© International Journal of Combinatorial Optimization Problems and Informatics, Vol. 9, No. 2, May-Aug 2018, pp. 3-10. ISSN: 2007-1558.

A-means: improving the cluster assignment phase of k-means for Big Data

Joaquín Pérez Ortega[1*], Nelva Nely Almanza Ortega[1], Jorge A. Ruiz-Vanoye[2], Rodolfo A. Pazos R.[3], Socorro Sáenz Sánchez[1], José María Rodríguez Lelis[1], Alicia Martínez Rebollar[1]
[1] Tecnológico Nacional de México/CENIDET
[2] Universidad Autónoma del Estado de Hidalgo, México
[3] Tecnológico Nacional de México/ITCM
jpo_cenidet@yahoo.com.mx

Abstract. This paper proposes a new criterion for reducing the processing time of the assignment of data points to clusters for algorithms of the k-means family, when they are applied to instances where the number *n* of points is large. Our criterion allows a point to be classified in an early stage, excluding it from distance calculations to cluster centroids in subsequent iterations. The proposed criterion uses knowledge of the distance of a point to its two closest centroids and their shifts in the last two iterations. By computer experimentation using synthetic and real instances, we found that this criterion reduces execution time to approximately 2/100 of the time by k-means and generates solutions whose quality is approximately reduced by less than 3%. These findings suggest the usefulness of our criterion for problems like those found in Big Data. The NP-hardness of k-means motivates the use of this heuristics.

Keywords: K-means, Algorithm complexity, Big Data, Heuristics, NP-hard.

1 Introduction

The current technology development has caused a significant increase in the amount of data generated and stored in public institutions as well as those in business, engineering, medicine, and transportation, among others. Therefore, there is a large justified interest in using the knowledge that can be extracted from those massive amounts of data, to make better decisions and to better understand their nature. However, extracting significant information from complex and constantly growing data sources poses new challenges [1].

Some of these data sources or repositories have sizes that exceed exabytes, and their size continues to grow [2]. The use and processing of large volumes of information have generated a large emerging computer field called Big Data [3]. In this sense, the contribution of this work consists of a strategy for solving the problem of data point clustering according to their attributes in the field of Big Data. Currently, there exists a large variety of clustering algorithms; however, k-means continues to be the preferred algorithm and is widely used in most of the real-life applications [4].

K-means is an iterative method, which consists of partitioning a set of n points into a certain number K of clusters. The complexity of k-means is $O(nKdl)$, where d is the number of dimensions and l represents the number of iterations carried out [5]. It has been shown that the k-means problem is NP-hard for $K \geq 1$ [6].

Let $X = \{x_i\}$, $i = 1, \ldots, n$ be the set of n d-dimensional points to be clustered into a set of K clusters, $C = \{c_k\}$, $k = 1, \ldots, K$. The k-means algorithm finds a partition such that the squared error between the empirical mean (centroid) of a cluster and the points in the cluster is minimized. Let μ_k be the mean of cluster c_k. The squared error between μ_k and the points in cluster c_k is defined by equation 1.

$$J(c_k) = \sum_{x_i \in C_k} \|x_i - \mu_k\|^2 \quad (1)$$

Received Jan 24, 2018 / Accepted Feb 11, 2018

The goal of k-means is to minimize the sum of the squared error over all the *K* clusters, which is expressed by equation 2.

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2 \quad (2)$$

In the standard version of k-means, the following four steps have been identified [7]:

1. Initialization. In this step, the initial centroids are defined for each of the *k* clusters.
2. Classification. For each data point, its distance to each of the centroids is calculated, and it is assigned to the cluster whose centroid is the closest.
3. Centroid calculation. For each cluster generated in the classification step, its centroid is recalculated.
4. Convergence. If the set of centroids remains unchanged in two consecutive iterations, the algorithm stops; otherwise, it continues in step 2.

This paper proposes a new criterion for reducing the processing time of the assignment of points to clusters for the k-means algorithm. By incorporating this criterion into k-means, a variant of k-means, called A-means (where the A refers to the assignment step), is the main contribution of this work.

