Introduction to Machine Learning

CS307 --- Fall 2022

Clustering

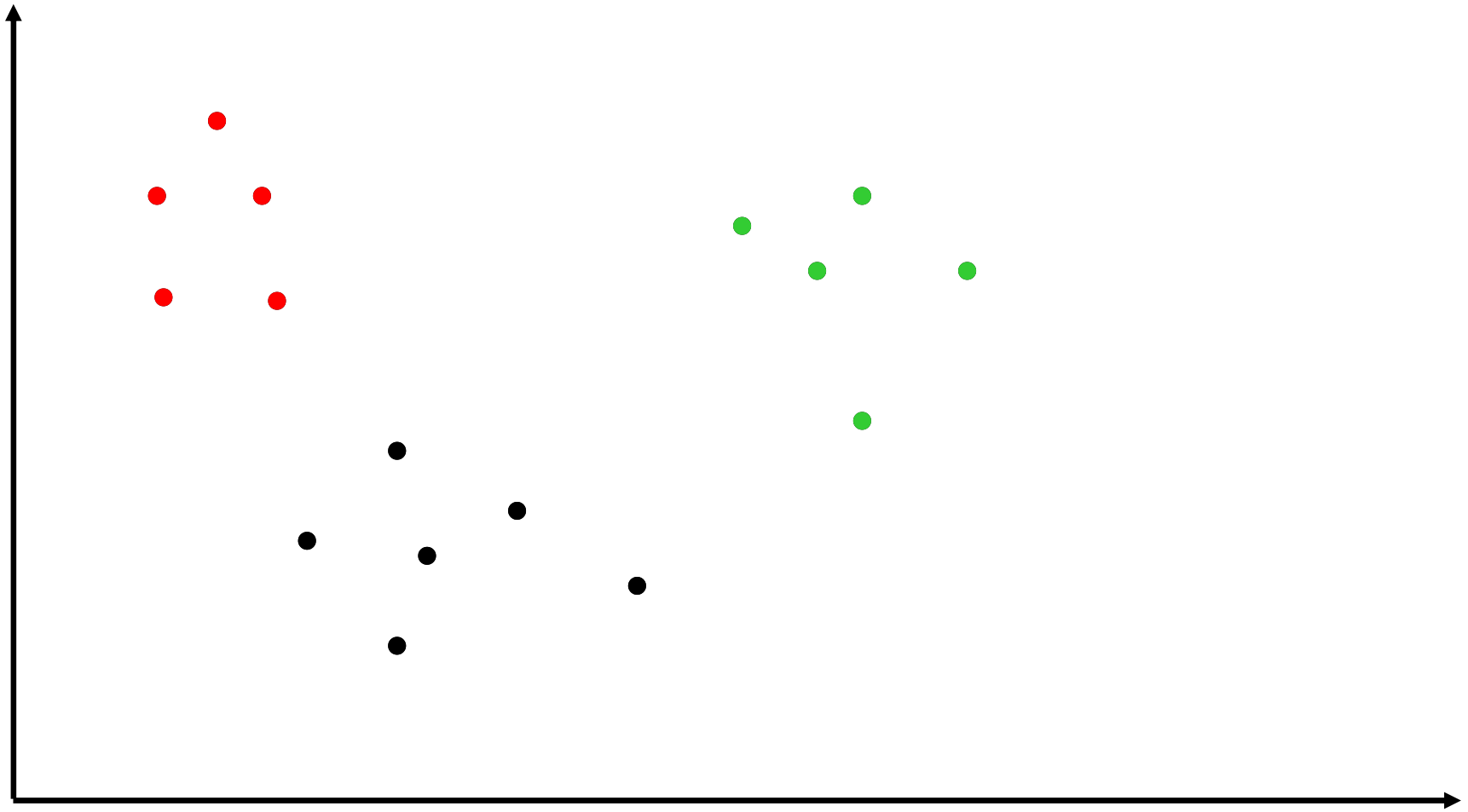
Clustering

Partition unlabeled examples into disjoint subsets of *clusters*, such that:

- Examples within a cluster are very similar
- Examples in different clusters are very different

Discover new categories in an *unsupervised* manner (no sample category labels provided).

