VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

# Machine Learning

## Clustering

Jan Platoš

November 22, 2023

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava

# Clustering

# Clustering

Given a set of data points, partition them into groups containing very similar points.

- Possible applications:
  - Data summarization
  - Customer Segmentation
  - Social network analysis
  - Preprocessing data for other algorithms (classification, outliers detection, etc.)

# Feature Selection for Clustering

- How to select only the features that are important?
- How to measure the ability of the feature to cluster objects?
- Two main approaches exists:
  - Filter models
  - Wrapper models

# Feature Selection for Clustering - Filter models

## Filter models

- Definition of measures that may evaluate the quality of a feature or a feature combination

## Term Strength

- Suitable for Text data or other sparse documents.
- A conditional probability that a selected term appear in a document $Y$ when it appear in a document $X$.
- The documents pairs are randomly sampled from similar documents.

$$Term\ strength = P(t \in Y | t \in X)$$

# Feature Selection for Clustering - Filter models

## Predictive Attribute Dependence

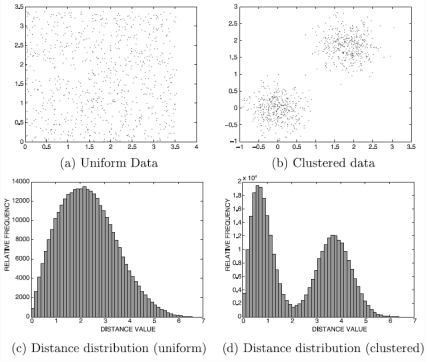
- Correlated features should end in better results than uncorrelated.
- Set of features should predict the value of another correlated feature.
- A regression or classification algorithm is used as a predictor.

## Predictive Attribute Dependence - Principle

- For each feature:
  - Use other features to predict the value of the selected feature.
  - Compute the accuracy of prediction and use it as relevance measure.
- All relevant features are used for clustering.

# Feature Selection for Clustering - Filter models

## Entropy




- Highly clustered data reflects some characteristics in the underlying distance distribution.
- The goal is to quantify the shape of the distance distribution on a given subset of features and to pick the most suitable subset.
- A systematic way to search possible combination of features.

# Feature Selection for Clustering - Filter models

## Entropy

- For $k$-features a k-dimensional space is defined.
- Define $m$ multidimensional grid regions over this space.
- The $p_i$ is a fraction of points in a region $i$.

$$E = \sum_{i=1}^{m} p_i \log(p_i)$$

- Distribution with poor clustering behavior results in high entropy.
- Alternatively an entropy over a distance distribution may be computed.

## Hopkins Statistics

- Evaluates the clustering tendency of the whole datasets/selected features (greedy approach may be used).
- $D$ is a dataset, $R$ is a random sample of $r$ points from $D$, sample $S$ of $r$ randomly generated points.
- The $a_1, a_2, \ldots, a_r$ are distances from points from $R$ to the nearest neighbors from $D$ and $b_1, \ldots, b_r$ are distances from points from $S$ to the nearest neighbors from $D$.

$$H = \frac{\sum_{i=1}^{r} b_i}{\sum_{i=1}^{r} (a_i + b_i)}$$

- $H$ is in the range $(0, 1)$.
- Uniformly distributed data will have $H = 0.5$, $H$ close to 1 means clustered data.

# Feature Selection for Clustering - Wrapper models

## Wrapper models

- Uses a cluster validity criterion for feature subset evaluation.
- The are highly imperfect.
- Search space of features is exponentially related to the dimensionality.
- The principle is sensitive to the choice of the validity criterion.
- Simpler methodology:
  - Cluster points according the selected feature subset.
  - Assign labels to the points according to the cluster the points belongs to.
  - Use supervised criterion to measure the quality of each feature.

# Representative-based Algorithms

- Simplest clustering algorithm.
- Based on the distance/similarity measure between points.
- Clusters are created in a single step.
- Hierarchical relationships do not exist among different clusters.
- The representatives are computed or selected from the cluster.
- A distance function is used for defined representatives to assign points into closest representative/cluster.

# Representative-based Algorithms

- A number of clusters k is usually defined by the user.
- A dataset $D$ contains $n$ data points $X_1, \ldots, X_n$.
- The goal is to determine $k$ representatives $Y_1, \ldots, Y_k$ that minimizes function $O$:

$$O = \sum_{i=1}^{n} \left[ min_j Dist(X_i, Y_i) \right]$$

- i.e. the sum of the distances of the different data points to their closest representatives needs to be minimized.

# Representative-based Algorithms

- The position of representatives and points assignment is not known a priori.
- An Iterative approach is used to solve the problem of representative/point assignment.
- General approach for representative-based algorithms:
  1. Initialize k representatives (random sampling or other method)
  2. Assign each data point to its closest representative using distance function $Dist(\cdot, \cdot)$
  3. Form a clusters from points assigned to each of the representative.
  4. Determine the optimal representative for each cluster $C_j$ using minimization of a local objective function $\sum_{X_i \in C_j} \left[ Dist(X_i, Y_j) \right]$

# Representative-based Algorithms

**Algorithm 1:** GenericRepresentative(Database: $D$, Number of Rpresentatives: $k$)

1 **begin**
2  Initialize representative set $S$;
3  **repeat**
4  Create clusters $(C_1, \ldots, C_k)$ by assigning each point in $D$ to closest representative in $S$ using the distance function $Dist(\cdot, \cdot)$;
5  Recreate set $S$ by determining one representative $Y_j$ for each $C_j$ that minimizes $\sum_{X_i \in C_j} Dist(X_i, Y_j)$;
6  **until** *convergence*;
7  **return** $(C_1, \ldots, C_k)$
8 **end**

## Definition of the distance function

- The distance function $Dist(X, Y)$ defines the behavior of the algorithm.
- The general definition is a $L_p$-norm.

$$Dist(X, Y) = \|X - Y\|_p^p = \left( \sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/p}$$

- Another possibility is a cosine measure.

$$\cos(X, Y) = \frac{X \cdot Y}{\|X\| \cdot \|Y\|} = \frac{\sum_{i=1}^{d} (x_i \cdot y_i)}{\sqrt{\sum_{i=1}^{d} x_i^2} \cdot \sqrt{\sum_{i=1}^{d} y_i^2}}$$

## Definition of the distance function

- Mahalanobis distance is a measure that takes into account a statistical distribution along each dimension

$$Maha(X, Y) = \sqrt{(X - Y)\Sigma^{-1}(X - Y)^T}$$

- Where $\Sigma$ is a covariance matrix where $\Sigma_{ij}$ is a variance between $i$-th and $j$-th dimension.

$$\Sigma = \begin{bmatrix} E\left[(X_1 - \mu_1)(X_1 - \mu_1)\right] & E\left[(X_1 - \mu_1)(X_2 - \mu_2)\right] & \cdots & E\left[(X_1 - \mu_1)(X_d - \mu_d)\right] \\ \vdots & & \ddots & \vdots \\ E\left[(X_d - \mu_d)(X_1 - \mu_1)\right] & E\left[(X_d - \mu_d)(X_2 - \mu_2)\right] & \cdots & E\left[(X_d - \mu_d)(X_d - \mu_d)\right] \end{bmatrix}$$

- $\mu_i$ is expected value of the dimension $i$.

## K-medians algorithm

- Uses an Manhattan distance for measuring distance between data points and representatives.

$$Dist(X, Y) = \|X - Y\|_1^1 = \sum_{i=1}^{d} |x_i - y_i|$$

- The optimal representatives for each cluster is a **median along each dimension in a cluster**.
- The $k$-medians algorithm is more robust than $k$-means: median is not as sensitive to outliers as the mean.

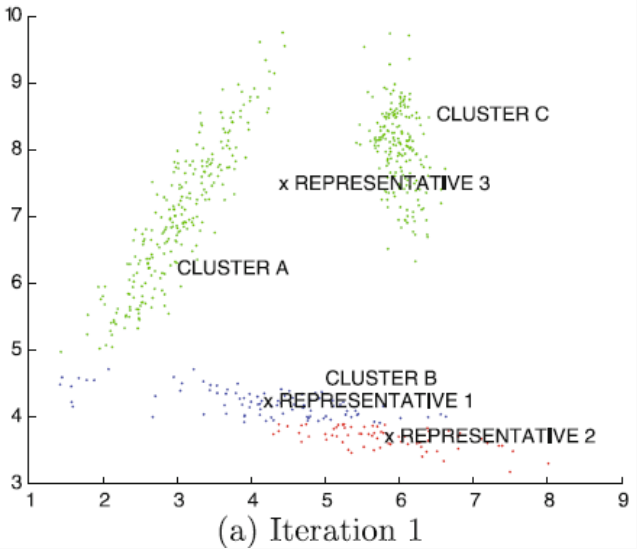
## K-means algorithm

- Uses an Euclidean distance for measuring distance between data points and representatives.

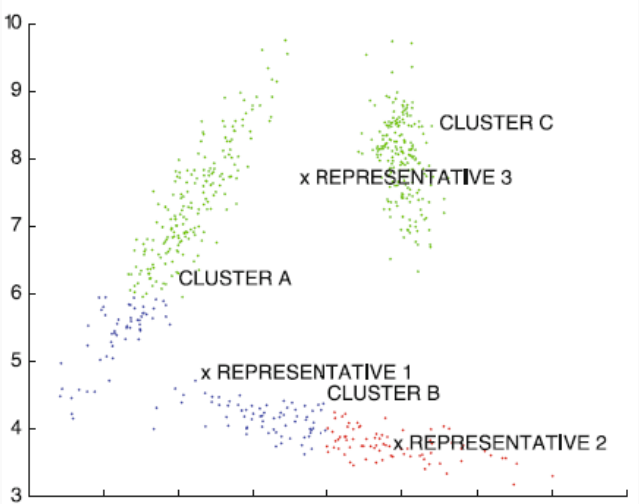
$$Dist(X, Y) = \|X - Y\|_2^2 = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

- The optimal representatives for each cluster is a mean.
- The $k$-means does not work well when the clusters are not spherical.
- A kernel variant of this algorithm is possible.

