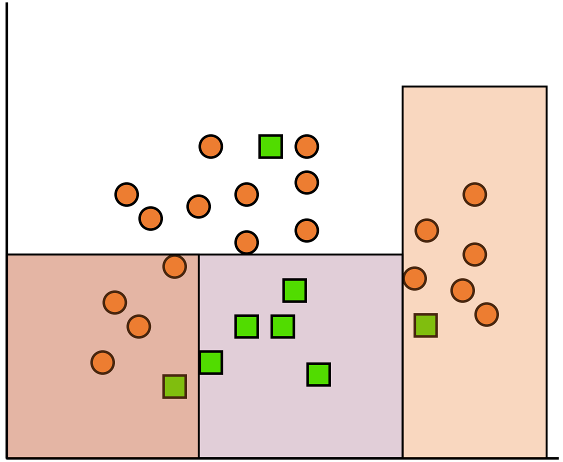
Smart Mobile Platform
Clustering

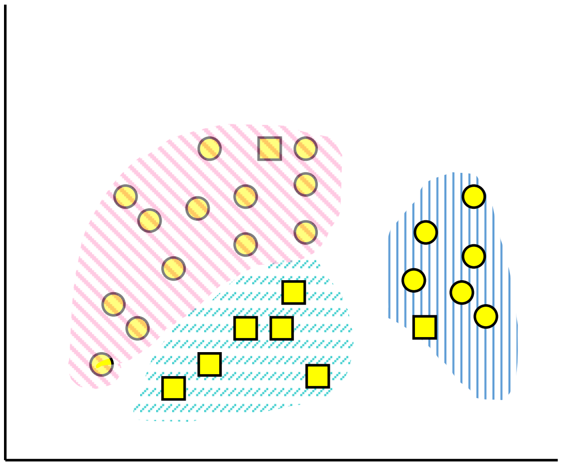
Prof. Joongheon Kim
Korea University, School of Electrical Engineering
https://joongheon.github.io
joongheon@korea.ac.kr

- **Introduction**
- Data Types and Representations
- Distance Measures
- Major Clustering Approaches
- Implementation
- Summary

- Classification vs. Clustering
  - Classification
    - Supervised Learning
    - Learns a method for predicting the instance class from pre-labeled (classified) instances

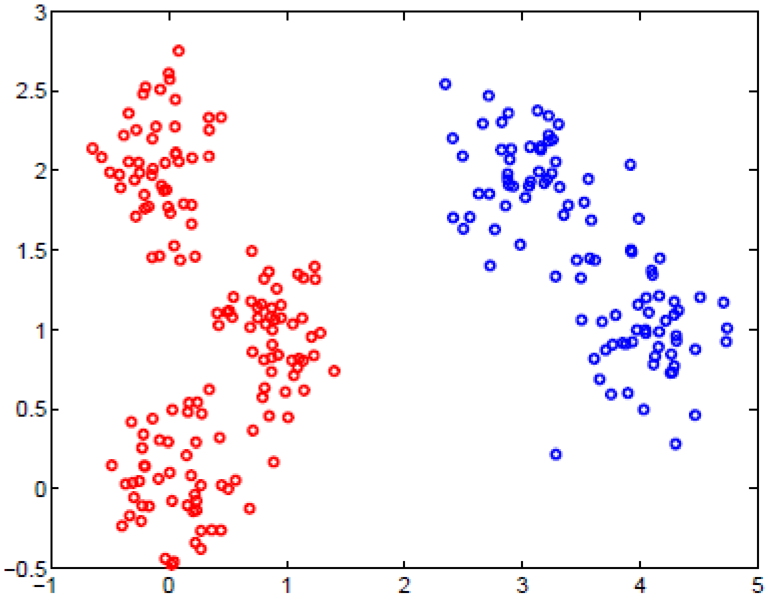


- Classification vs. Clustering
  - Clustering
    - Unsupervised Learning
    - Finds "natural" grouping of instances given un-labeled data

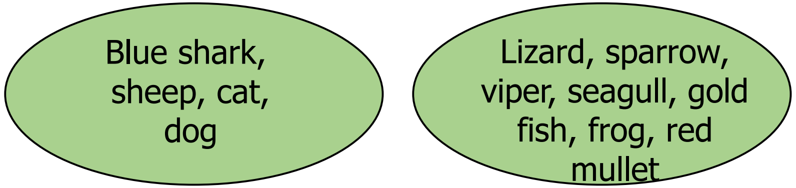


- Cluster: A collection/group of data objects/points
  - Similar (or related) to one another within the same group
  - Dissimilar (or unrelated) to the objects in other groups
- Cluster analysis
  - Find similarities between data according to characteristics underlying the data and grouping similar data objects into clusters
- Clustering Analysis: Unsupervised learning
  - No predefined classes for a training data set
  - Two general tasks: identify the "natural" clustering number and properly grouping objects into "sensible" clusters
- Typical applications
  - As a stand-alone tool to gain an insight into data distribution
  - As a preprocessing step of other algorithms in intelligent systems

