



INTRODUCTION TO DATA ANALYTICS

Cluster Analysis

Asst. Prof. Dr. Rathachai Chawuthai

Department of Computer Engineering

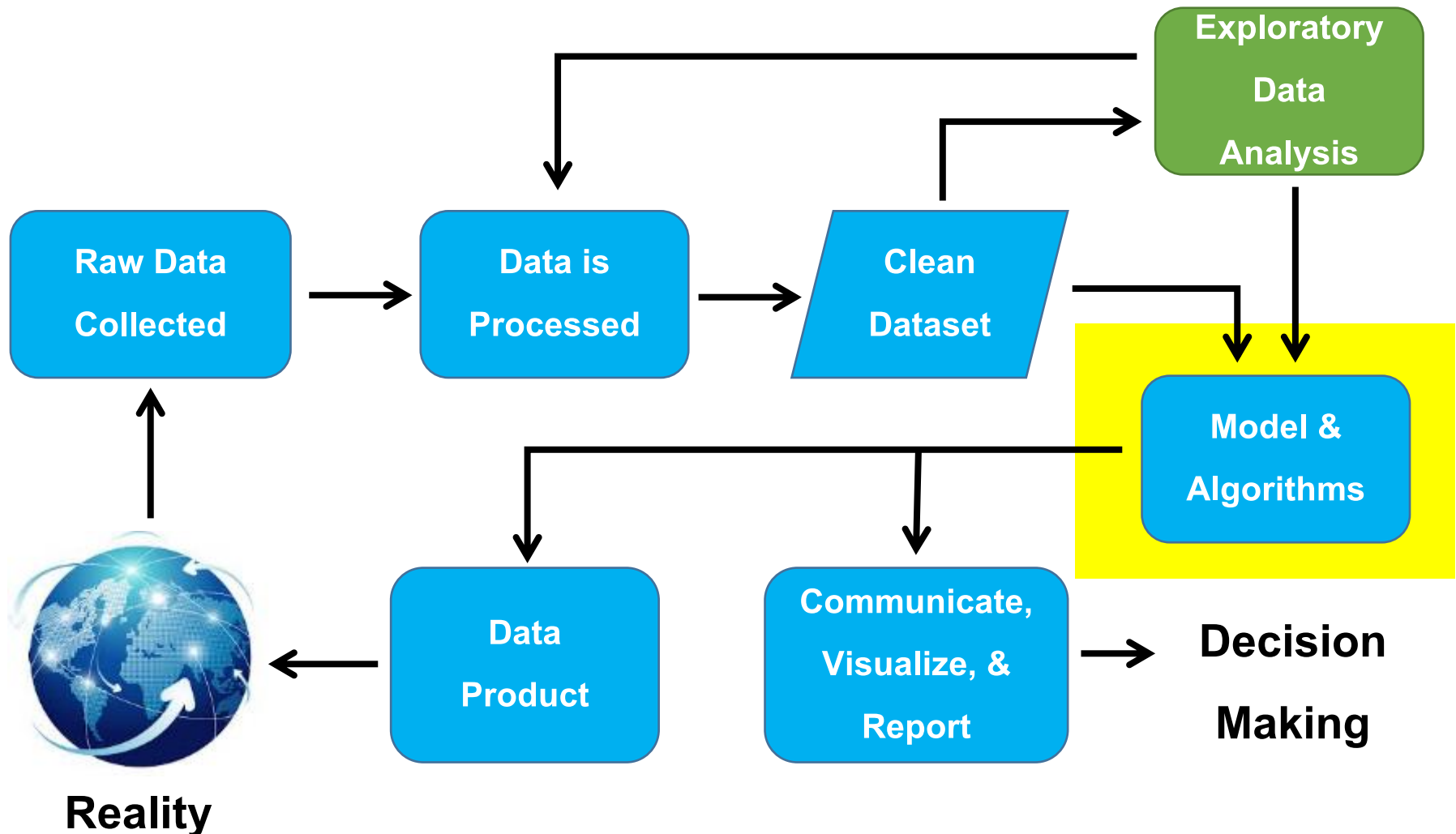
Faculty of Engineering

King Mongkut's Institute of Technology Ladkrabang

Agenda

- Clustering
- K-Means
- DBSCAN
- Cluster Validation
- Dimensionality Reduction

Data Science Process



Machine Learning

```
graph TD; ML[Machine Learning] --> SL[Supervised Learning]; ML --> UL[Unsupervised Learning]; SL --> R[Regression]; SL --> C[Classification]; UL --> Cl[Clustering]; R --- RList["• Linear Regression<br>• Polynomial Regression"]; C --- CList["• Decision Tree<br>• Logistic Regression<br>• Neural Network<br>• etc."]; Cl --- ClList["• K-Means<br>• DB-SCAN<br>• etc."];
```

Supervised Learning

Develop predictive model based on both input and output data

Regression

- Linear Regression
- Polynomial Regression

Classification

- Decision Tree
- Logistic Regression
- Neural Network
- etc.

Unsupervised Learning

Develop predictive model based on both input and output data

Clustering

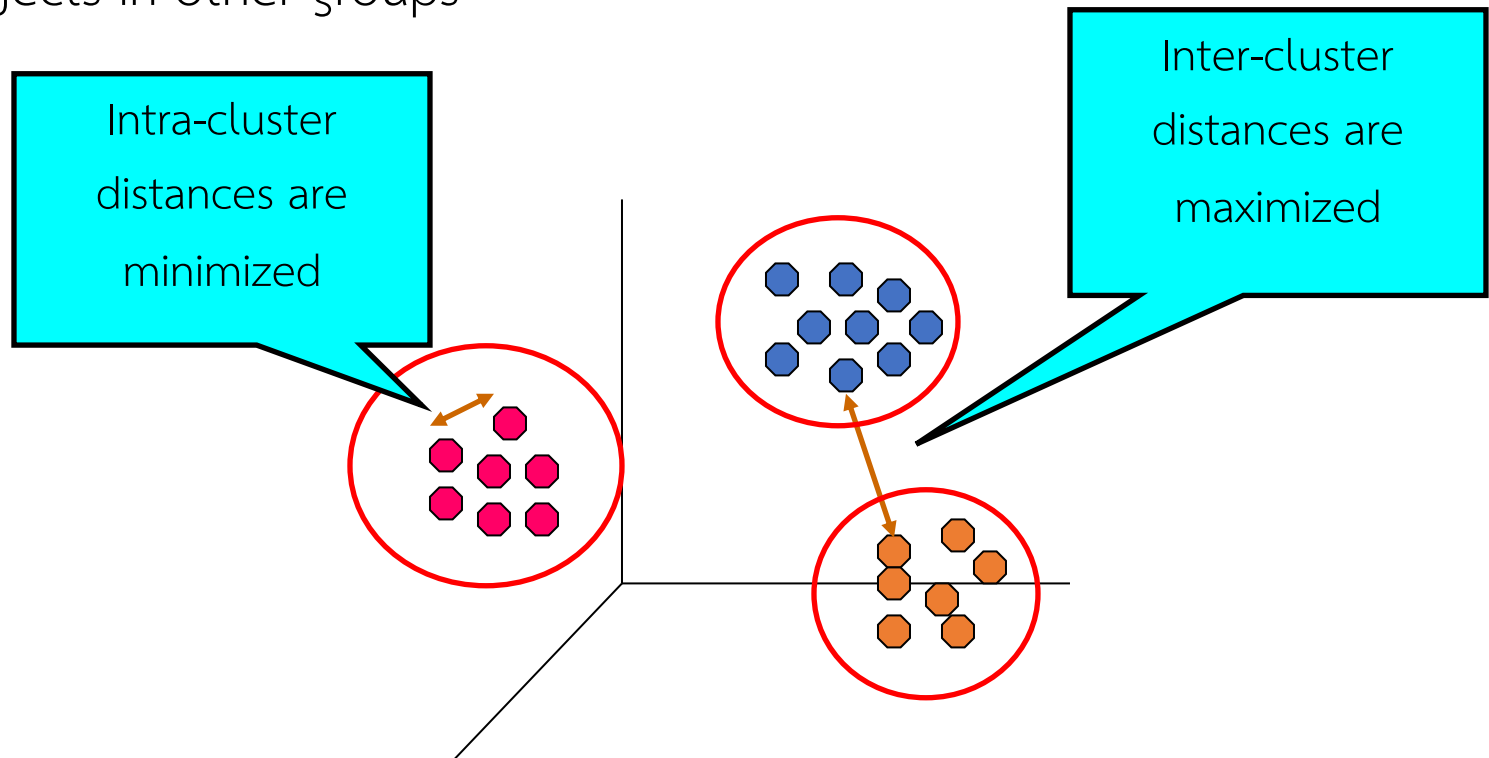
- K-Means
- DB-SCAN
- etc.

Clustering

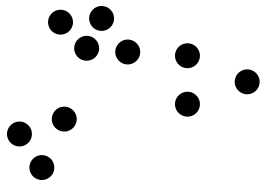
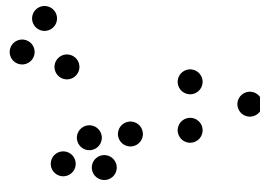


What is a Clustering?

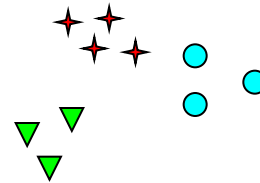
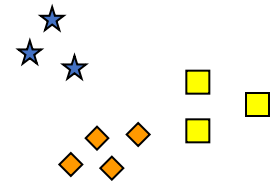
- In general a grouping of objects such that the objects in a group (cluster) are similar (or related) to one another and different from (or unrelated to) the objects in other groups



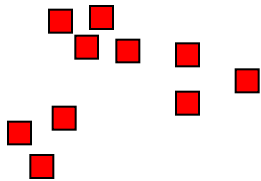
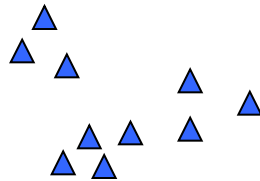
Notion of a Cluster can be Ambiguous



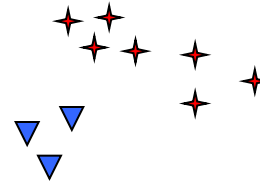
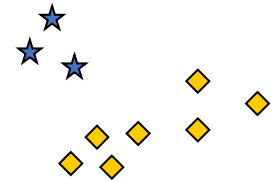
How many clusters?



6 Clusters



2 Clusters

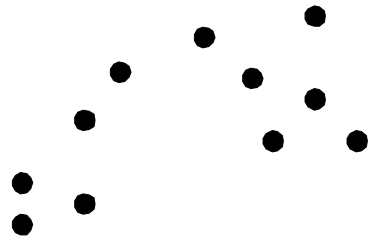
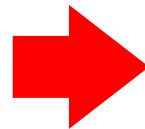
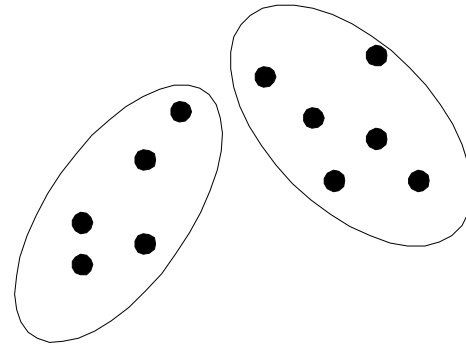
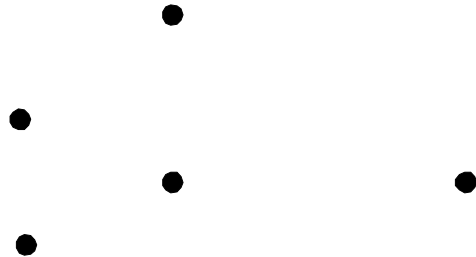
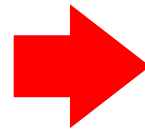
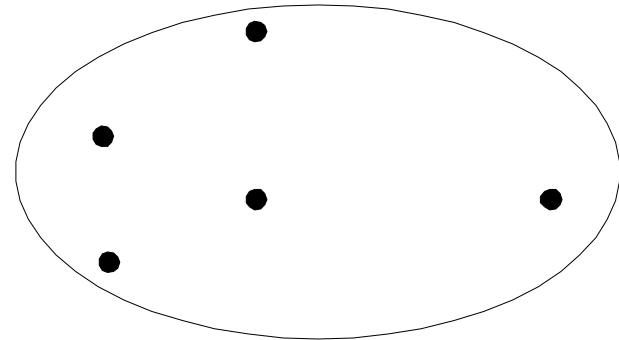


4 Clusters

Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- **Partitional Clustering**
 - A division data objects into subsets (clusters) such that each data object is in exactly one subset
- **Hierarchical clustering**
 - A set of nested clusters organized as a hierarchical tree

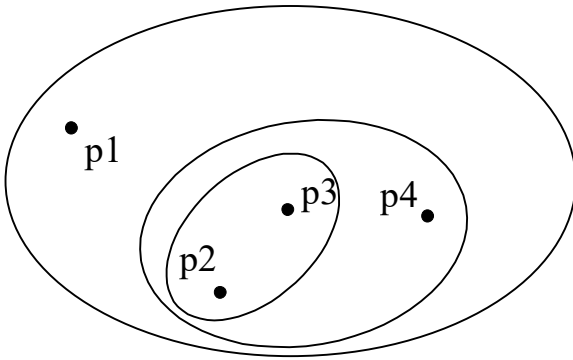
Partitional Clustering



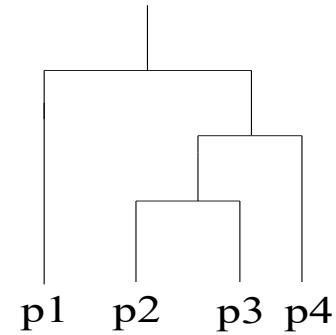
Original Points

A Partitional Clustering

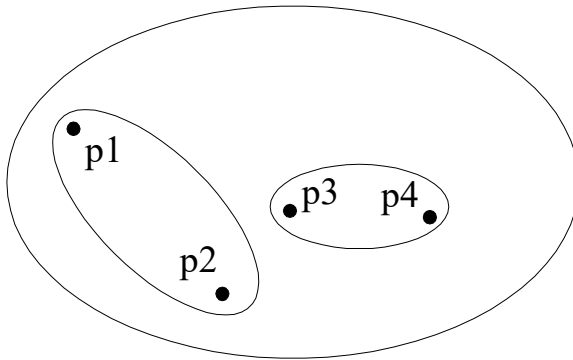
Hierarchical Clustering



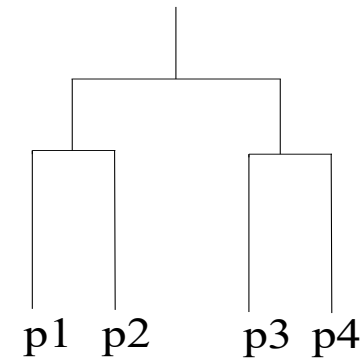
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Other types of clustering

