K-means vs Mini Batch K-means: A comparison

Javier Béjar
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
bejar@lsi.upc.edu

Abstract

Mini Batch K-means ([11]) has been proposed as an alternative to the K-means algorithm for clustering massive datasets. The advantage of this algorithm is to reduce the computational cost by not using all the dataset each iteration but a subsample of a fixed size. This strategy reduces the number of distance computations per iteration at the cost of lower cluster quality. The purpose of this paper is to perform empirical experiments using artificial datasets with controlled characteristics to assess how much cluster quality is lost when applying this algorithm. The goal is to obtain some guidelines about what are the best circumstances to apply this algorithm and what is the maximum gain in computational time without compromising the overall quality of the partition.

Keywords: Unsupervised Learning, Scalable Algorithms, K-means

1 Introduction

K-means [5, 7] is one of the most used clustering algorithms, mainly because of its good time performance. With the increasing size of the datasets being analyzed, this algorithm is losing its attractive because its constraint of needing the whole dataset in main memory. For this reason several methods have been proposed to reduce the temporal and spatial cost of the algorithm.

For example, in [3] it is proposed a modification in how is computed the assignment of examples to cluster prototypes using the triangular inequality. This method effectively reduces the number of distance computations each iteration, but maintaining the need of having all the dataset in memory. In [2] several strategies are used to reduce the amount of data needed to update the cluster centroids each iteration by selecting random samples of the dataset, by summarizing examples using sufficient statistics and by discarding examples that have a small impact on the clusters.

A different approach is the mini batch K-means algorithm ([11]). Its main idea is to use small random batches of examples of a fixed size so they can be stored in memory. Each iteration a new random sample from the dataset is obtained and used to update the clusters and this is repeated until convergence. Each mini batch updates the clusters using a convex combination of the values of the prototypes and the examples, applying a learning rate that decreases with the number of iterations. This learning rate is the inverse of number of examples assigned to a cluster during the process. As the number of iterations increases, the effect of new examples is reduced, so convergence can be detected when no changes in the clusters occur in several consecutive iterations. A detailed algorithm is presented in algorithm 1.

This methodology is also used in other domains such as artificial neural networks as a mean to reduce training time for the backpropagation algorithm [8].

The empirical results presented in [11] suggest that it can be obtained a substantial saving of computational time at the expense of some loss of cluster quality, but not extensive study of the algorithm has been done to measure how the characteristics of the datasets, such as the number of clusters or its size, affect the partition quality. Also, only the objective function that the k-means algorithm minimizes (distorsion) is used as measure of quality partition in the experiments. Because

Algorithm 1 Mini Batch K-Means algorithm

Given: k, mini-batch size b, iterations t, data set X
Initialize each c ∈ C with an x picked randomly from X
v ← 0
for $i \leftarrow 1$ **to** t **do**
  M ← b examples picked randomly from X
  for $x \in M$ **do**
    d[x] ← f(C,x)
  end
  for $x \in M$ **do**
    c ← d[x]
    v[c] ← v[c] + 1
    $\eta \leftarrow \frac{1}{v[c]}$
    c ← (1-η)c+ηx
  end
end

these algorithms only find a local solution for this function, similar values can be obtained by very different partitions. This suggests that other measures of agreement of partition should also be used for a better assessment of the quality of the partitions of the mini batch k-means algorithm.

The outline of this paper is as follows. In section 2, we discuss the characteristics of the experiments and how the artificial datasets have been generated. Section 3 analyzes the results for each one of the characteristics used to define the experiments. Finally, in section 4 general conclusion about the use of the algorithm will be presented.

2 Experimental setup

The implementation of k-means and minibatch k-means algorithms used in the experiments is the one available in the scikit-learn library [9]. We will assume that both algorithms use the initialization heuristics corresponding to the K-means++ algorithm ([1]) to reduce the initialization effects. Although, given that the mini batch k-means begins with a random sample of the dataset, the effect of this strategy will have less impact on the resulting partition.

For the experiments, synthetic datasets generated with different characteristics will be used, in order to perform testing of the algorithms in a wide range of types of datasets. Specifically, it will be tested the performance with different number of partitions, different cluster sizes, different spatial separation among the clusters, different number of relevant attributes and different ratio of noisy/irrelevant attributes. The datasets have been generated using the R package `clusterGeneration` that implements the methodology described in [10].

2.1 Synthetic datasets

The datasets generated correspond to sets of ellipsoidal clusters described by attributes modeled using gaussian distributions. These attributes can be divided into relevant and irrelevant attributes. The relevant attributes are the same set for all clusters and are generated independently using random orthonormal matrices. Also, random rotations are applied to the generated clusters. This means that each cluster has ellipsoidal shape with axis oriented in arbitrary directions. The irrelevant attributes are randomly generated using gaussian distributions with a variance similar to the sample variance of the relevant attributes.

