Unsupervised learning

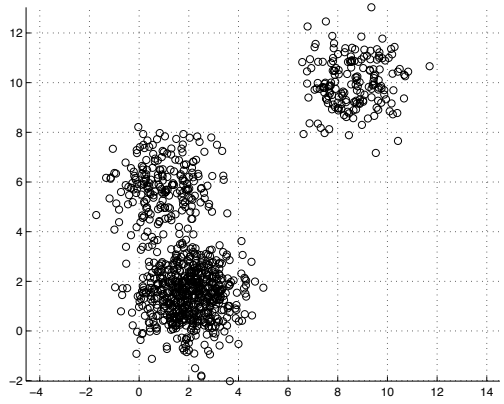
Devika Subramanian

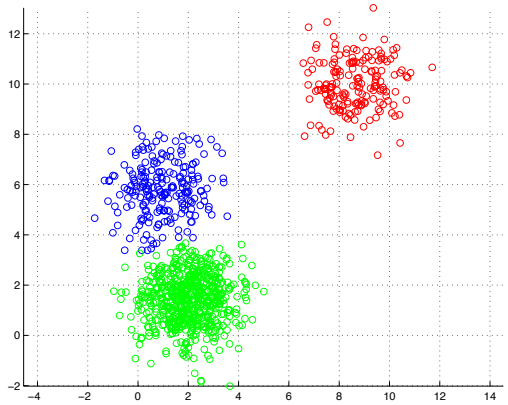
Unsupervised Learning

- No labels needed
- Good starting point for exploring data
  - Data structure
  - Outliers
  - Understanding of features
- Sometimes used as a precursor to supervised methods

K-means

- Given
  - m data points $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ in d-dimensional space
  - K desired clusters
- Goal
  - Partition the data points into the K clusters
  - Minimizing the inter-point distances within each cluster





http://www.mathworks.com/matlabcentral/fileexchange/26182-kernel-k-means

Notation

- μ_k is the D-dimensional center of cluster k
- 1-of-K coding scheme for encoding cluster membership

$$z_k^{(i)} = 1 \text{ if } x^{(i)} \text{ in cluster k}$$
$$= 0 \text{ otherwise}$$

Cost / Objective Function

- Sum of squared error

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} z_k^{(i)} \left\| x^{(i)} - \mu_k \right\|^2$$

The algorithm

- 0. Choose values for μ
- 1. Cluster assignment (E Step)
  - Assign each data point x to the best cluster
    - **Minimize J wrt z, keeping μ fixed**
- 2. Relocate means (M Step)
  - Update the value of each μ
    - **Minimize J wrt μ, keeping z fixed**
- Repeat until stable

E-step: assigning points to clusters

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} z_k^{(i)} \left\| x^{(i)} - \mu_k \right\|^2$$

- J is linear in z
- Each $x^{(i)}$ is independent
- Calculate J for each value of k for each point $x^{(i)}$
- Select the value of k that has the smallest value for J

$$z_k^{(i)} = \begin{cases} 1 \text{ if } k = \arg\min_j \left\| x^{(i)} - \mu_j \right\|^2 \\ 0 \text{ otherwise} \end{cases}$$

M-step: relocate means

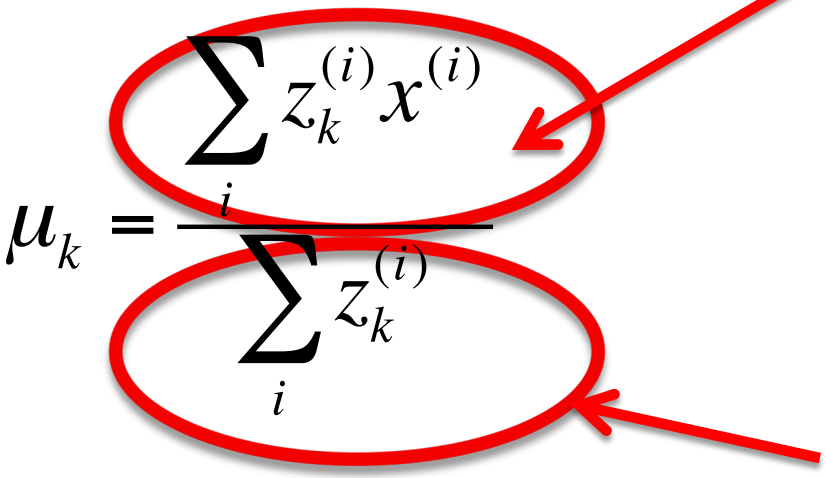
$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} z_k^{(i)} \left\| x^{(i)} - \mu_k \right\|^2$$

