Support vector clustering: An Evolutionary algorithm

Pradeep Jha[1],Krishan Kant Lavania [2],Surendar Sharma[3]

Arya Institute of Engineering and technology, Jaipur, Rajasthan

pradeep.jha1988@gmail.com [1],k@lavania.in [2],surendar.sharma@aryaaiet.ac.in [3]

Abstract:-

This paper review of Support Vector Clustering (SVC), which is inspired by the support vector machines, can overcome the limitation of clustering algorithms. SVC algorithm has two main steps[1]. a) SVM Training and b) Cluster Labeling .SVM training step involves construction of cluster boundaries and cluster labeling step involves assigning the cluster labels to each data point. Solving the optimization problem and cluster labeling is time consuming in the SVC training procedure. Many of the research efforts have been taken to improve the efficiency of cluster labeling step. Preprocessing procedures used for SVC to reduce SVC training set are Heuristics for Redundant-point Elimination (HRE) and Shared Nearest Neighbor (SNN) technique result in loss of data Due to fewer efforts taken by researchers to reduce execution time and accuracy of SVC training procedure.

***Keywords*- SVC, SNN, HRE**

1. Introduction to clustering

Cluster analysis is the organization of a collection of patterns (usually represented as a vector of measurements, or a point in a multidimensional space) into clusters based on similarity. Intuitively, patterns within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster [4]. An example of clustering is depicted in Figure 1.1

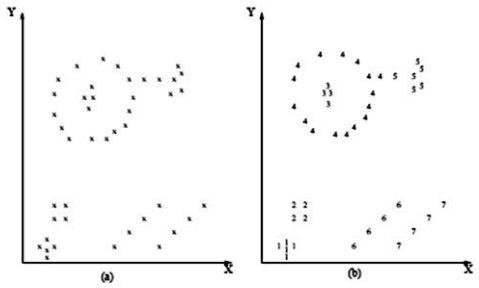


Figure 1: Data Clustering [4].

The input patterns are shown in Figure 1(a), and the desired clusters are shown in Figure 1(b). Here, points belonging to the same cluster are given the same label. The variety of techniques for representing data, measuring proximity (similarity) between data elements, and grouping data elements has produced a rich and often confusing assortment of clustering methods [4].

It is important to understand the difference between clustering unsupervised classification) and discriminant analysis (supervised classification). In supervised classification, we are provided with a collection of *labeled* (preclassified) patterns; the problem is to label a newly encountered, yet unlabeled, pattern. Typically, the given labeled (training) patterns are used to learn the descriptions of classes which in turn are used to label a new pattern. In the case of clustering, the problem is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are *data driven*; that is, they are obtained solely from the data. Clustering is useful in several exploratory pattern-analysis, grouping, decision-making, and machine-learning situations, including data mining, document retrieval, image segmentation, and pattern classification. However, in many such problems, there is little prior information (e.g., statistical models) available about the data, and the decision-maker must make as few assumptions about the data as possible. It is under these

restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data points to make an assessment (perhaps preliminary) of their structure [4].

Clustering has always been a tricky task in pattern classification. Many clustering algorithms have been proposed in the past years. Division of patterns, data items, and feature vectors into groups (clusters) is a complicated task since clustering does not presume any prior knowledge, which are the clusters to be searched for. There exist no class label attributes that would tell which classes exist. Some of the traditional clustering techniques are.

a) Hierarchical clustering algorithms
b) Partitional clustering algorithms
c) Nearest neighbor clustering
d) Fuzzy clustering.

Clustering algorithms are capable of finding clusters with different shapes, sizes, densities, and even in the presence of noise and outliers in datasets. Although these algorithms can handle clusters with different shapes, they still cannot produce arbitrary cluster boundaries to adequately capture or represent the characteristics of clusters in the dataset [5].

2. Components of a Clustering Task

Typical pattern clustering activity involves the following steps [4]:

1. Pattern representation (optionally including feature extraction and/or selection),
2. Definition of a pattern proximity measure appropriate to the data,
3. Clustering or grouping,
4. Data abstraction (if needed), and
5. Assessment of output (if needed).

Figure 2 [4] depicts a typical sequencing of the first three of these steps, including a feedback path where the grouping process output could affect subsequent feature extraction and similarity computations.

