Machine Learning

Lecture 7: K-Means

Feng Li

fli@sdu.edu.cn

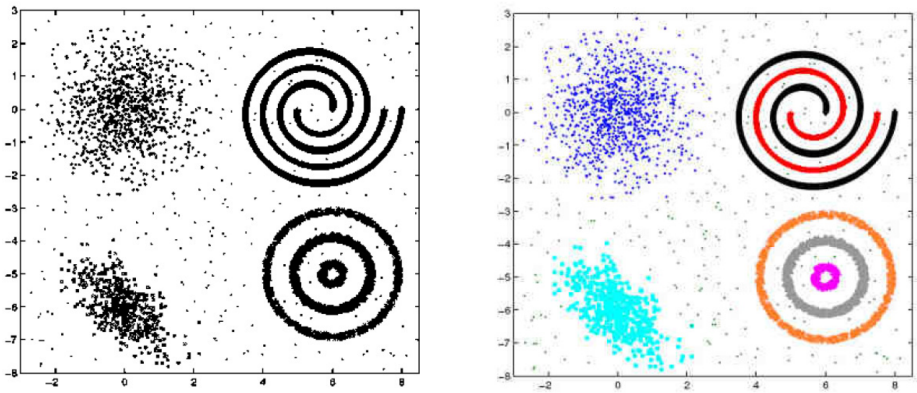
https://funglee.github.io

School of Computer Science and Technology
Shandong University

Fall 2017

Clustering

- Usually an unsupervised learning problem
- Given: N unlabeled examples $\{x_1, \cdots, x_N\}$; no. of desired partitions K
- Goal: Group the examples into K "homogeneous" partitions



(a) Input data (b) Desired clustering

- Loosely speaking, it is classification without ground truth labels
- A good clustering is one that achieves:
  - High within-cluster similarity
  - Low inter-cluster similarity

Similarity can be Subjective

- Clustering only looks at similarities, no labels are given
- Without labels, similarity can be hard to define
- Goal: Group the examples into K "homogeneous" partitions

- Thus using the right distance/similarity is very important in clustering
- Also important to define/ask: "Clustering based on what"?

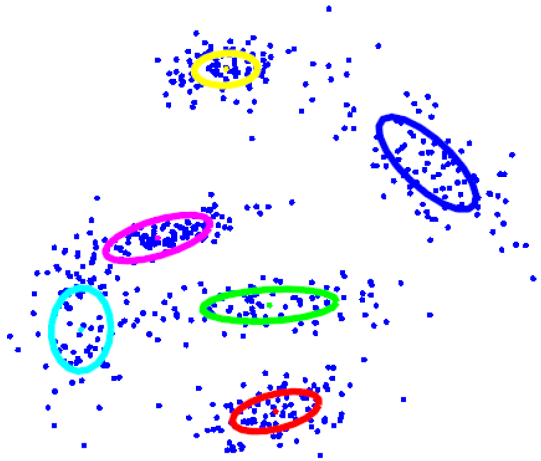
Clustering: Some Examples

- Document/Image/Webpage Clustering
- Image Segmentation (clustering pixels)

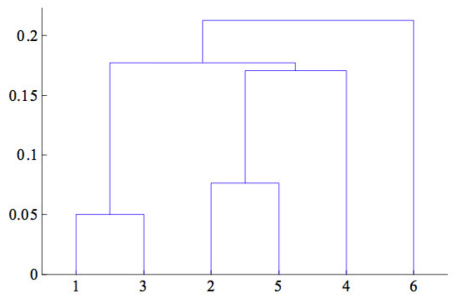
- Clustering web-search results
- Clustering (people) nodes in (social) networks/graphs
- ... and many more...

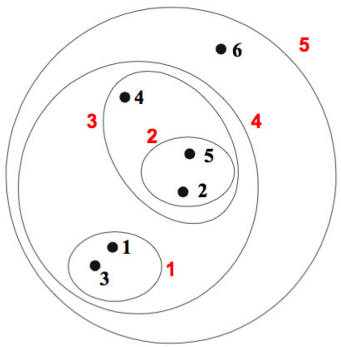
Types of Clustering

- Flat or Partitional clustering: Partitions are independent of each other



- Hierarchical clustering: Partitions can be visualized using a tree structure (a dendrogram)





K-Means Clustering Problem

- Given a set of observations $\mathbf{X} = \{x_1, x_2, \cdots, x_N\}$ $(x_i \in \mathbb{R}^D)$, partition the N observations into K sets $(K \leq N)$ $\{\mathcal{C}_k\}_{k=1,\cdots,K}$ such that the sets minimize the within-cluster sum of squares:

$$\arg\min_{\{\mathcal{C}_k\}} \sum_{i=1}^{K} \sum_{x \in \mathcal{C}_i} \|x - \mu_i\|^2$$

  where μ_i is the mean of points in set $\mathcal{C}_i$
- How hard is this problem?
  - The problem is NP hard, but there are good heuristic algorithms that seem to work well in practice, e.g., K-means algorithm and mixtures of Gaussians

K-Means Algorithm (Lloyd, 1957)

- In each iteration
  - (Re)-Assign each example x_i to its closest cluster center (based on the smallest Euclidean distance)

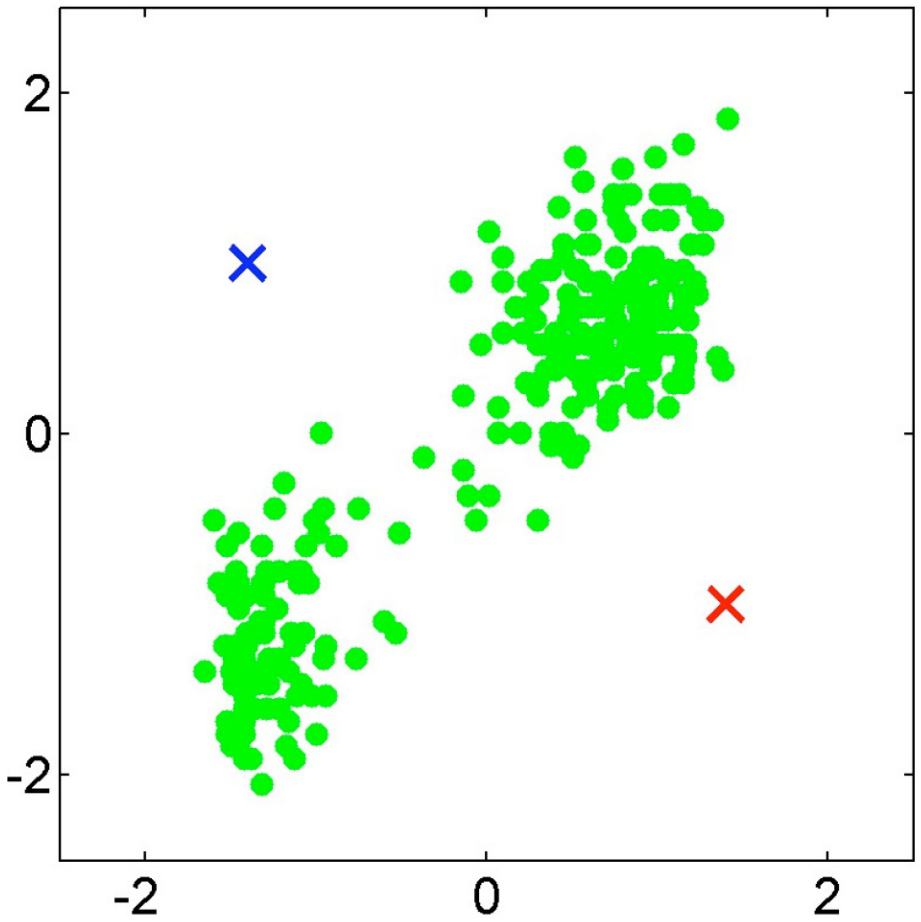
$$\mathcal{C}_{k^*} = \{x_i : k^* = \arg\min_k \|x_i - \mu_k\|^2\}$$

    ($\mathcal{C}_k$ is the set of examples assigned to cluster k with center μ_k)
  - Update the cluster means