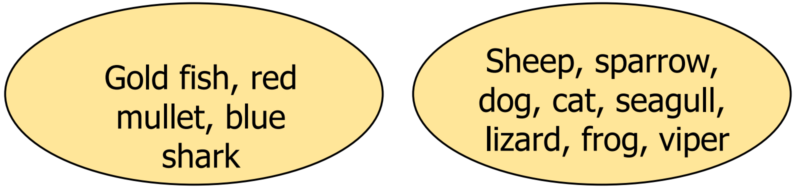
Introduction: How many clusters?



Introduction: Are they in the same cluster?

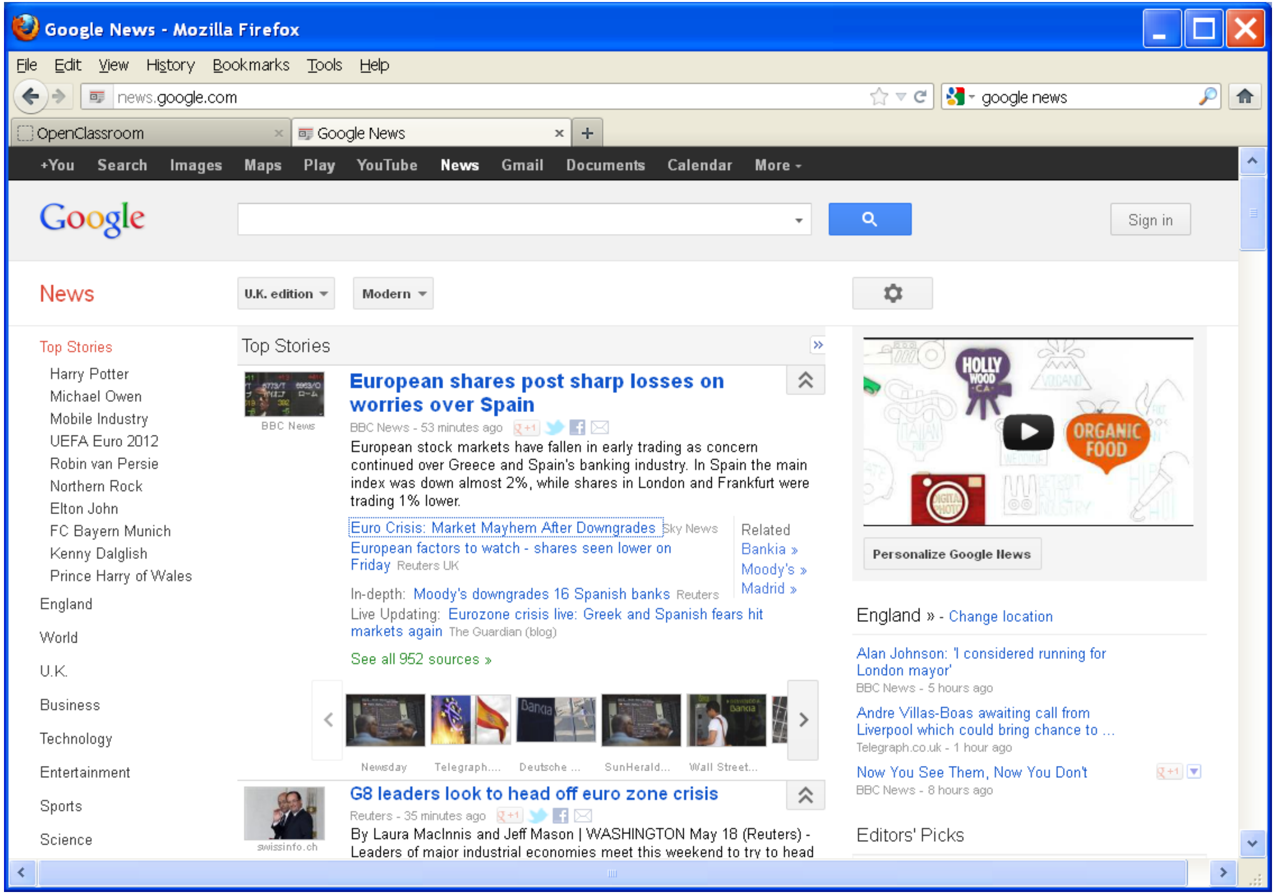


1. Two clusters
2. Clustering criterion:
   How animals bear their progeny

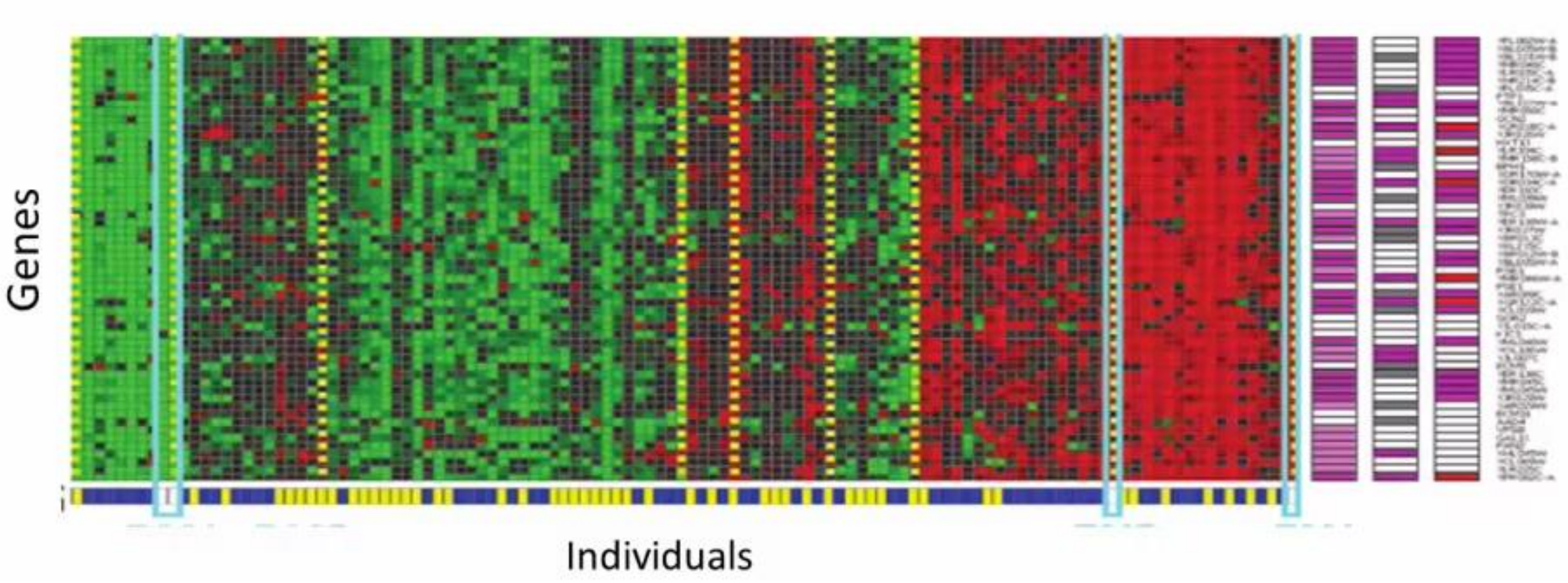


1. Two clusters
2. Clustering criterion:
