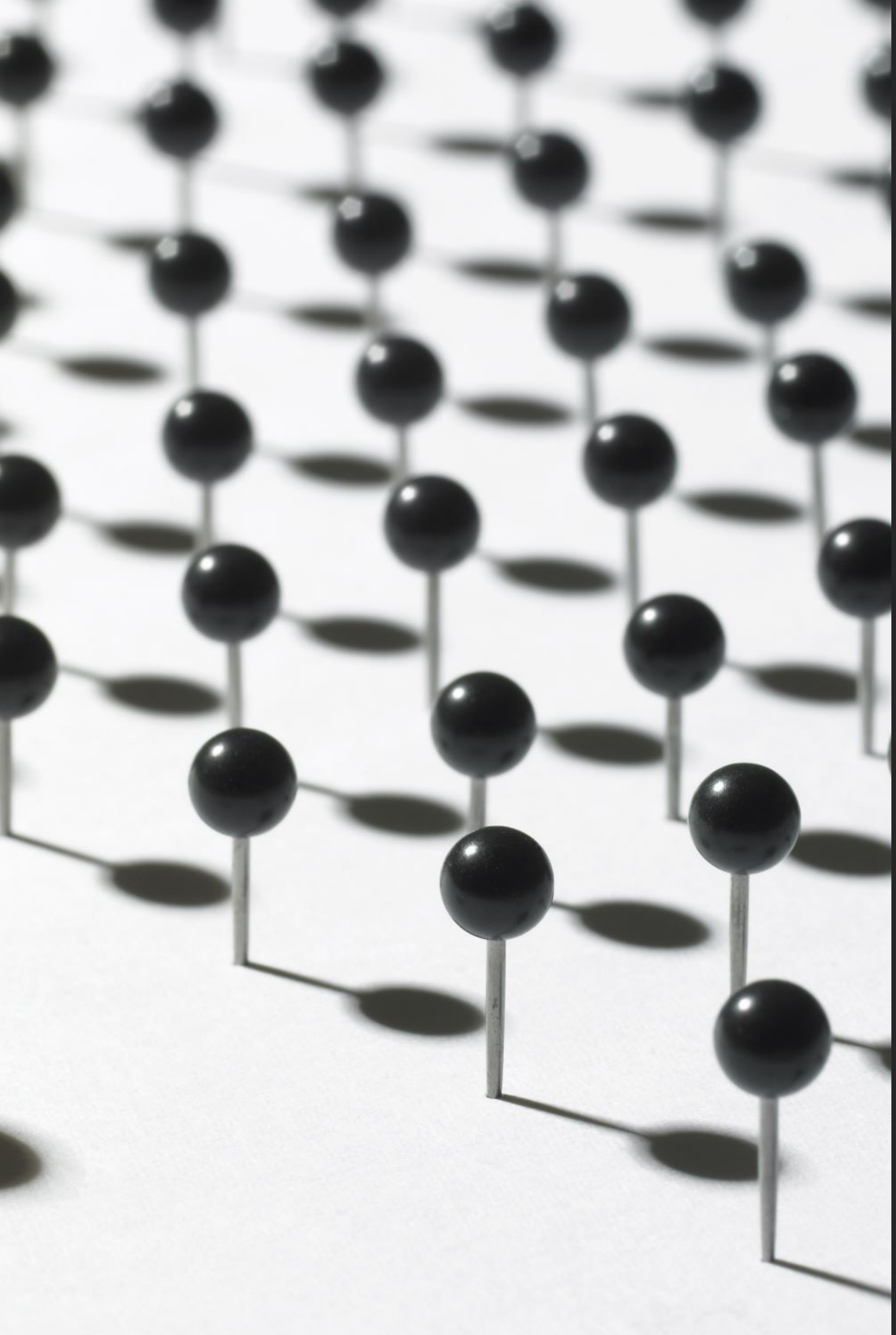
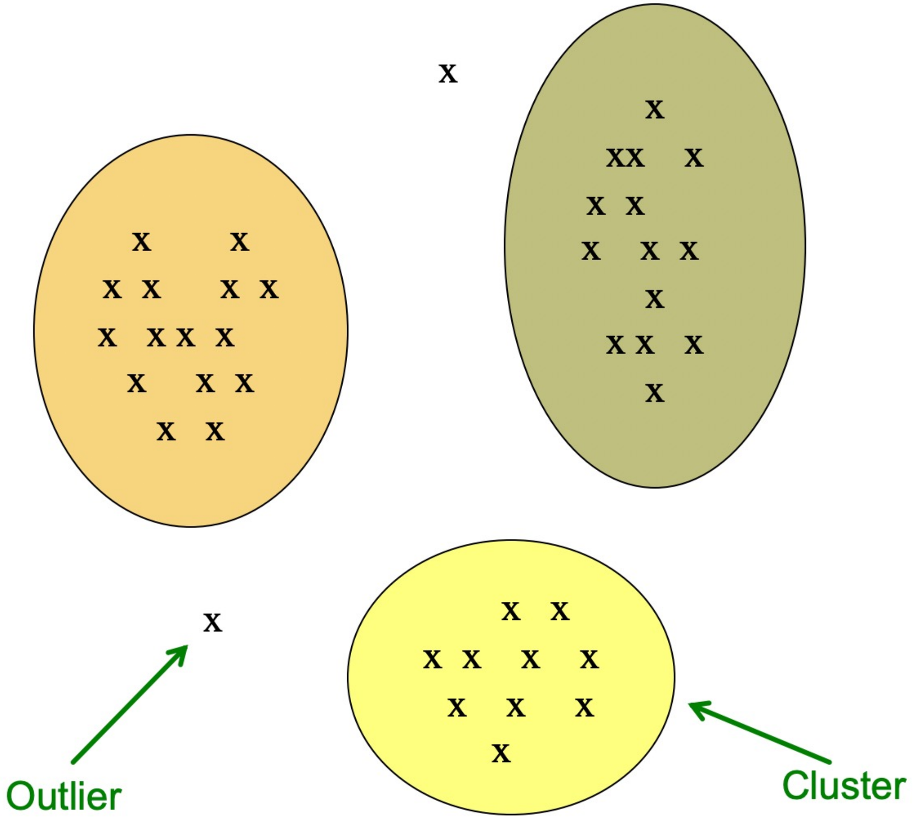
Clustering Methods



