Lecture 20: Clustering

- Wrap-up of neural nets (from last lecture
- Introduction to unsupervised learning
- K-means clustering

Unsupervised learning

- In supervised learning, data is in the form of pairs $\langle \mathbf{x}, y \rangle$, where $y = f(\mathbf{x})$, and the goal is to approximate f well.
- In *unsupervised learning*, the data just contains $\mathbf{x}$!
- Goal is to "summarize" or find "patterns" or "structure" in the data
- A variety of problems and uses:
  - Clustering: "Flat" clustering or partitioning, hierarchical clustering
  - Density estimation
  - Dimensionality reduction, for: visualization, compression, pre-processing
- The definition of "ground truth" is often missing: no clear error function, or at least many reasonable alternatives
- Often useful in exploratory data analysis, and as a pre-processing step for supervised learning

What is clustering?

- Clustering is grouping similar objects together.
  - To establish prototypes, or detect outliers.
  - To simplify data for further analysis/learning.
  - To visualize data (in conjunction with dimensionality reduction)
- Clusterings are usually not "right" or "wrong" – different clusterings/clustering criteria can reveal different things about the data.
- Clustering algorithms:
  - Employ some notion of distance between objects
  - Have an explicit or implicit criterion defining what a good cluster is
  - Heuristically optimize that criterion to determine the clustering
- Some clustering criteria and algorithms have natural probabilistic interpretations

K-means clustering

- One of the most commonly-used clustering algorithms, because it is easy to implement and quick to run.
- Assumes the objects (instances) to be clustered are n-dimensional vectors, $\mathbf{x}_i$.
- Uses a distance measure between the instances (typically Euclidian distance)
- The goal is to *partition* the data into K disjoint subsets

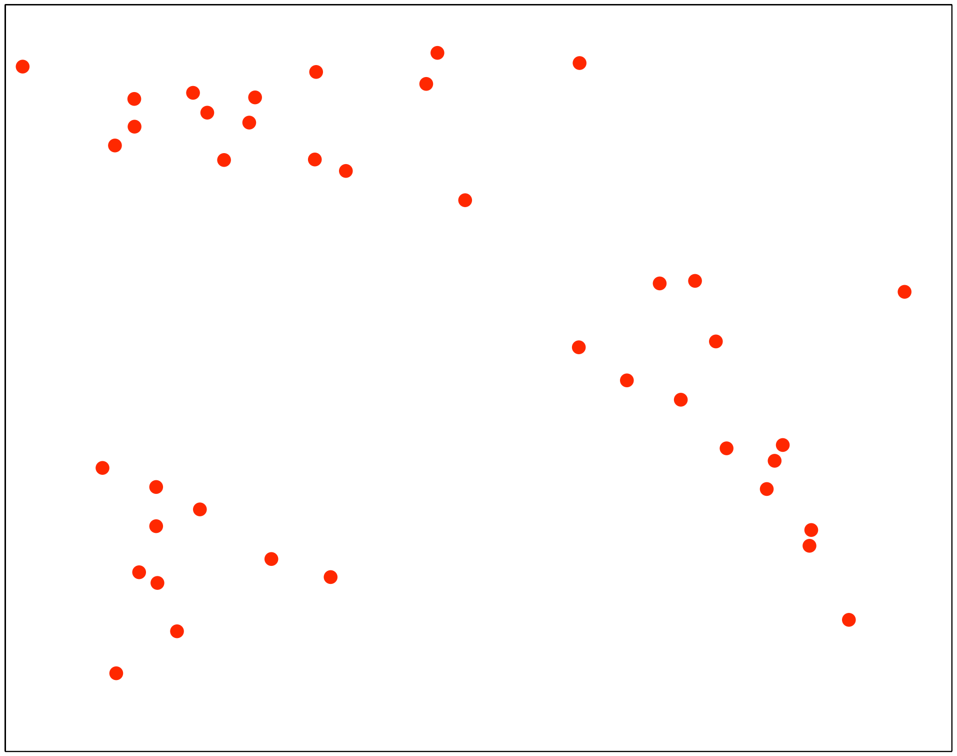
K-means clustering with real-valued data

- Inputs:
  - A set of n-dimensional real vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$.
  - K, the desired number of clusters.
- Output: A mapping of the vectors into K clusters (disjoint subsets), $C : \{1, \ldots, m\} \mapsto \{1, \ldots, K\}$.

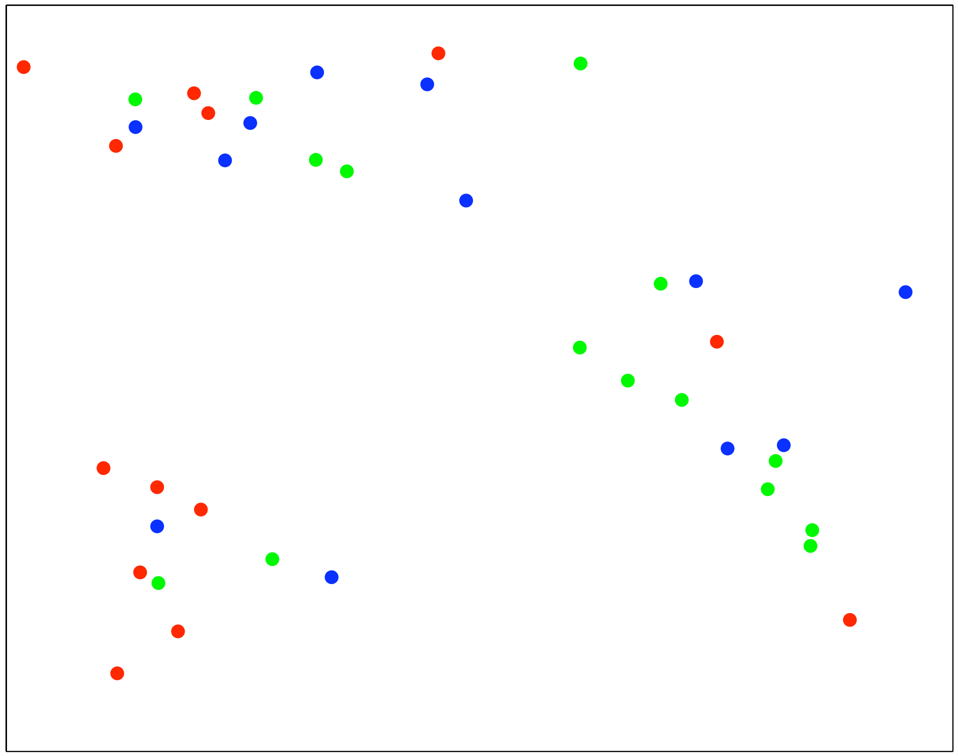
1. Initialize C randomly.
2. Repeat
   (a) Compute the *centroid* of each cluster (the mean of all the instances in the cluster)
   (b) Reassign each instance to the cluster with closest centroid

   until C stops changing.