The methodology for generating the data allows to control the minimum degree of separation for a cluster respect to the other clusters. This makes possible to generate datasets with different degrees

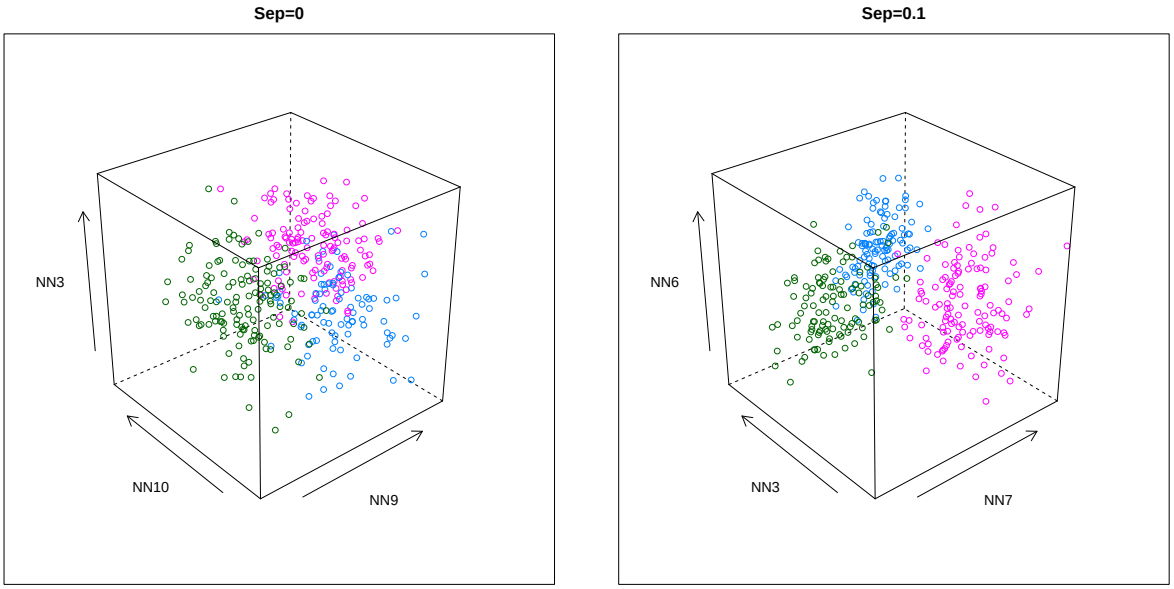

Fig. 1: Synthetic datasets with 3 clusters and separation 0 and 0.1

of separation, from highly overlapped to well separated clusters. Figure 1 shows two datasets using 3 relevant attributes with 3 clusters and different separation.

In the experiments we will vary the number of clusters, the number of examples in the clusters, the number of attributes, the ratio between the number of relevant and noisy attributes and the degree of separation between pairs of clusters. Given a cluster size, the size of all the clusters in a dataset is approximately the same.

Specifically, datasets have been generated for 5, 10 and 20 clusters with around 5000, 10000 and 20000 examples per cluster. The number of relevant attributes used is 10, 20 and 40. To test the effect of noise, datasets with additional irrelevant attributes have been generated adding 10%, 20% and 30% noisy attributes. Clusters are also generated with different degrees of separation, using 0 and 0.1 as the values of the parameter that controls the degree of separation in the dataset generation program. That accounts for highly and slightly overlapped clusters respectively.

2.2 Performance measures

In [11] the comparison of performance between both algorithm was measured using convergence time versus the value of the k-means objective function. Using only the objective function of k-means does not give a complete idea of the coincidence between the partitions obtained using both algorithms. The proposal is to use also validity measures used to compare the behavior of cluster algorithms from the clustering literature.

Cluster validity indices [4] are used to assess the validity of the partitions obtained by a clustering algorithm by comparing its characteristics to a reference or ideal partition. These indices give a relative or absolute value to the similarity among different partitions of the same data.

These indices can be classified in *external criteria*, *internal criteria* and *relative criteria*. The first ones assume that the evaluation of a partition is based on a pre-specified structure that reflects the natural structure of the dataset. For the second ones, the evaluation is performed in terms of quantities obtained from the values of the attributes that describe the clusters, these values are used to assess how separated and compact the clusters are. The last ones assume that the evaluation is performed comparing the partition to other partitions obtained using different parameters of the clustering algorithm or from other clustering algorithms. In our approach, we are going to use the three kinds of indices.

External and relative validity indices allow to compare different partitions independently of the type of the attributes used to describe the dataset, the distance/similarity function used to compare

the examples or the model used by the clustering algorithm to represent the clusters. In order to do that, only the cluster co-association between pairs of the examples is used.

The basis of these indices is that pairs of examples, when belong to a natural cluster in the data, usually appear in the same group when different partitions are obtained. This means that two partitions that maintain these co-association are more similar and represent a similar structure for the dataset.

In order to compute these indices the coincidence of each pair of examples in the groups of two clusterings has to be counted, there are four possible cases:

- The two examples belong to the same class in both partitions (a)
- The two examples belong to the same class in the first partition, but not in the second (b)
- The two examples belong to the same class in the second partition, but not in the first (c)
- The two examples belong to different classes in both partitions (d)

From this values different similarity indices can be defined for comparing two partitions, like the Rand statistic, the Jaccard coefficient or the Fowlkes & Mallows index. A frequently used measure in the literature is the Adjusted Rand statistic ([6]), that is a corrected for chance variation of the Rand statistic:

$$ARand = \frac{a - \frac{(a+c)(a+b)}{a+b+c+d}}{\frac{(a+c)+(a+b)}{2} - \frac{(a+b)(a+c)}{a+b+c+d}}$$

In our experiments we are going to use this index to measure the similarity of the partitions obtained by k-means and mini batch k-means to each other, using the measure as a relative criteria. We are also going to use this measure as an external criteria to compare the partitions of both algorithms to the true partition since it is available.

