Predictive Analytics

Chapter Seven

Data Clustering Algorithms

"Birds of a feather flock together."

Anasse Bari, Ph.D.

Learning Outcomes

- Learning the notion of data clustering
- Acquiring understanding of widely used clustering algorithms

Outline

- Defining Data Clustering
- Data Clustering and its relationship with Predictive/Data Analytics
- Data Clustering Algorithms Requirements
- Data Clustering Algorithms
  - Partitioning Algorithms
    - K-means
    - K-modes Huang, Joshua Zhexue. "Clustering Categorical Data with k-Modes." (2009): 246-250.
    - Hierarchal Algorithms
  - Density-based Algorithms
    - DBSCAN
  - Grid-based Algorithms
  - Biologically Inspired Algorithms
    - Birds Flocking Algorithms for Data Clustering
    - Flock by Leader Machine Learning Algorithm (by Anasse Bari et.al)
  - Large Scale Clustering Algorithms
    - BFR
    - CURE

Defining Data Clustering

- Data Clustering is a technology that aims to extract natural groupings of similar data objects from a given dataset.

- The groupings are often called clusters or data clusters.

- Every cluster is a subset of the given dataset that contains data objects that are related and similar.

- In many cases, every cluster has a **representative(s)** that best represents the data elements in a given cluster.

- Extracting data clusters from a large body of data (dataset) is *Data Model*:

  It represents a dataset by clusters and cluster representatives.

Formal Definition of a Data Clustering Problem

Consider a dataset $S\{s_1, \dots, s_n\}$ **consisting of n data points.** ***(offline or online setting)***

Data points s_i can be called *objects, cases, tuples, transactions, or instances.*

Every data point s_i **has a d number of attributes.**

Synonyms of attributes are: *features, variables, dimensions, and fields.*

Each data point s_i **is a vector in a d-dimensional space.**

From a conceptual point of view, this point-by-attribute data format corresponds to an n*d matrix, which constitutes the input to the clustering algorithm.

Goal: to assign points to a finite system of k-subsets (clusters).

In most (but not all) cases subsets do not intersect, and their union produces a full dataset with the possible exception of outliers:

Every is a cluster of points that belong to S. The general definition of the main problem was given at an abstract level.

$$S = C_1 \cup C_2 \cup C_3 \dots . C_k \cup C_{outliers}$$

Defining Data Clustering (cont'd)

- Data clustering exposes the inner structure in the data by extracting clusters of similar data objects from a dataset.

- The inner structure (data clusters) is an essential step towards *formulating ideas and hypotheses* about the structure of your data and *deriving insights* to better understand it.

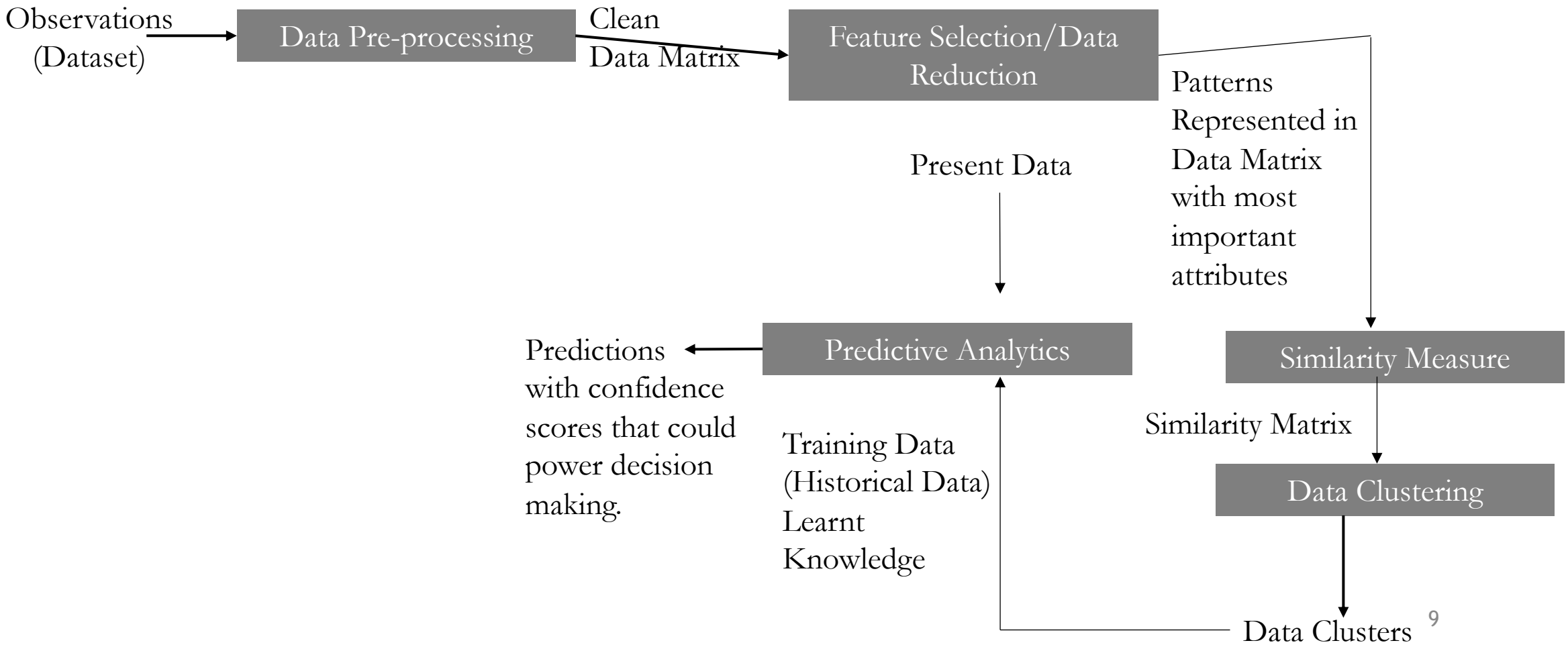
Defining Data Clustering (cont'd)

- In some cases, your data analytics task may be to seek a partition of a dataset into groups of similar items.
- Market segmentation is strategy in marketing that aims at analyzing market data and extracting subset of consumers that share common characteristics (needs, interests, behaviors, priorities…)
- Market segmentation defines which consumers to target and how to target them.
- Identifying clusters of similar customers can help marketers develop a marketing strategy that addresses the needs of specific clusters.

Data Clustering and its Relationship with Predictive Analytics

- Predictive Analytics learns from the past (historical data, past discovered patterns...), and learns present data to predict the future.
- Data Clustering can be used as **(1) a stand-alone tool to get insights about the data distribution and structure or (2) as a preprocessing step or learning phase for other building predictive analytics.**
- Data Clustering discovers hidden patterns (data clusters) from data which Predictive Analytics algorithms can leverage to learn from the past to make predictions.
- Data clustering extracts partners, learn types of past data - Predictive Analytics utilizes that learning to link new data object to old discovered object to predict its type.

Data Clustering and its Relationship with Predictive Analytics (Cont'd)



Definitions

- Cluster: a collection of data objects
- Data Clustering: the process (algorithms) of grouping data objects into similar objects
- Dataset: Collection of data. In the SQL database world, a dataset could also corresponds to a table or a joined tables (view) from a database
- Data object: a Data object is one instance in the dataset (e.g row in the dataset matrix) can be also called data element, data record, observation.
- Class Label of Data: the cluster label (a type or a category of the data elements)
- Unsupervised Clustering: no predefined classes' label for data clustering
- Supervised Clustering: class labels are defined for the data clustering
- Centroid (Cluster Representative)

Applications of Data Clustering at Glance

- Text Mining
  - Document Categorization
  - Automatic Detection of Topics
  - Summarization
- Web Mining
  - Web log analysis
  - Detection of Similar Access Patterns
- Bioinformatics
  - Gene expression data: detection of cancer genes
- Social Networks Analytics
  - Online Social Community Detection
- Others
  - Image processing
  - Market Analysis

Data Clustering Requirements

- Clustering algorithm shall achieve *High intra-class similarity*.
- Clustering algorithm shall achieve *Low inter-class similarity*.
- The quality of a clustering result depends on both the *similarity measure, the data* and *the clustering algorithms*.
- The quality of a clustering algorithm is also measured by its ability to discover some or all of the hidden pattern – Domain experts validation.

Data Clustering Requirements (Cont'd)

- The clustering algorithm shall be **scalable**.
- The clustering algorithm shall have **the ability to deal with different type of attributes**.
- The clustering algorithm shall be able **to discover of clusters of arbitrary shape.**
- The clustering algorithm shall have **require minimal requirement for domain knowledge to determine input parameters.**
- The clustering algorithm shall be able **to deal with noise and outliers.**
- The clustering algorithm shall **not be sensitive to the order of input records.**
- The clustering algorithm shall have **the ability to handle high dimensionality.**
- The clustering algorithm shall have **the ability to produce results that are interpretable and that can be usable.**

Outline

- Defining Data Clustering
- Data Clustering and its relationship with Predictive Analytics
- Data Clustering Algorithms Requirements
- Data Clustering Algorithms
  - Partitioning Algorithms (based on point's assignment)
    - K-means
  - Hierarchal Algorithms
  - Density-based Algorithms
    - DBSCAN
  - Grid-based Algorithms
  - Model-based Algorithms
  - Biologically Inspired Algorithms
    - Birds Flocking Algorithms for Data Clustering
    - Flock by Leader Machine Learning Algorithm

Data Clustering Algorithms

"**A**n **a**lgorithm **m**ust be **s**een to **b**e **b**elieved" Donald Knuth

Prerequisites

- Data Pre-processing

- Data Structures
  - Data Matrix
  - Dissimilarity (or similarity) Matrix

- Similarity Measures

Clustering Algorithms

- Partitioning Algorithms
- Hierarchy Algorithms
- Density-based Algorithms
- Grid-based Algorithms
- Model-based Algorithms
- Biologically Inspired Algorithms

Partitioning Algorithms

- Purpose: a partitioning based data clustering algorithm constructs iteratively partitions and then evaluate them by some criteria.
- Input: K number of clusters to discover and a Finite set $X \subseteq \Re^d$ of n data objects $X\{x_1, x_2, \ldots, x_n\}$
- Output: Finite set of Clusters $C\{c_1, c_1, \ldots, c_k\}$ and cluster representatives for every
- Goal: Minimize the objective function

$$ObjectiveFunction = \sum_{1}^{k} WeightofGeneratedClusters_{c_k} = \sum_{1}^{k} \sum_{(j,i)}^{|c_k|^2} \|Q_i - Q_j\|$$
$$Q_i, Q_j : Q_i, Q_j \in c_i \subseteq G_p$$

Note that objective function can be also expressed by minimizing the sum of the distance of every point to its "representative object" in each cluster (e.g. Euclidian Distance)

K-means Algorithm

- K-means is a partitioning based data clustering algorithm that is widely used in practice
- Algorithm Intent: Given a Dataset S, Given a K (integer), extract a partition of K clusters from S such a way to optimize the objective function.
- Input: a dataset S (data matrix)

  K number of cluster
- Output:
  - Cluster representatives (also called means or centroids).

    Each cluster is represented by the centroid of the clusters

    The algorithm converges to final and stable centroids

    How to derive final clusters?

    (from the cluster representatives, elements in every cluster can be derived by assigning each point to the nearest cluster representative.)

K-means Algorithm (Cont'd)

Algorithm:

Step 0: Randomly assign K data objects to be initial centroids $\boldsymbol{C_j}$ **: j € [0,k]**

Step 1: For every data object (vector) $\boldsymbol{O_i}$ **: i € [0,n]**

Find the nearest $\boldsymbol{C_j}$ centroid to $\boldsymbol{O_i}$ using a similarity distance $\boldsymbol{d}$

Assign $\boldsymbol{O_i}$ to $\boldsymbol{C_j}$'s cluster

Step 2: Re-computer the new centroids (mean data object: averaging the feature vectors of all points in a cluster)

Step 3: Go back to Step 1 and stop when the clusters do not change (the data objects in each clusters do not change)

K-means Algorithm (Cont'd)

Example:

Consider a data set of four types of medicines. Each medicine has two features (attributes): pH-index and Weight Index.

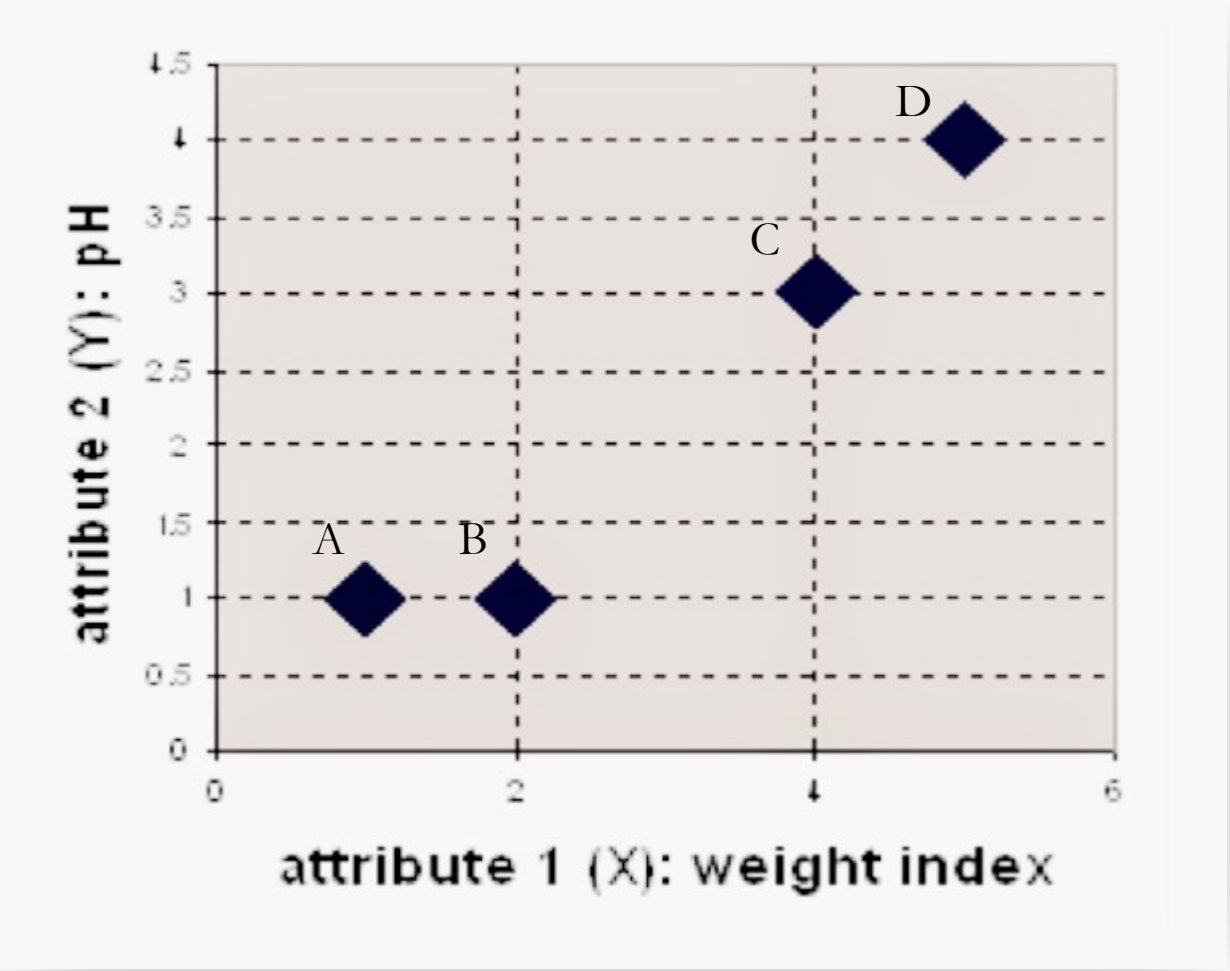
- Requirement: Group these data object (medicines) into two groups
- Input: K=2 and the dataset (data matrix shown below):