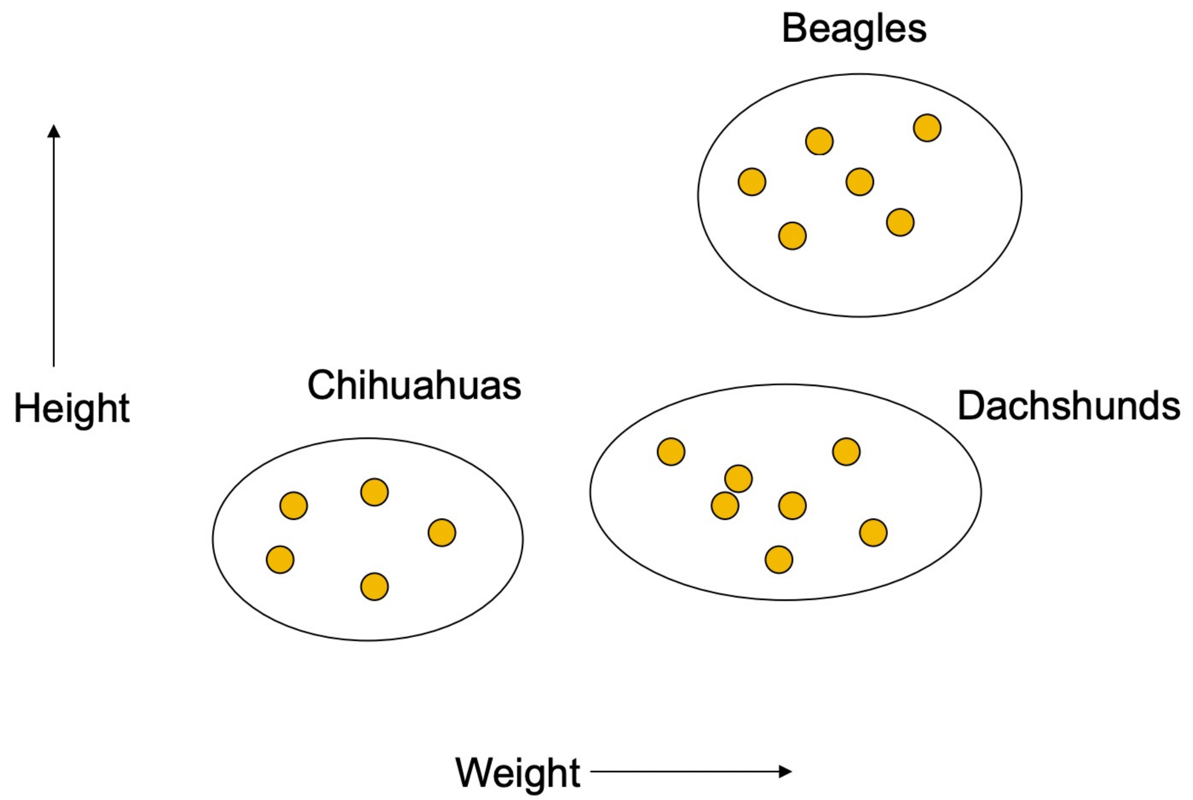
The Problem of Clustering

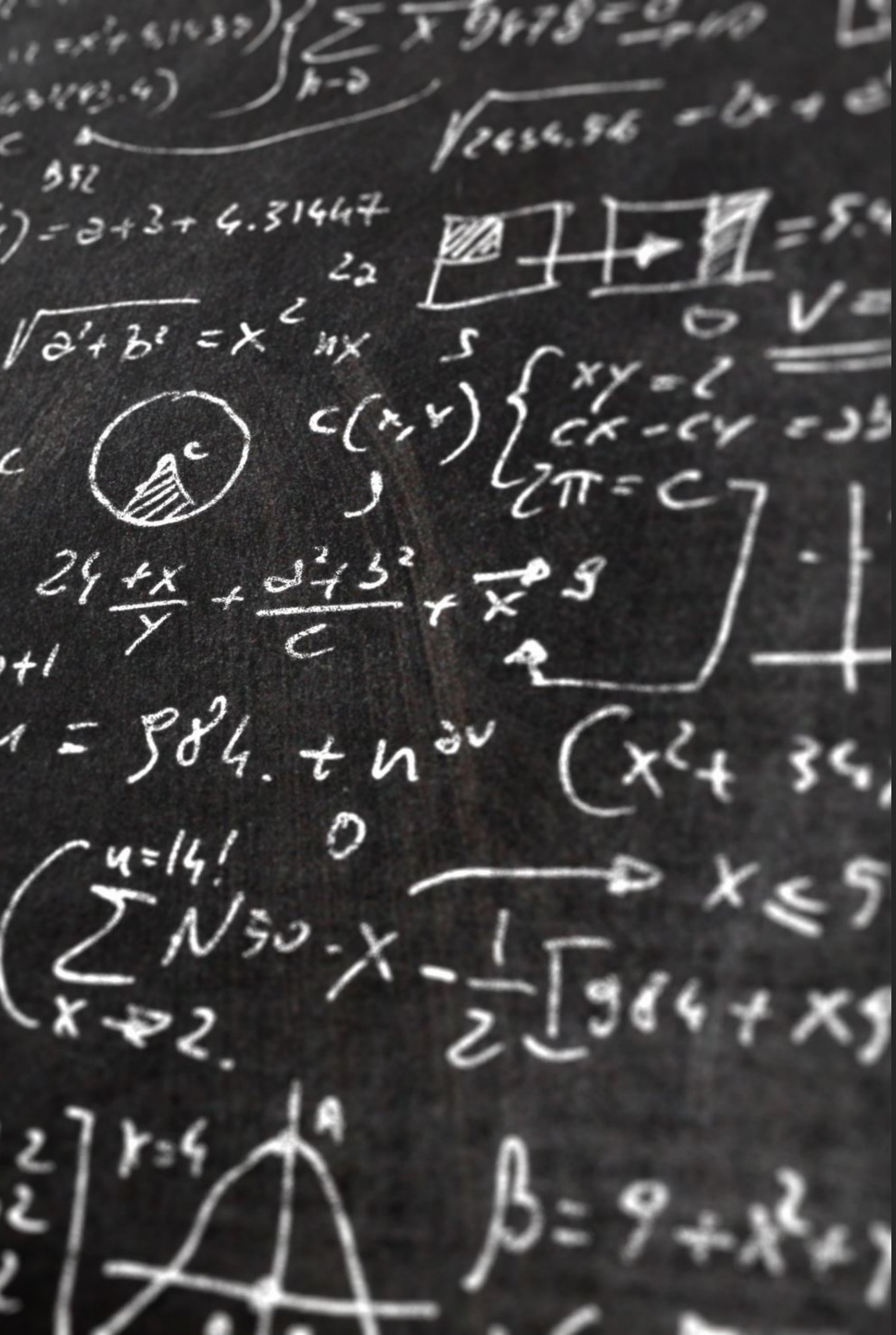
- Given a set of points, with a notion of distance between points, group the points into some number of clusters, so that
  - Members of the same cluster are close/similar to each other
  - Members of different clusters are dissimilar
- Usually:
  - Points are in a high-dimensional space
  - Similarity is defined using a distance measure such as Euclidean, Cosine, Jaccard etc