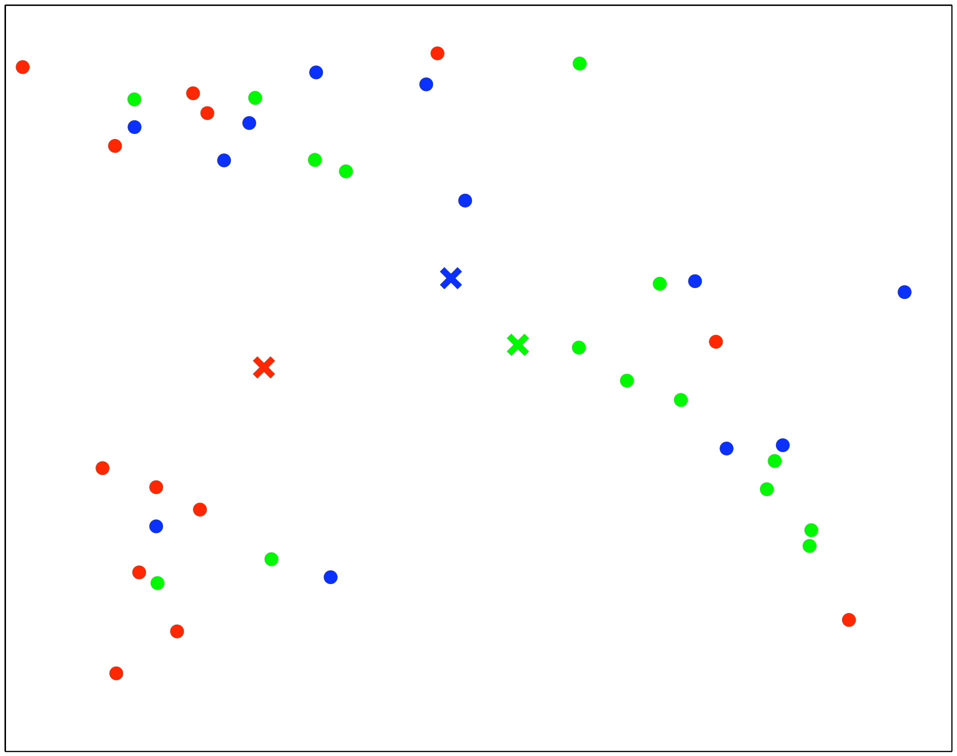
Example: initial data



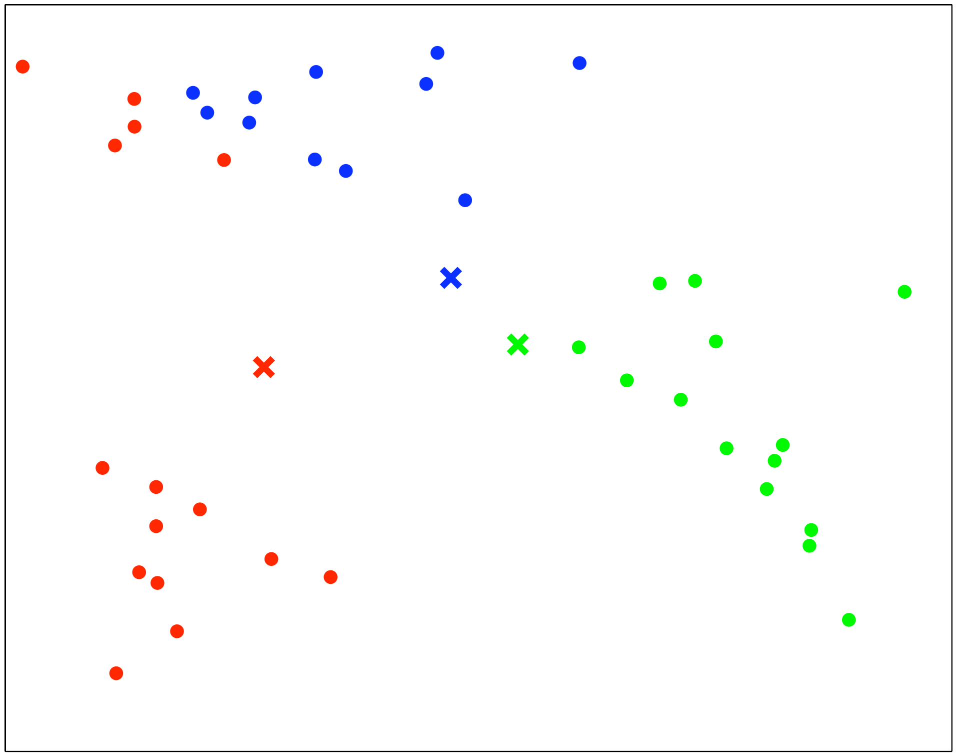
Example: assign into 3 clusters randomly



