K-means Clustering

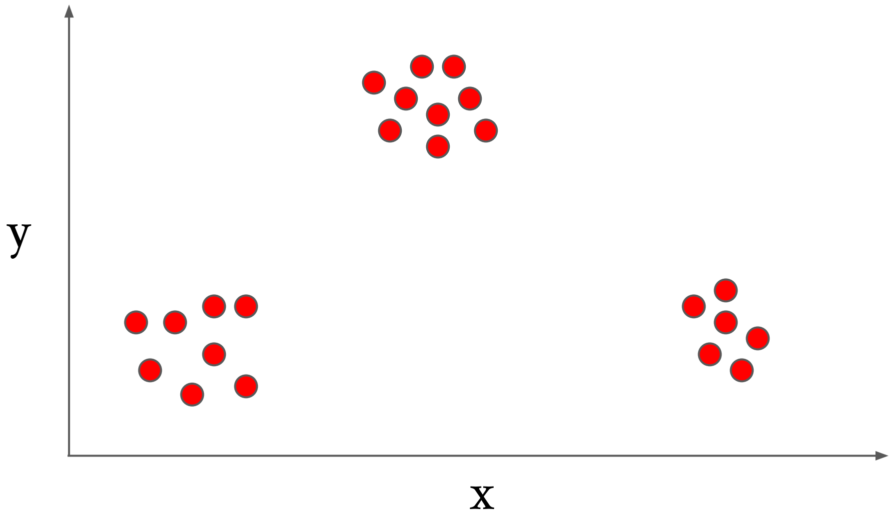
EE379K - Architectures for Big Data Sciences
Dr. Sriram Vishwanath
Department of Electrical and Computer Engineering
The University of Texas at Austin
Spring 2016

K-Means Intro

- K-means is an **unsupervised** clustering algorithm
- Data may have attributes, but no labels are required.
- We will cover a few forms of K-means
  - Standard k-means
  - Hierarchical k-means
  - Kernel k-means

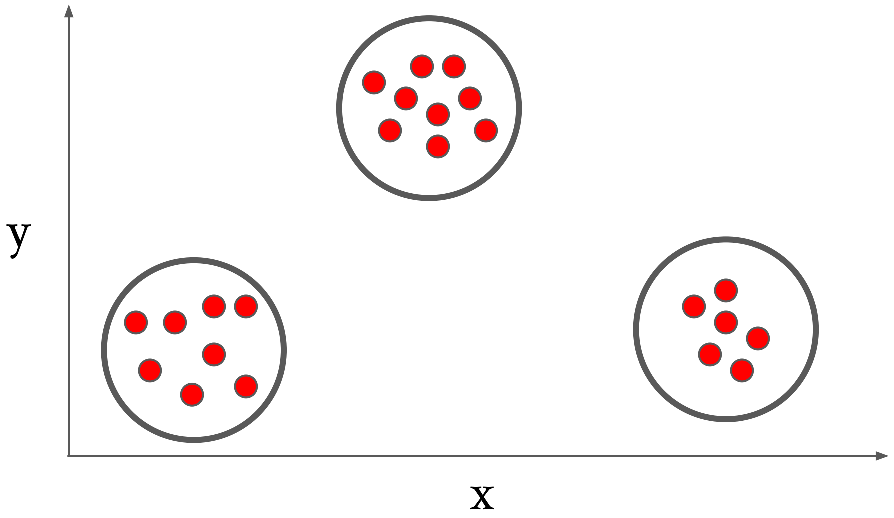
Clustering Overview

- Goal: Group a set of objects into clusters
- “Similar” objects should be in the same cluster



Clustering by Position

- Total of three clusters

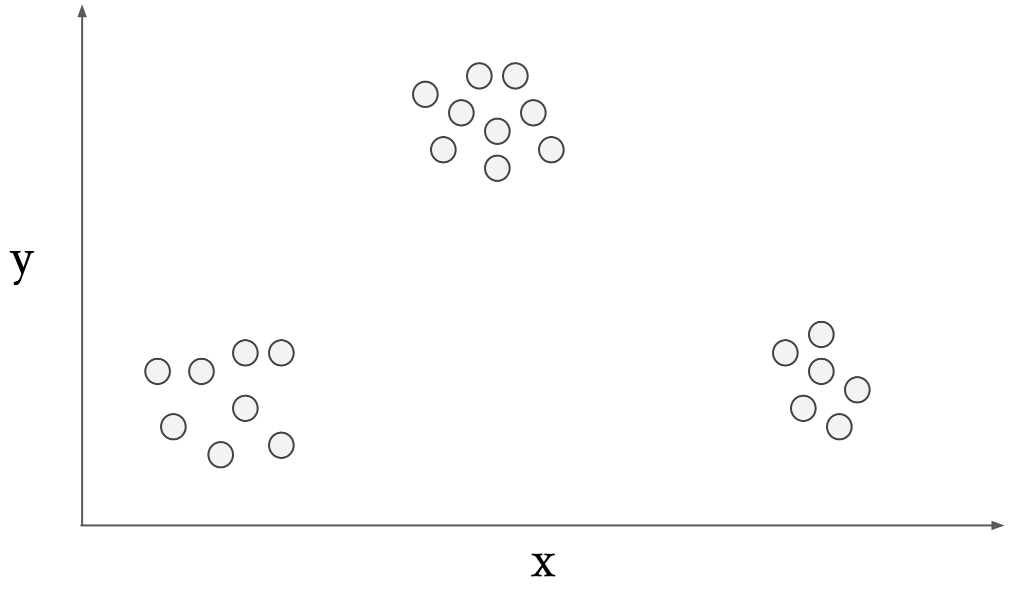


K-Means Clustering Algorithm

- Partitions **N** input points into **K** clusters
- Clusters defined by their **centroids** ("centers")
  - Once clusters are defined, you can identify "exemplars"
- Each input point $\mathbf{n_i}$ belongs to some cluster **k** (with centroid $\mathbf{c_k}$)
- Goal: Minimize distance from each point to the centroid of its cluster

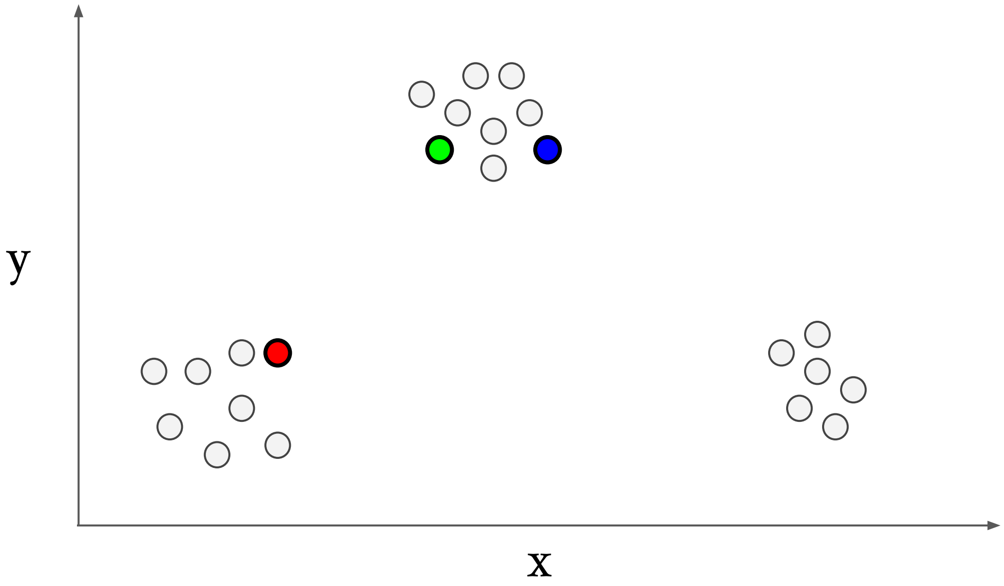
K-Means Example

- Segment the following points into three clusters (by position)