# Representative-based Algorithms - K-means Algorithm

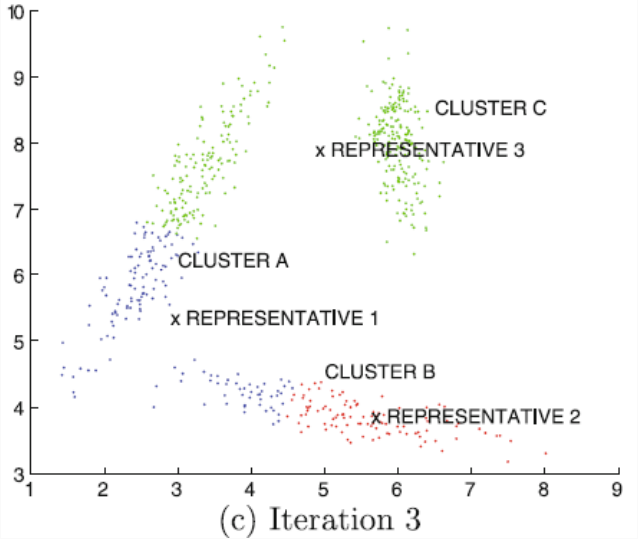


# Representative-based Algorithms - K-means Algorithm



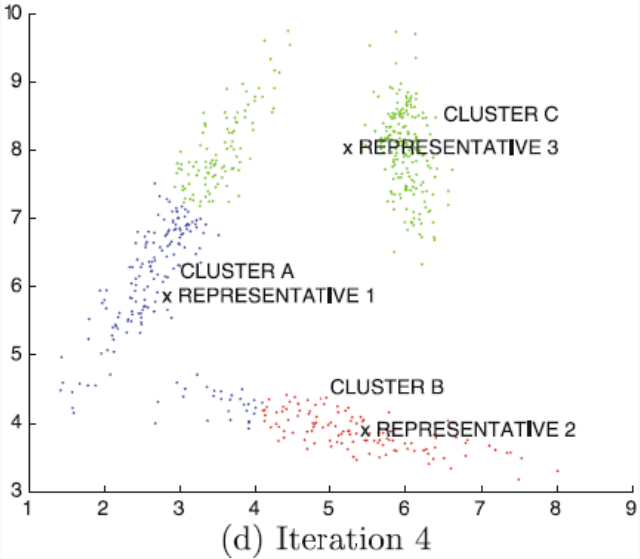
(b) Iteration 2

# Representative-based Algorithms - K-means Algorithm



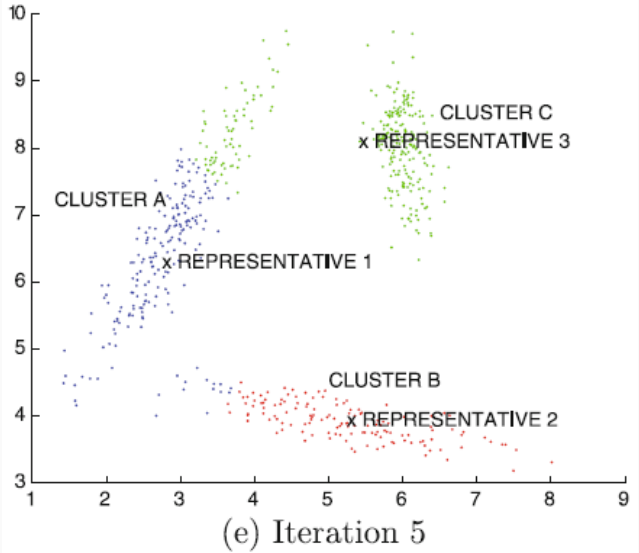
(c) Iteration 3

# Representative-based Algorithms - K-means Algorithm



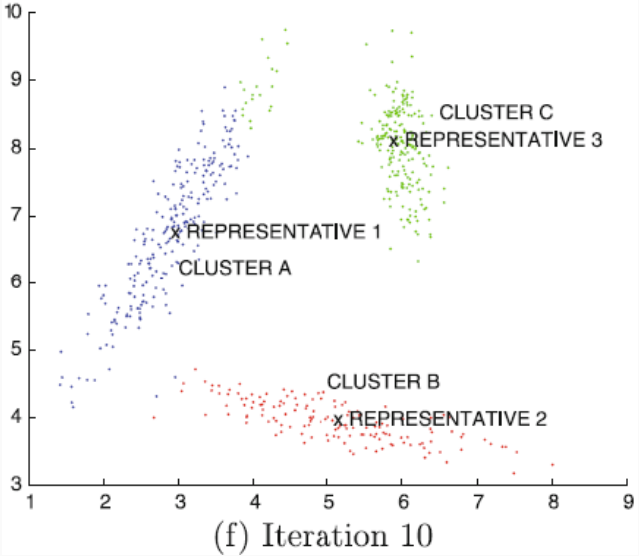
(d) Iteration 4

# Representative-based Algorithms - K-means Algorithm



(e) Iteration 5

# Representative-based Algorithms - K-means Algorithm



(f) Iteration 10

## K-medoids algorithm

- The representatives are always selected from the dataset.
- Why:
  - For means: the representatives may lie outside the cluster in an empty region due to presence of outliers.
  - Sometimes it is difficult to compute the optimal central representative of a set of data points of a complex data type, e.g. set of time series of varying lengths.
- The *k*-medoid approach may be defined for any data type as long as a distance/similarity function is known.

## K-medoids algorithm

- A set of representatives $Y$ is randomly selected from dataset.
- Iterative improvement is performed on the set of representatives:
  - Exhaustive search (extremely expensive).
  - A set of pairs $(X, Y)$ is randomly generated, $X$ from dataset and $Y$ from representatives, and the best pair is used for exchange.
- The $k$-medoids is slower than $k$-means.
- May be scalable implemented.

# Representative-based Algorithms - Implementation Issues

## Selection of $k$

- Very difficult automatically.
- Usually a large value is chosen.
- A post processing step may reduce number of clusters.

## Initialization

- Algorithms are robust to the choice of the initialization step.
- Random generation of the representatives.
- Random sampling of the dataset.
- A centroids of m randomly chosen samples.

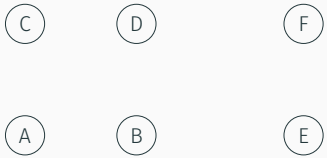
# Hierarchical Clustering

- Creates a hierarchical structure above the objects from the dataset.
- The different levels of clustering granularity provide different application-specific insights to the data.
- Hierarchical organization of the data allows even better flat cluster

# Hierarchical Clustering - Algorithm types

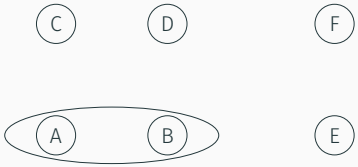
- Bottom-up (agglomerative) methods
  - Individual data objects are agglomerated into higher level clusters.
  - Objective function is used for computing similarity.
- Top-down (divisive) methods
  - Partitioning of the data objects into tree-like structure.
  - A flat clustering algorithm may be used for the partitioning in a given step.
  - A trade-off in balance of the tree between number of clusters and the number of objects in each cluster/leaf.

# Hierarchical Clustering - Bottom-Up Agglomerative Methods

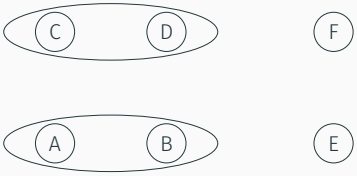



A B C D E F

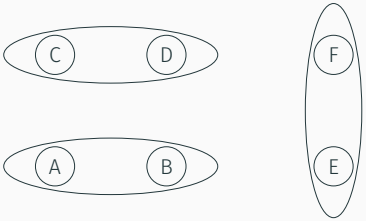
# Hierarchical Clustering - Bottom-Up Agglomerative Methods



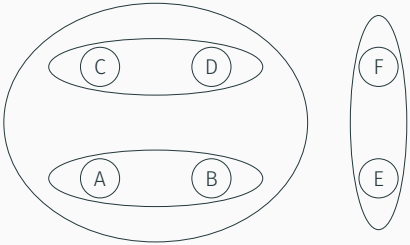

# Hierarchical Clustering - Bottom-Up Agglomerative Methods

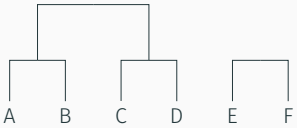



# Hierarchical Clustering - Bottom-Up Agglomerative Methods

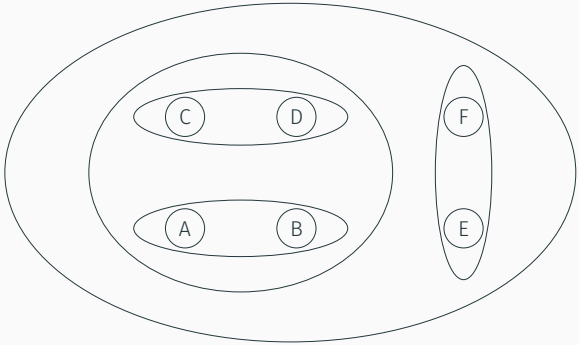


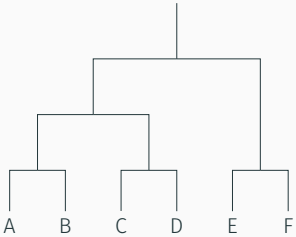

# Hierarchical Clustering - Bottom-Up Agglomerative Methods



# Hierarchical Clustering - Bottom-Up Agglomerative Methods



# Hierarchical Clustering - Bottom-Up Agglomerative Methods

- Iterative approach starting with individual data object.
- Two clusters are merged in each iteration.
- Each merging step reduces the number of clusters by one.
- A carefully selected measure for computation of the distance between individual objects need to be defined.
- A proper strategy for measuring the distance between clusters need to be defined also.
- A distance matrix should be stored in a memory, the computational complexity increases when not.

# Hierarchical Clustering - Bottom-Up Agglomerative Methods

**Algorithm 2:** AgglomerativeMerge(Dataset: $D$)

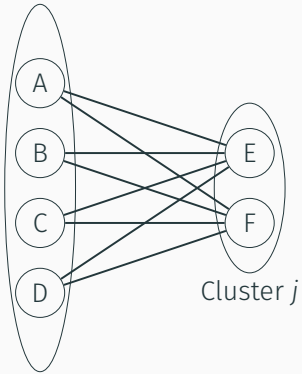
```
1  begin
2      Initialize n × n distance matrix M using D;
3      repeat
4          Pick the closest pair of clusters i and j using M;
5          Merge clusters i and j;
6          Delete rows/columns i and j from M and create a new row and column
             for newly merged cluster;
7          Update the entries of the new row and column of M;
8      until termination criterion;
9      return current merged cluster set
10 end
```

## Group Similarity Computation

- Distance between two groups of objects need to be computed.
- The distance is a function of the distances between all pairs of objects from different cluster.

$$D(C_i, C_j) = func_{\forall x \in C_i, \forall y \in C_j}(d(x, y))$$

- Different strategies exists.
- Each criterion has different advantages and disadvantages.