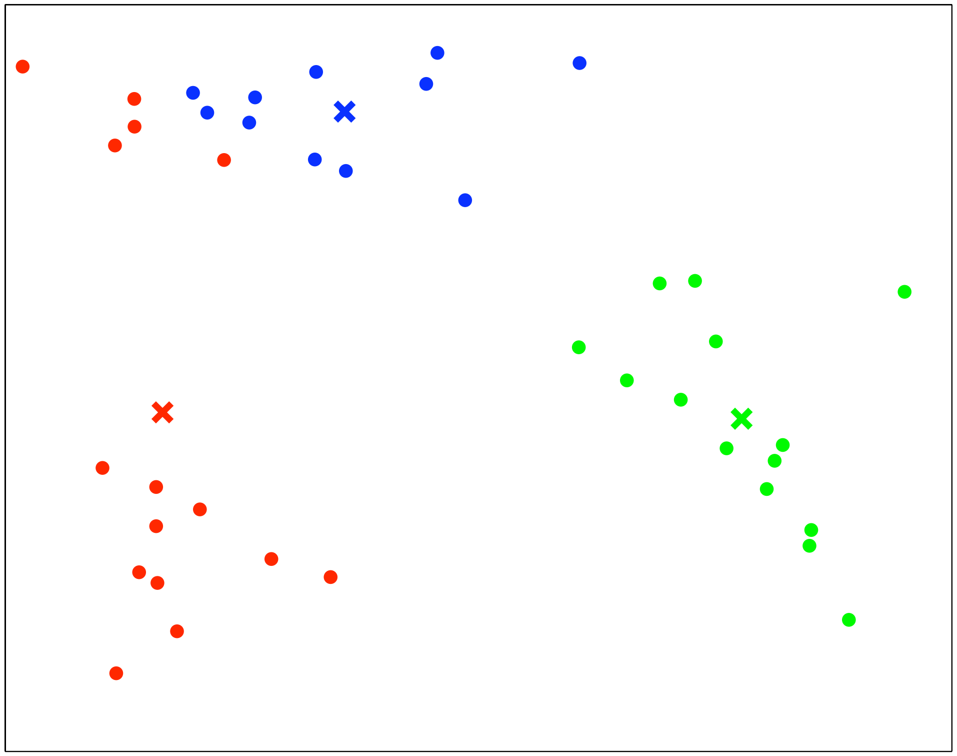
Example: compute centroids



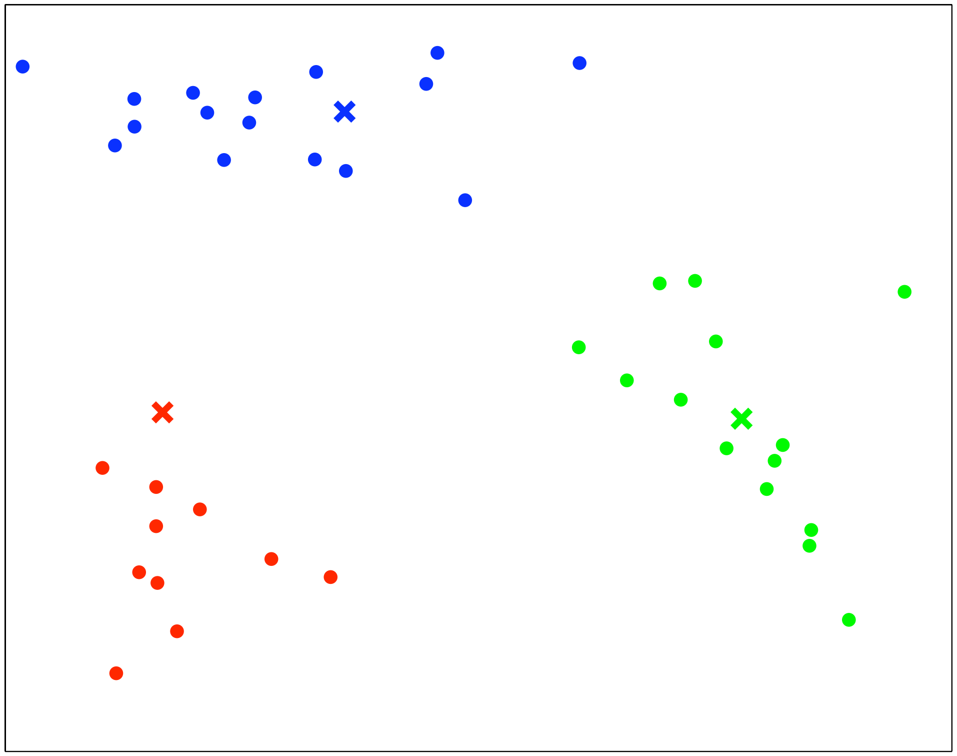
Example: reassign clusters



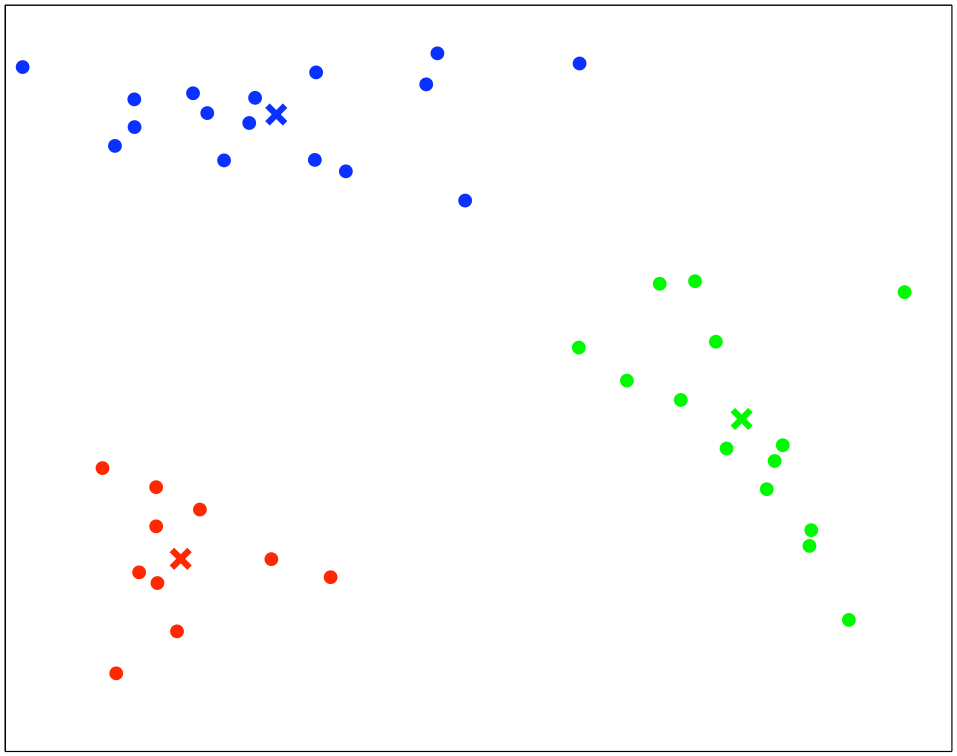
