Clustering

k-Means Algorithm

Mining of Massive Datasets

Leskovec, Rajaraman, and Ullman

Stanford University

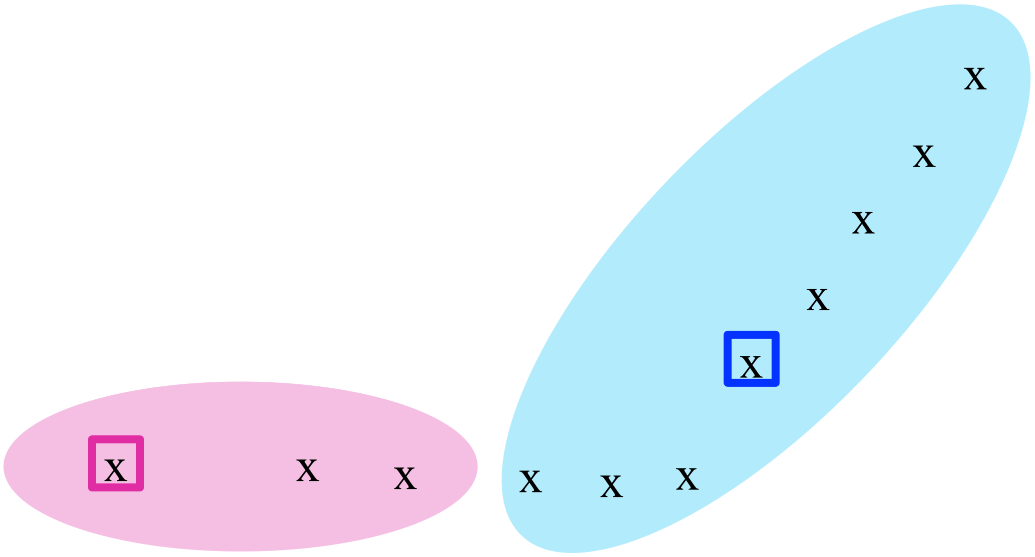
k–means Algorithm

- Assumes Euclidean space/distance
- Start by picking ***k***, the number of clusters
- Initialize clusters by picking one point per cluster
  - For the moment, assume we pick the *k* points at random

Populating Clusters

- **1)** For each point, place it in the cluster whose current centroid it is nearest
- **2)** After all points are assigned, update the locations of centroids of the ***k*** clusters
- **3)** Reassign all points to their closest centroid
  - Sometimes moves points between clusters
- **Repeat 2 and 3 until convergence**
  - **Convergence:** Points don’t move between clusters and centroids stabilize

