Periodicals of Engineering and Natural Sciences ISSN 2303-4521
Vol. 7, No. 2, June 2019, pp.448-457
Available online at: http://pen.ius.edu.ba

Clustering algorithms subjected to K-mean and gaussian mixture model on multidimensional data set

Saadaldeen Rashid Ahmed Ahmed[1], Israa Al_Barazanchi[2], Zahraa A. Jaaz[3], Haider Rasheed Abdulshaheed[4]

[1]Information technology - Altinbas university, İstanbul
[2,4]Baghdad College of Economic Sciences University, Baghdad
[3]College of Science - Computer department - AlNahrain University, Baghdad

Article Info

Received Jan 29, 2019

Keyword:

Datamining, k-means, Data processing, clustering, Gaussian mixture model, Data extraction

ABSTRACT

This paper explored the method of clustering. Two main categories of algorithms will be used, namely k-means and Gaussian Mixture Model clustering. We will look at algorithms within thesis categories and what types of problems they solve, as well as what methods could be used to determine the number of clusters. Finally, we will test the algorithms out using sparse multidimensional data acquired from the usage of a video games sales all around the world, we categories the sales in three main standards of high sales, medium sales and low sales, showing that a simple implementation can achieve nontrivial results. The result will be presented in the form of an evaluation of there is potential for online clustering of video games sales. We will also discuss some task specific improvements and which approach is most suitable.

Corresponding Author:

Saadaldeen rashid ahmed ahmed[1]
Information technology
Altinbas university
İstanbul, Turkey, https://orcid.org/0000-0003-2259-7437
Saadaljanabi78@gmail.com.

1. Introduction

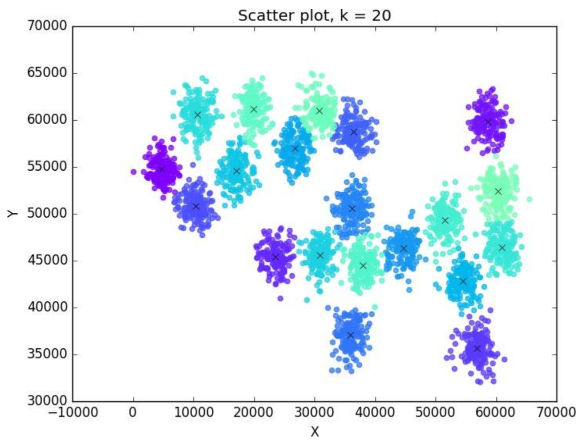
Given a set of data with different properties, say that they have different spatial location, sales, ratings etc., it might be the case that there is information about the relationship between data point not obvious on inspection. The method of clustering is a way to divide data into groups which have a high similarity to other members of the cluster, and low similarity to members in other clusters. This similarity can be expressed in the form of a sales function. However, this division is often ambiguous, and relies on the fact that the dissimilarities are distinguishable. [11, p. 201] To give an example of this relationship, let's assume that the data in question is produced by random variables X_1, .., X_n. In figure 1 we have an example where the variables have a uniform distribution, and if an algorithm find structure in the data it would only be a false positive. In figure 2 on the other hand, we see example of random distributions over different intervals. Here we can by inspection find two clusters, even though internally there are no significant correlation.

Depending on what kind of data that is used for clustering, as well as the amount of data available, there will be a difference in performance given different methods. There is also a question of time complexity, since the k-center clustering problem is NP-hard [3] we also have to consider running time as a factor. To prove the NP-hardness one can reduce it to the 3-GMM problem. This is omitted her, but in [14] we can see an example of a proof. In this report two categories of such methods will be explored, k-means and Gaussian Mixture Model clustering. Specifically, we will focus on how they can be applied to solving an online task. The task in question is determining how many hands are acting on a video games sale. Each data point corresponds to a finger, or digit, being registered on the surface. This generates x and y coordinates, x and y sales and rating. The data also contains pre-processed tracking of individual points which will not be used as a clustering parameter.