Example of Clusters and Outliers



Example: Doggie data





Clustering Problem: Documents

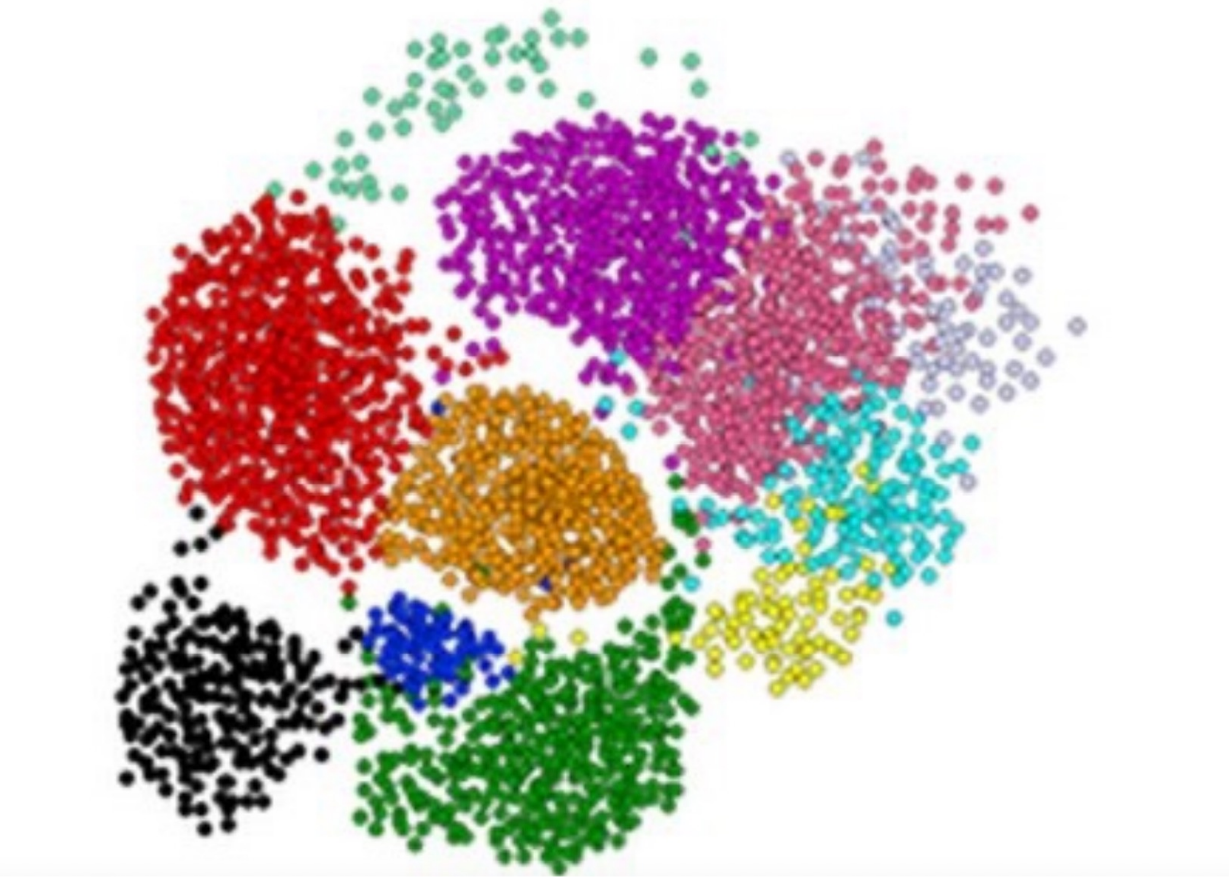
Finding topics:

- Represent a document by a vector $(x_1, x_2, \ldots, x_k)$, where $x_i = 1$ iff the i^{th} word (in some order) appears in the document
- Documents with similar sets of words may be about the same topic

More Examples

- Cluster customers based on their purchase histories
- Cluster products based on the sets of customers who purchased them
- Cluster DNA sequences based on edit distance

Clustering is hard!