- **Exclusive** (or non-overlapping) versus **non-exclusive** (or overlapping)
 - In non-exclusive clusterings, points may belong to multiple clusters.
 - Points that belong to multiple classes, or ‘border’ points
- **Fuzzy** (or soft) versus **non-fuzzy** (or hard)
 - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
 - Weights usually must sum to 1 (often interpreted as probabilities)
- **Partial** versus **complete**
 - In some cases, we only want to cluster some of the data

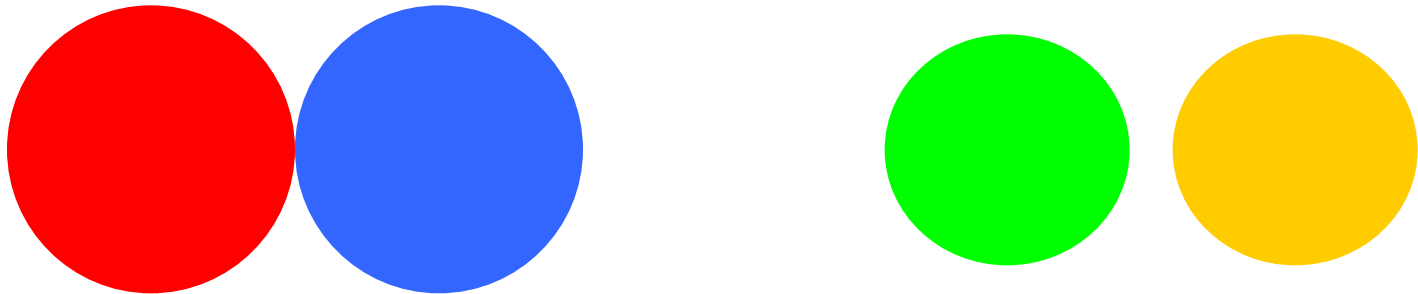
Types of Clusters: Well-Separated

- Well-Separated Clusters:
 - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



Types of Clusters: Center-Based

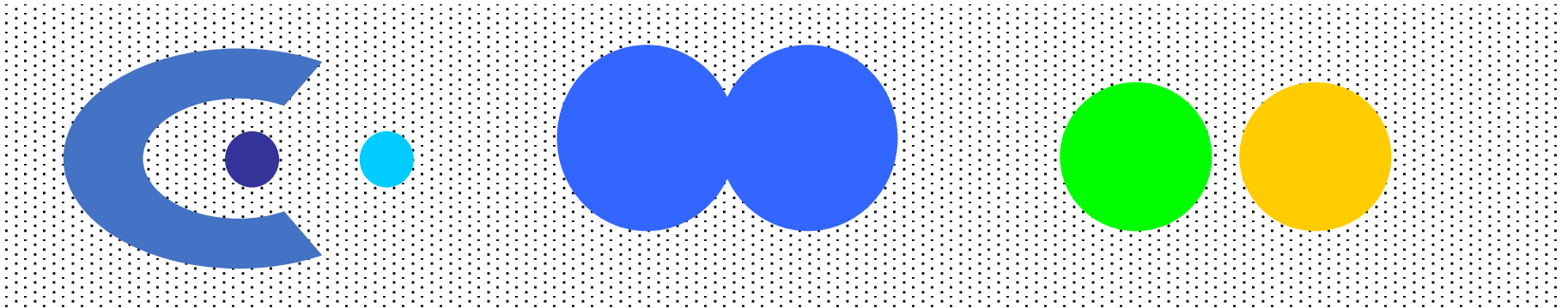
- Center-based
 - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the “center” of a cluster, than to the center of any other cluster
 - The center of a cluster is often a centroid, the minimizer of distances from all the points in the cluster, or a medoid, the most “representative” point of a cluster



4 center-based clusters

Types of Clusters: Density-Based

- Density-based
 - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
 - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Clustering Algorithms

- **K-Means**
- **DBSCAN**
- Hierarchical clustering
- PAM, CLARANS: Solutions for the **k-medoids** problem
- BIRCH: Constructs a **hierarchical tree** that acts a summary of the data, and then clusters the leaves.
- MST: Clustering using the **Minimum Spanning Tree**.
- ROCK: clustering **categorical data** by neighbor and link analysis
- LIMBO, COOLCAT: Clustering **categorical data** using **information theoretic** tools.
- CURE: **Hierarchical** algorithm uses different representation of the cluster
- CHAMELEON: **Hierarchical** algorithm uses **closeness and interconnectivity** for merging

K-Means



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
- Number of clusters, **K**, must be specified
- The objective is to **minimize the sum of distances** of the points to their respective **centroid**

K-means Clustering

- **Problem:** Given a set X of n points in a d -dimensional space and an integer K group the points into K clusters $C = \{C_1, C_2, \dots, C_k\}$ such that

$$Cost(C) = \sum_{i=1}^k \sum_{x \in C_i} dist(x, c_i)$$

is minimized, where c_i is the centroid of the points in cluster C_i

K-means Algorithm

- Also known as **Lloyd's algorithm**.
- K-means is sometimes synonymous with this algorithm

1: Select K points as the initial centroids.

2: **repeat**

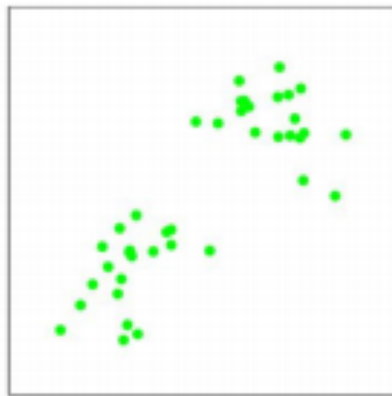
3: Form K clusters by assigning all points to the closest centroid.

4: Recompute the centroid of each cluster.

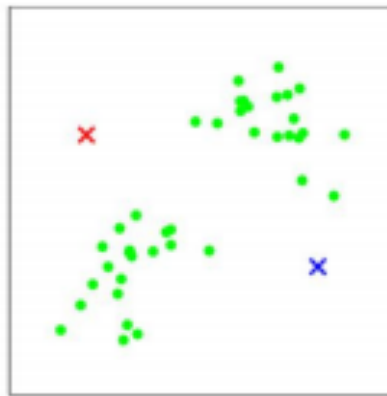
5: **until** The centroids don't change

- Initial centroids are often chosen **randomly**.
 - Clusters produced vary from one run to another.

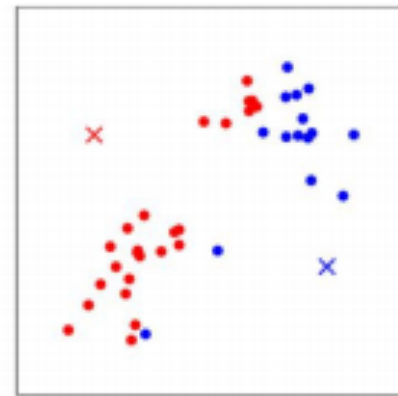
K-means: Steps



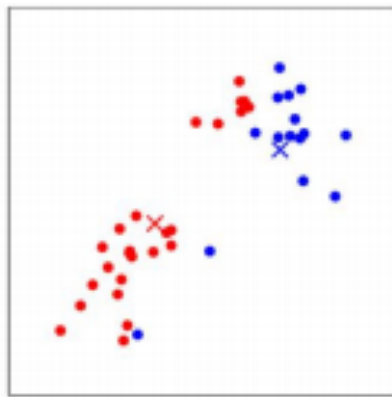
(a)



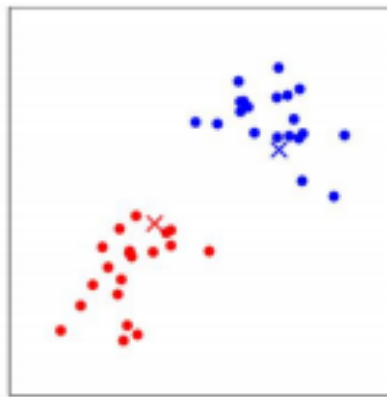
(b)



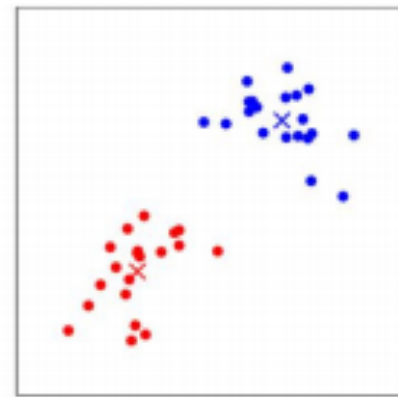
(c)



(d)

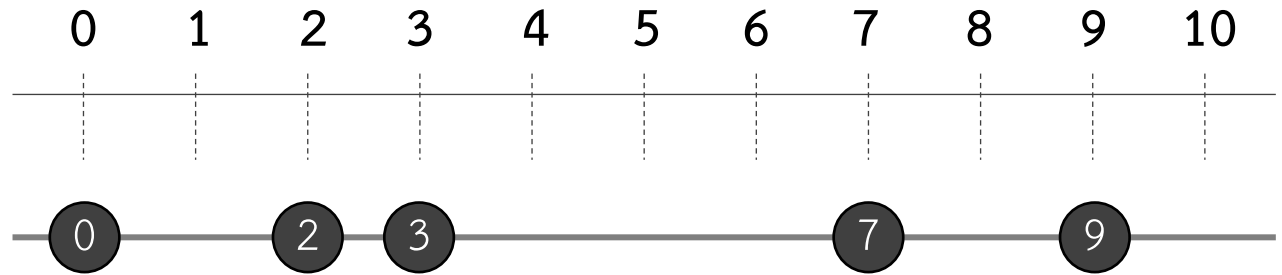


(e)



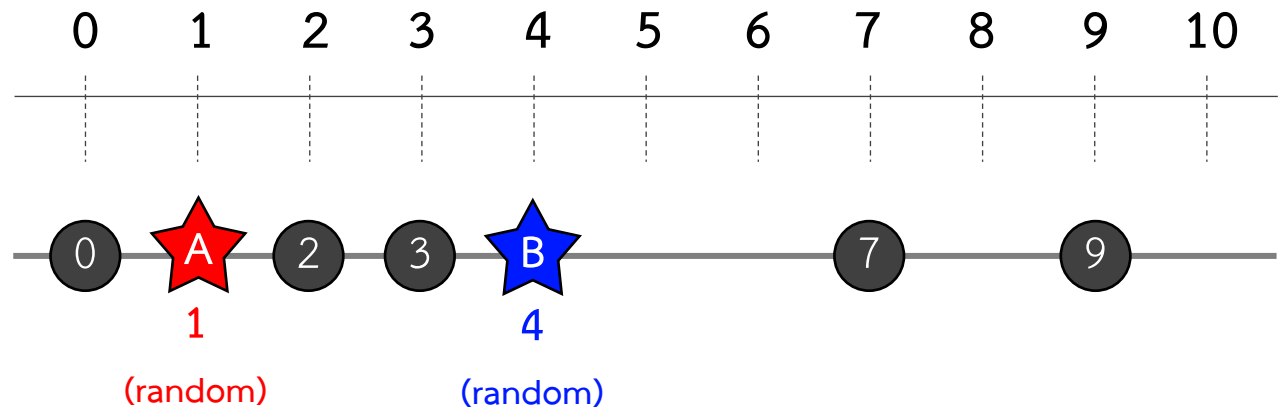
(f)

K-means: Example Steps (1 feature)



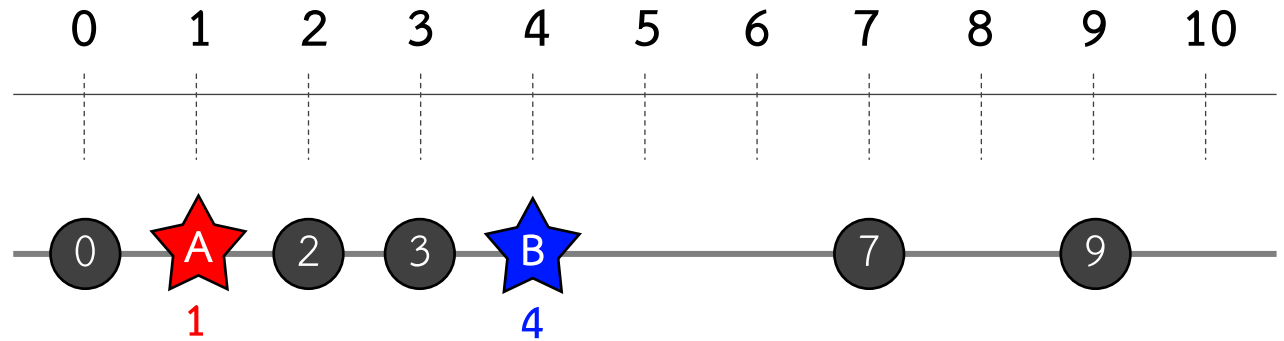
K-means: Example Steps (1 feature)