# - Bottom-Up Agglomerative Methods - Group Similarity Computation

- Best (single) linkage.
  - The distance is equal to the minimum distance between all pairs of objects.
  - Its corresponds to the closes pair of objects between the two groups.
  - Very efficient approach in discovering clusters of arbitrary shape.
  - Very sensitive to noise that connects different clusters.

$$D(C_i, C_j) = \min_{\forall x \in C_i, \forall y \in C_j} \{d(x, y)\}$$

- Worst (complete) linkage:
  - The distance is equal to the maximum distance between all pairs of objects.
  - Its corresponds to the farthest pair of objects between the two groups.
  - This criterion attempts to minimize the maximum diameter of a cluster.

$$D(C_i, C_j) = \max_{\forall x \in C_i, \forall y \in C_j} \{d(x, y)\}$$

- Group-average linkage
  - The distance is equal to the average distance between all pairs of objects.
  - A weighted average is used for computation.

$$D(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y)$$

- Closest centroid
  - The closest centroid are merged in each iteration.
  - The centroids lose information about the relative spreads of the clusters.
  - The method will not discriminate between clusters of varying sizes.
  - Typically larger clusters has statistically more likely centroid closer to each other than smaller clusters.

- Variance-based criterion
  - This criterion minimizes the cluster variance objective function during merging.
  - Merging always results into worsening of the clustering objective function due to loss of granularity.
  - Each cluster maintain $0^{th}$, $1^{st}$ and $2^{nd}$ order moment statistics.
  - The average squared error of a cluster $i$ is defined as:

$$SE_i = \sum_{r=1}^{d} \left( \frac{S_{ir}}{m_i} - \frac{F_{ir}^2}{m_i^2} \right)$$

  - $m_i$ is the number of points, $S_{ir}$ squared sum of the data points in a cluster across each direction $r$, $F_{ir}$ is a sum of data points along each direction.

- Variance-based criterion
  - The moment statistics of a merge of two clusters is the sum of the clusters moment statistics.
  - The change of a variance on executing merge on clusters $i$ and $j$ is defined as:

$$\Delta SE_{i \cup j} = SE_{i \cup j} - SE_i - SE_j$$

  - This change is always positive.
  - The pair of clusters with the smallest variance change is selected for merge.

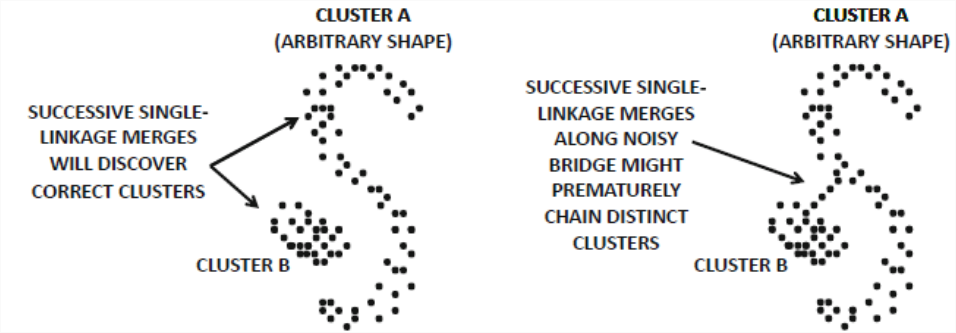
- Ward's method
  - Slightly different approach than Variance-based criterion.
  - It is unscaled sum of squared error as the merging criterion.

$$SE = \sum_{r=1}^{d} \left( m_i S_{ir} - F_{ir}^2 \right)$$

  - It's a variant of the centroid method.
  - It's the equivalent of the multiplication of the squared Euclidean distance between centroid and the harmonic mean of the number of points in each pair.
  - Larger clusters are penalized.

# - Bottom-Up Agglomerative Methods - Group Similarity Computation

- Single linkage method is able to discover clusters of arbitrary shape, but it may merge clusters connected by noise points.



# - Bottom-Up Agglomerative Methods - Group Similarity Computation

- Complete linkage method
  - Focus on the minimization of the maximum distance between pair of points.
  - This may be viewed as the approximation of the cluster diameter.
  - The method tries to create clusters with the similar diameter.
  - Larger natural cluster may be broken into small clusters.
  - The created clusters tends to be spherical.
- Group-average, variance and Ward's method are robust to the noise.

- A heap of sorted distances need to be maintained for efficient minimal distance determination.
- Initial matrix computation requires $O(n^2 \cdot d)$ time.
- The maintenance of the sort heap requires $O(n^2 \cdot \log n)$. The required space for distance matrix is $O(n^2)$.
- When a distance matrix didn't fit into memory the computation time is $O(n^3 \cdot d)$.

# - Bottom-Up Agglomerative Methods - Implementation notes

- The results is a binary tree of clusters.
- It is very difficult to control the structure of the tree.
- Difficult to use when the specific structure is desired.
- The method is sensitive to a small number of mistakes during merging process, therefore, its very sensitive to the noise.
- These methods are impractical for large dataset.
- Frequently combined with sampling and partitioning methods.

# Hierarchical Clustering - Top-Down Divisive Methods

- Uses a general-purpose flat-clustering algorithms as a subroutine.
- The algorithm initializes the tree at the root with all points.
- The particular node is split into multiple nodes in each iteration.
- The strategy for selection of the node to split may affect the balanced tree by height or number of clusters.
- When the clustering subroutine is stochastic, several trials are tested the best is selected.

# Hierarchical Clustering - Top-Down Divisive Methods

**Algorithm 3:** GenericTopDownClustering(Dataset: $D$, Flat Algorithm: $A$)

```
1 begin
2     Initialize tree T to root containing D;
3     repeat
4         Select a leaf node L in T based on pre-defined criterion;
5         Use algorithm A to split L into L_1, ..., L_k;
6         Add L_1, ..., L_k as children of L in T;
7     until termination criterion;
8     return tree T
9 end
```

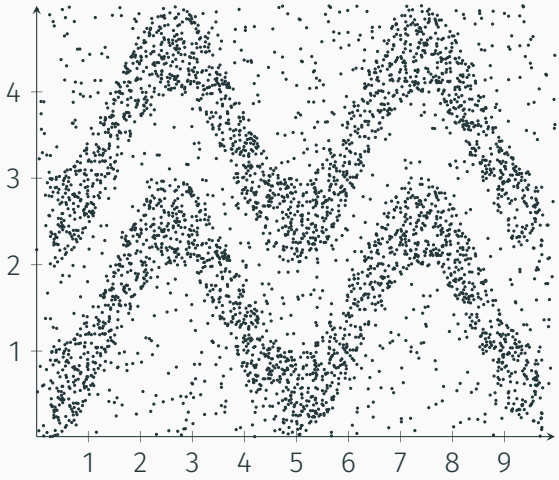
# Hierarchical Clustering - Top-Down Divisive Methods - Example

- Bisecting k-Means
  - Each node in split into two children with 2-means algorithm.
  - Several runs of randomized initialization are used.
  - The one with the best impact on the overall clustering objective is used.
  - Several strategies for node selection exists.

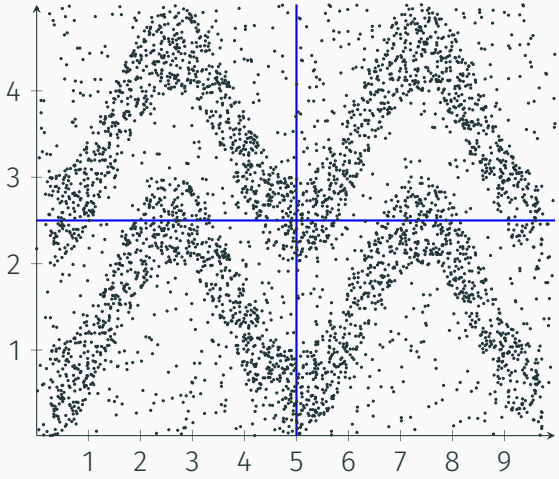
# Density-based Clustering

- Works in different based than distance based algorithm.
  - The shape of the cluster is not defined explicitly.
  - The number of clusters is also unknown and has not be defined explicitly.
- The idea is to identify fine-grained dense regions in the data.
- Clustering of these regions will produces arbitrary-shaped clusters.
- Its can be considered as two-level hierarchical clustering algorithm.
- The second level may be more detailed, due to lover number of points (similarly to the single-linkage agglomerative algorithm).