Next, the organization of this paper is described. In Section 2 the related works are briefly described. In Section 3 the dynamic behaviour of the k-means algorithm is presented, which hinted at devising the definition of a threshold for data point classification that is presented in Section 4. The experimental evaluation of this proposed threshold is presented in Section 5, and the conclusions from this work are summarized in Section 6.

2 Related works

There exist numerous investigations on the improvement of the k-means algorithm, most of them aiming at improving the initialization step, which allows choosing better initial centroids. Some of these works are presented in [8-17].

In the classification step of k-means, the membership of a data point to a cluster is determined based on the maximal closeness of the point to each of the centroids. The distance calculations are computationally costly, and the improvements to the k-means algorithm in this step have focused on reducing the number distance calculations by applying heuristics and ad-hoc data structures. In [18] a heuristic is proposed, which claims that, if the distance from a point to its cluster centroid decreases from that of the preceding iteration, the point membership to the cluster will remain unchanged, thus allowing to exclude the point from distance calculations in subsequent iterations. In [19, 20] unnecessary distance calculations are excluded by applying the triangular inequality. In [21] it is claimed that the points close to their cluster boundary have a probability of changing membership higher than that of the points close to their centroid. The points that satisfy a criterion of closeness to their current centroid, keep their membership and are excluded from subsequent distance calculations. In [22] a heuristic is proposed that reduce the distance calculations only to the set of centroids close to each point, which is based on the observation that points only change membership to neighbouring or very close clusters. In [23, 24] stable clusters are identified; i.e., those clusters that do not experience point exchanges in two successive iterations; the points in these clusters keep their membership permanently unchanged and are excluded from future distance calculations. Other works that use geometric information of the points and centroid shifting for reducing distance calculation are presented in [25, 26].

In [27-29] it is shown that using *kd-tree* data structures can increase the efficiency of the k-means algorithm. There also exist works that present implementations of improvements for two steps, like in [30] where different improving methods are proposed for the initialization and the classification steps by using information from clusters and their data points in a previous iteration. In [31] a heuristic is proposed, which accelerates the algorithm convergence by predicting centroid positions based on the statistical information from preceding iterations. Other recent approaches for improving the k-means algorithm use distance measures different from the Euclidean measure [32-34].

Additionally, regarding the convergence step, in [7, 35, 36] several improvements are proposed for stopping the algorithm when in two consecutive iterations the value of the squared error satisfies the following conditions: it is greater than that of the

preceding iteration [7], it is smaller than a predefined threshold [35], or it is smaller than a threshold defined by the largest centroid shifts in the first iteration [36].

3 Dynamic behavior of k-means

For studying the behavior of the k-means algorithm in the classification phase, a visual tool was used for the experimental two-dimensional analysis [37]. The tool allowed visualizing point clustering iteration by iteration, knowing some variable values (mainly the sum of squared errors), and detecting membership change of data points from one cluster to another, among others. The behavior of k-means was studied by solving six real instances from the UCI repository [38]: Wine, Glass, Heart, Liver, Diabetes and Vehicle. Additionally, synthetic instances were generated with uniformly distributed data. Eight synthetic instances were solved with up to 640,000 data points. All the instances were solved for three values of *k*: 50, 100 and 200.

As a result of the experimental study, the following observations were obtained: a) clusters near the center of the data space undergo the largest number of cluster-membership changes, b) peripheral clusters become stabilized sooner than central clusters; i.e., their points cease to change cluster membership, c) a point can only change membership to an adjacent cluster from one iteration to the next one, d) points close to the boundary of their clusters have a probability of changing membership higher than those close to their centroids.

It is worth mentioning that although the study was conducted with two-dimensional instances, it can be inferred that the observed behavior also applies for multidimensional points. In particular, experiments were conducted with some three-dimensional instances, and the observations above were confirmed.