Round 0



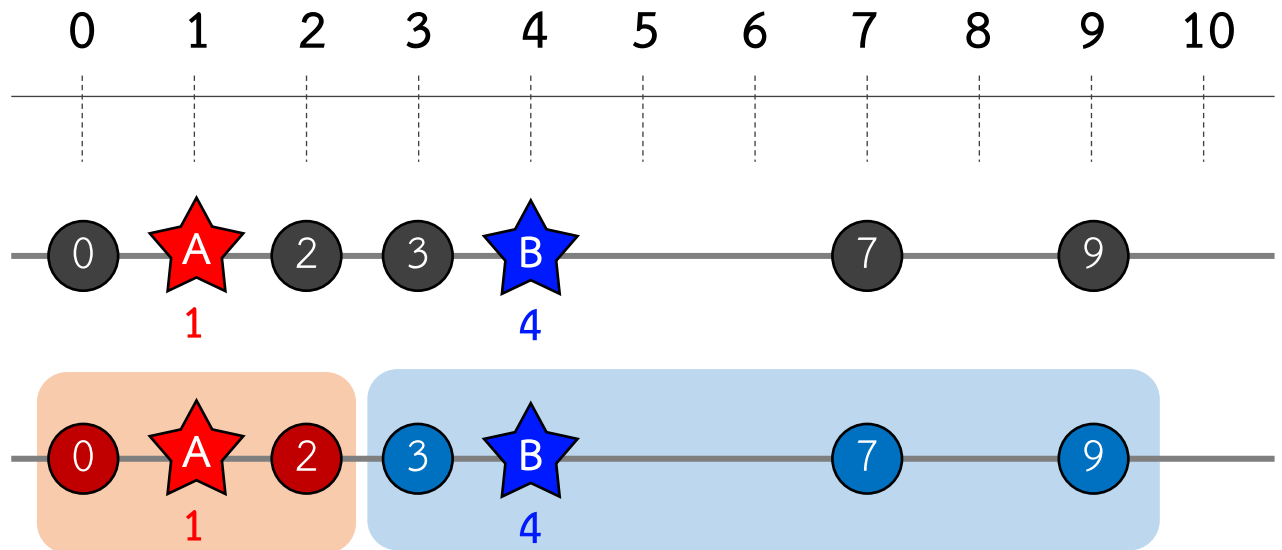
K-means: Example Steps (1 feature)

Round 1



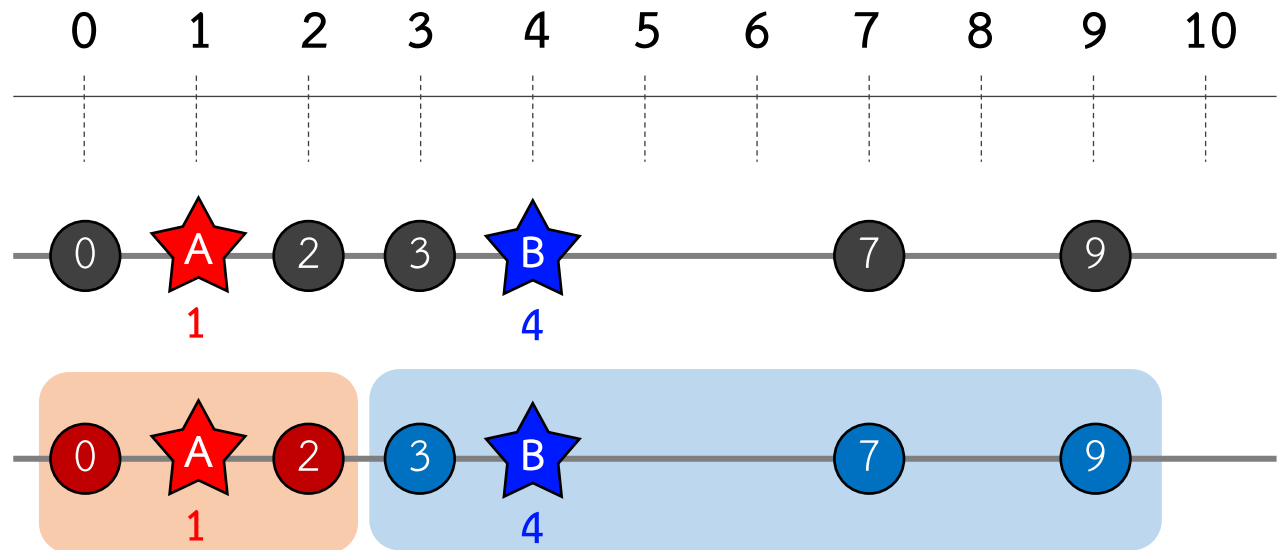
K-means: Example Steps (1 feature)

Round 1



K-means: Example Steps (1 feature)

Round 1

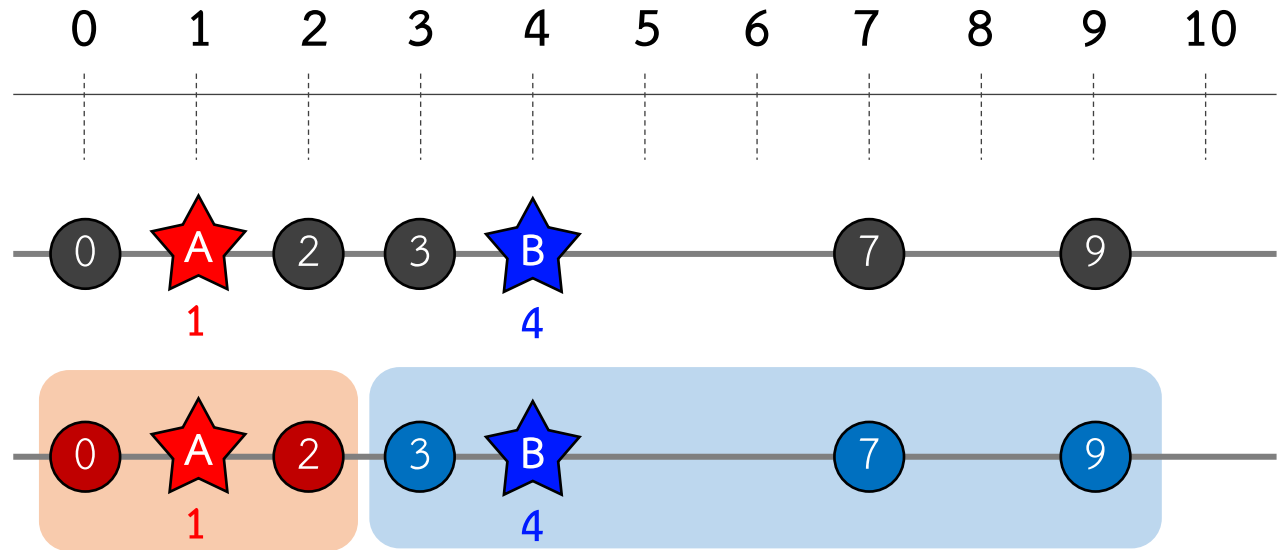


New Center of A = $(0+2)/2 = 1.0$

New Center of B = $(3+7+9)/3 = 6.3$

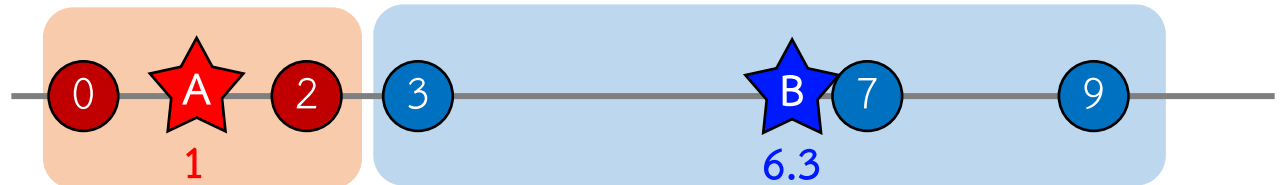
K-means: Example Steps (1 feature)

Round 1

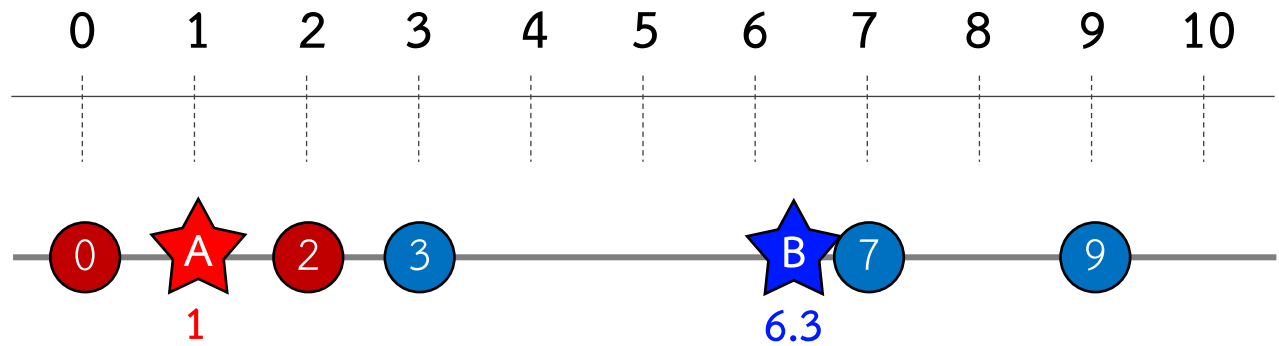


New Center of A = $(0+2)/2 = 1.0$

New Center of B = $(3+7+9)/3 = 6.3$

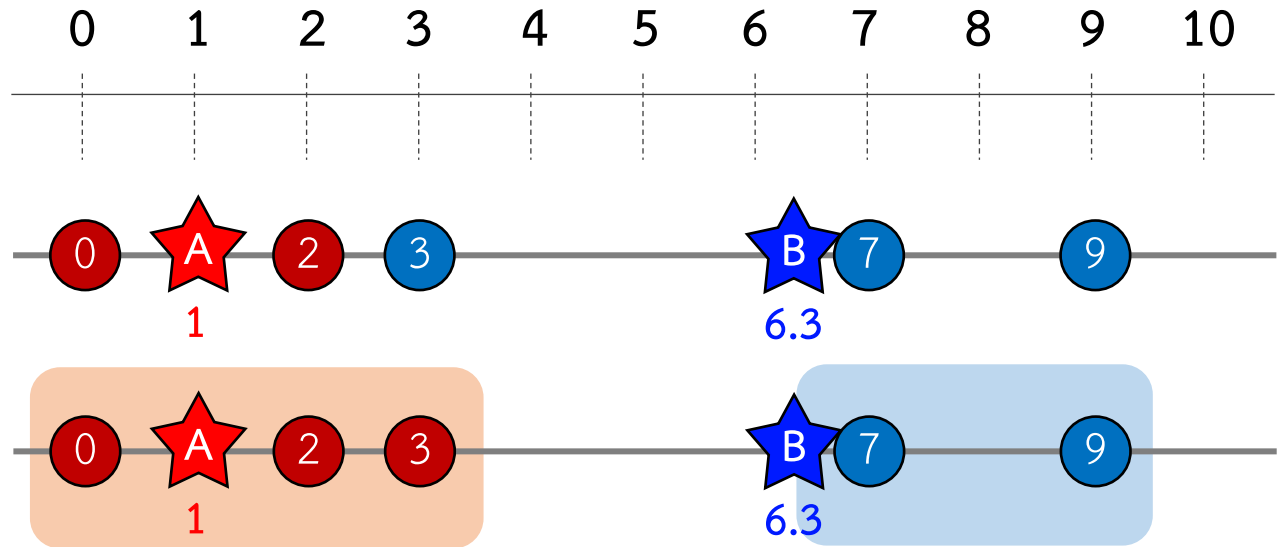


K-means: Example Steps (1 feature)



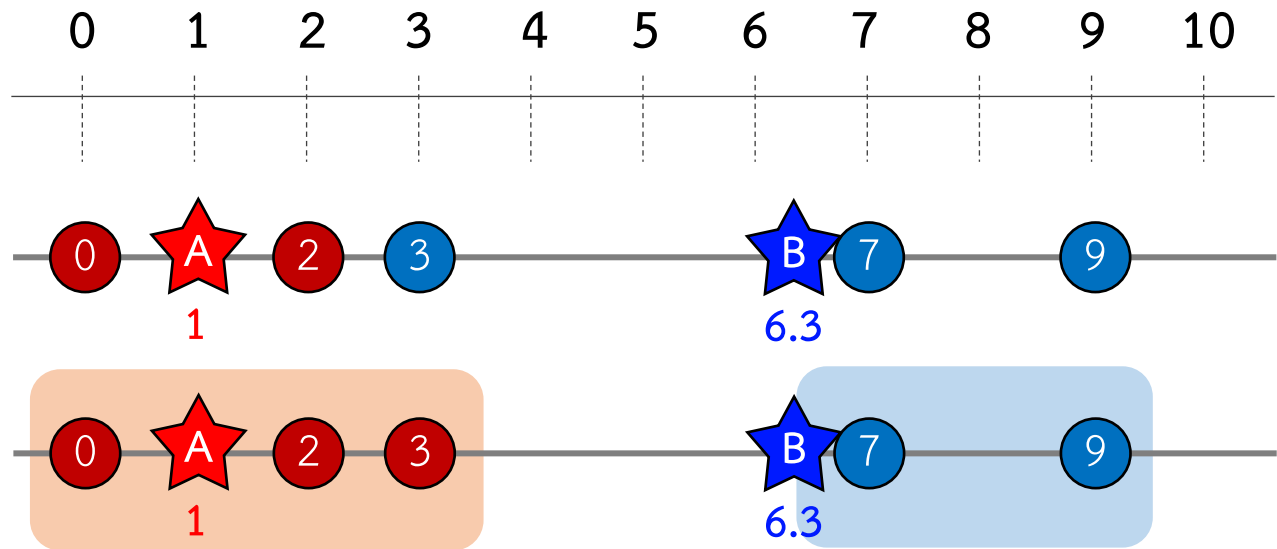
Round 2

K-means: Example Steps (1 feature)



Round 2

K-means: Example Steps (1 feature)