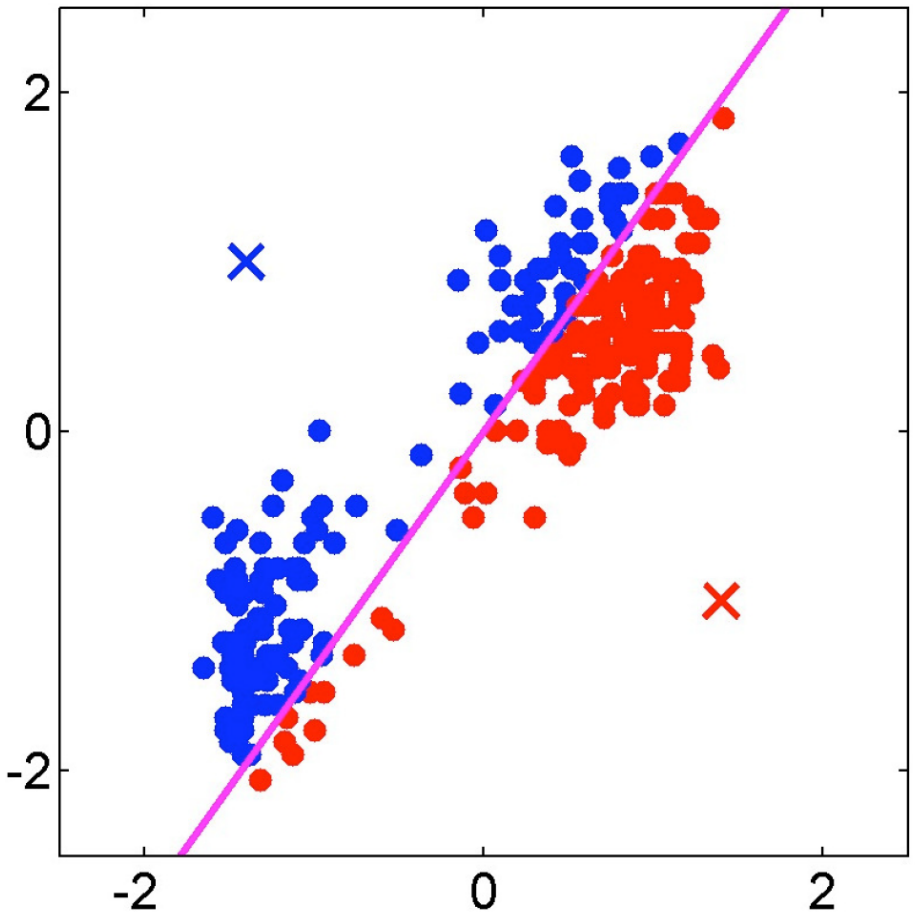
$$\mu_k = \text{mean}(\mathcal{C}_k) = \frac{1}{|\mathcal{C}_k|} \sum_{x \in \mathcal{C}_k} x$$

- Stop when cluster means or the "loss" does not change by much

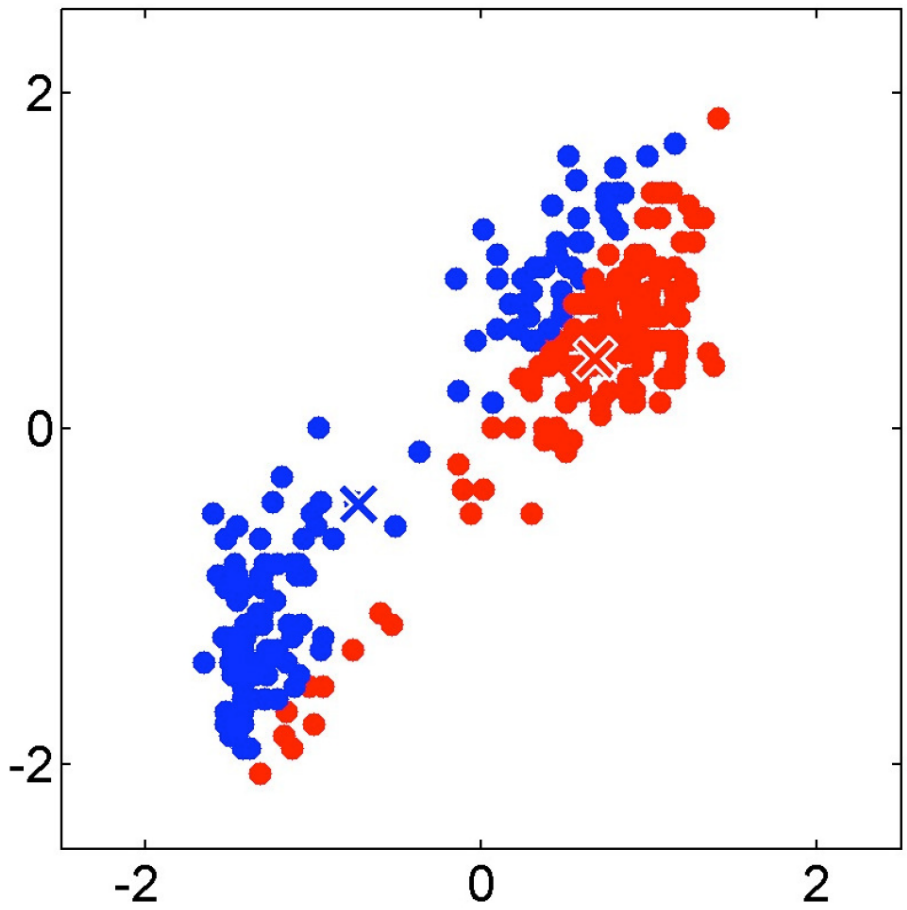
K-means: Initialization (assume K = 2)



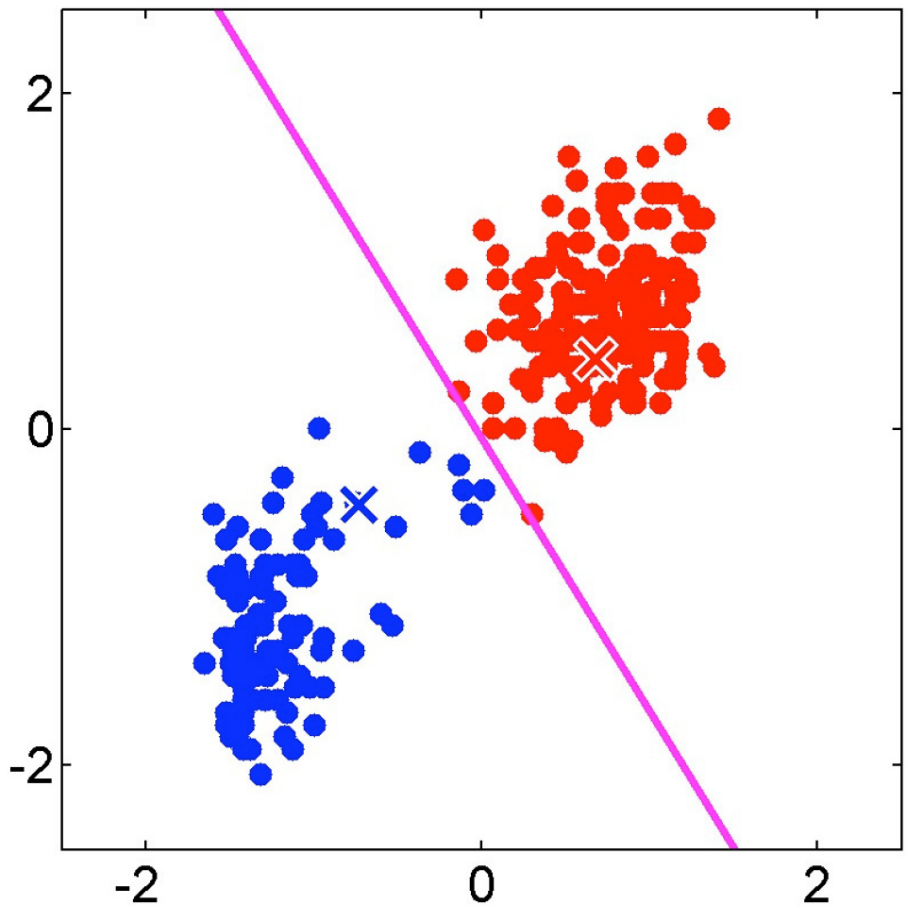
K-means iteration 1: Assigning points



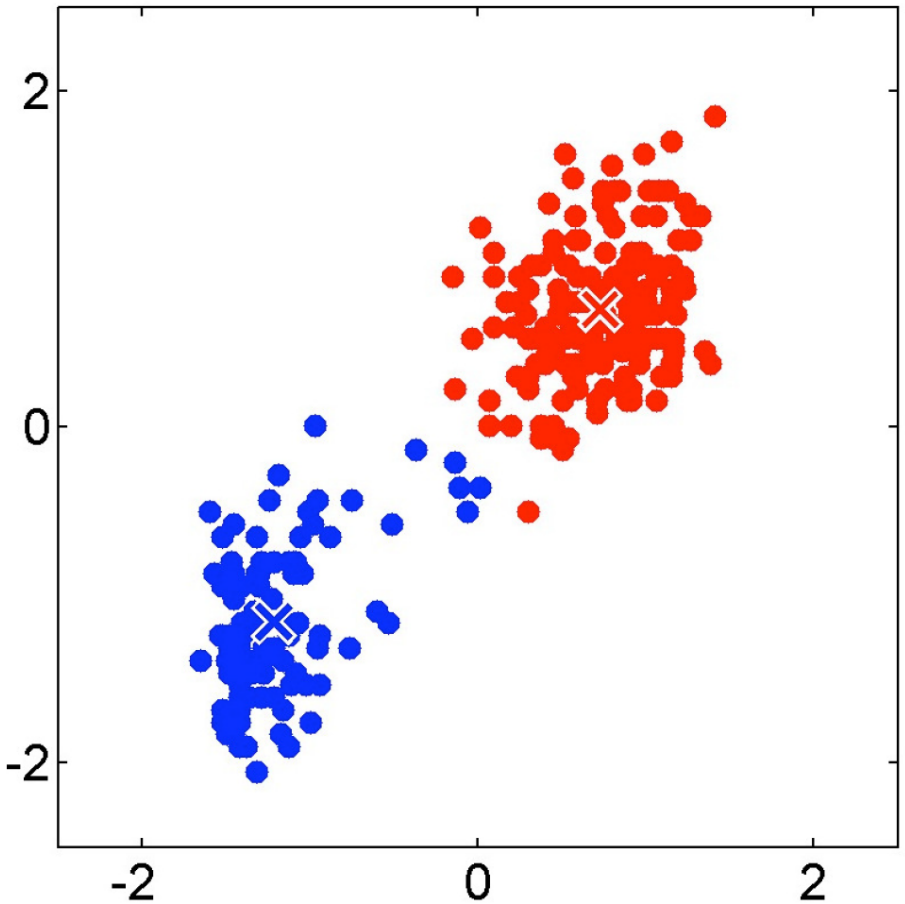
K-means iteration 1: Recomputing the centers