   Existence of lungs

Introduction: Real Applications (Google News)

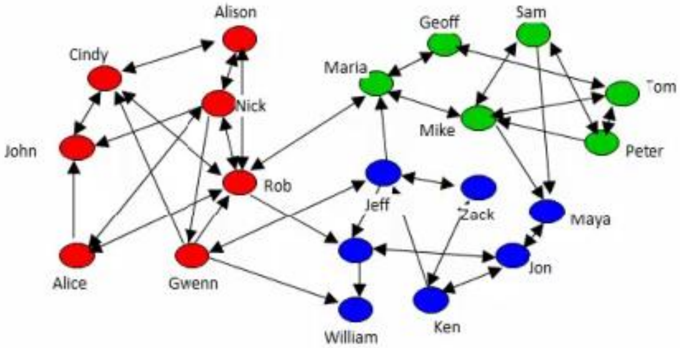


Introduction: Real Applications (Genetics Analysis)

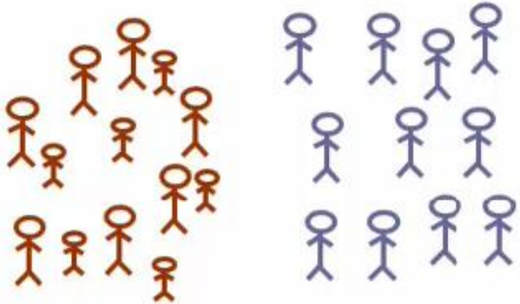


Introduction: Real Applications (Emerging Applications)

Organize computing clusters



Social network analysis



Market segmentation.