Medicine	Weight	pH-Index
A	1	1
B	2	1
C	4	3
D	5	4

K-means Algorithm (Cont'd)

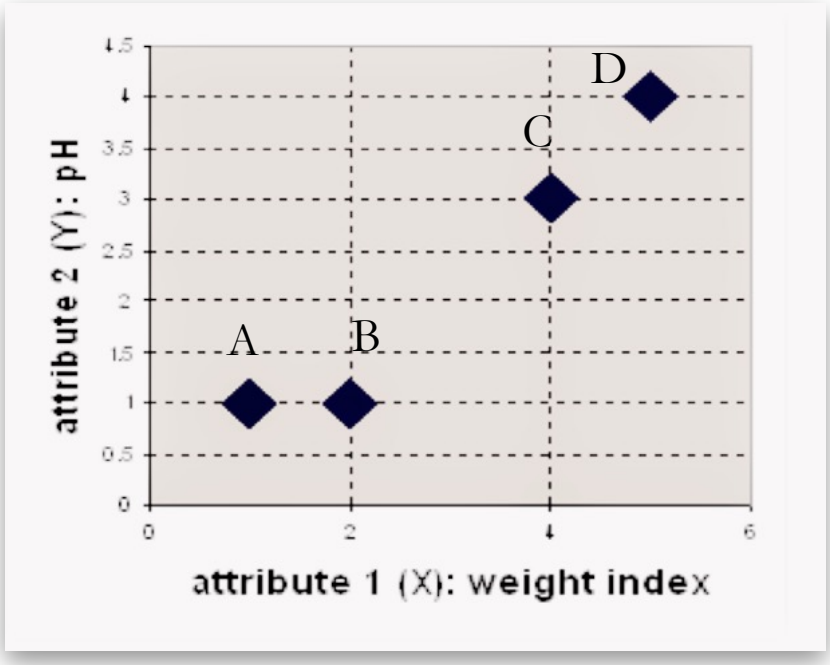
Data Matrix

Medicine	Weight	pH-Index
A	1	1
B	2	1
C	4	3
D	5	4



K-means Algorithm (Cont'd)

Medicine	Weight	pH-Index
A	1	1
B	2	1
C	4	3
D	5	4



Step 0: Initialize random centroids

$$c_1 = A, c_2 = B$$

Step 1: Assign data objects to the nearest centroids clusters using a similarity measure (In this case: Euclidean Distance)

$$D^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (2,1) & group-2 \end{matrix}$$

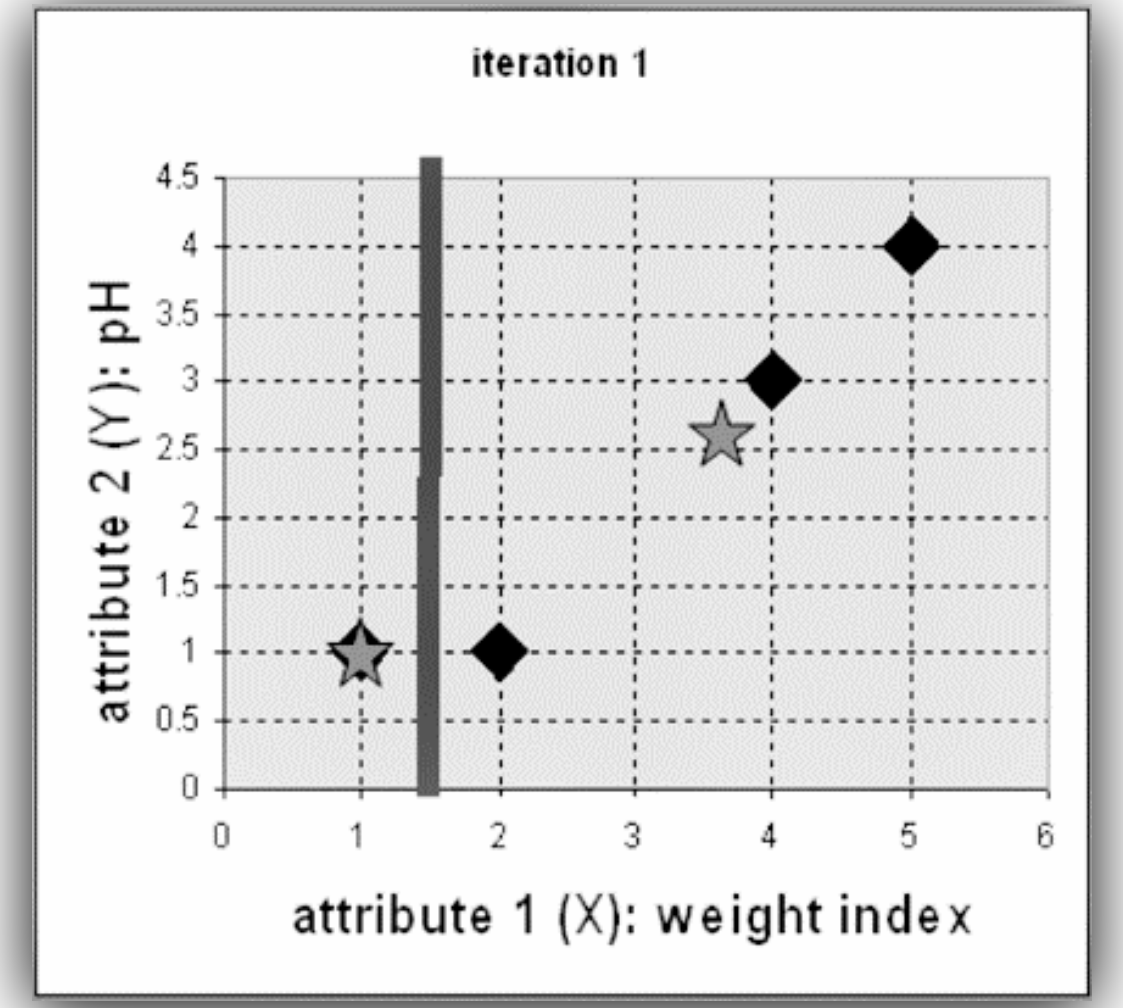
$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

$$d(D,c_1) = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

$$d(D,c_2) = \sqrt{(5-2)^2 + (4-1)^2} = 4.24$$

K-means Algorithm (Cont'd)



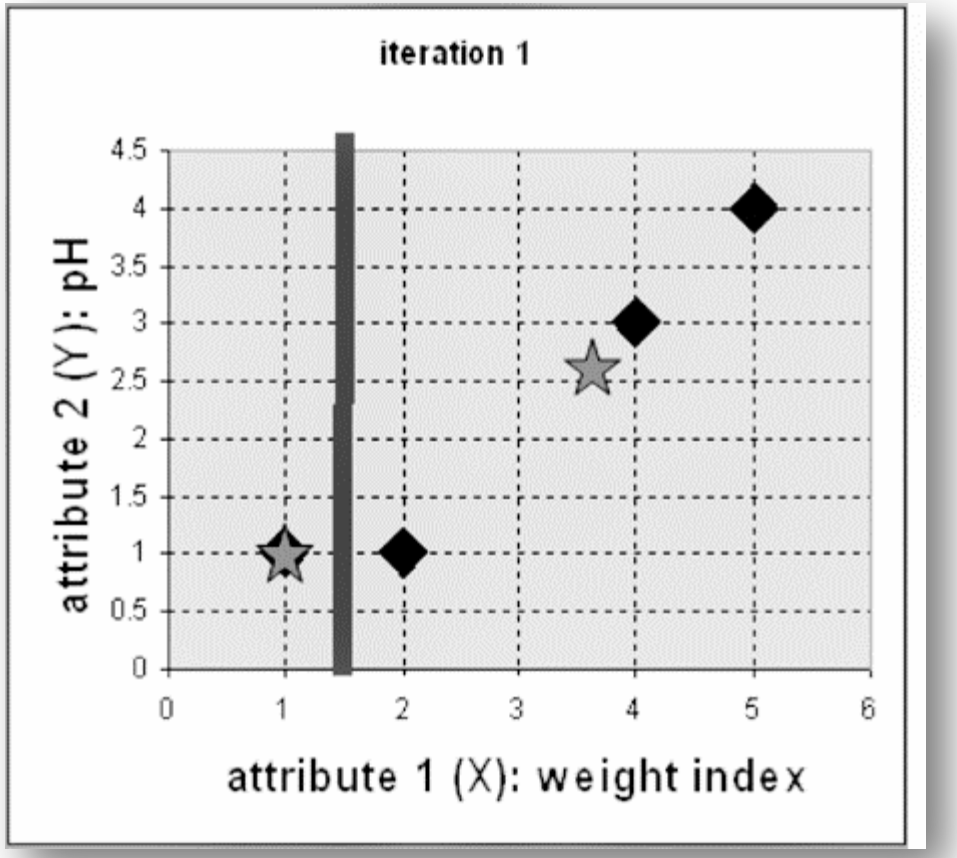
Step 2: Compute new centroids based on the new members in each cluster.

$$c_1 = (1,\ 1)$$

$$c_2 = \left(\frac{2+4+5}{3},\ \frac{1+3+4}{3}\right)$$

$$= \left(\frac{11}{3},\ \frac{8}{3}\right)$$

K-means Algorithm (Cont'd)



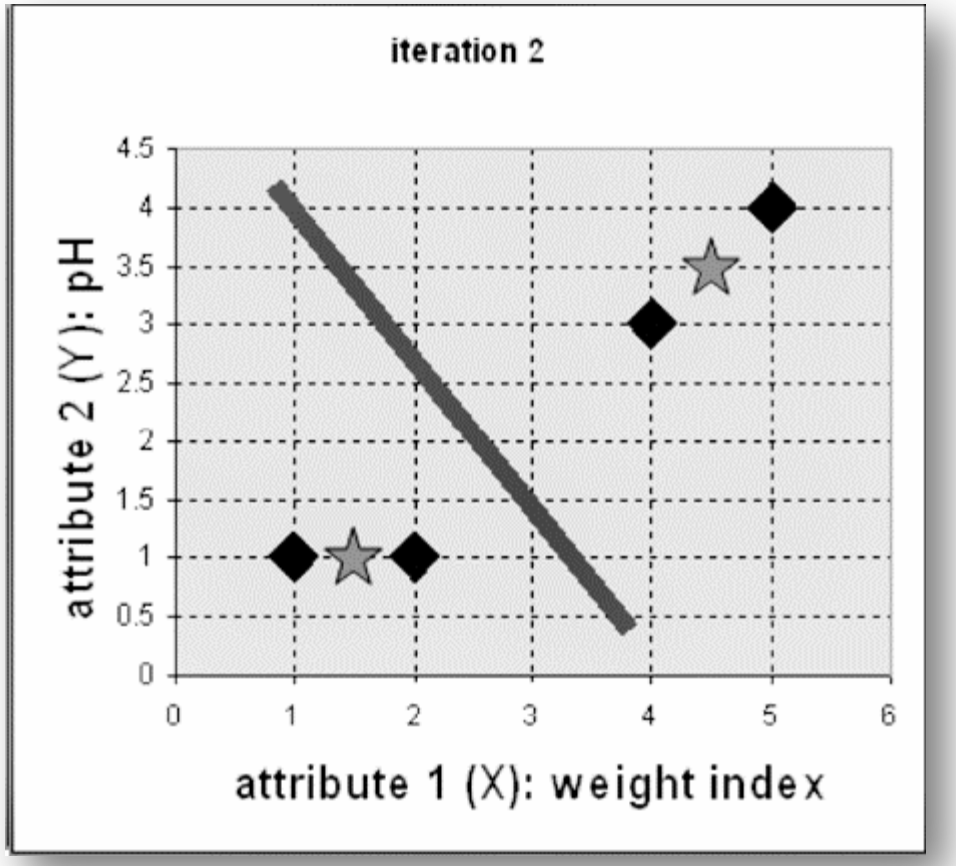
Step 2 (cont'd): Renew Cluster membership now that there are new centroids. Compute the distance of all objects to the new centroids. Re-assign data object to the new clusters.

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

K-means Algorithm (Cont'd)

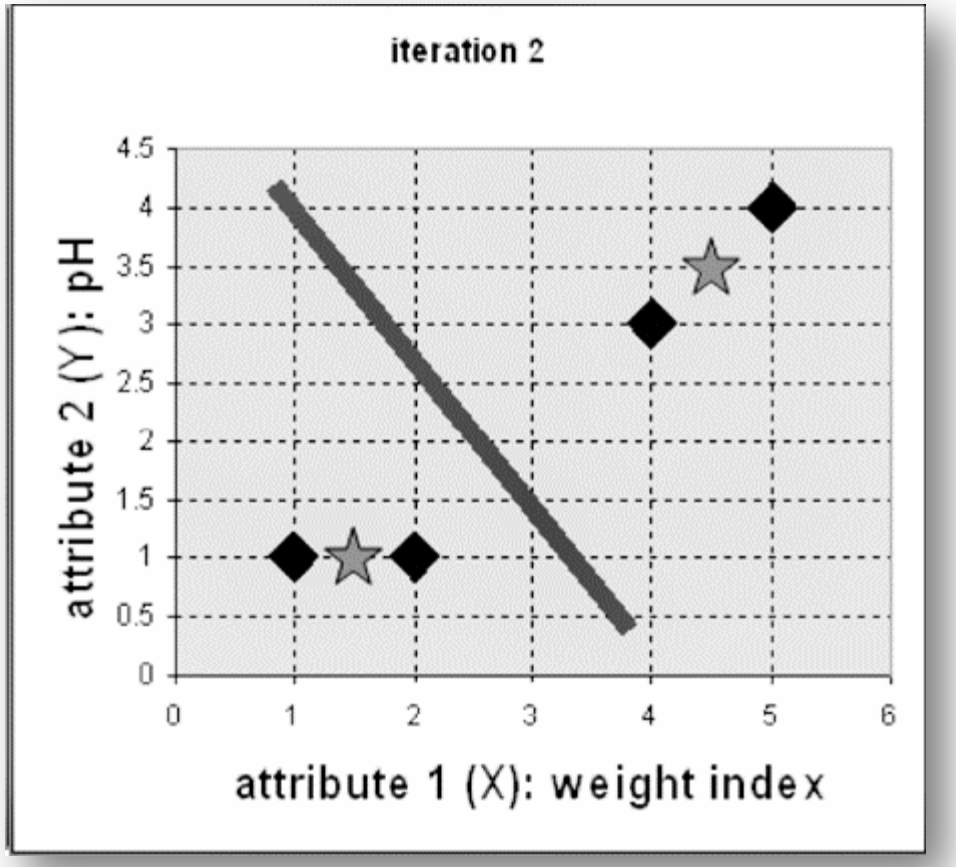


Step 3 (cont'd): Calculate the new centroids based on new membership

$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1\frac{1}{2}, 1)$$

$$c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = (4\frac{1}{2}, 3\frac{1}{2})$$

K-means Algorithm (Cont'd)



Step 3 (cont'd): Repeat same process until convergence. Stop due to no new assignment Membership in each cluster no longer change

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1\frac{1}{2}, 1) & group-1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

K-means Stopping Criteria

- Stopping criteria (Convergence):

  Possibilities:

  1. No change in the members of all discovered clusters
  2. Stop when the squared error is less than some small threshold value

K-means Stopping Criteria

- Stopping criteria (Convergence):

  Possibilities:

  1. No change in the members of all discovered clusters
  2. Stop when the squared error is less than some small threshold value

Properties of K-means

- K-means is an algorithm that is guaranteed to convergence and local optimum.
  - Assigned Reading on proof.
- K-means' running time is relatively performant:

  Time complexity **O(tKn),** where n is number of objects, K is number of clusters, and t is number of iterations. Normally, K, t << n.

