On Autonomous k-Means Clustering

Conference Paper · May 2005

DOI: 10.1007/11425274_24 · Source: DBLP

Tapio Elomaa
Tampere University

On Autonomous k-Means Clustering

Tapio Elomaa and Heidi Koivistoinen

Institute of Software Systems, Tampere University of Technology
P.O. Box 553, FI-33101 Tampere, Finland
{tapio.elomaa,heidi.koivistoinen}@tut.fi

Abstract. Clustering is a basic tool in unsupervised machine learning and data mining. One of the simplest clustering approaches is the iterative k-means algorithm. The quality of k-means clustering suffers from being confined to run with fixed k rather than being able to dynamically alter the value of k. Moreover, it would be much more elegant if the user did not have to supply the number of clusters for the algorithm.
In this paper we consider recently proposed autonomous versions of the k-means algorithm. We demonstrate some of their shortcomings and put forward solutions for their deficiencies. In particular, we examine the problem of automatically determining a good initial candidate as the number of clusters.

1 Introduction

In unsupervised learning instances without predefined classification are obtained and the task is to learn relevant information from them. The most common approach is clustering [1, 2], where one aims at finding a natural categorization of the given instances in mutually exclusive or partially overlapping classes. A simple and widely used clustering algorithm is k-means clustering [3–5]: Given k initial (e.g., random) cluster centers $C = \{ c_1, \ldots, c_k \}$ for the metric observations $X = \{ x_1, \ldots, x_n \}$, iterate the following until the cluster centers do not change.

1. For each observation point $x_i \in X$ determine the center $c_j \in C$ that is closest to it and associate x_i with the corresponding cluster.
2. Recompute the center locations (as the center of the mass of points associated with them).

The basic assumption underlying k-means is that there is a metric on the set of data points. It is required, e.g., in order to be able to compute the mean values. The metric is a distance function d that obeys the following three properties:

- positiveness: $d(x, y) = 0$ if and only if $x = y$,
- symmetry: $d(x, y) = d(y, x)$, and
- triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$.

The iterative approach minimizes the squared error — the squared Euclidean distance between the cluster centers and the observations associated with them.

One can view k-means clustering to be an instantiation of the generic Expectation Maximization (EM) algorithm [6] with assumptions on spherical Gaussian clusters, instance space partitioning by clusters, and equal weights on the underlying mixture model [7].

The obvious shortcomings of the basic k-means clustering are that the number of clusters needs to determined in advance and the computational cost with respect to the number of observations, clusters, and iterations. Though, k-means is efficient in the sense that only the distance between a point and the k cluster centers — not all other points — needs to be (re)computed in each iteration. Typically the number of observations n is much larger than k.

Recently there have been many approaches trying to alleviate both of these problems. Variations of k-means clustering that are supposed to cope without prior knowledge of the number of cluster centers have been presented [8, 9]. Several proposals for scaling up clustering and, in particular, k-means for massive data sets have been proposed [10–13]. Quite often these studies assume that a particular distance measure is applied. Moore [13] and Elkan [14] have shown how the triangle inequality can be used for any distance function to reduce the number of distance calculations required during the execution of k-means by carrying information over from iteration to iteration.

In this paper we review autonomously working variations of k-means clustering. I.e., k-means algorithms which, equipped with a statistical or other criterion, try to determine whether the current k-clustering of the given data is a good one, or could a better one be obtained by altering the number of clusters k. Obviously, the need to consider different values of k puts autonomous algorithms in danger of being far too inefficient. Our experiments demonstrate that they, in fact, do not scale up well in time consumption. The G-means algorithm [9] does not determine how to choose the initial value of k. In the X-means algorithm [8] the user is required to provide an upper and lower bound for the value of k. We try to avoid losing autonomous behavior by using efficient geometric schedule [15] to determine a lower bound for k.