Astronomical data analysis

- A technique demanded by many real world tasks
  - **Bank/Internet Security:** fraud/spam pattern discovery
  - **Biology:** taxonomy of living things such as kingdom, phylum, class, order, family, genus and species
  - **City-planning:** Identifying groups of houses according to their house type, value, and geographical location
  - **Climate change:** understanding earth climate, find patterns of atmospheric and ocean
  - **Finance:** stock clustering analysis to uncover correlation underlying shares
  - **Image Compression/segmentation:** coherent pixels grouped
  - **Information retrieval/organization:** Google search, topic-based news
  - **Land use:** Identification of areas of similar land use in an earth observation database
  - **Marketing:** Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
  - **Social network mining:** special interest group automatic discovery

Unsupervised Learning (Outline)

- Introduction
- **Data Types and Representations**
- Distance Measures
- Major Clustering Approaches
- Implementation
- Summary

- Discrete vs. Continuous
  - **Discrete Feature**
    - Has only a finite set of values
      e.g., zip codes, rank, or the set of words in a collection of documents
    - Sometimes, represented as integer variable
  - **Continuous Feature**
    - Has real numbers as feature values
      e.g., temperature, height, or weight
    - Practically, real values can only be measured and represented using a finite number of digits
    - Continuous features are typically represented as floating-point variables

- Data representations
  - Data matrix (object-by-feature structure)

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

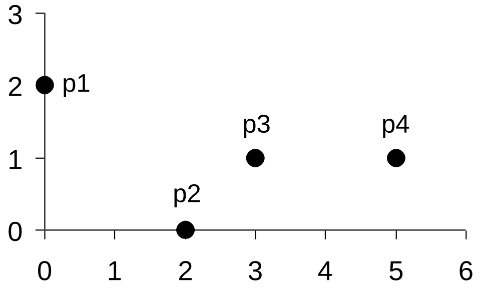
- *n* data points (objects) with *p* dimensions (features)
- Two modes: row and column represent different entities

  - Distance/dissimilarity matrix (object-by-object structure)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

- *n* data points, but registers only the distance
- A symmetric/triangular matrix
- Single mode: row and column for the same entity (distance)

- Examples



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Data Matrix

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix (i.e., Dissimilarity Matrix) for Euclidean Distance

- Introduction
- Data Types and Representations
- **Distance Measures**
- Major Clustering Approaches
- Implementation
- Summary

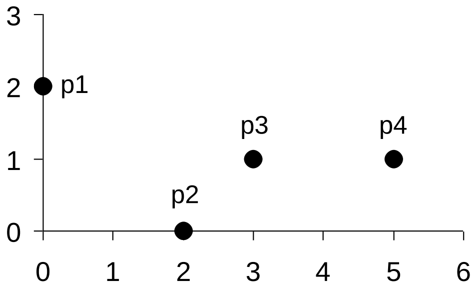
- **Minkowski Distance** (http://en.wikipedia.org/wiki/Minkowski_distance)
  - For $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{y} = (y_1, \ldots, y_n)$

$$\boxed{d(\vec{x}, \vec{y}) = (|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p)^{1/p}}$$

    - $p = 1$: Manhattan (city block) distance
    - $p = 2$: Euclidean distance

  - Do not confuse p with n, i.e., all these distances are defined based on all numbers of features (dimensions).
  - A generic measure: use appropriate p in different applications

Distance Measures: Minkowski Distance



L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

Distance Matrix for Manhattan Distance

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Data Matrix

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix for Euclidean Distance

- **Cosine Measure (Similarity vs. Distance)**
  - For $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{y} = (y_1, \ldots, y_n)$

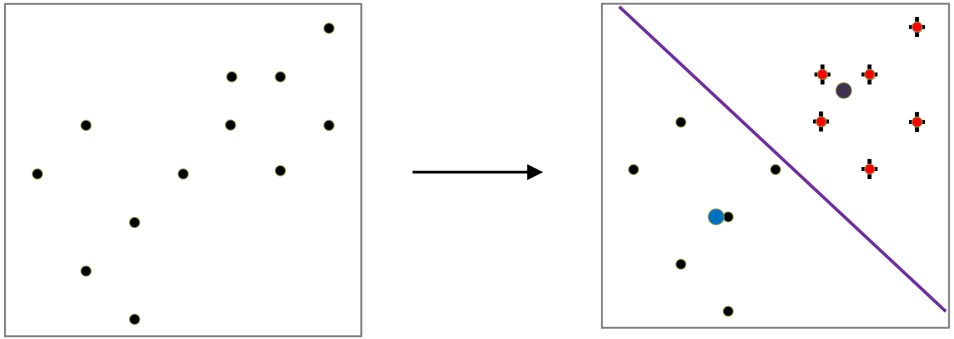
$$d(\vec{x}, \vec{y}) = 1 - \cos(\vec{x}, \vec{y})$$

$$\cos(\vec{x}, \vec{y}) = \frac{x_1 y_1 + \cdots + x_n y_n}{\sqrt{x_1^2 + \cdots + x_n^2}\sqrt{y_1^2 + \cdots + y_n^2}}$$