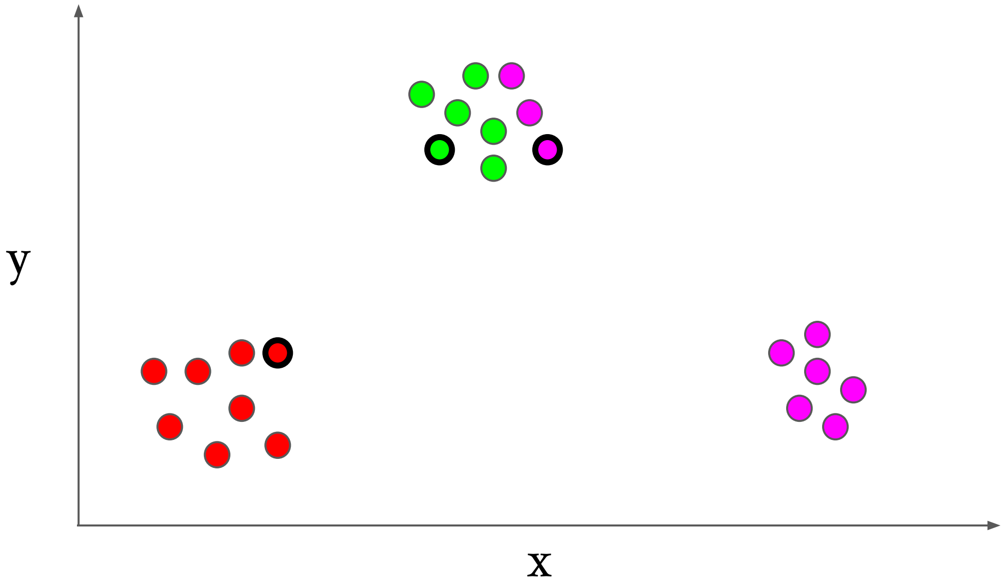
K-Means Example

1. K = 3. Pick three random data points as initial centroids



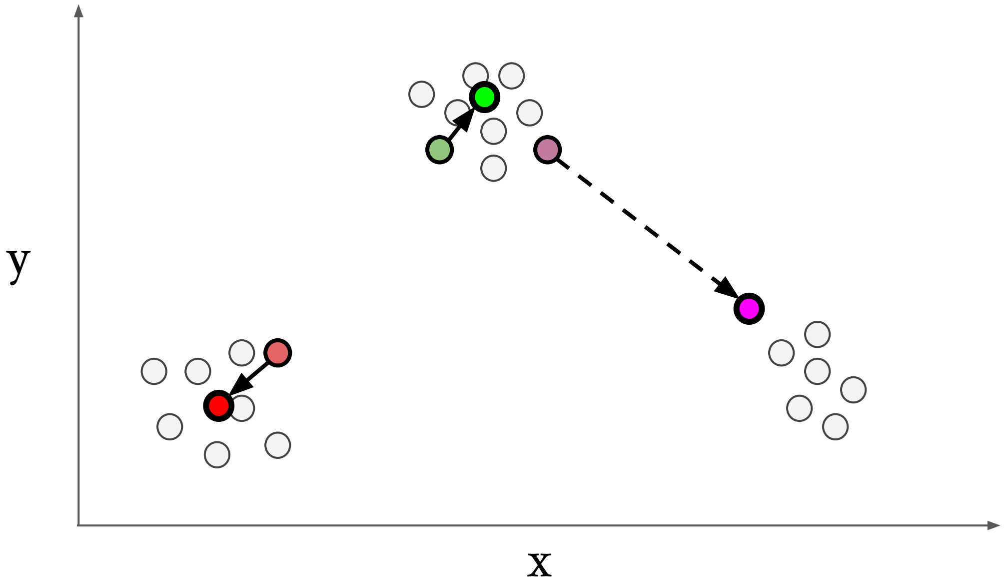
K-Means Example

2. Assign each point to its nearest centroid



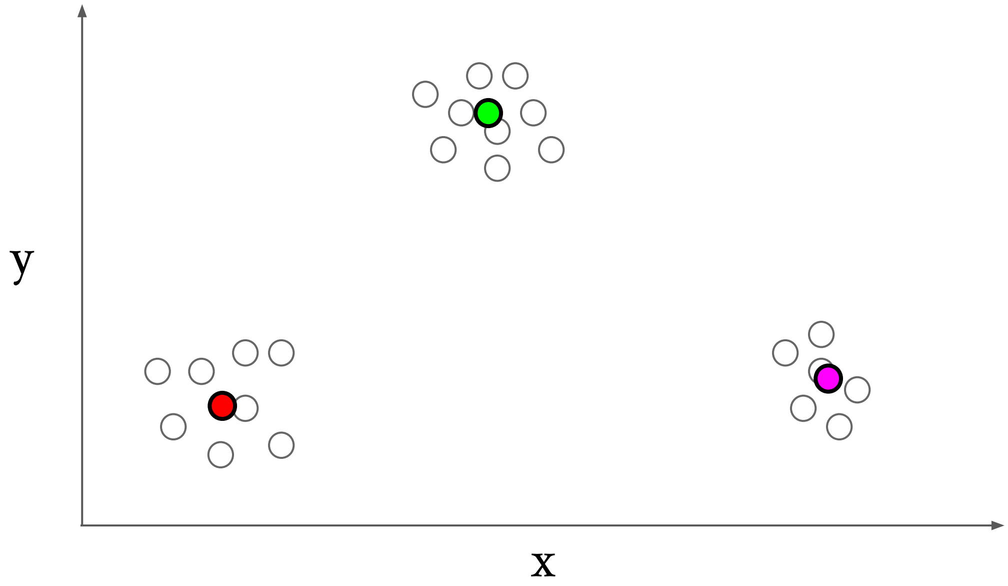
K-Means Example

3. Recompute each centroids location as the average location of its children

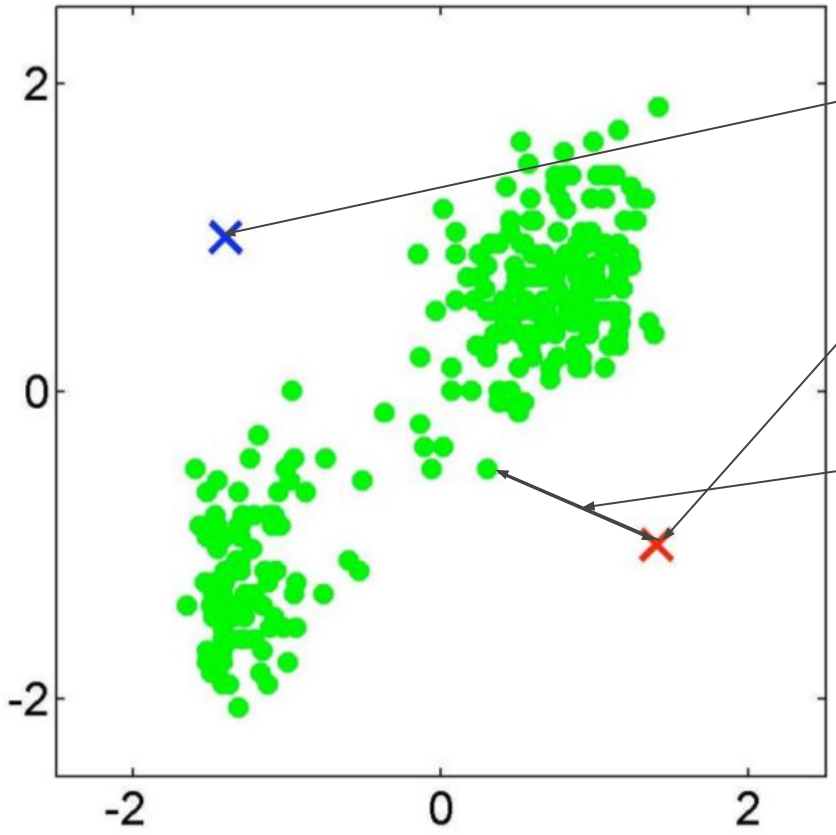


K-Means Example

4. Repeat assignment and update steps until no assignments change



K-means example with K=2



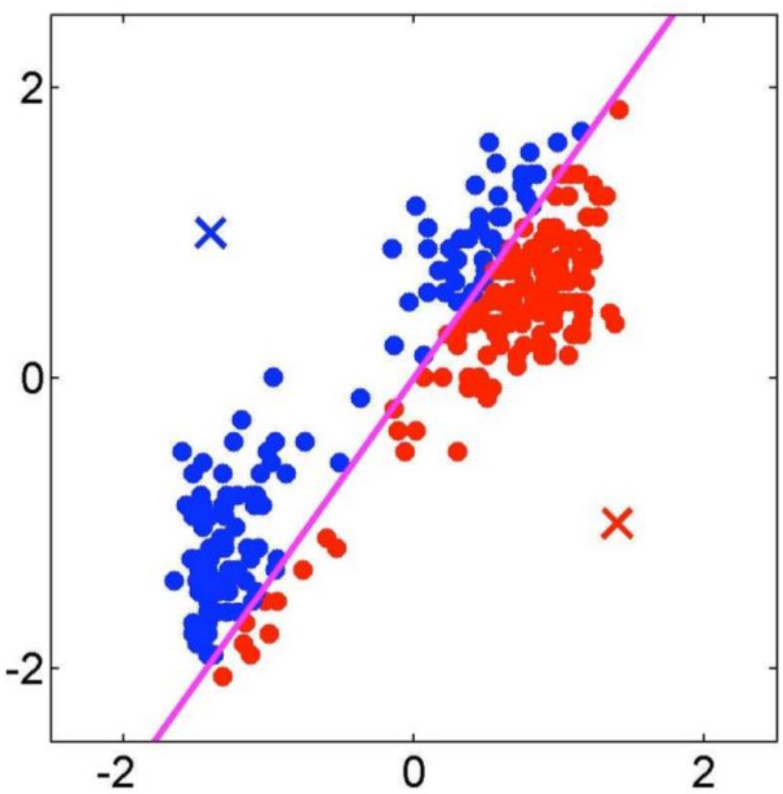
Cluster Centroids,
u_1 and u_2

Note: the cluster centroids are not data points, they are just markers chosen at random