Example: recompute centroids



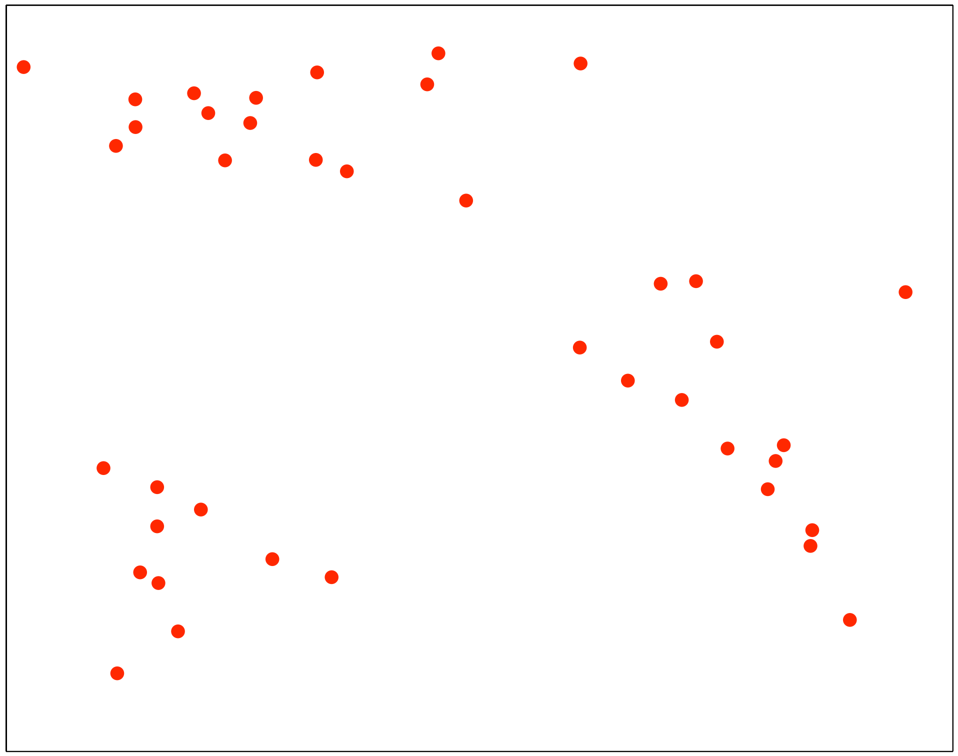
Example: reassign clusters



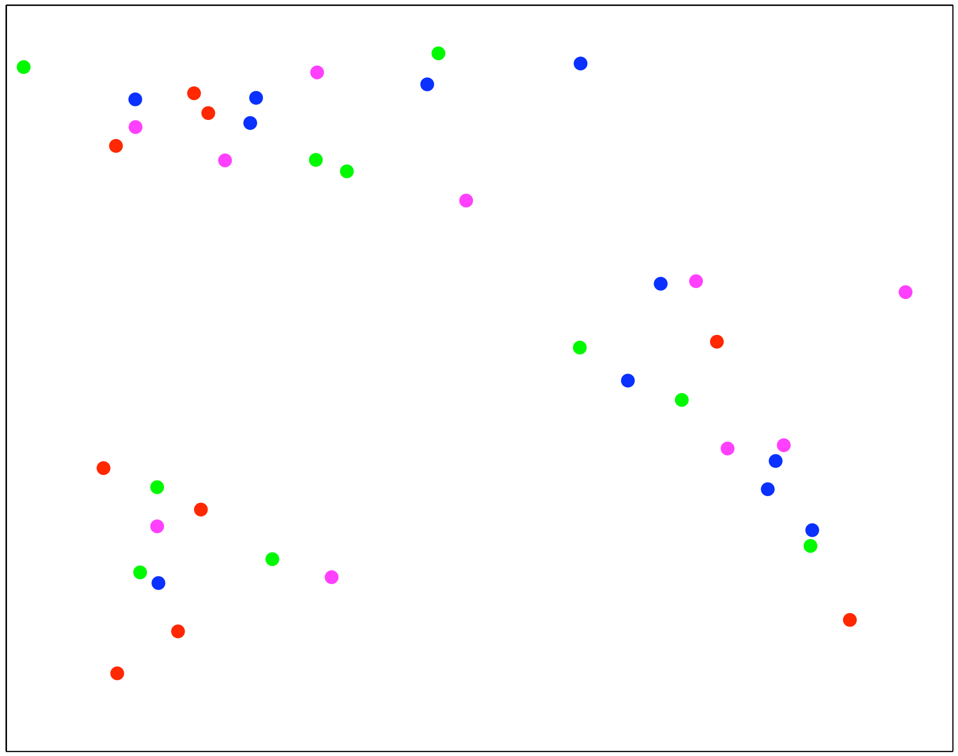
Example: recompute centroids – done!