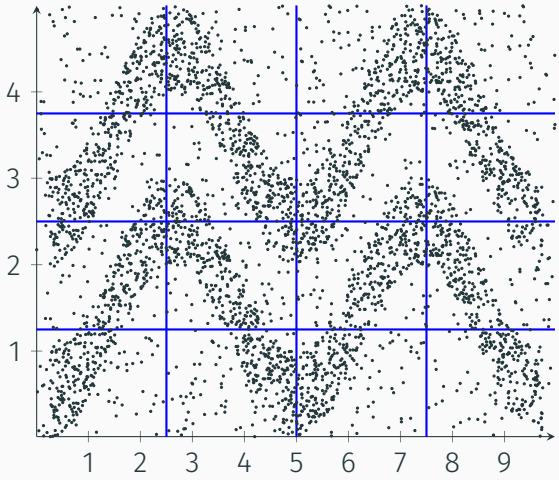
# Density-based Clustering



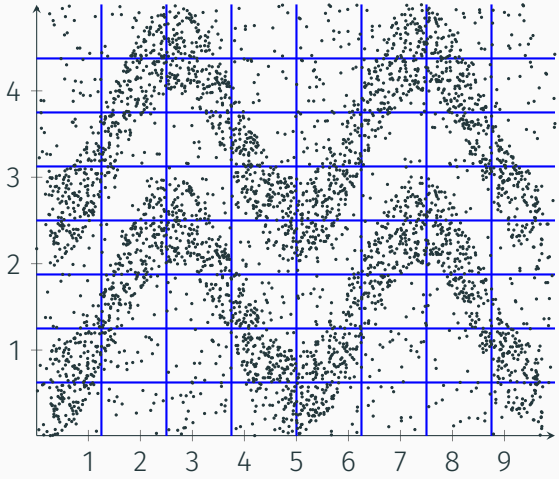
# Density-based Clustering



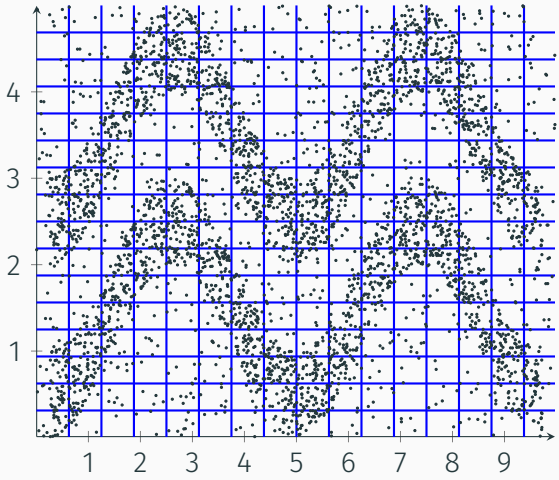
# Density-based Clustering



# Density-based Clustering



# Density-based Clustering



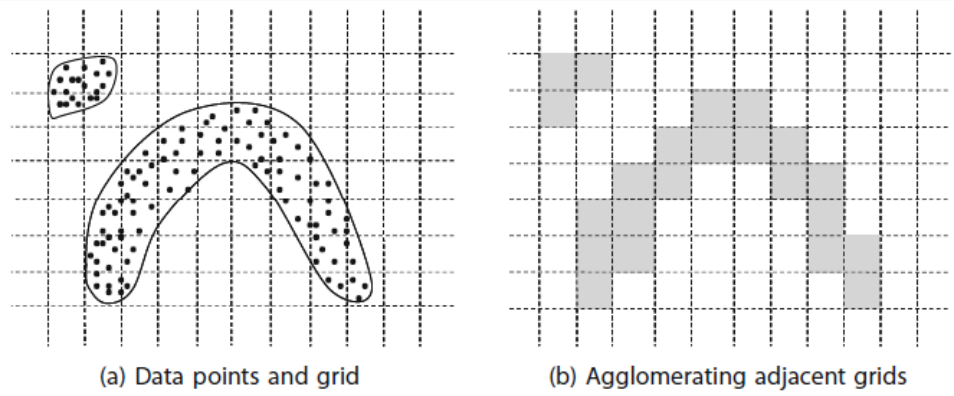
# Density-based Clustering - Grid-based algorithms

- The data is discretized into $p$ intervals (usually equiwidth).
- For $d$-dimensional data set, $p^d$ hyper-cubes is considered.
- A density threshold $\tau$ is used to determine the number of hyper-cubes that are dense. ($\tau$ is the minimal number of points in each hyper-cube).
- The clusters are formed from connected dense regions.
- The adjacent connectivity:
  - Two grid regions are adjacently connected if they share a side in common.
  - Two grid regions are adjacently connected if they share a corner in common.

# Density-based Clustering - Grid-based algorithms

- Two grid regions are density connected, if a path can be found from the region to the other containing only a sequence of adjacently connected grid regions.
- The goal of the grid-based clustering is to find connected regions.
- This may be simply done if we use a graph-based model.
- Connected region may be determined using breath-first or depth-first traversal.

# Density-based Clustering - Grid-based algorithms



(a) Data points and grid

(b) Agglomerating adjacent grids

**Algorithm 4:** GridBasedCluster(Dataset: $D$, Ranges: $p$, Density: $\tau$)

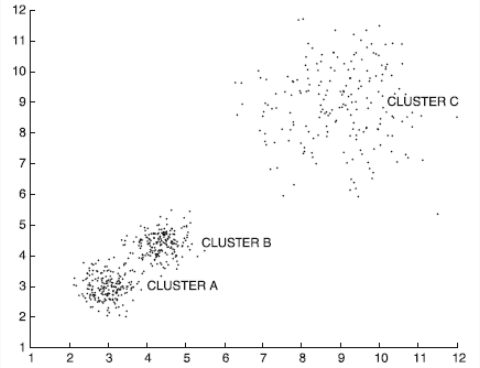
```
1 begin
2     Discretize each dimension of data D into p ranges;
3     Determine dense grid cells as density level τ;
4     Create graph in width dense grids are connected if they are adjacent;
5     Determine connected components of graph;
6     return current points in each connected components as a cluster
7 end
```

# Density-based Clustering - Grid-based algorithms

- The number of clusters is not pre-defined in advance. Two parameters need to be defined
  - Grid ranges $p$
  - Density threshold $\tau$
- Definition of these points is often very difficult.
- The wrong choice can lead to unintended consequences:
  - Too small p will lead into mixing points from different clusters.
  - Too big p will produce too many empty cells even within the cluster.
  - Too low τ will produce too many dense region even with only noise points.
  - Too high τ will reduce the cluster size or even the number of clusters.

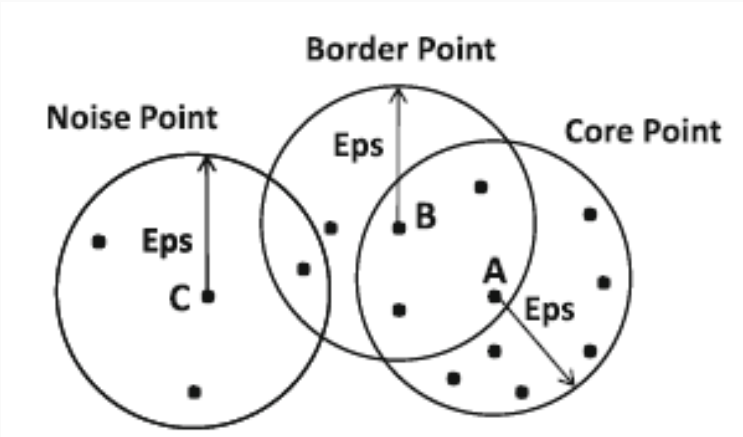
# Density-based Clustering - Grid-based algorithms

- The grid based methods deals well when the density of all natural clusters is similar.
- The varying density of the clusters cannot be reflected because the τ parameter is set globally.
- Rectangular regions are problematic for high-dimensional data.



# DBSCAN

- A very similar algorithm to the grid-based method.
- Rectangular regions are substituted with spherical area with defined radius *Eps*.
- The points are classified into three categories:
  - Core point – a point that has at least $\tau$ points in a radius *Eps*.
  - Border point – a points that has less than $\tau$ points in a radius of *Eps* but has a core point in the *Eps*-neighborhood.
  - Noise point – a point that is neither a core or border point.


Border Point
Noise Point
Core Point
Eps
Eps
Eps
B
A
C

# DBSCAN

**Algorithm 5:** DBSCAN(Dataset: *D*, Threshold: *Eps*)

```
1  begin
2      Determine the core, border and noise points;
3      Create connectivity graph from the core points;
4      begin
5          Each node correspons to the core point;
6          The edges are added only when the distance between the core points
             is lower than a Eps;
7      end
8      All connected components are identified;
9      The border points are assigned to the clusters with which they have the
         highest connectivity;
10     return clusters as connected components and noise points as outliers
11 end
```

# DBSCAN

- The principle of the algorithm better works with noise points.
- The contour of the clusters is more smooth.
- The definition of the *Eps*-radius and the $\tau$ need to be defined.
- The problems with the dense variance is the same as the grid-based methods.
- The parameters *Eps* and $\tau$ are related to one another in an intuitive way.
- The user define $\tau$ the parameter *Eps* may be determined in a data-driven way. (The *Eps* should capture most points as core points).

# DENCLUE

- It is based on the kernel density estimation.
- The kernel estimation is used to create a smooth profile of the density distribution.
- The density at point $X$ is denoted as $f(X)$:

$$f(X) = \frac{1}{n} \sum_{i=1}^{n} K(X - X_i)$$

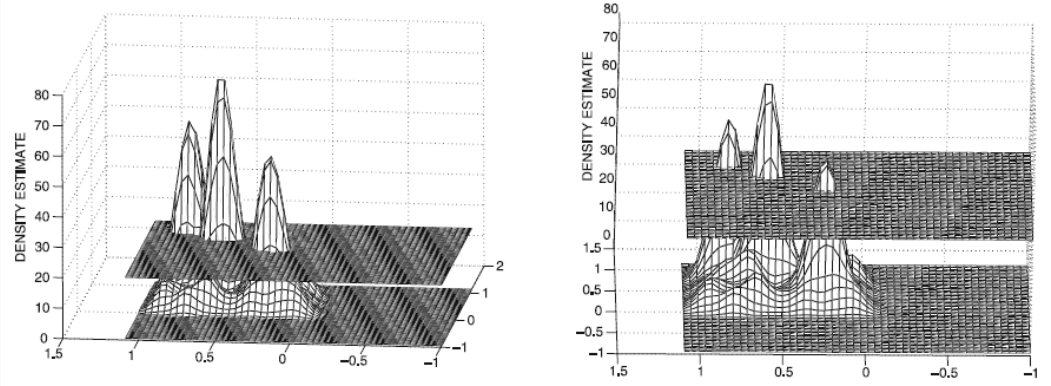
- Any possible kernel function may be used (Uniform, Gaussian, Triangular, ...).

- The kernels are used with respect to the data dimension $d$.
- The Gaussian is one of the common choice:

$$K(X - X_i) = \left(\frac{1}{h\sqrt{2\pi}}\right)^d e^{-\frac{\|X - X_j\|^2}{2h^2}}$$

- The Term $\|X - X_i\|$ is an Euclidean distance between data points.
- The $h$ is the bandwidth estimation that regulates the smoothness of the kernel estimation.
- The density threshold $\tau$ is defined and affects the cluster determination process.

# DENCLUE


DENSITY ESTIMATE
80
70
60
50
40
30
20
10
0
1.5
1
0.5
0
−0.5
−1
2
1
0
−1
DENSITY ESTIMATE
80
70
60
50
40
30
20
10
0
1.5
1
0.5
0
−0.5
−1
1.5
1
0.5
0
−0.5
−1

# DENCLUE

- The algorithm uses the notion of the density attractors.
- The idea is to treat each local peak of the density as the attractor.
- Each data point is associated to the density attractors using hill climbing method towards its relevant peak.
- The peaks that are connected by density path at least $\tau$ are connected.
- The algorithm uses iterative gradient ascent method with each data point.

$$X^{(t+1)} = X^{(t)} + \alpha \nabla f \left( X^{(t)} \right)$$

- The $\nabla f\left(X^{(t)}\right)$ is a $d$-dimensional vector of partial derivatives of the density estimation.
- The $\alpha$ is a step size.
- The partial derivatives of the kernel-density estimation function can be computed easily using the gradient of the constituent kernel-density values.

$$\nabla f\left(X\right) = \frac{1}{n}\sum_{i=1}^{n} \nabla K(X - X_i)$$

# DENCLUE - An algorithm

**Algorithm 6:** DENCLUE(Dataset: $D$, Threshold: $\tau$)

1 **begin**
2 Determine the density attractor of each dat apoint in a dataset with gradient ascent rule;
3 Create clusters of data points that converge to the same density attractors;
4 Discard clusters whose density attractors have density less than $\tau$ (noise and outliers);
5 Merge clusters whose density attractors are connected with a path of density at least $\tau$;
6 **return** *clusters*
7 **end**

# DENCLUE

- A different way for determination of the local optimum is setting gradient to zero.
- Usually a nonlinear system of equations appears.
- The update rules derived from this system has usually much faster convergence.
- The DBSCAN algorithm may be considered as a special function of the DENCLUE.