Example: *k* = 2

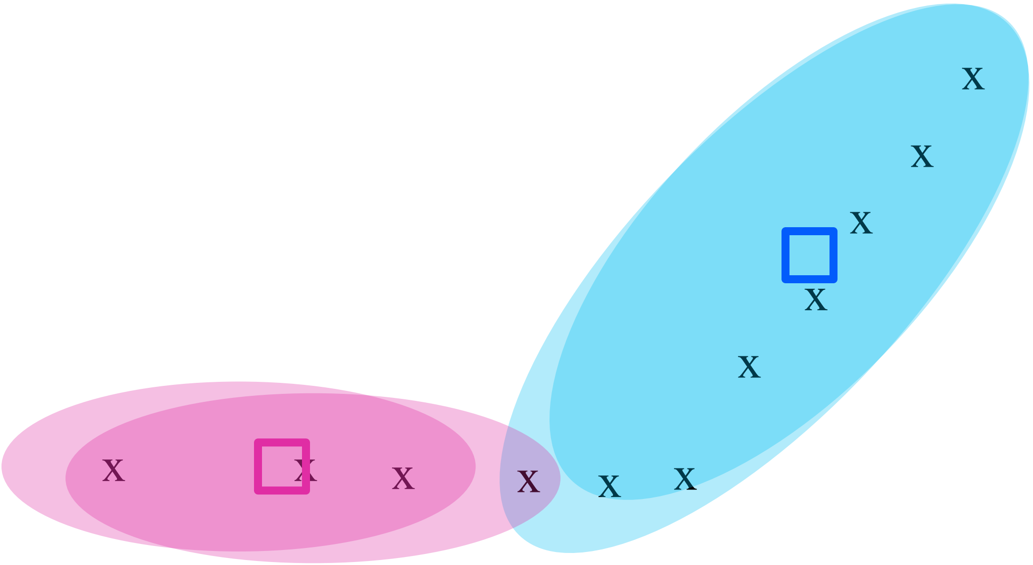


x … data point

□ … centroid

Round 1

Example: Assigning Clusters

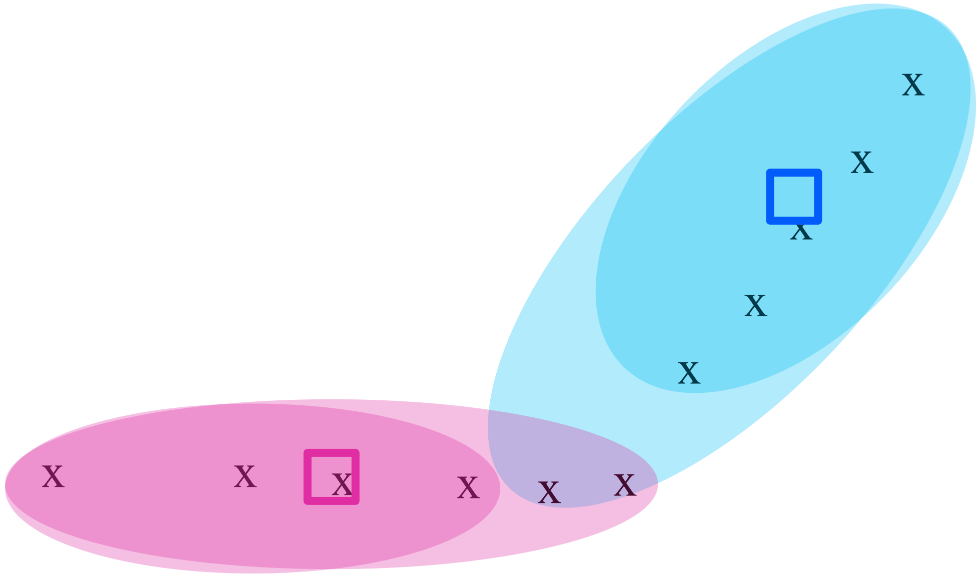


x … data point

□ … centroid

Round 2

Example: Assigning Clusters



x … data point

□ … centroid

Round 3