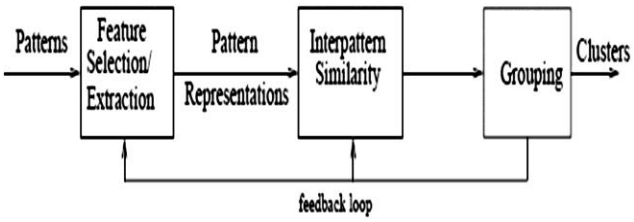


Figure 2: Stages in Clustering.

Pattern representation: It refers to the number of classes, the number of available patterns, and the number, type, and scale of the features available to the clustering algorithm. Some of this information may not be controllable by the practitioner. Feature selection is the process of identifying the most effectively subset of the original features to use in clustering. Feature extraction is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain an appropriate set of features to use in clustering.

Pattern proximity: It is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in the various communities [5]. A simple distance measure like Euclidean distance can often be used to reflect dissimilarity between two patterns, whereas other similarity measures can be used to characterize the conceptual similarity between patterns [5].

Clustering: The grouping step can be performed in a number of ways. The output clustering (or clustering's) can be hard (a partition of the data into groups) or fuzzy (where each pattern has a variable degree of membership in each of the output clusters). Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity. Partitional clustering algorithms identify the partition that optimizes (usually locally) a clustering criterion. Additional techniques for the grouping operation include probabilistic [4] and graph-theoretic [4] clustering methods.

Data abstraction: It is the process of extracting a simple and compact representation of a data set. Here, simplicity is either from the perspective of automatic analysis (so that a machine can perform further processing efficiently) or it is human-oriented (so that the representation obtained is easy to comprehend and intuitively appealing). In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster

prototypes or representative patterns such as the centroid [4]. How is the output of a clustering algorithm evaluated? What characterizes a 'good' clustering result and a 'poor' one? All clustering algorithms will, when presented with data, produce clusters-regardless of whether the data contain clusters or not. If the data does contain clusters, some clustering algorithms may obtain 'better' clusters than others.

Assessment of output: The assessment of a clustering procedure's output, then, has several facets. One is actually an assessment of the data domain rather than the clustering algorithm itself-data which do not contain clusters should not be processed by a clustering algorithm. The study of cluster tendency, wherein the input data are examined to see if there is any merit to a cluster analysis prior to one being performed, is a relatively inactive research area, and will not be considered further in this survey Cluster validity analysis, by contrast, is the assessment of a clustering procedure's output. Often this analysis uses a specific criterion of optimality; however, these criteria are usually arrived at subjectively. Hence, little in the way of 'gold standards' exist in clustering except in well-prescribed sub domains. Validity assessments are objective [4] and are performed to determine whether the output is meaningful. A clustering structure is valid if it cannot reasonably have occurred by chance or as an artifact of a clustering algorithm. When statistical approaches to clustering are used, validation is accomplished by carefully applying statistical methods and testing hypotheses. There are three types of validation studies. An external assessment of validity compares the recovered structure to an *a* priori structure. An internal examination of validity tries to determine if the structure is intrinsically appropriate for the data. A relative test compares two structures and measures their relative merit.

3. Hierarchical Clustering Algorithms

The operation of a hierarchical clustering algorithm is illustrated using the two-dimensional data set in Figure 3 [4]. This figure depicts seven patterns labeled A, B, C, D, E, F, and G in three clusters. A hierarchical algorithm yields a dendrogram representing the nested grouping of patterns and similarity levels at which groupings change. A dendrogram corresponding to the seven points in Figure 3 (obtained from the single-link algorithm) is shown in Figure.4 [4].

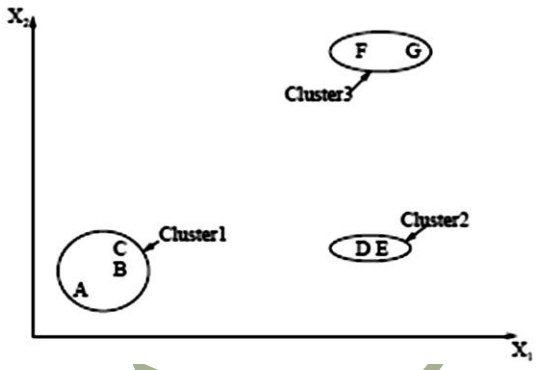


Figure 3: Points falling in three clusters.

The dendrogram can be broken at different levels to yield different clustering's of the data. Most hierarchical clustering algorithms are variants of the single-link, complete-link, and minimum-variance algorithms. Of these, the single-link and complete link algorithms are most popular. These two algorithms differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pair wise distances between patterns in the two clusters. In either case, two clusters are merged to form a larger cluster based on minimum distance criteria [4].