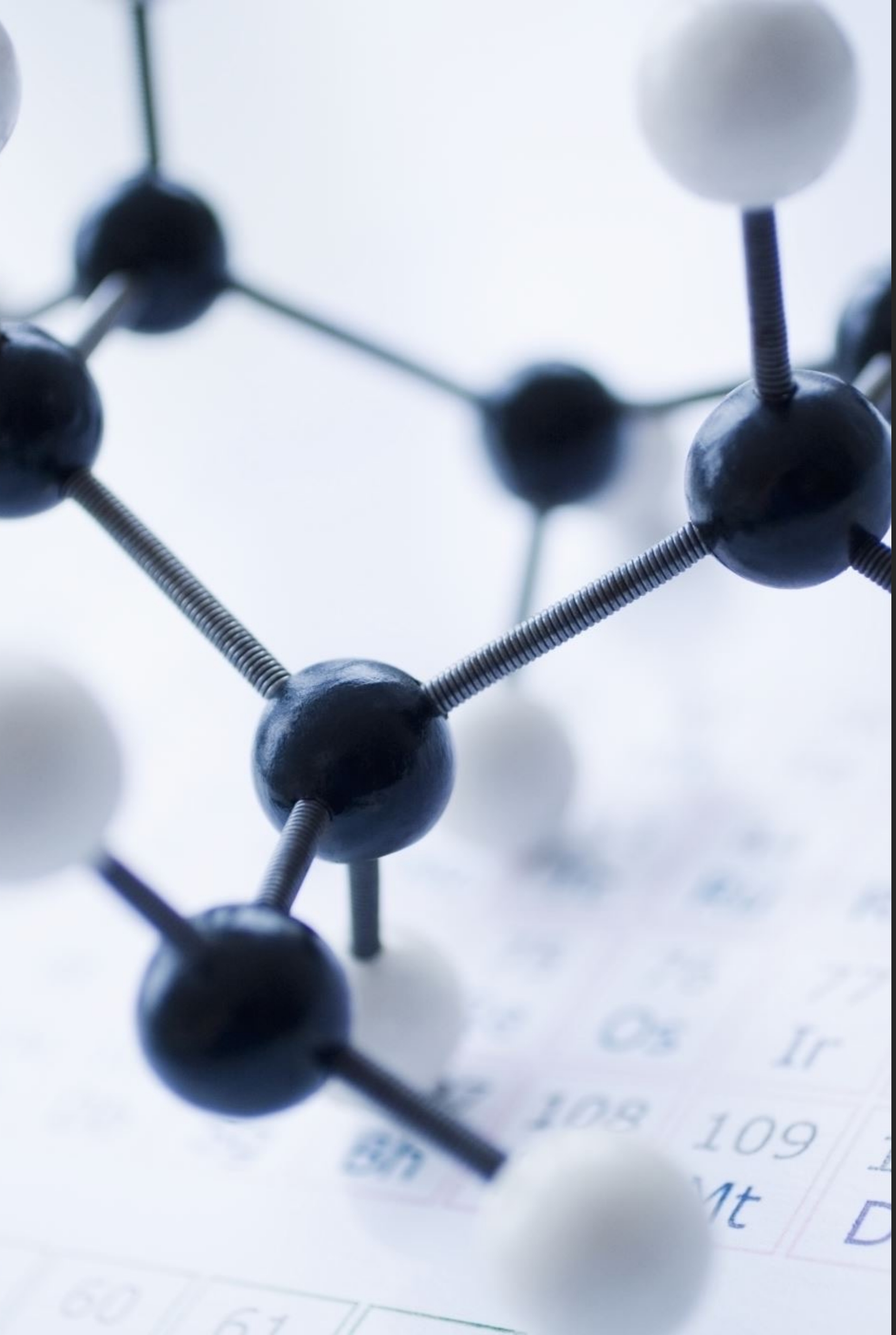


Why is it hard?

- Clustering in two dimensions looks easy
- Clustering small amounts of data looks easy
- Many applications involve not 2, but 10 or 10,000 dimensions
- High-dimensional spaces look different: Almost all pairs of points are very far from each other