Clustering Example



Similarity/Distance Metrics

Euclidian distance (L_2 norm):

$$L_2(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

L_1 norm:

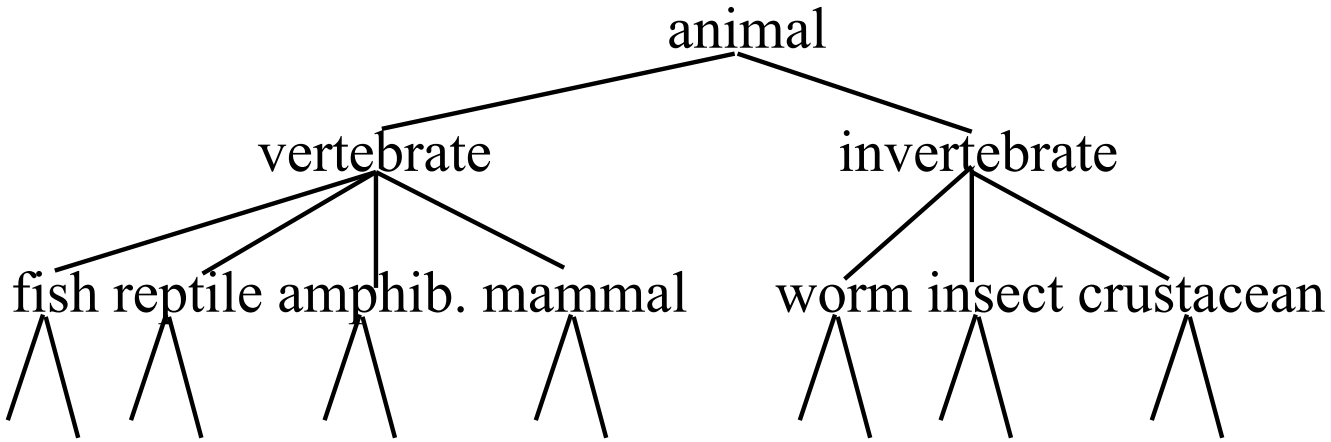
$$L_1(\vec{x}, \vec{y}) = \sum_{i=1}^{m} |x_i - y_i|$$

Cosine similarity

$$sim(\vec{x}, \vec{y}) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \bullet \vec{y}}{|\vec{x}| \cdot |\vec{y}|}$$

Hierarchical Clustering

Build a tree-based hierarchical taxonomy from a set of unlabeled examples.



Recursive application of a standard clustering algorithm can produce a hierarchical clustering.

Agglomerative vs. Divisive Clustering

Agglomerative (*bottom-up*) methods start with each example in its own cluster and iteratively combine them to form larger and larger clusters.

Divisive (*partitional, top-down*) separate all examples immediately into clusters.

Hierarchical Agglomerative Clustering (HAC)

Assumes a *similarity function* for determining the similarity of two instances.

Starts with all instances in a separate cluster and then repeatedly joins the two clusters that are most similar until there is only one cluster.

The history of merging forms a binary tree or hierarchy.

Basic algorithm:

Start with all instances in their own cluster.

Until there is only one cluster:

Among the current clusters, determine the two clusters, c_i and c_j, that are most similar.

Replace c_i and c_j with a single cluster $c_i \cup c_j$

Cluster Similarity

How to compute similarity of two clusters each possibly containing multiple instances?

- Single Link: Similarity of two most similar members.
- Complete Link: Similarity of two least similar members.
- Group Average: Average similarity between members.