K-means Disadvantages

- Need to specify K, the number of clusters, in advance
  - **How to select initial centroids?**
- Unable to handle noisy data and outliers (K-Medoids algorithm)
- Not suitable for discovering clusters with non-convex shapes
- The performance is determined by initialization and appropriate distance measure
- Forces all the points to join clusters (ignoring outliers)
- Applicable only when mean is defined, then what about categorical data? (K-mode algorithm)

K-means Variants

- There are several variants of K-means to overcome its weaknesses
- K-Medoids: resistance to noise and/or outliers
- **K-Modes: extension to categorical data clustering analysis**

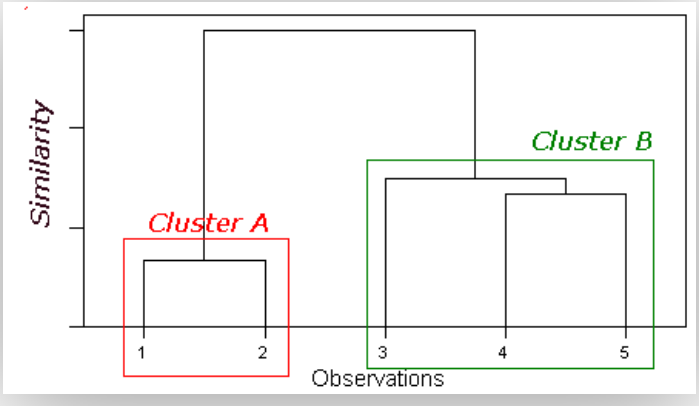
  Huang, Joshua Zhexue. "Clustering Categorical Data with k-Modes." (2009): 246-250.

- CLARA & BFR : extension to deal with large data sets
- Mixture models (EM algorithm): handling uncertainty of clusters

Hierarchical Algorithms

- Purpose: Generate a hierarchical decomposition of the set of data (or object) using some criterion.
- Input: The data matrix (Data set)

  The number of clusters k is not required as an input



- Output: Clustering tree, also called a dendrogram

  Leaves of the tree represent the individual objects

  Internal nodes of the tree represent the clusters

Hierarchical Algorithms (Cont'd)

- There are two main types of hierarchical clustering techniques: (1) *Agglomerative (bottom-up)* and (2) *Divisive (top-down)*
- *Agglomerative (Bottom-up)*
  - place each object in its own cluster (a singleton)
  - merge in each step the two most similar clusters until there is only one cluster left or the termination condition is satisfied
- *Divisive (top-down):*
  - start with one big cluster containing all the objects
  - divide the most distinctive cluster into smaller clusters and proceed until there are n clusters or the termination condition is satisfied

Hierarchical Algorithms (Cont'd)

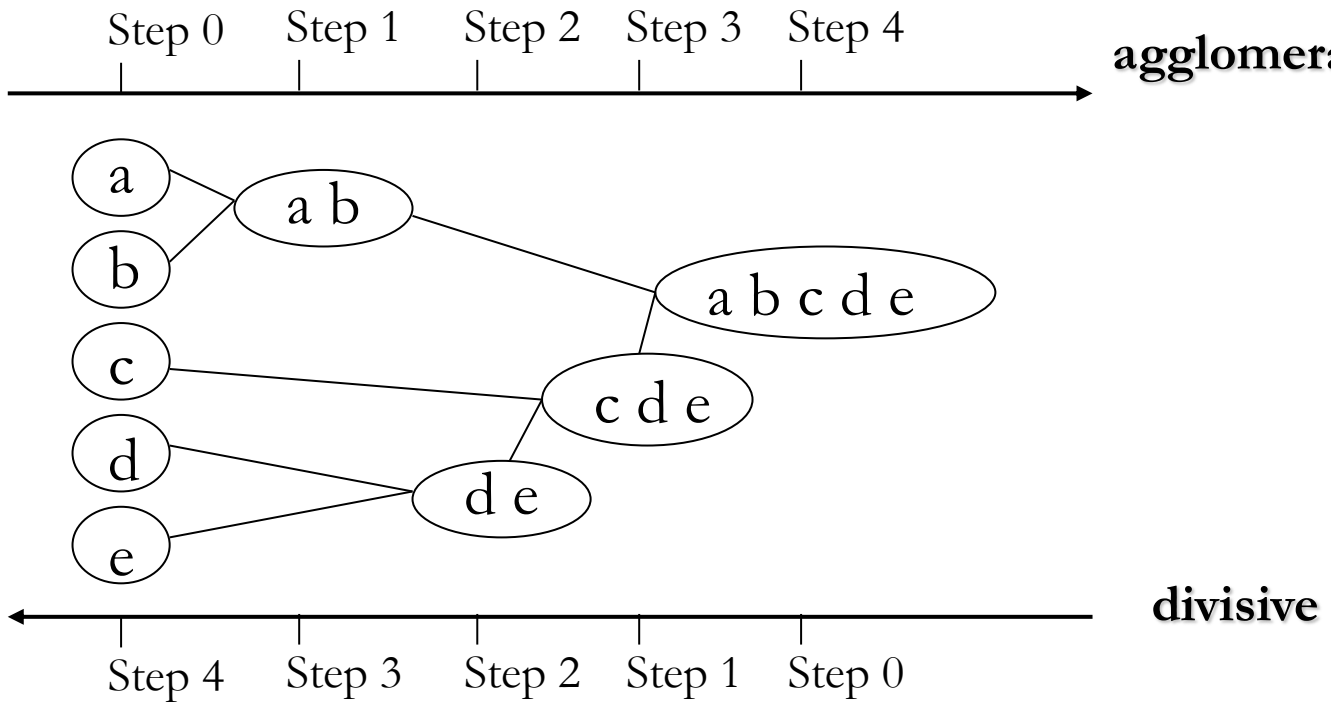
Agglomerative (bottom up)

1. start with 1 point (singleton)
2. recursively add two or more appropriate clusters
3. Stop when k number of clusters is achieved.

Divisive (top down)

1. Start with a big cluster
2. Recursively divide into smaller clusters
3. Stop when k number of clusters is achieved.

Hierarchical Algorithms (Cont'd)



Reference: www.cs.helsinki.fi/u/ronkaine/tilome/.../TiLoMe-020304.pdf

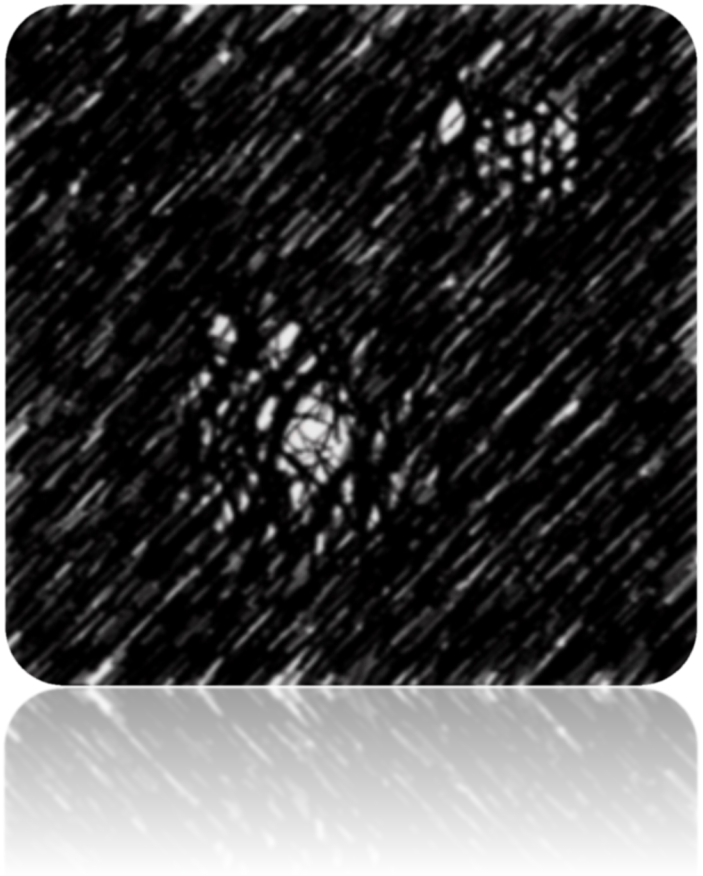
More on Hierarchical Clustering Algorithms

Assigned Reading: http://ijettjournal.org/volume-3/issue-1/IJETT-V3I1P203.pdf

Density-based Data Clustering

Intent: Density-based algorithms discover grouping in dataset based on connectivity and density functions.

Notion of Density: A data grouping can be seen as a ***dense region of data points***, which is separated by less dense regions from other regions of high density.



DBSCAN Algorithm

DBSCAN: Density-based special clustering

- Inspired from human way of discovering clusters.
- DBSCAN has been widely used for cases where data might have noise.

- Input:
  - Size of the neighborhood (Radius).
  - Minimum points to be considered for a cluster (MinPts)
- Output:
  - Core Points (analogy with cluster representatives in partitioning algorithms) – that can be used to generated the data clusters.
  - Noise Points (outliers)

How do we generate the clusters?

DBSCAN Algorithm (cont'd)

Consider the following terminology related to DBSCAN:

- **Density at point Q_i :** number of points within a circle of radius - r
- **Dense Region:** A circle of radius r that contains at least MinPts points
- **Density Edge Points :** Q_1 is directly density-reachable from Q_2 if Q_1 belongs to the neighborhood of Q_2 of radius r
- **Density Connected Points:** Density Q_0 is density-reachable from Q_n if there is exist a set of points Q_1,..., Q_{n-1}, where

  Q_{i+1} is directly density-reachable from Q_i
- **Core, border and noise point (see next slide)**

DBSCAN Algorithm (cont'd)

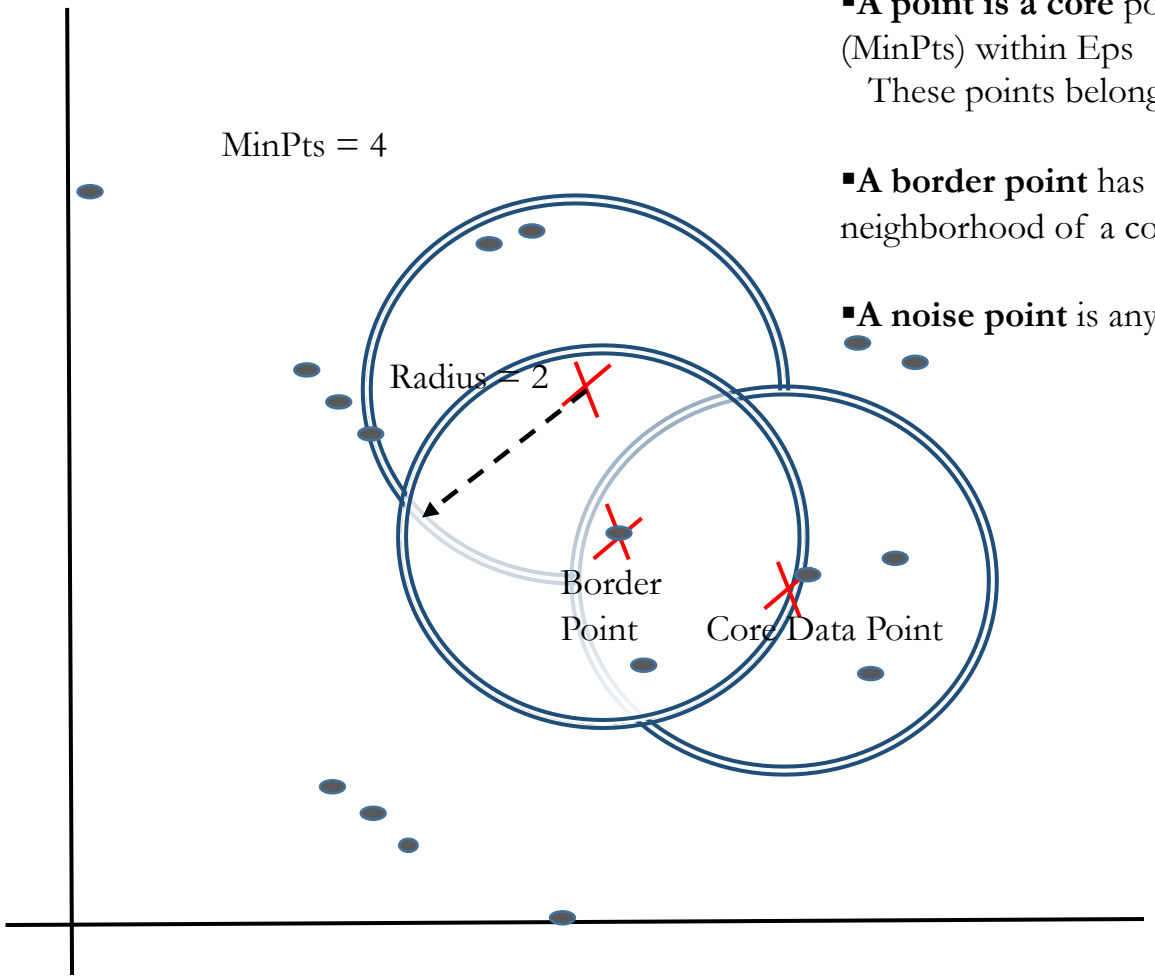
- **A point is a core** point if it has more than a specified number of points (MinPts) within Eps

  These points belong in a dense region and are at the interior of a cluster

- **A border point** has fewer points 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.

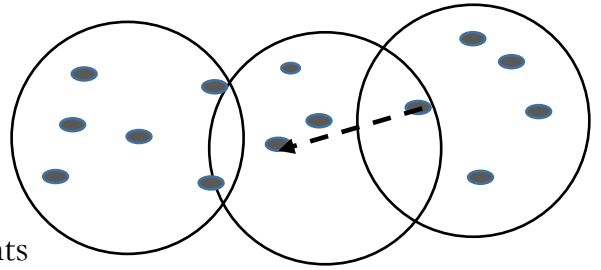
DBSCAN Algorithm (cont'd)



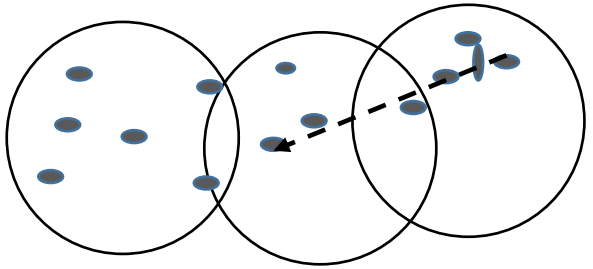
- **A point is a core** point if it has more than a specified number of points (MinPts) within Eps
  These points belong in a dense region and are at the interior of a cluster

- **A border point** has fewer points 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.



Directly **Connected** Points



Density Connected Points

DBSCAN Algorithm at Glance

- **Step 0:** Label points as core, border and noise

- **Step 1:** Report and Eliminate noise points

- **Step 2:** 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.