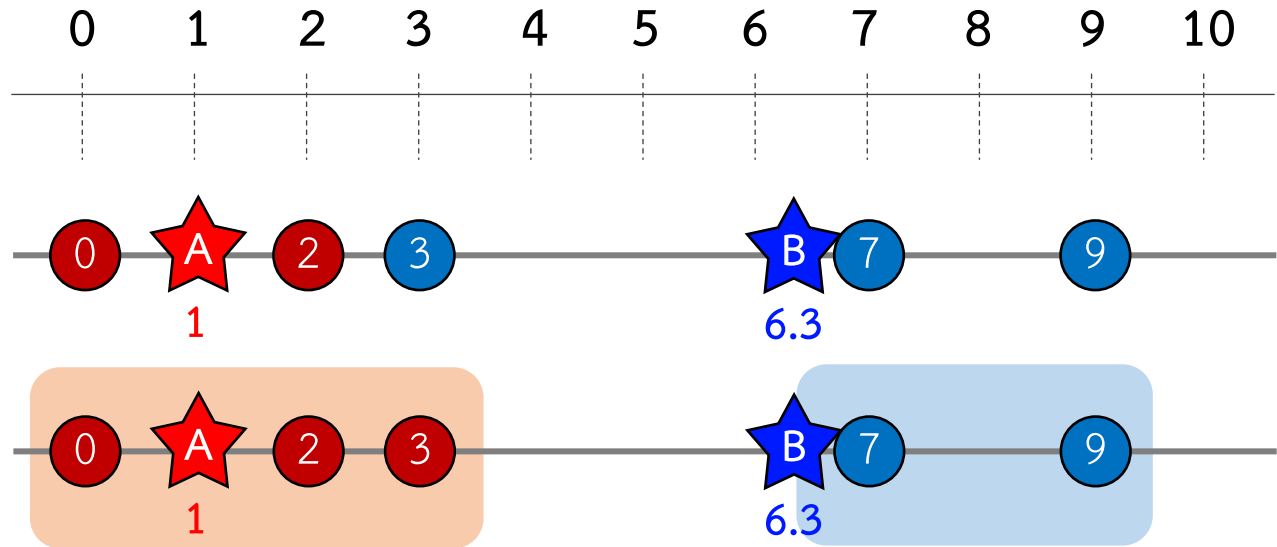
Round 2

$$\text{New Center of A} = (0+2+3)/3 = 1.7$$

$$\text{New Center of B} = (7+9)/2 = 8$$

K-means: Example Steps (1 feature)

Round 2

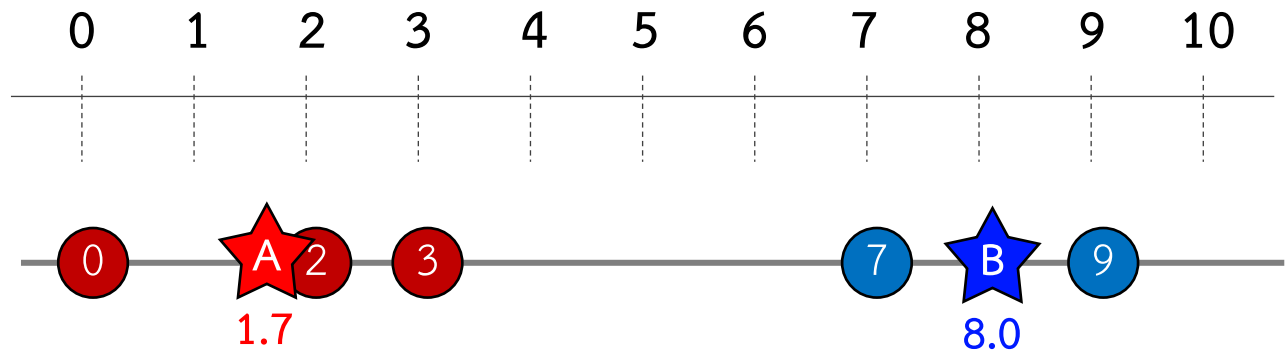


New Center of A = $(0+2+3)/3 = 1.7$

New Center of B = $(7+9)/2 = 8$

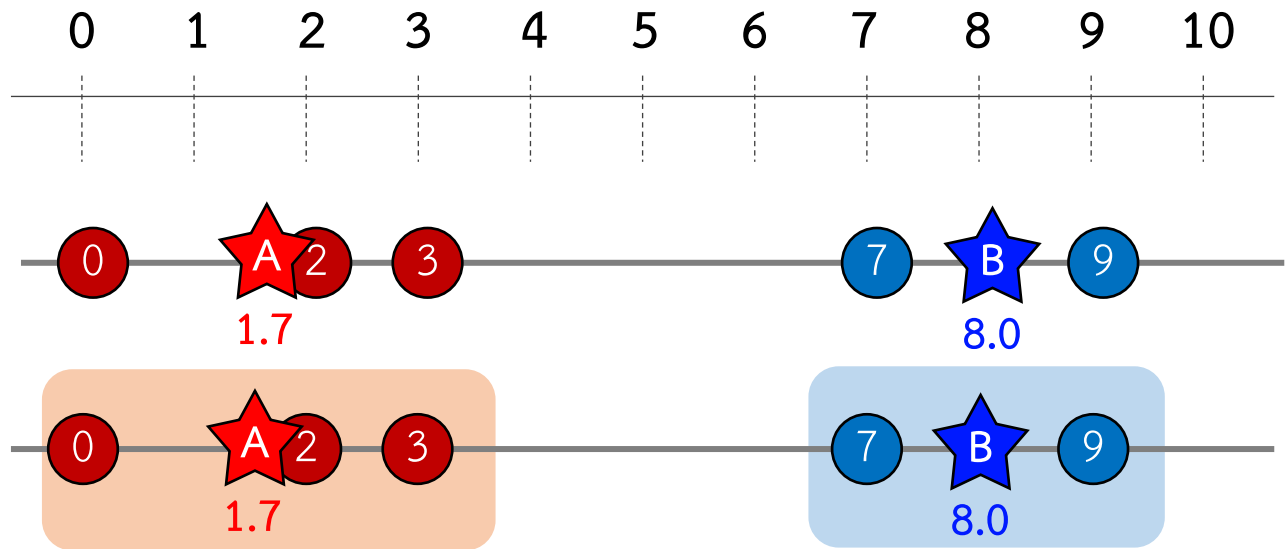


K-means: Example Steps (1 feature)



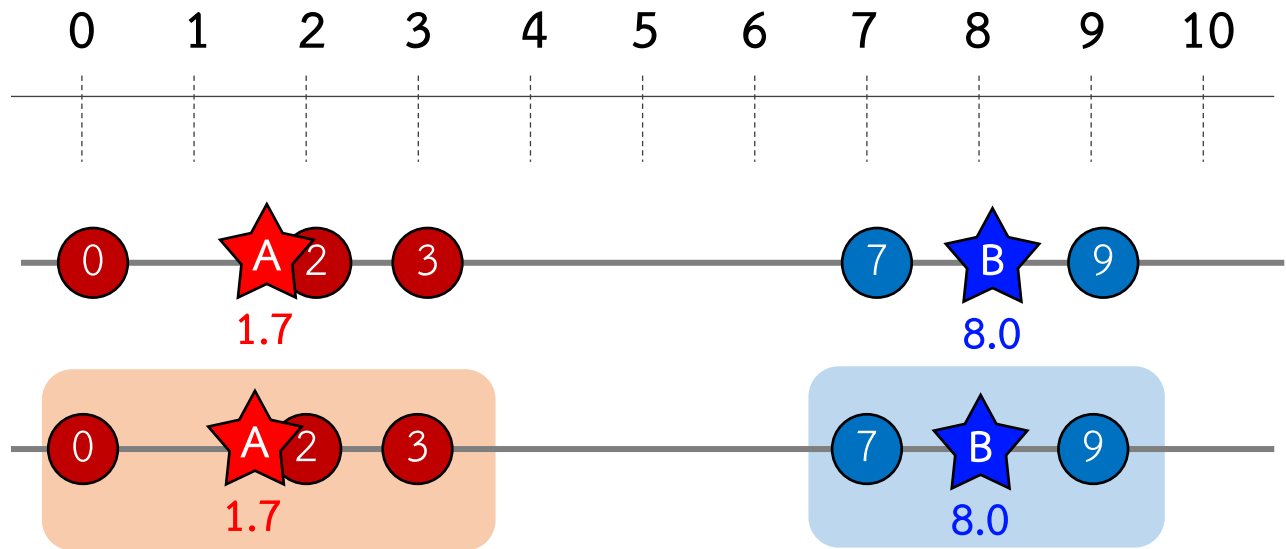
Round 3

K-means: Example Steps (1 feature)



Round 3

K-means: Example Steps (1 feature)

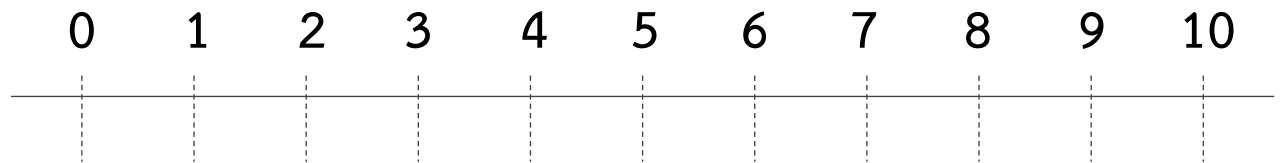


Round 3

$$\text{New Center of A} = (0+2+3)/3 = 1.7$$

$$\text{New Center of B} = (7+9)/2 = 8$$

K-means: Example Steps (1 feature)



Round 3

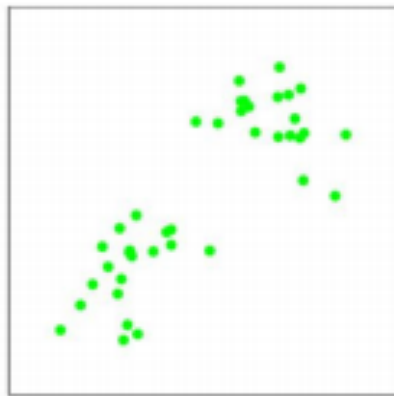


$$\text{New Center of A} = (0+2+3)/3 = 1.7$$

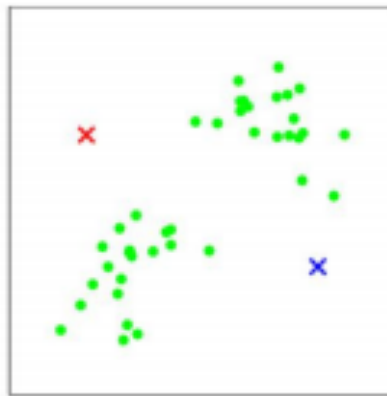
$$\text{New Center of B} = (7+9)/2 = 8$$



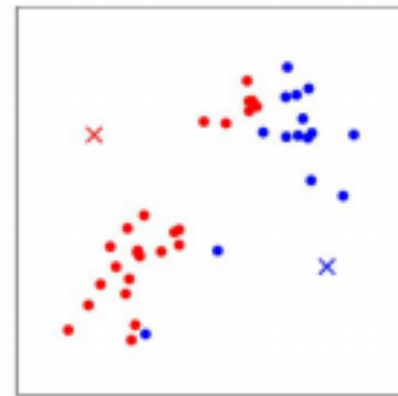
K-means: Example (2 features)



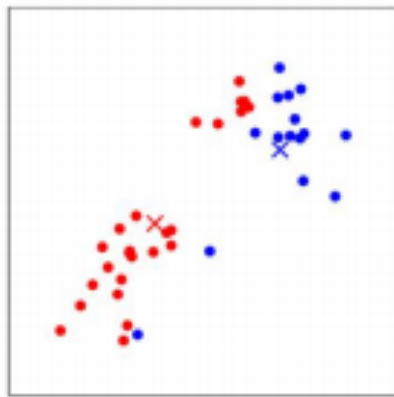
(a)



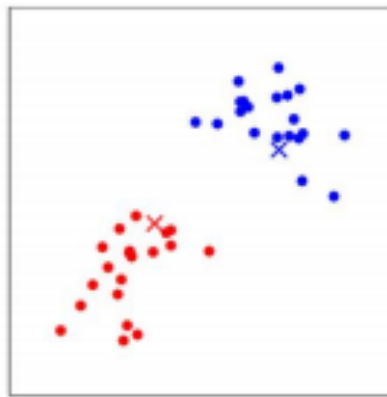
(b)



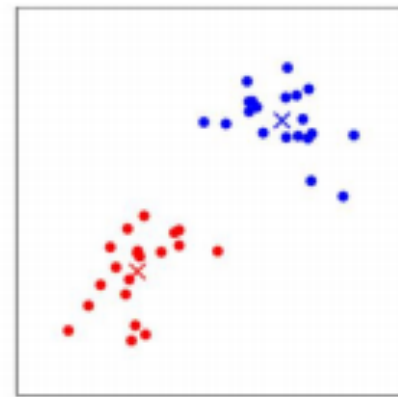
(c)



(d)