Distance from cluster centroid to point i: $|| u_2 - x_i ||$

Figures taken from Piyush Rai’s Machine Learning class, University of Utah

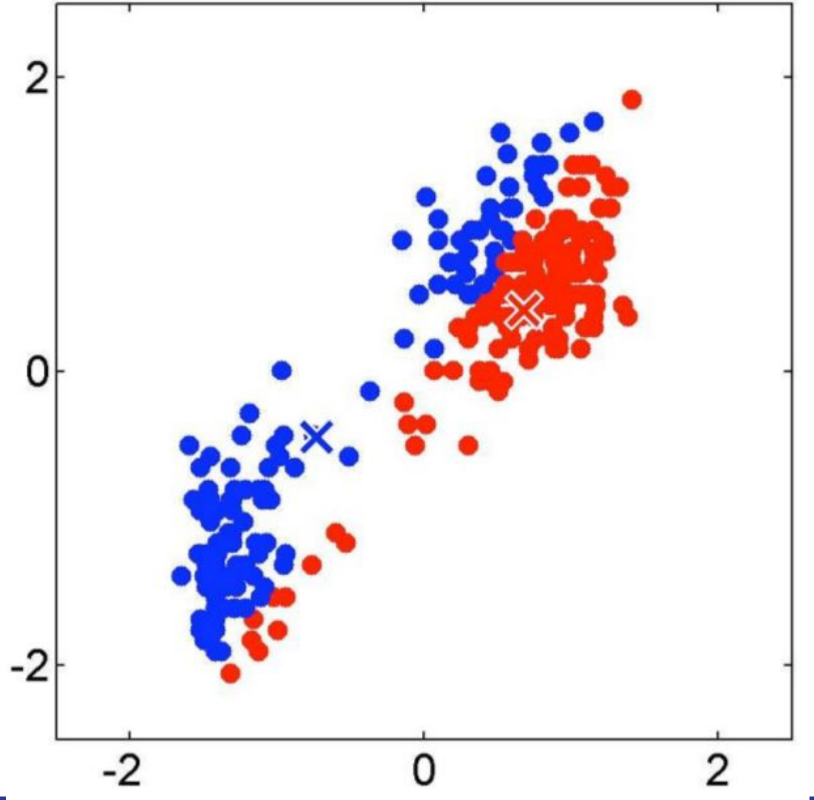
Step 1: Assign clusters to all the data points



Assign each data point the cluster centroid with minimum distance

Figures taken from Piyush Rai's Machine Learning class, University of Utah

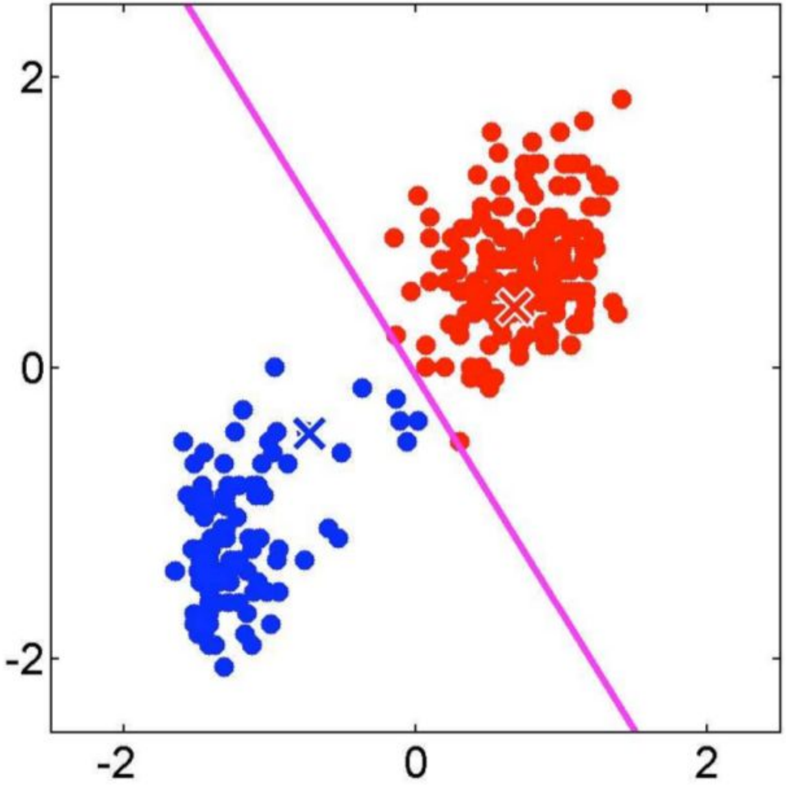
Step 2: Move the cluster centroids



Move the cluster centroids to the average of all points belonging to that cluster

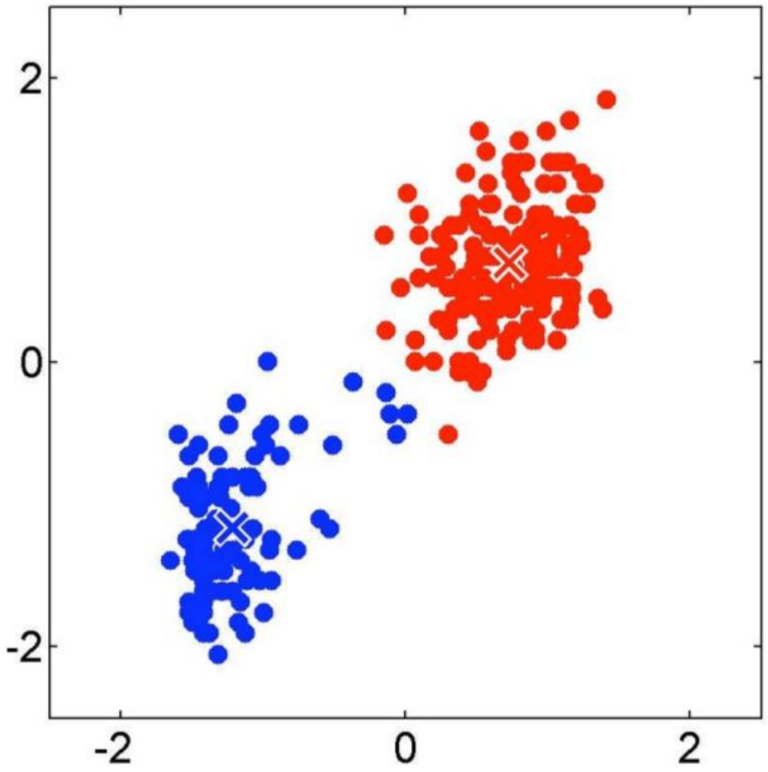
Figures taken from Piyush Rai's Machine Learning class, University of Utah

Step 3: Re-assign clusters to data points



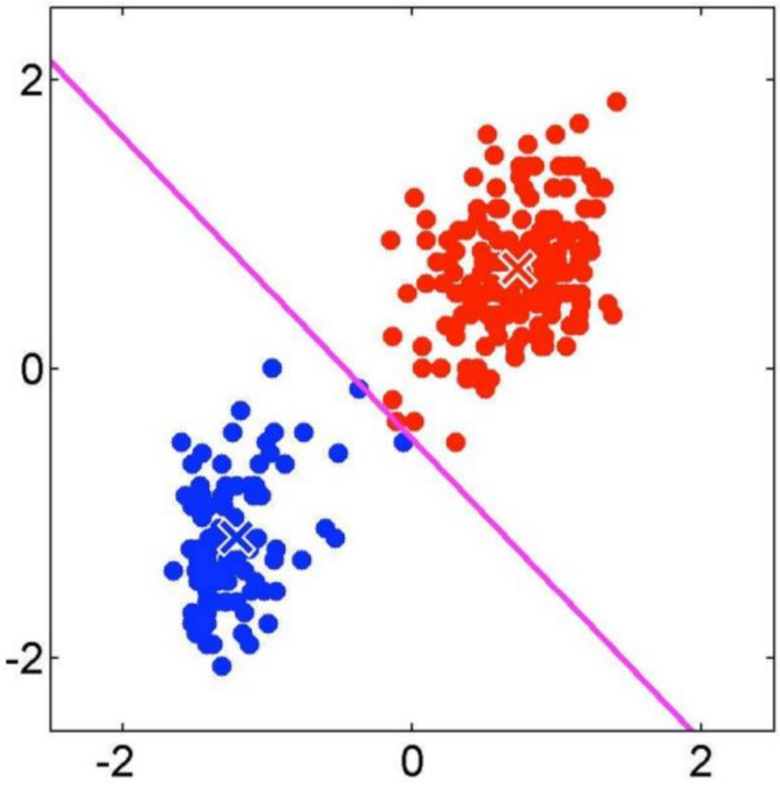
Figures taken from Piyush Rai's Machine Learning class, University of Utah