4 Proposed improvement

The improvement proposed in this work is an upgrade of the *Early Classification* (EC) heuristics [21], which reduces the number of distance calculations in the classification step. This heuristic is motivated by observations *c* and *d* (described before). The heuristics define two concepts: the equidistance index and the equidistance threshold. The equidistance index is defined by the absolute value of the difference of the distances from a point *i* to its two closest centroids, as shown in expression 3.

$$\alpha_i = \text{abs}\ (||i-\mu_1||^2 - ||i-\mu_2||^2) \quad (3)$$

The equidistance threshold is defined by the sum of the two largest centroid shifts concerning the previous iteration, as shown in expression 4.

$$\beta_j = m_1 + m_2 \quad (4)$$

It is important to point out that the EC heuristics is applied from the third iteration onwards. The purpose of EC is to determine the definitive membership for those points that have a small probability of changing membership in subsequent iterations, excluding them from future calculations.

The criterion for determining the definitive membership for a data point *i* to its current cluster is that it satisfies the condition that its equidistance index α_i is greater than the equidistance threshold.

Our improvement consists of a new threshold concept, which will be called *Improved Threshold* (IT).

The proposed concept is inspired by observations *c* and *d* in Section 3, in particular, it was observed that the centroids neighbouring a cluster might have shifts of different magnitude between two consecutive iterations.

The Improved Threshold concept is defined regarding the sum of the shifts of the two centroids $\hat{u}_m$ and $\hat{u}_n$ closest to point *i*, as shown in expression 5.

$$\hat{U}_i = \hat{u}_m + \hat{u}_n \tag{5}$$

In case that α_i is greater than $\hat{U}_i$ then point *i* keeps permanently its membership to its current cluster.

Fig. 1 shows a fragment of the data space, where the X's denote the positions of the centroids in a neighborhood in a previous iteration, and the squares, the positions in the current iteration. Notice that the shift of the centroid of cluster 5 is greater than that of the other clusters; consequently, the data points close to the boundary of the central cluster *i* and the neighbor cluster 5 have a higher probability of changing membership in subsequent iterations. The shaded areas denote the regions comprising the points with a high probability of changing membership, in this case, points *i* and *j*. Point *l*, since it lays outside of the shaded areas, keeps its membership permanently to the central cluster and it can be excluded from future calculations.

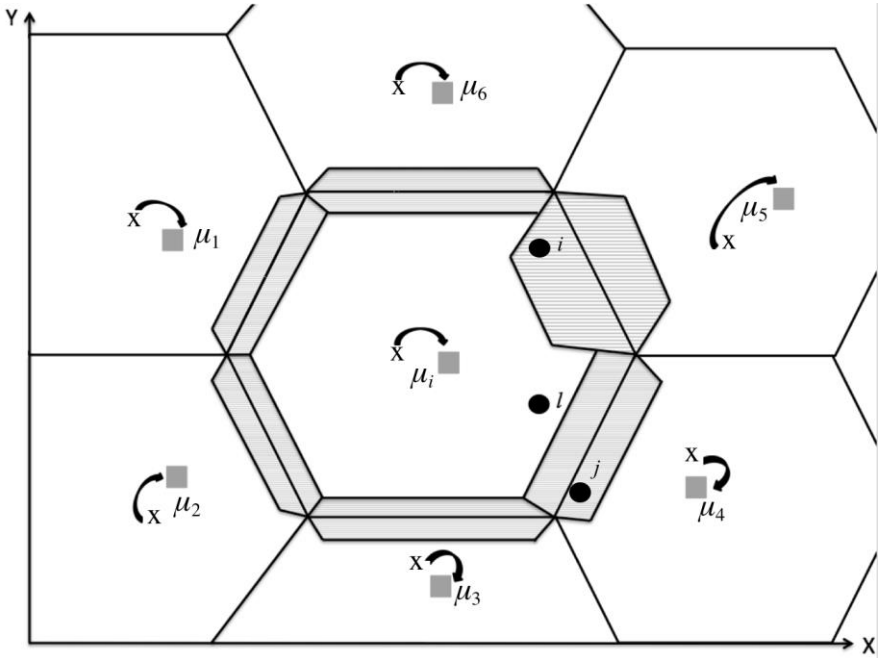

Fig. 1. The possibility of changing the membership of two points detected by Improved Threshold.