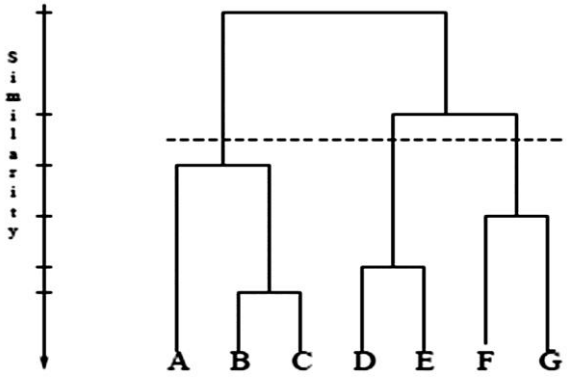


Figure 4: The dendrogram obtained using the single-link algorithm

4. Partitional Algorithms

A partitional clustering algorithm obtains a single partition of the data instead of a clustering structure, such as the dendrogram produced by a hierarchical technique. Partitional methods have advantages in applications involving large data sets for which the construction of a dendrogram is computationally prohibitive. A problem accompanying the use of a Partitional algorithm is the choice of the number of desired output clusters. A seminal paper [4] provides guidance on this key design decision. The Partitional techniques usually produce clusters by optimizing a criterion function defined either locally (on a subset of the patterns) or globally (defined over all of the patterns). Combinatorial search of the set of possible labeling for an optimum value of a criterion is clearly computationally prohibitive. In practice, therefore, the algorithm is typically run multiple times with different starting states, and the best configuration obtained from all of the runs is used as the output clustering.

5. Squared Error Algorithms

The most intuitive and frequently used criterion function in partitional clustering techniques is the squared error criterion, which tends to work well with isolated and compact clusters. The *k*-means is the simplest and most commonly used algorithm employing a squared error criterion [4]. It starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers until a convergence criterion is met (e.g., there is no reassignment of any pattern from one cluster to another, or the squared error ceases to decrease significantly after some number of iterations). The *k*-means algorithm is popular because it is easy to implement, and its time complexity is $O(n)$, where n is the number of patterns. A major problem with this algorithm is that it is sensitive to the selection of the initial partition and may converge to a local minimum of the criterion function value if the initial partition is not properly chosen.

Squared Error Clustering Method

(1) Select an initial partition of the patterns with a fixed number of clusters and cluster centers.

(2) Assign each pattern to its closest cluster center and compute the new cluster centers as the centroids of the clusters. Repeat this step until convergence is achieved, i.e., until the cluster membership is stable.

(3) Merge and split clusters based on some heuristic information, optionally repeating step 2 [4].

***k*-Means Clustering Algorithm**

(1) Choose *k* cluster centers to coincide with *k* randomly-chosen patterns or *k* randomly defined points inside the hypervolume containing the pattern set.

(2) Assign each pattern to the closest cluster center.

(3) Recompute the cluster centers using the current cluster memberships.

(4) If a convergence criterion is not met, go to step 2. Typical convergence criteria are: no (or minimal) reassignment of patterns to new cluster centers, or minimal decrease in squared error [4].

Several variants [4] of the *k*-means algorithm have been reported in the literature. Some of them attempt to select a good initial partition so that the algorithm is more likely to find the global minimum value.

6. Nearest Neighbor Clustering

Since proximity plays a key role in our intuitive notion of a cluster, nearest neighbor distances can serve as the basis of clustering procedures. An iterative procedure was proposed in Lu and Fu; it assigns each unlabeled pattern to the cluster of its nearest labeled neighbor pattern, provided the distance to that labeled neighbor is below a threshold. The process continues until all patterns are labeled or no additional labeling occur. The mutual neighborhood value (described earlier in the context of distance computation) can also be used to grow clusters from near neighbors [4].

7. Fuzzy Clustering

Traditional clustering approaches generate partitions; in a partition, each pattern belongs to one and only one cluster. Hence, the clusters in a hard clustering are disjoint. Fuzzy clustering extends this notion to associate each pattern with every cluster using a membership function. The output of such algorithms is a clustering, but not a partition [4].