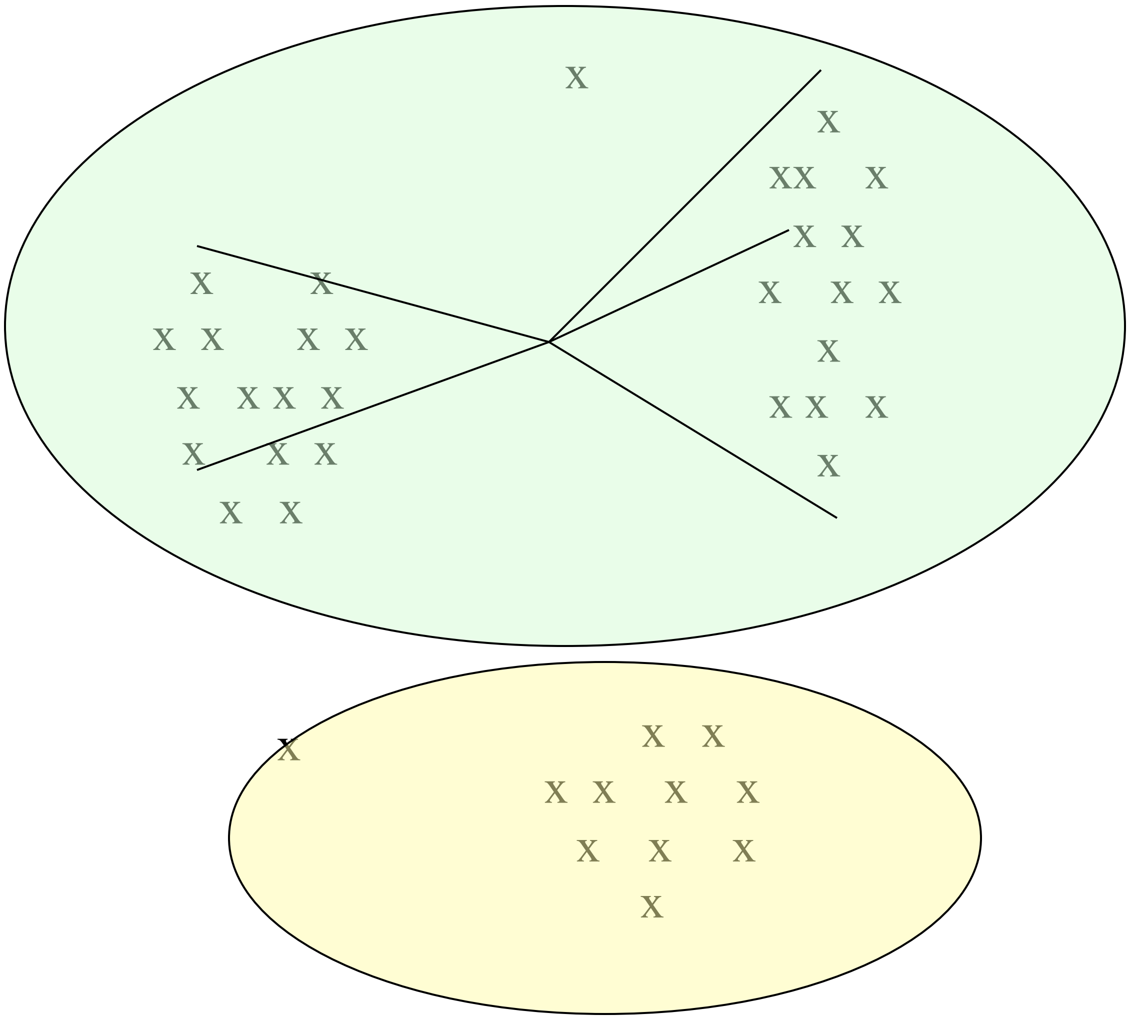
Picking the right value for *k*

How to select *k*?

- Try different *k*, looking at the change in the average distance to centroid, as *k* increases.

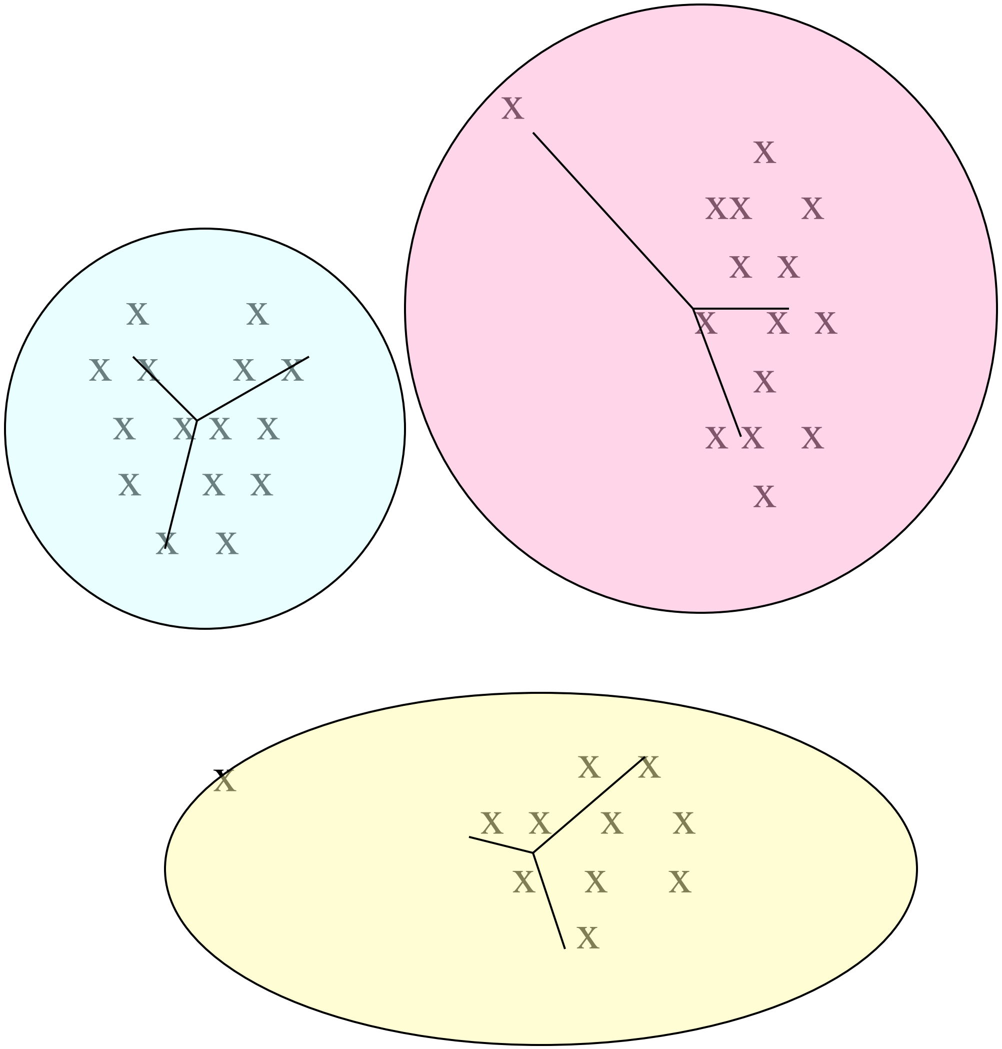
Example: Picking *k*

Too few;
many long
distances
to centroid.