- **Step 3:** Assign border points to the cluster of the closest core point.

```
DBSCAN(Dataset, r MinPts)
  C = 0
  for each unvisited point P in Dataset mark P as visited
    N = regionQuery(P, r)
    if sizeof(N) < MinPts
      mark P as NOISE
    else
      C = next cluster
      expandCluster(P, N, C,r, MinPts)
```

```
expandCluster(P,N,C,r,MinPts)
  add P to cluster C
  for each point P' in N
    if P' is not visited
      mark P' as visited
      N' = regionQuery(P', r)
      if sizeof(N') >= MinPts
        N = N joined with N'
    if P' is not yet member of any cluster
      add P' to cluster C
```

DBSCAN

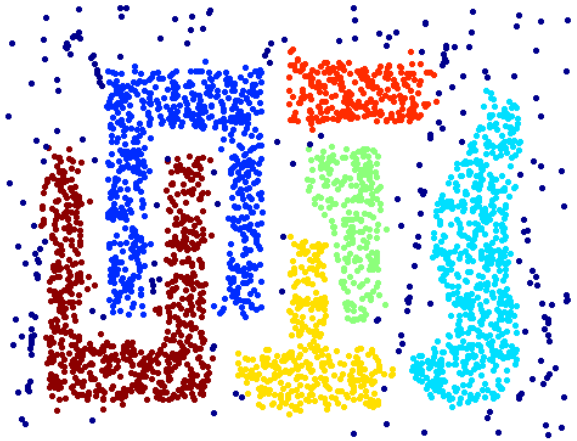
Complexity with indexing structure: O(n*log(n))

Advantages of DBSCAN

according to the experiment cited in the literature

- Handles noise points
- Offers a sense of the density of the data
- Able to handle clusters of different shapes and sizes
- Does not require number of clusters in the data a priori
- Can find arbitrarily shaped clusters
- Even clusters completely surrounded by a different cluster
- Mostly insensitive to the ordering of the points in the database
- Only border points might swap cluster membership

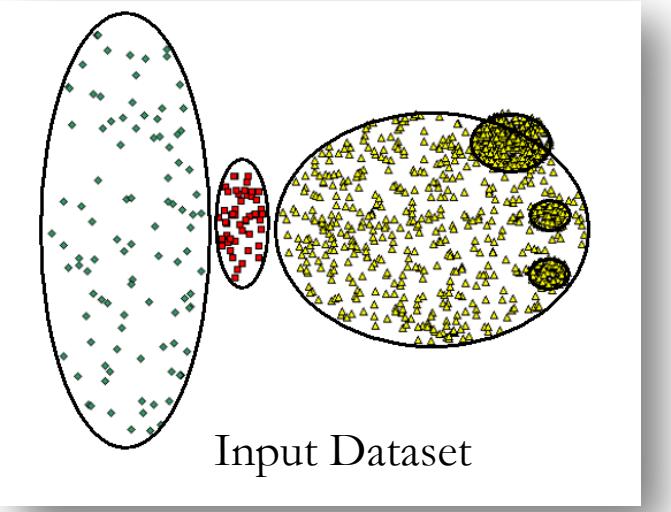
Input Dataset

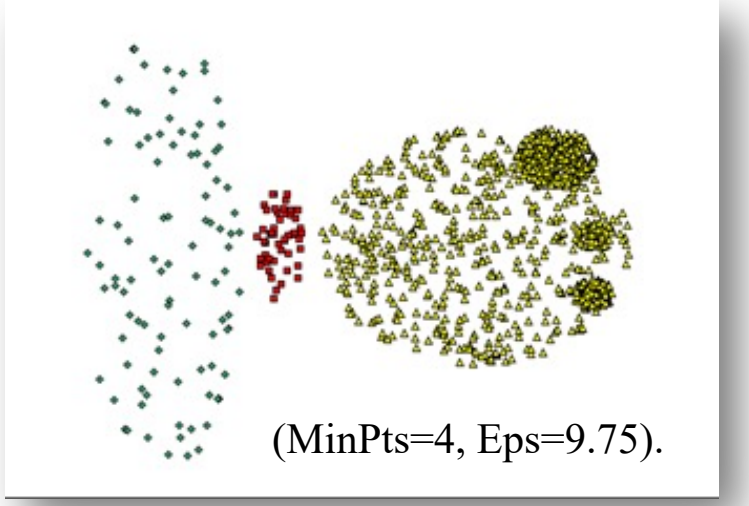


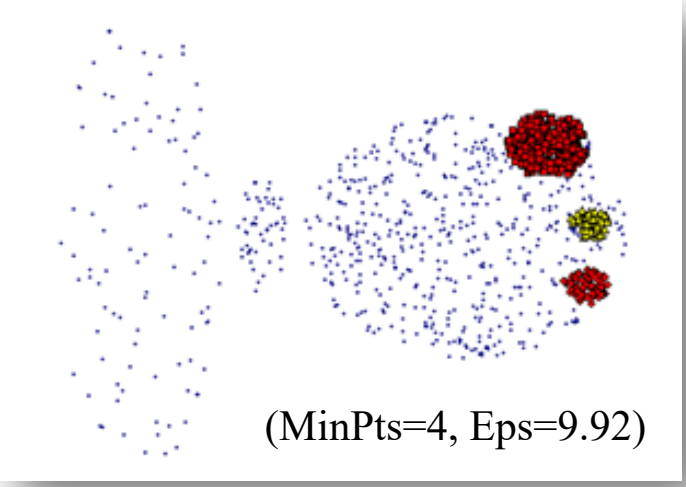
Discovered Clusters

Disadvantages of DBSCAN

- Appropriate parameters R (Eps) and MinPts
  - Numerous experiments indicates best MinPts = 4
- Clustering datasets with large difference in densities
- "Curse of dimensionality"
  - In every algorithm based on the Euclidean distance for high-dimensional data sets


Input Dataset


(MinPts=4, Eps=9.75).


(MinPts=4, Eps=9.92)

Reference: www2.cs.uh.edu/~ceick/ML/Topic9.ppt

Grid-based

Purpose: grouping is based on multiple-level **granularity structure**

Creating imaginary girds in your datasets and cluster according to the density into the grids

Grid-based Clustering Algorithms

Grid-based Clustering Algorithm at a Glance

- Construct cells (set of grid cells) – Define the boundaries
- Assign data objects to cell they belong to.
- Compute the density of each cell.
- Remove cells that has **density** less than a certain threshold ***t***.
- Define a set of grid-cells.
- Assign objects to **the appropriate grid cell** and **compute the density of each cell**.
- Discard cells whose density is below a certain threshold t.
- Build clusters from adjacent groups of dense cells.
- Form clusters from contiguous (adjacent) groups of dense cells using an objective functions that would be minimized.

Advantages of Grid-based Clustering

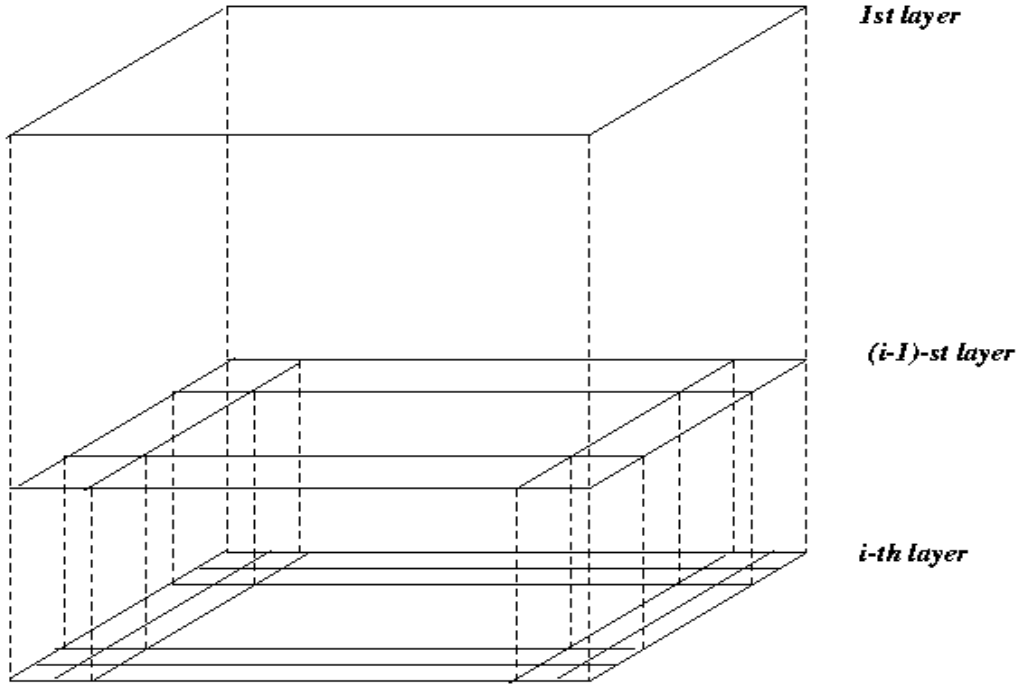
- Relatively Fast:
  - Clustering is performed on summaries and not on individual objects; complexity is usually O(#-populated-grid-cells) and not O(#objects)
  - Easy to determine which clusters are neighbors
  - Shapes are limited to union of grid-cells

Grid-Based Clustering Methods

- Using multi-resolution grid data structure
- Clustering complexity depends on the number of populated grid cells and not on the number of objects in the dataset
- Several interesting methods (in addition to the basic grid-based algorithm)
  - STING (a **ST**atistical **IN**formation Grid approach) by Wang, Yang and Muntz (1997)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)

STING: A Statistical Information Grid Approach

- Wang, Yang and Muntz (VLDB'97)
- The spatial area is divided into rectangular cells
- There are several levels of cells corresponding to different levels of resolution



STING: A Statistical Information Grid Approach

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
  - *count*, *mean*, *s*, *min*, *max*
  - type of distribution—normal, *uniform*, etc.
- Use a top-down approach to answer spatial data queries

STING: Query Processing

- Used a top-down approach to answer spatial data queries
- Start from a pre-selected layer—typically with a small number of cells
- From the pre-selected layer until you reach the bottom layer do the following:
  - For each cell in the current level compute the confidence interval indicating a cell's relevance to a given query;
    - If it is relevant, include the cell in a cluster
    - If it irrelevant, remove cell from further consideration
    - otherwise, look for relevant cells at the next lower layer
- Combine relevant cells into relevant regions (based on grid-neighborhood) and return the so obtained clusters as your answers.

STING: A Statistical Information Grid Approach

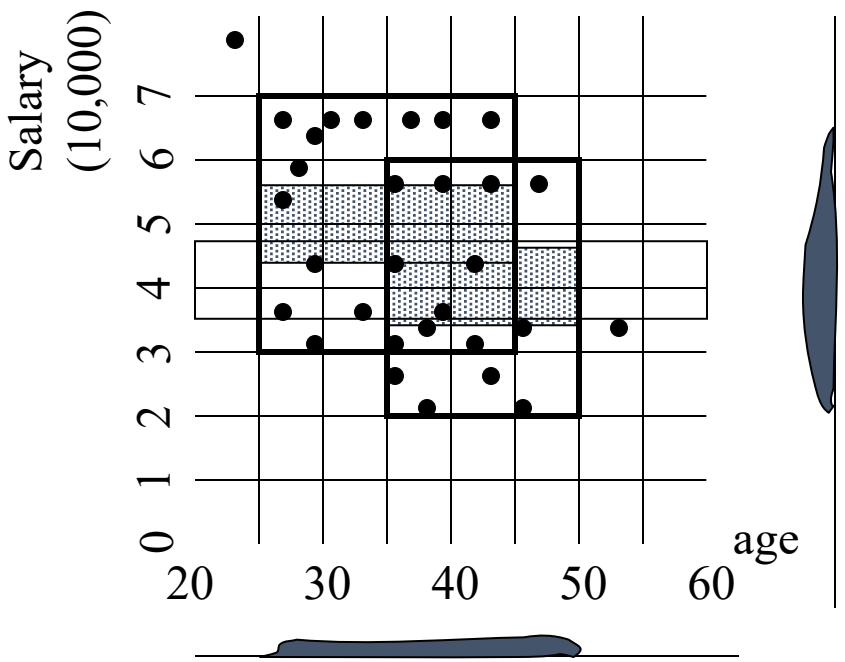
- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - *O(K),* where *K* is the number of grid cells at the lowest level
- Disadvantages:
  - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

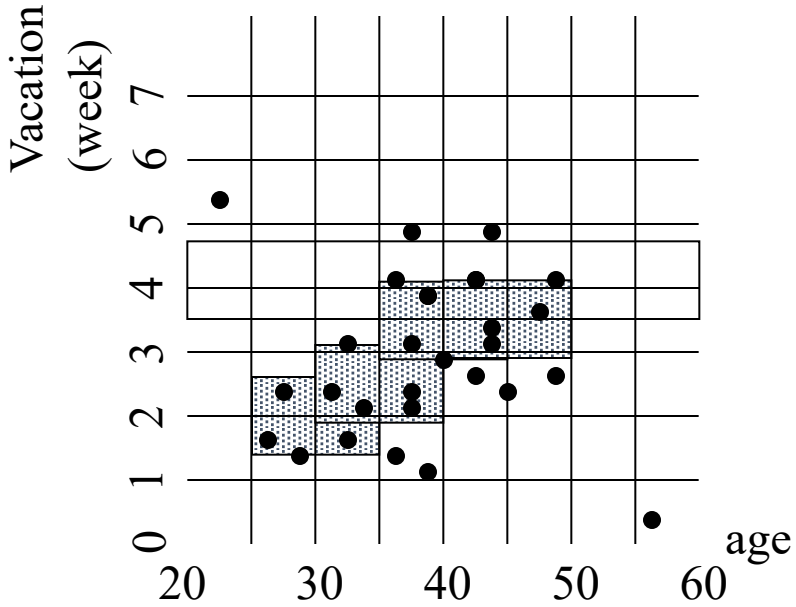
CLIQUE (Clustering In QUEst)

- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98).
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space
- CLIQUE can be considered as both density-based and grid-based
  - It partitions each dimension into the same number of equal length interval
  - It partitions an m-dimensional data space into non-overlapping rectangular units
  - A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter
  - A cluster is a maximal set of connected dense units within a subspace

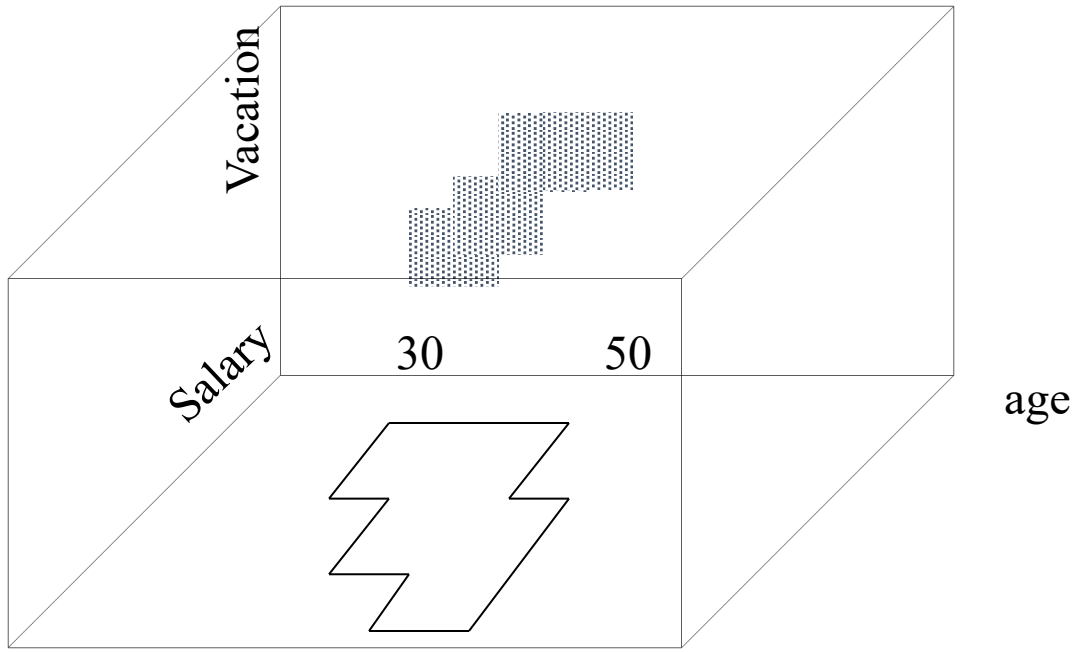
CLIQUE: The Major Steps

- Partition the data space and find the number of points that lie inside each cell of the partition.
- Identify the subspaces that contain clusters using the Apriori principle
- Identify clusters:
  - Determine dense units in all subspaces of interests
  - Determine connected dense units in all subspaces of interests.
- Generate minimal description for the clusters
  - Determine maximal regions that cover a cluster of connected dense units for each cluster
  - Determination of minimal cover for each cluster


