Clustering

1. Objectives

- Techniques to group data into related classify datasets and provide categorical labels, e.g., sports, technology, kid, etc.
- Detection of patterns
- Models to predict certain future behaviors.

2. Clustering

2.1. Definitions

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Grouping a set of data objects into clusters
- Clustering is unsupervised classification: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

2.2. General Applications

- Text mining:
  - Document categorization
  - Detection of topics
  - Summarization
- Text Mining:
  - Web log analysis
  - Detection of groups of similar access patterns

- Bio-informatics:
    - Gene expression data: detection of cancer genes
- Others:
    - Image processing
    - Market analysis
    - Etc.

2.3. What is a good clustering?

- A good clustering method will produce high quality clusters with
    - **High intra-class** similarity
    - **Low inter-class** similarity
- The quality of a clustering result depends on both the **similarity measure** used by the method and its implementation.
- The quality of a clustering method is also measured by its ability **to discover** some or all of the hidden patterns.

2.4. Requirements

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

3. Data Structures

- Data Matrix (two modes)

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

- Dissimilarity (or similarity) matrix

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

4. Similarity Measures

- Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, which is typically metric: $d(i, j)$
- There is a separate "quality" function that measures the "goodness" of a cluster.
- The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, ordinal and ratio variables.

- Weights should be associated with different variables based on applications and data semantics.
- It is hard to define "similar enough" or "good enough"
  - The answer is typically highly subjective.
- Type of data in clustering analysis
  - Interval-scaled variables
  - Binary variables
  - Nominal, ordinal, and ratio variables
  - Variables of mixed types

4.1. Standardize data

- Calculate the ***mean absolute deviation***:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + ... + |x_{nf} - m_f|)$$

Where

$$m_f = \frac{1}{n}(x_{1f} + x_{2f} + ... + x_{nf}).$$

- ***z-score:*** Calculate the standardized measurement

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is more robust than using standard deviation

- Computation of data similarity
  - <u>Distances</u> are normally used to measure the <u>similarity</u> or <u>dissimilarity</u> between two data objects
  - Some popular ones include: *Minkowski distance*:

$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + ... + |x_{i_p} - x_{j_p}|^q)}$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two *p*-dimensional data objects, and *q* is a positive integer.

  - If *q = 1*, *d* is Manhattan distance

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + ... + |x_{i_p} - x_{j_p}|$$

  - *If q = 2, d* is Euclidean distance:

$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + ... + |x_{i_p} - x_{j_p}|^2)}$$

  - Properties:
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$

- Also, one can use weighted distance, parametric Pearson product moment correlation, or other disimilarity measures

4.2. Binary variables

- A contingency table for binary data

	1	0	*sum*
1	a	b	$a+b$
0	c	d	$c+d$
sum	$a+c$	$b+d$	p

- Simple matching coefficient (invariant, if the binary variable is *symmetric*):

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- Jaccard coefficient (noninvariant if the binary variable is *asymmetric*):

$$d(i,j) = \frac{b+c}{a+b+c}$$

- Example:

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- gender is a symmetric attribute
- The remaining attributes are asymmetric binary
- Let the values Y and P be set to 1, and the value N be set to 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$
$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$
$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

4.3. Nominal Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

$$d(i,j) = \frac{p-m}{p}$$

- Method 2: use a large number of binary variables
  - Creating a new binary variable for each of the *M* nominal states

4.4. Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - Replace x_{if} by their rank:

$$r_{if} \in \{1, \ldots, M_f\}$$

  - Map the range of each variable onto [0, 1] by replacing *i*-th object in the *f*-th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

- Compute the dissimilarity using methods for interval-scaled variables

4.5. Ratio-scaled variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale, such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - Treat them like interval-scaled variables—not a good choice! (why?—the scale can be distorted)
  - Apply logarithmic transformation: $y_{if} = log(x_{if})$
  - Treat them as continuous ordinal data treat their rank as interval-scaled