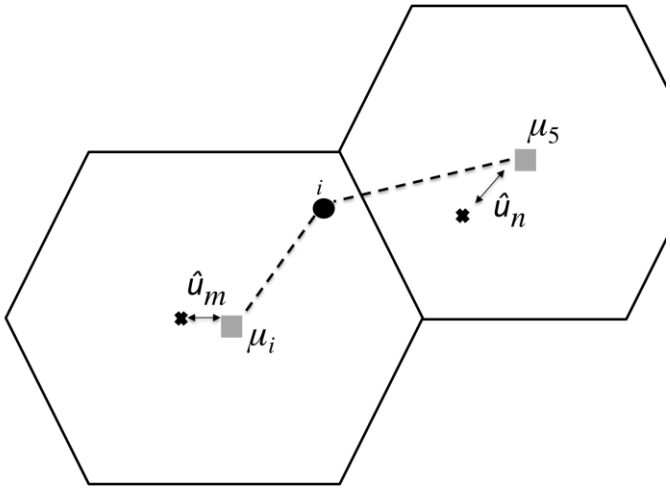

Fig. 2. The elements of the Improved Threshold concept.

Figure 2 shows that the centroids closest to point i are μ_i and μ_5. Notice that the shift of μ_i in two consecutive iterations is $\hat{u}_m$ and that of μ_5 is $\hat{u}_n$.

The pseudocode of A-means is presented next.

```
A-means{
begin
1. Initialization:
      N:={x1, … xn};
      M:={μ1, … μk};
2. Classification: For xi ∈ N and μk ∈ M, calculate the Euclidian distance of each xi
to the k centroids and assign point xi to the cluster with the closest centroid μk.
      Calculate the equidistance index αi
      Calculate the improved threshold Ûi
      If αi > Ûi
            then xi is permanently assigned to its cluster and is excluded from
      future distance calculations.
3.   Centroid calculation: Calculate the centroid for each cluster.
4. Convergence: If M:={μ1,… μk} remains unchanged in two consecutive iterations, stop
the algorithm; otherwise, go back to step 2.
end
}
```

5 Evaluation of the proposed improvement

For evaluating the Improved Threshold concept, several sets of synthetic and real instances were solved using Early Classification (EC), A-means and the k-means algorithm. The algorithms were coded in the C language with a GCC 4.9.2 compiler, and they were implemented on a Mac mini with the Yosemite 10.10 operating system, a Core i5 processor at 2.8 GHz and 16 GB of RAM.

To carry out statistical inference, it was determined that the size of the sample of runs be 30 for each instance. Therefore, all the instances were solved 30 times with each of the algorithms and for each value of k. For each run, a set of initial centroids was previously generated, to start the three algorithms with the same centroids. The results reported for each instance are the averages of the execution time and the solution quality over the 30 runs.

5.1 Test instances and experimental results

It is worth pointing out that the size of these instances is greater than those reported in most of the publications on k-means improvements. A major limitation for experimenting with larger instances was the extremely large time that the k-means algorithm spends for obtaining a solution.

In Table 1 the real and synthetic instances are described. The first and second columns indicate the name and type of the instance, the third shows the number n of data points, and the last one presents the number of attributes or dimensions. The real instances were obtained from the UCI Machine Learning Repository [38]. The synthetic instances were randomly generated using (and scaling up) a 0-1 interval for the number of attributes or dimensions. These instances were selected because they have large n and d values, which allows studying algorithm performance for large values of nkd.

Table 1. Description of experiment instances

Name	Type	*n*	*d*
DSAS	Real	1,140,000	45
POKER	Real	1,000,000	11
2ms2*d*	Synthetic	2,000,000	2
2ms4*d*	Synthetic	2,000,000	4
2ms5*d*	Synthetic	2,000,000	5
2ms6*d*	Synthetic	2,000,000	6
2ms7*d*	Synthetic	2,000,000	7
1m2*d*	Synthetic	1,000,000	2
1m3*d*	Synthetic	1,000,000	3

Table 2 shows the average results for 30 runs. The first and second columns indicate the name of the instance and the value of *k* used for its solution. Each of the next three column pairs shows the execution time and the value of the solution quality obtained using the k-means algorithm, EC and A-means.