Salary
(10,000)
0 1 2 3 4 5 6 7
20 30 40 50 60
age


Vacation
(week)
0 1 2 3 4 5 6 7
20 30 40 50 60
age

$\tau = 3$


Vacation
Salary
30
50
age

Strength and Weakness of *CLIQUE*

- Strength
  - It *automatically* finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
  - It is *insensitive* to the order of records in input and does not presume some canonical data distribution
  - It scales *linearly* with the size of input and has good scalability as the number of dimensions in the data increases
- Weakness
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method

Outline

- Defining Data Clustering
- Data Clustering and its relationship with Predictive Analytics
- Data Clustering Algorithms Requirements
- Data Clustering Algorithms
  - Partitioning Algorithms
    - K-means
  - Hierarchal Algorithms
  - Density-based Algorithms
    - DBSCAN
  - Grid-based Algorithms
  - Biologically Inspired Algorithms
    - Birds Flocking Algorithms for Data Clustering
    - Flock by Leader Machine Learning Algorithm

Biologically Inspired Data Clustering

Reference Talk from Predictive Analytics Business 2015, Bari et. al
Reading assignment and discussion (see class webpage to download the papers mentioned bellow)

[1] "SFLOSCAN: A Biologically Inspired Data Mining Framework for Community Identification in Dynamic Social Networks", IEEE International Conference on Computational Intelligence 2011 (SSCI 2011), 2011.

[2] "Flock by Leader: A Novel Machine Learning Biologically-Inspired Clustering Algorithm", IEEE International Conference of Swarm Intelligence, China 2012; and appears as book chapter in Advances in Swarm Intelligence, 2012 Edition – Springer.

The BFR Algorithm

Large-Scale Data Clustering

Extension of k-means to large data

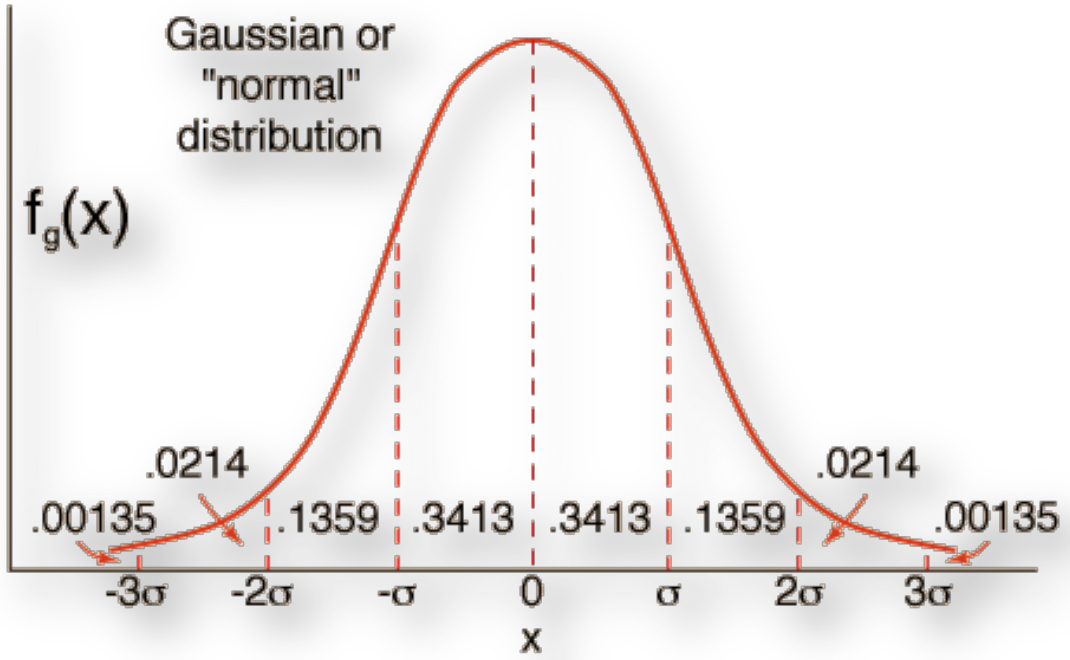
BFR Algorithm (Cont'd)

- Named after Bradley Fayyad and Reina (more info about the co-author)
- It is a variation of K-means clustering algorithm to handle very large datasets (especially datasets that Do NOT fit all at once in memory)
- One constraint: The BFR algorithm assumes that the clusters to be discovered are distributed around their centroid in Euclidean space.

More details:

- Clusters are in the form of ellipses
- Assumes that points belonging to the same cluster are positioned around the centroid of the cluster they belong to and along every dimension.
- Assumes that the standard deviation across different dimension may not be the same across all dimensions.

BFR Algorithm (Cont'd)



- The figure show a variable x, and it is frequency distribution
- The mean is zero and the standard deviation is sigma
- Notice that more than 60% of the points are about one sigma from the mean

BFR Algorithm (Cont'd)

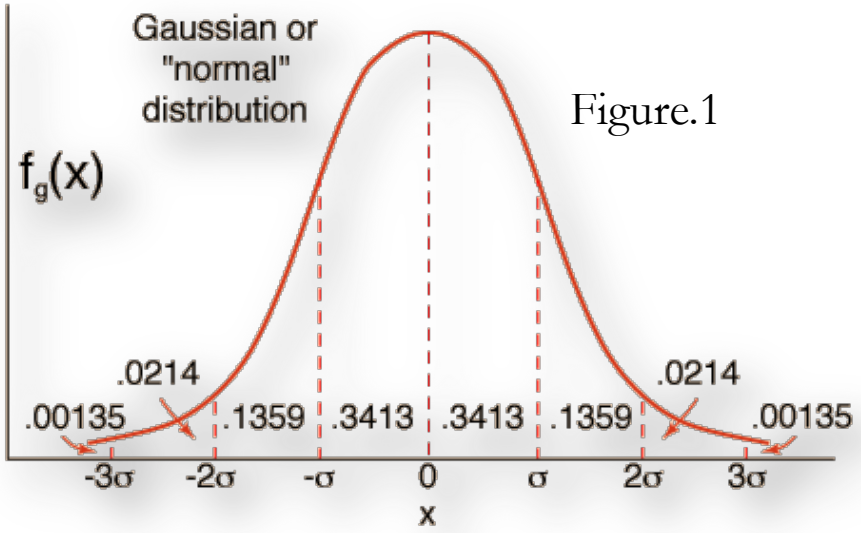
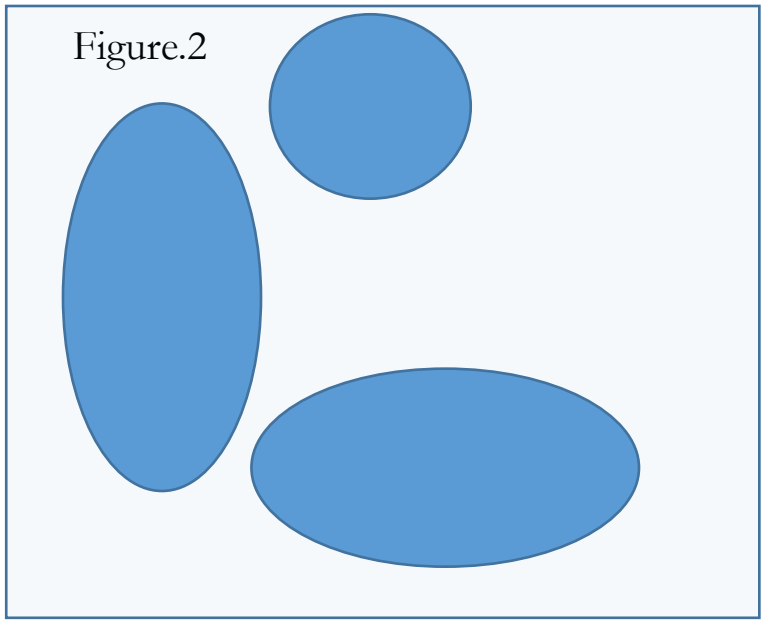


Figure.1

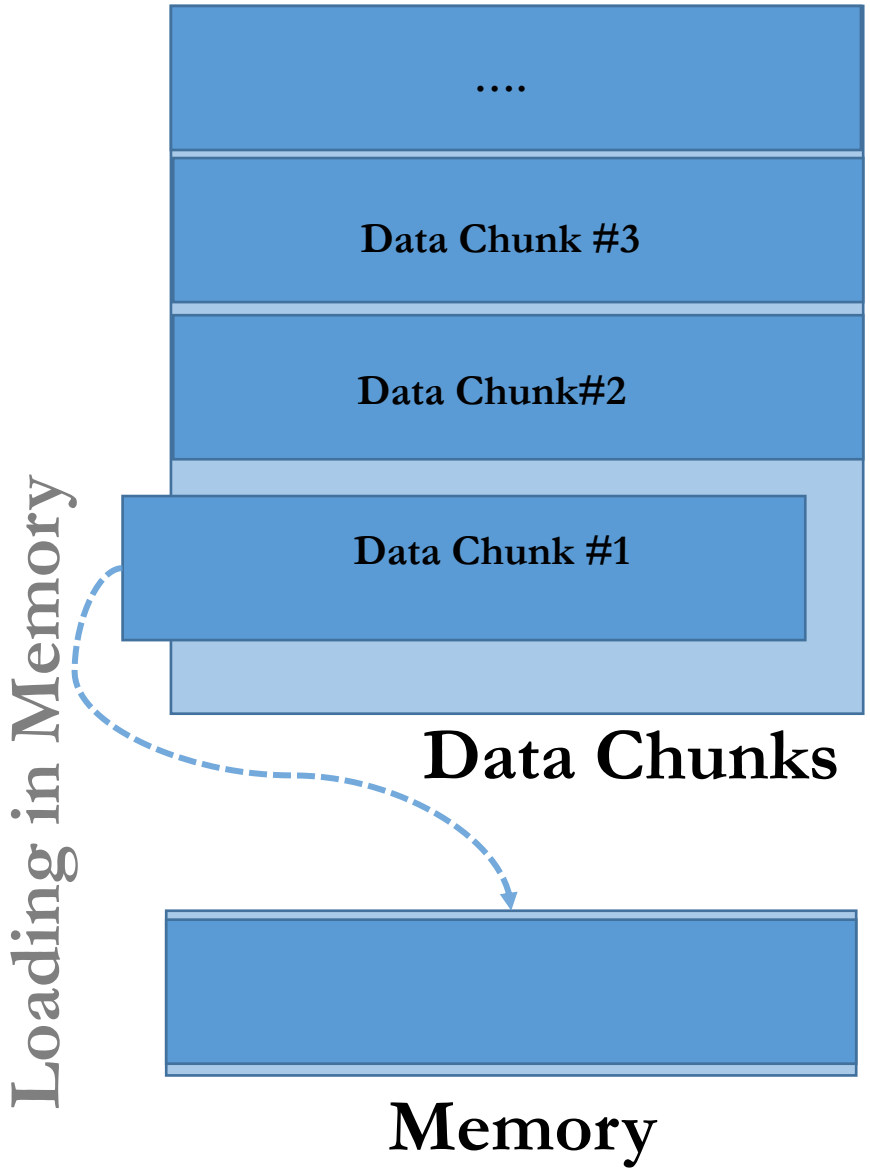


- Notice that more ***than 60% are about one sigma from the mean***
- BFR assumes normal distribution around the cluster mean
- BFR assumption on normal distribution for the clusters means the clusters of shapes in
- Figure 2. (circles or ellipses like shapes – high standard deviation across X or Y axis..)

How does BFR Algorithm work?

Input: **Data** ***(Large Data that can not FIT all at once in Memory)***

- Chunks of Data (data points are read from disk one memory full at a time.
- BFR is designed to maintain only metadata of each memory read-data (the set of data points that could fit in memory) More on the metadata on next slide.
- Once data is being loaded in memory, metadata is being generated. That is the only peace of data that is being kept on file (memory)
- Data points are NOT being story in Memory



How does BFR Algorithm work? (cont'd)

- Data points ready at you memory size at a time
- Only metadata is being saved and kept in memory
- Initial data points are being loaded into memory and metadata about them is being generated as follows:
  - Select k number of initial centroids (similar to k-means)
  - Select a small random sample
  - Clusters the small sample based on the initial centroids
  - You can use any clustering algorithm from literature to cluster those points
  - Next..next slide

BFR Metadata (Cluster summary with three sets)

Everytime you load data chunk into memory, the algorithm summarizes these points into three sets:

- **Discard Set**
- **Compression Set**
- **Retained Set**

BFR Metadata (Cluster summary with three sets)

Discard Set (DS) the set contains the set of points that will be discarded and will not be stored. The points that are of a high proximity to the centroid will be summarized and discarded.

BFR Metadata (Cluster summary with three sets)

Compression set (CS) consists of points that are not at a close proximity to any centroid (initial centroids) and are close to each other. These points can be considered as points of their own cluster. These points are also summarized but not assigned to any existing cluster (they are being compressed)

BFR Metadata (Cluster summary with three sets)

Retained set (CS) consists of points that stored in memory. Those points are being summarized.

BFR Metadata (Cluster summary with three sets)

What do we mean by ***"summarized sets"*** ? Next Slide…

Overview:

The discard set : points close enough to a centroid to be summarized.

The compression set : groups of points that are close together but not close to any centroid. They are summarized, but not assigned to a cluster.

The retained set : isolated points.