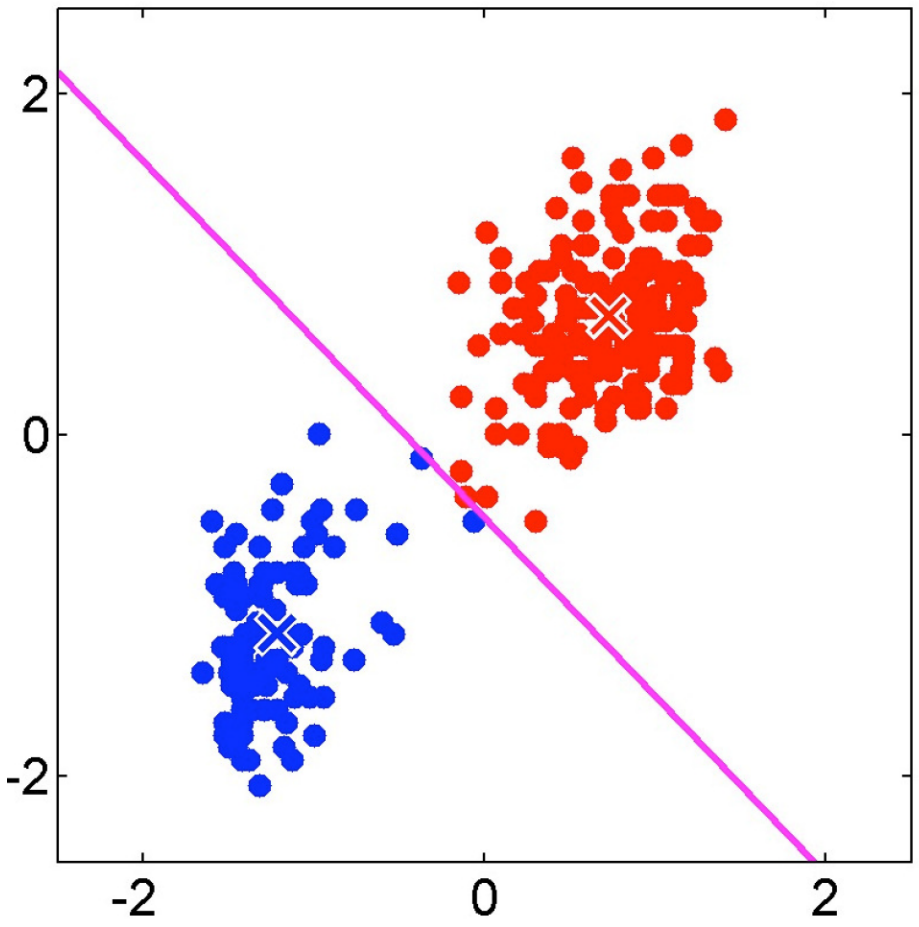
K-means iteration 2: Assigning points



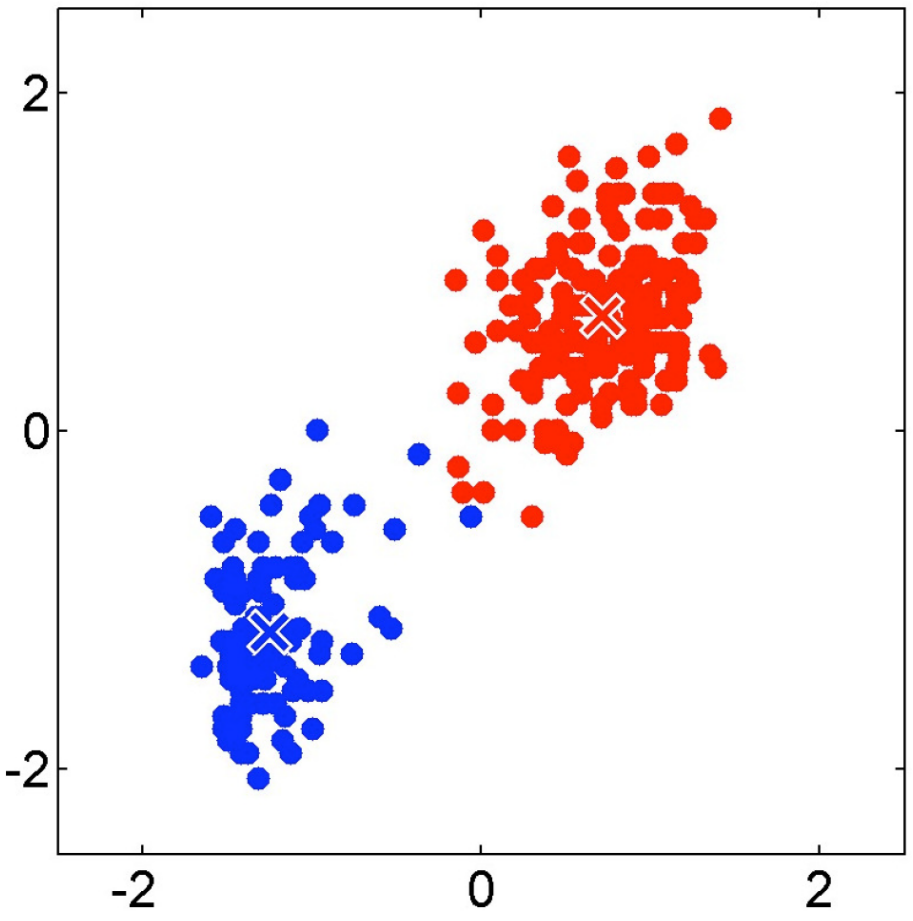
K-means iteration 2: Recomputing the centers