Table 2. Experimental results for the instances

Name	*k*	K-means		Early Classification		A-means	
		Time hrs.	Quality	Time min.	Quality	Time min.	Quality
DSAS	100	16.31	8735706	78.12	8774982	23.09	8901816
DSAS	50	8.17	10112416	32.71	10138960	11.99	10229458
POKER	200	10.32	3828572	21.48	3850631	10.21	3869804
2ms2d	200	9.01	53705	4.92	55241	2.88	55952
2ms4d	100	6.59	10767	4.78	110702	2.94	112000
2ms6d	100	9.99	131870	7.05	136137	4.34	137809
2ms5d	200	24.97	434339	17.06	440133	9.01	443515
2ms7d	100	8.17	776353	14.83	785203	7.83	791080
1m2d	50	0.54	55731	0.62	55731	0.41	56345
1m2d	100	1.19	39181	1.19	39181	0.74	39658
1m2d	200	3.37	26839	2.42	27633	1.40	28018
1m3d	50	0.96	130427	1.05	132805	0.67	133865

In Table 3 the first and second columns indicate the name of the instance and the value of *k*. The third and fourth columns show the percentage of time reduction and the difference in solution quality for EC concerning k-means. The fifth and six columns show the percentage of time reduction and the difference in solution quality for A-means with respect to k-means. Finally, the last two columns show the percentage of time reduction and the difference in solution quality for A-means concerning EC.

Table 3. Percentage of time reduction and difference in solution quality for the instances

Name	*k*	Early Classification versus k-means		A-means versus k-means		A-means versus Early Classification	
		Time	Quality	Time	Quality	Time	Quality
DSAS	100	92.02	-0.44	97.64	-1.90	70.44	-1.44
DSAS	50	93.32	-0.26	97.55	-1.15	63.32	-0.89
POKER	200	96.53	-0.57	98.35	-1.07	52.48	-0.49
2ms2d	200	99.08	-2.85	99.46	-4.18	41.51	-1.28
2ms4d	100	98.79	-2.81	99.25	-4.01	38.37	-1.17
2ms6d	100	98.82	-3.23	99.27	-4.50	38.37	-1.22
2ms5d	200	98.86	-1.33	99.39	-2.11	47.21	-0.76
2ms7d	100	96.97	-1.13	98.40	-1.89	47.20	-0.74
1m2d	50	98.09	-3.18	98.73	-4.31	33.69	-1.10
1m2d	100	98.33	-2.91	98.95	-4.16	37.49	-1.21
1m2d	200	98.80	-2.95	99.30	-4.39	41.97	-1.39
1m3d	50	98.18	-1.82	98.82	-2.63	35.34	-0.79

Based on the experimental results, it was found that, in the solution of all the instances, the A-means algorithm obtained results faster than EC, in the best case reducing execution time by 70.44% and in the worst case by 33.7%. The differences in quality were respectively -1.44 % and -1.1 %. The experimental results show that A-means is much more efficient than k-means for all the instances: an average time reduction of 98% and a quality reduction of only 3% on average.

Additionally, when comparing the results for A-means with those from EC and k-means, it was observed that as the value of *nkd* (complexity of k-means per iteration) increases it does so the percentage of time reduction of algorithm A-means with respect to EC and k-means. The superior results of A-means with large values of *k* are encouraging for its application for solving large instances.

6 Conclusions

A reasonable criterion is proposed for reducing the execution time for the k-means algorithm without affecting the solution quality significantly, when it is applied to solving large instances, like those found in Big Data. Its advantage was shown by applying it in the cluster assigning step of one of the most used variants of the k-means family. Hinted by the observation that a data point close to the centroid of its cluster has a low probability of changing cluster membership, the proposed criterion consists of permanently assigning a point to a cluster in any iteration when the difference of the distances from the point to its two closest centroids becomes larger than the certain established threshold. Based on intensive computational experiments with synthetic and real instances, we found that this criterion reduces the processing time significantly at the expense of a relatively small decrease in the quality of the results obtained. Additionally, the best results are obtained for the largest instance sizes; i.e., those where the value of *nkd* is high. Therefore, this behavior is an indicator of the usefulness of applying the proposed criterion for large-instance clustering. Finally, but not less important, this classification criterion is not incompatible with other improvements that apply to the initialization or convergence steps of k-means.