Clustering algorithm of the day

- K-means clustering



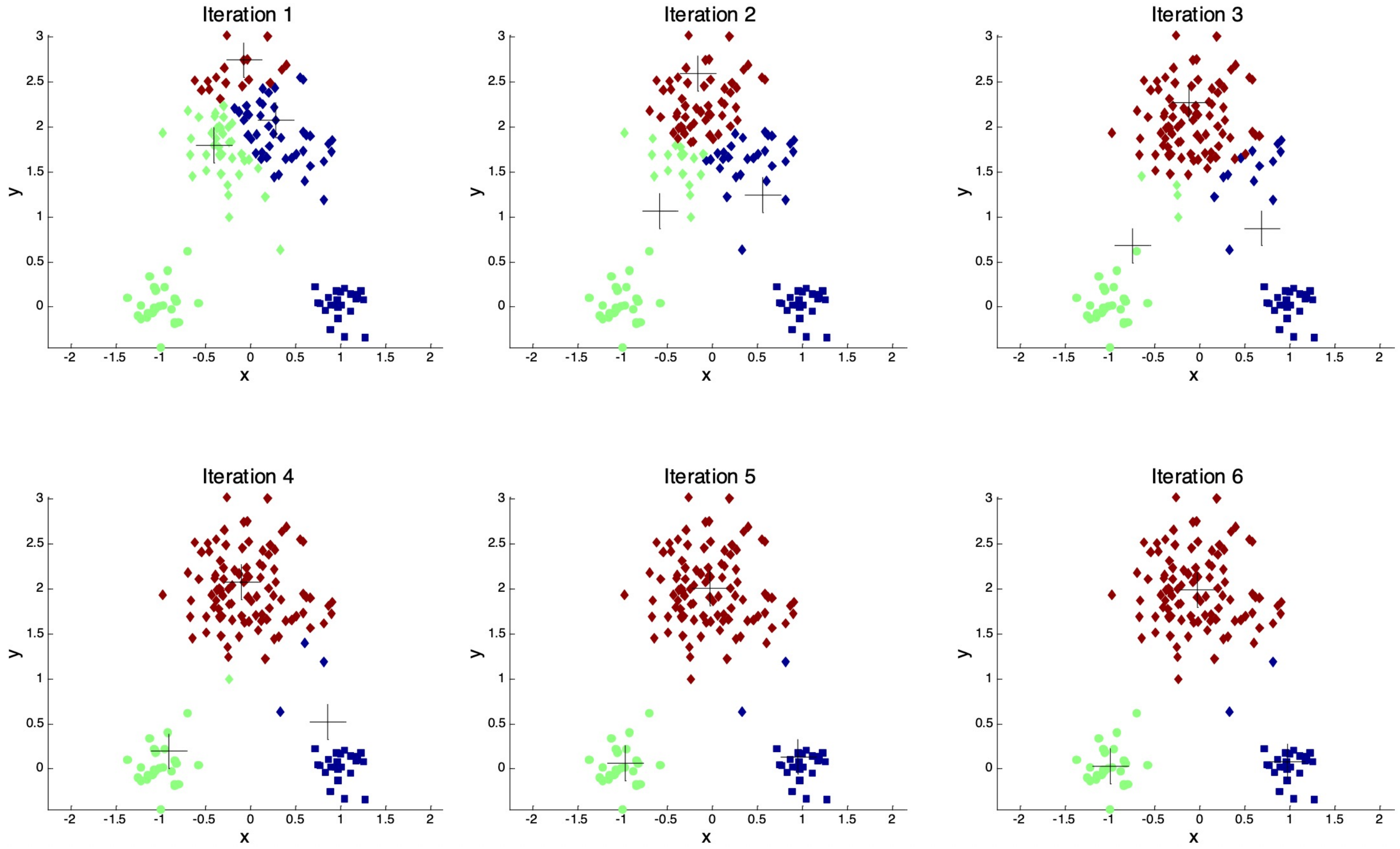
K-means Clustering

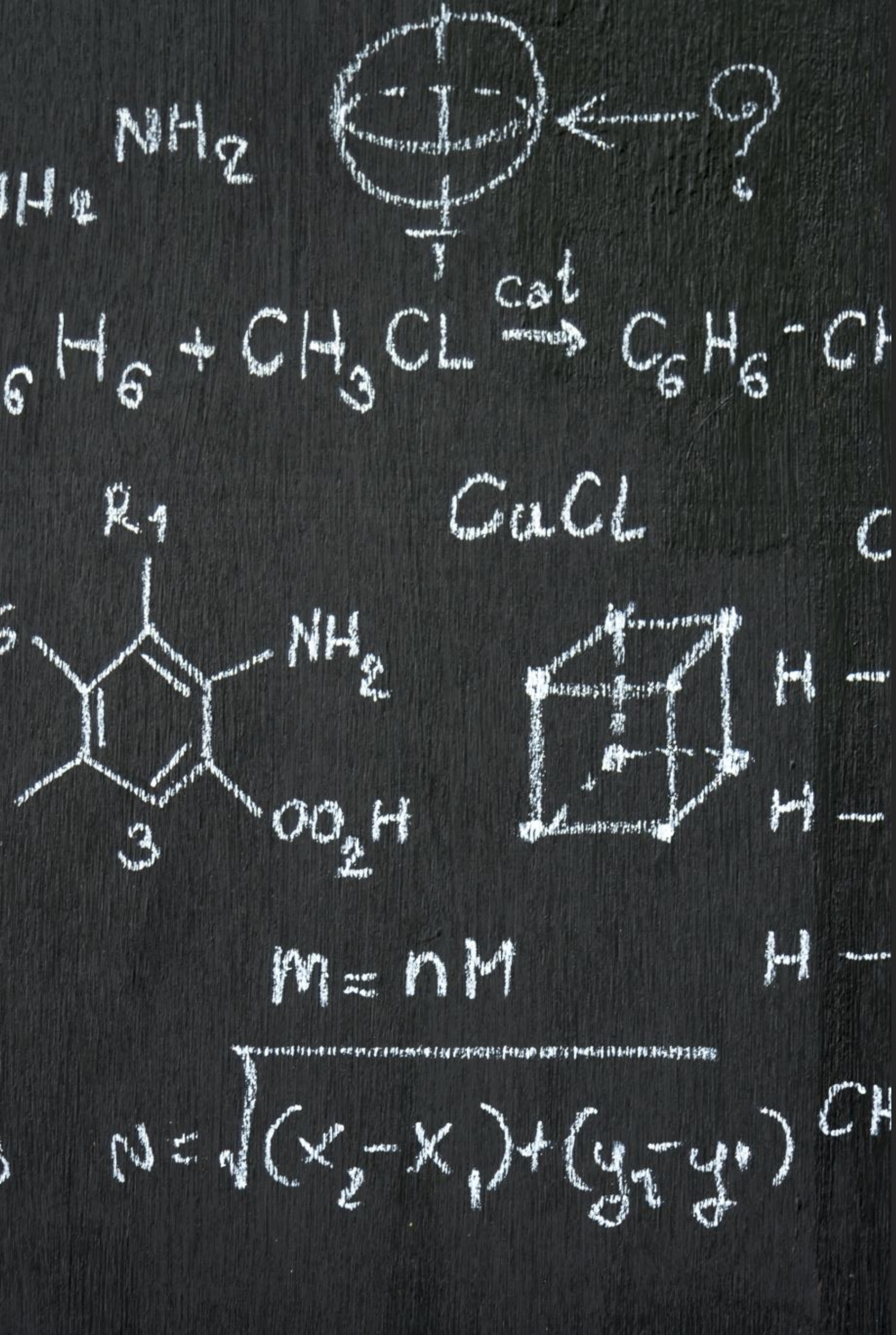
- Partitional clustering approach
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The number of clusters K, must be specified
- The basic algorithm is very simple

K-means Clustering Algorithm

Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change

Importance of Choosing Initial Centroids





K-means Clustering

- Given the data set $\{x_1, x_2. . . , x_N\}$ where each X_i is a D-dimensional vector.
- Our goal is to partition the data set into some number k of clusters.
- μ_k, where k = 1, . . . , K, in which μ_k is a prototype associated with the k^{th} cluster (representing the centers of the clusters).
- Our goal is then to find an assignment of data points to clusters, as well as a set of vectors $\{\mu_k\}$, such that the sum of the squares of the distances of each data point to its closest vector μ_k, is a minimum.

K-means Clustering

- For each data point x_n, we introduce a corresponding set of binary indicator variables $r_{nk} \in \{0, 1\}$, where k = 1, 2, . . ., K describing which of the K clusters the data point x_n is assigned to, so that if data point x_n is assigned to cluster k then $r_{nk} = 1$, and $r_{nj} = 0$ for $j = k$.

K-means Clustering

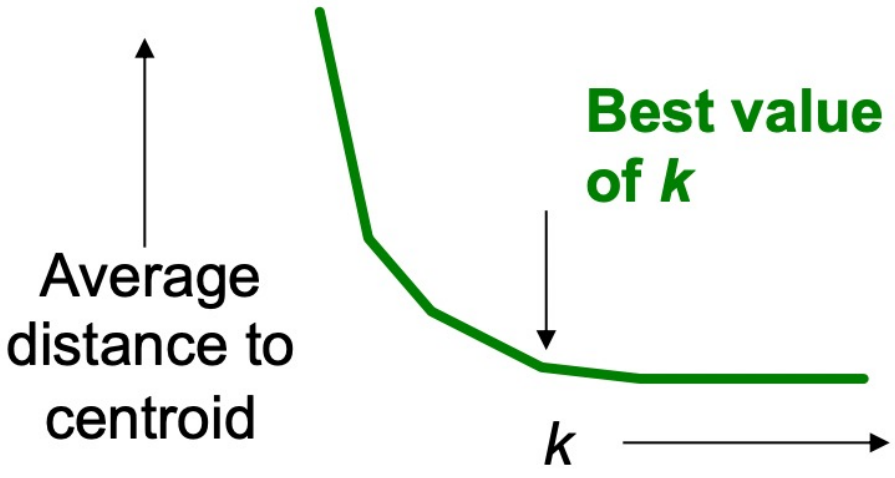
- We can then define an objective function, which represents the sum of the squares of the distances of each data point to its assigned vector μ_k

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- Our goal is to find values for the $\{r_{nk}\}$ and the $\{\mu_k\}$ so as to minimize J.