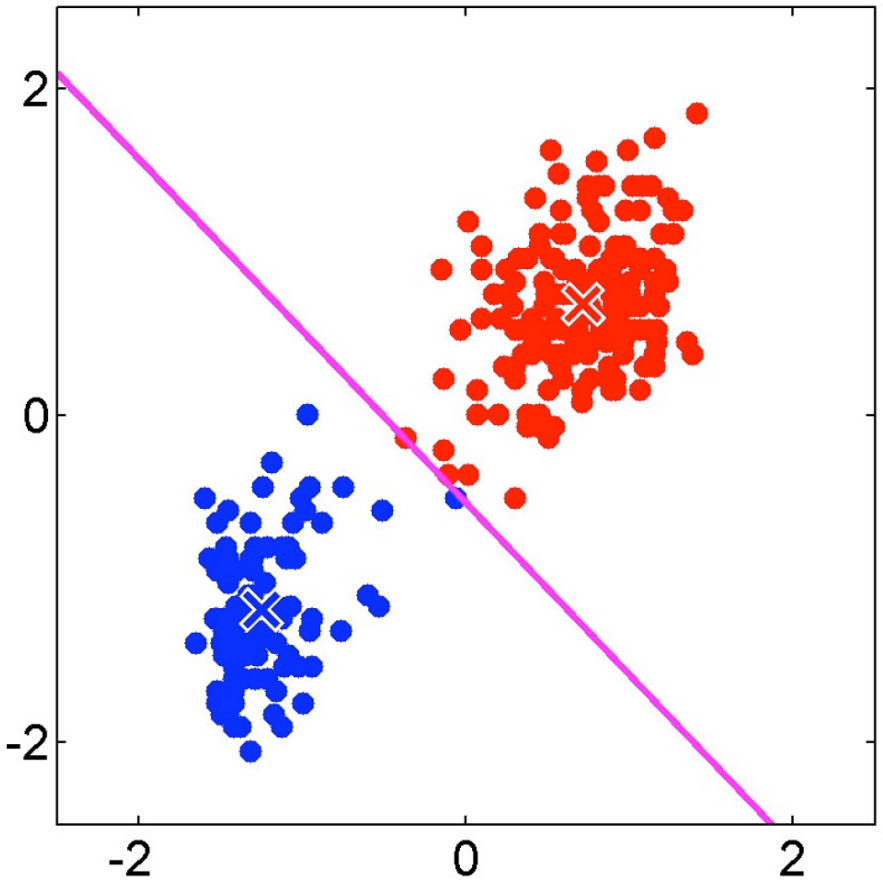
K-means iteration 3: Assigning points



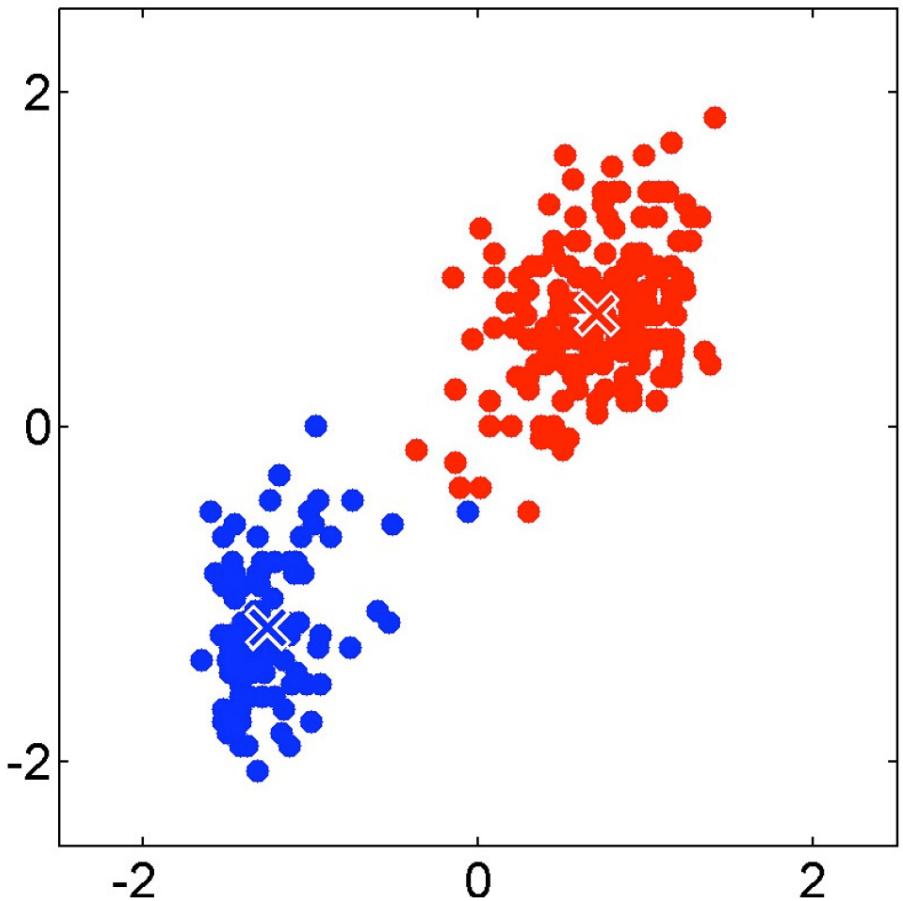
K-means iteration 3: Recomputing the centers



K-means iteration 4: Assigning points



K-means iteration 4: Recomputing the centers



What Loss Function is K-means Optimizing?

- Let $\mu_1, \mu_2, \cdots, \mu_K$ be the K cluster centroids (means)
- Let $z_{i,k}$ be s.t. $z_{i,k} = 1$ if x_i belongs to cluster k, and 0 otherwise
  - $z_i = [z_{i,1}, z_{i,2}, \cdots, z_{i,k}]$ represents a length K one-hot encoding of x_i
- Define the distortion or "loss" for the cluster assignment of x_i

$$\ell(\{\mu_k\}, x_i, z_i) = \sum_{k=1}^{K} z_{i,k} \|x_i - \mu_k\|^2$$

- Total distortion over all points defines the K-means "loss function"

$$L(\mu, \mathbf{X}, \mathbf{Z}) = \sum_{i=1}^{N} \sum_{k=1}^{K} z_{i,k} \|x_i - \mu_k\|^2 = \|\mathbf{X} - \mathbf{Z}\mu\|^2$$

  where $\mathbf{Z}$ is $N \times K$ and μ is $K \times D$
- The K-means problem is to minimize the above objective function w.r.t. μ and $\mathbf{Z}$

K-means Objective

- Consider the K-means objective function

$$L(\mu, \mathbf{X}, \mathbf{Z}) = \|\mathbf{X} - \mathbf{Z}\mu\|^2$$

- It is a non-convex objective problem
  - Many local minima possible
- Also NP-hard to minimize in general (note that $\mathbf{Z}$ is discrete)
- The K-means algorithm we saw is a heuristic to optimize this function
- K-means algorithm alternated between the following two steps
  - Fix μ, minimize w.r.t. $\mathbf{Z}$ (assign points to closest centers)
  - Fix $\mathbf{Z}$, minimize w.r.t. μ (recompute the center means)
- Note: The algorithm usually converges to a local minima (though may not always, and it may just convergence "somewhere"). Multiple runs with different initializations can be tried to find a good solution.

Convergence of K-means Algorithm

- Each step (updating $\mathbf{Z}$ or μ) can never increase the objective
- When we update $\mathbf{Z}$ from $\mathbf{Z}^{(t-1)}$ to $\mathbf{Z}^{(t)}$

$$L(\mu^{(t-1)}, \mathbf{X}, \mathbf{Z}^{(t)}) \leq L(\mu^{(t-1)}, \mathbf{X}, \mathbf{Z}^{(t-1)})$$

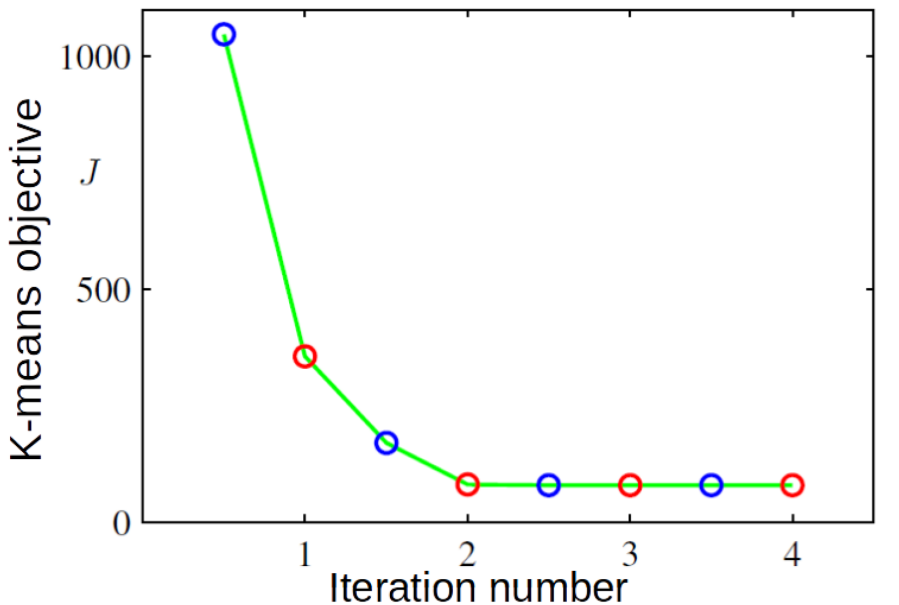
  because the new $\mathbf{Z}^{(t)} = \arg\min_{\mathbf{Z}} L(\mu^{(t-1)}, \mathbf{X}, \mathbf{Z})$
- When we update μ from $\mu^{(t-1)}$ to $\mu^{(t)}$

$$L(\mu^{(t)}, \mathbf{X}, \mathbf{Z}^{(t)}) \leq L(\mu^{(t-1)}, \mathbf{X}, \mathbf{Z}^{(t)})$$

  because the new $\mu^{(t)} = \arg\min_{\mu} L(\mu, \mathbf{X}, \mathbf{Z}^{(t)})$

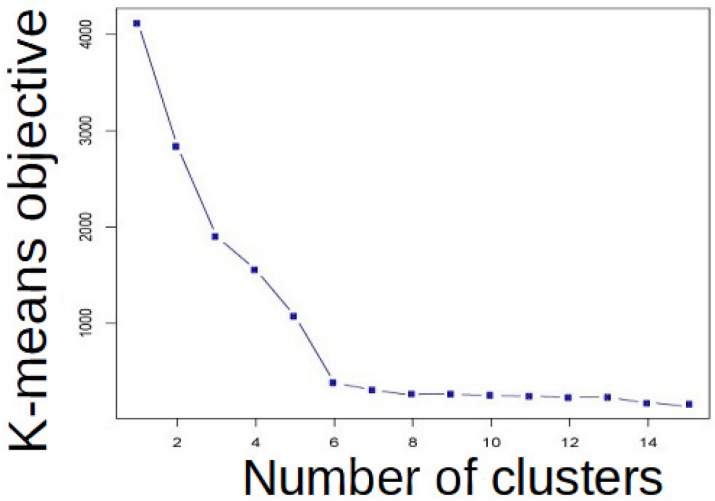
Convergence of K-means Algorithm (Contd.)

- Thus the K-means algorithm monotonically decreases the objective



Choosing K

- One way to select K for the K-means algorithm is to try different values of K, plot the K-means objective versus K, and look at the "elbow-point"



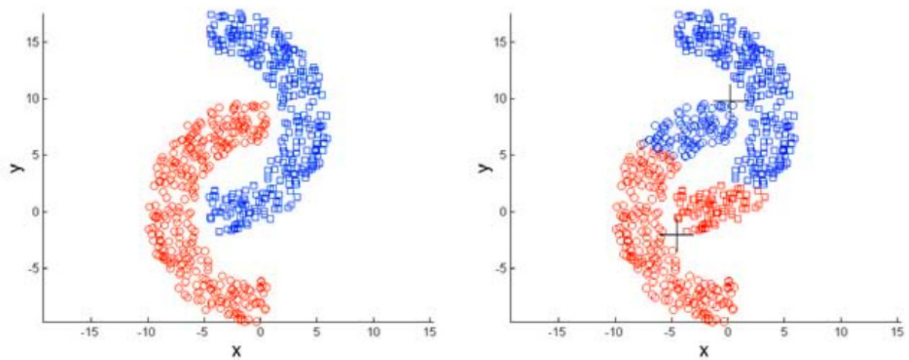
- For the above plot, $K = 6$ is the elbow point
- Can also information criterion such as AIC (Akaike Information Criterion)

$$AIC = 2L(\hat{\mu}, \mathbf{X}, \hat{\mathbf{Z}}) + K \log D$$

and choose the K that has the smallest AIC (discourages large K)