The remainder of this paper is organized as follows. In Section 2 we review recent improvements on some aspects of k-means clustering. In particular, we concentrate on automatic detection of the suitable number of clusters and accelerating the execution of the k-means algorithm. In Section 3 we briefly study Hamerly and Elkan's [9] G-means algorithm. Approximating the number of clusters prior to running the autonomous algorithm is the topic of Section 4. In Section 5 we present and empirical evaluation of the proposed approach. Finally, Section 6 presents the concluding remarks of this study.

2 Improving *k*-Means Clustering

The k-means algorithm is known to converge to a local minimum of the average squared distance from the points to their cluster centers. Unfortunately, the basic iterative algorithm is also known to be notoriously slow for databases of practical interest. Obviously, initializing the cluster centers in an informative way — rather than randomly — may lead to faster convergence and better clustering quality [7, 16]. An approach that is often used to improve the quality of k-means clustering is to run the algorithm with many different initializations [5].

2.1 Determining the (Right) Value of k

Not only the center locations but also their number has a significant impact on the final result of k-means clustering. Obviously, zero-error clustering can be obtained by making each different data point its own cluster, but such grouping of the data is of no value. In practice, the parameter k is most often chosen by the user based on her assumptions, experience, and prior knowledge. Choosing a too large value for k further slows down the algorithm and unnecessarily splits the natural clusters that exist in the data. A too small value for k, on the other hand, requires that some of the existing natural clusters be represented by a single cluster center.

Ideally the value of k should automatically adapt to the input data. There are two possible directions to proceed: One can either increase the number of clusters as required or start with a sufficiently large number of cluster centers and reduce their number whenever possible. For example the minimum description length (MDL) method of Bischof, Leonardis, and Selb [17] follows the latter approach.

The X-means algorithm of Pelleg and Moore [8], on the other hand, works by creating new clusters by splitting an existing one in two whenever a better fit to the data is obtained. The splitting decision in X-means is based on the *Bayesian information criterion* (BIC). Moreover, X-means also caches information that remains unchanged over iterations. The empirical experiments of Pelleg and Moore [8] demonstrate X-means to be more efficient than the original k-means and to produce better clusterings. However, the statistical power of BIC has been questioned [9]. It is claimed that BIC tends to overfit by choosing too many cluster centers when the data is not strictly spherical.

The G-means algorithm of Hamerly and Elkan [9] neither requires the user to supply the number of cluster centers in advance and also works by splitting clusters in two. The decision whether to split a cluster or not is based on a statistical test (Anderson-Darling) measuring if the data currently assigned to a cluster center appear to be Gaussian. If the data do not appear to be Gaussian, it is divided in two in an attempt to improve the overall Gaussian fit. According to experiments [9] G-means often outperforms X-means, the reason being that BIC is ineffective in penalizing for the model's complexity. G-means is not free from all parameter values, since the user is required to supply a value for the significance level of the statistical test. However, the significance level of a statistical test is somewhat more intuitive than the number of clusters.

2.2 Accelerating k-Means Clustering

Pelleg and Moore [18] combined an additional data structure, *kd*-tree, to k-means clustering. The *kd*-tree represents a hyper-rectangular partitioning of the instance space. By this additional data structure one can update centers in bulk rather than point by point. By geometric reasoning Pelleg and Moore are able to show how points contained in hyper-rectangles of the *kd*-tree can be updated all at once. The same technique underlies the acceleration method of X-means algorithm, *blacklisting* [8].

Moore [13] continued this line of research further by defining another data structure, *anchors hierarchy*. Similarly as in using *kd*-trees, the intent is to take advantage of a data structure that, by using cached sufficient statistics, summarizes statistical information from a large data set. The anchors hierarchy only respects triangle inequality, which is the only (non-trivial) requirement that can be posed to a distance metric.

Also Elkan [14] proposed to accelerate k-means through the use of triangle inequality. His algorithm keeps track of the distances between cluster centers. By combining these distances with known distances between a data point and its cluster center, the point's distance from some other cluster centers can be ignored because they are known to be further away from the point in question. Thus, the number of distance calculations in k-means can be reduced radically.