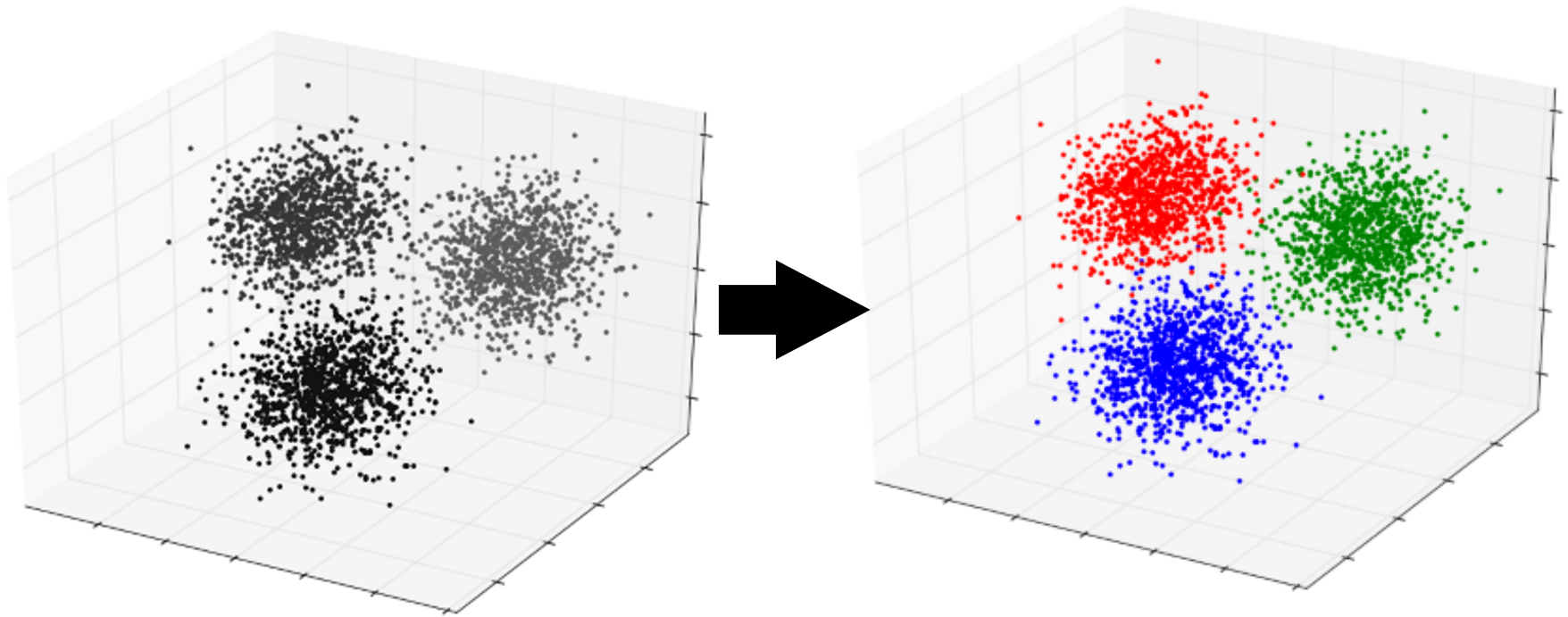


(e)

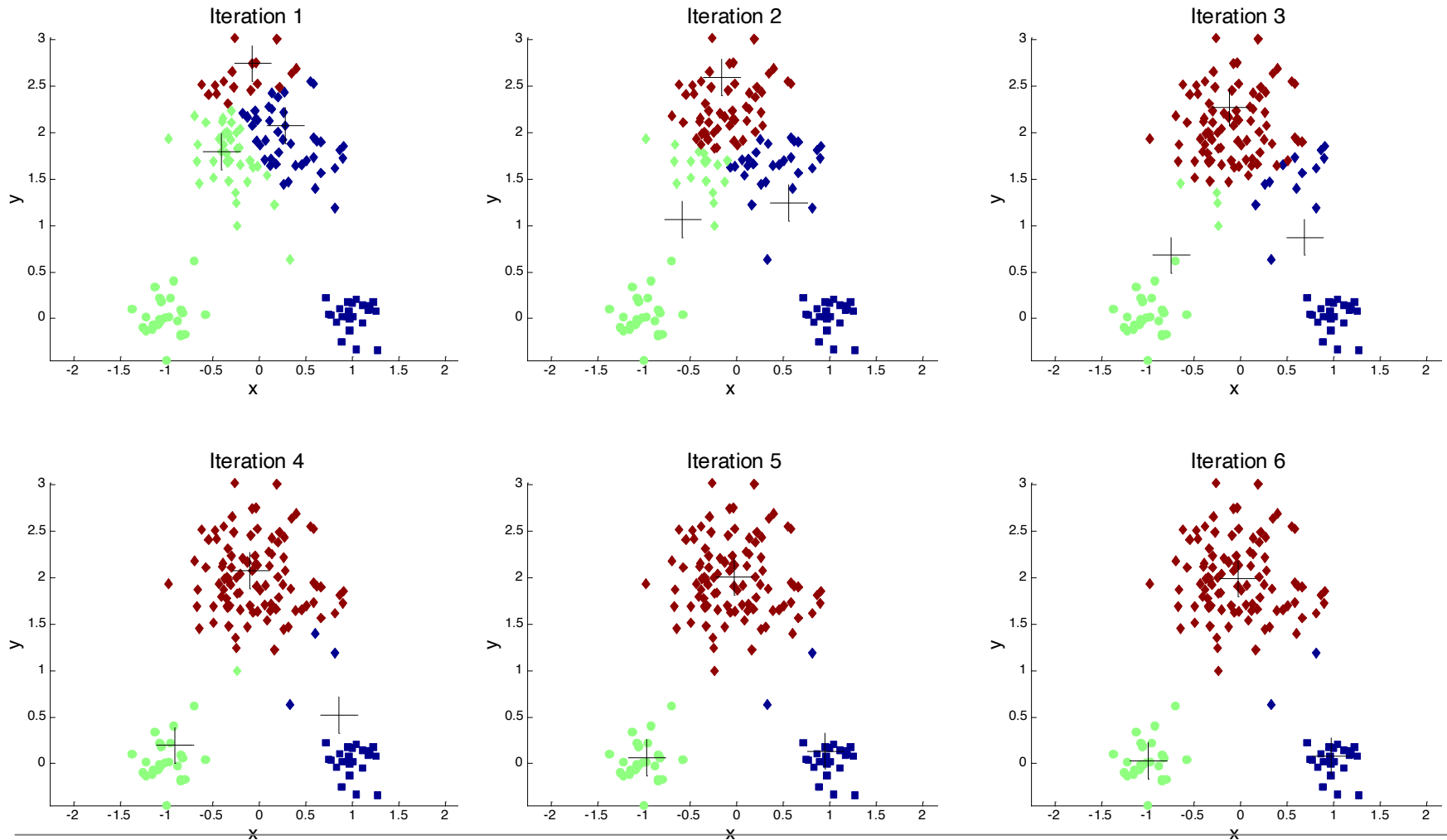


(f)

K-means: Example (3 features)

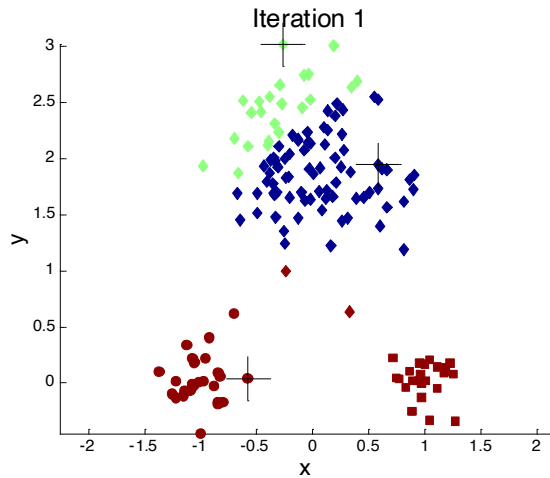
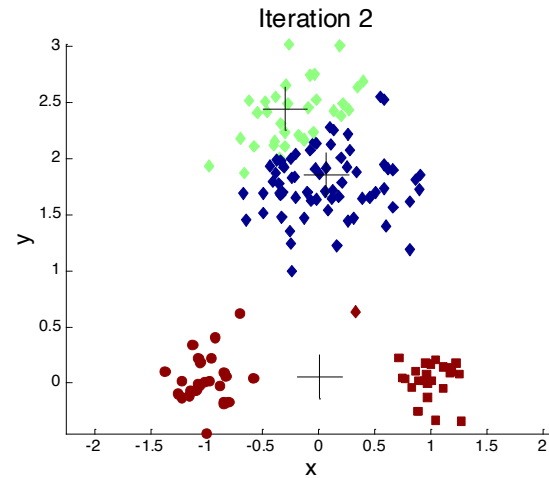
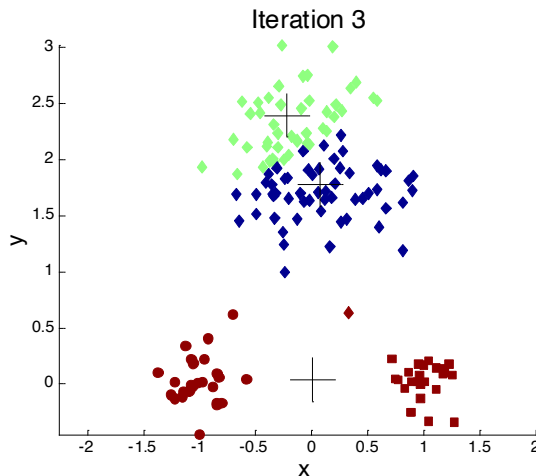
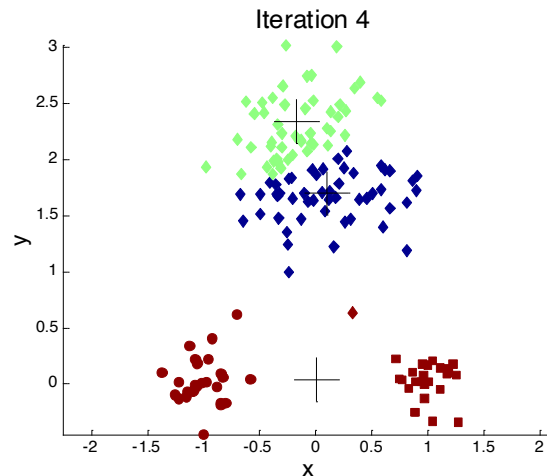
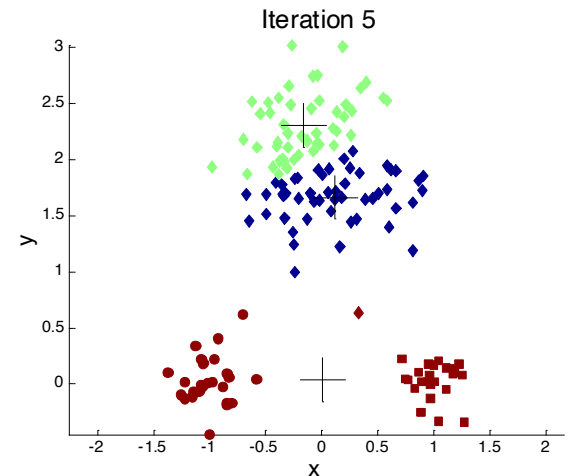


Importance of Choosing Initial Centroids (A)



Ref: • Min-Hashing, “Locality Sensitive Hashing Clustering”

Importance of Choosing Initial Centroids (B)



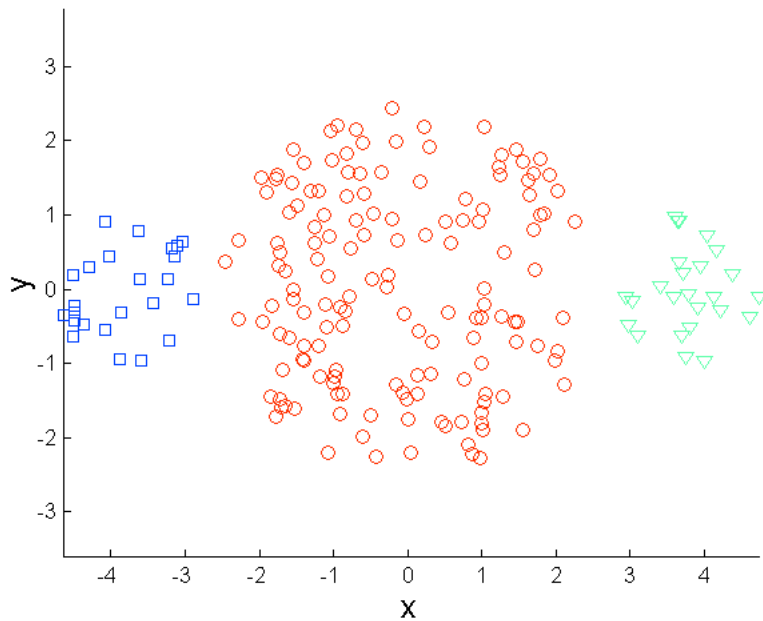
Dealing with Initialization

- Do **multiple runs** and select the clustering with the smallest error
- Select original set of points by methods other than random . E.g., pick the most distant (from each other) points as cluster centers