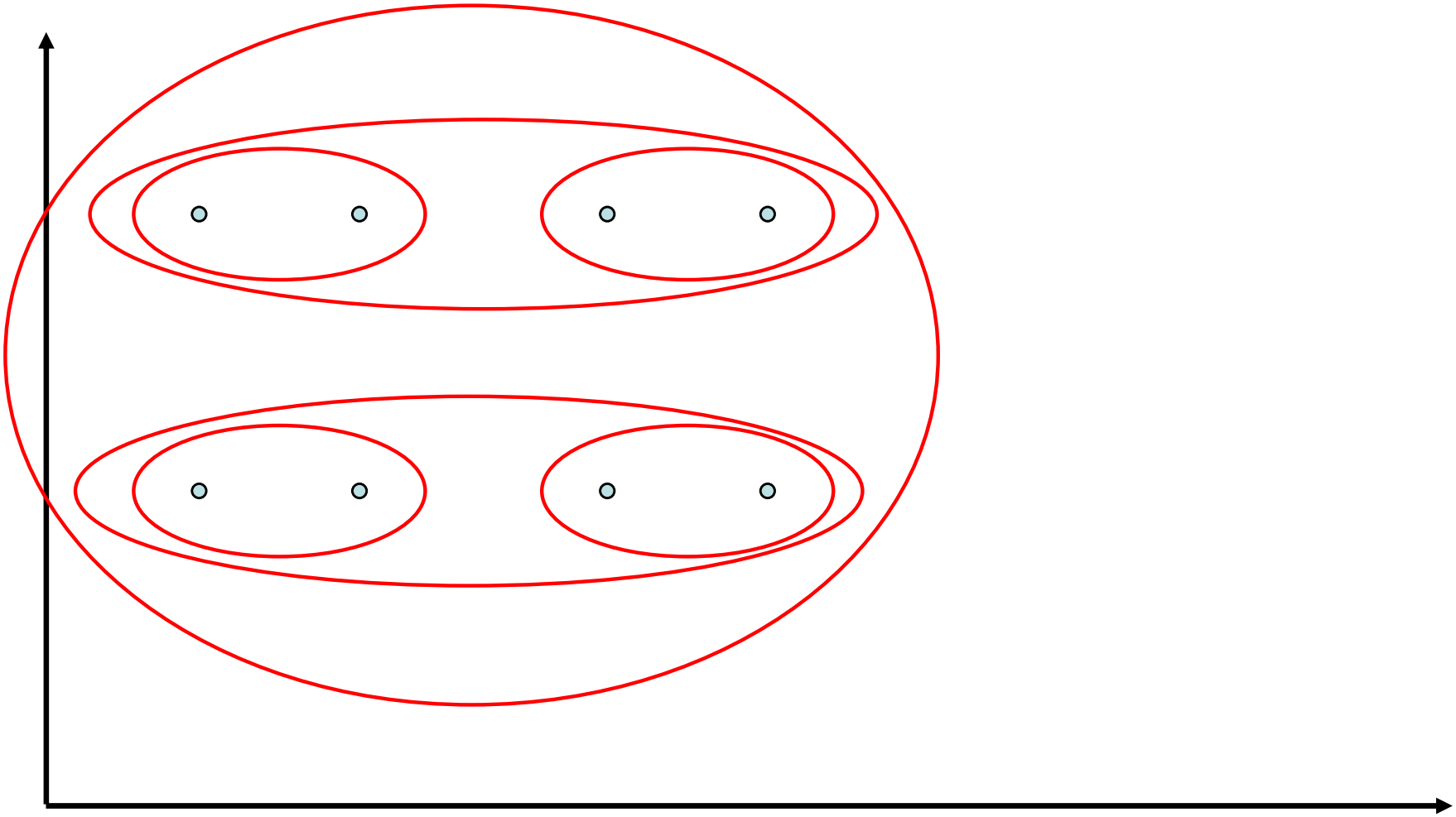
Single Link Agglomerative Clustering

Use maximum similarity of pairs:

$$sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y)$$

Can result in "straggly" (long and thin) clusters due to *chaining effect*.