K-means: Some Limitations

- Makes hard assignments of points to clusters
  - A point either completely belongs to a cluster or does not belong at all
  - No notion of a soft assignment (i.e., probability of being assigned to each cluster: say $K = 3$ and for some point x_i, $p_1 = 0.7$; $p_2 = 0.2$; $p_3 = 0.1$)
- Works well only is the clusters are roughly of equal sizes
- Probabilistic clustering methods such as Gaussian mixture models can handle both these issues (model each cluster using a Gaussian distribution)
- K-means also works well only when the clusters are round-shaped and does badly if the clusters have non-convex shapes



- Kernel K-means or Spectral clustering can handle non-convex

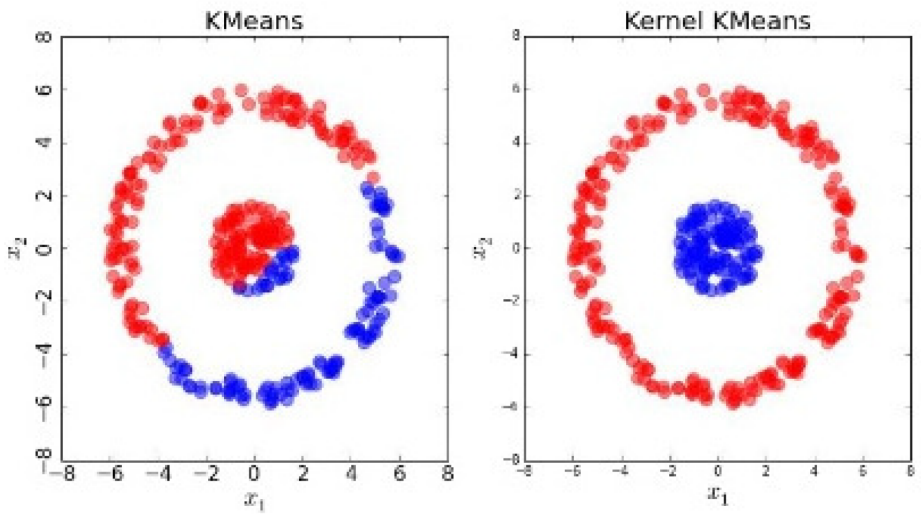
Kernel K-Means

- Basic idea: Replace the Euclidean distance/similarity computations in K-means by the kernelized versions.

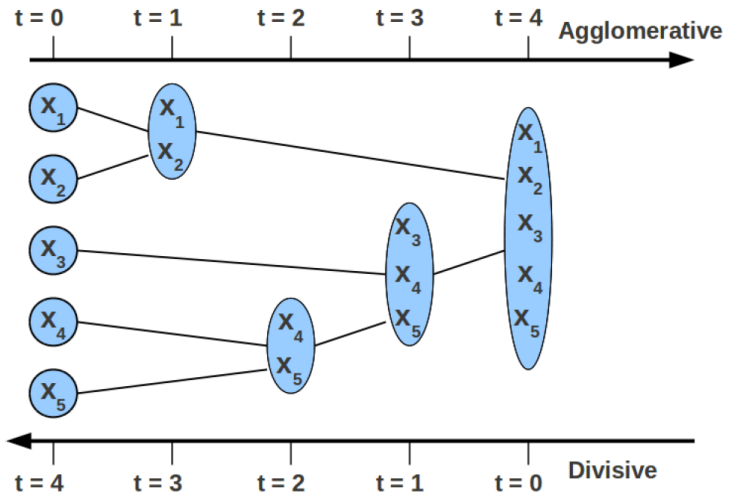
$$\begin{aligned} d(x_i, \mu_k) &= \|\phi(x_i) - \phi(\mu_k)\| \\ \|\phi(x_i) - \phi(\mu_k)\|^2 &= \|\phi(x_i)\|^2 + \|\phi(\mu_k)\|^2 - 2\phi(x_i)^T\phi(\mu_k) \\ &= k(x_i, x_i) + k(\mu_k, \mu_k) - 2k(x_i, \mu_k) \end{aligned}$$

  where $k(\cdot, \cdot)$ denotes the kernel function and ϕ is its (implicit) feature map

- Note: ϕ does not have to be computed/stored for data $\{x_i\}$ or the cluster means $\{\mu_k\}$ because computations only depend on kernel evaluations

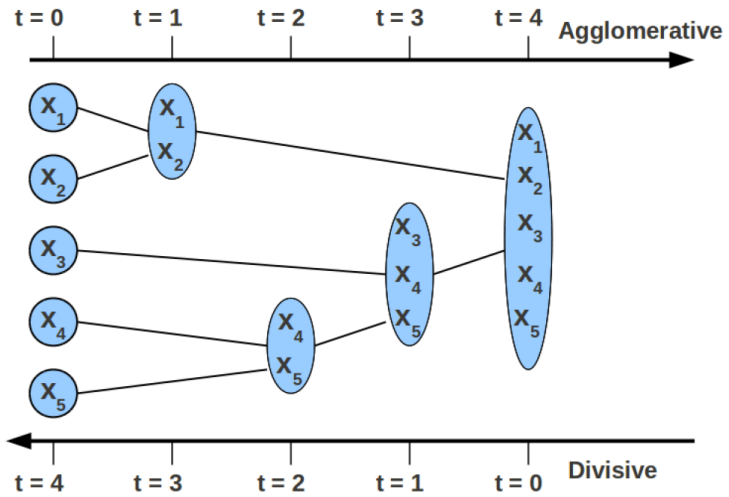


Hierarchical Clustering



- Agglomerative (bottom-up) Clustering
  1. Start with each example in its own singleton cluster
  2. At each time-step, greedily merge 2 most similar clusters
  3. Stop when there is a single cluster of all examples, else go to 2
- Divisive (top-down) Clustering
  1. Start with all examples in the same cluster
  2. At each time-step, remove the outsiders from the least cohesive cluster
  3. Stop when each example is in its own singleton cluster, else go to 2
- Agglomerative is more popular and simpler than divisive (but less accurarate)

Hierarchical Clustering



- Agglomerative (bottom-up) Clustering
  1. Start with each example in its own singleton cluster
  2. At each time-step, greedily merge 2 most similar clusters
  3. Stop when there is a single cluster of all examples, else go to 2
- Divisive (top-down) Clustering
  1. Start with all examples in the same cluster
  2. At each time-step, remove the outsiders from the least cohesive cluster
  3. Stop when each example is in its own singleton cluster, else go to 2
- Agglomerative is more popular and simpler than divisive (but less accurarate)

Hierarchical Clustering (Contd.)

- Metric: A measure of distance between pairs of observations
  - Euclidean distance: $\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
  - Squared Euclidean distance: $\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
  - Manhattan distance: $\|a-b\|_1 = \sum_i |a_i - b_i|$
  - Maximum distance: $\|a-b\|_\infty = \max_i |a_i - b_i|$
  - Mahalanobis distance: $\sqrt{(a-b)^T S^{-1} (a-b)}$ (S is the Covariance matrix)
- How to compute the dissimilarity between two clusters R and S?
  - Min-link or single link: results in chaining (clusters can get very large)

$$d(R,S) = \min_{x_R \in R, x_S \in S} d(x_R, x_S)$$

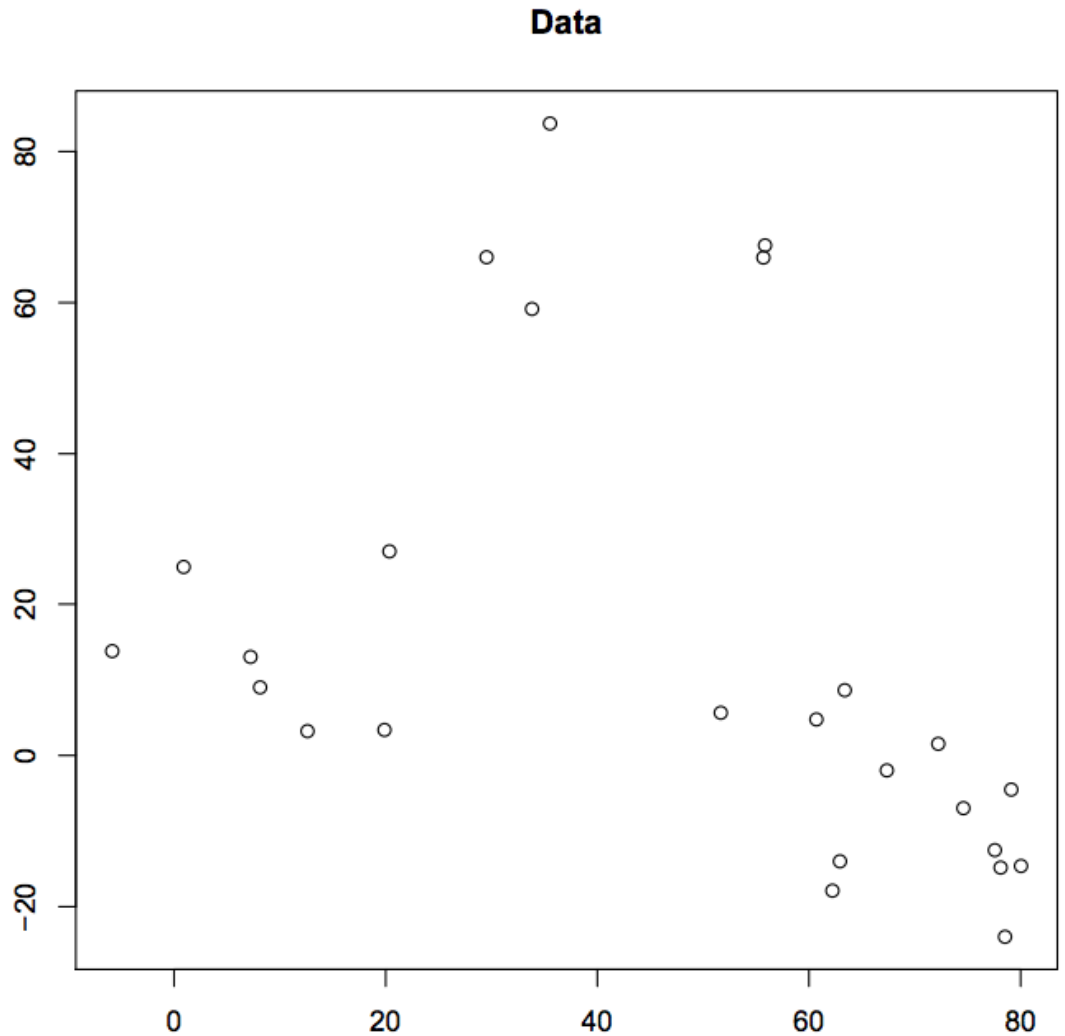
  - Max-link or complete-link: results in small, round shaped clusters

