Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 4, April 2013, pg.155 – 159

SURVEY ARTICLE

A Survey on Various Clustering Techniques with K-means Clustering Algorithm in Detail

Supreet Kaur[1], Usvir Kaur[2]

[1]Student of masters of technology Computer Science, Department of Computer Science Engineering, Sri Guru Granth Sahib World University, Fatehgarh Sahib, Punjab, India

[2]Assistant Professor, Department of Computer Science and Engineering, Sri Guru Granth Sahib World University, Fatehgarh Sahib, Punjab, India

[1] Preet_inder88@yahoo.co.in; [2] Usvirkaur@gmail.com

Abstract— Clustering is the division of data into groups of similar objects. In clustering, some details are disregarded in exchange for data simplification. Clustering can be viewed as a data modeling technique that provides for concise summaries of the data. Clustering is therefore related to many disciplines and plays an important role in a broad range of applications. The applications of clustering usually deal with large datasets and data with many attributes. Exploration of such data is a subject of data mining. This survey concentrates on clustering algorithms from a data mining perspective with K means Clustering algo.

Key Terms: - Clustering; types; Froggy Algorithm; k-means; algo

I. INTRODUCTION [9]

We provide a comprehensive review of different clustering techniques in data mining. Clustering refers to the division of data into groups of similar objects. Each group, or cluster, consists of objects that are similar to one another and dissimilar to objects in other groups. When representing a quantity of data with a relatively small number of clusters, we achieve some simplification, at the price of some loss of detail (as in lossy data compression, for example).Clustering is a form of data modeling, which puts it in a historical perspective rooted in mathematics and statistics. From a machine learning perspective, clusters correspond to hidden patterns, the search for clusters is unsupervised learning, and the resulting system represents a data concept. Therefore, clustering is unsupervised learning of a hidden data concept. Clustering as applied to data mining applications encounters three additional complications:

(a) large databases,
(b) objects with many attributes, and
(c) Attributes of different types. These complications tend to impose severe computational requirements that present real challenges to classic clustering algorithm.

II. IMPORTANT ISSUES [4]

The properties of clustering algorithms of concern in data mining include:

- Type of attributes an algorithm can handle
- Scalability to large datasets
- Ability to work with high-dimensional data

- Ability to find clusters of irregular shape
- Handling outliers
- Time complexity (we often simply use the term *complexity*)
- Data order dependency
- Labeling or assignment (hard or strict vs. soft or fuzzy)
- Reliance on a priori knowledge and user-defined parameters
- Interpretability of results

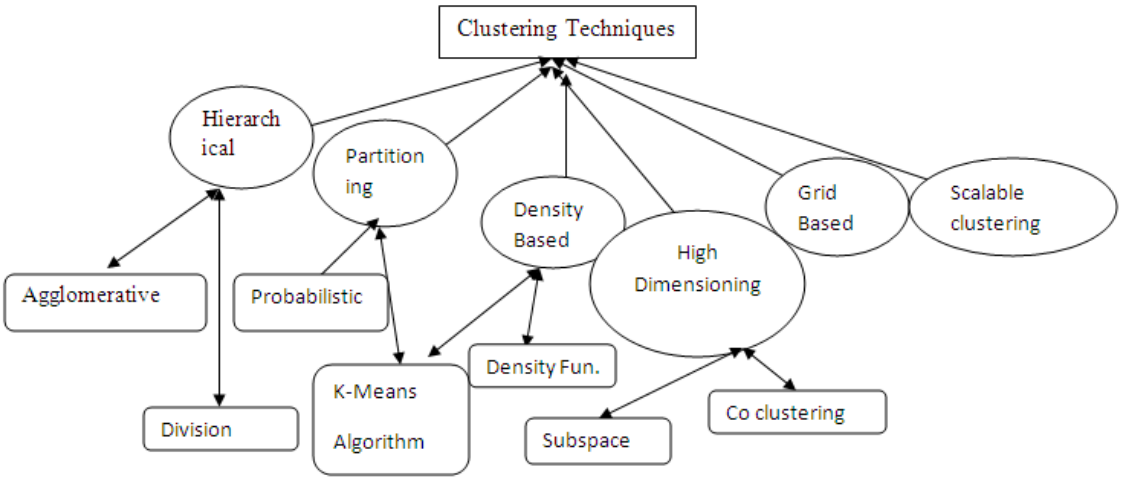


Fig.1 represents various Clustering techniques

- ***Hierarchical methods***-Agglomerative algorithms**,** Divisive algorithms
- ***Partitioning relocation methods***-Probabilistic clustering**,** *k*-medoids methods**,** *k*-means methods
- **Density-based partitioning methods**-Density-based connectivity clustering Density functions clustering
- **Grid-based methods**
- **Methods based on co-occurrence of categorical data**
- **Other clustering techniques** Constraint-based clustering**,** Graph partitioning**,** Clustering algorithms and supervised learning, Clustering algorithms in machine learning
- **Scalable clustering algorithms**
- **Algorithms for high-dimensional data**-Subspace clustering, Co clustering technique