His observation is that cluster centers that are further away from the current cluster center c of an instance x than twice the distance between x and c cannot become the center "owning" x. Therefore, there is no need to explicitly calculate the distance between x and such a center. This observation also holds for those centers for which an upper bound of the distance from c rather than the exact distance is known.

As a decision problem minimum error clustering is a computationally hard problem. Therefore, it is natural to try to approximate the solution. Many different approaches, often based on subsampling, which achieve excellent efficiency with guaranteed performance for this problem have been developed [19, 20].

3 G-Means Algorithm

Let us review closer Hamerly and Elkan's [9] G-means algorithm. It starts by using k-means clustering to associate the given observations to the given initial clusters. The number of initial clusters may of course be one or the algorithm can be initialized otherwise. The clusters are then checked using a statistical test to detect whether all their associated data follows Gaussian distribution. If not, the cluster is partitioned further in an attempt to find Gaussian sub-clusters.

```
$G$-means($X, \alpha, \{ c_1, \ldots, c_k \}$)

1. $C \leftarrow k$-means($X, \{ c_1, \ldots, c_k \}$);
   % Let $X_c \subseteq X$ denote the data assigned to $c \in C$
2. while $\exists c \in C$ such that $X_c$ does not follow Gaussian distribution according
   to a statistical test (at confidence level $\alpha$) do
   (a) Determine two new cluster centers $c_a$ and $c_b$;
   (b) $C \leftarrow C \setminus \{ c \} \cup \{ c_a, c_b \}$;
   (c) $G$-means($X_c, \alpha, \{ c_a, c_b \}$);
3. $k$-means($X, C$);
```

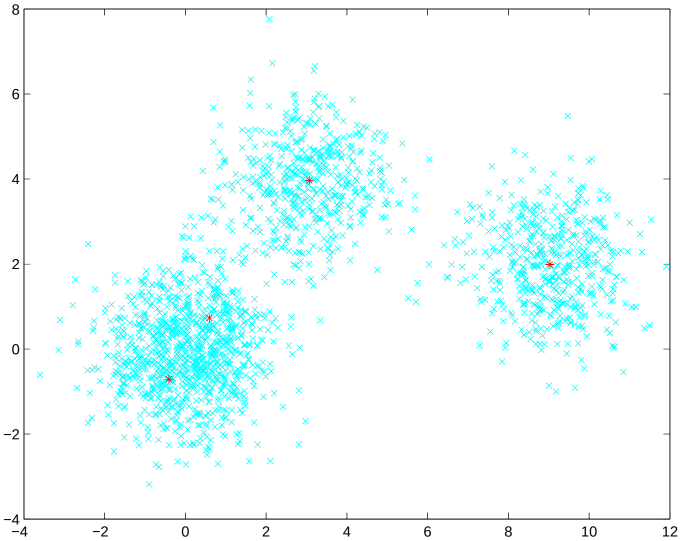

Fig. 1. Clusters of 1000, 500, and 500 instances. G-means divides the densest cluster needlessly in two.

The potential new cluster centers that may replace center c are $c \pm m$, where m is as follows. Determine the main principal component s of the data. Let its eigenvalue be λ. Then $m = s\sqrt{2\lambda/\pi}$. The idea is to place the new cluster centers to their expected locations under the assumption that the associated data does not come from a single Gaussian distribution.

Observe that once k-means has stabilized the cluster centers (Step 1), any instance $x \in X_c$ that is associated with a cluster c, that subsequently is subject to a division, is improbable to be associated with any other cluster than one of the newly created clusters, c_a and c_b. Thus, there is no need to (re)calculate the distances between instances and cluster centers other than the new ones. Therefore, one can implement the rest of the algorithm by recursive partitioning of the identified clusters. This makes G-means a relatively efficient algorithm. Moreover, the implementation of the algorithm orders the data so as to avoid needless recalculations. However, at the end one must rerun k-means to ensure that all data points are associated with correct cluster centers.