Single Link Example



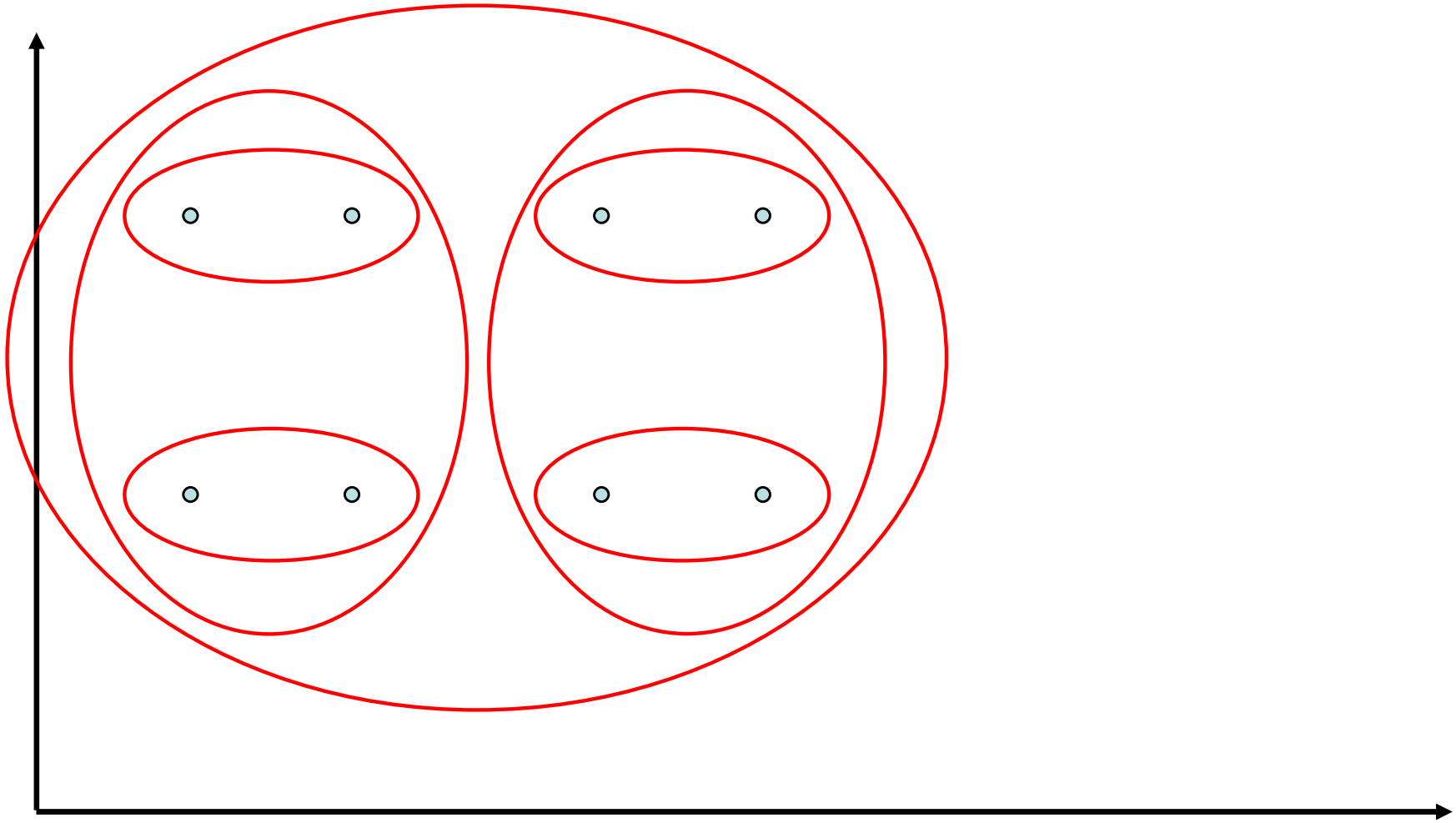
Complete Link Agglomerative Clustering

Use minimum similarity of pairs:

$$sim(c_i, c_j) = \min_{x \in c_i, y \in c_j} sim(x, y)$$

Makes more "tight," spherical clusters that are typically preferable.