BFR Metadata: *Cluster summary with three sets*

Metadata that is being kept in memory consists of:

- N: the number of points
- Vector SUM: whose ith component is the sum of the coordinates of the points in the ith dimension
- SUMSQ: i^{th} component = sum of squares of coordinates in i^{th} dimension

For each cluster, the discard set (DS) is summarized by:

- *N, Vector SUM and SUMSQ*

BFR Metadata: *Cluster summary with three sets*

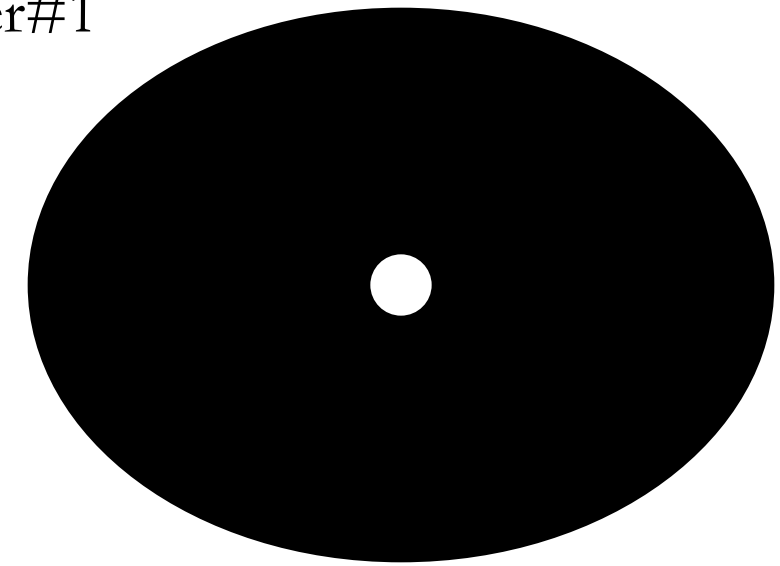
Metadata that is being kept in memory consists of:

For each cluster, the discard set (DS) is summarized by:

- *N, Vector SUM and SUMSQ*

Variance of a cluster's discard set in dimension i can be computed by: (SUMSQ i / N) – (SUM i / N) 2
And the standard deviation is the square root of that.

Cluster#1



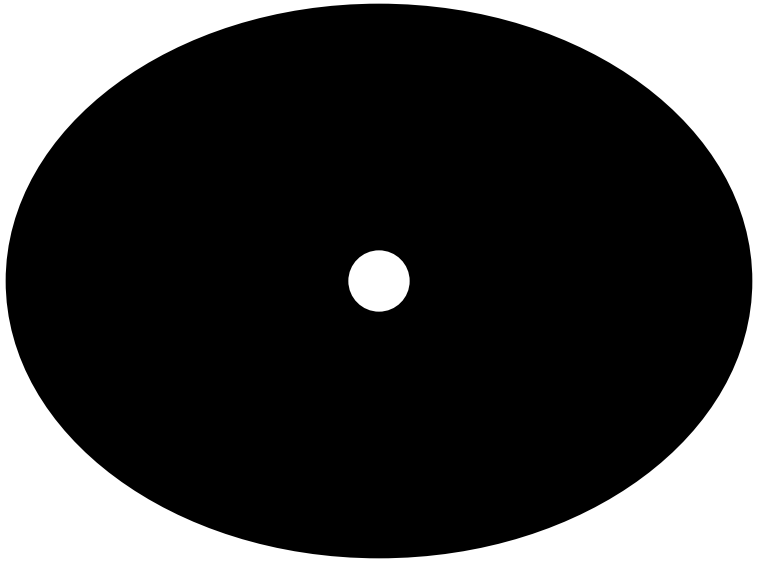
All Cluster#1 points are in the **DS**.

BFR Metadata: *Cluster summary with three sets*

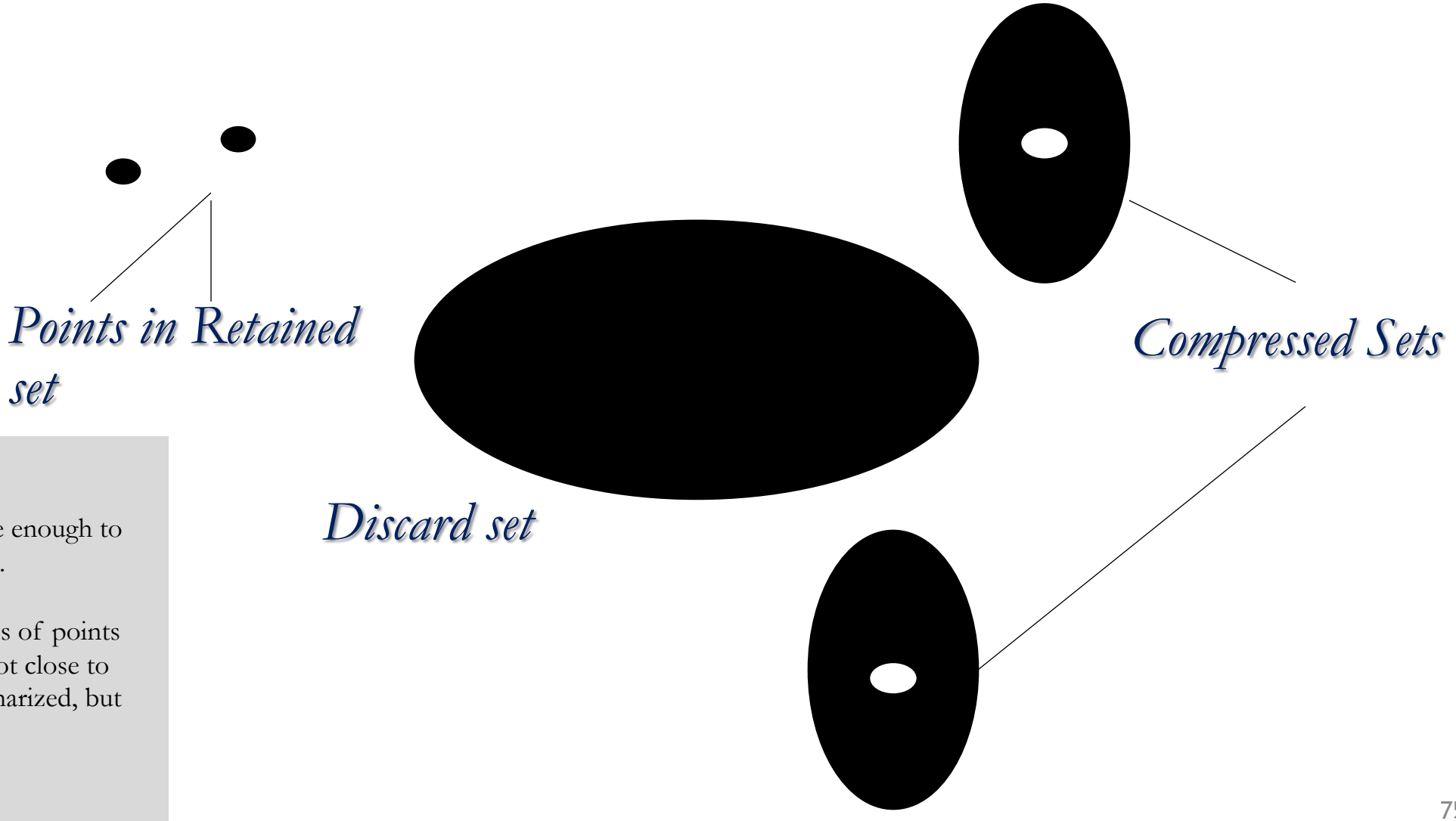
Metadata that is being kept in memory consists of:

For each cluster, the discard set (DS) is summarized by:

- *N, Vector SUM and SUMSQ*



BFR Metadata: *Cluster summary with three sets*



Overview:

The discard set : points close enough to a centroid to be summarized.

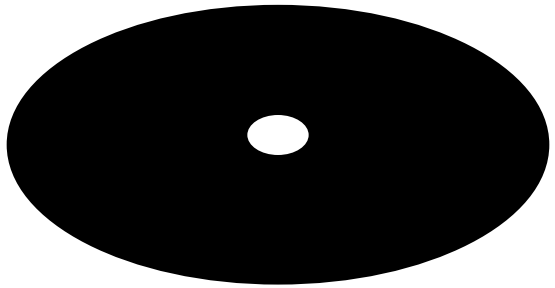
The compression set : groups of points that are close together but not close to any centroid. They are summarized, but not assigned to a cluster.

The retained set : isolated points.

BFR More Details on Cluster Summary

A cluster can be presented with 2d+1 (Centroid, Centroid and Variance)

- $2d + 1$ values represent any size cluster
  - d = number of dimensions
- Average in each dimension (the centroid) can be calculated as SUM_i / N
  - SUM_i = i^{th} component of SUM



- Variance of a cluster's discard set in dimension i is: $(SUMSQ_i / N) – (SUM_i / N)^2$
  - And standard deviation is the square root of that along dimension i.
- Next step: Actual clustering

Loading a Data Chunk into Memory

Processing the "Memory-Load" of points (1): (every time you load data in memory do:)

1) Find those points that are "sufficiently close" to a cluster centroid and add those points to that cluster and the DS

These points are so close to the centroid that
they can be summarized and then discarded

2) Use any main-memory clustering algorithm to cluster the remaining points and the old RS

Clusters go to the CS; outlying points to the RS

Loading a Data Chunk into Memory (cont'd)

3) DS set: Adjust statistics of the clusters to account for the new points

Add Ns, SUMs, SUMSQs

4) Consider merging compressed sets in the CS

5) If this is the last round, merge all compressed sets in the CS and all RS points into their nearest cluster

BFR's Important Questions

How do we decide if a point is "close enough" to a cluster that we will add the point to that cluster?

- Define a threshold with regards to how many standard deviations the point is far from the centroid using the Mahalanobis distance of a point to the closest centroid

Let s first review the Standard Deviation measure.

Primer on Standard Deviation

Standard Deviation (SD) is a statistical metric for measuring the amount of dispersion of a collection of data points. It measure variability of a population.

Low SD reflects that the data values are very close to the centroid (mean)

High SD reflects that the values are spread out (a big range of values)

Primer on Standard Deviation (Cont'd)

Example:

- Consider the following one dimensional data points 2,4, 4,4,5,5,7, 9.
- Calculate the mean, the mean is 5
- Calculate the deviations of each data point from the mean, and square the result of each

$$(2-5)^2 = (-3)^2 = 9 \qquad (5-5)^2 = 0^2 = 0$$
$$(4-5)^2 = (-1)^2 = 1 \qquad (5-5)^2 = 0^2 = 0$$
$$(4-5)^2 = (-1)^2 = 1 \qquad (7-5)^2 = 2^2 = 4$$
$$(4-5)^2 = (-1)^2 = 1 \qquad (9-5)^2 = 4^2 = 16.$$

Reference: Wikipedia

Primer on Standard Deviation (Cont'd)

Example:

- The variance is the mean of these values:

$$\frac{9+1+1+1+0+0+4+16}{8} = 4.$$

- The Standard Deviation is and the population standard deviation is equal to the square root of the variance which is 2 in this example.

Reference: Wikipedia

Back to BFR's Important Questions

How do we decide if a point is "close enough" to a cluster that we will add the point to that cluster?

- **O**ne possible solution is to define a threshold with regards to how many standard deviations the point is far from the centroid using the Mahalanobis distance of a point to the closest centroid.

Mahalanobis Distance

- Consider a point $X(x_1, \ldots, x_d)$ where d is the number of dimensions
- Consider the Centroid of the data to be $C(c_1, \ldots, c_d)$
- Normalize by Standard Deviation in each dimension as follows: $y_i = (x_i - c_i) / \sigma_i$
- σ_i is the Standard Deviation at dimension i
- The D be the mahalanobis distance from the centroid to point X

$$D(X, C) = \sqrt{\sum_{i=1}^{d} \left(\frac{X - C_i}{\sigma_i}\right)^2}$$

Distance being normalized in each dimension: $y_i = (x_i - c_i) / \sigma_I$

Take sum of the squares of the y_i *(the number of std. deviation xi is away from the centroid)*

Reference: J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Mahalanobis Distance

- σ_i is the Standard Deviation at dimension i
- Let d be the mahalanobis distance from the centroid to point X

$$D(X, C) = \sqrt{\sum_{i=1}^{d} \left(\frac{X - C_i}{\sigma_i}\right)^2}$$

- If X is one standard deviation from the mean then $X - C_i = \sigma_i$
- So $D(X, C)$ will be equal to $\sqrt{d}$ d is the number of dimensions

If clusters are normally distributed in d dimensions, then after transformation, one standard deviation = $\sqrt{d}$. I.e., 70% of the points of the cluster will have a Mahalanobis distance $< \sqrt{d}$.

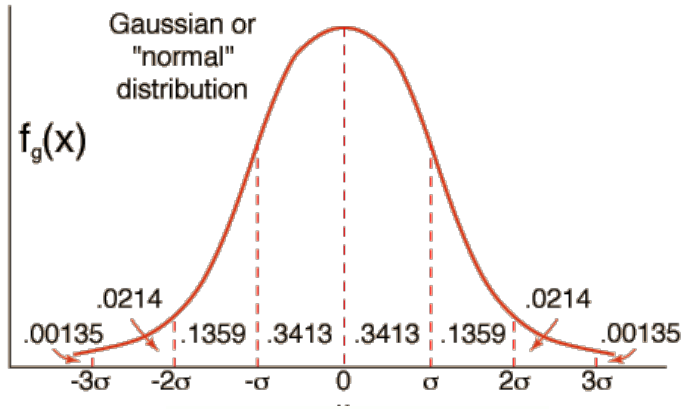
Accept a point for a cluster if its M.D. is < some threshold, e.g. 4 standard deviations.

Reference: J. Leskovec, A. Rajaraman, J. Ullman: Mining of Massive Datasets, http://www.mmds.org

Mahalanobis Distance Acceptance Criterion

- If clusters are normally distributed in $\boldsymbol{d}$ dimensions, then after transformation, one standard deviation $= \sqrt{\boldsymbol{d}}$ **(y = 1)**
  - i.e., 68% of the points of the cluster will have a Mahalanobis distance $< \sqrt{\boldsymbol{d}}$

- Accept a point for a cluster if its M.D. is **<** some threshold, e.g. **2 or 3** standard deviations



Another question: **H**ow do we decide whether two compressed sets (CS) deserve to be combined into one?

Should 2 CS clusters be combined?

Should Two CS subclusters be combined?