The objective function minimized by k-means as suggested by [11] can be used as internal criteria. But in this case, given that the value of this measure is unbound, we are going to use the quotient between the value of this measure obtained by both algorithm as a relative criteria:

$$DistortionRatio(C_K, C_{MBK}) = \frac{Distortion(C_{MBK})}{Distortion(C_K)}$$

When the value of this ratio is larger than 1, the quality of the partition from the mini batch K-means will be worse than the partition from K-means and the opposite if it is lower than one.

3 Results analysis

For each combination of number of clusters, number of attributes, number of examples, level of separation and level of noise, ten datasets were generated, for a total of 2160 datasets. Each run of the K-means and mini batch K-means algorithms for a dataset was repeated 10 times and the best result was kept.

As mentioned in the previous section, three criteria will be used to assess the performance of the mini batch k-means algorithm. The first one will be the ratio between the k-means and mini batch k-means objective function final values. The objective function for both algorithms is the sum for all the examples of the square euclidean distance to its nearest centroid (distortion).

A value closer to one for this ratio does not mean that the partitions are identical, because similar values can be obtained for different solutions. Because of this, the agreement among the partitions obtained by both algorithms and the agreement to the true partition measured by the adjusted rand index will also be used. Due to that the computational cost of this measure is quadratic on the number of examples, the value will be approximated using a sample of the 10% of the examples.

The different graphics in the next sections will show these measures for different batch sizes to also assess the effect of this parameter in the quality of the result. All results presented are the average for ten datasets of the measures used.

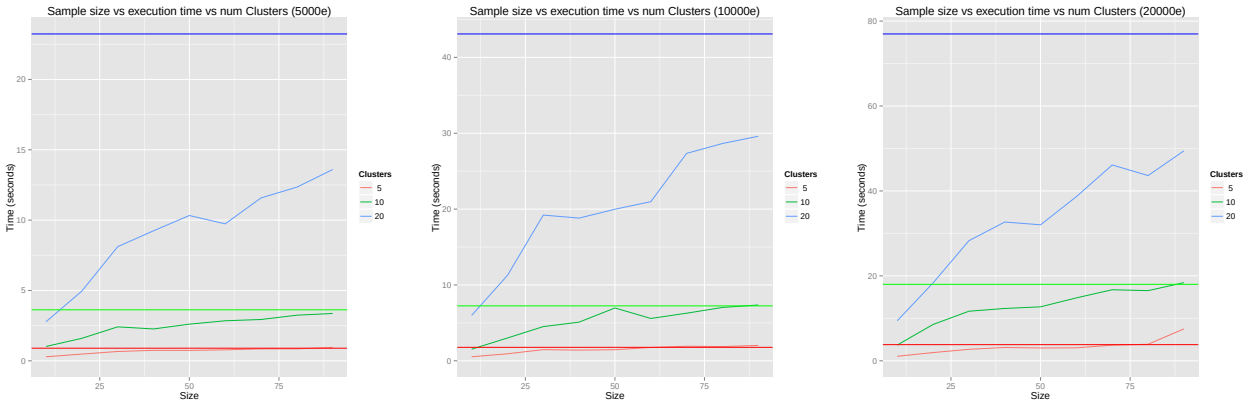

Fig. 2: Execution time for different number of clusters and cluster sizes

3.1 Number of clusters

The graphics shown in this section compare the effect of the number of clusters. Datasets with clusters of different sizes and slightly separated are used.

An expected result is that the time needed for convergence using only a sample of the dataset is less than the time needed using all the data. In figure 2 it is shown the time for convergence using different minibatch sizes (from 10% to 90% of the dataset) and different number of clusters for a fixed cluster size. In the graphic is marked as an horizontal line the time needed by k-means for each different number of clusters.

As the number clusters and the number of examples increase, the relative saving in computational time also increases. The saving in computational time it is more noticeable only when the number of clusters is very large. The effect of the batch size in the computational time is also more evident when the number of clusters is larger. From the graphic it can be seen that the computational time increases linearly as the size of the dataset increases linearly (doubling the number of examples doubles the time needed for both algorithms). There is not an evident reason because the computational time of the mini batch k-means algorithm is drastically reduced when the number of clusters is large.

The effect of the number of clusters in the ratio of the evaluation function is shown in figure 3. As the number of clusters increases, the ratio also increases. This means that to obtain a similar evaluation function gets harder when the number of clusters is high. Almost identical behavior can be observed for the similarity to the resulting k-means partition measured with the adjusted Rand index as can be seen in figure 4. It can be concluded that, increasing the number of clusters, decreases the similarity of the mini batch k-means solution to the k-means solution.

Figure 5 shows the similarity of the k-means partitions (straight line) and the mini batch partitions to the true partition. It can be observed that the partitions obtained with the mini batch k-means diverge from the true partition in the same proportion they diverge from the k-means partitions. This indicates that these partitions cannot get closer to the optimal solution than the solutions of k-means and that the distance increases with the number of clusters.

Comparing the values between the ratio of the objective function and the adjusted rand index, it can be said that, despite that the agreement between the partitions decreases as the number of clusters increases, the objective function does not degrade at the same rate. This could mean that the final partitions are different, but closer in quality. If the goal is to obtain the same results than the k-means algorithm, it is obvious from the measures that the increase in the number of clusters will make this goal more and more difficult. Although, this difference in the rate of decrease could be an effect of the measures.