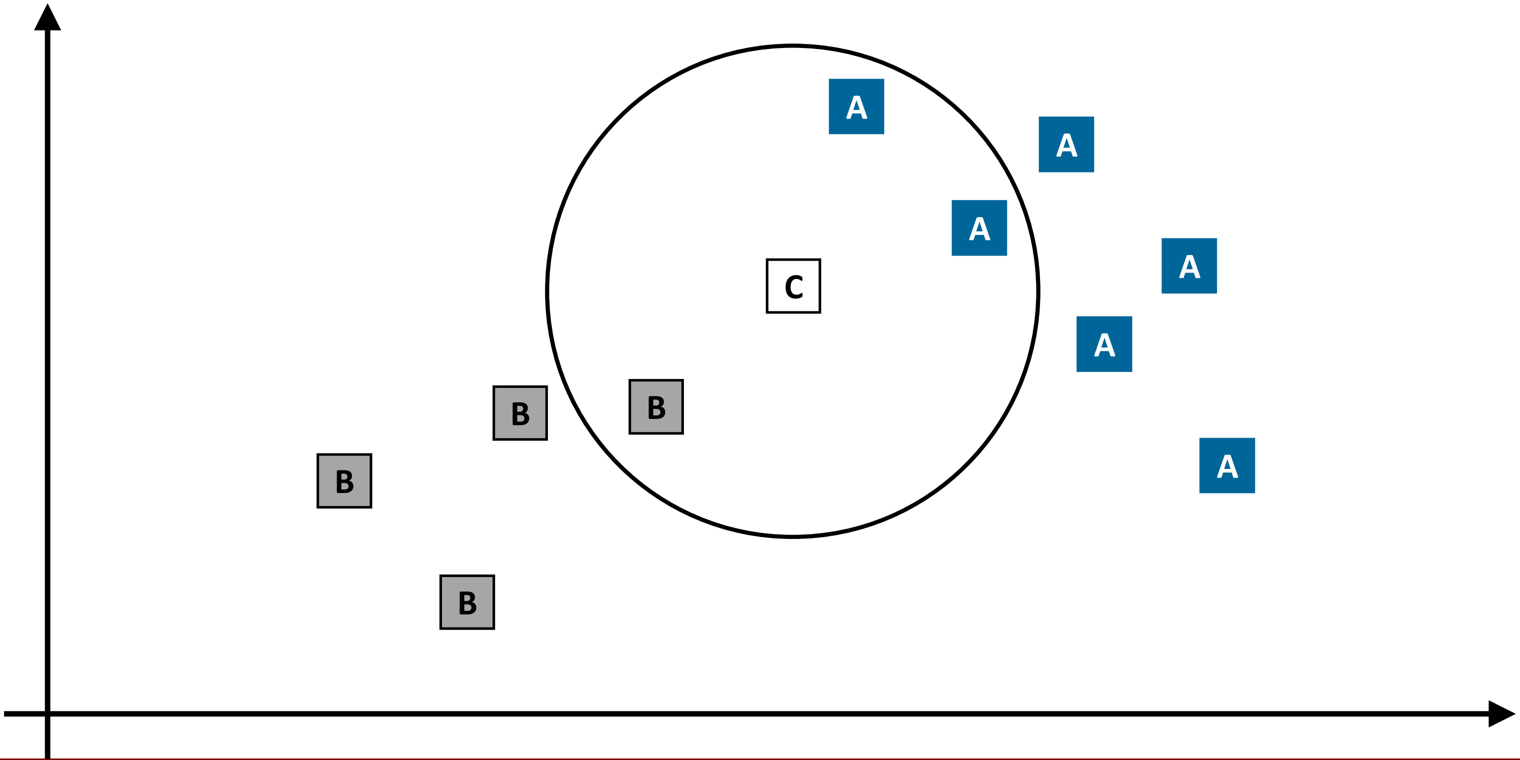
  - Property: $0 \leq d(\vec{x}, \vec{y}) \leq 2$
  - Nonmetric vector objects: keywords in documents, gene features in micro-arrays, …
  - Applications: information retrieval, biologic taxonomy, ...