1.1 K-means

When referring to k-means, many find it synonymous to Stuart P. Lloyd's least square algorithm from 1982 [13]. Since k-means is a NP hard problem, we need some way of reducing the time complexity, which Lloyd's algorithm does. The issue is however that it does not give any guaranties to how close. We can come. In fact, this algorithm can produce arbitrarily bad clusters [3].

The algorithm is as follows: Start by creating k empty clusters and for each cluster, choose at random a data point making the coordinates of this point the center of the cluster. Proceed by iterating over each data point and calculate which cluster it is closes to and adding the point to closest cluster. When this is done, recalculate the center of each cluster by taking the mean of the assigned points. Continue this process until the centers of the clusters no longer changes.



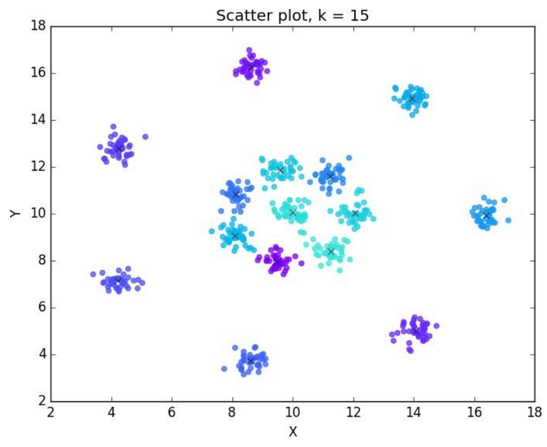


Figure 1: Successful convergence using k-means. Data set **Figure 2:** Successful convergence using k-means. Data set from [9,18]

1.2 Theory

We formulate the problem as follows: Define a set P of points $p_1, p_2, ..., p_n$, $p \in (\mathrm{R}^m, L_2)$, m being the number of features or dimensionality. For each pair p_i, p_j we have a sales given the function $d_{i,j} = sales(p_i, p_j)$. The assumption is all points are in metric space such that $s_{i,j} = s_{j,i}$, $s_{i,i} = 0$, $s_{i,j} \geq 0$ and $s_{i,k} \leq s_{i,j} + s_{j,}$
So, denoting features x_k, y_k, $k = 1, .., m$ from p_i, p_j respectively, we get the following expression:

$$sales(p_i, p_j) = ||p_i - p_j|| \tag{1}$$

The objective is to include each p_i in one of k clusters $C = c_1, c_2, ..., c_k$ determined by the sales function. The center $c_{\theta,i}$ can be calculated as follows. This minimization we can view as a cost function which is monotonically decreasing. This way, algorithm always converges. To show this, let c(t) , ..., c(t) , C(t), ..., C(t) denote centers.

$$sales(C(t+1), ..., C(t+1); c(t) , ..., c(t)) \leq sales(C(t), ..., C(t); c(t) , ..., c(t)) \tag{2}$$

We then proceed by recalculating the cluster center.

$$sales(C(t+1), ..., C(t+1); c(t+1), ..., c(t+1)) \leq sales(C(t+1), ..., C(t+1); c(t) , ..., c(t)) \tag{3}$$

1.3 K-Means with GMM

As we have discussed earlier, the k-means algorithm is a way to tackle the NP-hard nature of the problem of clustering, but it generates a new problem, namely accuracy. In [3] the k-means algorithm is presented as a way to get some guarantees on the accuracy of the k-means algorithm, through a randomized seeding technique. The bound that k-means proposes is as follows [6]:

For any set of data points, $EM[\varphi] \leq 8(link + 2)$

What the k-means adds to the original k-means is the fact that we can choose the initial clusters using a more strategic manner than simply choosing at random. The probabilities can be calculated as follows: Where D is the squared sales to the closest cluster center already in C. The paper also proves that k- mean is $Ø(log(k)$-Competitive, and that the analysis is in fact tight.