Respect to the size of the random batches, for all the experiments, the variance of the measures is very large and it looks like the quality of the final result is not highly related to this size. The reason of this large variance is probably that the initialization strategy has less stabilization effect on the solutions, because its computation is done in a random sample instead of using the whole dataset as

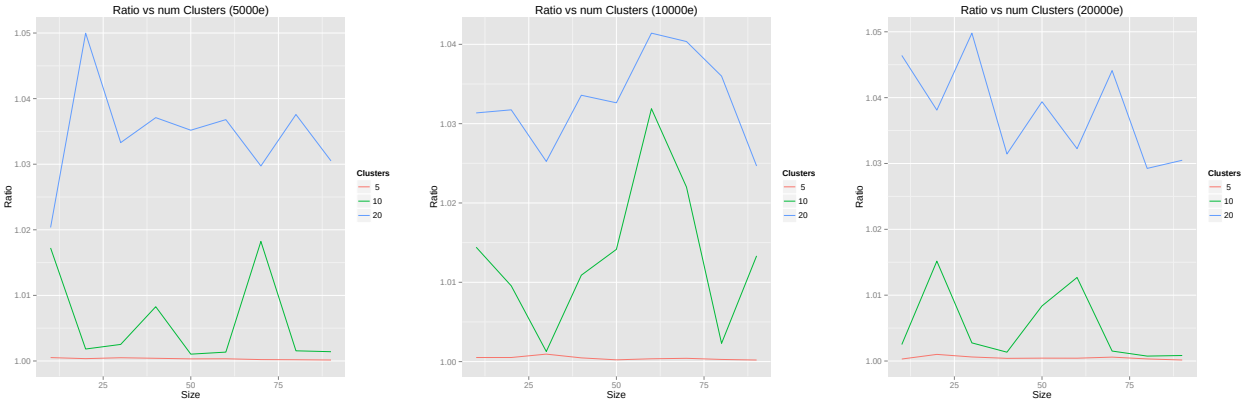

Fig. 3: Ratio of quality function for different number of clusters and example data size

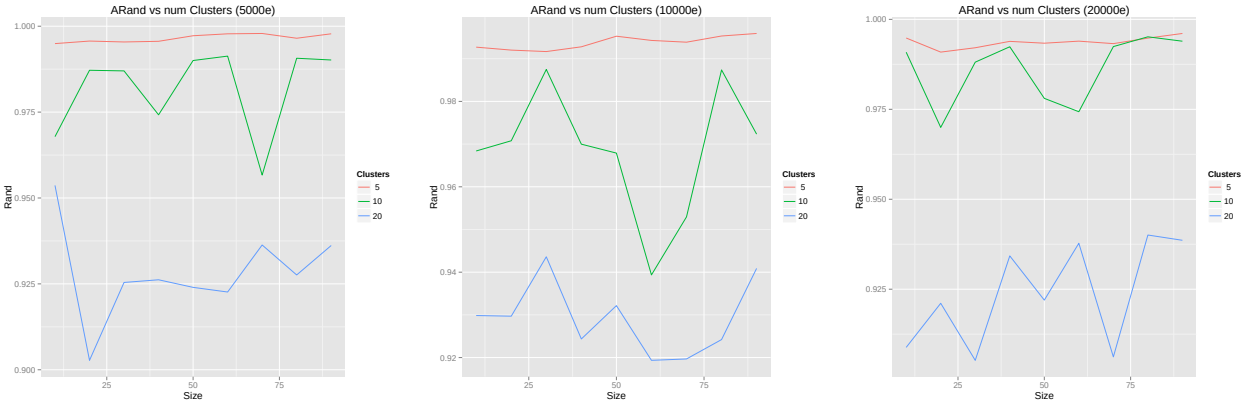

Fig. 4: Similarity for different number of clusters and example data size to k-means partition

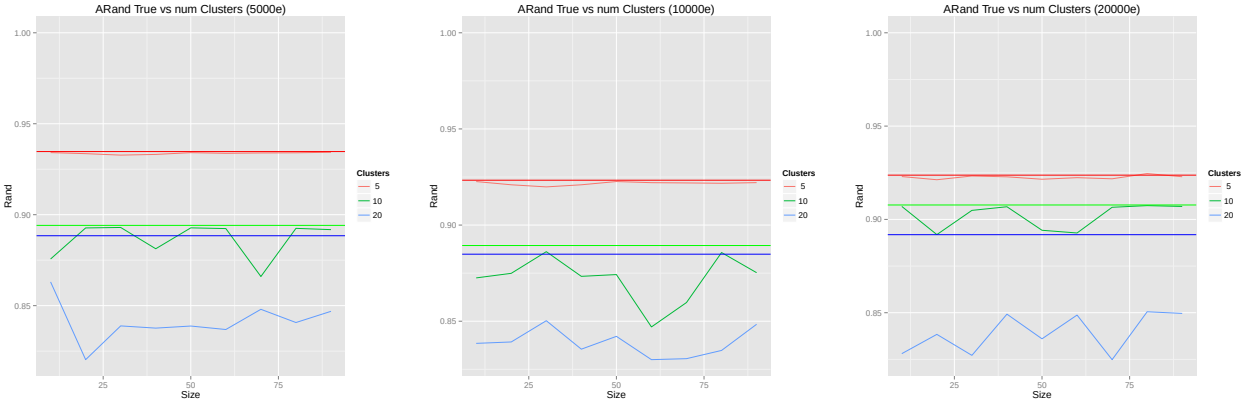

Fig. 5: Similarity for different number of clusters and example data size to true partition