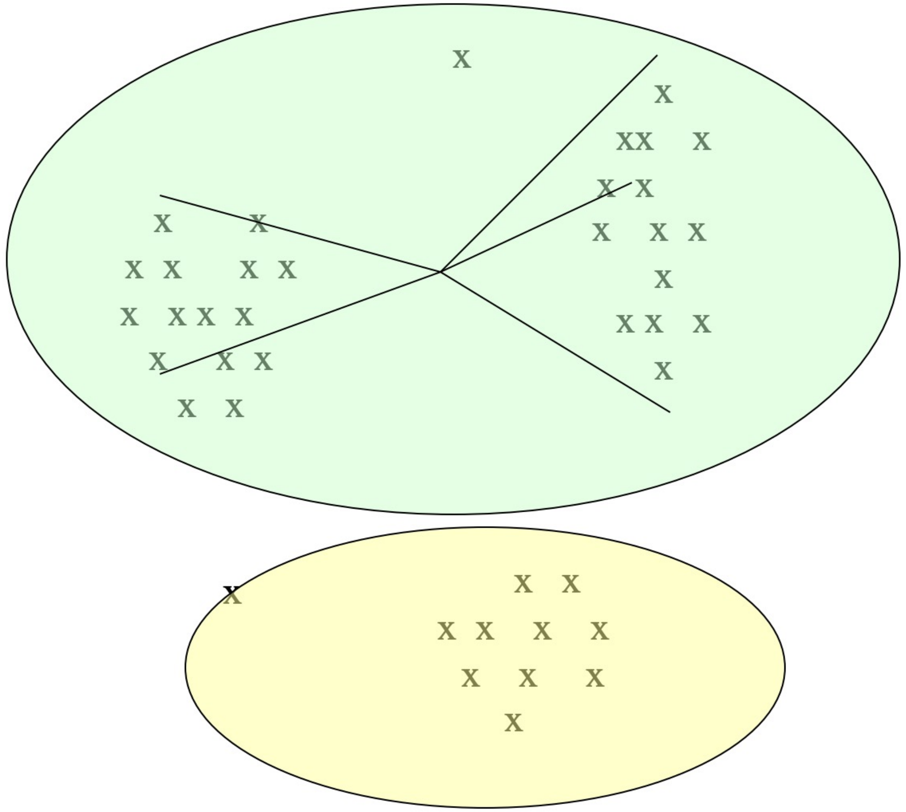
Importance of Choosing the Right k: The Elbow Method

- How to select k?
  - Try different k, looking at the change in the average distance to centroid as k increases
  - Average falls rapidly until right k, then changes little