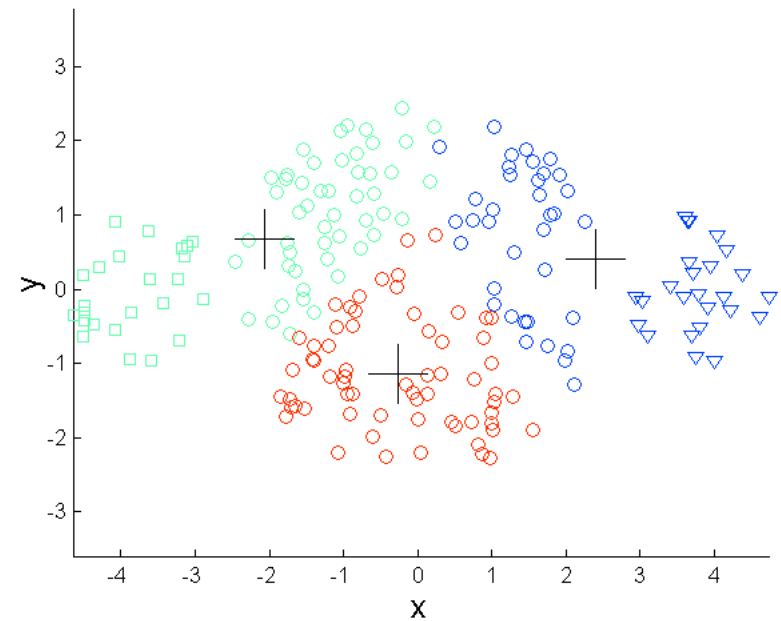
Limitations of K-means

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

Limitations of K-means: Differing Sizes

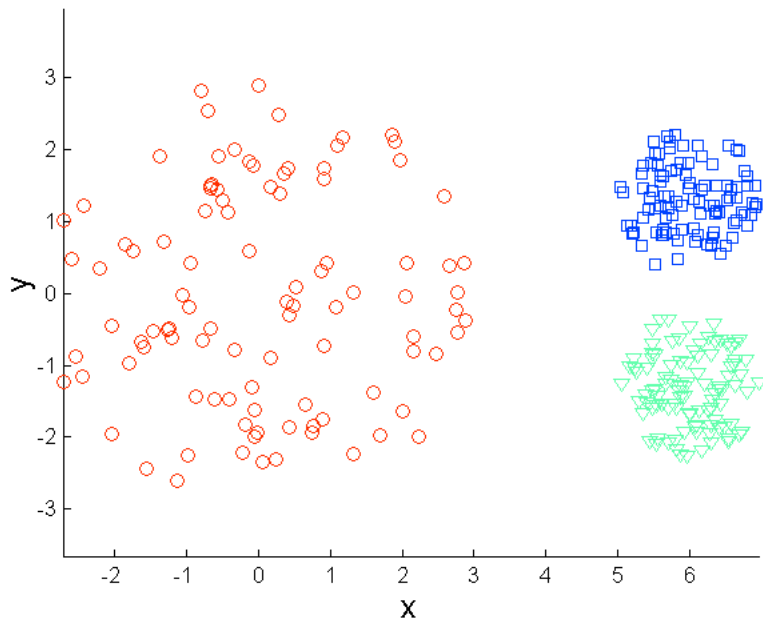


Original Points

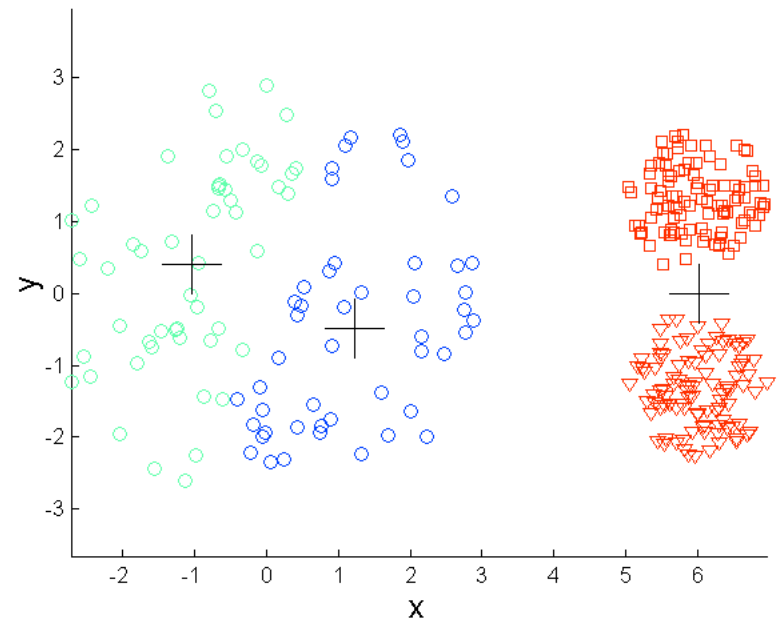


K-means (3 Clusters)

Limitations of K-means: Differing Density

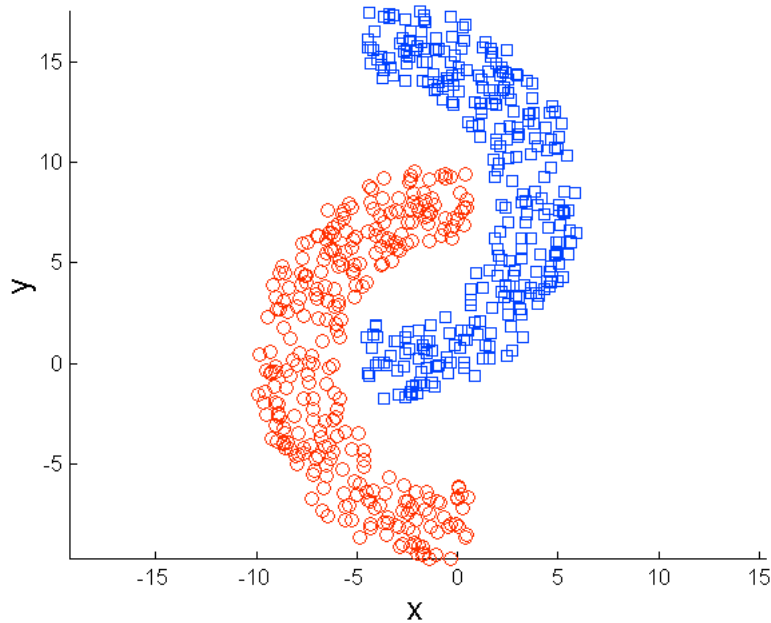


Original Points

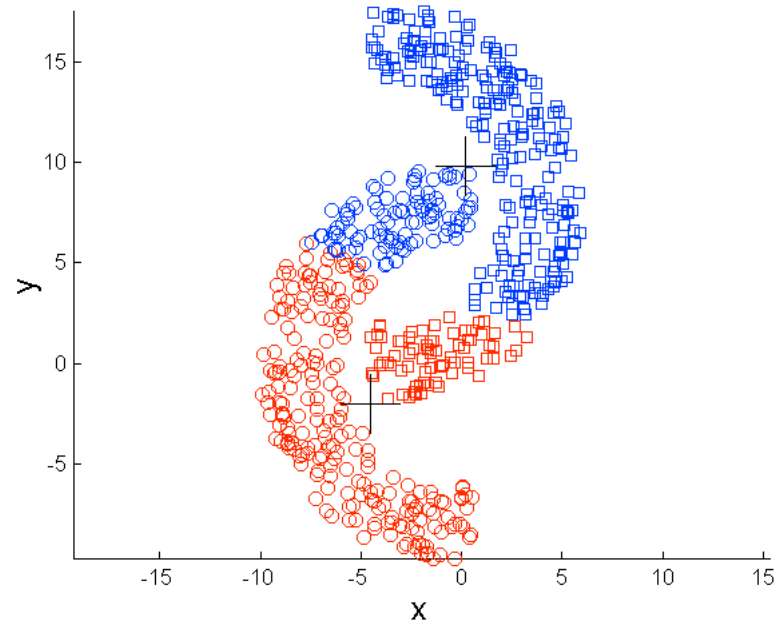


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

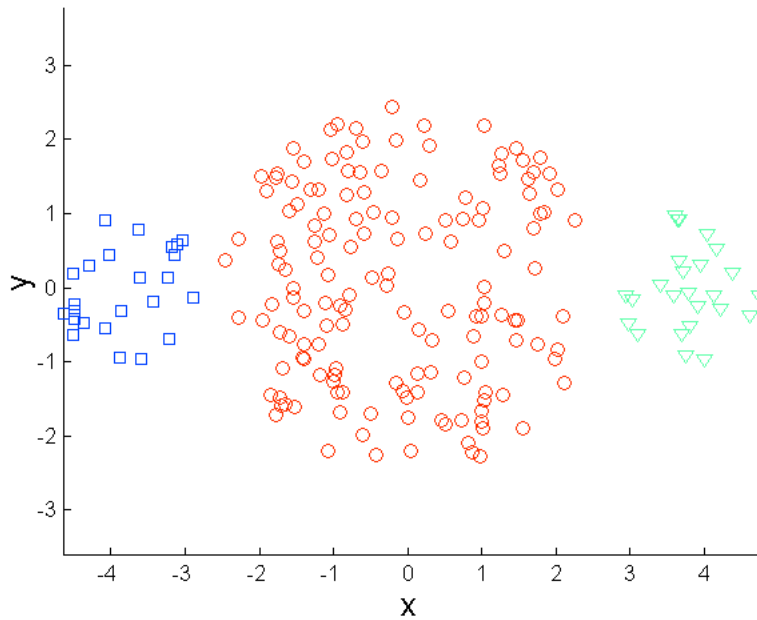


Original Points

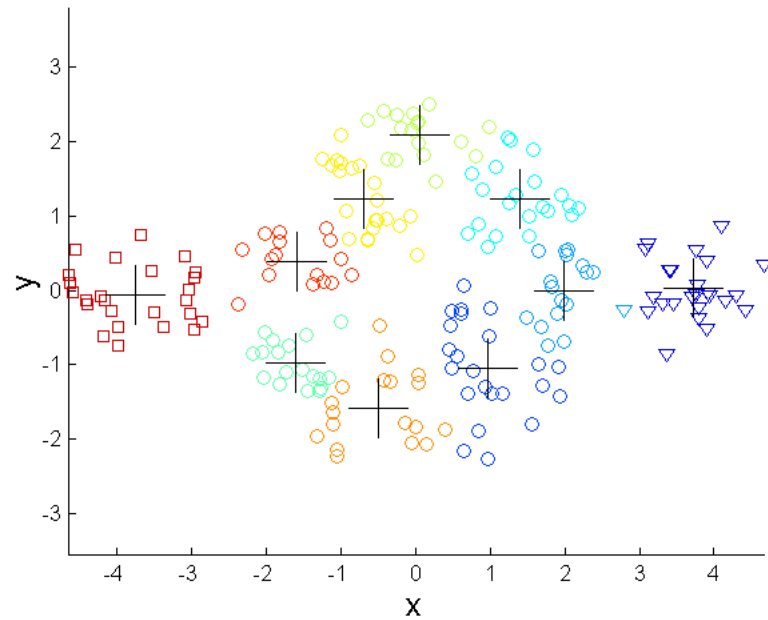


K-means (2 Clusters)

Overcoming K-means Limitations



Original Points

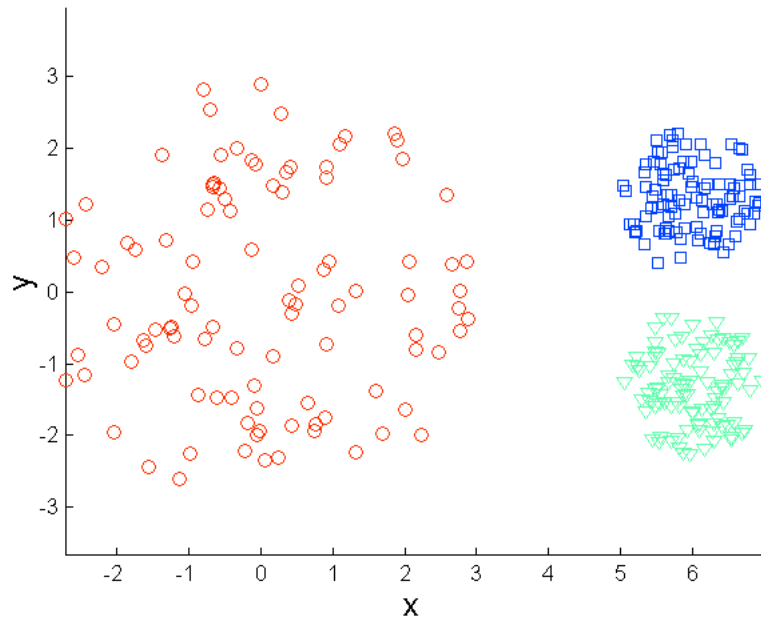


K-means Clusters

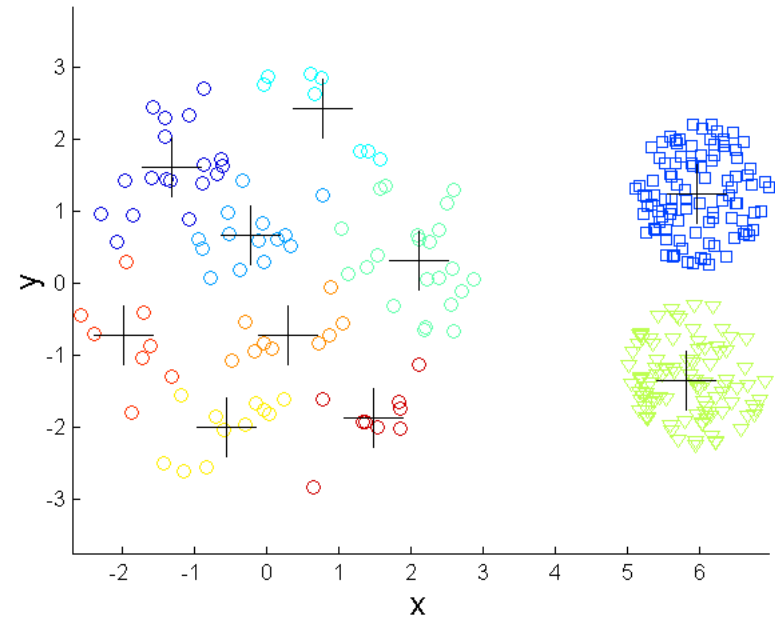
One solution is to use many clusters.

Find parts of clusters, but need to put together.