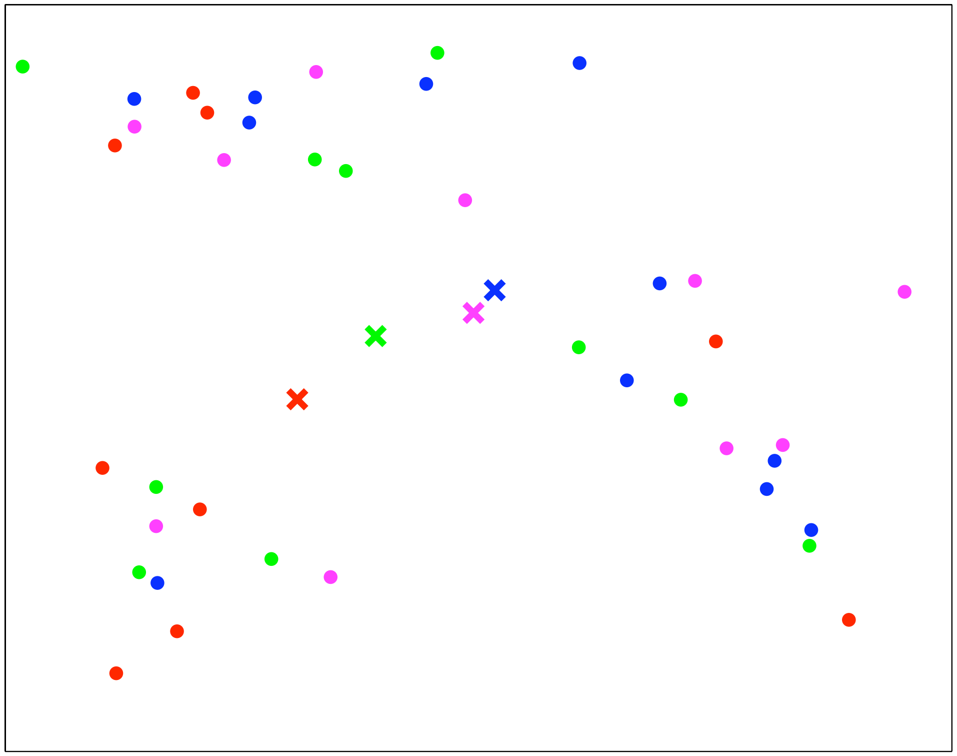
What if we do not know the right number of clusters?



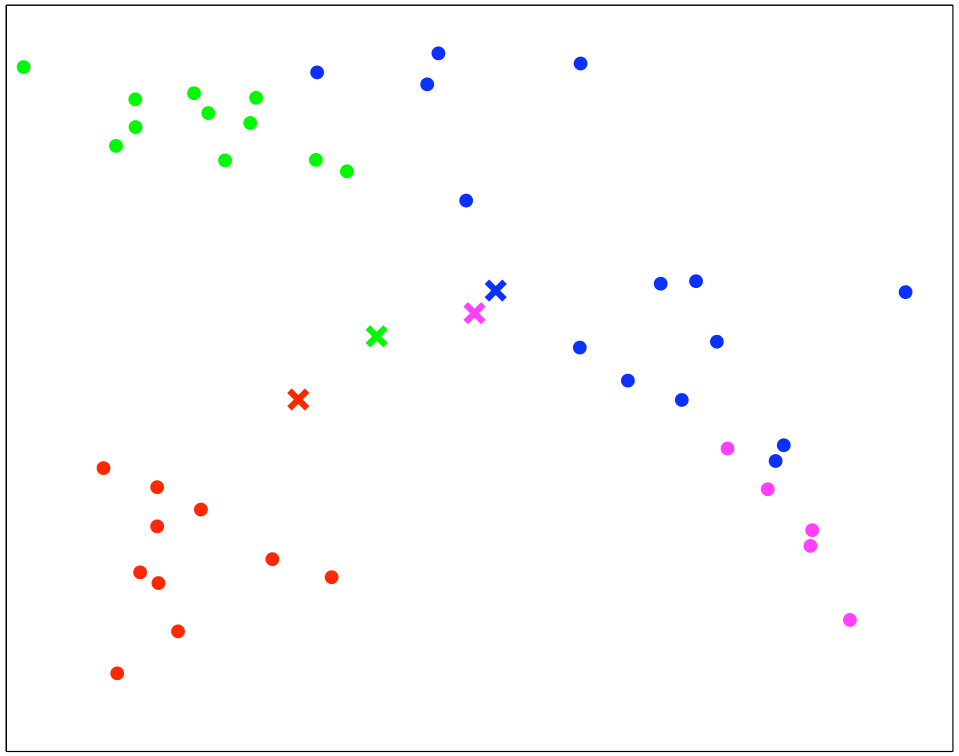
Example: assign into 4 clusters randomly