4.6. Variables of mixed types

- A database may contain all the six types of variables
  - Symmetric binary, asymmetric binary, nominal, ordinal, interval and ratio
- One may use a weighted formula to combine their effects:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}}$$

  - f is binary or nominal:
    $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, or $d_{ij}^{(f)} = 1$ o.w.
  - f is interval-based: use the normalized distance
  - f is ordinal or ratio-scaled
    - compute ranks r_{if} and
    - treat z_{if} as interval-scaled

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

5. Clustering approaches

5.1. Major approaches

- Partitioning algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Density-based: based on connectivity and density functions
- Grid-based: based on a multiple-level granularity structure
- Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other

5.2. Partitioning approach

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters
- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
    - Global optimal: exhaustively enumerate all partitions
    - Heuristic methods: *k-means* and *k-medoids* algorithms
    - *k-means* (MacQueen'67): Each cluster is represented by the center of the cluster
    - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

6. The K-means clustering method

- Input: n objects (or points) and a number k

- Algorithm 1:
  - Step 1: Randomly place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
  - Step 2: Assign each object to the group that has the closest centroid.
  - Step 3: When all objects have been assigned, recalculate the positions of the K centroids.
  - Repeat Steps 2 and 3 until the stopping criteria is met.

- Algorithm 2:
  - Step 1: Partition objects into *k* nonempty subsets
  - Step 2: Compute seed points as the centroids of the clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
  - Step 3: Assign each object to the cluster with the nearest seed point
    - Go back to Step 2, stop when no more new assignment

  - Example

- Stopping criteria:
  - No change in the members of all clusters
  - when the squared error is less than some small threshold value α:
    - Squared error se

$$se = \sum_{i=1}^{k} \sum_{p \in c_i} \| p - m_i \|^2$$

where mi is the mean of all instances in cluster ci

    - se(j) < α

- Properties of k-means
  - Guaranteed to converge
  - Guaranteed to achieve local optimal, not necessarily global optimal. Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*

- Analysis
  - Strength: *Relatively efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$

  - Weakness
    - Applicable only when *mean* is defined, then what about categorical data?
    - Need to specify *k,* the *number* of clusters, in advance
    - Unable to handle noisy data and *outliers*
    - Not suitable to discover clusters with *non-convex shapes*

- Variations of K-means method:

    - A few variants of the *k-means* which differ in
        - Selection of the initial *k* means
        - Dissimilarity calculations
        - Strategies to calculate cluster means
    - Handling categorical data: *k-modes* (Huang'98)
        - Replacing means of clusters with modes
        - Using new dissimilarity measures to deal with categorical objects
        - Using a frequency-based method to update modes of clusters
        - A mixture of categorical and numerical data: *k-prototype* method