References

1. Raykov, Y. P., Boukouvalas, A., Baig, F., Little, M. A.: What to do when K-means Clustering Fails: A Simple yet Principled Alternative Algorithm. PloS one. 11, 9 (2016)
2. Kambatla, K., Kollias, G., Kumar, V., Grama, A.: Trends in Big Data Analytics. Journal of Parallel and Distributed Computing. 74, 7, 2561-2573 (2014)
3. Chen, C.P., Zhang, C.Y.: Data-intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data. Information Sciences. 275, 314-347 (2014)
4. Berkhin, P.: A Survey of Clustering Data Mining Techniques. In: Grouping Multidimensional Data, pp. 25-71 Springer-Verlag, Heidelberg (2006)
5. Jain, A. K.: Data clustering: 50 years beyond K-means. Pattern Recognition Letters. 31, 8, 651-666 (2010)
6. Mahajan, M., Nimbhorkar, P., Varadarajan, K.: The planar k-means problem is NP-hard. Theoretical Computer Science. 442, 13-21 (2012)
7. Pérez, J., Pazos, R., Cruz, L., Reyes, G., Basave, R., Fraire, H.: Improving the Efficiency and Efficacy of the K-means Clustering Algorithm Through a New Convergence Condition. In: International Conference on Computational Science and Its Applications, pp. 674-682. Springer, Berlin, Heidelberg (2007)
8. Heckerman, D., Meila, M.: An Experimental Comparison of Several Clustering and Initialization Methods. Machine Learning. 42, 9-29 (2001)
9. Celebi, M. Emre, Hassan A. Kingravi, Patricio A. Vela.: A Comparative Study of Efficient Initialization Methods for the K-means Clustering Algorithm. Expert Systems with Applications. 40, 1, 200-210 (2013)
10. Zhanguo, X., Shiyu, C., Wentao, Z.: An Improved Semi-Supervised Clustering Algorithm Based on Initial Center Points. Journal of Convergence Information Technology. 7, 5, 317-324 (2012)
11. Tzortzis, G., Likas, A.: The MinMax k-Means Clustering Algorithm. Pattern Recognition. 47, 7, 2505-2516 (2014)
12. Salman, R., Vojislav K., Qi L., Strack, R. Test, E.: Two-Stage Clustering with k-Means Algorithm. Recent Trends in Wireless and Mobile Networks. 110-122, Springer Berlin, Heidelberg (2011)
13. Li, C. S.: Cluster Center Initialization Method for K-means Algorithm Over Data Sets with Two Clusters. Procedia Engineering. 24, 324-328 (2011)
14. El Agha, M., Wesam, M. A.: Efficient and Fast Initialization Algorithm for K-means Clustering. International Journal of Intelligent Systems and Applications. 1, 21-31 (2012)