Compute the variance of the combined subcluster

N, ***SUM***, and ***SUMSQ*** allow us to make that calculation quickly
Combine if the combined variance is below some threshold

Many alternatives: Treat dimensions differently,
consider density (density based clustering)

The CURE Data Clustering Algorithm

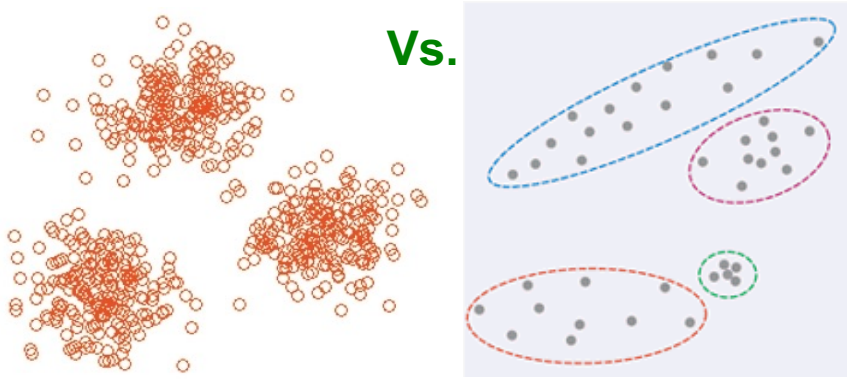
Clustering Using Representatives

Extension of *k-means* to clusters of arbitrary shapes

The CURE Algorithm

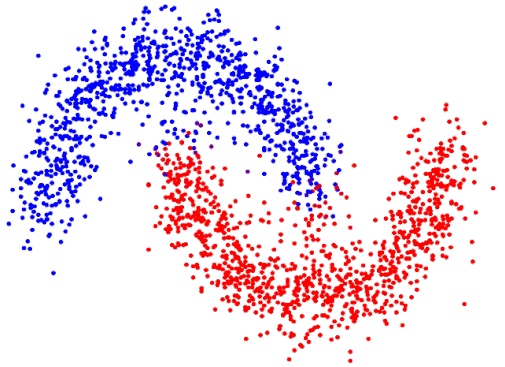
Problem with BFR/*k*-means:

- Assumes clusters are normally distributed in each dimension
- And axes are fixed – ellipses at an angle are *not OK*



CURE (Clustering Using Representatives):

- Assumes a Euclidean distance
- Allows clusters to assume any shape
- Uses a collection of representative points (no one representative) to represent clusters



Starting CURE

2 Pass algorithm. Pass 1:

0) Pick a random sample of points that fit in main memory

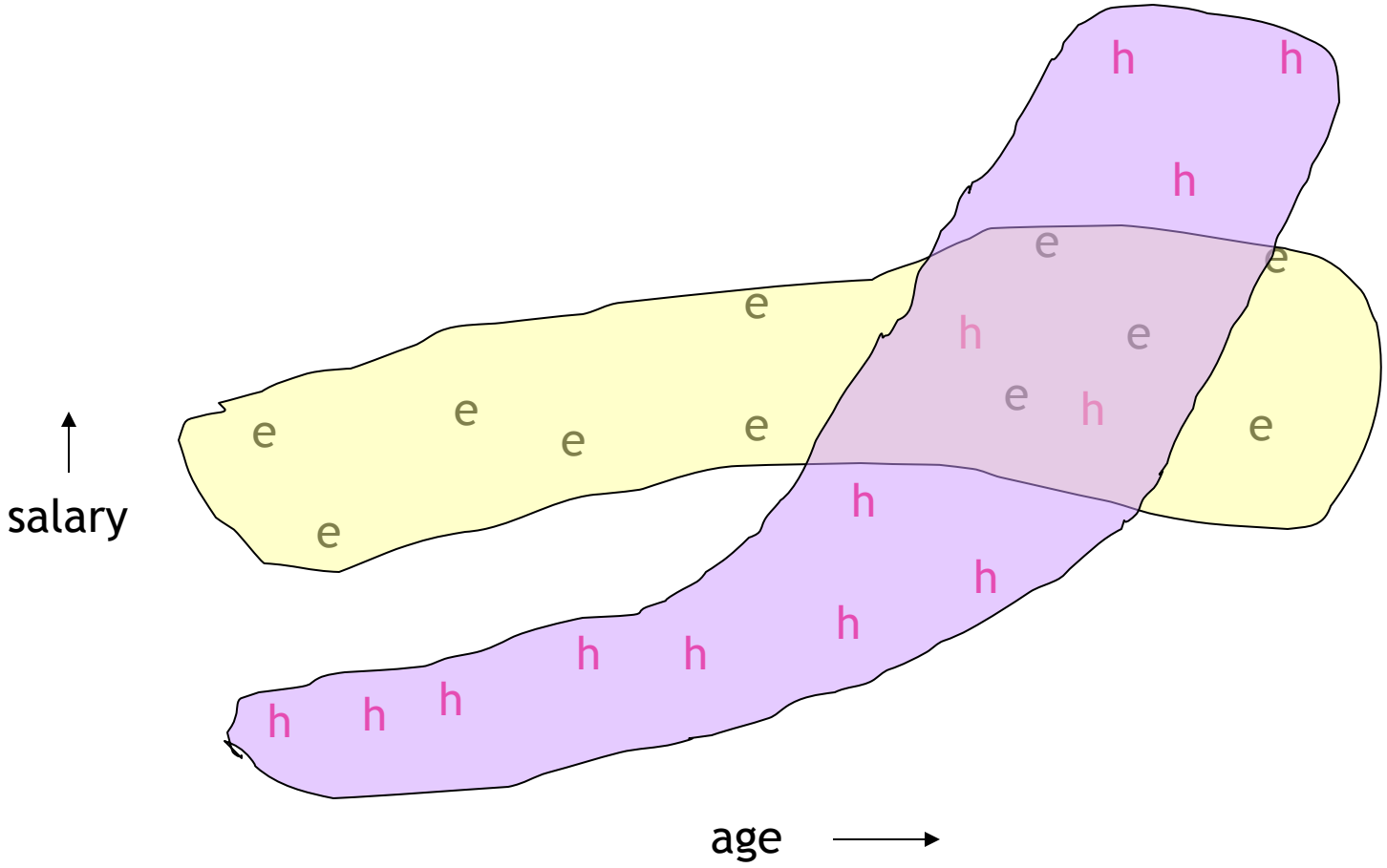
1) Initial clusters:

- Cluster these points hierarchically – group nearest points/clusters

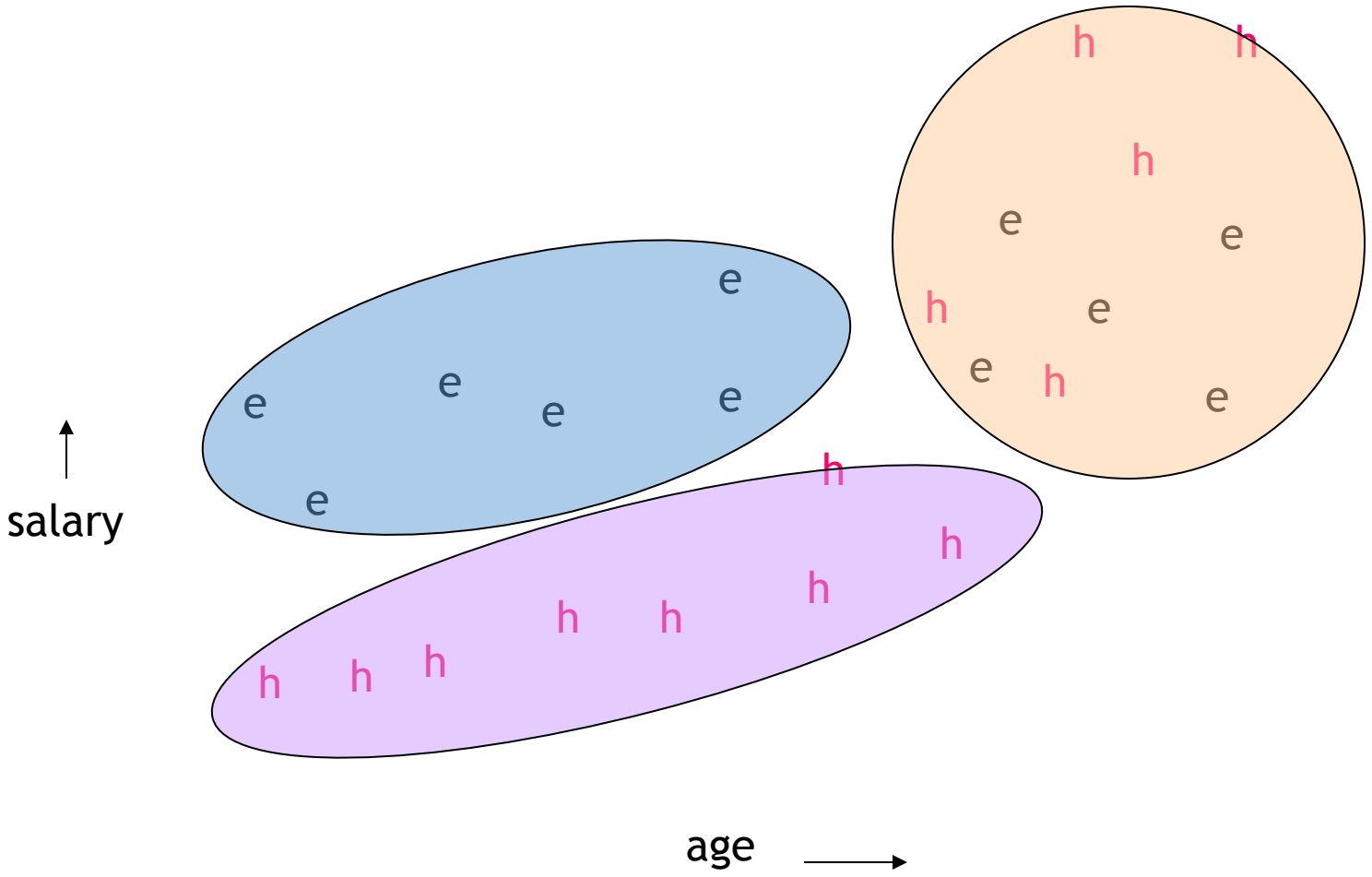
2) Pick representative points:

- For each cluster, pick a sample of points, as dispersed as possible
- From the sample, pick representatives by moving them (say) 20% toward the centroid of the cluster