III. GENERAL DISCUSSION ON K-MEANS METHODS [9]

The *k*-means algorithm is by far the most popular clustering tool used nowadays in scientific and industrial applications. The name comes from representing each of the *k* clusters *Cj* by the mean (or weighted average) *cj* of its points, the so-called *centroid*. While this representation does not work well with categorical attributes, it makes good sense from a geometrical and statistical perspective for numerical attributes. The sum of distances between elements of a set of points and its centroid expressed through an appropriate distance function is used as the objective function. For example, the *L*2norm-based objective function, the sum of the squares of errors between the points and the corresponding centroids, is equal to the total intracluster variance

$$E(C) = \sum_{\substack{0 \le i \le m \\ 0 < j < n}} P(i,j) \, \|pi - pj\|$$

The sum of the squares of errors (SSE) can be regarded as the negative of the log-likelihood for a normally distributed mixture model and is widely used in statistics. Therefore, the *k*-means algorithm can be derived from a general probabilistic framework. Note that only means are estimated. A simple modification would normalize individual errors by cluster radii (cluster standard deviation), which makes a lot of sense when clusters have different dispersions. An objective function based on the *L*2 norm has many unique algebraic properties. For example, it coincides with pair wise errors,

$$E(C)' = 1/2 \sum_{0 \le i \le m} \sum_{0 < j < n} P(i,j) \| pi \quad pj \|$$

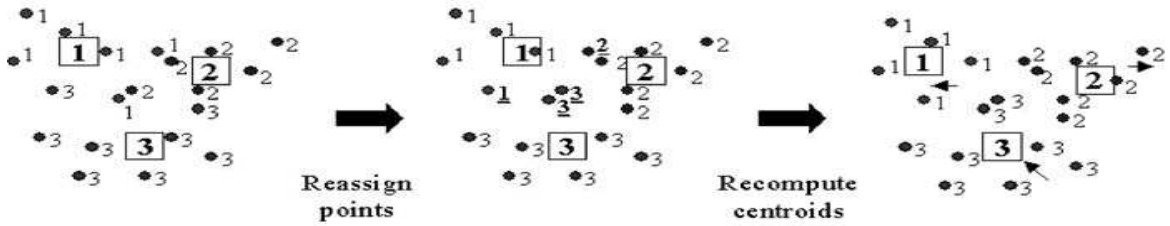
and with the difference between the total data variance and the inter cluster variance. Therefore, cluster separation and cluster tightness are achieved simultaneously. Two versions of *k*-means iterative optimization are known. The first version is similar to the EM algorithm and consists of two-step major iterations that:

(1) Reassign all the points to their nearest centroids, and (2) recomputed centroids of newly assembled groups. Iterations continue until a stopping criterionis achieved (for example, no reassignments happen). This version, known as Forgy's algorithm, has many advantages:

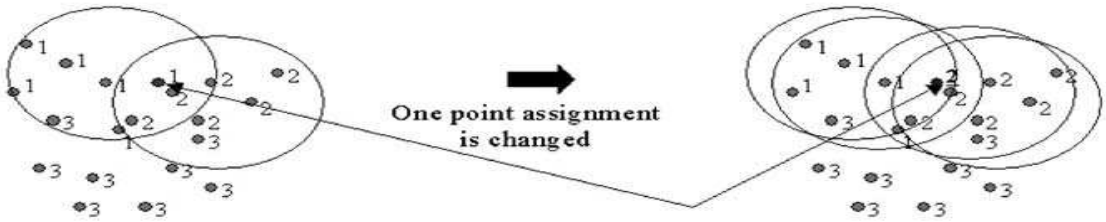
- It easily works with any *Lp* norm,
- It allows straightforward parallelization.
- It does not depend on to data ordering.

The second (classic in iterative optimization) version of *k*-means reassigns points based on a detailed analysis of how moving a point from its current cluster to any other cluster would affect the objective function. If a move has a positive effect, the point is relocated and the two centroids are recomputed. It is not clear that this version is computationally feasible, because the outlined analysis requires an inner loop over all member points of involved clusters affected by centroids shifts. However, in the *L*2 case it is known from that computing the impact on a potential cluster can be algebraically reduced to finding a single distance from its centroid to a point in question. Therefore, in this case both versions have the same computational complexity.

There is experimental evidence that compared with Forgy's algorithm; the second (classic) version frequently yields better results. In particular, Dhillon et al. [4] noticed that a Forgy's spherical *k*-means (using cosine similarity instead of Euclidean distance) has a tendency to get stuck when applied to document collections. They noticed that a version that reassigned



Two-step major iterations (Forgy's algorithm)



Iterative optimization (with centroid recomputation)

Fig.2 K-means Algo [9]

points and immediately recomputed centroids works much better. Figure 2 illustrates both implementations. Besides these two versions, there have been other attempts to find better *k*-means objective functions. For example, the early algorithm ISODATA [4] used merges and splits of intermediate clusters.