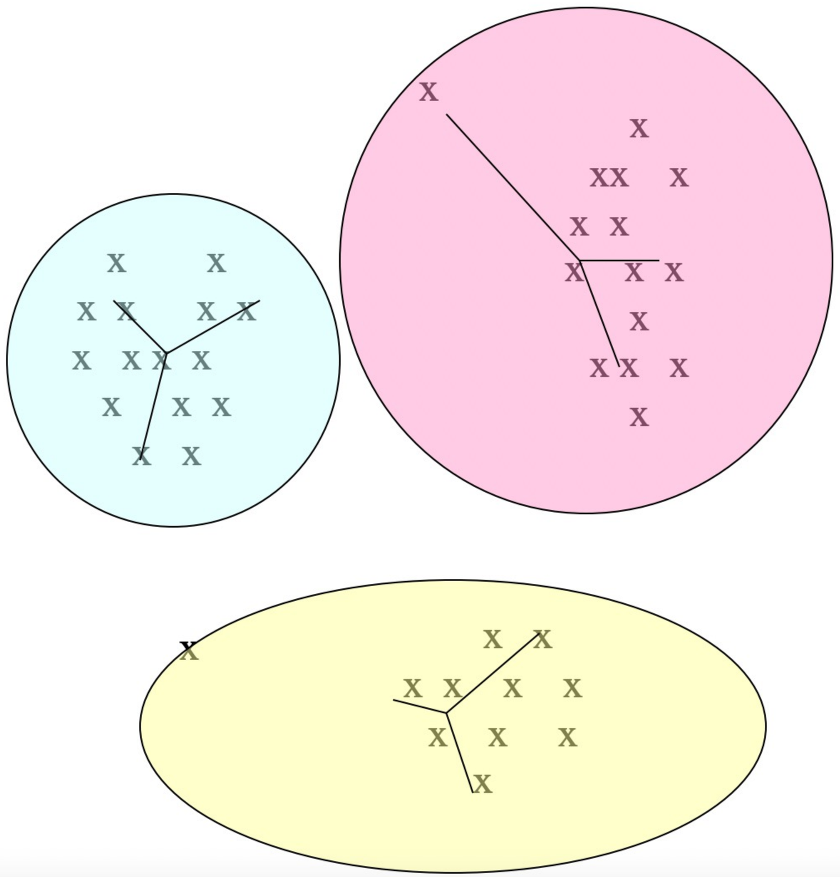
Example: Picking the Right k

Too few;
many long
distances
to centroid



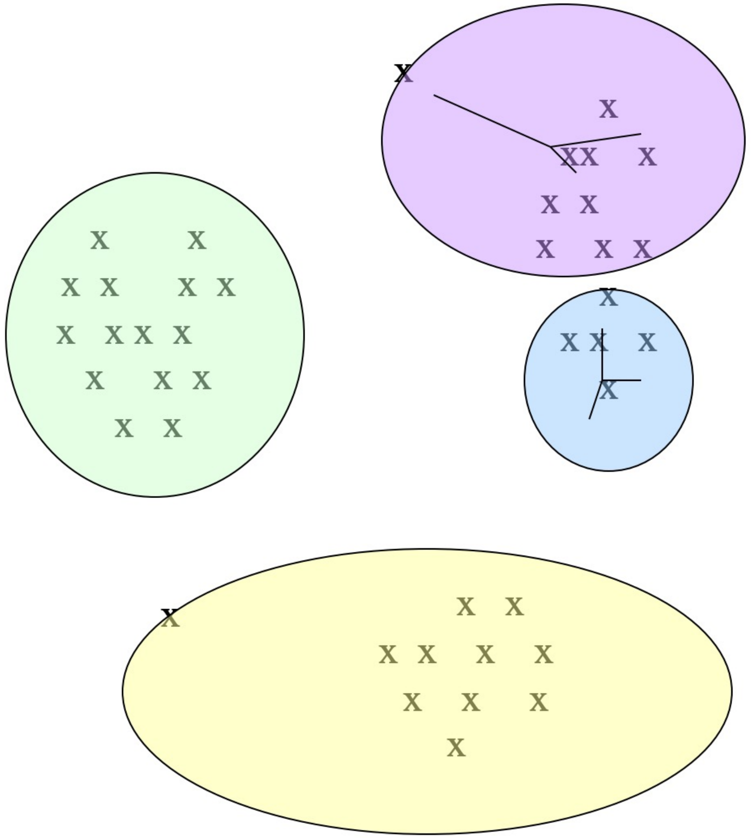
Example: Picking the Right k

Just right;
distances
rather short



Example: Picking the Right k

Too many;
little improvement
in average
distance



Pre-processing and Post-processing

Pre-processing

- Normalize the data
- Eliminate outliers

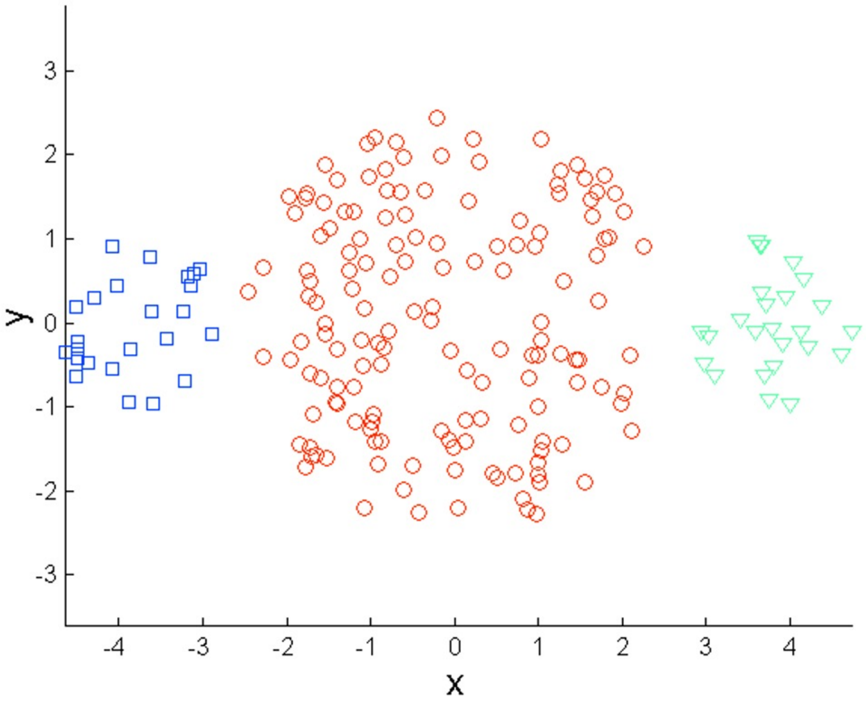
Post-processing

- Eliminate small clusters that may represent outliers
- Split ‘loose’ clusters, i.e., clusters with relatively high SSE
- Merge clusters that are ‘close’ and that have relatively low SSE