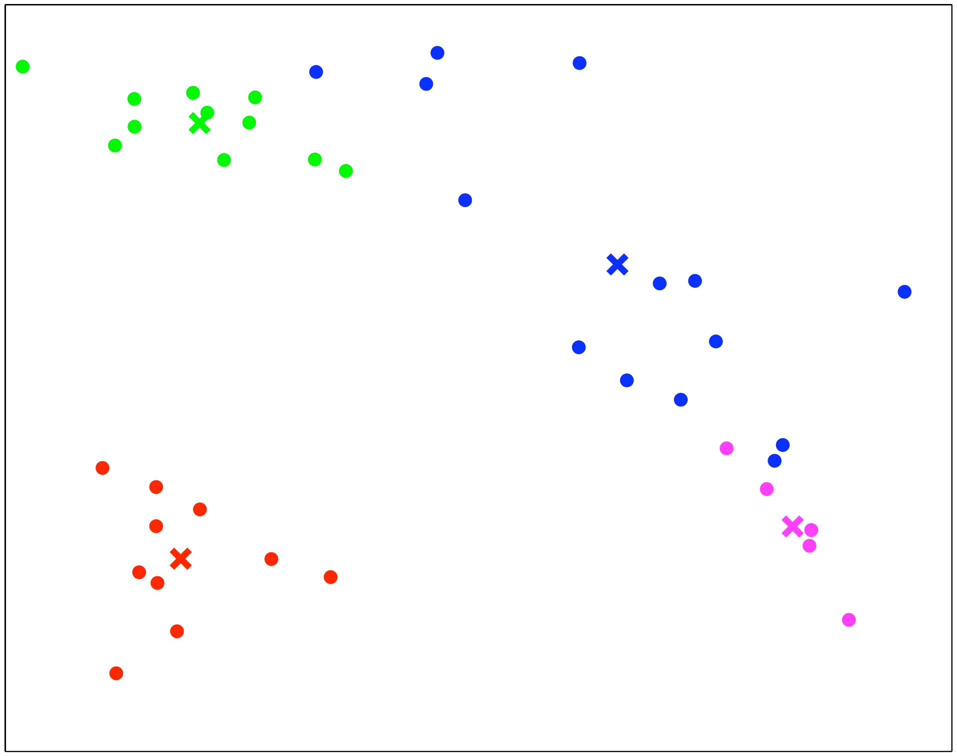
Example: compute centroids



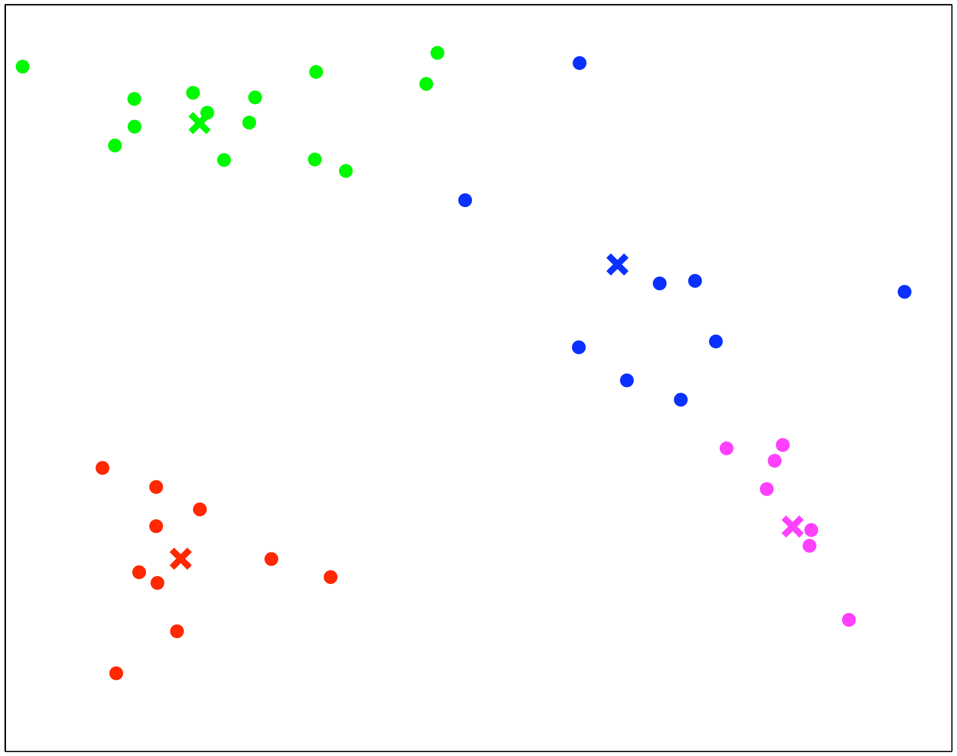
Example: reassign clusters



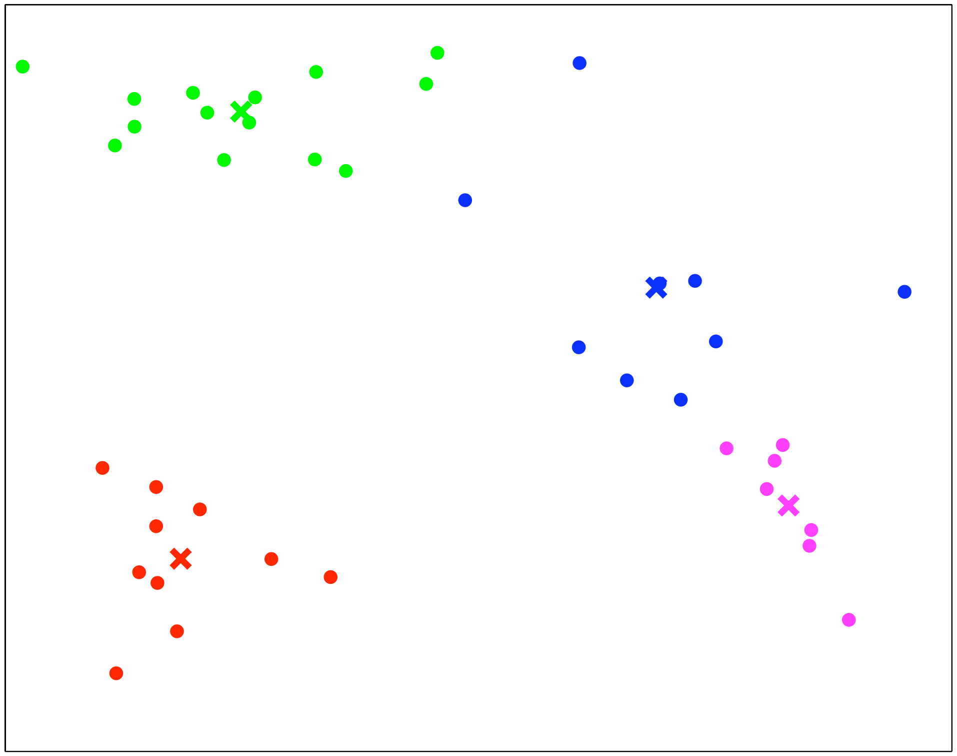
Example: recompute centroids



Example: reassign clusters



Example: recompute centroids – done!



Assessing the quality of the clustering

- If the clustering is used as a pre-processing step for supervised learning, measure the performance of the supervised learner
- Measure the "tightness" of the clusters: points in the same cluster should be close together, points in different clusters should be far apart
- Tightness can be measured by:
  - the minimum distance between points in different clusters
  - the maximum distance between points in the same cluster
  - the average distance between points in the same cluster (can be normalized by average distance between clusters)
- Several different "figures of merit" have been proposed (e.g. silhouette method)
- Problem: these measures usually favour large numbers of clusters, so some form of regularization or description length penalty is necessary

Typical applications of K-means clustering

- Pre-processing step for supervised learning
- Data inspection/experimental data analysis
- Discretizing real-valued variables in non-uniform buckets.
- Data compression