Step 4: Move cluster centroids



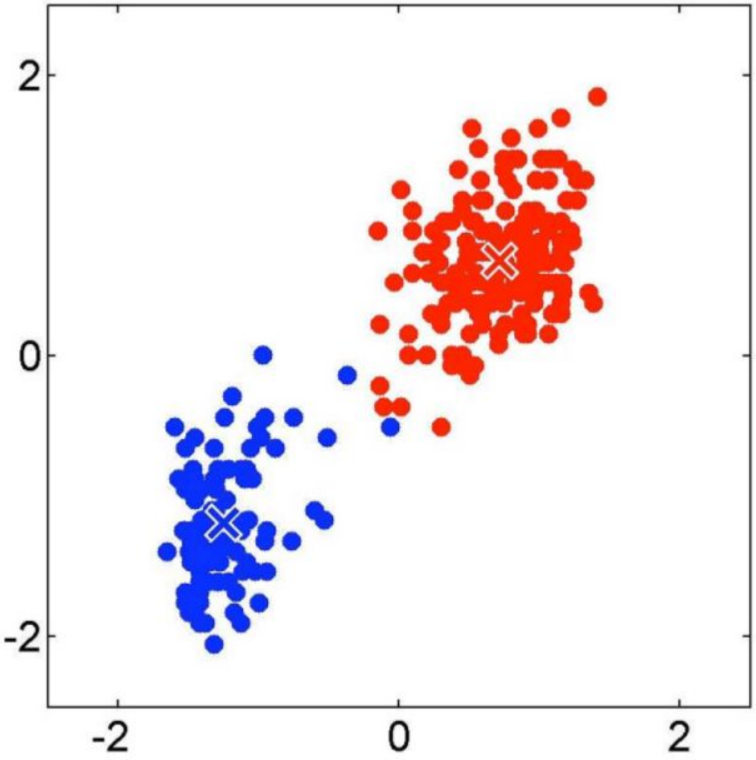
Figures taken from Piyush Rai's Machine Learning class, University of Utah

Step 5: Re-assign clusters to data points



Figures taken from Piyush Rai's Machine Learning class, University of Utah

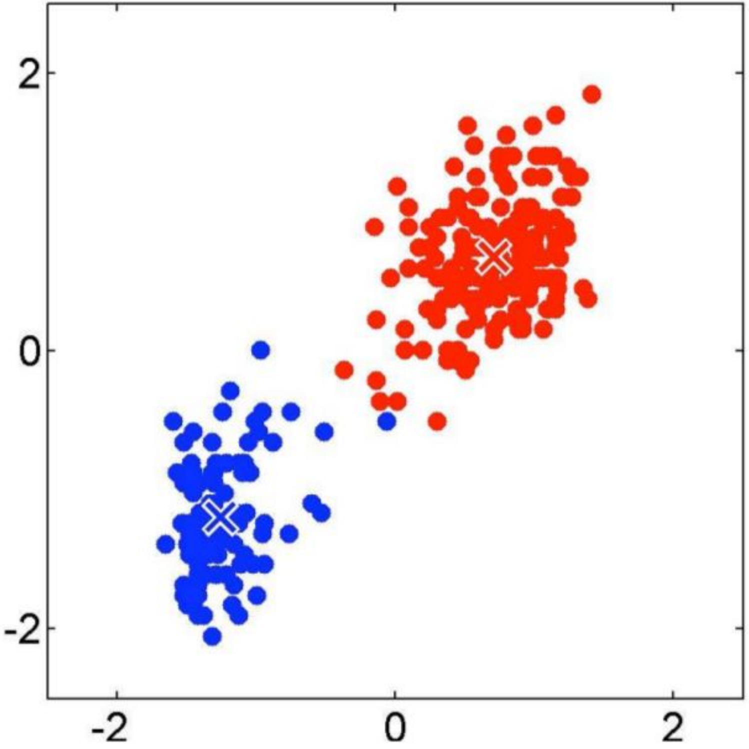
Step 6: Move the cluster centroids



At this point the cluster centroids moved only a tiny amount

Figures taken from Piyush Rai's Machine Learning class, University of Utah

Termination



Repeat Step 1 and 2 until the cluster centroids do not move any more

Figures taken from Piyush Rai's Machine Learning class, University of Utah

K-Means Optimization Objective

- Minimize the **average squared distance** between cluster centroids and the points in the clusters
- K-means attempts to minimize the **Objective Function**:

number of clusters number of cases

centroid for cluster j

case i

$$\text{objective function} \leftarrow J = \sum_{j=1}^{k} \sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$$

Choosing Initial Centroids

- The final clustering is highly dependent on the initial centroids
- If initial centroids are poorly chosen, the algorithm falls into **local-minima** and produces a suboptimal clustering
- There are a few ways to approach this problem

Choosing Initial Centroids: Multiple Runs

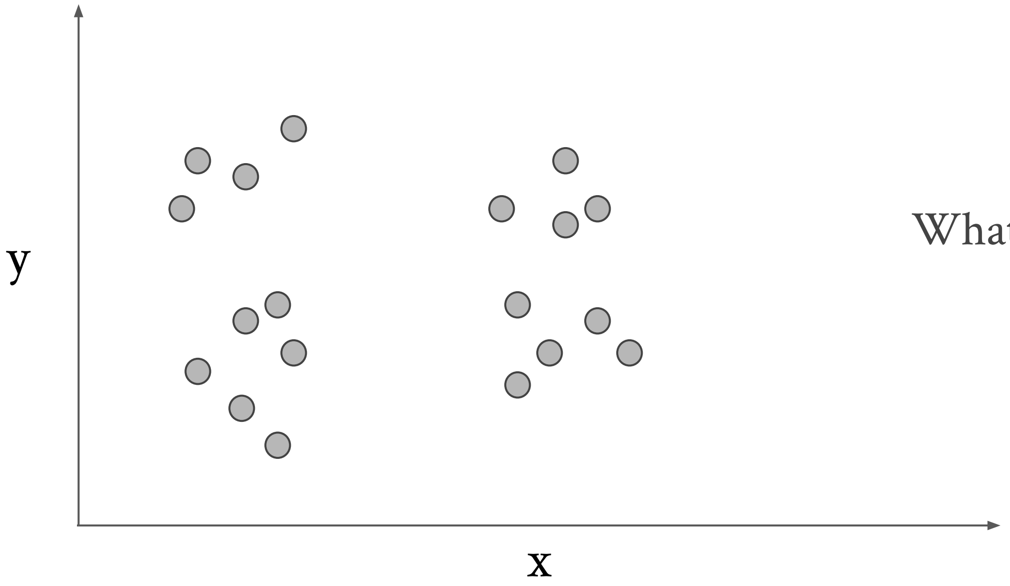
- Choose initial centroids randomly
- Run k-means multiple times with different centroids each time
- Choose the run with the lowest objective function value
- Can become expensive!

Choosing Initial Centroids: K-means++

1. Choose one centroid uniformly at random among the data points
2. For each data point **x**, compute **D(x)**
   - **D(x):** distance from **x** to the nearest centroid that has already been chosen
3. Choose one point **x** at random as a new centroid, using a weighted probability distribution
   - The point is chosen with probability proportional to **D(x)**
4. Repeat steps 2 and 3 until **K** centers are chosen

K-Means: Picking K

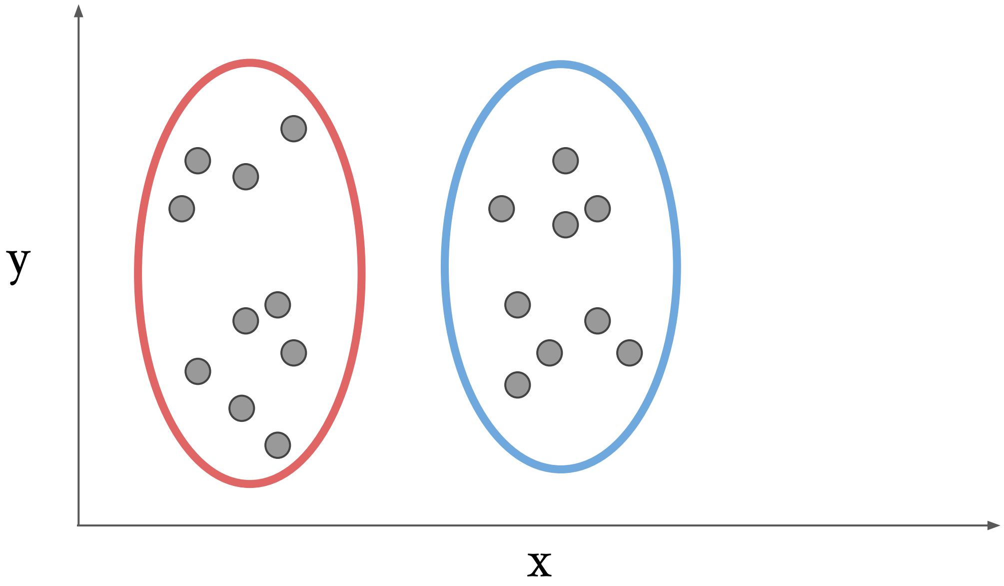
- It's often hard to pick K



What's the "right" K?