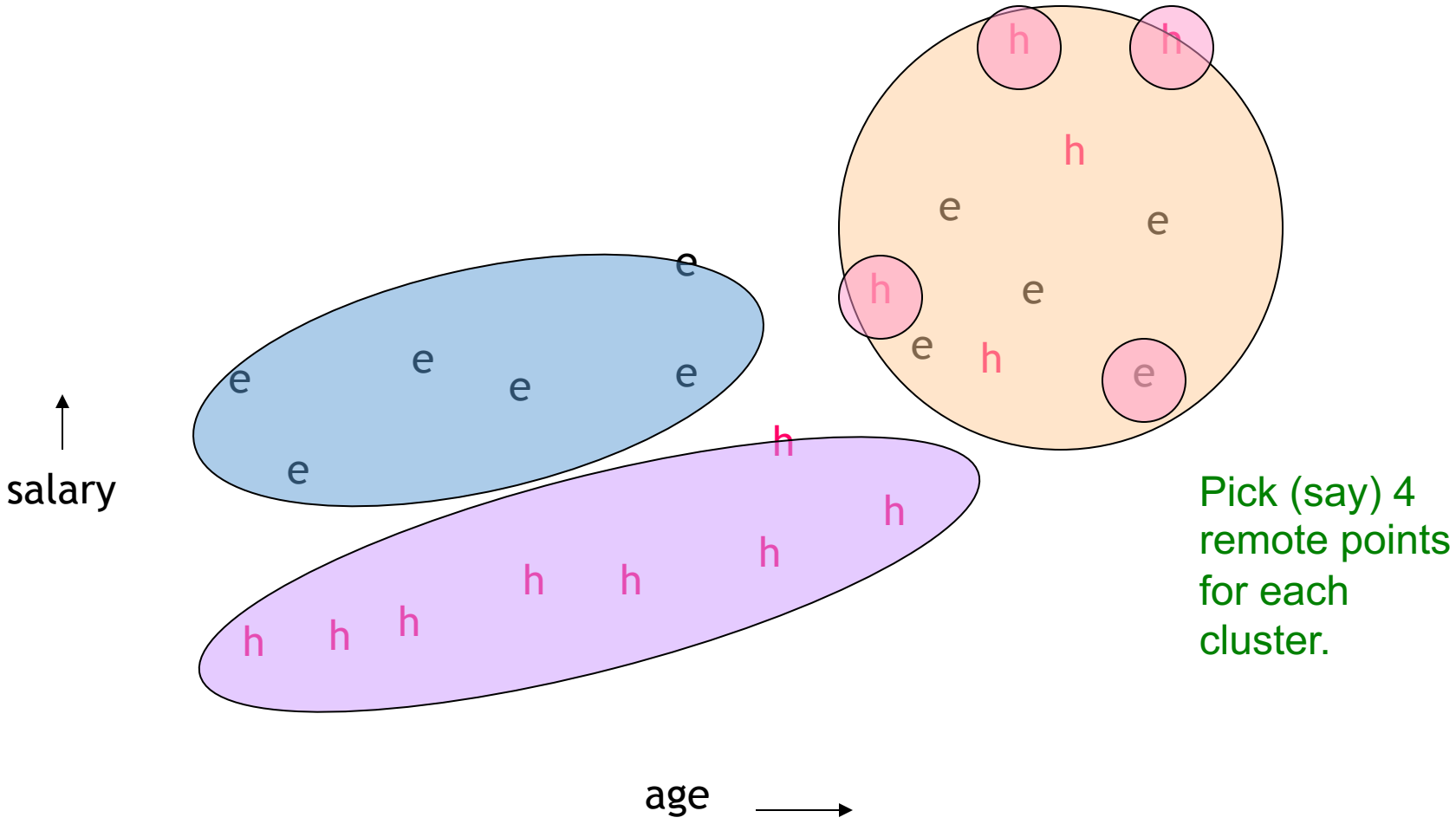
Example: Stanford Salaries



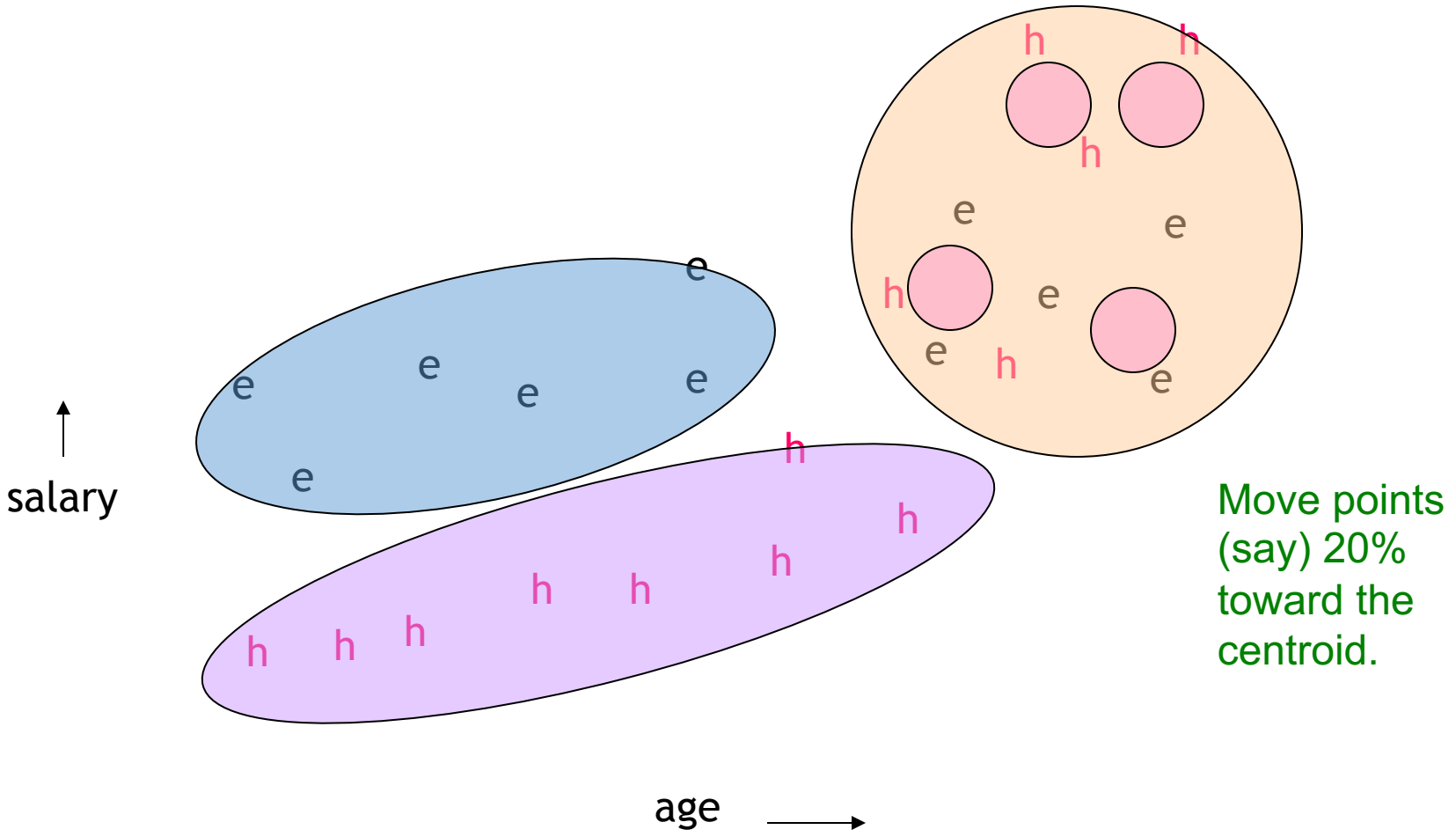
Example: Initial Clusters



Example: Pick Dispersed Points



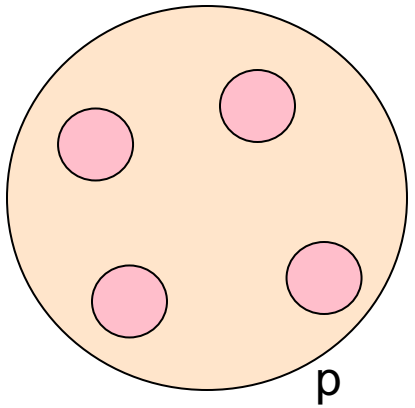
Example: Pick Dispersed Points



Finishing CURE

Pass 2

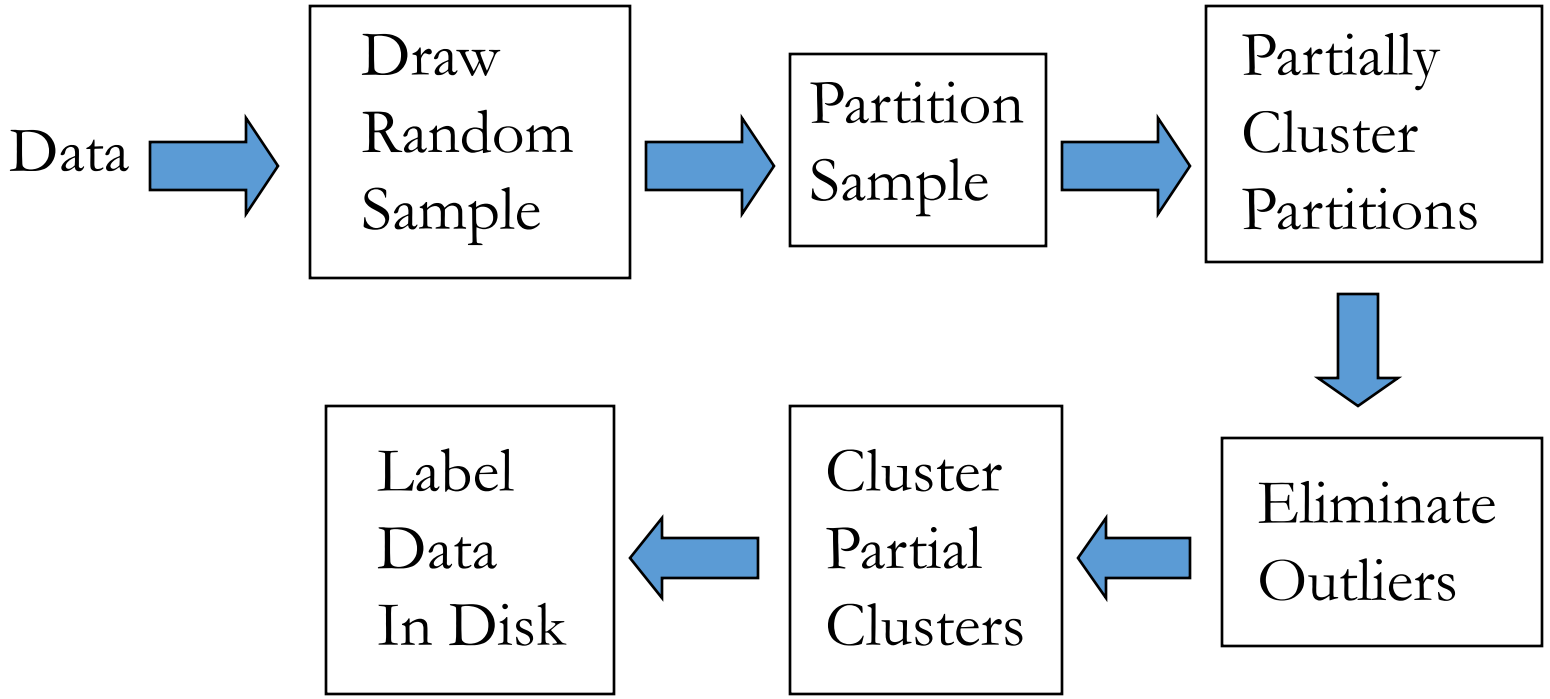
Now, rescan the whole dataset and visit each point ***p*** in the data set



Place it in the "closest cluster"

Normal definition of "closest":
Find the closest representative to ***p*** and assign it to representative's cluster

Six Steps in CURE Algorithm



Notes credits to Anna Putnam & Diane Xu, UCLA

Example

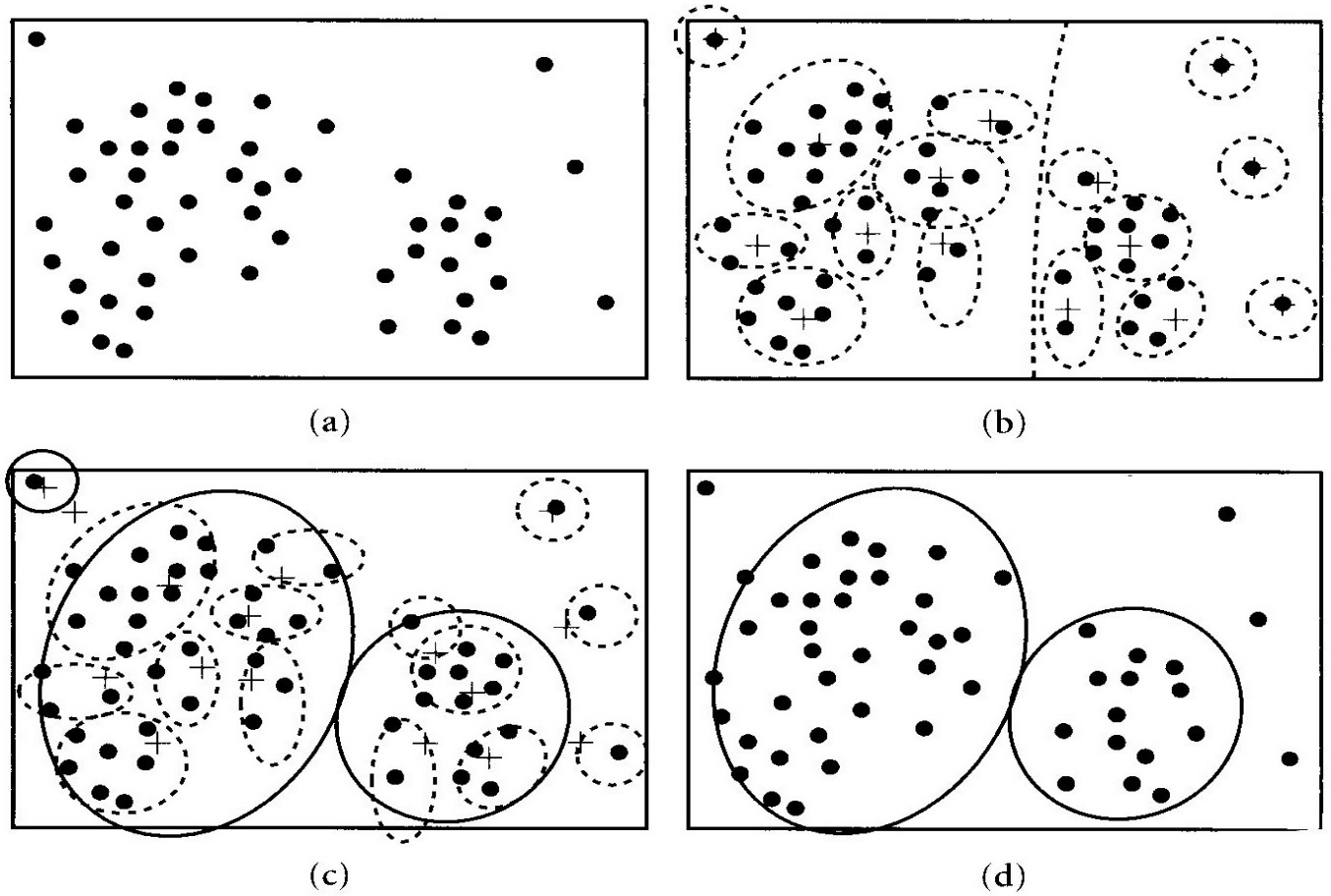


Figure 8.7 Clustering of a set of points (or objects) by CURE. (a) A random sample of objects. (b) The objects are partitioned and partially clustered. Representative points for each cluster are marked by a "+". (c) The partial clusters are further clustered. For each new cluster the representative points are "shrinked" or moved toward the cluster center. (d) The final clusters are of nonspherical shape.

CURE's Advantages

- Relatively accurate:
  - Adjusts well to geometry of non-spherical shapes.
  - Scales to large datasets
  - Less sensitive to outliers
- More efficient:
  - Space complexity: $O(n)$
  - Time complexity: $O(n^2 \log n)$ ($O(n^2)$ if dimensionality of data points is small)

Notes credits to Anna Putnam & Diane Xu, UCLA

Feature: Random Sampling

- Key idea: apply CURE to a random sample drawn from the data set rather than the entire data set.
- Advantages:
  - Smaller size
  - Filtering outliers
- Concerns: may miss out or incorrectly identify certain clusters!
  - Experimental results show that, with moderate sized random samples, we were able to obtain very good clusters.

Notes credits to Anna Putnam & Diane Xu, UCLA

Feature: Partitioning for Speedup

- Partition the sample space into p partitions, each of size n/p.
- Partially cluster each partition until the final number of clusters in each partition reduces to n/(pq). (q > 1)
- Collect all partitions and run a second clustering pass on the n/p partial clusters
- Tradeoff: sample size vs. accuracy

Notes credits to Anna Putnam & Diane Xu, UCLA

Feature: Labeling Data on Disk

- Input is a randomly selected sample.
- Have to assign the appropriate cluster labels to the remaining data points
- Each data point is assigned to the cluster containing the representative point closest to it
- Advantage: using multiple points enables CURE to correctly to distribute the data points when clusters are non-spherical or non-union

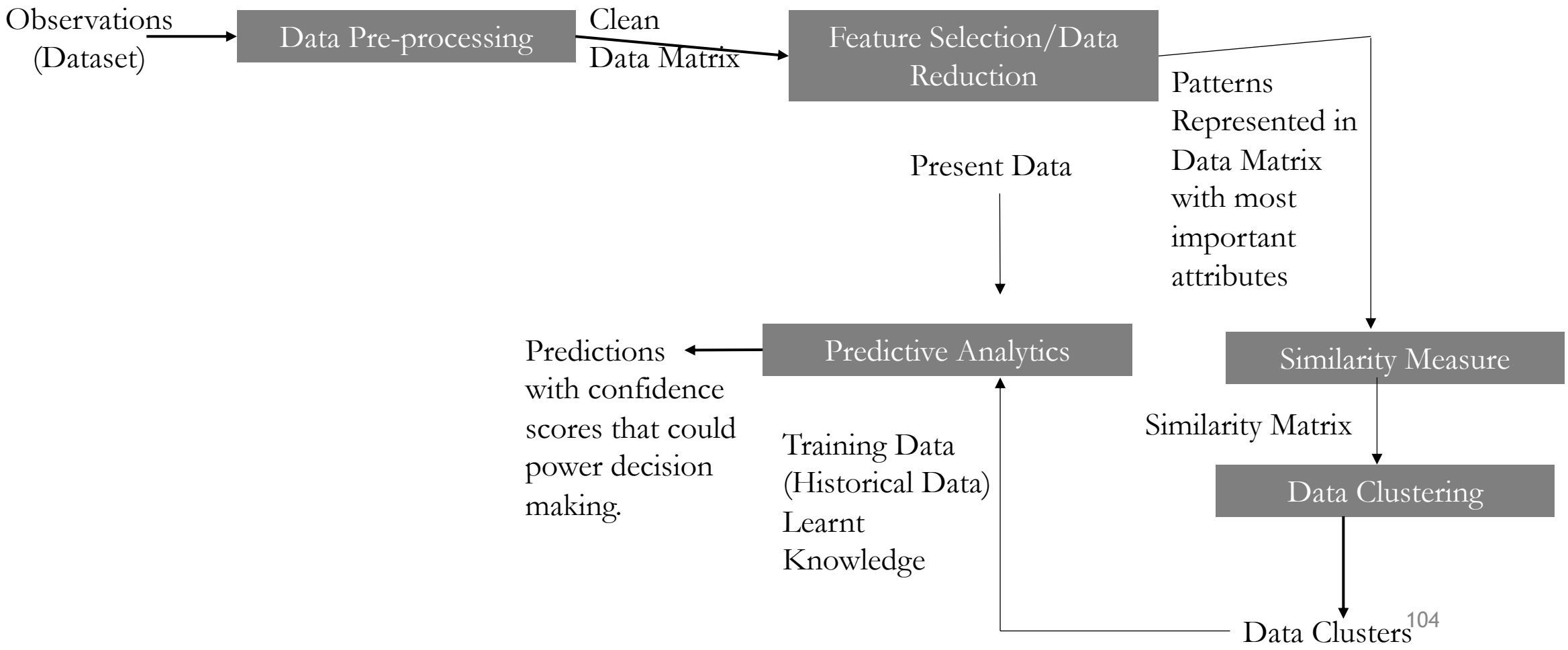
Notes credits to Anna Putnam & Diane Xu, UCLA

Feature: Outliers Handling

- Random sampling filters out a majority of the outliers.
- The remaining few outliers in the random sample are distributed all over the sample space and gets further isolated.
- The clusters which are growing very slowly are identified and eliminated as outliers.
- Use a second level pruning to eliminate merging-together outliers: outliers form very small clusters.

Notes credits to Anna Putnam & Diane Xu, UCLA

Summary: *Data Clustering and its Relationship with Predictive Analytics*



Summary

- Clustering: *Given a set of points, with a notion of distance between points, group the points into some number of clusters*
- Algorithms to remember:
  - K-means
  - DBSCAN
  - BFR
  - Hierarchical Clustering
  - Flock by Leader (Bio-Inspired Algorithm)
  - CURE

References

- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. SIGMOD'98
- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- M. R. Anderberg. Cluster Analysis for Applications. Academic Press, 1973.
- M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure, SIGMOD'99.
- P. Arabie, L. J. Hubert, and G. De Soete. Clustering and Classification. World Scietific, 1996
- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. KDD'96.
- M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. SSD'95.
- D. Fisher. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2:139-172, 1987.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. In Proc. VLDB'98.
- S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. SIGMOD'98.
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Printice Hall, 1988.

References

- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.
- E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. VLDB'98.
- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- P. Michaud. Clustering techniques. Future Generation Computer systems, 13, 1997.
- R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. VLDB'94.
- E. Schikuta. Grid clustering: An efficient hierarchical clustering method for very large data sets. Proc. 1996 Int. Conf. on Pattern Recognition, 101-105.
- G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.
- W. Wang, Yang, R. Muntz, STING: A Statistical Information grid Approach to Spatial Data Mining, VLDB'97.
- T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH : an efficient data clustering method for very large databases. SIGMOD'96.