$$d(R,S) = \max_{x_R \in R, x_S \in S} d(x_R, x_S)$$

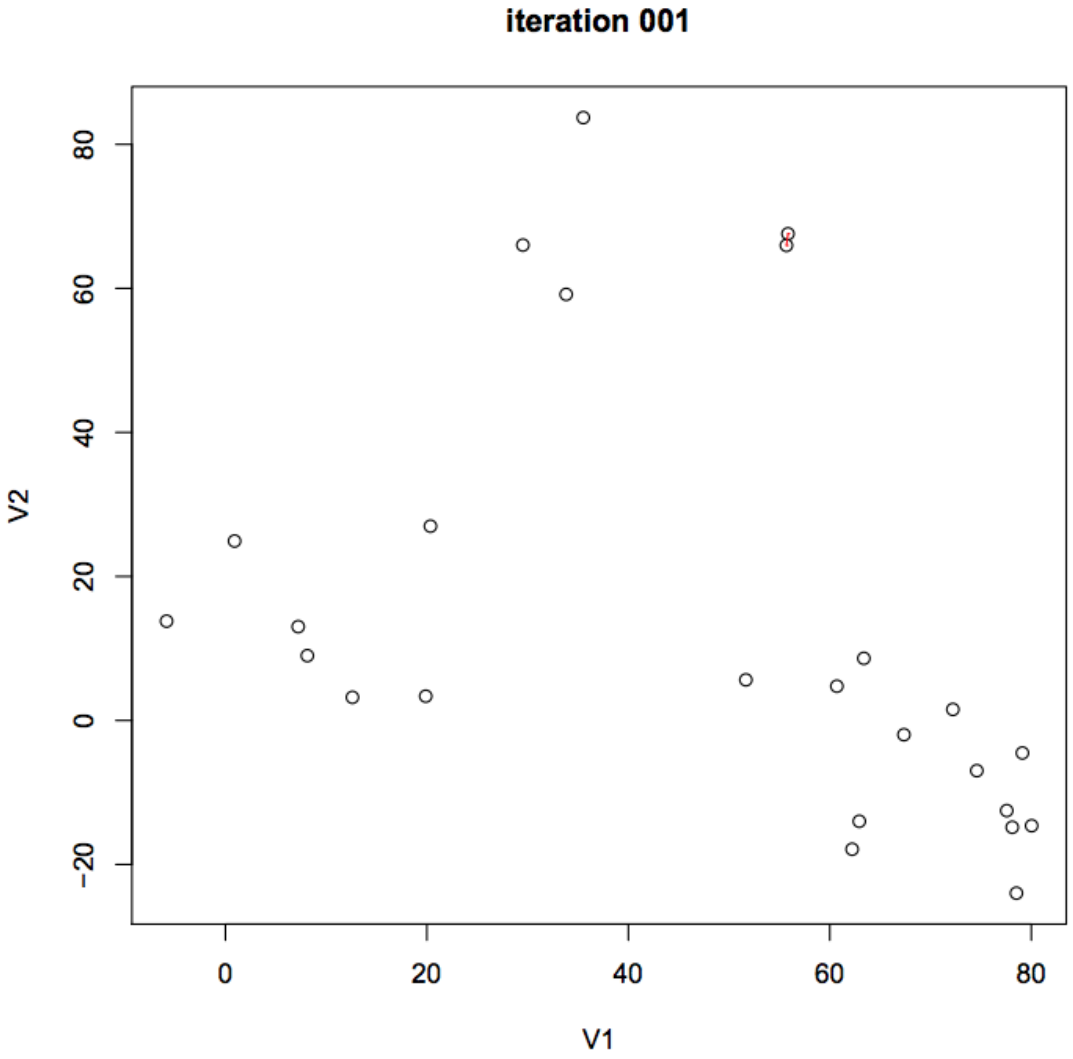
  - Average-link: compromise between single and complex linkage