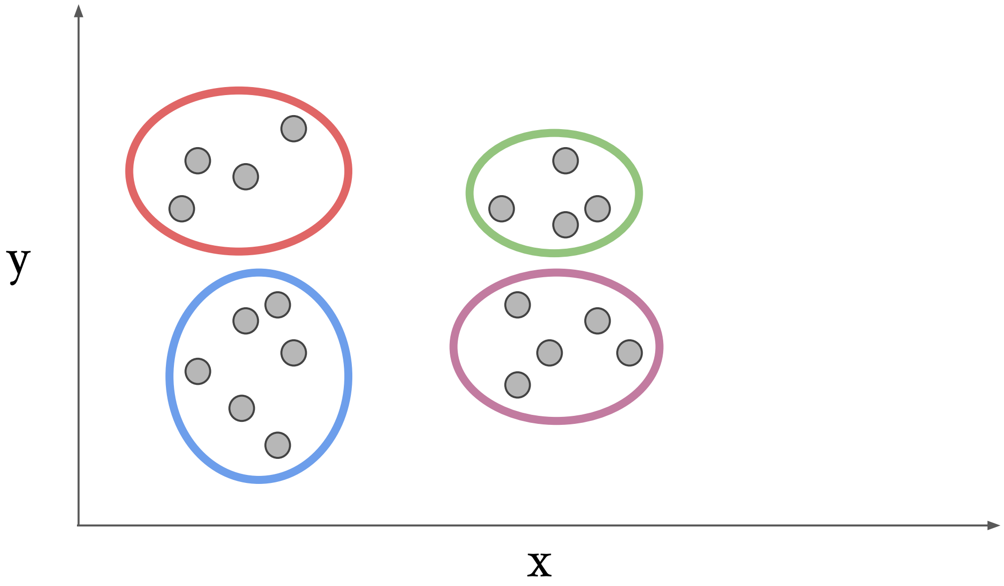
K-Means

- K = 2?



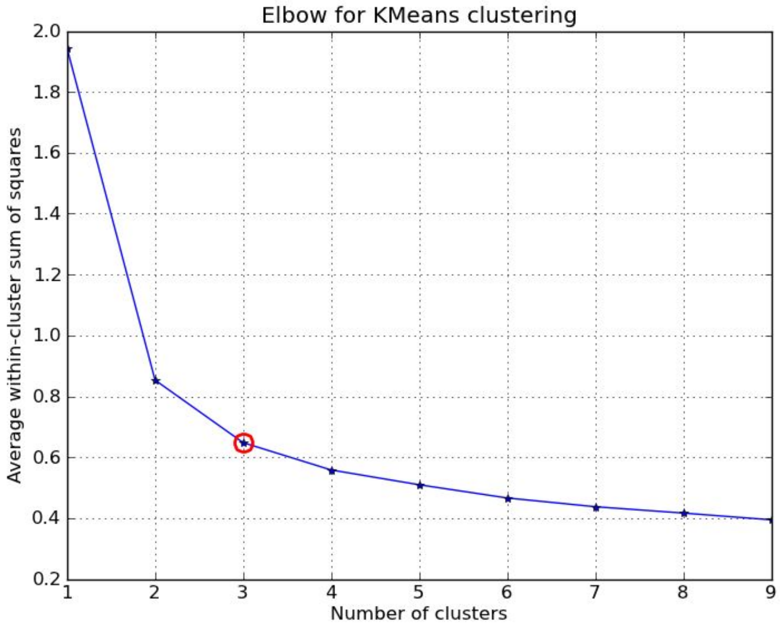
K-Means

- K = 4?



Choosing K: Elbow Method

- Plot the objective function obtained from running K-means with a range of K-values. Choose the “elbow” value.

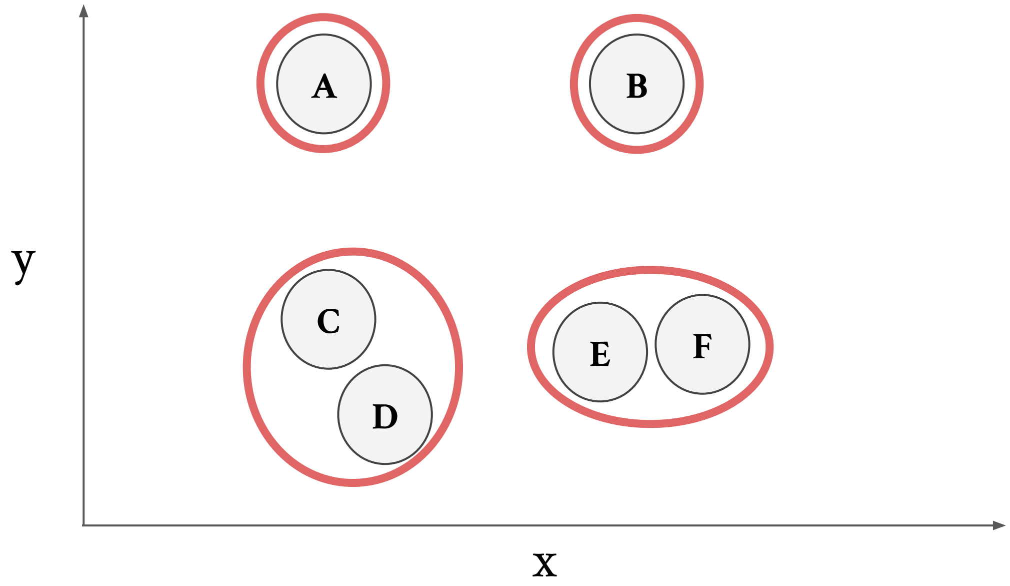


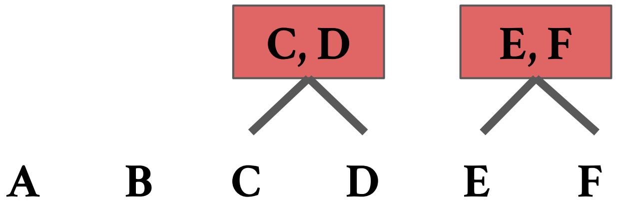
Choosing K: Hierarchical Method

- K defines the **scale** of the structures
- Low K produces coarse structures, high K produces fine structures
- How can we capture both scales?