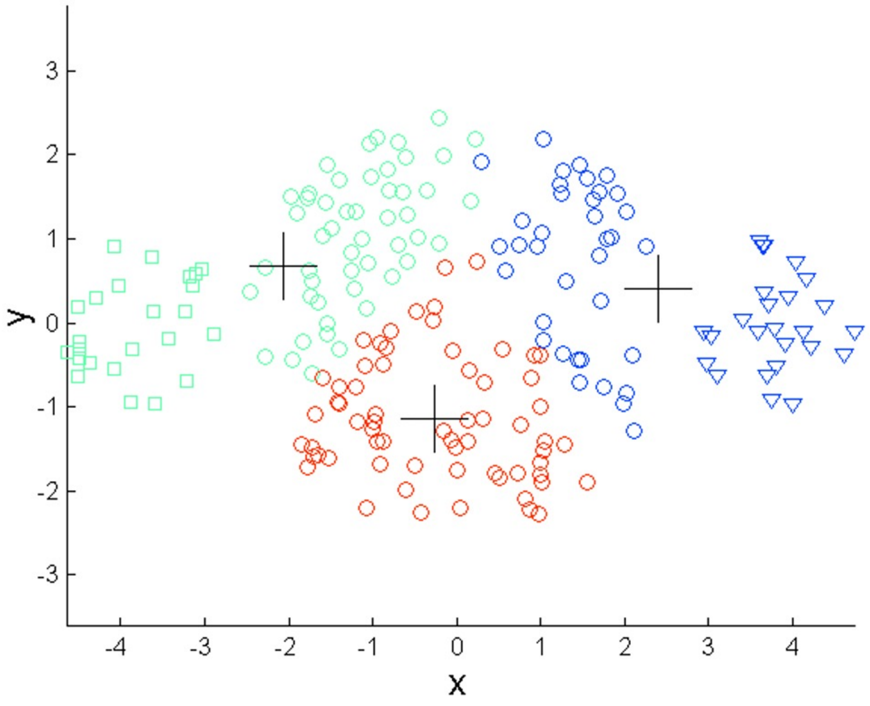
Limitations of K-means

- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

Limitations of K-means: Differing Sizes

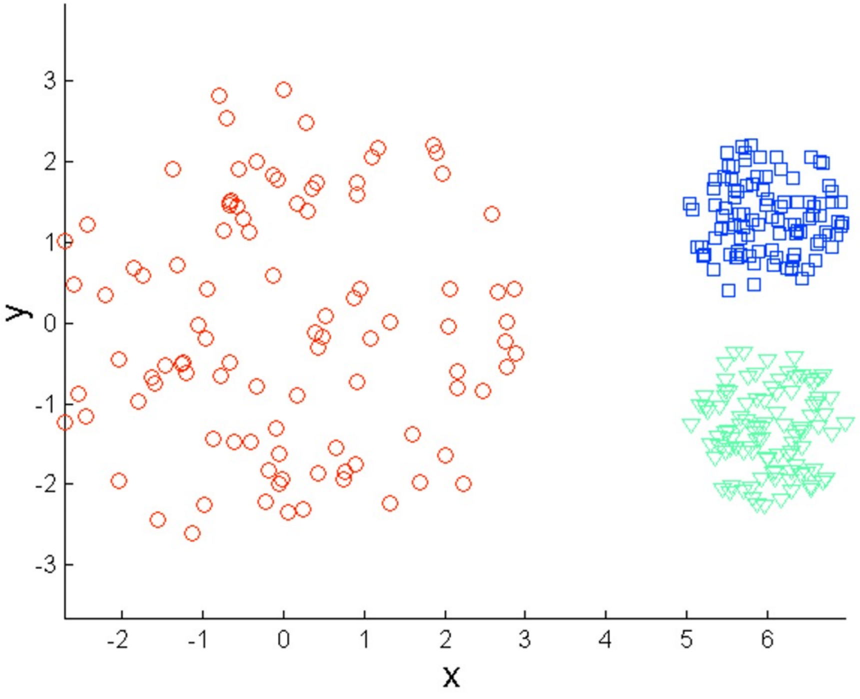


Original Points

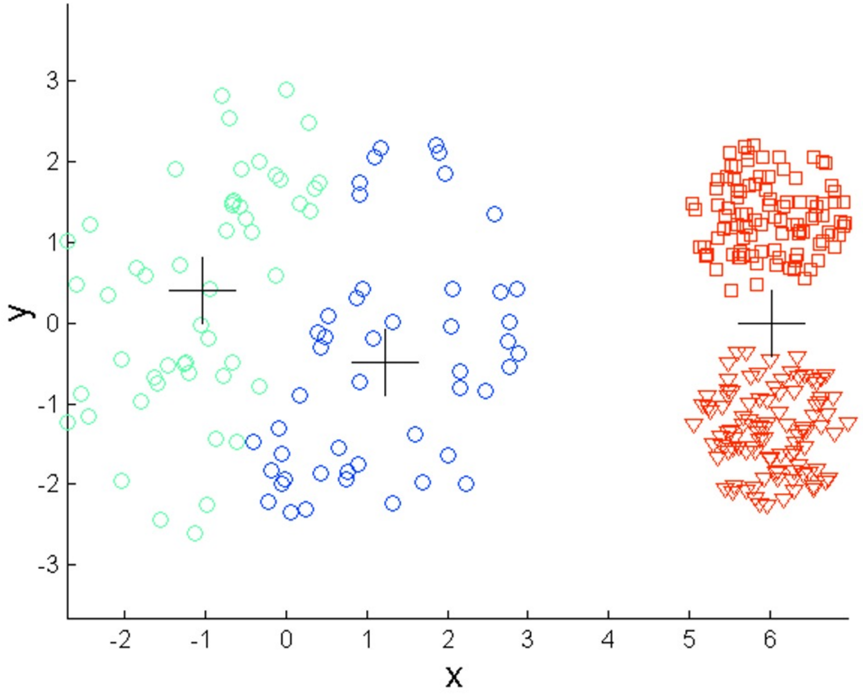


K-means (3 Clusters)

Limitations of K-means: Differing Density

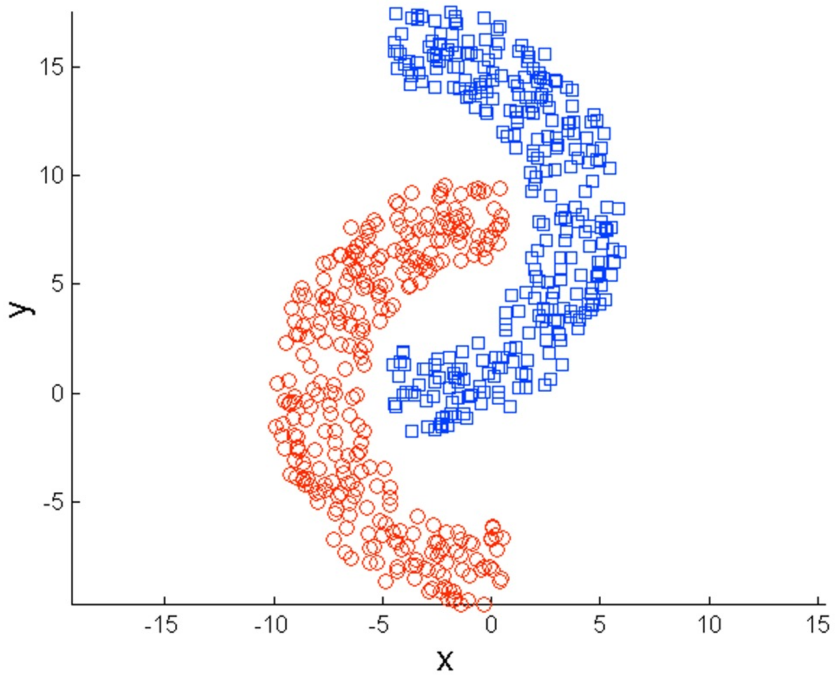


Original Points

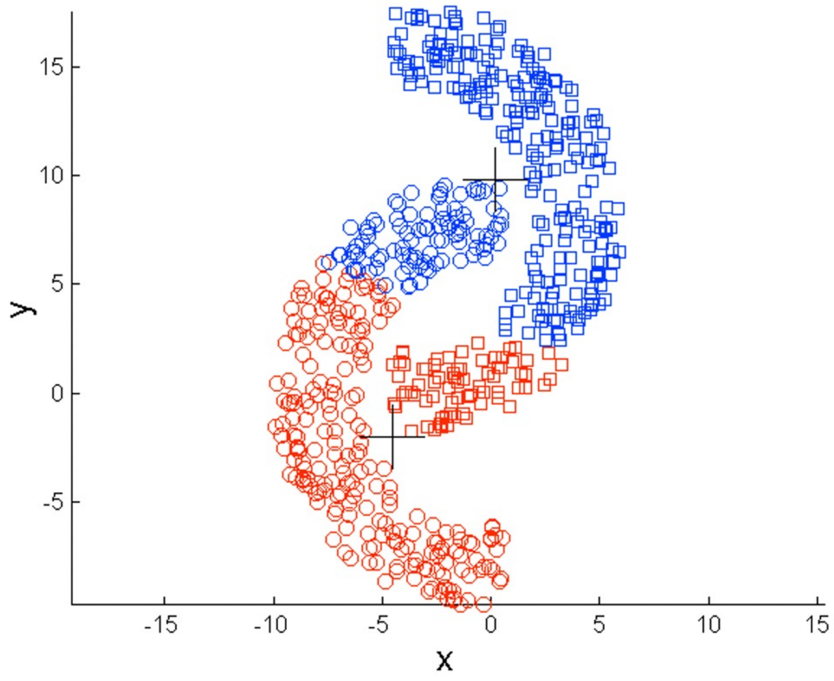


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



Original Points



K-means (2 Clusters)

Problems with K-Means Clustering

- K-Means Clustering works only for clusters which represent gaussian distributions. Hence, we cannot use K-Means clustering for finding complex clusters or non-convex clusters.
- The K-Means Algorithm is very sensitive to initialization, and hence one must be careful while initializing the cluster means.
- The Algorithm can get stuck at a local optima, finding clusters different from those originally wanted. This is also a factor affected by the initialization of the cluster means.

Some other clustering algorithms: Hierarchical clustering

- Agglomerative: Start with the points as individual clusters and at each step, merge the closest pair of clusters until only one cluster (or k clusters) left

- Divisive: Start with one, all-inclusive cluster and At each step, split a cluster until each cluster contains a point (or there are k clusters)

- Strengths: Do not have to assume a particular number of clusters. They may correspond to meaningful taxonomies

Implementation of K-means clustering from scratch with

- https://medium.com/nerd-for-tech/k-means-python-implementation-from-scratch-8400f30b8e5c
- https://analyticsarora.com/k-means-for-beginners-how-to-build-from-scratch-in-python/#K-means-from-Scratch-in-Python
- A more detailed explanation: https://towardsdatascience.com/create-your-own-k-means-clustering-algorithm-in-python-d7d4c9077670