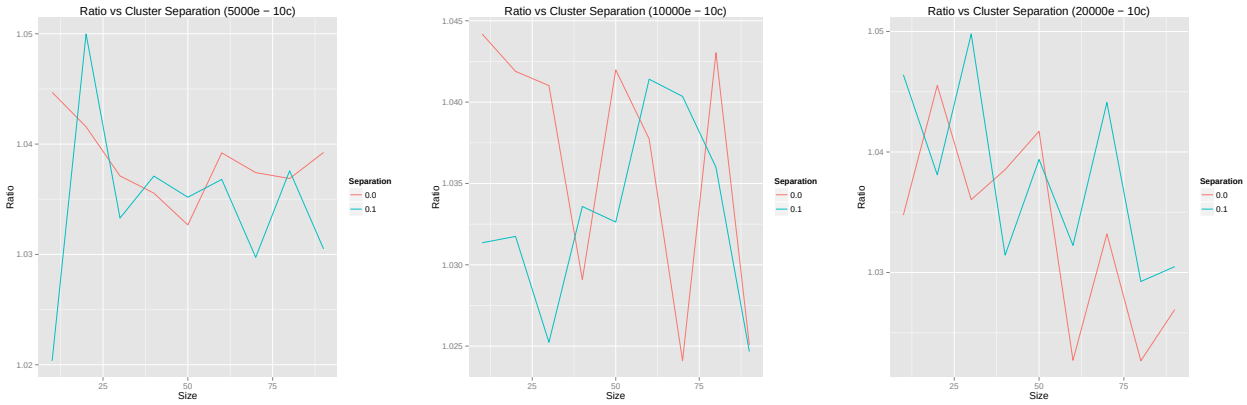

Fig. 6: Ratio of quality function for different cluster separation and example data size

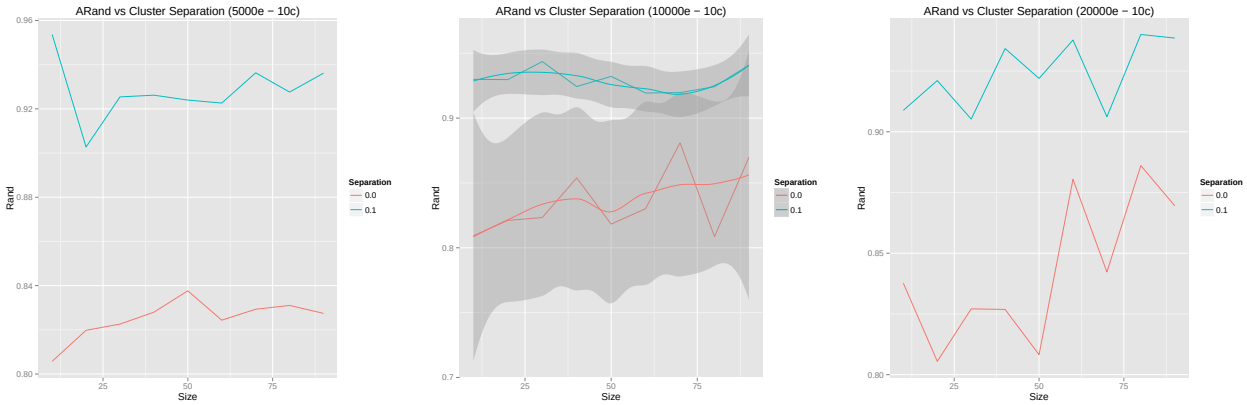

Fig. 7: Similarity for different cluster separation and example data size

is done for k-means. This little effect of the size of the batch in the final quality means that a large computational saving can be obtained by using a small value for this parameter.

3.2 Cluster Separation

The graphics shown in this section compare the effect of cluster separation. Only experiments with 10 clusters of different cluster sizes are presented. The partitions are obtained for highly (0.0) and slightly (0.1) overlapped clusters.

The ratio of the objective function (figure 6) does not always show a very clear difference between the two algorithms. Intuitively, the separation of the clusters should affect the quality of the clusters. This hints that perhaps this measure is not completely adequate for comparing the partitions in this circumstances. The difference is more clear for the adjusted Rand index (figure 7). The datasets with highly overlapped clusters result in lower quality partitions than ones with the slightly overlapped clusters, despite the apparent similarity in objective function value. This difference maintains for different clusters sizes.

In real datasets, is more usual to have overlapped clusters than to have separated clusters. This means that, the more overlapped are the clusters, the more will diverge the partition obtained by mini batch k-means from the one obtained from k-means. This is something to take in account in practice.

The effect of the size of the mini batch is more clear for the highly overlapped clusters. As the size of the batch increases, the variance of the result also increases. This can be observed in figure 7, there is a clear difference on the variability of the solutions for different overlapping when the size of the batch increases. In this figure also can be observed that increasing the size of the batch impacts positively in the quality of the final partition when the clusters are highly overlapped. This tendency does not appear for only slightly overlapped clusters.