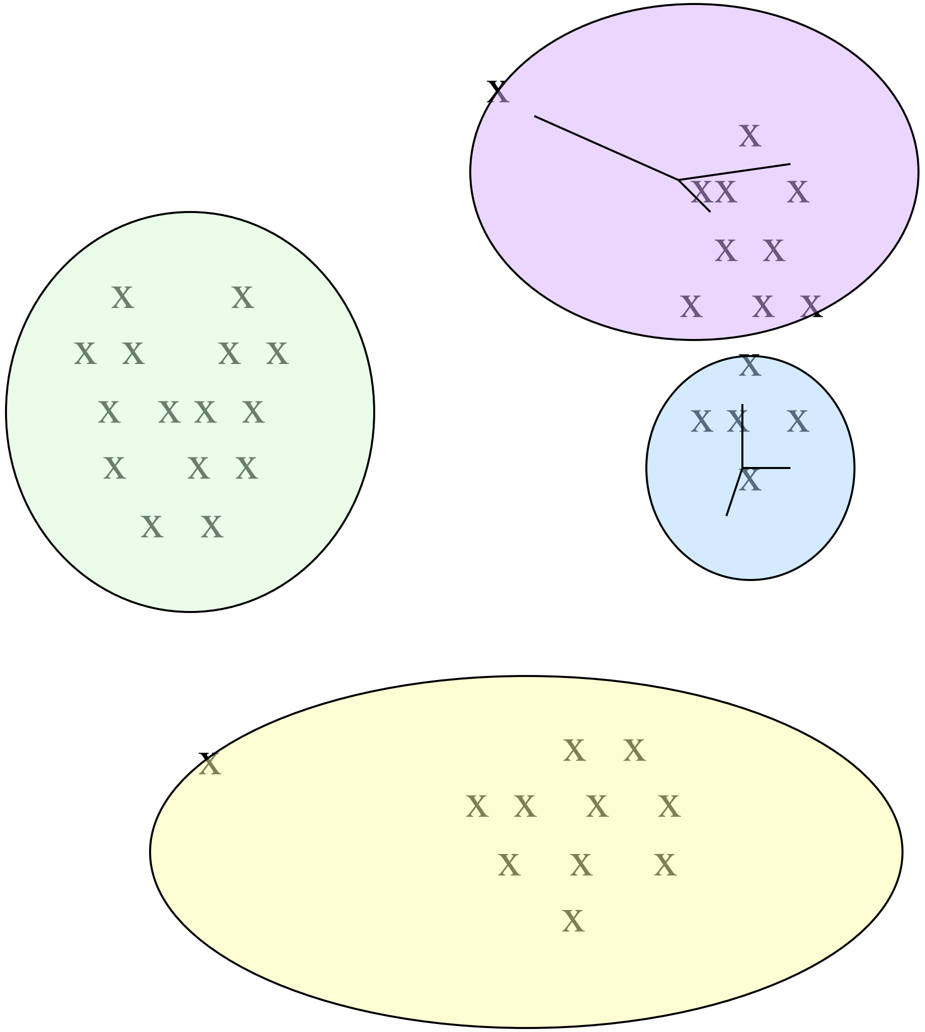
Example: Picking *k*

Just right;
distances
rather short.



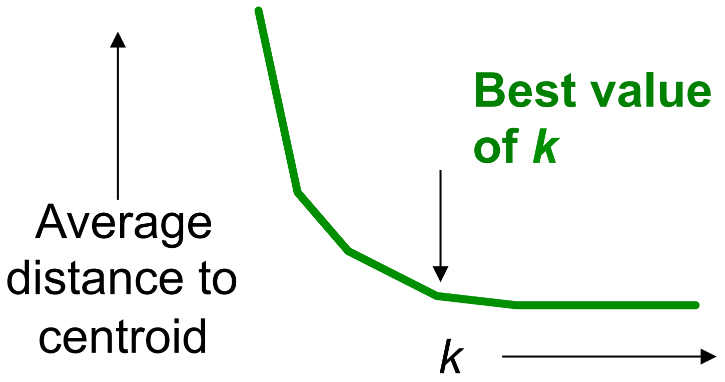
Example: Picking *k*

Too many;
little improvement in average distance.



Picking the right value for *k*

Average falls rapidly until right *k*, then falls much more slowly



Picking the initial *k* points

- Approach 1: Sampling
  - Cluster a sample of the data using hierarchical clustering, to obtain k clusters
  - Pick a point from each cluster (e.g. point closest to centroid)
  - Sample fits in main memory

- Approach 2: Pick "dispersed" set of points
  - Pick first point at random
  - Pick the next point to be the one whose minimum distance from the selected points is as large as possible
  - Repeat until we have *k* points

Complexity

- In each round, we have to examine each input point exactly once to find closest centroid
- Each round is *O(kN)* for *N* points, *k* clusters
- But the number of rounds to convergence can be very large!
- Can we cluster in a single pass over the data?