15. Lee, S.S., Lin, J.C.: An Accelerated K-means Clustering Algorithm Using Selection and Erasure Rules. Journal of Zhejiang University SCIENCE C. 13, 10, 761-768 (2012)
16. Arthur, D., Vassilvitskii, S.: k-means++: The Advantages of Careful Seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1027-1035 (2007)
17. Zahra, S., Ghazanfar, M. A., Khalid, A., Azam, M. A., Naeem, U., Prugel-Bennett, A.: Novel centroid selection approaches for KMeans-clustering based recommender systems. Information Sciences. 320, 156-189 (2015)
18. Fahim, A.M., Salem, A.M., Torkey, F.A., Ramadan, M.A.: An Efficient Enhanced K-means Clustering Algorithm. Journal of Zhejiang University SCIENCE A. 7, 10, 1626-1633 (2006)
19. Elkan, C.: Using the Triangle Inequality to Accelerate K-means. In: Proceedings of the 20th International Conference on Machine Learning, pp. 147-153 (2003)
20. Hamerly, G.: Making K-means Even Faster. In: Proceedings of the SIAM International Conference on Data Mining, pp. 130-140 (2010)
21. Pérez, J., Pires, C. E., Balby, L., Mexicano, A., Hidalgo, M. Á.: Early Classification: A New Heuristic to Improve the Classification Step of K-Means. Journal of Information and Data Management. 4, 2, 94-103 (2013)
22. Pérez, J., Pazos R., Olivares V., Hidalgo M., Ruiz J., Martínez A., Almanza, N., González, M.: Optimization of the K-means Algorithm for the Solution of High Dimensional Instances. In: Proceedings of International Conference of Numerical Analysis and Applied Mathematics. AIP Conference Proceedings Vol. 1738 (2016)
23. Pérez, J., Pazos R., Hidalgo M., Almanza N., Diaz-Parra O., Santaolaya R., Caballero V.: An Improvement to the K-means Algorithm Oriented to Big Data. In: Proceedings of International Conference of Numerical Analysis and Applied Mathematics. AIP Conference Proceedings Vol. 1648 (2015)
24. Lai, J.Z., Huang, T.J., Liaw, Y.C.: A Fast K-means Clustering Algorithm Using Cluster Center Displacement. Pattern Recognition. 42, 11, 2551-2556 (2009)
25. Chiang, M.C., Tsai, Ch.W., Yang, Ch.S.: A Time-Efficient Pattern Reduction Algorithm for K-means Clustering. Information Sciences. 181, 4, 716-731 (2011)
26. Lai, J.Z. and Huang, T.J.: Fast Global K-means Clustering Using Cluster Membership and Inequality. Pattern Recognition. 43,5, 1954-1963 (2010)
27. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An Efficient K-means Clustering Algorithm: Analysis and Implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 24,7, 881-892 (2002)
28. Lai, J.Z., Liaw, Y.C.: Improvement of the K-means Clustering Filtering Algorithm. Pattern Recognition. 41,12, 3677-3681 (2008)
29. Vrahatis, M.N., Boutsinas, B., Alevizos, P., Pavlides, G.: The New K-Windows Algorithm for Improving the K-means Clustering Algorithm. Journal of Complexity. 18, 1, 375-391 (2002)
30. Qi, J., Yu, Y., Wang, L., Liu, J., Wang, Y.: An Effective and Efficient Hierarchical K-means Clustering Algorithm. International Journal of Distributed Sensor Networks. 13, 8, 1-17 (2017)
31. Borlea, I.D., Precup, R.E., Dragan, F., Borlea, A.B.: Centroid Update Approach to K-Means Clustering. Advances in Electrical and Computer Engineering. 17, 4, 3-10 (2017)
32. Zhang, T., Ma, F.: Improved rough k-means clustering algorithm based on weighted distance measure with Gaussian function. International Journal of Computer Mathematics. 94,4, 663-675 (2017)
33. Chakraborty, S., Das, S.: k−Means clustering with a new divergence-based distance metric: Convergence and performance analysis. Pattern Recognition Letters. 100, 67-73 (2017).
34. Nielsen, F., Sun, K.: Clustering in Hilbert simplex geometry. arXiv:1704.00454 (2017)
35. Lam, Y. K., Peter W.M.T.: eXploratory K-Means: A New Simple and Efficient Algorithm for Gene Clustering. Applied Soft Computing. 12, 3, 1149-1157 (2012)
36. Mexicano, A., R. Rodríguez, S. Cervantes, P. Montes, M. Jiménez, N. Almanza, A. Abrego.: The Early Stop Heuristic: A New Convergence Criterion for K-means. In: International Conference of Numerical Analysis and Applied Mathematics. AIP Conference Proceedings Vol. 1738 (2016)
37. Pérez-Ortega, J., Almanza-Ortega, N. N., Rodríguez-Lelis, J. M., Sáenz- Sánchez, S.: VICK-Means: Una Herramienta Visual de Apoyo al Análisis del Comportamiento del Algoritmo K- Means. In: Congreso Interdisciplinario de Cuerpos Académicos, Edición Internacional. Guanajuato, México (2017)
38. Lichman, M., UCI Machine Learning Repository, (2013) http://archive.ics.uci.edu/ml