Complete Link Example



Computing Cluster Similarity

After merging c_i and c_j, the similarity of the resulting cluster to any other cluster, c_k, can be computed by:

Single Link:

$$sim((c_i \cup c_j), c_k) = \max(sim(c_i, c_k), sim(c_j, c_k))$$

Complete Link:

$$sim((c_i \cup c_j), c_k) = \min(sim(c_i, c_k), sim(c_j, c_k))$$

Group Average Agglomerative Clustering

Use average similarity across all pairs within the merged cluster to measure the similarity of two clusters.

$$sim(c_i, c_j) = \frac{1}{\left|c_i \cup c_j\right|\left(\left|c_i \cup c_j\right| - 1\right)} \sum_{\vec{x} \in (c_i \cup c_j)} \sum_{\vec{y} \in (c_i \cup c_j): \vec{y} \neq \vec{x}} sim(\vec{x}, \vec{y})$$

Compromise between single and complete link.

Averaged across all ordered pairs in the merged cluster instead of unordered pairs *between* the two clusters

Non-Hierarchical Clustering

Types of non-hierarchical clustering

- **Hard clustering (e.g., K-means clustering)**
- **Soft clustering (e.g., EM)**

Typically must provide the number of desired clusters, *k*.

Randomly choose *k* instances as *seeds*, one per cluster.

Form initial clusters based on these seeds.

Iterate, repeatedly reallocating instances to different clusters to improve the overall clustering.

Stop when clustering converges or after a fixed number of iterations.

Clustering Criterion

Evaluation function that assigns a (usually real-valued) value to a clustering

- Clustering criterion typically a function of
  - within-cluster similarity and
  - between-cluster dissimilarity

Optimization

- Find clustering that maximizes the criterion
  - Global optimization (often intractable)
  - Greedy search

Centroid-Based Clustering

Assumes instances are real-valued vectors.

Clusters based on *centroids*, *center of gravity*, or mean of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|}\sum_{\vec{x}\in c}\vec{x}$$

Reassignment of instances to clusters is based on distance to the current cluster centroids.

K-Means Algorithm

Let d be the distance measure between instances.

Select k random instances $\{s_1, s_2, \ldots s_k\}$ as seeds.

Until clustering converges or other stopping criterion:

For each instance x_i:

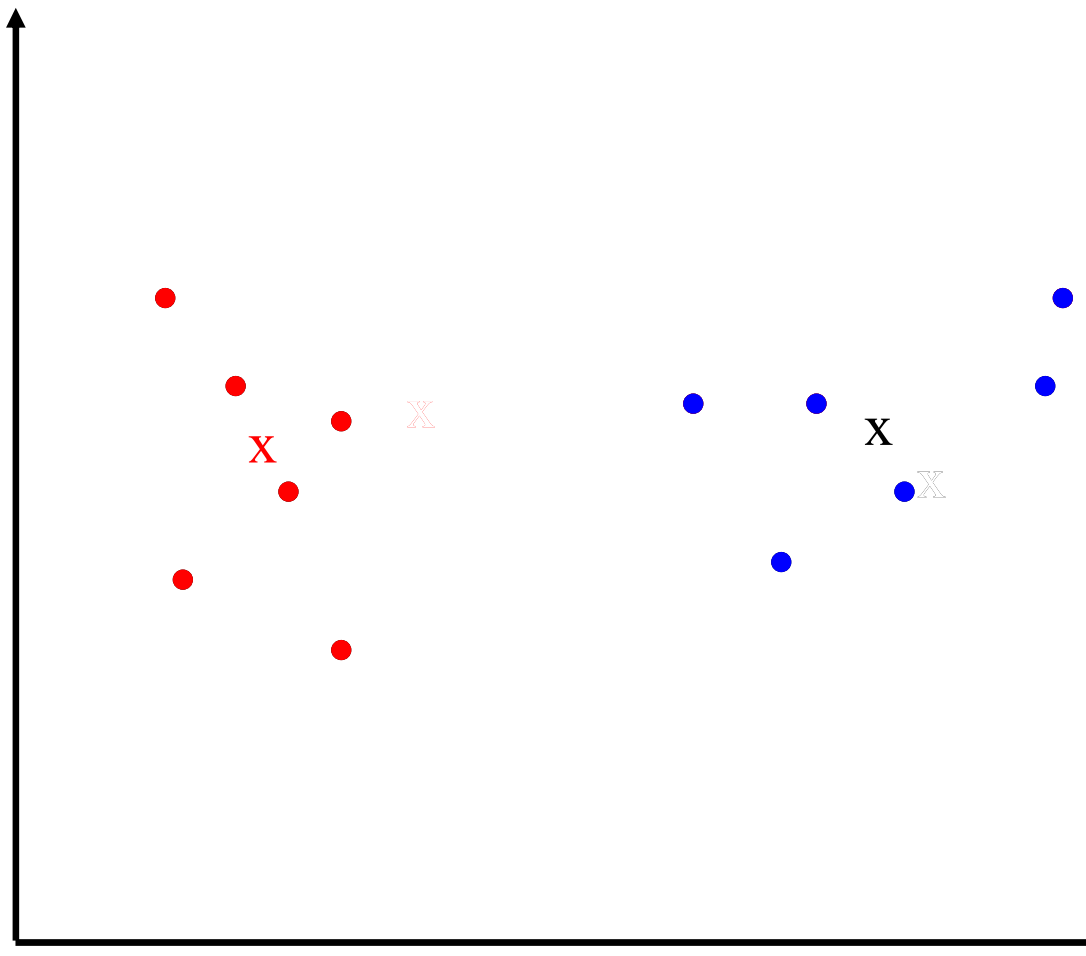
Assign x_i to the cluster c_j such that $d(x_i, s_j)$ is minimal.

(*Update the seeds to the centroid of each cluster*)

For each cluster c_j

$s_j = \mu(c_j)$

K Means Example
(K=2)



Pick seeds

Reassign clusters

Compute centroids

Reasssign clusters

Compute centroids

Reassign clusters

Converged!

Distortion

Given..

•an encoder function: ENCODE : $\Re^m \rightarrow [1..k]$

•a decoder function: DECODE : $[1..k] \rightarrow \Re^m$

Define...

$$\text{Distortion} = \sum_{i=1}^{R} \left(\mathbf{x}_i - \text{DECODE}[\text{ENCODE}(\mathbf{x}_i)]\right)^2$$

Distortion

Given..

•an encoder function: ENCODE : $\Re^m \rightarrow [1..k]$

•a decoder function: DECODE : $[1..k] \rightarrow \Re^m$

Define...

$$\text{Distortion} = \sum_{i=1}^{R} \left(\mathbf{x}_i - \text{DECODE}[\text{ENCODE}(\mathbf{x}_i)]\right)^2$$

We may as well write

$$\text{DECODE}[j] = \mathbf{c}_j$$

$$\text{so} \quad \text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

The Minimal Distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_k$ have when distortion is minimized?

The Minimal Distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_k$ have when distortion is minimized?

(1) $\mathbf{x}_i$ must be encoded by its nearest center ….why?

$$\mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)} = \underset{\mathbf{c}_j \in \{\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_k\}}{\arg\min} (\mathbf{x}_i - \mathbf{c}_j)^2$$

..at the minimal distortion

The Minimal Distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_k$ have when distortion is minimized?