1.4 Gaussian Mixture Model

GMM clustering is an approach to solving the clustering problem which might not be the fastest but is very effective in certain cases. The version we have researched is the agglomerative GMM clustering which can be described as a bottom up approach to clustering. At initiation each data point is its own cluster. The algorithm then iterates over the clusters determining which pair is, for example, closes (single-link) or furthest away (complete-link) from each other [5]. These clusters will then be merged and form a new cluster. If no stopping criteria is used, the algorithm will eventually have one single cluster containing all data points. For the evaluation I have used the complete-link approach since it suited the data best. To support why this is relevant to this project, n we find that the complete-link GMM is an $Ø(log(k))$-approximation to the diameter k-clustering problem.

In figure 1 and 2 we find an example of a data sets from [7] and [8] respectively, where GMM clustering might perform better than others algorithm. I.e. when the data is not spherical, or there is a clear link between data points that cannot be captured that easily by statistical properties. However, the method is highly susceptible to outliers [19-22]. As one can see in the figures the convergence is not perfect and omitted are the next few iterations where the clusters become quite deformed.

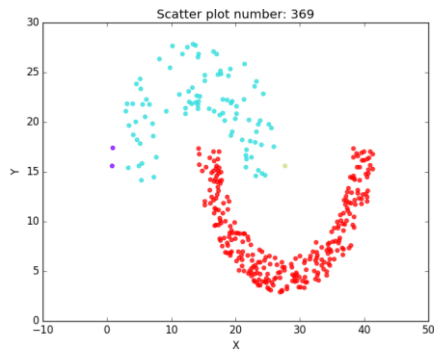


Figure 3: Convergence using single-link AHC. Data set [8,7].

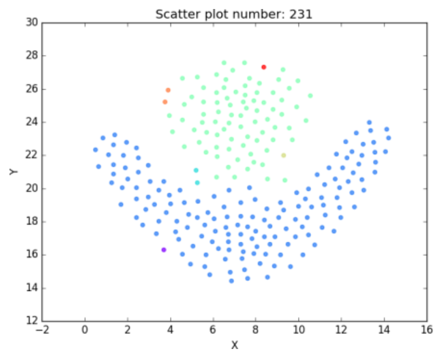


Figure 4: Convergence using single-link GMM. Data set

2. Methodology

The method is highly susceptible to outliers. As the convergence is not perfect and omitted are the next few iterations where the clusters become quite deformed, Lets define a set P of points $p_1, p_2, ..., p_n$, $p \in \mathrm{R}^m$, m being the dimensionality. For each pair p_i, p_j we can calculate a sales given the function $d_{i,j} = sales(p_i, p_j)$.The assumption is all points are in metric space such that $d_{i,j} = d_{j,i}$, $d_{i,i} = 0$, $d_{i,j} \geq 0$ and $d_{i,k} \leq d_{i,j} + d_{j,k}$. What we have to work with is a hollow lower triangular matrix M, which is to be search through. The objective is to include each p_i in one of k clusters $C = c_1, c_2, ..., c_k$ according to some criteria such as the ones discussed above.

We have employed the single-link clustering. To show the validity of this method we can draw a parallel to the problem of finding a minimum spanning tree. In [11, 22] we find an example of this using Kruskal's algorithm, where the analogous version of the algorithm would work as follows: Start with a strongly connected graph G where points $p_1, p_2, ..., p_n$ represent a vertices and for each point p_i, p_j we have an edge with the weight equal to $d_{i,j}$. Now, instead of merging, lets remove the heavy's edge in the graph. Do this k-1 times and the result will be a minimum spanning tree. [12]

2.1 Adding peripheral cluster

One addition to k-means that I have been experimenting with is to add a dummy cluster. The method is quite a non-scientific but could be an interesting addition to get a better score in practice. The idea is to add a cluster, for this problem we used a cluster positioned at 0 as a reference. Since it does not fit anywhere in the clusters that are active it is easily recognized as a cluster containing only this point. What it potentially does is that it makes the total variation within the rest of the clusters relatively smaller. Imagine having only one hand, a reasonable clustering is just having all fingers as there is own clusters, or maybe one for the thumb and one for the rest of the digit. Having two hands means we have a sale between hands that is larger than the sales between fingers, making it easier to identify as separate clusters. The idea would be to artificially create this sale.