- Introduction
- Data Types and Representations
- Distance Measures
- **Major Clustering Approaches**
- Implementation
- Summary

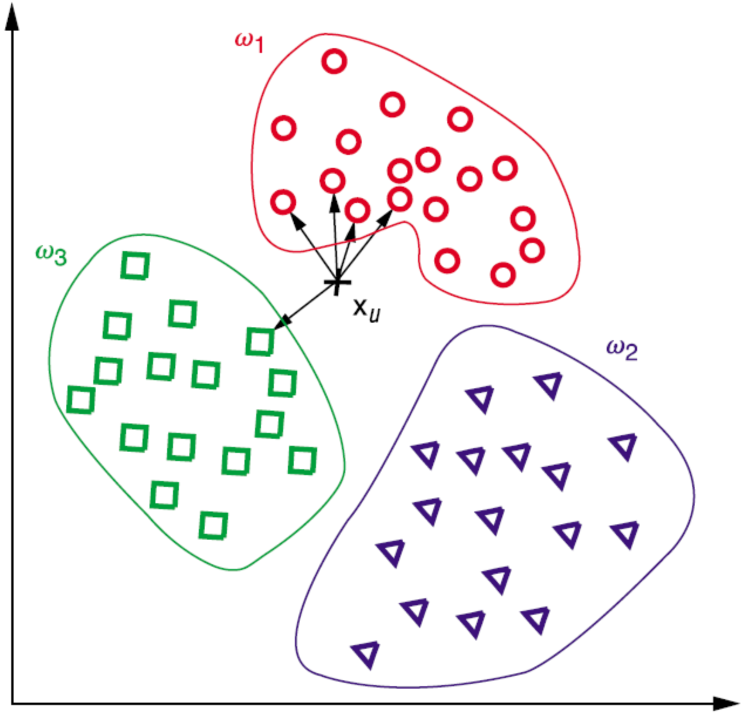
- Partitioning Approach
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square distance cost
  - Typical methods: K-means, K-medoids, CLARANS, ……



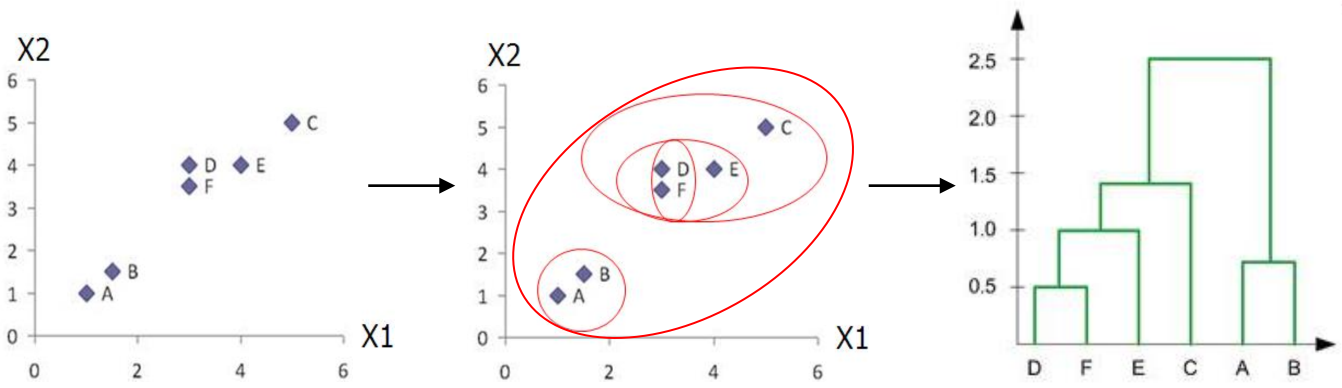
- Partitioning Approach
  - kNN (k Nearest Neighbor: k=3)



- Partitioning Approach
  - kNN

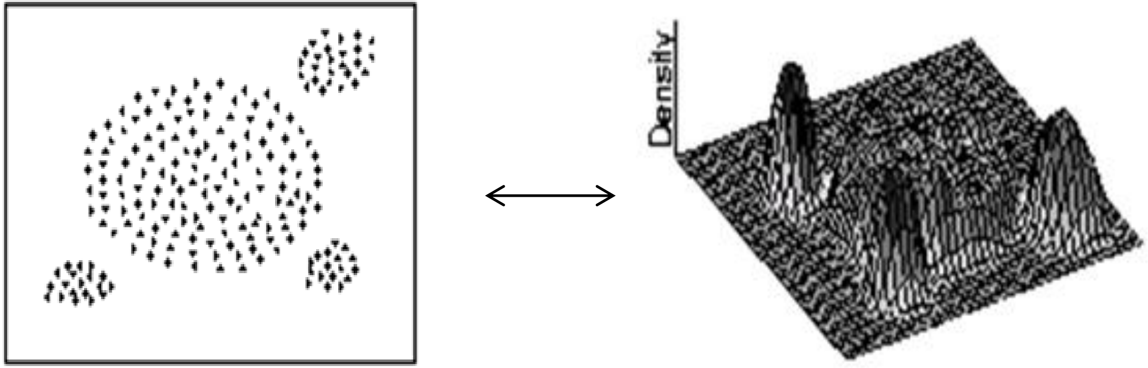


- Hierarchical Approach
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Agglomerative, Diana, Agnes, BIRCH, ROCK, ……