# Cluster Validation

- Internal Validation Criteria
  - Sum of Square Distances to Centroids
  - Intra-cluster to Inter-cluster distance ratio.
  - Silhouette coefficient
  - Probabilistic measure
- External Validation Criteria
  - Purity
  - Gini index
  - Entropy

# Cluster Validation - Internal Validation Criteria

- Useful when no external criteria is available.
- The major problem if internal criteria is that they are biased toward one or another algorithms.
- The criteria is usually borrowed from the objective function used by certain algorithms.
- The main usage of these criteria is for comparison of the algorithm from the same class or different run of the same algorithm.

### Sum of Square Distances to Centroids

- Useful when centroids are determined – mainly distance-based algorithms.
- The sum of squared distances of each point to corresponding centroid is used as a quality measure.
- The smaller value indicate better clustering quality.

$$SSQ = \sum_{X \in D} dist(X, C)^2$$

- Where $C$ is the closest centroid to $X$.

## Intra-cluster to Inter-cluster distance ratio

- Based on sets of random pairs of objects.
- The $P$ is a set of pairs that belong to the same cluster.
- The $Q$ is a set of pairs that does not belong to the same cluster.
- The average distances are defined as follows:

$$Intra = \frac{1}{|P|} \sum_{(X_i, X_j) \in P} dist\left(X_i, X_j\right)$$

$$Inter = \frac{1}{|Q|} \sum_{(X_i, X_j) \in Q} dist\left(X_i, X_j\right)$$

- The ratio $Intra/Inter$ is a quality measure. Smaller values means higher quality.

## Silhouette Coefficient

- Compares similar distances as the previous one.
- $Davg_i^{in}$ is the average distance of $X_i$ to data points within the cluster of $X_i$.
- $Dmin_i^{out}$ is the minimum of the average distances to all other clusters.

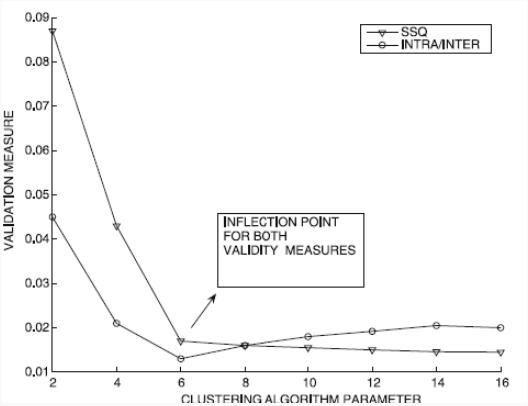
$$S_i = \frac{Dmin_i^{out} - Davg_i^{in}}{\max\left\{Dmin_i^{out}, Davg_i^{in}\right\}}$$

- The overall silhouette coefficient is the average of the data point-specific coefficients.
- The value is in the range $\{-1, 1\}$. Large positive values indicate highly separated clustering, large negative value indicate a ”mixing” between clusters.

# Cluster Validation - Internal Validation Criteria

## Parameter tuning

- Each algorithm has several parameters that need to be set manually (number of clusters, density threshold, etc.).
- The internal measures may be used for precise definition of these parameters



# Cluster Validation - External Validation Criteria

- These criteria are available when the ground truth is know.
- In the real datasets, the ground truth is not know usually.
- An approximation may be achieved using available class labels.
- These labels should not correspond to the natural clusters.
- Despite these problems, external evaluation criteria are preferable.
- The number of natural clusters may not reflect the number of classes.

# Cluster Validation - External Validation Criteria

- When the number of determined clusters and the number of classes is equal, a confusion matrix is useful.

| Cluster Indices | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 97 | 0 | 2 | 1 |
| 2 | 5 | 191 | 1 | 3 |
| 3 | 4 | 3 | 87 | 6 |
| 4 | 0 | 0 | 5 | 195 |

| Cluster Indices | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 33 | 30 | 17 | 20 |
| 2 | 51 | 101 | 24 | 24 |
| 3 | 24 | 23 | 31 | 22 |
| 4 | 46 | 40 | 44 | 70 |

## Cluster purity and class-based Gini index

- Let $m_{ij}$ represents the number of data points from the class $i$ that are mapped to determined cluster $j$.
- The number of data points in true class/cluster is denoted as $N_i$. The number of data points in determined cluster is denoted as $M_i$.

$$N_i = \sum_{j=1}^{k_d} m_{ij} \quad \forall i = 1 \dots k_t$$

$$M_i = \sum_{j=1}^{k_t} m_{ij} \quad \forall i = 1 \dots k_d$$

- $k_t$ is number of class labels, $k_d$ is number of clusters.

## Cluster purity and class-based Gini index

- High quality algorithm-determined cluster j should contain data points that are dominated by a single class.
- The number of points in a dominant class of a cluster $j$ is defined as:

$$P_j = \max_i \{m_{ij}\}$$

- The overall purity is defined as:

$$Purity = \frac{\sum_{j=1}^{k_d} P_j}{\sum_{j=1}^{k_d} M_j}$$

- Purity may be computed for ground truth clusters as well.
- The average of these values may be used as a final measure.

## Cluster purity and class-based Gini index

- Purity ignores the distribution of the non-dominant classes.
- The Gini index of determined cluster evaluates even this property.

$$G_j = 1 - \sum_{i=1}^{k_t} \left( \frac{m_{ij}}{M_j} \right)^2$$

- The $G_j$ is small when the values are skewed, values close to $1 - \frac{1}{k_t}$ when the distribution is uniform.
- The average Gini coefficient is weighted by the number of points $M_j$

$$G_{average} = \frac{\sum_{j=1}^{k_d} G_j \cdot M_j}{\sum_{j=1}^{k_d} M_j}$$

## Entropy

- The Entropy measures the same characteristic of the data as the Gini index.
- The Gini index of determined cluster evaluates even this property.

$$E_j = -\sum_{i=1}^{k_t} \left( \frac{m_{ij}}{M_j} \right) \cdot \log \left( \frac{m_{ij}}{M_j} \right)$$

- The average Entropy is weighted by the number of points $M_j$.

$$E_{average} = \frac{\sum_{j=1}^{k_d} E_j \cdot M_j}{\sum_{j=1}^{k_d} M_j}$$

# Clustering - Clustering Interpretations

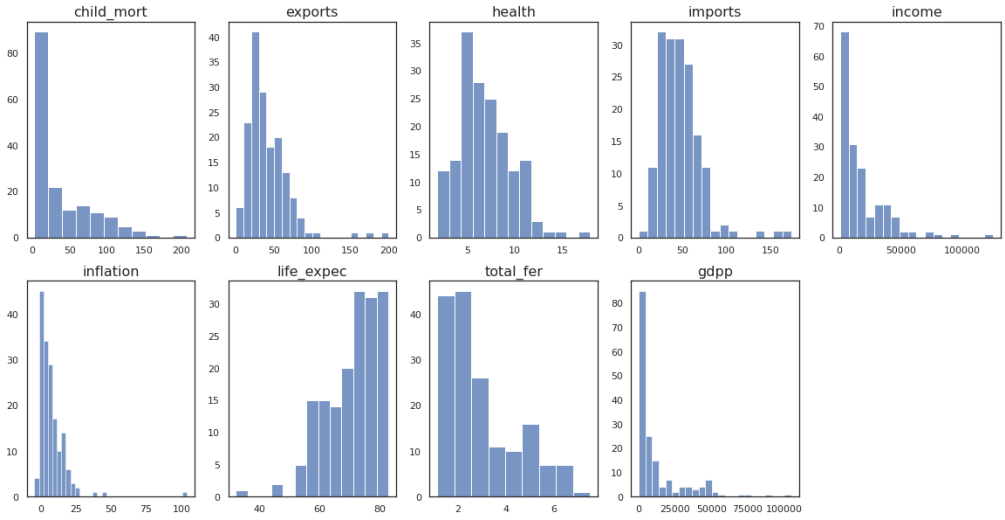
- The quality of the clusters is not as important as the meaning.
- The detected clusters represents the group of objects.
- The groups should contain different type of object to be meaningful.
- Evaluation of the meaning may be done using exploration analysis between clusters.

- **Unsupervised Learning on Country Data** (Link)
- **Objective:** To categorize the countries using socio-economic and health factors that determine the overall development of the country.
- **Shape:** 167 rows, 10 columns.
- **Description:** Float numbers with different distribution.

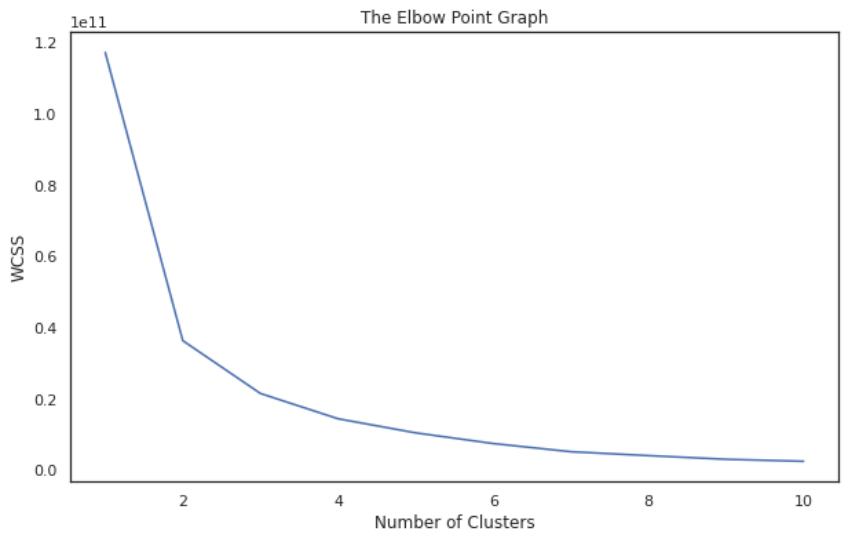
# Clustering - Clustering Interpretations - Example - Columns

| | |
|---|---|
| country | Name of the country |
| child_mort | Death of children under 5 years of age per 1000 live births |
| exports | Exports of goods and services per capita. Given as percentage of the GDP per capita |
| health | Total health spending per capita. Given as percentage of GDP per capita |
| imports | Imports of goods and services per capita. Given as percentage of the GDP per capita |
| Income | Net income per person |
| Inflation | The measurement of the annual growth rate of the Total GDP |
| life_expec | The average number of years a new born child would live if the current mortality patterns are to remain the same |
| total_fer | The number of children that would be born to each woman if the current age-fertility rates remain the same. |