Hierarchical Clustering

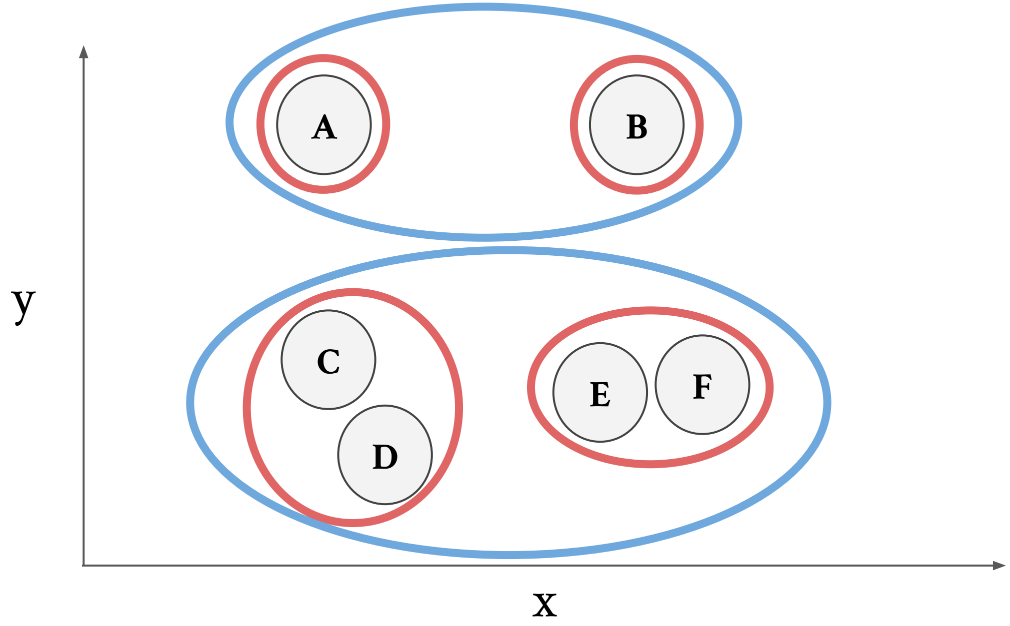
- Generate a **hierarchy** of clusters

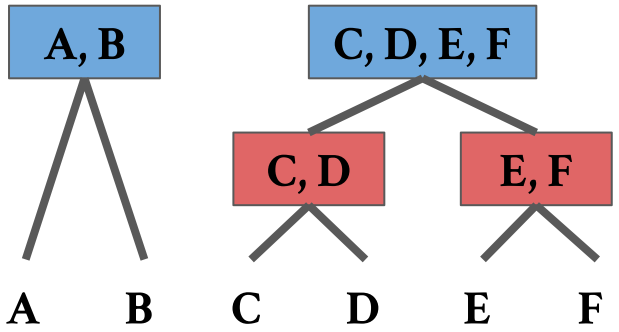




Hierarchical Clustering

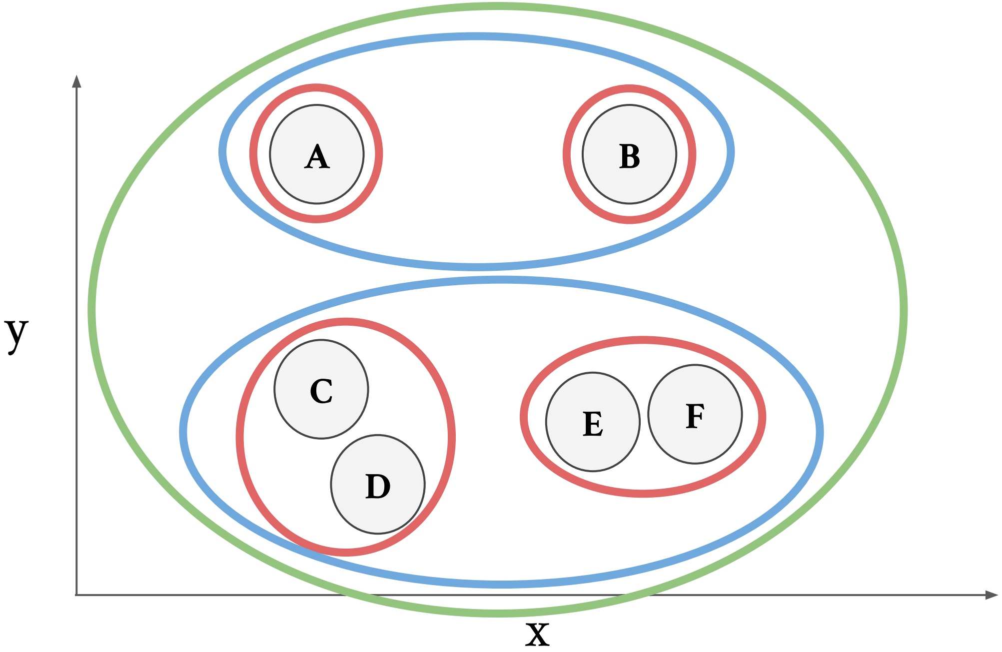
- Generate a **hierarchy** of clusters

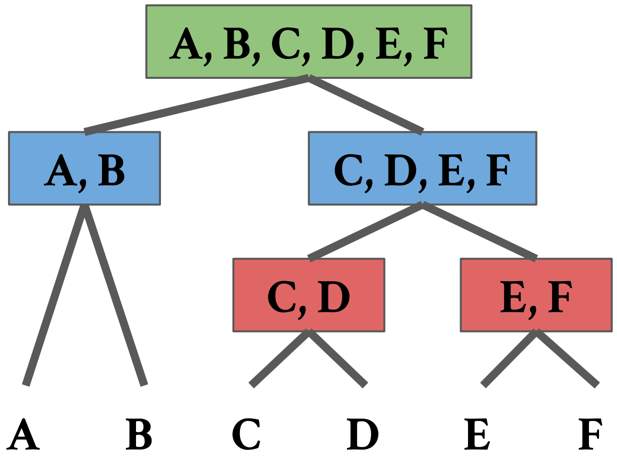




Hierarchical Clustering

- Generate a **hierarchy** of clusters





Hierarchical Clustering

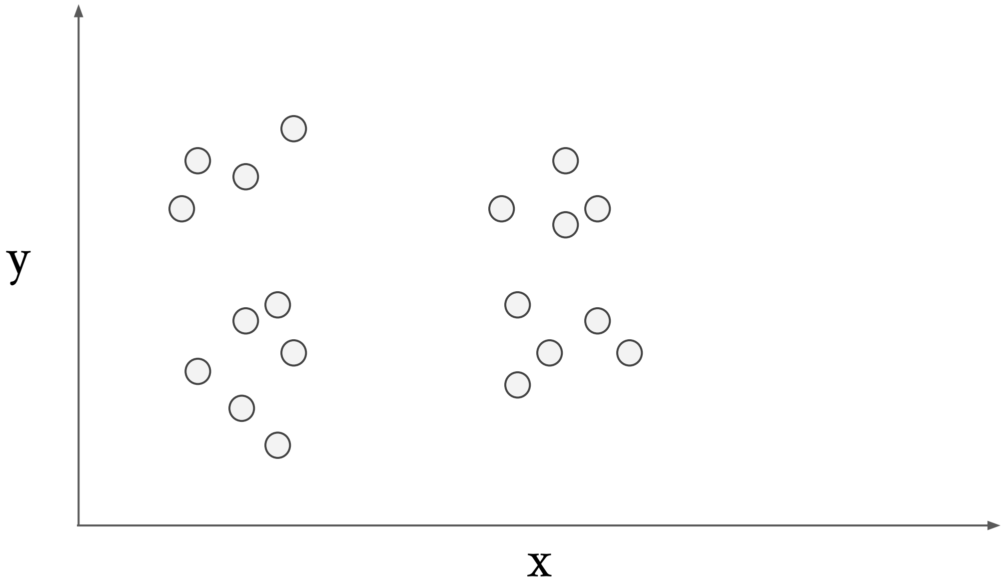
- Two general types
- Agglomerative (bottom-up)
  - Fine clusters first
- Divisive (top-down)
  - Coarse clusters first
- Previous example was agglomerative

Hierarchical K-Means

- Divisive clustering algorithm
- Apply k-means to generate K clusters
- Run k-means on each of the K clusters separately
- Recurse until finished