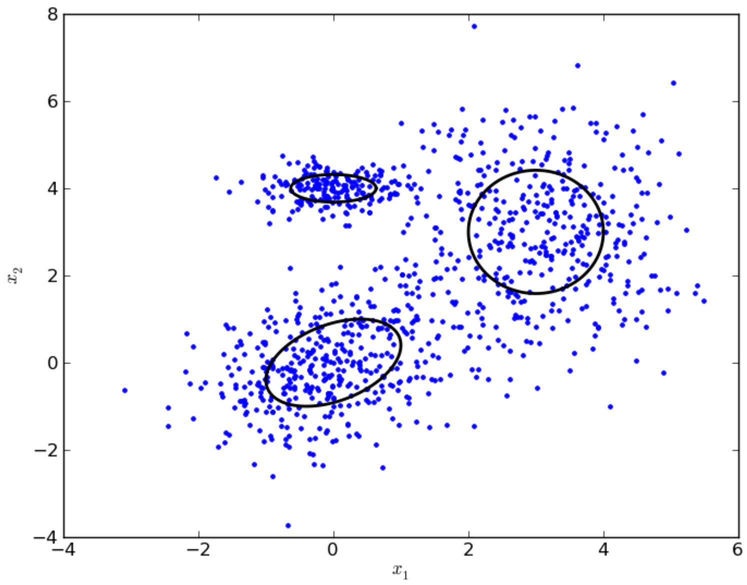
Major Clustering Approaches

- Density-based Approach
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue, ……