8. Support Vector Machine

Support Vector Machine is supervised Machine Learning technique. Support Vector Machine (SVM) was first introduced in 1992 by Boser, Guyon, and Vapnik [41]. Support Vector Machines (SVMs) are a set of related supervised learning methods used for classification and regression [2]. They belong to a family of generalized linear classifiers. In another terms, Support Vector Machine (SVM) is a classification and regression prediction tool that uses Machine Learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. Support Vector machines can be defined as systems which use hypothesis space of a linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. Support vector machine was initially popular with the Neural Information Processing Systems (NIPS) community and now is an active part of the Machine Learning research around the world. SVM becomes famous when, using pixel maps as input; it gives accuracy comparable to sophisticated neural networks with elaborated features in a handwriting recognition task. It is also being used for many applications, such as hand writing analysis, face analysis and so forth, especially for pattern classification and regression based applications. SVMs were developed to solve the classification problem, but recently they have been extended to solve regression problems

9. Support Vector Clustering (SVC)

Support Vector Clustering was first introduced in 2000 by Asa Ben-Hur, David Horn, Hava T. Siegelmann and Vladimir Vapnik [5]. The support vector clustering (SVC) algorithm is inspired by the support vector machines and solves a global optimization problem by turning the Lagrangian into the dual quadratic form [1, 9].

The main objective is to find the smallest enclosing hypersphere in the transformed high-dimensional feature space that contains most of the data points. The original input space can always be mapped to some higher-dimensional feature space where the training set is separable. The hypersphere is then mapped back to the original data space to form a set of contours, which are regarded as the cluster boundaries in the original data space. The SVC algorithm includes two key steps: SVM training and cluster labeling. The former determines the hypersphere construction and the distance definition from a point's image in the feature space to the hypersphere center. The latter aims to assign each data point to its corresponding cluster [1].

10. Existing Support Vector Clustering Techniques

Support Vector Clustering (SVC) involves following steps [2]: It is shown in following figure 5.

Data Preprocessing: Eliminates insignificant points and gives reduced training set.

1. Kernel-parameter Tuning: Gives the value of (C, q).
2. Optimization using SMO Algorithm: Solving dual for Lagrange multipliers.
3. Cluster Labeling: Labeling the data points with cluster labels.

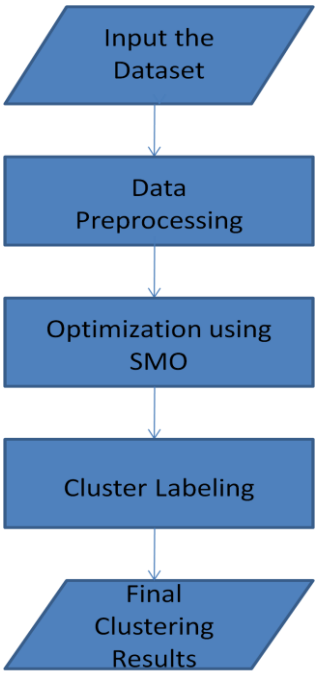


Figure 5: Flowchart of the SVC Procedure [2].

The SVC procedure which is inspired by the support vector machine technique includes two key steps: SVM training and cluster labeling, which are summarized in figure .6 [1]. The former determines the hypersphere construction and the distance definition from a point's image in the feature space to the hypersphere center. The latter aims to assign each data point to its corresponding cluster. The labeling of each data point with its corresponding cluster is known as cluster labeling.

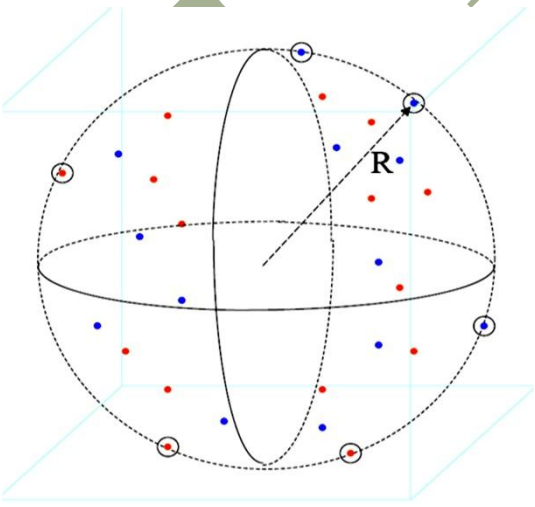


Figure 6: Feature Space: The Sphere [40].