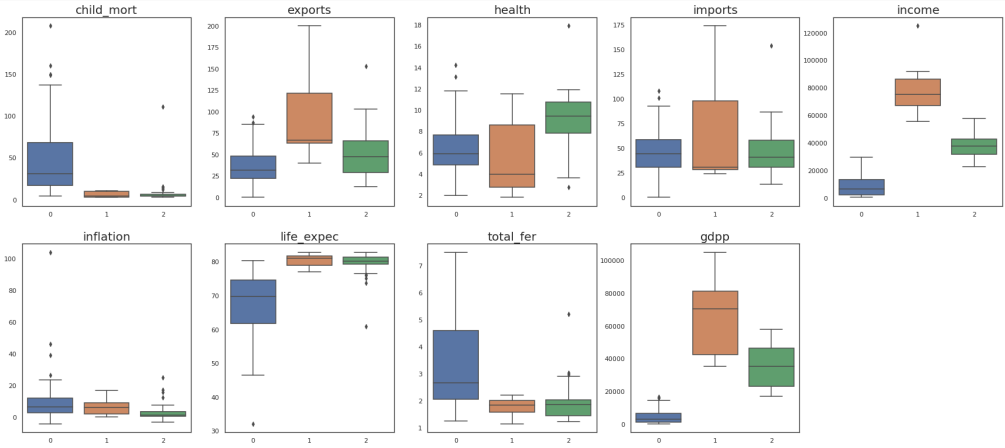
# Clustering - Clustering Interpretations - Example



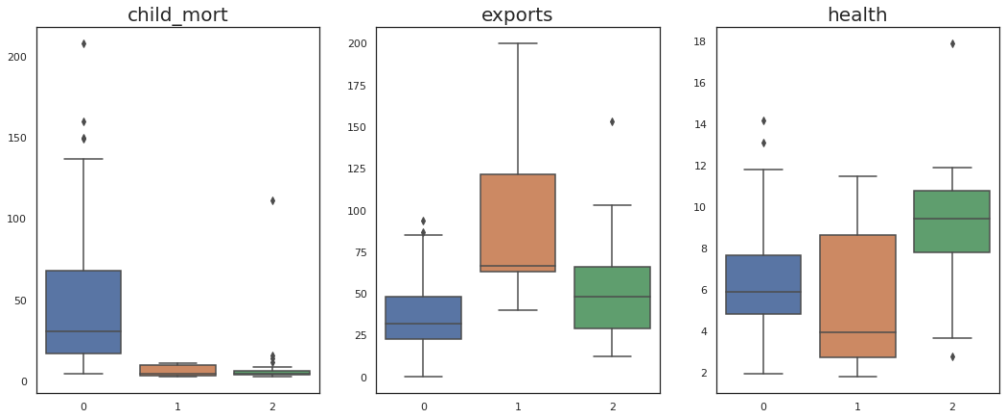
# Clustering - Clustering Interpretations - Example



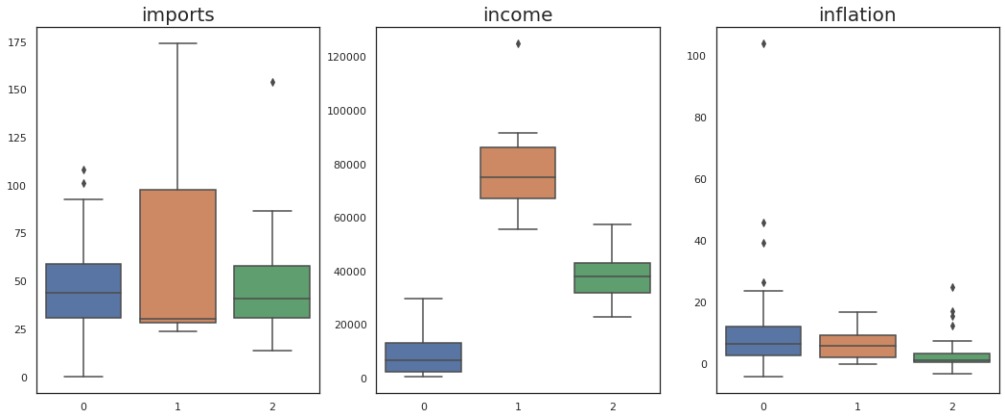
# Clustering - Clustering Interpretations - Example



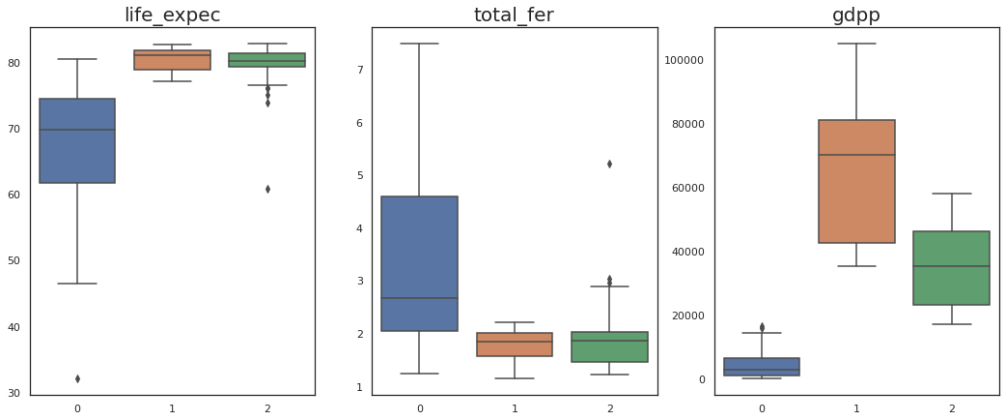
# Clustering - Clustering Interpretations - Example



# Clustering - Clustering Interpretations - Example



# Clustering - Clustering Interpretations - Example



# Clustering - Clustering Interpretations - Example

| | |
|---|---|
| Cluster 0 | Afghanistan, Albania, Algeria, Angola, Antigua and Barbuda, Argentina, Armenia, Azerbaijan, Bangladesh, Barbados, Belarus, Belize, Benin, Bhutan, Bolivia, Bosnia and Herzegovina, Botswana, Brazil, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Cape Verde, Central African Republic, Chad, Chile, China, Colombia, Comoros, Congo, Dem. Rep., Congo, Rep., ... |
| Cluster 1 | Brunei, Kuwait, Luxembourg, Norway, Qatar, Singapore, Switzerland |
| Cluster 2 | Australia, Austria, Bahamas, Bahrain, Belgium, Canada, Cyprus, Czech Republic, Denmark, Equatorial Guinea, Finland, France, Germany, Greece, Iceland, Ireland, Israel, Italy, Japan, Malta, Netherlands, New Zealand, Oman, Portugal, Saudi Arabia, Slovenia, South Korea, Spain, Sweden, United Arab Emirates, United Kingdom, United States |

## Difficult clustering scenarios

- Categorical data clustering
  - Problematic definition of the mean of the cluster.
  - Difficult definition of the similarity.
- Scalable clustering
  - Clustering of large amount of data.
  - Many algorithm needs several passes.
  - The data cannot be accommodated into memory.
- High-dimensionality clustering
  - Many irrelevant attributes.
  - Too high dimensionality suffers by the curse of dimensionality.

## Improved unsupervised algorithms

- Semi-supervised clustering
  - Partial information about underlying clusters is available.
  - Such information may improve the clustering results.
- Interactive and visual clustering
  - The feedback from the user is used for clustering improvement.
  - A visual interaction significantly helps with understanding of the data.
- Ensemble clustering
  - Multiple models is used for clustering.
  - The best model or combination of models is chosen.

# Clustering - Special Clustering algorithms

- Difficult computations:
  - Distance computation.
  - Representative determination.
  - Density estimation as well.
- Binarization may be used for conversion of the categorical data.
  - Algorithms have to be modified to deal with binary data.

# Categorical Data Clustering - Representative-based Clustering

- Centroid of a categorical data set
  - Centroid is equivalent to a probability histogram of each attribute.
  - For a *d*-dimensional dataset is a centroid a set of *d* different histograms.

| Row | Color | Shape |
|---|---|---|
| 1 | Blue | Square |
| 2 | Red | Circle |
| 3 | Green | Cube |
| 4 | Blue | Cube |
| 5 | Green | Square |
| 6 | Red | Circle |
| 7 | Blue | Square |
| 8 | Green | Cube |
| 9 | Blue | Circle |
| 10 | Green | Cube |

| Attribute | Histogram | Mode |
|---|---|---|
| Color | Blue = 0.4<br>Green = 0.4<br>Red = 0.2 | Blue or Green |
| Shape | Cube=0.4<br>Square=0.3<br>Circle=0.3 | Cube |

# Categorical Data Clustering - Representative-based Clustering

- Calculating similarity to centroids
  - A match similarity may be used.
  - The probability of an attribute value is summed up for each attribute.
  - The total similarity is then calculated.
  - The object is assigned to the cluster with the highest similarity value.
- The rest of the algorithm is the same as for numeric data.
- This principle works well for non skewed data.
- Skewed data may be modified using weights that prefer rare items.
- The weights have to be incorporated into similarity measurement and histogram generation.

# Categorical Data Clustering - $k$-Modes Clustering

- A mode is selected for as a representative attribute.
- A mode is the most probable value of an attribute in a cluster.
- The mode is usually not a one of the objects in a cluster.
- The representative is also a categorical data object.
- The modes should not be used for sparse data.
- The modes works well for evenly distributed data.
- The unevenly distributed data may be normalized.

## Robust Clustering using Links (ROCK)

- Agglomerative bottom-up based approach.
- Shared nearest-neighbor metric is used as a merging criterion.
- Due to its computational complexity, ROCK uses only sample points.
- The computed prototype clusters are used created.
- The remaining points are assigned to these clusters in the final pass.
- The data are converted into binary form.
- When each attribute may have many values, sparse dataset is created (very similar to the market data).
- Each record is then treated as a transaction.

## Robust Clustering using Links (ROCK)

- The similarity between transactions is computed using Jaccard coefficient.

$$Sim(T_i, T_j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j|}$$

- Two points $T_i$ and $T_j$ are defined to be neighbors if the similarity between them is greater than a specific threshold $\theta$.
- This concept defines a graph structure on the data items.
- The $Link(T_i, T_j)$ denotes the shared nearest-neighbor similarity function, i.e. the number of shared nearest neighbors between $T_i$ and $T_j$.
- The $Link(T_i, T_j)$ is used as a merging criterion for agglomerative algorithms.