By Hamerly and Elkan's [9] experiments X-means and G-means are equally good at finding the correct k and maximizing the BIC statistic on spherically distributed data. For non-spherical data, on the other hand, G-means is substantially better at finding the correct ks. Fig. 1, demonstrates a situation where G-means, nevertheless, overestimates the number of clusters in a very clear-cut three-cluster spherical data. The cluster on the left contains twice as many instances as those on the right.

Another problem still affecting G-means is its inefficiency for data that actually contains many clusters. The growth rate appears to be exponential. For example, we ran G-means with data that had 500 true clusters. The algorithm was able to come up with a clustering having 502 clusters, which is a very good result considering the large number of clusters. The time required to produce this result was on average somewhat above 8 minutes on a PC, while data with 600 clusters already requires over 1 hour and 20 minutes to come up with a result of 607 clusters.

4 Prior Approximation of the Number of Clusters

A straightforward way to improve G-means is to start the algorithm from a more realistic lower bound of the number of required clusters than 1. However, we do not want to destroy the autonomous behavior of the algorithm by requiring the user to supply the lower bound. Instead, we aim at developing an efficient automatic analysis of the input data to provide such a bound for G-means.

There are several ways to choose the locations of the k initial cluster centers. Methods proposed in the literature include [7]: choose random locations, run k clustering problems, take the mean of the whole data and randomly perturb it k times, take the k densest bins along any of the coordinates. Moreover, one can iterate the algorithm with different initial selections [5] — at least for relatively small data sets.

Determining a suitable value for k is a model selection problem, which is being tackled by X-means and G-means during model construction. However, as our preceding example demonstrates, even these algorithms would benefit from a better initial value for the number of eventual clusters. The squared error of k-means clustering monotonically decreases with increasing number of clusters. Therefore, it cannot be used as the basis of model selection.

None of the simple cluster localization methods mentioned above lend themselves as a useful method of initializing k. The approach of Bradley and Fayyad [7, 16] works by repetitively subsampling the data, clustering each subsample to k means and, finally, clustering over the cluster centers obtained in different repetitions. Thus, this initialization smooths the approximations obtained from different repetitions.

What X-means does exactly is the following. Assuming a user supplied lower and upper bound for the number of clusters, it finds all locally optimal k-means clusterings within this range, evaluates them using the BIC criterion, and at the end returns the clustering that evaluates the best.

We combine the clustering of subsamples idea of Bradley and Fayyad [7] with the BIC scoring of k-means clusterings idea of Pelleg and Moore [8]. However, as we need to have a conservative sampling strategy that never over-estimates and BIC is known to be weak to penalize against the growth of the number of clusters, we rather employ the Anderson-Darling test used in G-means anyway. In summary, our initialization approach is the following.

1. Draw a subsample S of m instances of the data.

Table 1. Average results over 15 repetitions of the empirical evaluation.

