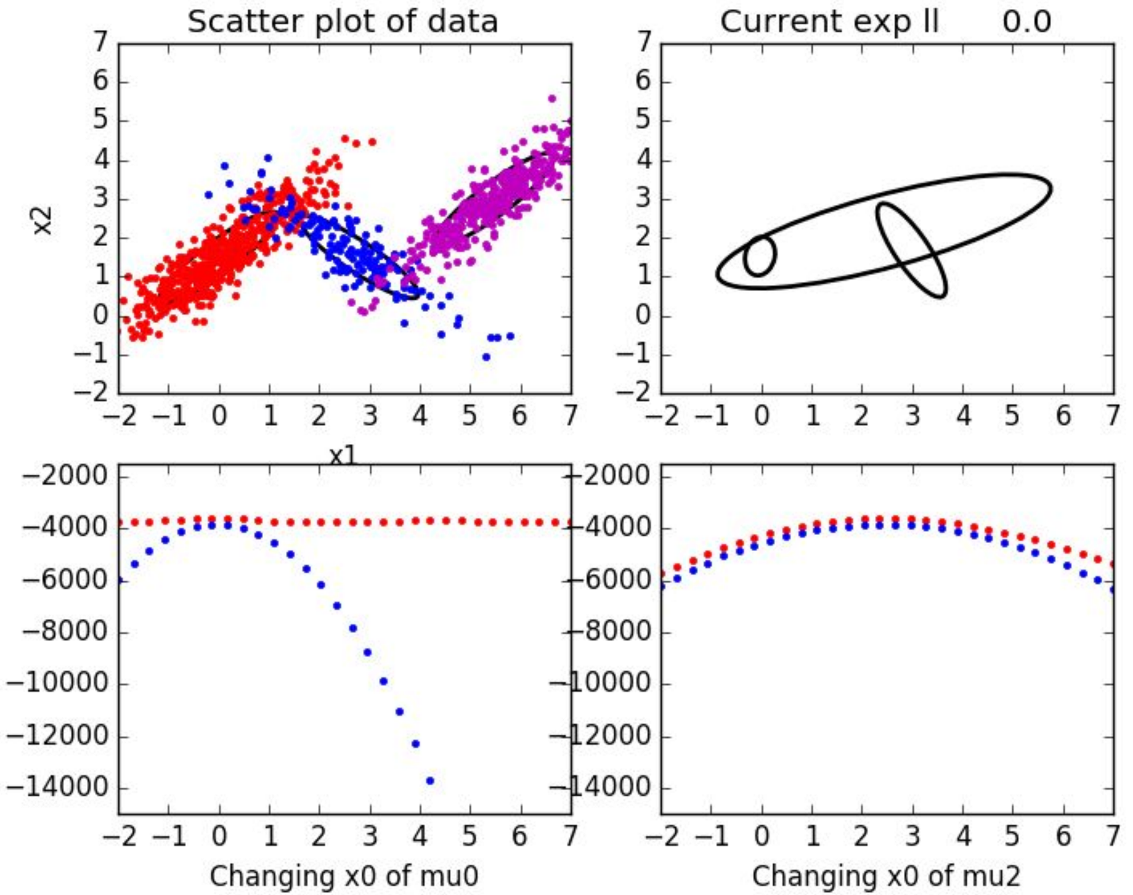
Main Insights

An iterative approach to finding the MLE

- Useful when MLE has no closed form solution
- Similar methods: SGD, Conjugate Gradient Method, Gauss-Newton Method