## Robust Clustering using Links (ROCK)

- The clusters are merged is a cumulative number of shared neighbors between clusters $C_i$ and $C_j$ is large.

$$GroupLink(C_i, C_j) - \sum_{T_u \in C_i, T_v \in C_j} Link(T_u, T_v)$$

- Large clusters usually have larger number of links than smaller, i.e. normalized linkage criterion may be used.

$$V(C_i, C_j) = \frac{GroupLink(C_i, C_j)}{E\left[CrossLink(C_i, C_j)\right]}$$

$$E\left[CrossLink(C_i, C_j)\right] = E\left[Intra(C_i \cup C_j)\right] - E\left[Intra(C_i)\right] - E\left[Intra(C_j)\right]$$

- *Intra* is the expected number of the intra-cluster links.

## Robust Clustering using Links (ROCK)

- The expected number intra-cluster links is estimated using cluster size and similarity threshold $\theta$.
- The ROCK uses the following equation

$$Intra(C_i) = q^{1+2 \cdot f(\theta)}$$

$$f(\theta) = \frac{1-\theta}{1+\theta}$$

- The merges are successively performed until a total of k clusters remains.
- The remaining points (not sampled for the algorithm) are assigned to the clusters.

## CLARA

- A scalable implementation of the $k$-medoids algorithm.
- $k$-medoids
  - Initial clusters are selected randomly.
  - A set of pairs $(X, Y)$ is randomly generated, $X$ from dataset and $Y$ from representatives, and the best pair is used for exchange.
- Based on the Partitioning Around Medoids (PAM).
  - All possible $(k \cdot (n - k))$ pairs are evaluated for replacement and the best is selected.
  - This is repeated until convergence to the local optimum.
  - $O(kn^2d)$time complexity for $d$-dimensional dataset.

## CLARA

- Due to high computational complexity only subset of points is selected for exhaustive search.
- A sampling fraction f is selected, where $f << 1$.
- Non-sampled points are assigned to the nearest medoids.
- The sampling is repeated over independently chosen samples os the same size $f \cdot n$.
- The best clustering is then selected.
- Time complexity if for one iteration

$$O(k \cdot f^2 \cdot n^2 \cdot d + k \cdot (n - k))$$

## CLARANS

- Solves problem of CLARA when no good choice of medoids is present in any of the sample.
- CLARANS states for Clustering Large Applications based on Randomized Search.
- The algorithm works with the full data set (no samples).
- The algorithm iteratively attempts exchanges between random medoids with random non-medoids.

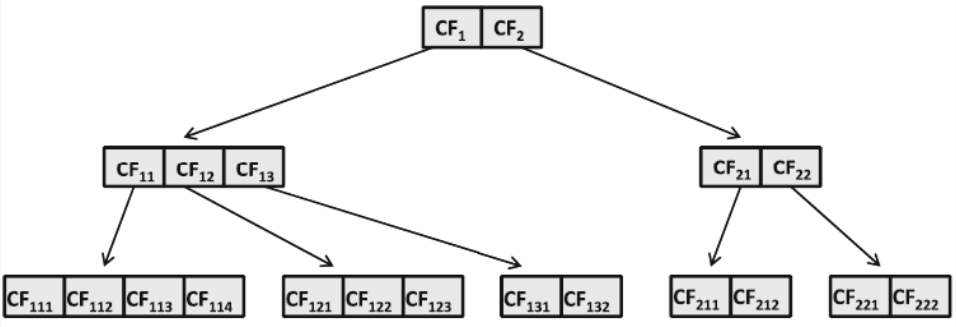
## CLARANS

- The quality of the exchange is checked after each attempts.
  - When improves, the exchange is made final.
  - When not, unsuccessful exchange attempts counter is incremented.
- A local optimal solution is found when a user-specified number of unsuccessful attempts *MaxAttempt* is reached.
- This process of finding the local optimum is repeated for a user-specified number of iterations - *MaxLocal*.
- The clustering objective is evaluated for each iteration and the best is selected as the optimal.
- The advantage of CLARANS over CLARA is that a greater diversity of the search space is explored.

## BIRCH

- The Balanced Iterative Reducing and Clustering using Hierarchies.
- A combination of the top-down hierarchical clustering with the k-means clustering.
- A special data structure CF-Tree is used.
  - A height balanced data structure for cluster organization.
  - Each node has branching factor at most B (the arity, similarly to B-tree).
  - It is designed to support dynamic insertions.

# Scalable Data Clustering - CF Tree data structure



## BIRCH

- Each node contains a summary of each of at most B subclusters.
- This info is called cluster feature (CF) vector.
  - The SS is a vector of a sum of the squares of the points in the cluster (2nd order moment).
  - The LS is a vector of a linear sums of the points in the cluster (1st order moment).
  - The m is the number of points in the cluster (0th order moment).

## BIRCH

- It has important properties
  - Each CF can be represented as a linear sum of the CFs of the individual data points.
  - The CF of a parent node in the CFTree is the sum of the cluster features of its children.
  - The CF of a merged cluster can also be computed as the sum of the CFs of the constituent clusters.
  - Incremental updates of the CF vector can be efficiently achieved by adding the CF vector of a data point to that of the cluster.
  - The CFs can be used to compute useful properties of a cluster (e.g. radius and centroid).

## BIRCH

- Each leaf node is the CF-tree has a diameter threshold $T$.
- The value $T$ regulates the granularity of the clustering, the height of the tree and the aggregate number of clusters at the leaf nodes.
- Lower $T$ leads to larger number of fine-grained clusters.
- The $T$ is based on the size of the dataset due to memory size (smaller dataset may use smaller $T$).
- The $T$ is increased when the tree can no longer be kept in memory.

## BIRCH

- The insertion into tree is made in top-down approach.
- The closest centroid is used in each level for the point.
- The CFs are updated along the path of the insertion by addition.
- At the leaf node, the point is inserted into corresponding cluster only if it not violate the $T$ diameter condition.
- Otherwise a new cluster is create with only this point inside.
- The cluster is inserted into the tree, a split may be necessary.
- When the splitting leads into memory constraint violation, the tree is rebuild with higher $T$ threshold.

## CURE

- The Clustering using Representatives.
- An bottom-up agglomerative hierarchical algorithm.
- It replaces the direct computation of the distances between all pair of points with computation using representatives for better efficiency.
- These representatives are carefully chosen to capture the shape of each of the current clusters.
- The first representative is chosen to be a data point farthest from the center of the cluster.
- The second is farthest from the first.
- The third is chosen to be farthest from the closest from the two, etc.
- The representatives tends to be on the contour of the cluster.

## CURE

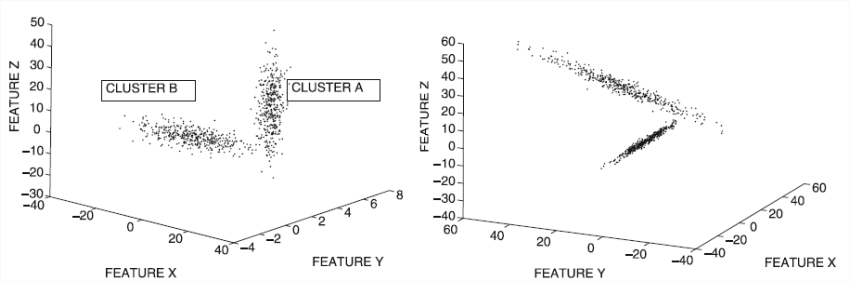
- Typically, a small number of representatives is chosen from each clusters.
- The farthest distance favors outliers to be selected.
- A shrinking toward the center is used to avoid this.
- The shrinking replaces the representatives with synthetic new ones, where the new has distance from the center as a fraction of the original one.
- The distances are measured between cluster representatives only.
- Small clusters that remain small during clustering are eliminated as an outliers.
- An complexity improvement is also achieved using random sampling of the dataset and partitioning of these samples that are reclustered.

## CURE - An Algorithm

1. Sample $s$ points from the database $D$ of size $n$.
2. Divide the sample $s$ into $p$ partitions of size $s/p$ each.
3. Cluster each partition independently using the hierarchical merging to $k$ clusters in each partition. The overall number $k \cdot p$ of clusters across all partitions is still larger than the user-desired target $k$.
4. Perform hierarchical clustering over the $k \cdot p$ clusters derived across all partitions to the user-desired target $k$.
5. Assign each of the $(ns)$ non-sample data points to the cluster containing the closest representative.

# High-Dimensional Clustering

- Many irrelevant features cause noise in the clustering process.
- Feature selection methods should be used.
- Feature selection may be impossible due to distance concentration phenomenon.



# High-Dimensional Clustering

- Projected Clustering
  - A cluster is defined as a set of points together with a subspace.
  - It is very problematic to define the cluster subspace using set of features.
  - E.g. arbitrarily oriented projected clusters, generalized projected clusters, correlation clusters.
- A local dimensionality reduction may be used.
  - Subspace clustering
  - Projected clustering

# High-Dimensional Clustering

- Subspace clustering
  - Cluster overlapping is allowed.
  - Hypercubes are mined from the data (similarly to the pattern mining or density-based clustering).
  - Density threshold is defined by user.
- Projected clustering
  - Cluster overlapping is not allowed.
  - A concise summary of the data is created.

## CLIQUE - CLustering In QUEst

- A generalization of the grid-based methods.
- The grid ranges $p$ for each dimension and density threshold $\tau$.
- The $\tau$ is the minimum number of points in a regions.
- Grid region are detected over a relevant subset of dimensions (traditional methods detect dense regions among all dimension).
- Each subset if taken as an item and the minimum support is set to $\tau$.
- Original algorithm used - Apriori algorithm.
- The adjacency is defined only between the same subset of dimensions.
- A Quantitative pattern mining method.
- An output is usually very large.

**Algorithm 7:** CLIQUE(Database: $D$, Ranges: $p$, Density: $\tau$)

```
1 begin
2     Discretize each dimension of data set D into p ranges;
3     Determine dense combinations of grid cells at minimum support τ using
        any frequent pattern mining algorithm;
4     Create graph in which dense grid combinations are connected if they are
        adjacent;
5     Determine connected components of graph;
6     return (point set,subspace) pair for each connected component
7 end
```

## PROCLUS - PROjected CLUStering

- A medoid-based clustering approach.
- A three phase algorithm:
  1. Initial phase
     - Select small candidate set M of medoids.
     - These medoids will restricts the Hill Climbing algorithm
  2. Iterative phase
     - Medoid based technique for hill climbing to better solution until convergence.
  3. Final phase
     - Data points are assigned to the relevant medoids
     - Outliers are removed.