Overcoming K-means Limitations

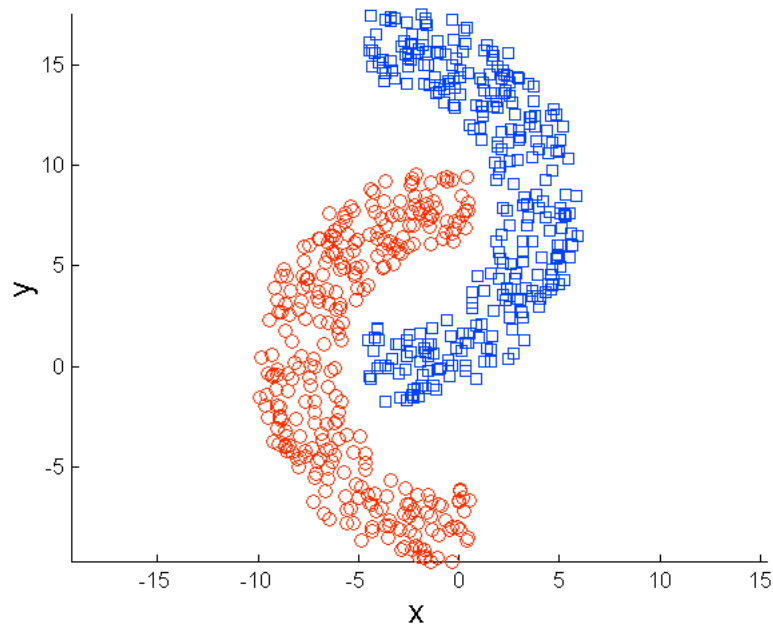


Original Points

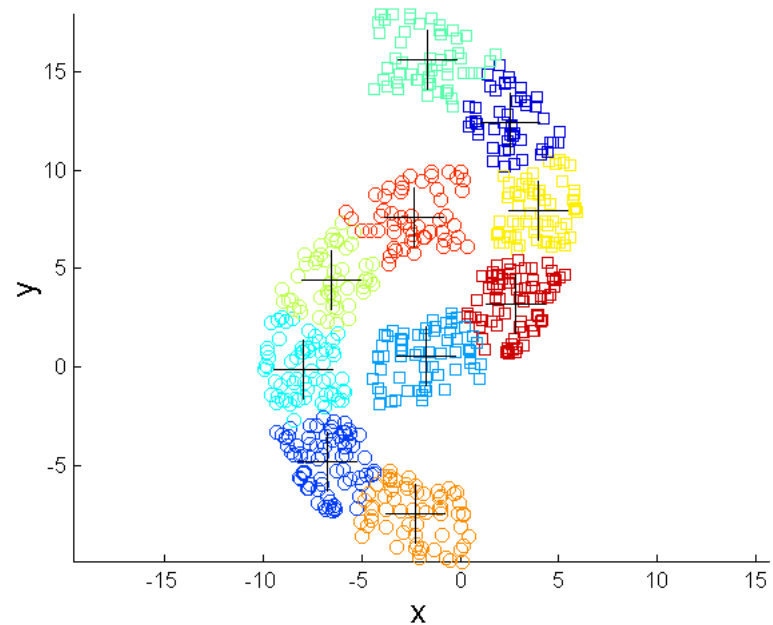


K-means Clusters

Overcoming K-means Limitations



Original Points



K-means Clusters

Variations

- **K-medoids**

- Similar problem definition as in K-means, but the centroid of the cluster is defined to be one of the points in the cluster (the **medoid**).

- **K-centers**

- Similar problem definition as in K-means, but the goal now is to minimize the maximum **diameter** of the clusters (diameter of a cluster is maximum distance between any two points in the cluster).

Python

```
>>> from sklearn.cluster import KMeans
>>> import numpy as np
>>> X = np.array([[1, 2], [1, 4], [1, 0],
...               [4, 2], [4, 4], [4, 0]])
>>> kmeans = KMeans(n_clusters=2, random_state=0).fit(X)
>>> kmeans.labels_
array([0, 0, 0, 1, 1, 1], dtype=int32)
>>> kmeans.predict([[0, 0], [4, 4]])
array([0, 1], dtype=int32)
>>> kmeans.cluster_centers_
array([[ 1.,  2.],
       [ 4.,  2.]])
```

DBSCAN



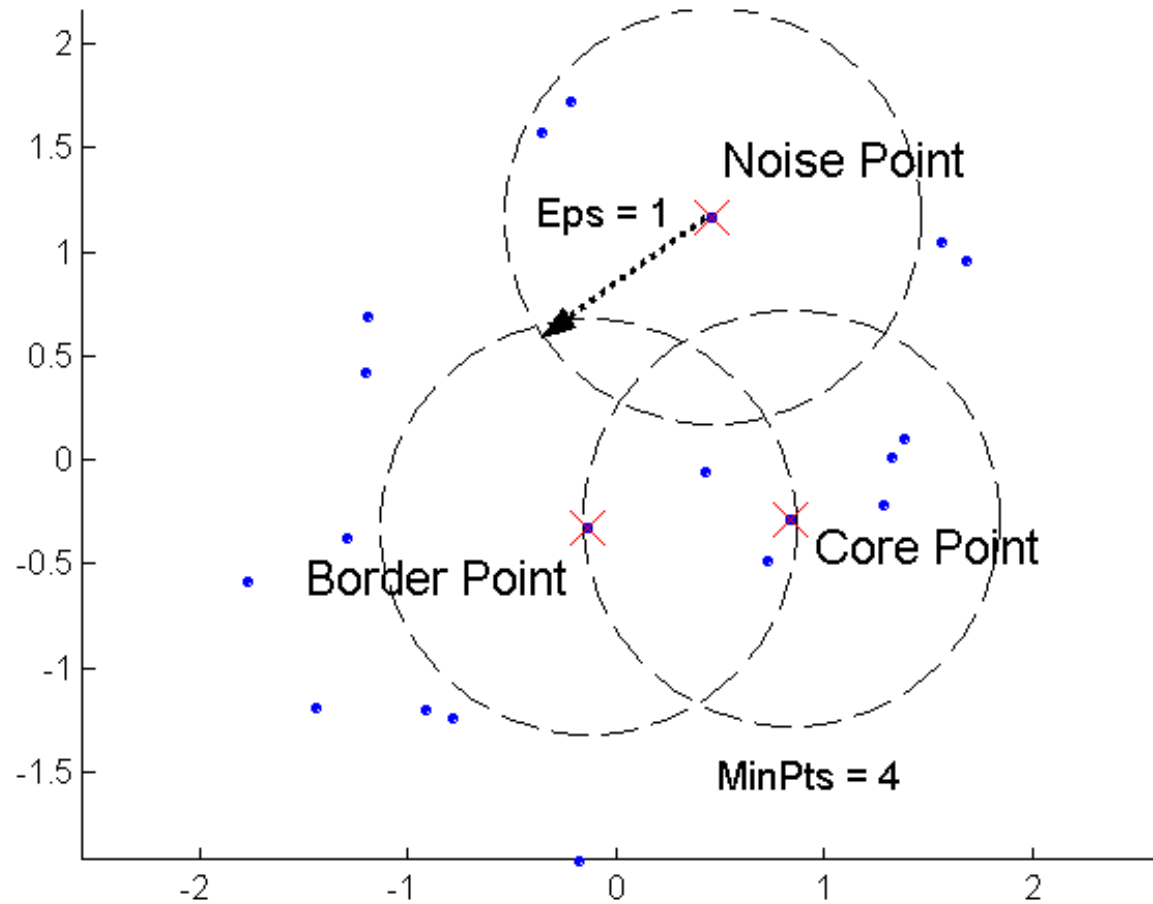
DBSCAN: Density-Based Clustering

- DBSCAN is a Density-Based Clustering algorithm
- Reminder: In density based clustering we partition points into dense regions separated by not-so-dense regions.
- Important Questions:
 - How do we measure density?
 - What is a dense region?
- DBSCAN:
 - Density at point p : number of points within a circle of radius Eps
 - Dense Region: A circle of radius Eps that contains at least $MinPts$ points

DBSCAN

- Characterization of points
 - A point is a **core point** if it has more than a specified number of points (**MinPts**) within **Eps**
 - These points belong in a **dense region** and are at the **interior** of a cluster
 - A **border point** has fewer than **MinPts** within **Eps**, but is in the neighborhood of a **core** point.
 - A **noise point** is any point that is not a core point or a border point.

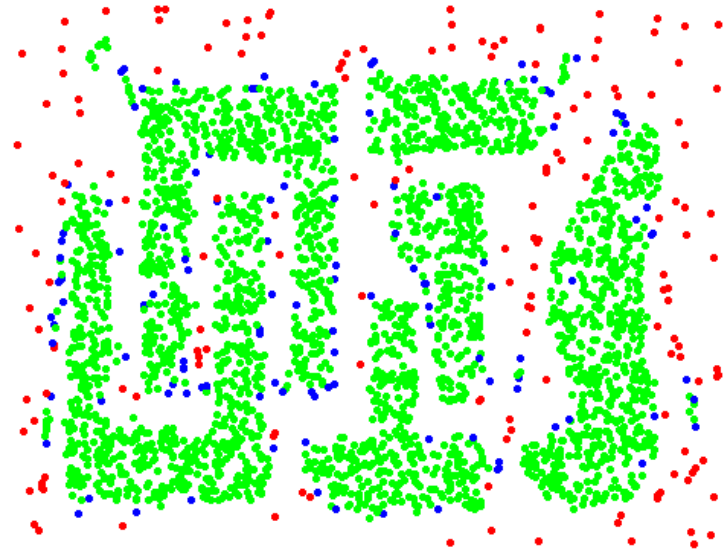
DBSCAN: Core, Border, and Noise Points



DBSCAN: Core, Border, and Noise Points



Original Points



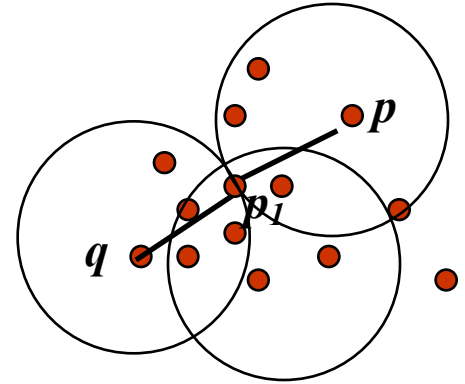
Point types: **core**,
border and **noise**

Eps = 10, MinPts = 4

Density-Connected points

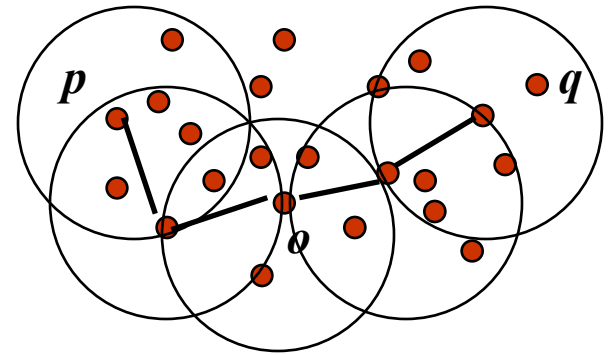
- Density edge

- We place an **edge** between two core points **q** and **p** if they are within distance **Eps**.



- Density-connected

- A point **p** is **density-connected** to a point **q** if there is a **path of edges** from **p** to **q**



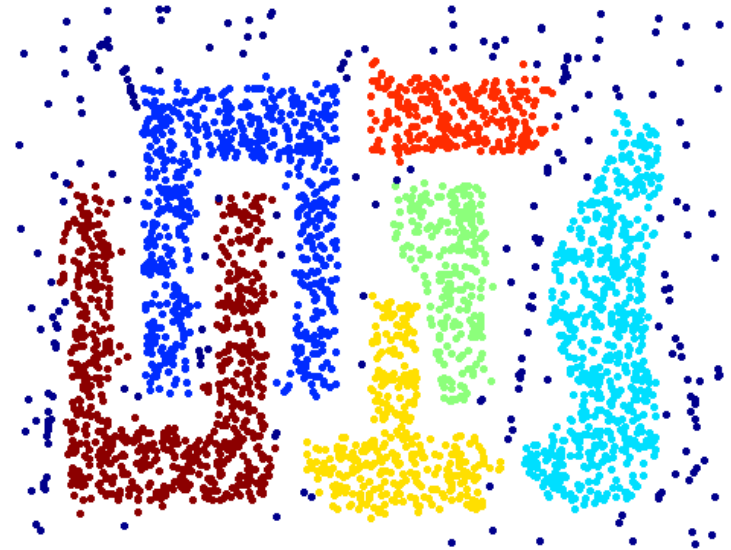
DBSCAN Algorithm

- Label points as **core**, **border** and **noise**
- Eliminate **noise** points
- For every **core** point **p** that has not been assigned to a cluster
 - Create a new cluster with the point **p** and all the points that are **density-connected** to **p**.
- Assign **border** points to the cluster of the closest core point.

When DBSCAN Works Well



Original Points



Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

Cluster Validation



Different Aspects of Cluster Validation

1. Determining the **clustering tendency** of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
 - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the ‘correct’ number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.

Measures of Cluster Validity

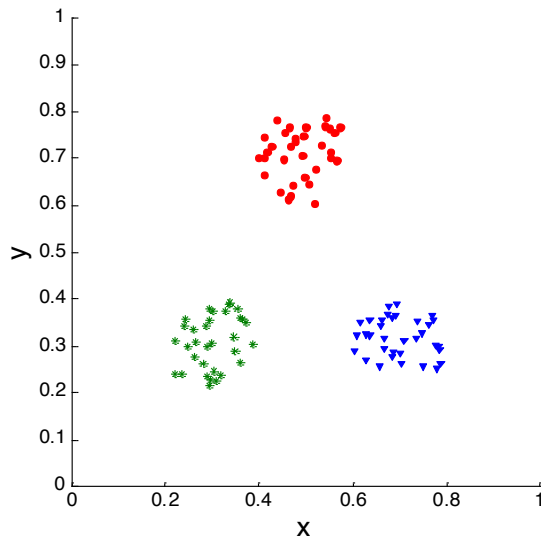
- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
 - **External Index:** Used to measure the extent to which cluster labels match externally supplied class labels.
 - Entropy
 - **Internal Index:** Used to measure the goodness of a clustering structure *without* respect to external information.
 - Sum of Squared Error (SSE)
 - **Relative Index:** Used to compare two different clusterings or clusters.
 - Often an external or internal index is used for this function, e.g., SSE or entropy
- Sometimes these are referred to as **criteria** instead of **indices**
 - However, sometimes criterion is the general strategy and index is the numerical measure that implements the criterion.