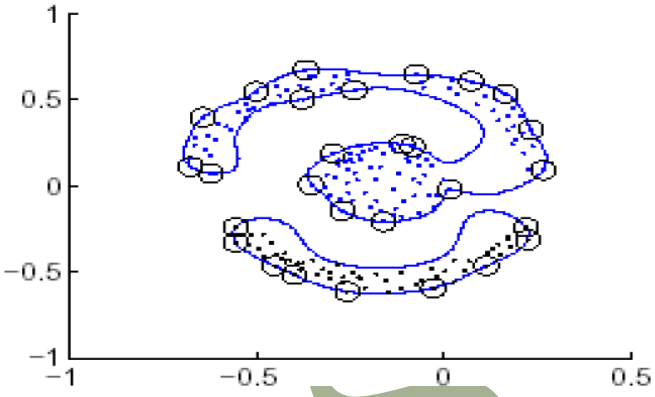


Figure 7: Data Space [7].

Given a set of data points $x_j \in \Re^d$, j = 1,…, N and a nonlinear mapping function $\Phi : \Re^d \rightarrow F$, the objective is to find a hypershpere with the minimal radius R, such as

$$\| \Phi(x_j) - \alpha \|^2 \leq R^2 + \xi_j ,$$

where α is the center of the hypersphere and $\xi_j \geq 0$ are the slack variables allowing soft constraints. The primal problem is solved in its dual form by introducing the Lagrangian

$$L = R^2 - \Sigma_j (R^2 + \xi_j - \| \Phi(x_j) - \alpha \|^2)\beta_j - \Sigma_j \xi_j \mu_j + C \Sigma_j \xi_j,$$

where $\beta_j \geq 0$ and $\mu_j \geq 0$ are Lagrange multipliers and $C \Sigma_j \xi_j$ is a penalty term with C as a regularization constant [1, 9]. The dual form of the constrained optimization is constructed as

$$\max W = \Sigma_j \Phi(x_j) - \Sigma_{i,j} \beta_i \beta_j \Phi(x_i) \cdot \Phi(x_j),$$

subject to the constraints:

(1) $0 \leq \beta_j \leq C$,

(2) $\Sigma_j \beta_j = 1$ for j=1, 2, N

Using the kernel representation $k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, Eq. (3) is rewritten as

$$\max W = \Sigma_j k(x_j, x_j)\beta_j - \Sigma_{i,j} \beta_i \beta_j k(x_i, x_j), \text{ ---------- } (6)$$

The Gaussian kernel $k(x_i, x_j) = e^{-q\|x_i - x_j\|2}$ is usually used for SVC algorithms, while polynomial kernels do not generate tight contour representations of clusters [1, 9].

Furthermore, for each data point x, the distance of $\Phi(x)$ to the center is calculated as

$$R^2(x) = \|\Phi(x) - \alpha\|^2 = k(x, x) - 2\,\Sigma_j \beta_j k(x_j, x) - \Sigma_{i,j} \beta_i \beta_j k(x_i, x_j)$$

The points that lie on the cluster boundaries are defined as support vectors (SVs), which satisfy the conditions $\xi j = 0$ and $0 < \beta_j < C$. The points with $\xi_j > 0$ and $\beta_j = C$ lie outside the boundaries and are called bounded support vectors (BSVs). The rest of the data points lie inside the clusters. It is shown in figure 1.7 [40]. Note that the increase of the Gaussian kernel width parameter σ can increase the number of SVs, therefore causing the contours to change shape. By iteratively decreasing (increasing) σ from a certain large (small) value, SVC can form agglomerative (divisive) hierarchical clusters [1].

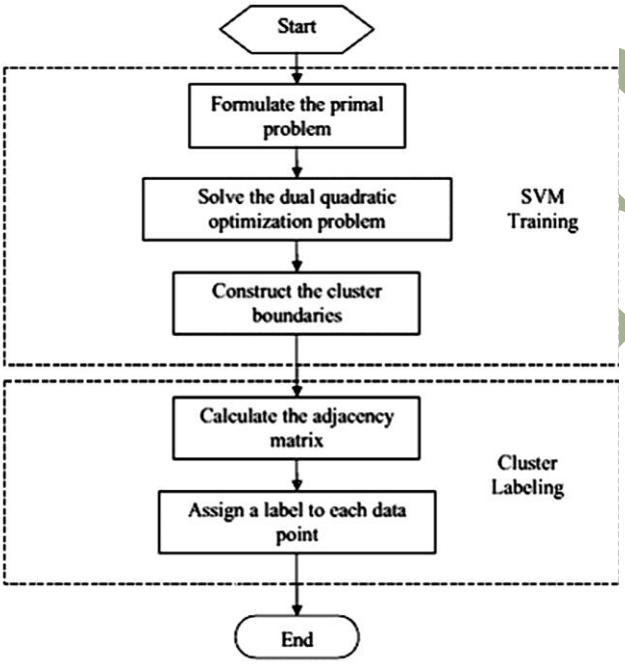


Figure 8: Flowchart of SVC algorithm [1].