2.2 Dataset Description

The dataset contains estimated Video games sales in different regions of World i.e; North America sales, European Union sales, Asia sales, Japan sales, global sales etc. Its year of release, Genre, critic score, critic count, user score, user count, Developer and Rating of video games. Based on these instances the clustering is done in three categories of sales in High, Medium and Low-sales clusters around the globe. For data set, both algorithms got a fairly good score. This is not very odd since the entire data set is supposed to function as a reference of an "easy" clustering task. Looking at the majority of the time we have a uniform sale from frame to frame. What we can identify from a purely ocular inspection is that the changes seem to happen when in a curve [12]. Data set seem to be the hardest of the data sets to clusters. Both k-means and Gaussian Mixture Model clustering returns a score of around 50 % in table 1. Also, in 2 the result is worse than for the other data sets. The conclusion that we can draw, is that there is a large alternation between 2 and 3 clusters. A large portion of the time two hands have no clear sales record, but instead they are aligned in parallel. In data set we find a middle ground of the previous two clusters of high and medium sales. There are no obvious causes for not converging, except the one we discussed in Data set.

2.3 Conditioning on previous frames

Continuing the thought from "Evaluating the algorithms", one change that could be made to the sales function is a measurement of similarity to the clusters in previous frames. Seeing as though the data contains the tracking of individual fingers, we could add a punishment when converging to a cluster not in the previous frame. For example, let us say that we have continuously found some data points in one cluster, then we get closer to another cluster. At some point we could face a situation where the sales between data points are very similar through both clusters such that $var(C_i)$ $var(C_j)$ $var(C_i + C_j)$. At this point we might also have the same sales and it would be reasonable that the convergence would be to find one cluster. However, we have this information from previous frames saying that data points belong to different clusters, making it even more probable that they should be separated.

What I do now is to cluster using a cluster function Cf with P_i as input, but in reality we should input P_i, Cf (P_{i-1}), .., Cf (P_1) since there is a high correlation between different frames. This is too much data to process, but one could extend the method I have been using to recording sales and limit the input to the previous frame. In some sense we could employ a Markov assumption as follows:

$$Cf(P_i|P_{i-1},...,P_1) \approx Cf(P_i|P_{i-1}) \qquad (4)$$

To support this lets compare table 1 and 2. We can conclude from table 2 that the clustering is fairly consistent. However, if we are constantly converging to the wrong clusters, it's not really a good measure. Looking at table 1 however, speaks to the fact that we actually do find the right clustering number quite often.

2.4 Adding constraints to sales function

One somewhat non algorithmic addition one could make to the sales function is to add the physical constraint of the assignment. For example, a cluster in this setting is a hand, which we can assume has no more than six digits. Also, spacing between individual digits has a limit. Imagine the sales between your first and fifth digit, one could add a max that is around some 99 percentiles of sales found in games.

2.5 Combinatorial clustering in K-Mean and GMM

When doing online clustering, speed is a factor. To speed up the method I have used a combinatorial GMM (sequential, agglomerative, Gaussian Mixture Model, non-overlapping) clustering method. As suggested by [5][12] we can describe the sales as follows:

$$s(h,k) = \alpha_i s(i,k) + \alpha_j s(j,k) + \beta s(i,j) + \gamma|s(i,k) - s(j,k)| \qquad (5)$$

Where h is the new GMM cluster and k being a cluster already defined in our set C. The parameters α, β, γ can be set in the following way:

- Single-linkage: $\alpha = 0.5, \beta = 0\ \gamma = -0.5$
- Complete linkage $\alpha = 0.5, \beta = 0\ \gamma = 0.5$

When analyzing how a AHC clustering algorithm would perform using the description used in the introduction, one can conclude that it will be $0(n_3)$. Since n clusters will potentially be reduced to 1, we need n iterations. For each iteration we need to compare each pair i,j which in takes ^{2} iterations. The combinatorial GMM algorithm does preform a bit better $Ø(n^2 log(n))$ [5]. It more practical since we do not have to recalculate sales in the same sense and uses a dynamic programming technique and save the previous sales.