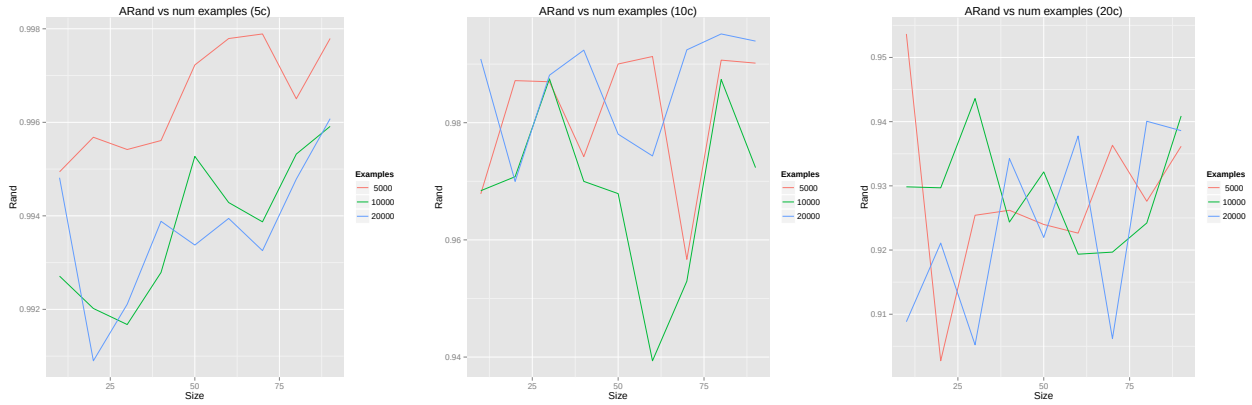

Fig. 8: Similarity to k-means partition for different cluster size

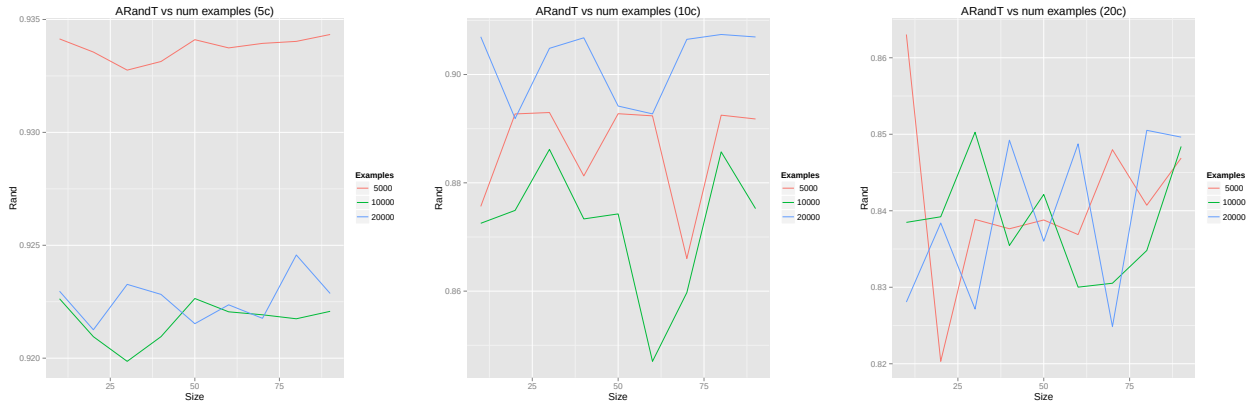

Fig. 9: Similarity to true partition for different cluster size

3.3 Number of examples

The graphics shown in this section compare the effect of cluster size. Experiments with slightly separated clusters are presented.

Figure 8 shows the rand index for different cluster sizes and number of clusters. Except for 5 clusters, the other graphics show an equivalent behavior despite the size of the clusters. This means that increasing the size of the dataset has low impact in the similarity of the final partition compared with the result of k-means.

Figure 9 shows a slightly different behavior. These graphics compare the similarity of mini batch k-means partitions to the true partition. It seems that smaller clusters are able to maintain better the similarity to the true partition, as long as the number of clusters is low. The reason of this behavior probably is that for smaller sizes, there are less difference among the subsets of examples in consecutive iterations. Increasing the number of clusters equalizes the behavior respect to the number of examples.

Another reason for this behavior could be also the way the algorithm updates the clusters. Having a larger dataset decreases faster the influence of new examples each iteration because the total number of processes examples is used in the learning rate. This makes the algorithm to converge quickly, so the initial batches have a larger impact on the final partition.

This makes the strategy of mini batch k-means less adequate for larger datasets, because the initial random batches have too much influence on the results. This problem suggest that changing how the learning rate is computed reducing their relative influence could result in a better quality for the final partition.

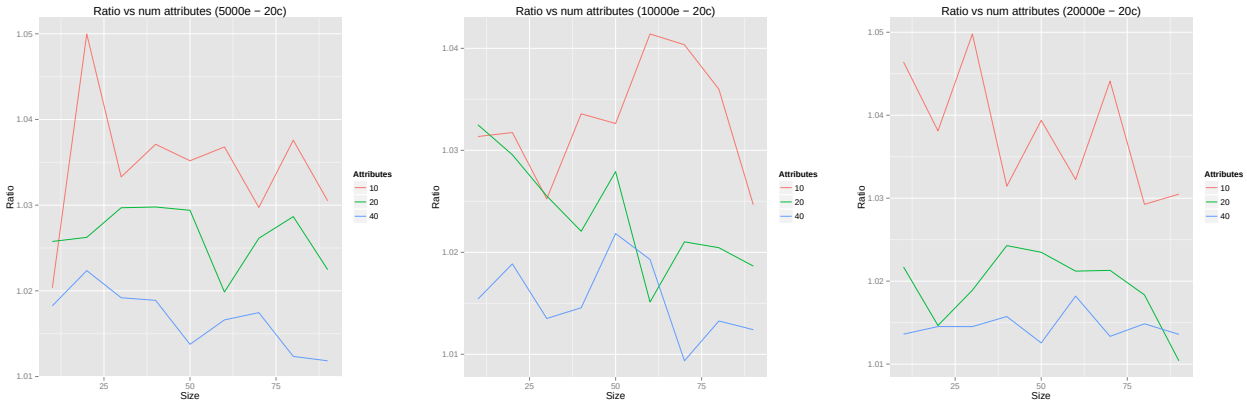

Fig. 10: Ratio of quality function for different number of attributes and example data size