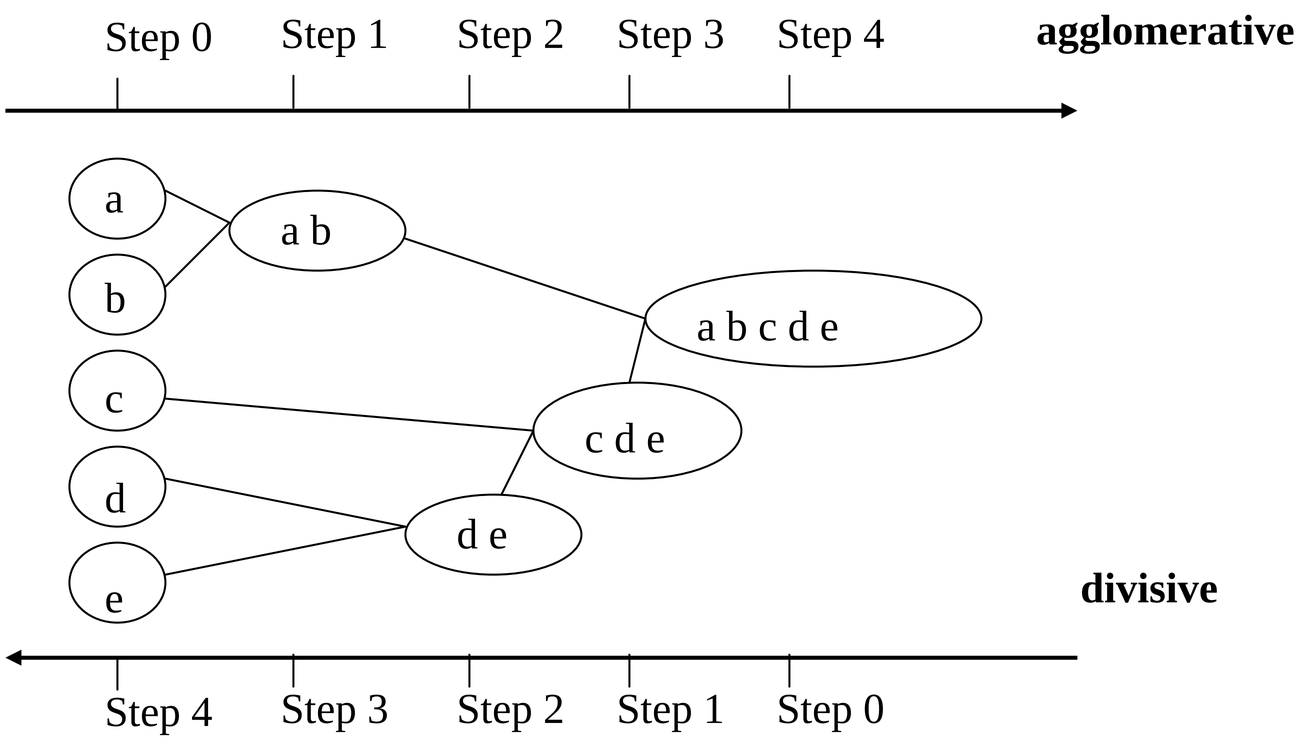
- Drawbacks of k-mean method
    - The k-means algorithm is sensitive to outliers !
        - Since an object with an extremely large value may substantially distort the distribution of the data.
    - K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.

7. The K-medoids Clustering Method

- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
    - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
    - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)

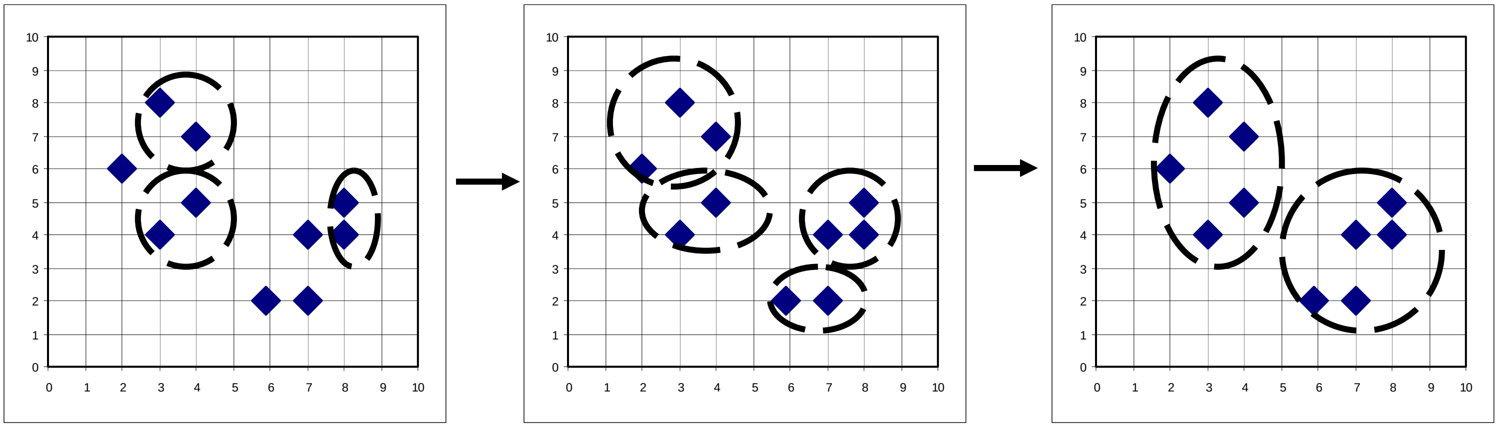
8. Hierarchal Clustering

- Use distance matrix as clustering criteria.
- This method does not require the number of clusters ***k*** as an input, but needs a termination condition