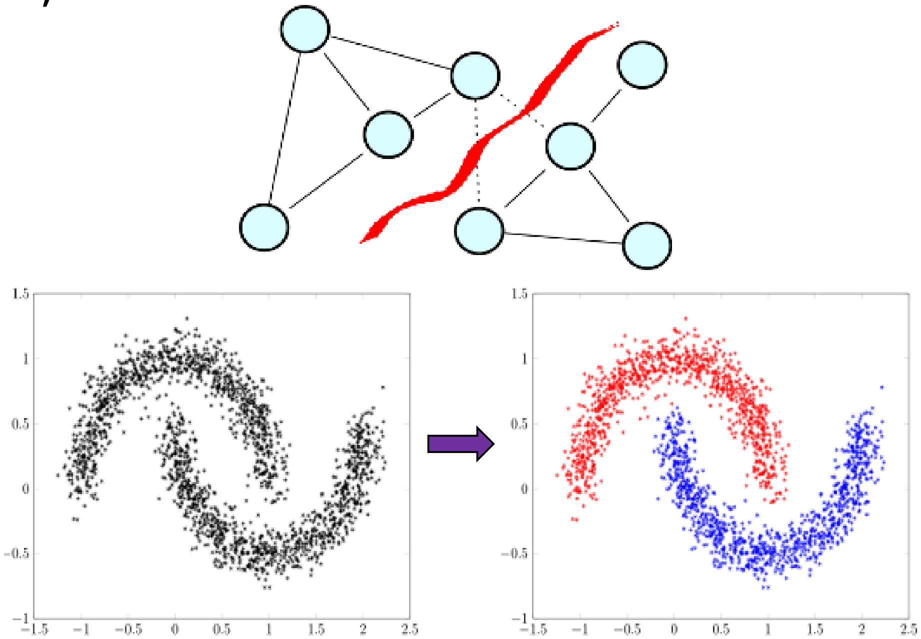


Major Clustering Approaches

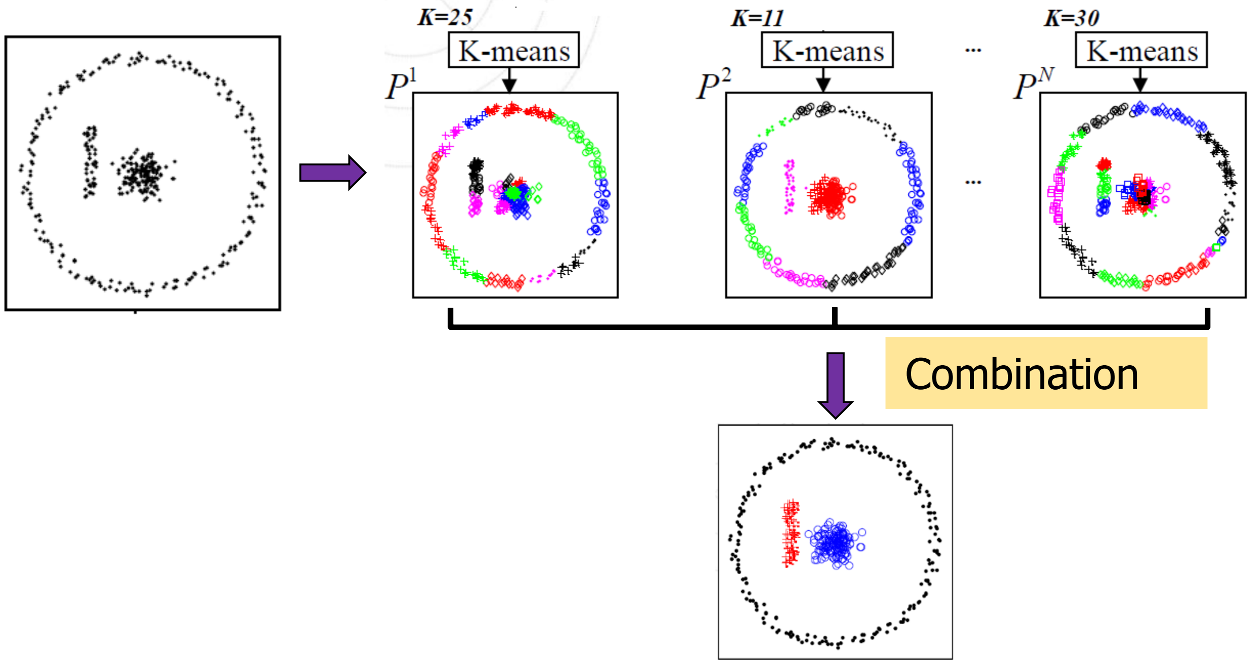
- Model-based Approach
  - A generative model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: Gaussian Mixture Model (GMM), COBWEB, ……



- Spectral Clustering Approach
  - Convert data set into weighted graph (vertex, edge), then cut the graph into sub-graphs corresponding to clusters via spectral analysis
  - Typical methods: Normalized-Cuts, ……



- Clustering Ensemble Approach
  - Combine multiple clustering results (different partitions)
  - Typical methods: Evidence-accumulation based, graph-based ……



- Introduction
- Data Types and Representations
- Distance Measures
- Major Clustering Approaches
- **<u>Implementation</u>**
- Summary

KNN with TensorFlow

```
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

import tensorflow as tf
import numpy as np

num_training_images = 5000 # maximum: 55000
num_testing_images = 200 # maximum: 5000
num_pixels_MNIST = 28*28
pixel_train, onehot_train = mnist.train.next_batch(num_training_images) # Train
pixel_test,  onehot_test  = mnist.test.next_batch(num_testing_images) # Test, # num_testing_images == len(pixel_test)
print('pixel_train:', pixel_train.shape) # (5000x784)
print('onehot_train:', onehot_train.shape) # (5000x10)
print('pixel_test:', pixel_test.shape) # (200x784)
print('onehot_test:', onehot_test.shape) # (200x10)

TRAIN = tf.placeholder("float", [None, num_pixels_MNIST]) # None: batch size, 784: num of images
TEST  = tf.placeholder("float", [num_pixels_MNIST]) # 784: num of images

distance = tf.reduce_sum(tf.abs(tf.add(TRAIN, tf.negative(TEST))), reduction_indices=1) # print(distance), 5000-by-1
K=5
values,indices=tf.nn.top_k(-distance,k=K,sorted=False)
accuracy = 0.
```

KNN with TensorFlow

```python
with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(num_testing_images):
        knn_index = sess.run(indices, feed_dict={TRAIN: pixel_train, TEST: pixel_test[i,:]})

        look_up = np.zeros(10) #[0,0,0,0,0,0,0,0,0,0]
        for ii in np.argmax(onehot_train[knn_index],axis=1) :
            look_up[ii] += 1

        prediction = np.argmax(look_up)

        print("Test: ", i, "Prediction: ", prediction, "Actual: ", np.argmax(onehot_test[i]))
        if prediction == np.argmax(onehot_test[i]):
            accuracy += 1./num_testing_images

    print("Accuracy: ", accuracy*100 ,"percentage")
```

- Introduction
- Data Types and Representations
- Distance Measures
- Major Clustering Approaches
- Implementation
- **<u>Summary</u>**

- Clustering analysis groups objects based on their (dis)similarity and has a broad range of applications.
- Measure of distance (or similarity) plays a critical role in clustering analysis and distance-based learning.
- Clustering algorithms can be categorized into partitioning, hierarchical, density-based, model-based, spectral clustering as well as ensemble approaches.
- There are still lots of research issues on cluster analysis;
  - finding the number of "natural" clusters with arbitrary shapes
  - dealing with mixed types of features
  - handling massive amount of data – Big Data
  - coping with data of high dimensionality
  - performance evaluation (especially when no ground-truth available)