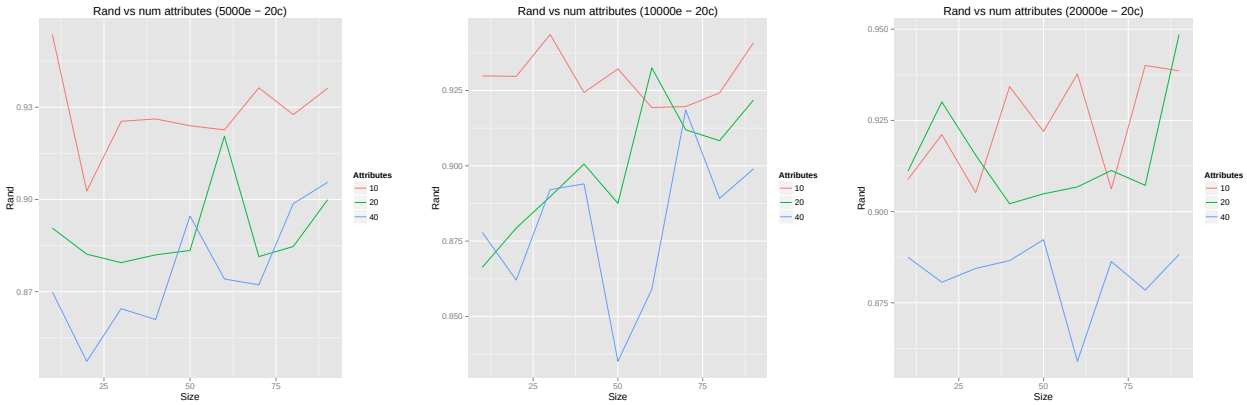

Fig. 11: Similarity to k-means partitions for different number of attributes and example data size

3.4 Number of attributes

The graphics shown in this section compare the effect of the number of attributes. Only experiments with 20 slightly separated clusters and different cluster size are presented.

Figure 10 shows the ratio between the evaluation function of the solutions for the two algorithms using different number of attributes to generate the clusters. It appears to be a difference among partitions with different attributes, the more attributes are, the more similar are the final partitions in quality. This behavior does not appear when the adjusted rand index is used as shown in figure 11. The number of attributes has the opposite influence in the similarity according to this measure.

A possible explanation for this contradiction is in the objective function itself. This measure uses the square of the distances to the centroids. As an effect of the curse of the dimensionality, distances among examples tend to get more similar when the number of attributes increases. As a consequence, the ratio of the objective functions tends to one for larger number of attributes. This makes the quality measure less adequate for comparing partitions with respect to the number of attributes than the adjusted rand index. The computation of this measure is independent of the examples representation, so the differences in this measure represents changes in the characteristics of the partitions and not in their description.

Intuition suggest that a larger number of attributes should make more difficult to discover the clusters, as the adjusted rand index indicates. The conclusion is that the mini batch k-means will loss quality as a function of the number of attributes. This loss of quality seems independent of the number of examples because the range of the values of this measure are the same. It seems, though, that the variance of this value for different batch sizes is reduced when a larger dataset is used.