The popularity of the *k*-means algorithm is well deserved, since it is easily understood, easily implemented, and based on the firm foundation of analysis of variances. The *k*-means algorithm also has certain shortcomings:

(1) The result depends greatly on the initial guess of centroids

(*2)* The computed local optimum may be quite different from the global one, • It is not obvious how to choose a good value for *k*,

The process is sensitive to outliers,

- The basic algorithm is not scalable,
- Only numerical attributes are covered,
- Resulting clusters can be unbalanced (in Forgy's version, even empty). A simple way to mitigate the effects of cluster initialization was suggested by Bradley and Fayyad . First, *k*-means is performed on several small samples of data with a random initial guess. Centroids of the best system constructed this way are suggested as an intelligent initial guess to ignite the *k*-means algorithm on the full data. Another interesting attempt is based on genetic algorithms, as discussed later. No initialization actually guarantees a global minimum for *k*-means. This is a general problem in combinatorial optimization, which is usually tackled by allowing uphill movements. In our context, simulated annealing was suggested in. Zhang[2] suggested another way to rectify the optimization process by soft assignment of points to different clusters with appropriate weights (as EM does), rather than moving them decisively from one cluster to another. The weights take into account how well a point fits into the recipient cluster. This process involves the so-called *harmonic means*. In this regard, we wish to clarify that the EM algorithm makes soft (fractional) assignments, while the reassignment step in Forgy's version exercises "winner-take-all" or hard assignment. A brilliant earlier analysis of where this subtle difference leads has been conducted by Kearns et al.

For a thorough treatment of *k*-means scalability, see Bradley et al.'s excellent Study. A generic method to achieve scalability is to preprocess or *squash* the data. Such preprocessing usually also takes care of outliers. Preprocessing has drawbacks. It results in approximations that sometimes negatively affect final cluster quality. Pelleg and Moore suggested how to directly (without any squashing) accelerate the *k*-means iterative process by utilizing *KD* trees. The algorithm *X*-means goes a step further: in addition to accelerating the iterative process, it tries to incorporate a search for the best *k* in the process itself. While more comprehensive criteria for finding optimal *k* require running independent *k*-means and then comparing the results (costly experimentation), *X*-means tries to split a part of the already constructed cluster based on the outcome of the BIC criterion. This gives a much better initial guess for the next iteration and covers a user specified range of admissible *k*. The tremendous popularity of *k*-means algorithm has brought to life many other extensions and modifications. Mahalanobis distance can be used to cover hyper ellipsoidal clusters. The maximum of intra cluster variances, instead of the sum, can serve as an objective function. Generalizations that incorporate categorical attributes are also known: the term *prototype* is used in this context instead of the term *centroid*.

IV. CONCLUSION

Clustering algorithms share some important common issues that need to be addressed to make them successful. Some issues are so ubiquitous that they are not even specific to unsupervised learning and can be considered as a part of an overall data mining framework. Other issues are resolved in certain algorithms we presented. In fact, many algorithms were specifically designed to address some of these issues and k-means is focused on these ISSUES which can be addressed in Next research.

- Assessment of results,
- Choice of appropriate number of clusters,
- Data preparation,
- Proximity measures,
- Handling outliers

REFERENCES

[1] N. Duhan, A. K. Sharma and Bhatia K. K., "Page Ranking Algorithms: A Survey, Proceedings of the IEEE International Conference on Advance Computing, 2009, 978-1-4244-1888-6.

[2] S. Brin, and Page L., "The Anatomy of a Large Scale Hypertextual Web Search Engine", Computer Network and ISDN Systems, Vol. 30, Issue 1-7, pp. 107-117, 1998.

[3] Larry Page, and Sergey Brin, Rajeev Motwani, Terry Winograd, "The PageRank Citation Ranking: Bring Order to the Web", Technical report in Stanford U, 1998.

[4] R. Cooley, B. Mobasher, and Srivastava, J., "Web Mining: Information and pattern discovery on the World Wide Web". In proceedings of the 9th IEEE International Conference on tools with Artificial Intelligence (ICTAI' 97).Newposrt Beach,CA 1997.

[5] J. Kleinberg, "Hubs, Authorities and Communities", ACM Computing Surveys, 31(4), 1999.

[6] Gyanendra Kumar, Neelam Duahn, and Sharma A. K., "Page Ranking Based on Number of Visits of Web Pages", International Conference on Computer & Communication Technology (ICCCT)-2011, 978-1-4577-1385-9.
[7] R. Kosala, and H. Blockeel, "Web Mining Research: A Survey", SIGKDD Explorations, Newsletter of the ACM Special Interest Group on Knowledge Discovery and Data Mining Vol. 2, No. 1 pp 1-15, 2000.
[8] Wenpu Xing and Ghorbani Ali, "Weighted PageRank Algorithm", Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR '04), IEEE, 2004.
[9] P. Berkhin," A Survey of Clustering Data Mining Techniques", A Book Journal Published in 2008-09.