The data points are clustered together according to the adjacency matrix A, which is based on the observation that any corresponding path in the feature space, which connects a pair of data points belonging to different clusters, must exit from the hypersphere. Given each pair of x_i and x_j, their adjacency value is defined as

$$A_{ij} = 1, \text{ if } R(x_i + \gamma(x_j - x_i)) \le R, \gamma \in [0, 1] \quad \text{---------(8)}$$

0, otherwise.

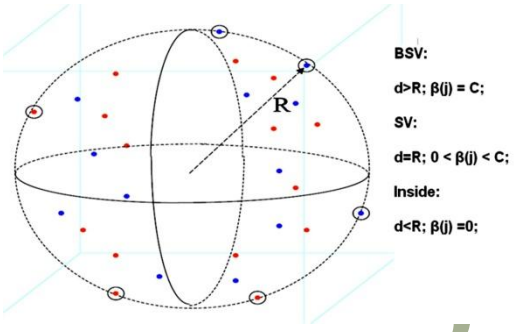
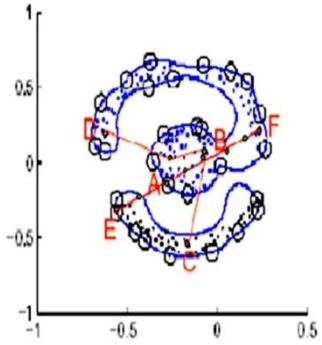


Figure 9: BSVs, SVs and Internal data points [7].

The number of sampling points for each edge between two data points is usually around 20 [5]. It is shown in figure 8 [7]. The overall computational complexity of the labeling step is $O(N^2)$, which becomes a critical issue for large-scale data sets. Ben-Hur et al. [5] suggested a heuristic that only calculates the adjacency value between support vectors to lower the time complexity to $O((N-N_{bsv})N^2_{sv})$, where N_{bsv} is the number of BSVs and N_{sv} is the number of SVs. However, this heuristic still has quadratic complexity when N_{sv} is greater than 0.05N − 0.1N [5].



$$A_{ij} = \begin{cases} 1 & \text{if, for all } y \text{ on the line segment connecting } x_i \text{ and } x_j,\ R(y) \le R \\ 0 & \text{otherwise} \end{cases}$$

Figure 10: Cluster Analysis: Adjacency matrix [7].

11. Conclusion

Support Vector Clustering is one of the techniques in pattern recognition. Support Vector Clustering is Kernel-Based Clustering. Division of patterns, data items, and feature vectors into groups (clusters) is a complicated task since clustering does not assume any prior knowledge, which are the clusters to be searched for. There exist no class label attributes that would tell which classes exist. Thus clustering serves in particular for exploratory data analysis with little or no prior knowledge. Some of the traditional clustering techniques are Hierarchical clustering algorithms, Partitional clustering algorithms, Nearest neighbor clustering, and Fuzzy clustering.

12. References

[1] R. X. Donald, C. Wunsch, "Clustering", IEEE Press Series on Computational Intelligence, 2009, pp. 172-187.

[2] Fedrik Gran, "Pattern recognition using Support Vector Machine", A Master Thesis, Matematikcentrun, LTH, 2002, pp. 5-24.

[3] J. S. Wang, J. C. Chiang, "An Efficient Data Preprocessing Procedure for Support Vector Clustering", Journal of Universal Computer Science, 2009, pp. 705-721.

[4] A. Jain, M. Murty, P. Flynn, "Data Clustering: A Review", ACM Computing Surveys,1999, pp. 264-323.

[5] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, "Support Vector Clustering", Journal of Machine Learning Research 2, 2001, pp. 125-137.

[6] A. K. Jain and R. C. Dubes, "Algorithms for clustering data", Prentice Hall, Englewood Cliffs, NJ, 1988.

[7] Dennis Decoste, Bernhard Scholkopf, "Training Invariant Support Vector Machines", Kluwer Academic Publishers, Netherlands, Machine Learning, 46,161-190, 2002.