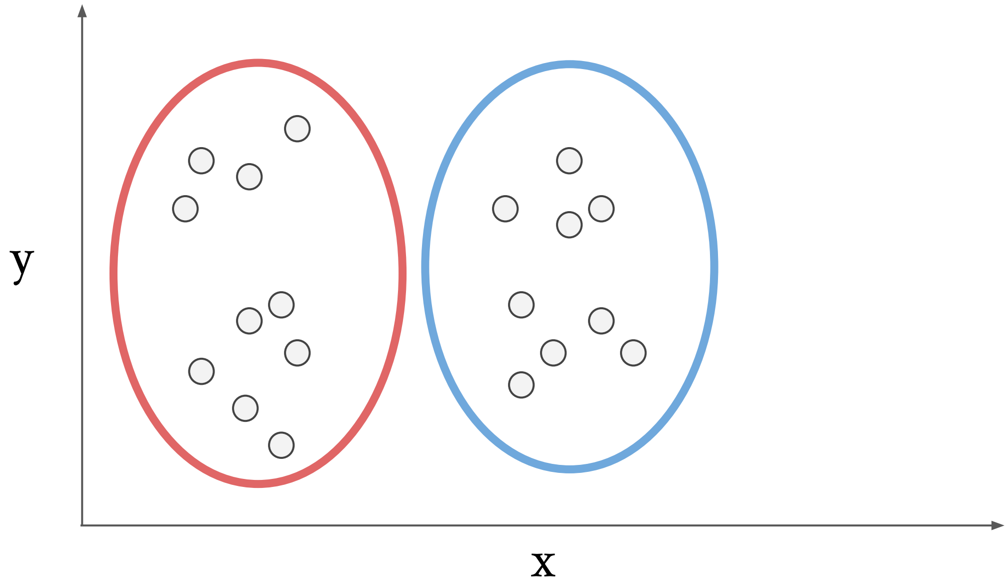
Hierarchical K-Means Example

- K = 2



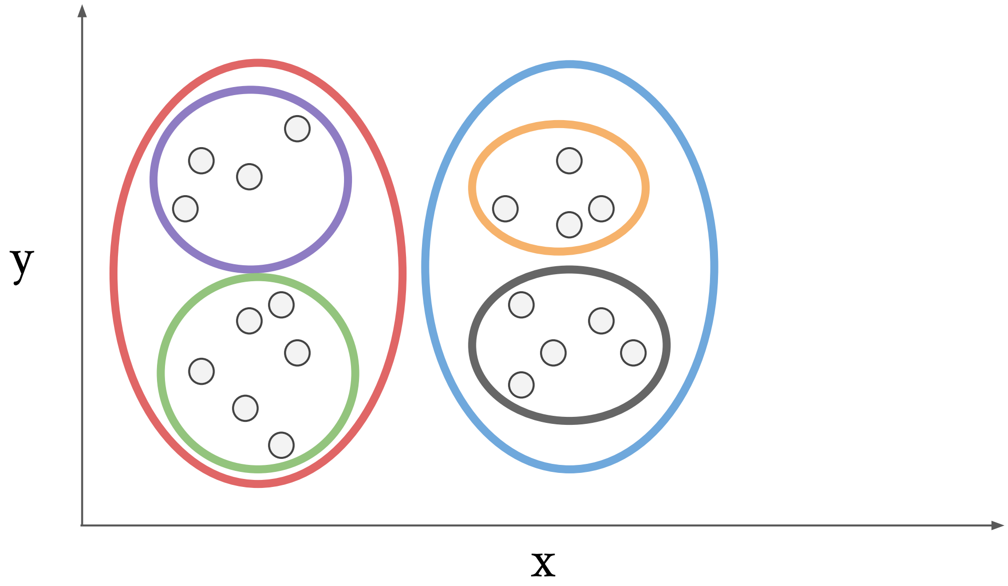
Hierarchical K-Means Example

- Apply k-means



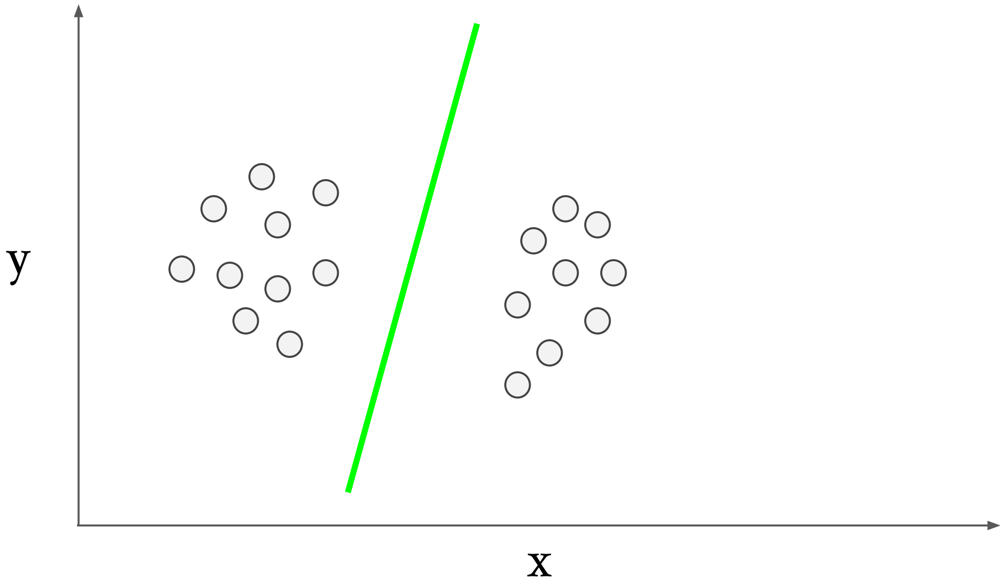
Hierarchical K-Means Example

- Apply k-means on each cluster separately



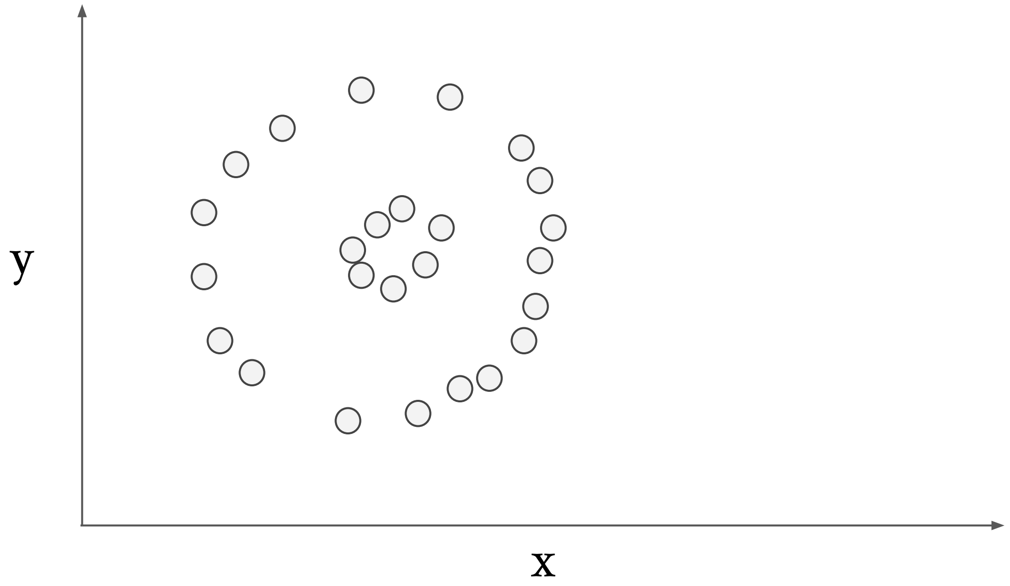
Kernel K-Means

- Standard K-means works well when data is linearly separable



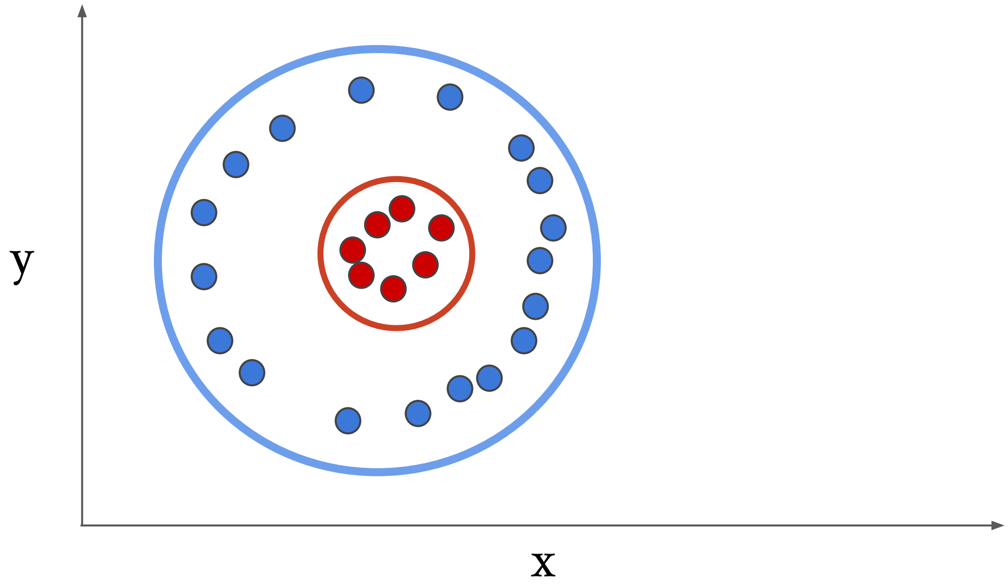
Kernel K-Means

- Standard K-means fails when data is not linearly separable



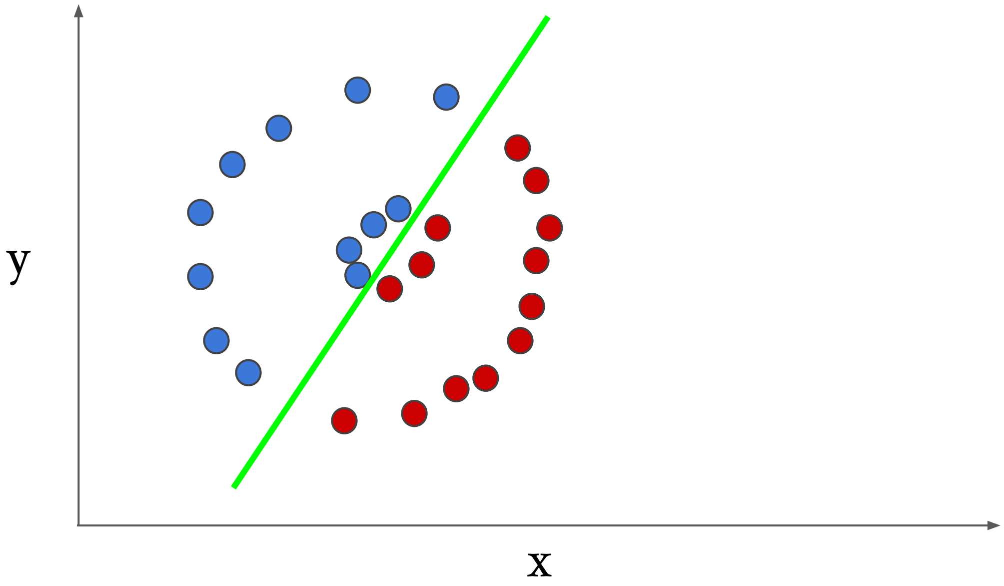
Kernel K-Means

- What we want:



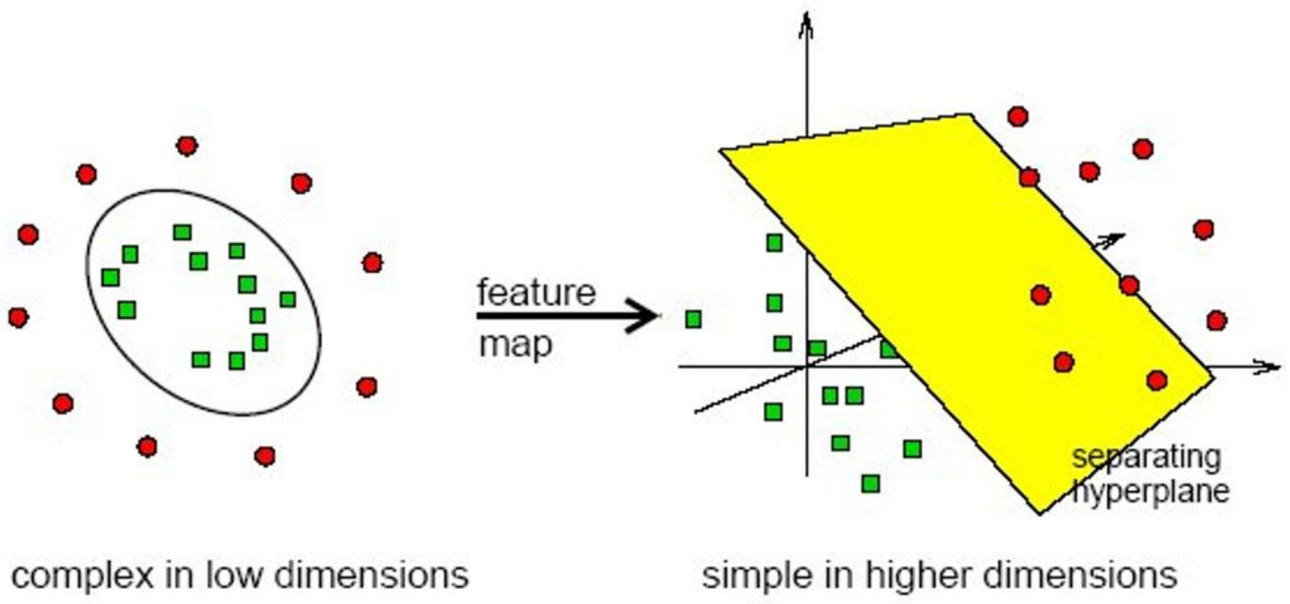
Kernel K-Means

- What we get:



Kernel K-Means: Feature Space

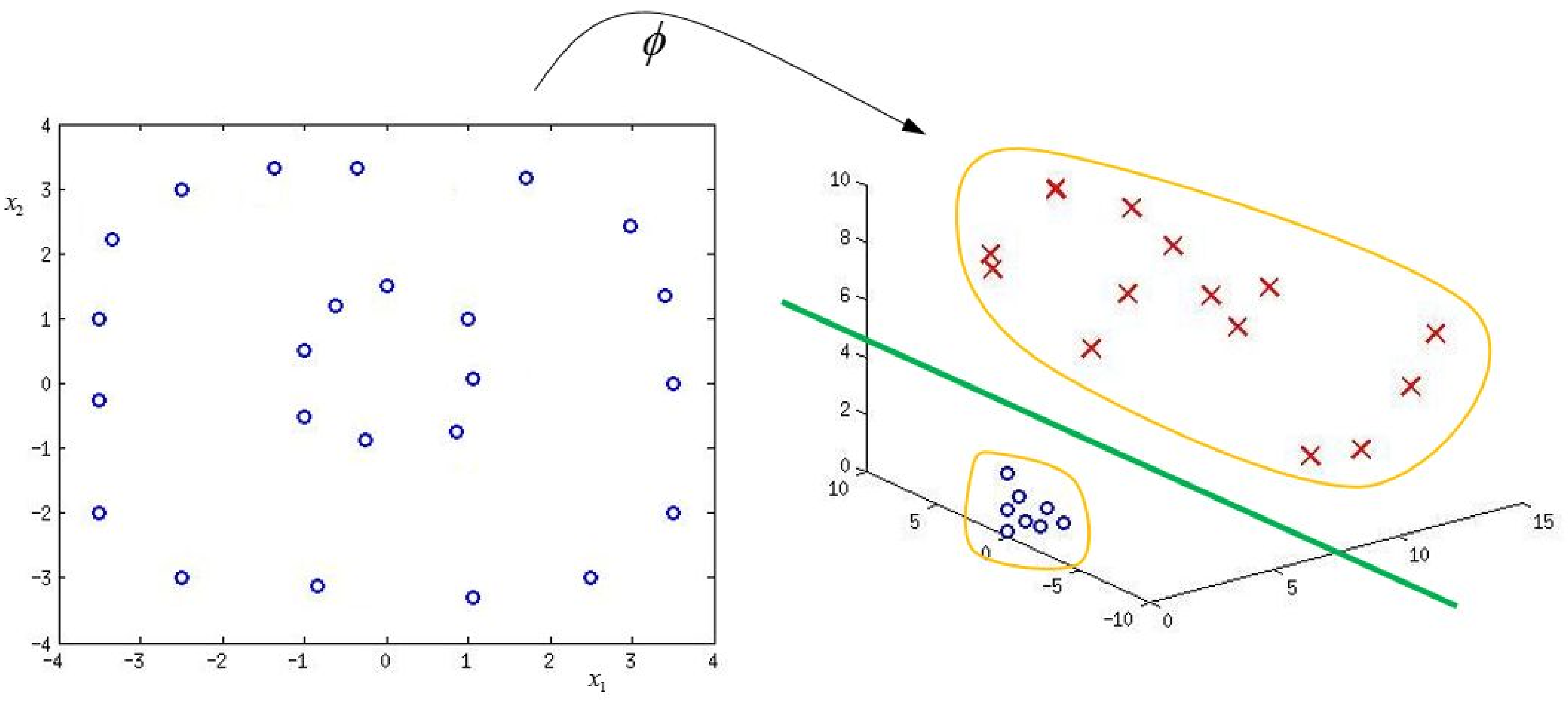
- Solution: Move to a higher dimension (called the **Feature Space**)
- Transform each point: $\mathbf{x} \rightarrow \mathbf{\Phi(x)}$



Kernel K-Means: The Kernel

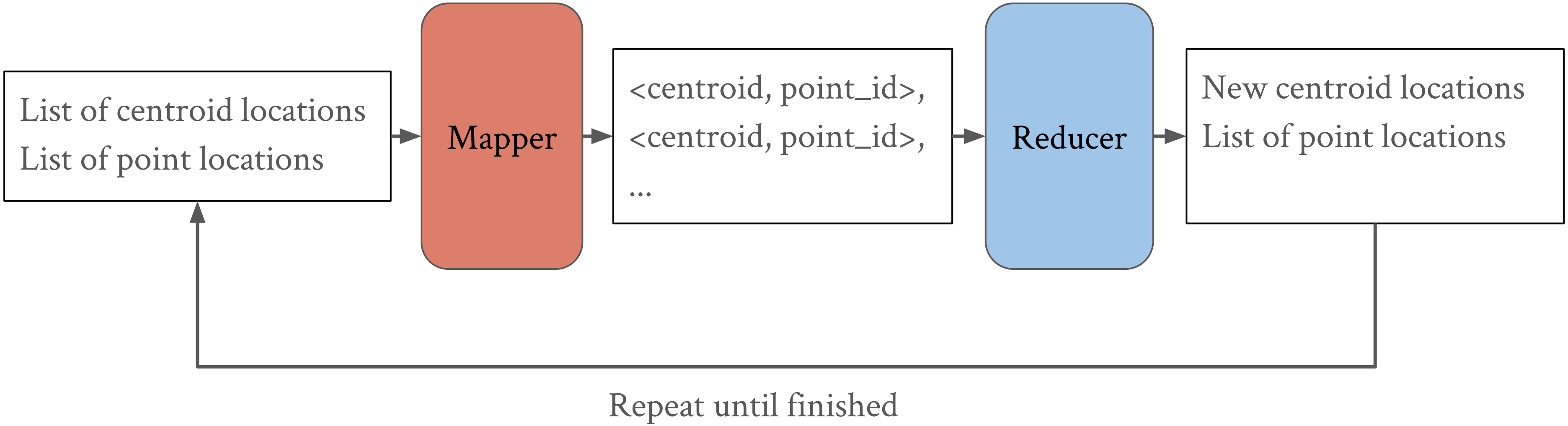
- Now, how do we define the distance between **Φ(x)** and **Φ(y)**?
- Use a kernel **K(x, y)** that returns the distance. **K** is a matrix
- Polynomial Kernel: **K(x, y)** = **<Φ(x) ,Φ(y)>**

Kernel K-Means: Polynomial Kernel Example



Distributed K-Means

- Implementing K-means using MapReduce
- General algorithm:



Distributed K-Means: Mapper

- For each point:
  - Calculate it’s distance to all centroids
  - Find the closest centroid **c**
  - Emit **<c, point_id>**

Distributed K-Means: Reducer

- For each centroid:
  - Make a list of all the points that belong to the centroid
  - Average their locations to get the new centroid location **c'**
  - Emit **<centroid_id, c'>**
- For each point
  - Emit **<point_id, point_location>**