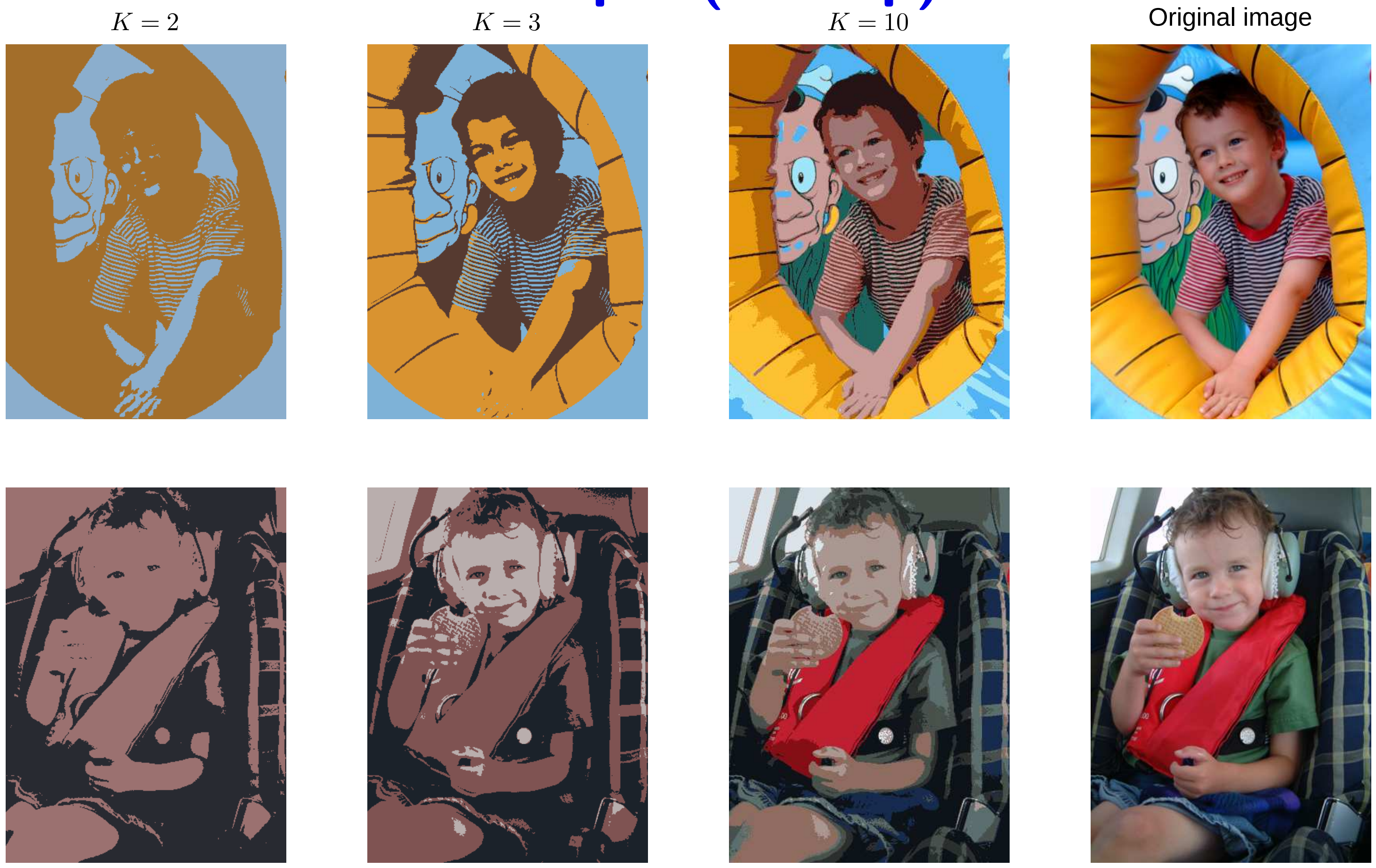
Example application: Color quantization

- Suppose you have an image stored with 24 bits per pixel
- You want to compress it so that you use only 8 bits per pixel (256 colours)
- You want the compressed image to look *as similar as possible* to the original image

$\Rightarrow$ Perform $K-$means clustering on the original set of colour vectors with $K = 256$ colours.

- Cluster centres (rounded to integer intensities) form the entries in the 256-colour colormap
- Each pixel represented by 8-bit index into colormap

Example (Bishop)

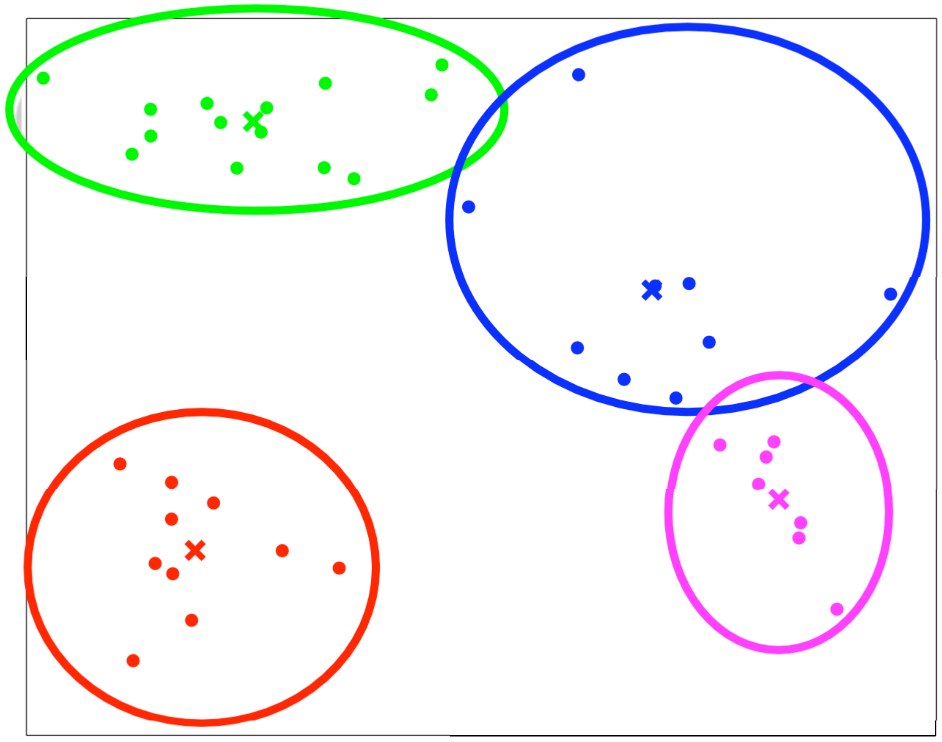


More generally: Vector quantization with Euclidean loss

- Suppose we want to send all the instances over a communication channel
- In order to compress the message, we cluster the data and *encode each instance as the centre of the cluster* to which it belongs
- The *reconstruction error* for real-valued data can be measured as Euclidian distance between the true value and its encoding
- An optimal K-means clustering minimizes the reconstruction error among all possible codings of the same type

Why the sum of squared Euclidean distances?