## PROCLUS - PROjected CLUStering

- Initial phase may:
  - A random sample of M points is taken proportionally to the desired number of clusters.
  - The set is reduced using farthest distance approach.
  - This algorithm may incorporate many outliers, but also well separated seeds.
- Iterative phase:
  - A set of k is selected from the set M.
  - Bad clusters are iteratively replaced with new point from M.
  - Each medoid is associated with set of dimensions according to the statistical distribution of the data points. (based on locality).
  - The bad medoid is the medoid with least number of points.

## ORCLUS - The arbitrarily ORiented projected CLUStering

- The arbitrarily ORiented projected CLUStering.
- A method is able to find clusters that are not axis parallel (arbitrarily oriented).
- These clusters are also referred to as correlation clusters.
- The number of clusters and the dimensionality of each subspace as an input parameter.
- The k pairs $(C_i, E_i)$ are returned, clusters with relevant subspace.
- A combination of hierarchical and $k$-means clustering algorithm.

## ORCLUS - The arbitrarily ORiented projected CLUStering

- The algorithm merges the hierarchical representatives.
- The algorithm starts with k initial seeds $S$.
- The current number of seeds are reduced over successive merging iterations.
- The points are assigned to the seeds using representative-based alg. but within the associated subspace $E_i$.
- The idea behind is
  - During first iterations, the clusters may not necessarily correspond very well to the natural lower dimensional subspace clusters in the data.
  - In later iterations, the clusters are more refined, and therefore subspaces of lower rank may be extracted.

# Semi-supervised Clustering

- A wide variety of alternative solutions may be found by various algorithms.
- The quality of these alternative clusterings may be ranked differently by different internal validation criteria depending on the alignment between the clustering criterion and validation criterion.
- Semi-supervision relies on the external application-specific criteria to guide the clustering process.
- It is important to understand that different clusterings may not be equally useful from an application-specific perspective.
- The utility of a clustering result is, after all, based on the ability to use it effectively for a given application.

# Semi-supervised Clustering

- Point-wise supervision
  - Labels are associated with individual data points.
  - Very similar to the data classification.
- Pairwise supervision
  - Must-link and cannot-link constraints are provided for the individual data points.
  - Defined which points should be in the same cluster and which points should not.
  - Also known as constrained clustering.

# Semi-supervised Clustering - Point-wise supervision

- Soft supervision – points with different labels may mix in clusters.
- Hard supervision – points from different labels are not allowed to mix.
- Semisupervised clustering with seeding
  - Initial seeds for representative clustering is chosen form points with different labels.
  - A standard k-means algorithm is used.
  - A further modification for hard- supervision need to be provided.
  - In some cases, the weights are used while computing centroids.
- Agglomerative algorithms
  - Group-wise distance function incorporates the class label distribution among the clusters when soft-supervision is used.
  - Many different strategies for this “incorporation” was defined.
  - Hard-supervision is one among them.

# Semi-supervised Clustering - Pairwise supervision

- The constraints had to be carefully defined.
- The case Must(A-B), Must(B-C), Cannot(A-C) is problematic.
- K-means adaptation
  - The centroids are selected randomly.
  - The points are assigned in random order.
  - Each points is assigned to the closest centroid that does not violate any constraints.
  - When a points cannot be assigned the algorithm ends and the clusters from the last successful clustering is reported.
  - When no clustering is found at all, the first iteration is repeated several times.

# Human and Visually Supervised Clustering

- Replaces supervision incorporated into data by feedback from the user.
- Automatic cluster isolation is very difficult.
- Semantic feedback as an intermediate process in standard clustering algorithms.
  - A useful approach when objects are interpretable.
- Visual feedback in algorithms specifically designed for human-computer interaction.
  - User provide visual representation of the cluster structure of the data in different subsets of attributes.

# Human and Visually Supervised Clustering - Modification on standard algorithms

- Representative-based algorithms
  - User may review the cluster after each iteration
  - User may discard too small clusters.
  - User may merge close clusters. Centroid of removed cluster are replaced with the random one.
- Hierarchical algorithms
  - The user may select the final merged clusters from the top ranked choices.
  - This will reduce the noise sensitivity.

# Visual clustering

- A helpful scenario when semantic interpretability is difficult or impossible.
- A series of iterations
  - In each iteration a set of distinguishable set of points is determined.
  - A 2D subspace when de distance between randomly selected point is minimal is selected.
  - The subspace may be axis-parallel or arbitrary.
  - The cluster membership statistics is recorded based on user input.
  - The visualization is based on the density based approach.

# Clustering Ensembles

- A combination of different clustering algorithms may provide more information about real clustering ability of the data.
- The ensemble may be created in different ways:
  - A different models is combined (representative, hierarchical, density based).
  - The same modes with different settings.
  - Different subsets of points are selected.
  - Different subsets if dimension are selected.

# Clustering Ensembles

- Combination of the results from the components:
  - Hyper-graph partitioning
    - Each separate clustering create a complete subgraph from points in the same clusters.
    - Partitioning algorithm is used to separate the final clusters.
    - A balanced partitioning is forced using constraints.
  - Meta-clustering Algorithm
    - A graph based approach when nodes are associated to each cluster.
    - Each node represented a set of objects.
    - The edges are added between nodes when the Jaccard coefficient is non zero.
    - Nodes from the same clustering are not connected.
    - Partitioning algorithm is applied.
    - Points are assigned to the clusters according to the point membership to the clusters.

# Self-Organizing Maps

- The main objective of SOMs is to transform a complex high dimensional discrete input space into a simpler low-dimensional discrete output space by preserving the topology in the data but not the actual distances.
- It is an unsupervised learning algorithm which uses simple heuristic method capable of discovering hidden non-linear structure in high dimensional data.

# Self-Organizing Maps - Advantages

- They do not make assumptions regarding the distributions of variables nor do they require independence among variables.
- They are easier to implement and are able to solve non-linear problems of high complexity.
- They effectively cope with noisy and missing data, very small dimensional and samples of unlimited size

## Competition

- Output neurons in a SOM compete with each other to best represent the particular input sample.
- The success is measured using the a discriminant function = *input vector* is compared with the weight vector of each node.
- Neuron with its weights most similar to the input is declared the winner of the competition.
- There are a number of different functions that can be used to determine the best matching unit (BMU) (the winning neuron).
- The most most commonly used one is the Euclidean distance.

# Self-Organizing Maps - Principles

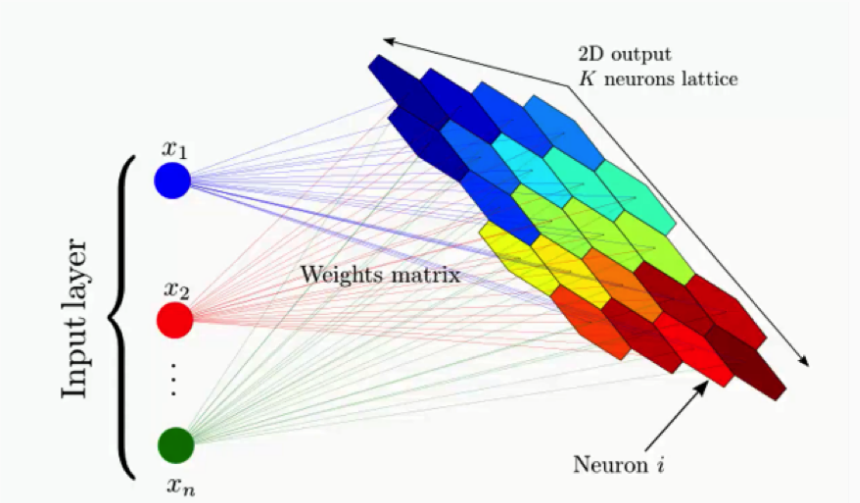
## Co-operation

- Self-Organizing Maps is a topographic organization in which nearby locations in the output space represents inputs with similar properties.

## Adaptation

- The weight vectors of the BMU and its neighboring units in the map are adjusted in favor of the higher values of the discriminant function.

# Self-Organizing Maps - Structure




https://www.superdatascience.com/blogs/the-ultimate-guide-to-self-organizing-maps-soms

# Self-Organizing Maps - Algorithm

- Input is a $d$-dimensional vector $x = (x_1, x_2, \ldots, x_d)$
- Output is a *Kohonen layer* - neurons usually arranged into 2D grid of rectangular or hexagonal units.
- Each neuron has $d$ dimensional weight vector $w_i = (w_{i1}, w_{i2}, \ldots, w_{id})$.
- The number of nodes in the output denotes the maximum number of clusters and influences the accuracy and generalization capability of the SOM.

# Self-Organizing Maps - Algorithm

- Initial weights of the neurons are initialized.
- For each input vector $x$ a similarity to each neuron is computed:

$$d_i(x) = \|x - w_i\|$$

- Best matching unit is selected based on the minimal distance/maximal similarity:

$$bmu(x) = \underset{i}{\operatorname{argmin}} \left(d_i\left(x\right)\right)$$

# Self-Organizing Maps - Algorithm

- The weight vectors of the BMU and its neighboring units in the output layer are adjusted to become more representative of the features that characterize the input space.

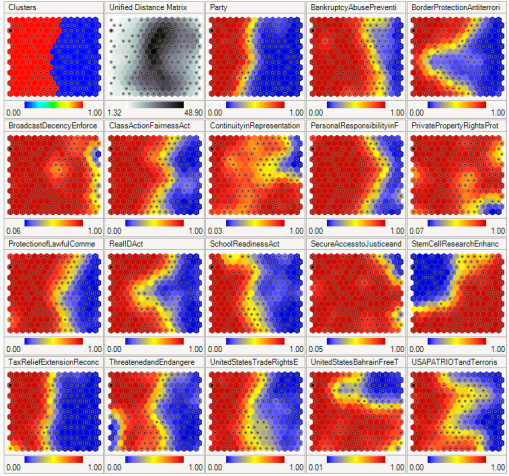
$$w_i(t+1) = w_i(t) + \alpha(t)h(t)\left[x - w_i(t)\right]$$

- $\alpha(t)$ is the learning rate, that may change over time $t$.
- $h(t)$ is spatio-temporal decay functions defined as:

$$h(t) = \exp\left(-\frac{dist(bmu, n)}{2\sigma^2(t)}\right)$$

- $dist(bmu, n_i)$ is the lateral (spacial) distance between $bmu$ neuron and neuron $n_i$.
- $\sigma(t)$ is the effective width/radius of the neighborhood as time $t$.

# Self-Organizing Maps



Questions?