$$d(R,S) = \frac{1}{|R||S|} \max_{x_R \in R, x_S \in S} d(x_R, x_S)$$

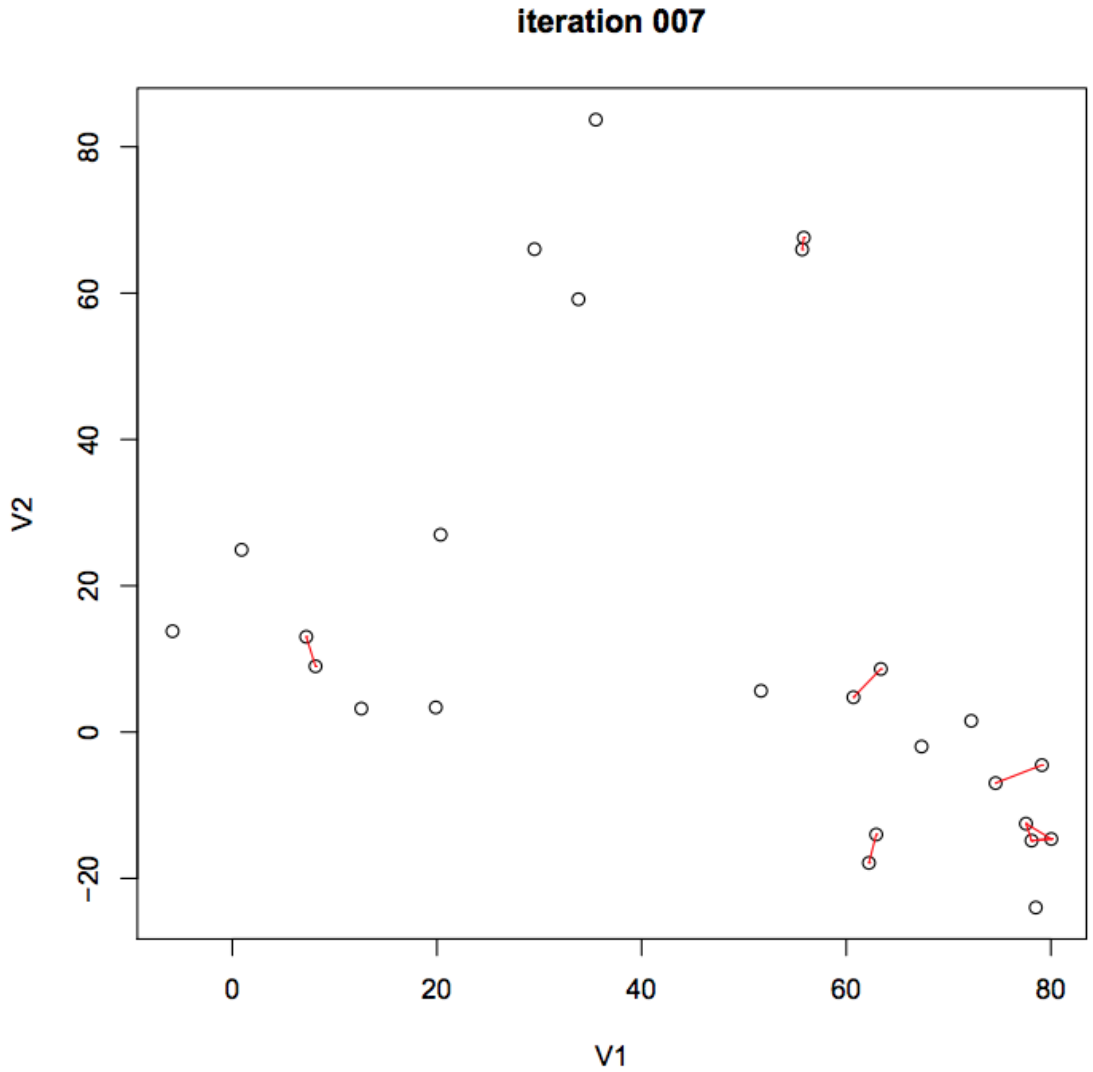
Example: Agglomerative Clustering



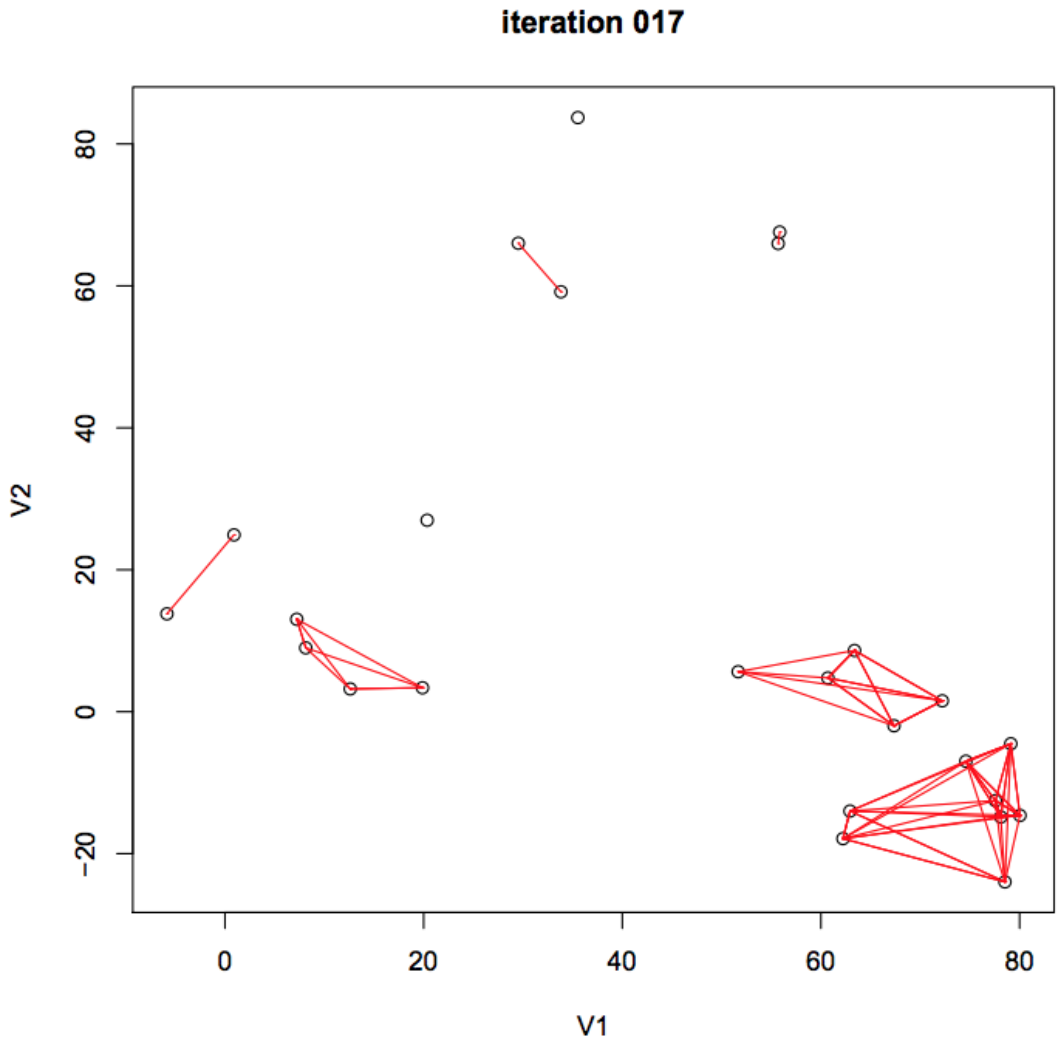
Example: Agglomerative Clustering



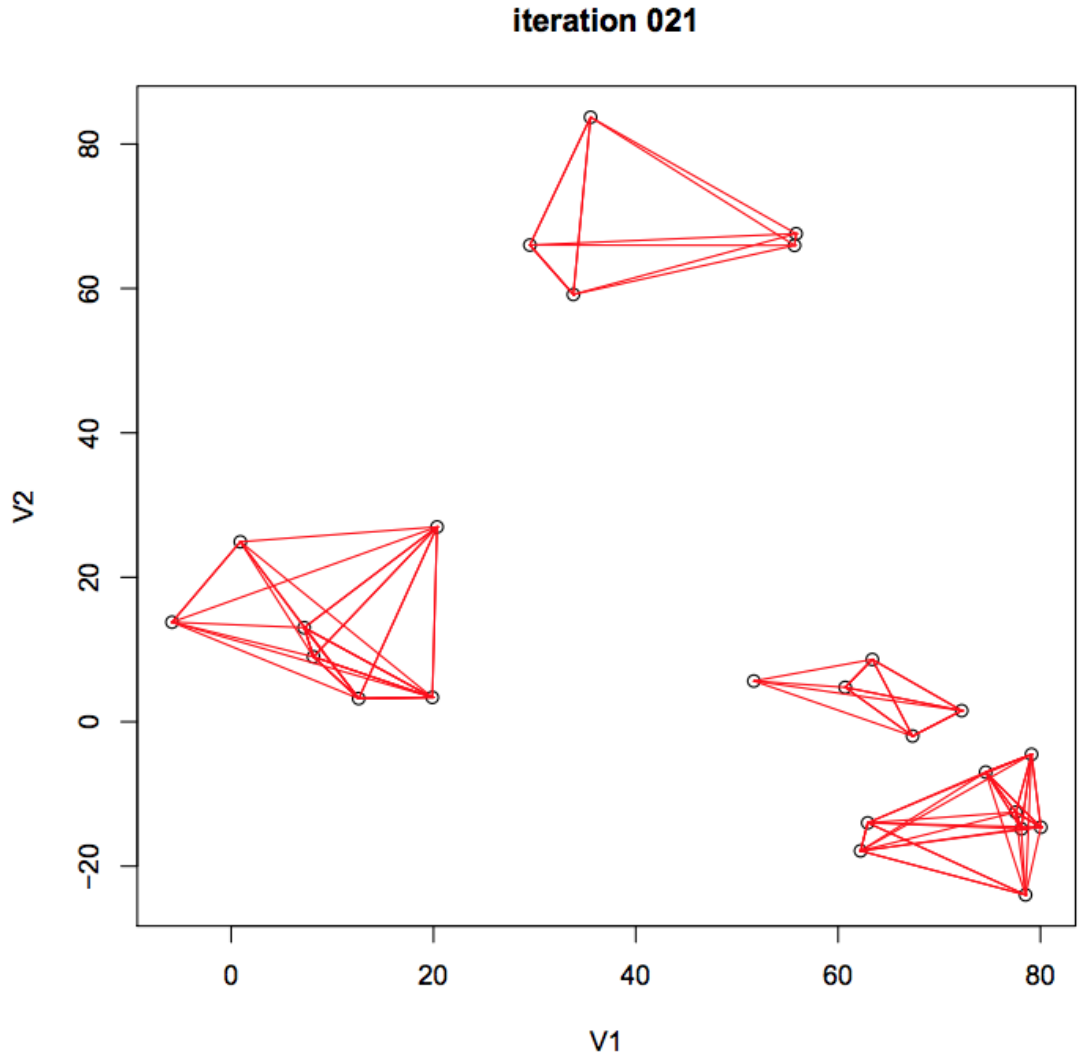
Example: Agglomerative Clustering



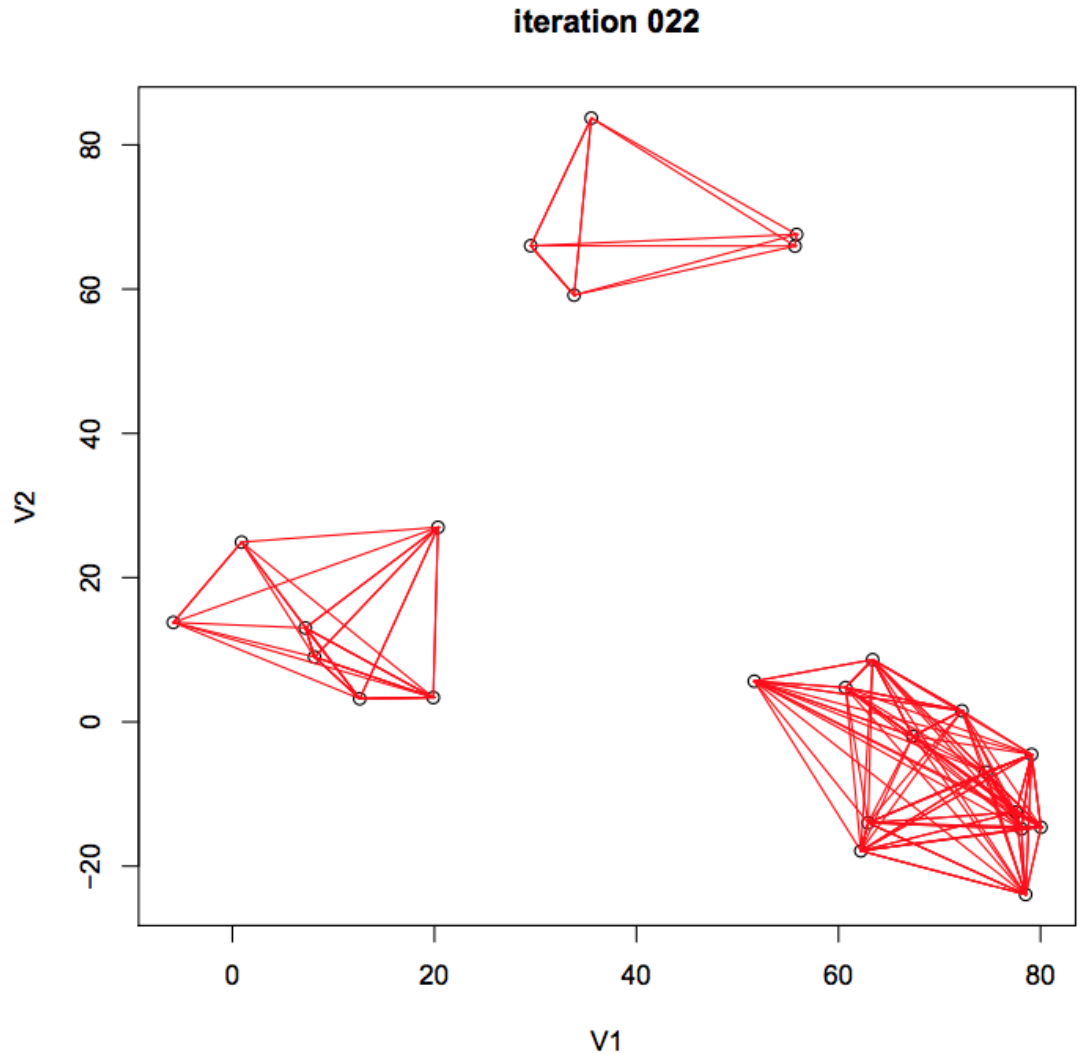
Example: Agglomerative Clustering



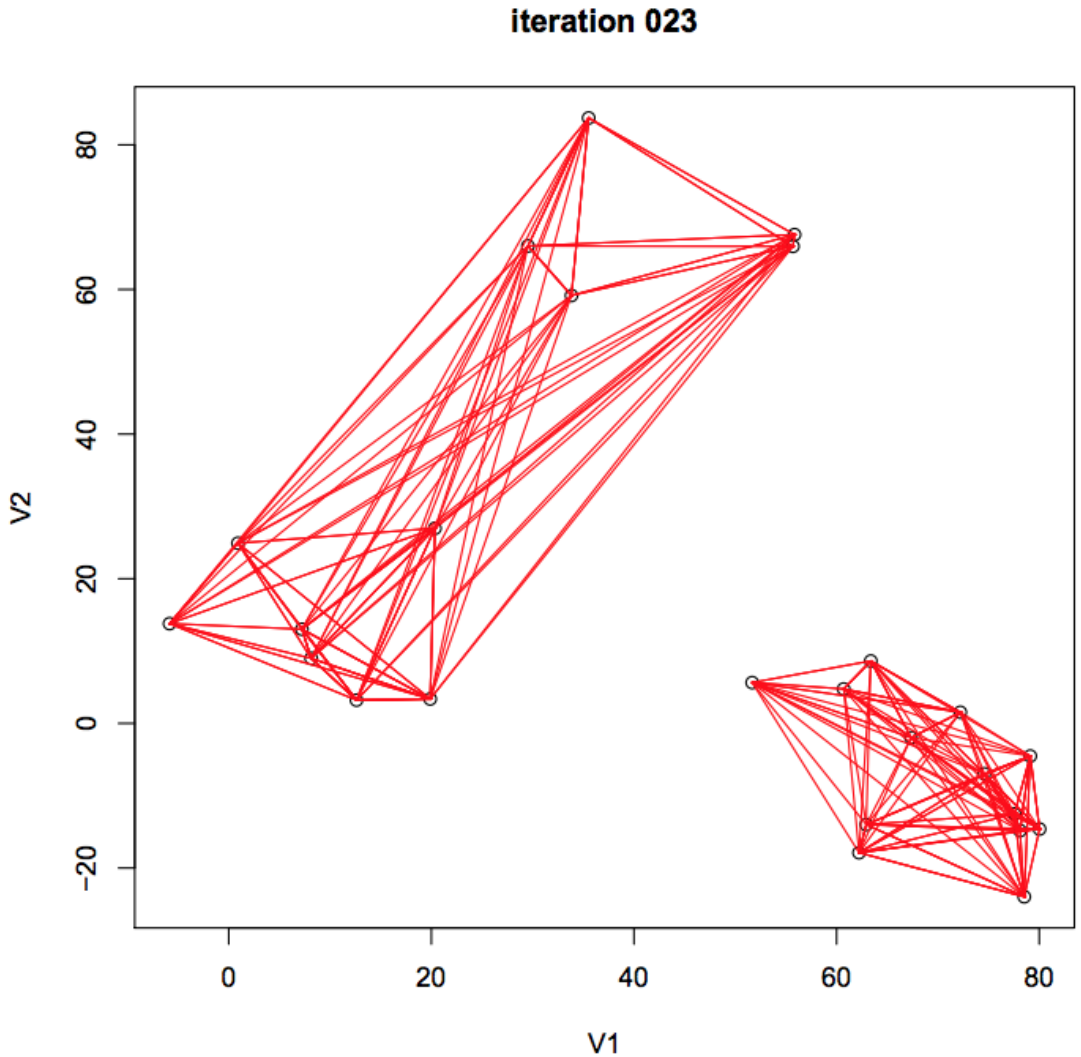
Example: Agglomerative Clustering



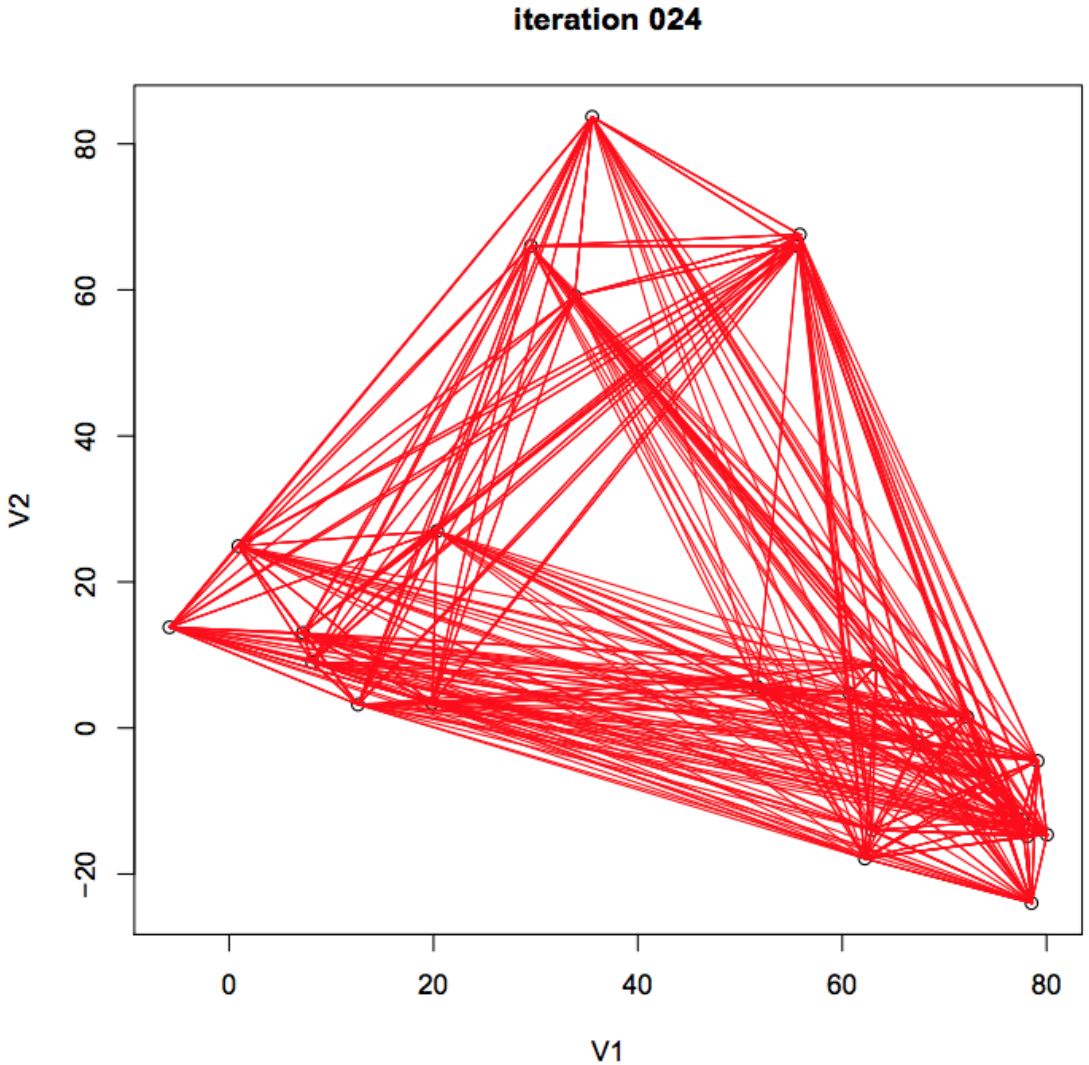
Example: Agglomerative Clustering



Example: Agglomerative Clustering



Example: Agglomerative Clustering



Flat vs Hierarchical Clustering

- Flat clustering produces a single partitioning
- Hierarchical Clustering can give different partitionings depending on the level-of-resolution we are looking at
- Flat clustering needs the number of clusters to be specified
- Hierarchical clustering does not need the number of clusters to be specified
- Flat clustering is usually more efficient run-time wise
- Hierarchical clustering can be slow (has to make several merge/split decisions)
- No clear consensus on which of the two produces better clustering

Thanks!

Q & A