- Like all iterative methods,susceptible to converging in local minima

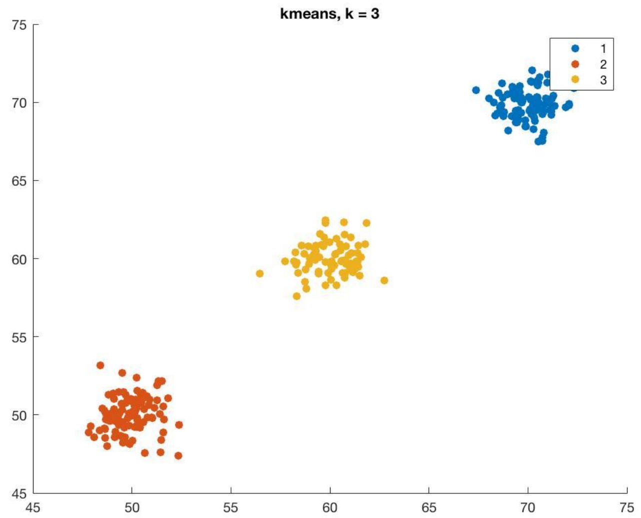


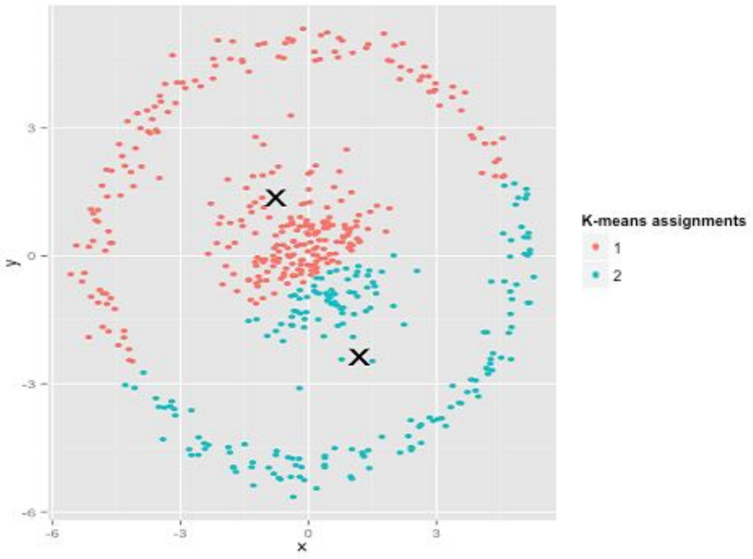


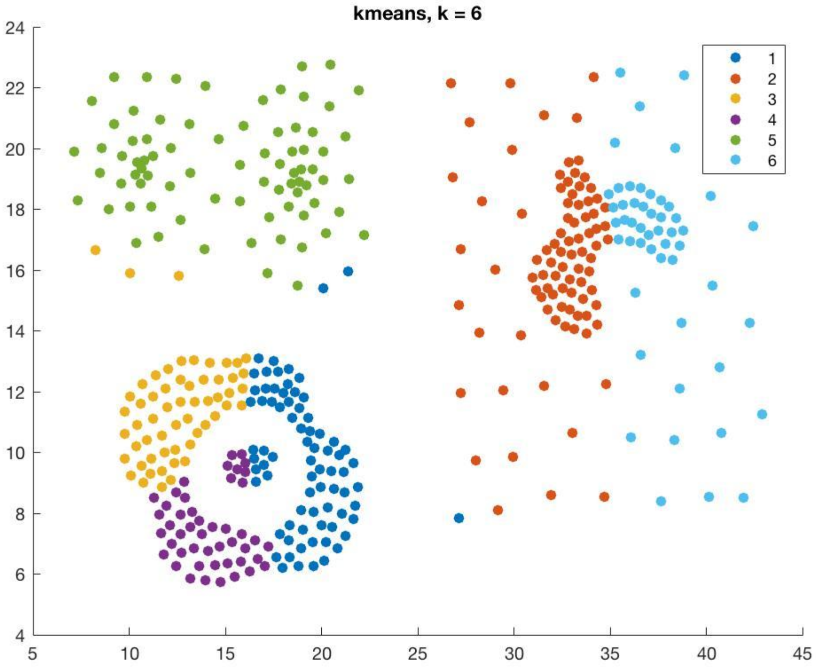
Scatter plot of data
Current exp ll 0.0
x2
x1
Changing x0 of mu0
Changing x0 of mu2