		G-means		G-means2		
Instances	Clusters	Time	Clusters	Init. k	Time	Clusters
50	100	4.9 ±0.8	96.0 ±2.0	17.1 ±1.1	4.6 ±0.4	98.4 ±0.4
	200	21.4 ±18.9	191.7 ±4.7	34.1 ±2.1	17.2 ±3.3	197.7 ±0.3
	300	37.3 ±4.6	288.7 ±2.3	59.7 ±4.3	40.3 ±6.2	295.3 ±3.7
	400	73.1 ±8.5	381.9 ±0.9	68.3 ±4.3	72.7 ±7.5	393.5 ±6.5
	500	118.9 ±7.3	488.1 ±5.9	102.4 ±25.6	120.0 ±10.6	491.7 ±0.3
	600	181.6 ±56.0	591.1 ±8.1	128.0 ±0.0	173.7 ±49.4	591.5 ±0.5
	700	251.0 ±18.9	683.7 ±4.3	128.0 ±0.0	227.2 ±38.0	687.9 ±2.1
100	100	8.3 ±1.1	97.1 ±0.9	34.1 ±2.1	7.4 ±0.1	99.1 ±0.1
	200	34.8 ±5.0	197.5 ±2.5	68.3 ±1.5	35.1 ±24.5	198.5 ±0.5
	300	84.8 ±12.3	302.2 ±0.8	128.0 ±0.0	87.3 ±14.8	297.1 ±0.1
	400	155.9 ±1.2	398.2 ±2.8	128.0 ±0.0	157.0 ±21.0	396.9 ±1.1
	500	247.0 ±30.0	518.1 ±0.9	256.0 ±0.0	241.1 ±12.6	495.1 ±1.1
	600	417.3 ±17.4	607.1 ±7.1	256.0 ±0.0	420.9 ±8.8	593.7 ±3.7
	700	570.6 ±208.9	703.2 ±4.8	256.0 ±0.0	557.9 ±140.78	695.2 ±5.8

2. Using progressive sampling with a geometric schedule [15] for values $k = 2^1, 2^2, 2^3, \ldots$ execute k-means(S, k); evaluate the resulting clustering using the statistical criterion.
3. Stop geometric sampling when the criterion does not improve anymore or $k = \lfloor \log m \rfloor$.
4. Choose the k and the associated clustering that obtained the best score.

Altogether there are at most $\log m$ calls for k-means, but all with a small sample of the data. As long as $m \ll n$, one can expect that the time spent in searching for the lower bound of k is easily regained in time savings during the execution of G-means.

5 Empirical Evaluation

We now compare G-means algorithm without prior approximation of the number of clusters with the approach outlined above (G-means2). Of course, our approach cannot be competitive when there are only few true clusters in the data. Therefore, our evaluation will be carried out using generated data that contains quite many clusters. The data was generated randomly: the desired number of random (but clearly separable) cluster centers of 50 or 100 instances each were first drawn. The required number of instances were drawn equiprobably around the centers. The range of the number of clusters ranges from 100 to 700. In our experiments the number of repetitions is 15.

Table 1 summarizes the results of our experiments with the significance parameter α value 0.5 in choosing the initial cluster center candidates. The table gives for each experiment the number of instances per cluster, the number of true clusters, the average time (in seconds) and the average number of found clusters for G-means, the average initial k value chosen by the geometric sampling schedule, and the average time and number of clusters for G-means2.

As a general trend of these experiments we can observe that G-means2 is on the average slightly more efficient (counting in the random sampling phase) than the original algorithm. The advantage with these numbers of clusters, though, is not as large as we would have expected in advance. Moreover, it also approximates the true number of clusters on the average more faithfully than G-means.

G-means2 does not obtain a larger edge on time consumption probably due to the fact that the cluster centers chosen by subsampling can be way off of true cluster center locations. Therefore, the algorithm may have to loop through a large number of clusters that need to be divided, while in the original k-means one can always determine some of the eventually formed clusters not to need further attention. Also, the geometric sampling strategy may sometimes lead to grave under-estimates of the number of required clusters. In these cases the modified algorithm usually ends up spending more time than the original one.

The fact that G-means2 on the average comes up with a more correct number of clusters than the original algorithm is an added bonus that we did not expect in advance. On the other hand, it is not surprising that if the recursive algorithm is allowed to start from a better lower bound for the required number of clusters than 1, it will end up with better end result. Both algorithms, though, are on the average off by only one or two per cents.

6 Conclusion and Further Work

We have proposed to use subsampling for determining a starting value for the search of the number of clusters, not only the locations of the initial cluster centers. In this process we employ the geometric sampling schedule and a statistical test to decide when to stop growing the initial value. Our empirical experiments demonstrate this to be a promising approach: it leads on the average to better results than executing G-means starting from one initial cluster.