Subjective reason: It produces nice, round clusters.



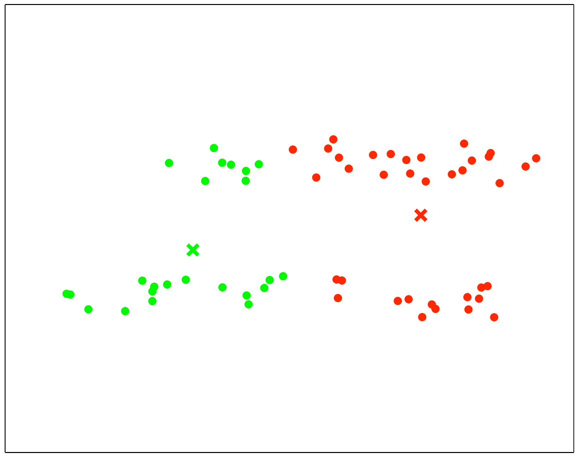
Why the sum of squared Euclidean distances?

Objective reason: Maximum Likelihood Principle

- Suppose the data really does divide into K clusters.
- Suppose the data in each cluster is generated by independent samples from a multivariate Gaussian distribution, where:
  - The mean of the Gaussian is the centroid of the cluster
  - The covariance matrix is of the form $\sigma^2 I$
- Then the probability of the data is highest when the sum of squared Euclidean distances is smallest.

Why *not* the sum of squared Euclidean distances?

1. It produces nice round clusters!



2. Differently scaled axes can dramatically affect results.

3. There may be symbolic attributes, which have to be treated differently

Questions

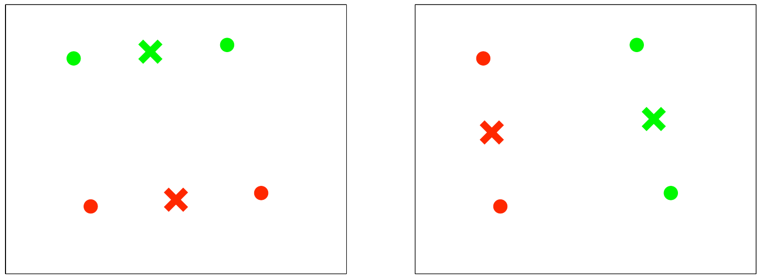
- Will K-means terminate (assuming for concreteness Euclidean distance function)?
- Will it always find the same answer?
- How should we choose the initial cluster centres?
- Can we automatically choose the number of centres?

Does K-means clustering terminate?

- For given data $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ and a clustering $\mathbf{C}$, consider the sum of the squared Euclidian distance between each vector and the center of its cluster:
$$J = \sum_{i=1}^{m} \left\|\mathbf{x}_i - \mu_{C(i)}\right\|^2 ,$$
where $\mu_{C(i)}$ denotes the centroid of the cluster containing $\mathbf{x}_i$.
- There are finitely many possible clusterings: at most K^m.
- Each time we reassign a vector to a cluster with a nearer centroid, J decreases (by definition)
- Each time we recompute the centroids of each cluster, J decreases (or stays the same), because we find the maximum likelihood estimates of the means, given the current cluster assignments
- Thus, the algorithm must terminate.

Does K-means always find the same answer?

- K-means is a version of coordinate descent, where the parameters are the assignments of points to clusters, C_i and the cluster centre coordinates, μ_i
- The error function (sum of squared Euclidean distances from vectors to their cluster centroid) has many local minima!
- The solution found is *locally optimal*, but *not globally optimal*
- Because the solution depends on the initial assignment of instances to clusters, random restarts will give different solutions



$J = 0.22870$ $\qquad$ $J = 0.3088$

Finding good initial configurations

- The initial configuration can influence the final clustering
- Assigning each item to a random cluster in $\{1, \ldots, K\}$ is unbiased, but typically results in cluster centroids near the centroid of all the data in the first round.
- A different heuristic tries to spread the initial centroids around as much as possible:
  - Place first centre on top of a randomly chosen data point
  - Place second centre on a data point as far away as possible from the first one
  - Place the i-th centre as far away as possible from the closest of centres 1 through $i-1$
- K-means clustering typically runs quickly. With a randomized initialization step, you can run the algorithm multiple times and take the clustering with best value of the objective

Choosing the number of clusters

- A difficult problem, ideas are floating around
- Delete clusters that cover too few points
- Split clusters that cover too many points
- Add extra clusters for "outliers"
- Minimum description length: minimize loss + complexity of the clustering
- Use a hierarchical method first

K-means-like clustering in general

- Given a set of instances (need not be real vectors),
  - Choose a notion of pairwise distance / similarity between instances
  - Choose a scoring function for the clustering
  - Optimize the scoring function, to find a good clustering.
- For most choices, the optimization problem will be intractable. Local optimization is often necessary.

Distance metrics

- Euclidean distance
- Hamming distance (number of mismatches between two strings)
- Travel distance along a manifold (e.g. for geographic points)
- Tempo / rhythm similarity (for songs)
- Shared keywords (for web pages), or shared in-links
- ...

Scoring functions

- Minimize: Summed distances between all pairs of instances in the same cluster. (Also known as "within-cluster scatter.")
- Minimize: Maximum distance between any two instances in the same cluster. (Can be hard to optimize.)
- Maximize: Minimum distance between any two instances in different clusters.

Common uses of K-means

- Often used in exploratory data analysis
- Often used as a pre-processing step before supervised learning
- In one-dimension, it is a good way to discretize real-valued variables into non-uniform buckets
- Used in speech understanding/recognition to convert wave forms into one of k categories (vector quantization)

Summary of K-means

- Fast way of partitioning data into K clusters
- It minimizes the sum of squared Euclidean distances to the clusters centroids
- Different clusterings can result from different initializations
- Can be interpreted as fitting a mixture distribution