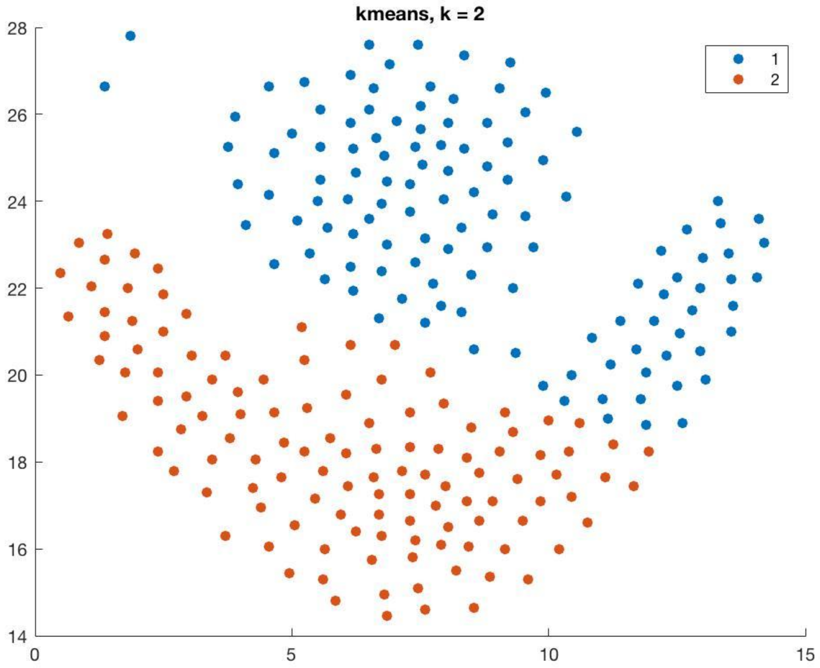
Kernel k-means

- K-means does not perform well on non-linearly separated data









Kmeans on non-linearly separated data

Kernel k-means

- Basic idea: map data onto a higher dimensional space to capture non-linearity, apply k-means on this mapping
- How is mapping done? Replace Euclidean distance with a non-linear mapping function $\phi(x)$
- Why is this good? No need to calculate the actual mapping, can directly use the mapping of the inner product of mapping, i.e. $\phi(x)^T \cdot \phi(x)$, which is much simpler to calculate

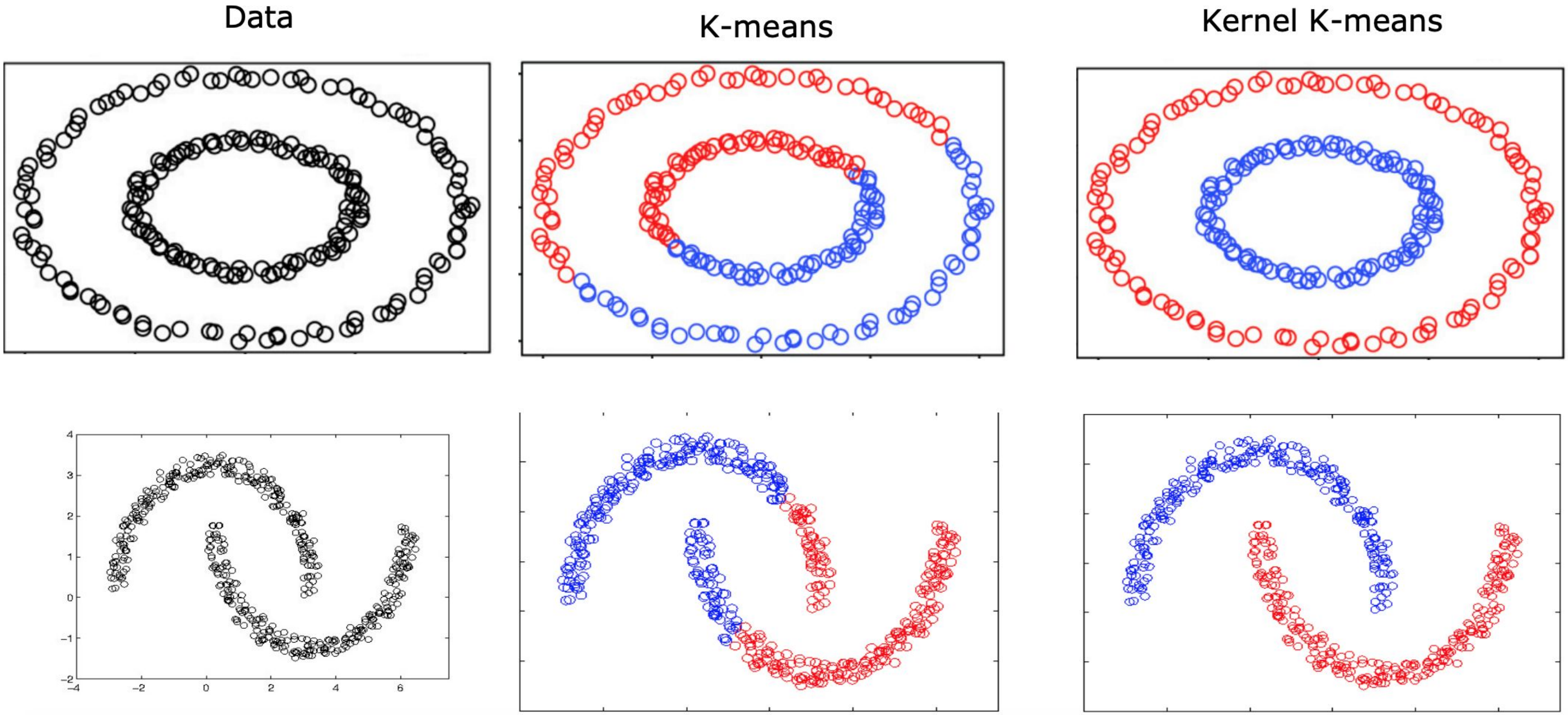


Image source :http://www.cse.msu.edu/~cse902/S14/ppt/kernelClustering.pdf