We intend to study more refined, but still efficient, methods of choosing the number and locations of the cluster centers. If the chosen initial value of k now is 2^i, then there is still room to look for a higher value within the range $2^i \dots 2^{i+1}$. Moreover, at the moment we do not know comprehensively the significance of the α parameter in determining the initial cluster centers. It is also possible to consider other ways of determining the initial value of k than using the Anderson-Darling test. Model selection is a widely studied topic in machine learning, and one of the approaches arising in there could be employed to this task as well.

Of course, it is also our plan to carry out a wider experimental study of the algorithm. So far only similarly generated random data was included in the tests. We intend to test the algorithm also with real-world data sets.

Acknowledgements

We thank Greg Hamerly for giving his Matlab-code of G-means at our disposal.

References

1. Fisher, D.: Knowledge acquisition via incremental conceptual clustering. Machine Learning **2** (1987) 139–172
2. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys **31** (1999) 264–323
3. MacQueen, J.B.: On convergence of k-means and partitions with minimum average variance (abstract). Annals of Mathematical Statistics **36** (1965) 1084
4. Forgy, E.: Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics **21** (1965) 768
5. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley & Sons (1973)
6. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Royal Statistical Society, Series B **39** (1977) 1–38
7. Bradley, P.S., Fayyad, U.M.: Refining initial points for k-means clustering. In: Proc. 15th Intl. Conf. on Machine Learning, Morgan Kaufmann (1998) 91–99
8. Pelleg, D., Moore, A.: X-means: Extending k-means with efficient estimation of the number of clusters. In: Proc. 17th Intl. Conf. on Machine Learning, Morgan Kaufmann (2000) 727–734
9. Hamerly, G., Elkan, C.: Learning the k in k-means. In: Advances in Neural Information Processing Systems 16. MIT Press (2004)
10. Ng, R.T., Han, J.: Efficient and effective clustering methods for spatial data mining. In: Proc. 20th Intl. Conf. on Very Large Data Bases, Morgan Kaufmann (1994) 144–155
11. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: Proc. ACM SIGMOD Intl. Conf. on Management of Data, ACM Press (1995) 103–114
12. Guha, S., Rastogi, R., Shim, K.: Cure: An efficient clustering algorithm for large datasets. In: Proc. ACM SIGMOD Intl. Conf. on Management of Data, ACM Press (1998) 73–84
13. Moore, A.W.: The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In: Proc. 16th Conf. on Uncertainty in Artificial Intelligence, Morgan Kaufmann (2000) 397–405
14. Elkan, C.: Using the triangle inequality to accelerate k-means. In: Proc. 20th Intl. Conf. on Machine Learning, AAAI Press (2003) 147–153
15. Provost, F., Jensen, D., Oates, T.: Efficient progressive sampling. In: Proc. 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, ACM Press (1999) 23–32
16. Fayyad, U.M., Reina, C., Bradley, P.S.: Initialization of iterative refinement clustering. In: Proc. 4th Intl. Conf. on Knowledge Discovery and Data Mining, AAAI Press (1998) 194–198
17. Bischof, H., Leonardis, A., Selb, A.: MDL-principle for robust vector quantisation. Pattern Analysis and Applications **2** (1999) 59–72
18. Pelleg, D., Moore, A.W.: Accelerating exact k-means algorithms with geometric reasoning. In: Proc. 5th Intl. Conf. on Knowledge Discovery and Data Mining, AAAI Press (1999) 277–281
19. Har-Peled, S., Mazumdar, S.: On coresets for k-means and k-median clustering. In: Proc. 36th Annual Symp. on Theory of Computing, ACM Press (2004) 291–300
20. Kumar, A., Sabharwal, Y., Sen, S.: A simple linear time $(1 + \varepsilon)$-approximation algorithm for k-means clustering in any dimensions. In: Proc. 45th Annual IEEE Symp. on Foundations on Computer Science, IEEE Press (2004) 454–462