- J is quadratic in μ
- Each $x^{(i)}$ is independent
- To minimize J keeping z fixed; we take derivative wrt μand set equal to 0

$$2 \sum_{i=1}^{N} z_k^{(i)} (x^{(i)} - \mu_k) = 0$$

$$\mu_k = \frac{\sum_i z_k^{(i)} x^{(i)}}{\sum_i z_k^{(i)}}$$

A closer look at relocated cluster mean



sum of the values of the points assigned to cluster k

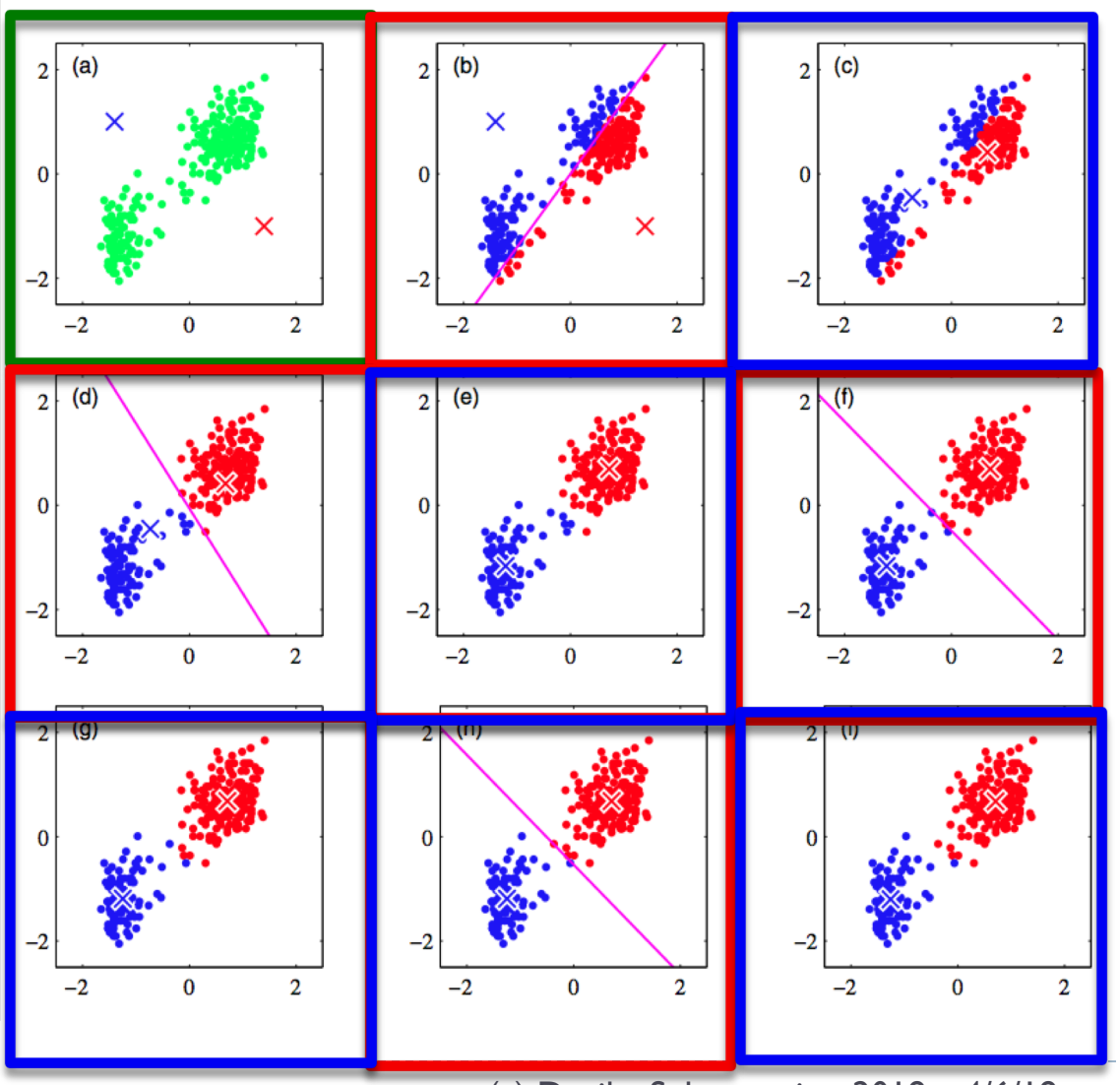
number of data points assigned to cluster k