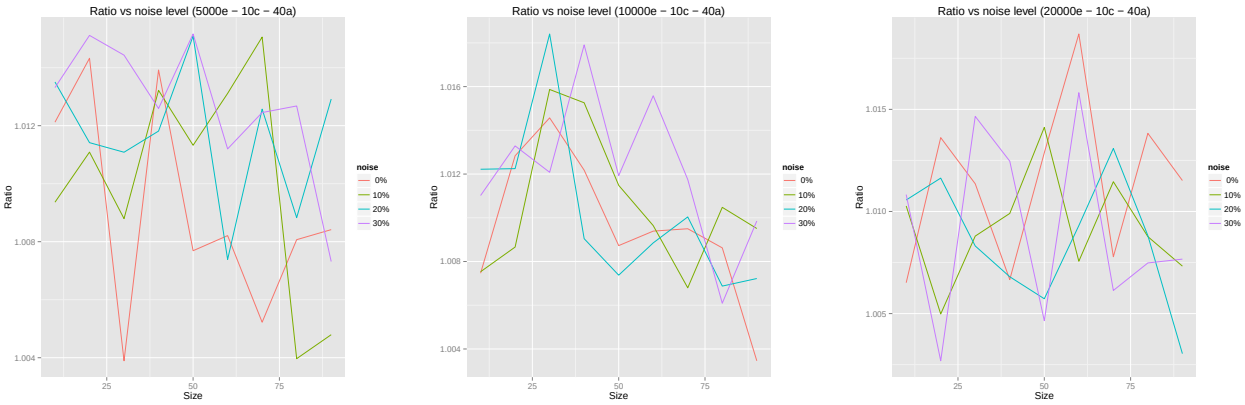

Fig. 12: Ratio of quality function for different noise level

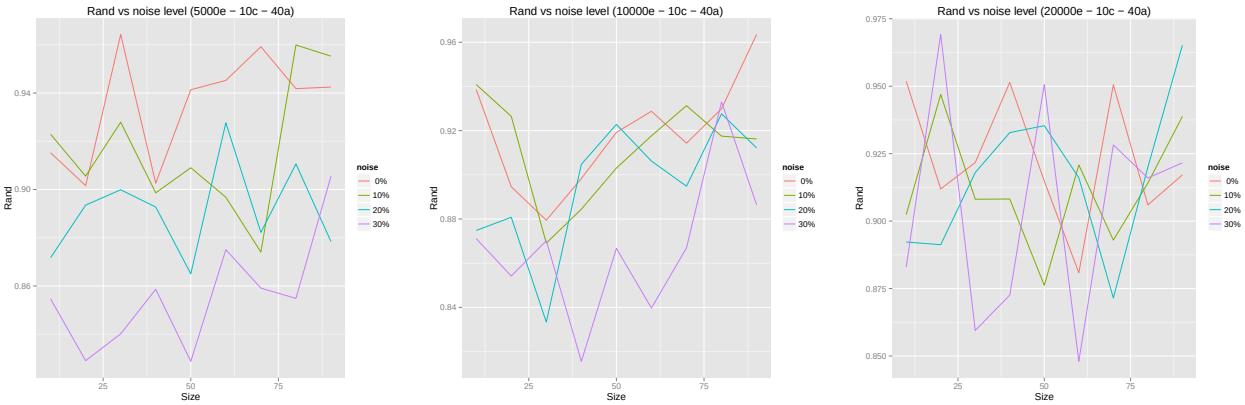

Fig. 13: Similarity for different noise level

3.5 Level of noise

The graphics shown in this section compare the effect of the number of irrelevant/noisy attributes. Only experiments with 10 slightly separated clusters, different cluster sizes and 40 attributes are presented.

Figures 12 and 13 show the influence of irrelevant attributes in the results. Different levels of noise (from 10% to 30%) seem to have a similar influence in the ratio of the objective function and the similarity among partitions. It does not appear a clear difference for different levels of noise except for partitions with 5000 examples according to the similarity measure with the adjusted rand. Augmenting the size of the dataset equalizes the effect of the noise, obtaining very close values independently of the noise level. Larger datasets increase the variance of the results for different batch sizes.

A possible explanation is that, being the clusters slightly separated, these limited levels of noise does not have a large impact in the final partition. It could be concluded that the mini batch algorithm is tolerant to a reasonable amount of noise, having little impact on the quality of the resulting partitions.

4 Conclusions and future work

From the experiments, a first conclusion is that to use the sum of square euclidean distances to the centroids to compare the quality of partitions from k-means and mini batch k-means as proposed in [11] is not adequate. Inconsistent conclusions can be drawn for the influence of the number of attributes from this measure. The adjusted rand index seems a better choice because is independent of the representation of the data and clusters.

Mini batch k-means has the main advantage of reducing of computational cost of finding a partition. This cost is proportional to the size of the sample batch used and this difference is more evident when

the number of clusters is larger.

The number of clusters has an important impact in the difference between the partition obtained by k-means and mini batch k-means. This difference ranges from 2% of loss of quality for a small number of clusters (less than 10) to more than 8% for a larger number of clusters (more than 20). This difference also shows when comparing the partitions of both algorithms to the true partition. In fact, as the number of clusters increases, their respective difference to the true partition also increases. This means that it is not probable that applying the mini batch k-means algorithm to very large dataset with a large number of clusters will result in equivalent partitions to the ones from k-means.

Also the level of overlapping among the clusters has a large influence in the similarity, only observable using the adjusted rand index. Large overlapping means a decrease in the relative similarity between the partitions obtained by both algorithms.

The number of attributes has also a small influence on the similarity, obtaining less similar partitions as the number of attributes grows, but this influence is less noticeable than the previous factors.

The size of the dataset and limited levels of irrelevant attributes does not present a clear impact on the quality of the partitions. This is important because the aim of this algorithm is to process large datasets.

Given these results, it is worth exploring other strategies related to the selection of the examples for the mini batches or to how the update of the centroids is computed during the iterations. The goal of this strategies would be to reduce the impact of the number of clusters and to obtain partitions more similar to the ones obtained by k-means.

Several ideas can be borrowed from the neural networks literature, as for example to divide the dataset in folds and use alternatively each fold for training, to use larger batches at the beginning of the algorithm and to reduce progressively the size of the batches as convergence approach or to change the way the learning rate is computed, so the initial batches has less impact on the final partition.

References

[1] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *SODA*, pages 1027–1035. SIAM, 2007.

[2] Paul S. Bradley, Usama M. Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In Rakesh Agrawal, Paul E. Stolorz, and Gregory Piatetsky-Shapiro, editors, *KDD*, pages 9–15. AAAI Press, 1998.

[3] Charles Elkan. Using the triangle inequality to accelerate k-means. In Tom Fawcett and Nina Mishra, editors, *ICML*, pages 147–153. AAAI Press, 2003.

[4] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001. 10.1023/A:1012801612483.

[5] John A. Hartigan. *Clustering Algorithms*. Wiley series in probability and mathematical statistics. John Wiley & Sons, 1975.

[6] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[7] Anil K. Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[8] Yann LeCuni, Leon Bottoui, Genevieve B. Orr, and Klaus-Robert Miuller. Efficient backprop. Image Processing Research Department AT& T Labs-Research, 100 Schulz Drive, Red Bank, NJ 07701-7033, USA 2 Willamette University, 900 State Street, Salem, OR 97301, USA, 1998.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot,

and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[10] Weiliang Qiu and Harry Joe. Generation of random clusters with specified degree of separation. *Journal of Classification*, 23(2):315–334, 2006.

[11] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM, 2010.