(2) The partial derivative of Distortion with respect to each center location must be zero.

(2) The partial derivative of Distortion with respect to each center location must be zero.

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

$$= \sum_{j=1}^{k} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

OwnedBy($\boldsymbol{c}_j$) = the set of records owned by Center $\boldsymbol{c}_j$.

$$\frac{\partial \text{Distortion}}{\partial \mathbf{c}_j} = \frac{\partial}{\partial \mathbf{c}_j} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

$$= -2 \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)$$

$$= 0 \text{ (for a minimum)}$$

(2) The partial derivative of Distortion with respect to each center location must be zero.

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

$$= \sum_{j=1}^{k} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

OwnedBy($\boldsymbol{c}_j$) = the set of records owned by Center $\boldsymbol{c}_j$.

$$\frac{\partial \text{Distortion}}{\partial \mathbf{c}_j} = \frac{\partial}{\partial \mathbf{c}_j} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)^2$$

$$= -2 \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} (\mathbf{x}_i - \mathbf{c}_j)$$

$$= 0 \text{ (for a minimum)}$$

Thus, at a minimum: $\mathbf{c}_j = \frac{1}{|\text{OwnedBy}(\mathbf{c}_j)|} \sum_{i \in \text{OwnedBy}(\mathbf{c}_j)} \mathbf{x}_i$

At the Minimum Distortion

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties must centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_k$ have when distortion is minimized?

(1) $\mathbf{x}_i$ must be encoded by its nearest center

(2) The partial derivative of Distortion with respect to each center location must be zero.

Improving a Suboptimal Configuration

$$\text{Distortion} = \sum_{i=1}^{R} (\mathbf{x}_i - \mathbf{c}_{\text{ENCODE}(\mathbf{x}_i)})^2$$

What properties can be changed for centers $\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_k$ have when distortion is not minimized?

(1) Change encoding so that $\mathbf{x}_i$ is encoded by its nearest center

(2) Set each Center to the centroid of points it owns.

There's no point applying either operation twice in succession.

But it can be profitable to alternate.

…And that's K-means!

Will we find the optimal configuration?

- Not necessarily.
- Can you invent a configuration that has converged, but does not have the minimum distortion?

Seed Choice

Results can vary based on random seed selection.

Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.

Select good seeds using a heuristic or the results of another method.

Trying to find good optima

- Idea 1: Be careful about where you start
- Idea 2: Do many runs of k-means, each from a different random start configuration
- Many other ideas floating around.