Algorithm Details

- Convergence guaranteed
  - J is reduced at each step
  - Local minimum possible
- Can be slow
  - O(mK)
  - Compute distance for each point for every cluster
  - Faster variations exist
  - Often run multiple times with different starting values for mu_k

Old Faithful Example

Initialize



Assignment / E Step

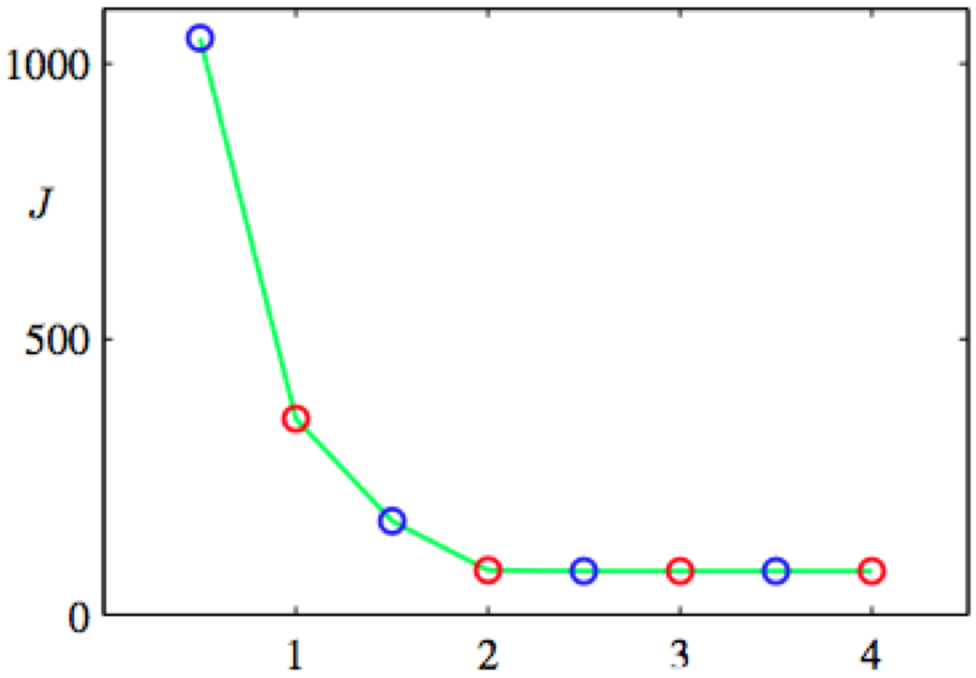
Relocate / M Step

Final State

Cost Function over Time

- Plot of J after each step
- Red points are E steps
- Blue points are M steps

- Could have chosen better initial points
  - Random



Choices

- Number of clusters K
  - Prior knowledge
  - Application requirements
  - Experimentation
    - Try a few, choose the one that minimizes the cost function
    - Try a few, choose the one the minimizes a different function
    - Start with 2, then 4, 8, …
- Initial cluster means
  - Random sample from the dataset
  - Results from clustering a small subset of the data
  - Global mean with noise / perturbation

More Choices

- Distance Function
  - Doesn't have to be Euclidean distance
  - Any dissimilarity measure Nu(x, x') can be used
  - This gives us a distortion measure

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} z_k^{(i)} \nu(x^{(i)}, \mu_k)$$

K-medoids algorithm

Pros / Cons

- Pros
  - + Straight-forward
  - + Somewhat scalable O(mK)
  - + Effective
  - + Guaranteed to converge
- Cons
  - – Requires a distance metric
  - – Vulnerable to outliers and midpoints
  - – May find local minima
  - – Dependent on initial choices
  - – Some clusters can become empty
  - – Each point is assigned to a single cluster

Best For

- Convex shapes
- Similar sized clusters
- K << m

An Application

- Image Segmentation
  - Partition an image
  - Group similar regions together
  - Use the RGB 3D numerical representation of each pixel
  - K corresponds to the available colors