Measuring Cluster Validity Via Correlation

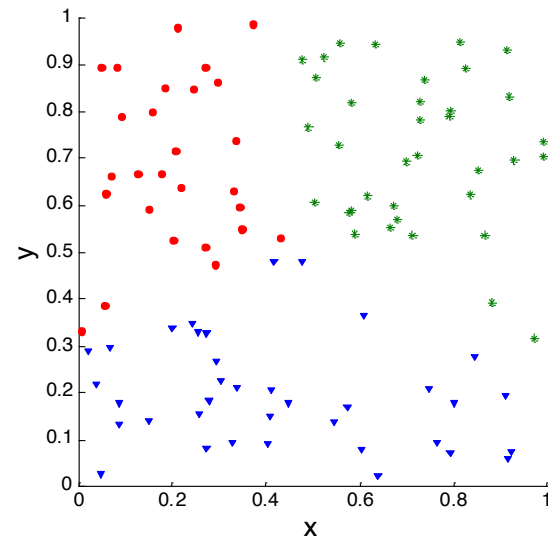
- Two matrices
 - Proximity Matrix
 - Ideal Similarity Matrix
 - One row and one column for each data point
 - An entry is 1 if the associated pair of points belong to the same cluster
 - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
 - Since the matrices are symmetric, only the correlation between $n(n-1) / 2$ entries needs to be calculated.
- High correlation indicates that points that belong to the same cluster are close to each other.
- Not a good measure for some density or contiguity based clusters.

Measuring Cluster Validity Via Correlation

- Correlation of ideal similarity and proximity matrices for the K-means clusterings of the following two data sets.



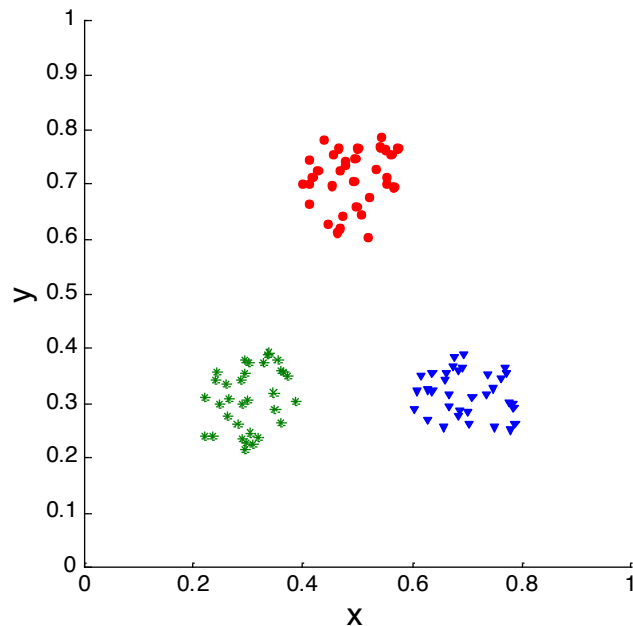
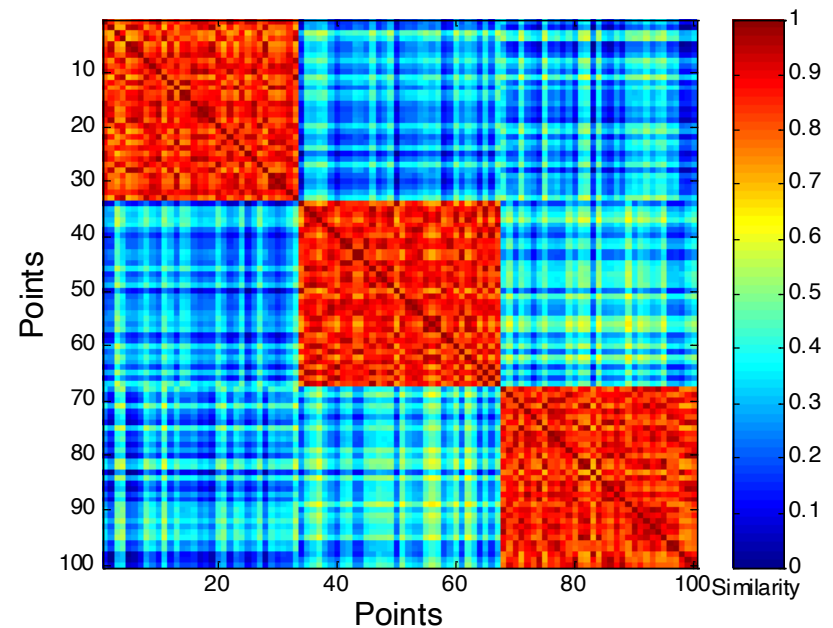
Corr = -0.9235



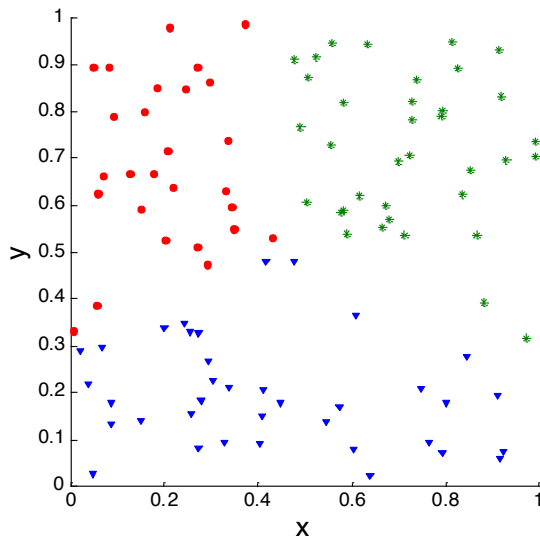
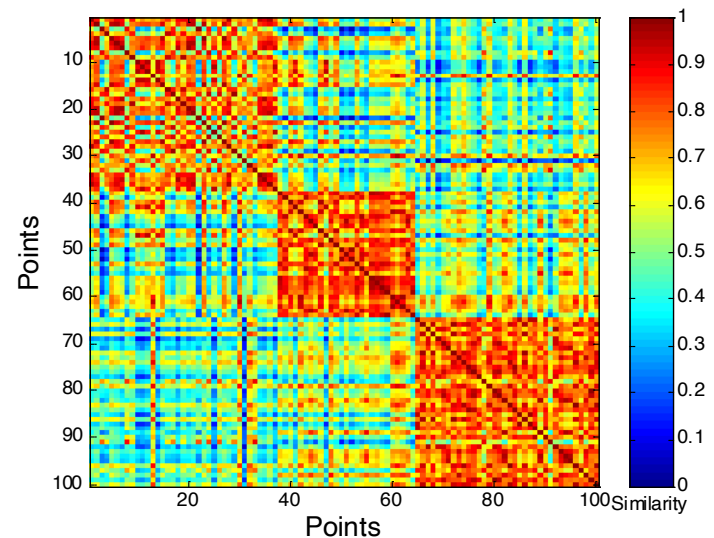
Corr = -0.5810

Using Similarity Matrix for Cluster Validation

- Order the similarity matrix with respect to cluster labels and inspect visually.

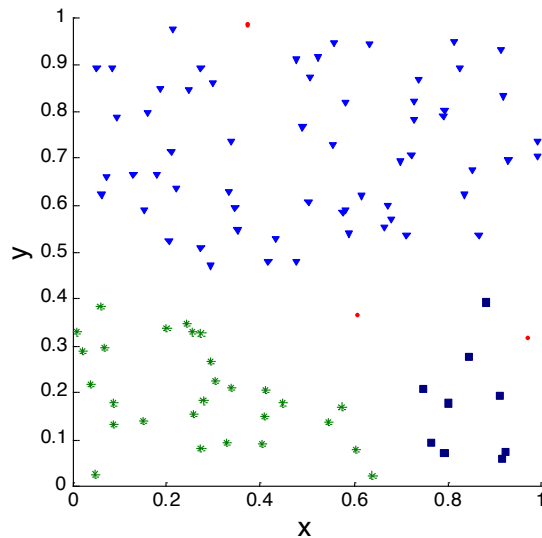
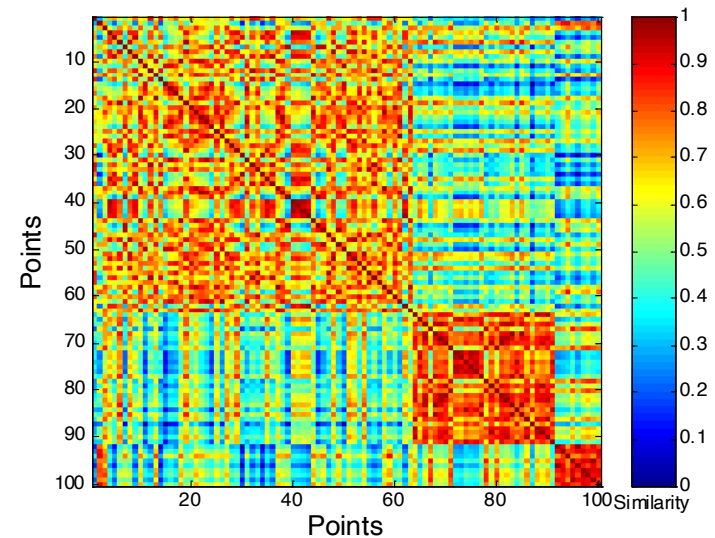


Using Similarity Matrix for Cluster Validation



K-means

Using Similarity Matrix for Cluster Validation



DBSCAN

Dimensionality Reduction



The curse of dimensionality

- Real data usually have **thousands**, or **millions** of dimensions
 - E.g., web documents, where the dimensionality is the vocabulary of words
 - Facebook graph, where the dimensionality is the number of users
- Huge number of dimensions causes problems
 - Data becomes very **sparse**, some algorithms become meaningless (e.g. density based clustering)
 - The **complexity** of several algorithms depends on the dimensionality and they become infeasible.

Dimensionality Reduction

- Usually the data can be described with fewer dimensions, without losing much of the meaning of the data.
 - The data **reside** in a space of lower dimensionality
- Essentially, we assume that some of the data is noise, and we can approximate the useful part with a lower dimensionality space.
 - Dimensionality reduction does not just reduce the amount of data, it often brings out the **useful** part of the data

Latent factor model

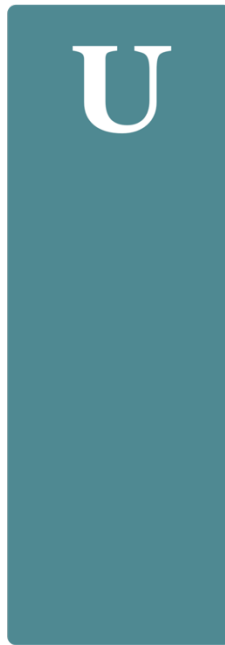
- Rows (columns) are linear combinations of k latent factors
 - E.g., in our extreme document example there are two factors
- Some noise is added to this rank- k matrix resulting in higher rank
- SVD retrieves the latent factors (hopefully).

SVD (Singular Value Decomposition)

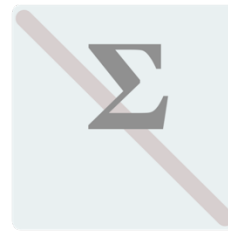


data matrix

=



left singular
vectors



diagonal of
singular
values



right singular
vectors

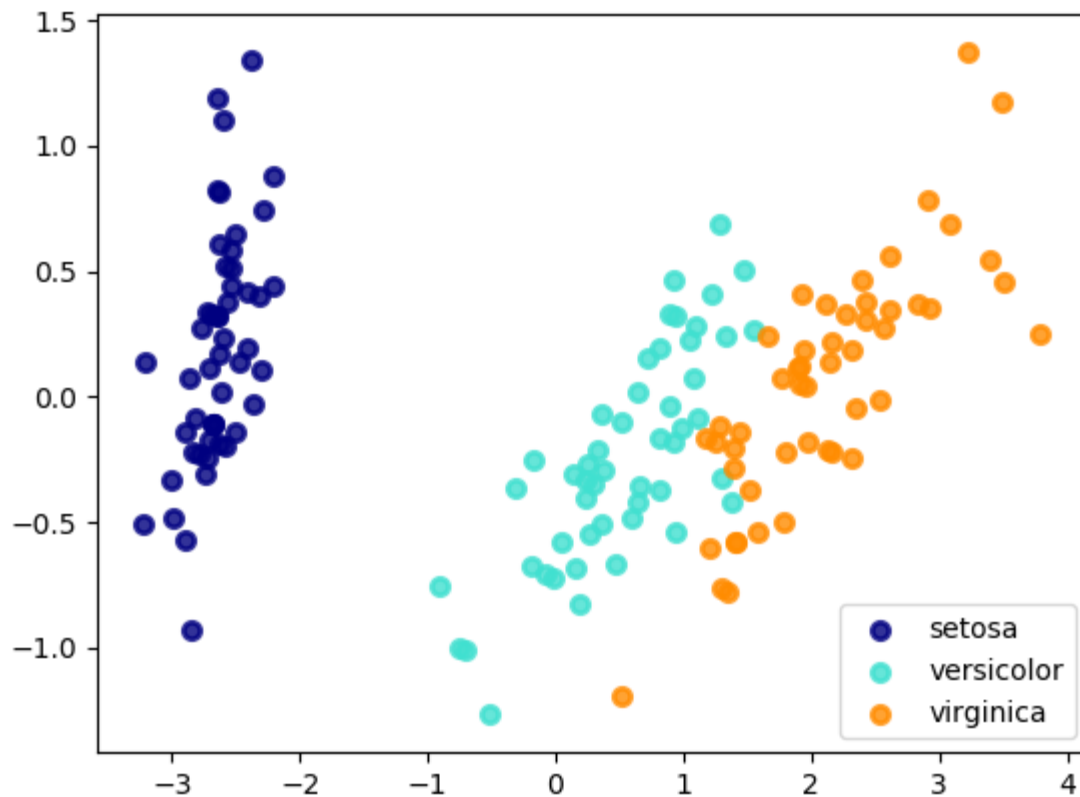
SVD

- Example

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

Plot

- Dimensionality Reduction of the Iris Dataset



Python

```
>>> import numpy as np
>>> from sklearn.decomposition import PCA
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
>>> pca = PCA(n_components=2)
>>> pca.fit(X)
PCA(copy=True, iterated_power='auto', n_components=2, random_state=None,
     svd_solver='auto', tol=0.0, whiten=False)
>>> print(pca.explained_variance_ratio_)
[ 0.99244...  0.00755...]
>>> print(pca.singular_values_)
[ 6.30061...  0.54980...]
```

“

Thanks to big data, machines can now be
programmed to the next thing right.
But only humans can do the next right thing.

”

Dov Seidman