Common uses of K-means

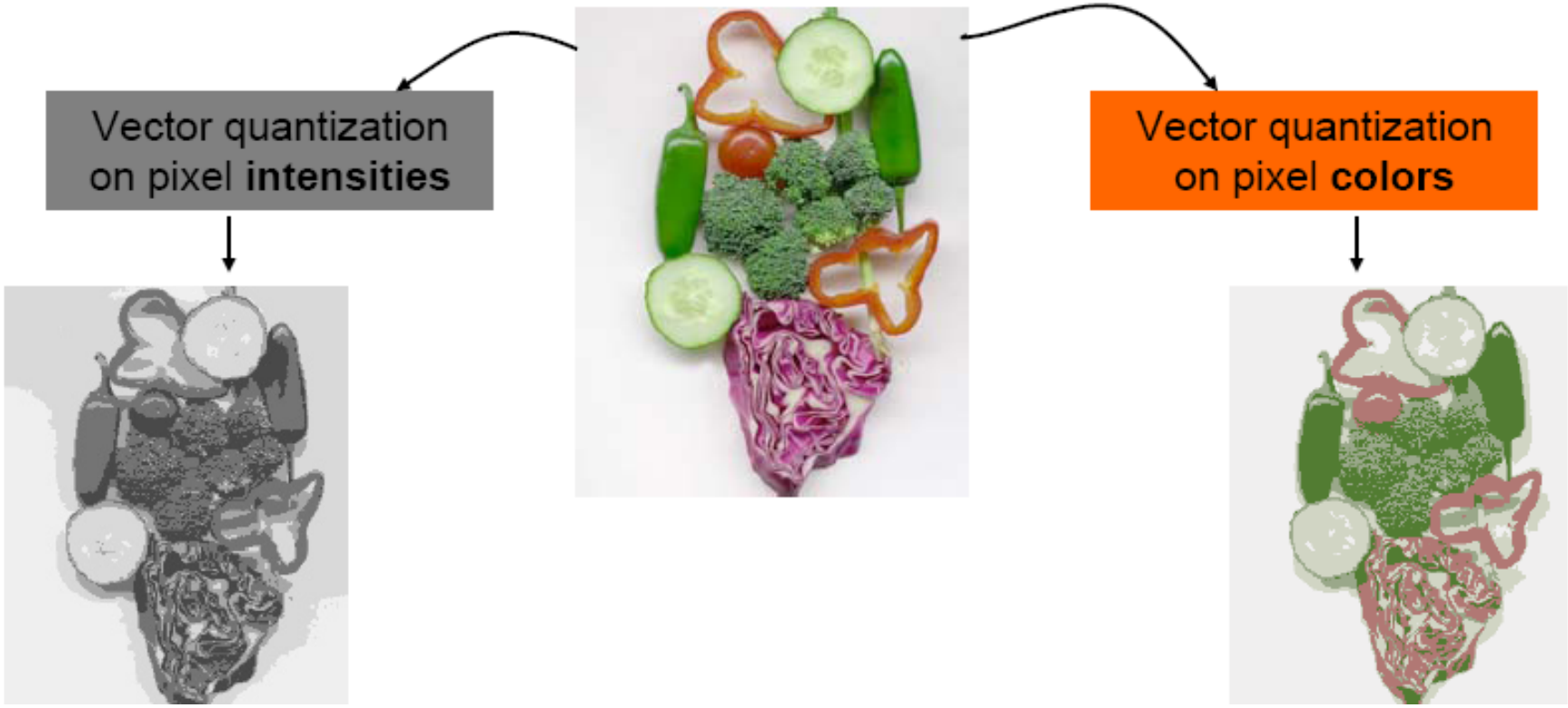
- Often used as an exploratory data analysis tool
- In one-dimension, a good way to quantize real-valued variables into k non-uniform buckets
- Used on acoustic data in speech understanding to convert waveforms into one of k categories (i.e. Vector Quantization)

Example: Image Segmentation

- Once K-means is performed, the resulting cluster centers can be thought of as *K labeled data points* for 1-NN on the entire training set, such that each data point is labeled with its nearest center. This is called **Vector Quantization**.

Example: Image Segmentation

- Once K-means is performed, the resulting cluster centers can be thought of as *K labeled data points* for 1-NN on the entire training set, such that each data point is labeled with its nearest center. This is called **Vector Quantization**.



Soft Clustering

Clustering typically assumes that each instance is given a "hard" assignment to exactly one cluster.

Does not allow uncertainty in class membership or for an instance to belong to more than one cluster.

Soft clustering gives probabilities that an instance belongs to each of a set of clusters.

Each instance is assigned a probability distribution across a set of discovered categories (probabilities of all categories must sum to 1).

The EM Algorithm

Iterative method for learning probabilistic categorization model from unsupervised data.

Initially assume random assignment of examples to categories.

Learn an initial probabilistic model by estimating model parameters θ from this randomly labeled data.

Iterate following two steps until convergence:

Expectation (E-step): Compute $P(c_i \mid E)$ for each example given the current model, and probabilistically re-label the examples based on these posterior probability estimates.

Maximization (M-step): Re-estimate the model parameters, θ, from the probabilistically re-labeled data.