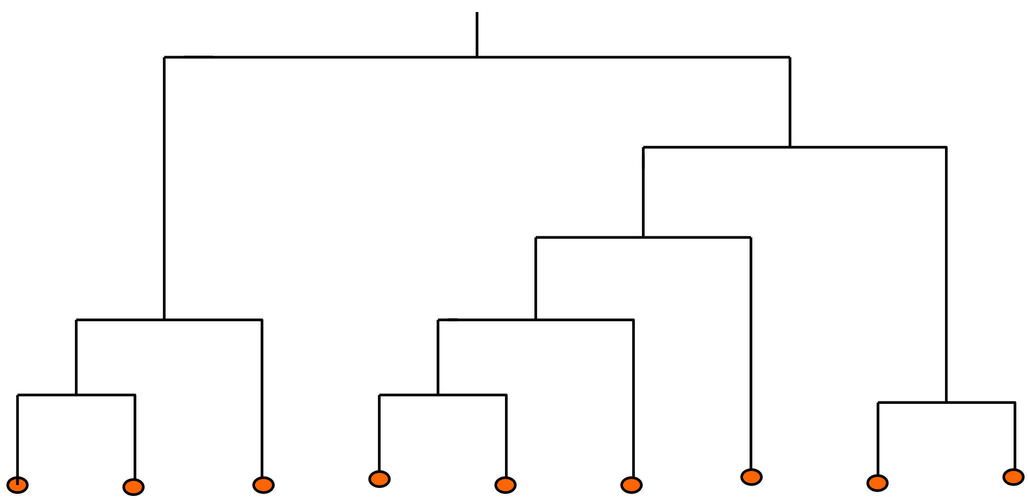


8.1. AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Use the Single-Link method and the dissimilarity matrix.
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster

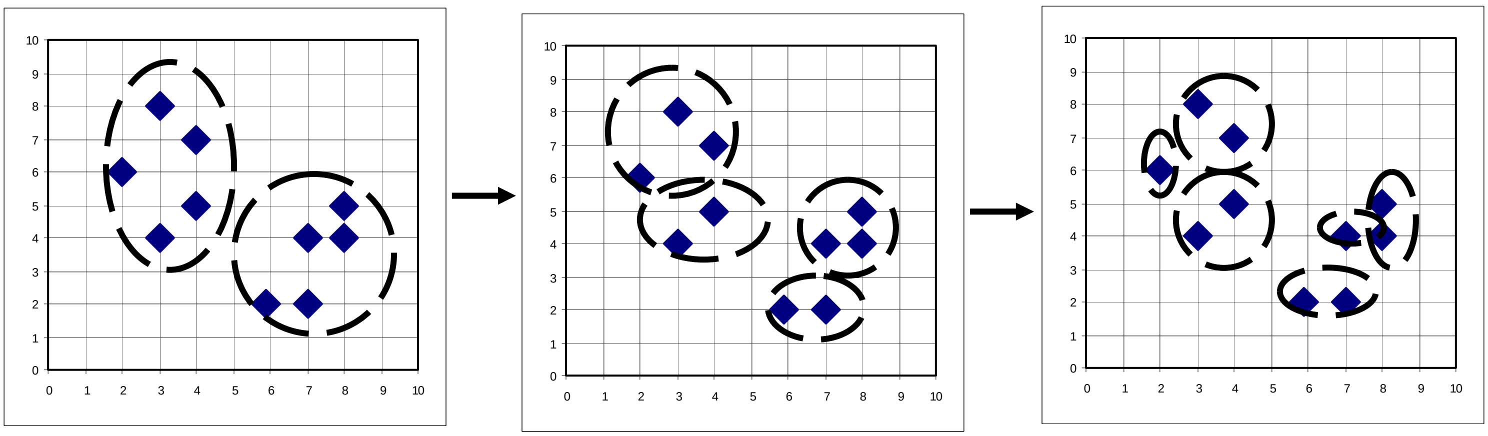


- **A *Dendrogram*** Shows How the Clusters are Merged Hierarchically
    - o Decompose data objects into a several levels of nested partitioning (<u>tree</u> of clusters), called a <u>dendrogram</u>.
    - o A <u>clustering</u> of the data objects is obtained by <u>cutting</u> the dendrogram at the desired level, then each <u>connected component</u> forms a cluster.



8.2. Divisive Analysis: DIANA

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



8.3. Analysis of hierarchical clustering

- Major weakness of agglomerative clustering methods
  - o <u>do not scale</u> well: time complexity of at least $O(n^2)$, where n is the number of total objects

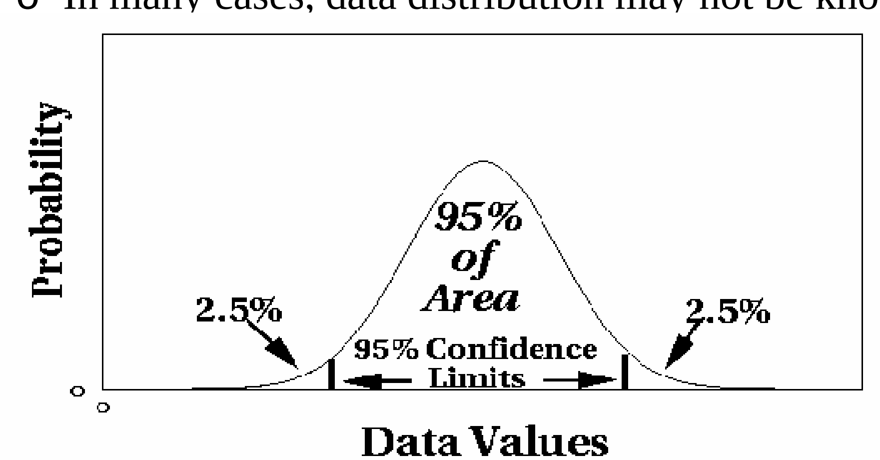
- Integration of hierarchical with distance-based clustering
  - o <u>BIRCH (1996)</u>: uses CF-tree and incrementally adjusts the quality of sub-clusters
  - o <u>CURE (1998</u>): selects well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction
  - o <u>CHAMELEON (1999)</u>: hierarchical clustering using dynamic modeling.

9. Outliers

- What are outliers?
  - The set of objects are considerably dissimilar from the remainder of the data
  - Example: Sports: Michael Jordon, Wayne Gretzky, ...
- Problem
  - Find top n outlier points
- Applications:
  - Credit card fraud detection
  - Telecom fraud detection
  - Customer segmentation
  - Medical analysis

9.1. Statistical Approach

- Assume a model underlying distribution that generates data set (e.g. normal distribution)
- Use discordancy tests depending on
  - Data distribution
  - Distribution parameter (e.g., mean, variance)
  - Number of expected outliers
- Drawbacks
  - Most tests are for single attribute
  - In many cases, data distribution may not be known



9.2. Distance-Based Approach

- Introduced to counter the main limitations imposed by statistical methods
    - We need multi-dimensional analysis without knowing data distribution.
- Distance-based outlier: A Outlier(p, D)-outlier is an object O in a dataset T such that at least a fraction p of the objects in T lies at a distance greater than D from O
- Algorithms for mining distance-based outliers
    - Index-based algorithm:
        - Use R-tree indexing structure.
        - It takes $O(k*n^2)$ without the cost of building the tree.
    - Nested-loop algorithm:
        - Divide the dataset into blocks and look for outliers in block by block.
        - It has the same complexity as index-based algorithm.
    - Cell-based algorithm:
        - Divide the data space into cells and look for outliers cell-by-cell rather than point-by-point.
        - It takes $O(n^2)$.