2.6 Determining k-Cutoff in K-Means

When determining the most likely number of clusters, a common way is to look at a dendrogram [4]. An example can be seen in figure 1 where one can clearly see how clusters are merged. The dendrograms also shows the dissimilarity between the two clusters. A common approach is to use a cutoff value *k*, but for this assignment finding the largest sales was a fairly accurate way to guess k. In some sense we are allowing minor adjustments to the clustering, but when a big adjustment happens, it is interpreting as a "derailing" from the true clustering. It turns out as we will see in proceeding sections, that this is a fairly good assumption for convergence. In mathematical terms we can describe the selection process as follows:

$\Delta_{d,i} = s_i - s_{i-1}\ i = 2, ..., k_{max}$

$$k = i \in max(\Delta_{d,i}) \qquad (6)$$

Where s_i is the sales between the two clusters who are to be merged at iteration i.

2.7 Data Processing for K-Mean

The code employed comes straight from the k-means theory. The idea of using K-Mean as a way of determining k comes from [17].

1. For k = *kmin,...,kmax*
2. Select one data point at random to be the initial cluster. Use the selection method from k-means
3. Run until convergence:
4. For all points, check which cluster is closest
5. Update cluster center in accordance to the new data points
6. Check sales to clusters previous location, break if below a threshold Calculate K-Means
7. Return the result corresponding to the best K-Means

2.8 Data Processing for GMM

When implementing the algorithm, we have followed the GMM method described in [5] originally from Anderberg 1973 [2] and goes as follows:

1. Create a priority queue P for iteration *1 , ..., N − 2*
2. Search for the smallest value in dataset using P. Replace *Ci, Cj* with a new cluster *Ch.* Update D and P to account for changes.
3. Check for any big changes, chose k corresponding to the clustering that was present before the largest change.

3. Results

The data sets, we have used for the evaluation is provided by Universal Dataset for Clustering. There are three recordings from one of their Video-Games Sales. The data points are the registered sales active on the video games sales, and the clustering task will be to assign each of these data points to a cluster representing one sales point. Each sale is generating a x and y coordinate, a x and y sales as well as rating per frame. The Data also contains pre-processed tracking of individual points. The games have been sampled at rate of 150 per sales record.

At this stage the question is focused on if there is a good way to make a correct clustering, but when formulating the problem one should note that there is a need for fast processing time due to the fact that any practical application will be online. The idea behind the three different recordings is that they should differ in how hard it is to distinguish between the clusters. To start-of, both algorithms, we have used for this evaluation are in some sense very simple. The result produced are good because of the data used is generated by the authors and used in the right context. In the discussion, we have proposed extensions to the algorithms as to better suit this specific problem.

When running the algorithm, we assign a weight *k* to the parameters where k_1 being spatial coordinates, k_2 being sales and k_3 being rating. In table 2 we have presented the result of a simple test. We chose the frames containing four hands and looked at how many times the algorithm guessed right. In the table we can see average percentage (10 iterations), what method was used, what data set was used as well as the values of *k*.

Table 1: Result from cluster number guess.

% correct	Method	Data set
0.5580227982782385	GMM	Video Game Sales
0.5728032024750462	GMM	Video Game Sales
0.5143995189260154	GMM	Video Game Sales
0.5359751037344395	k-Mean	Video Game Sales
0.18595132743362813	k-Mean	Video Game Sales
0.1048153618906944	k-Mean	Video Game Sales

To analyze this data there is a need for tracking. As we have mentioned previously, there is already a pre-processed tracking of individual data points across frames. What we want is a way to check how the current frame F_i relates to the previous frame F_{i-1}. Define the function Cf to be a function that given a set of data points from a frame P_i returns a number of clusters. To analyze how well the algorithms tracks between frames we have used the following test: For every iteration, we converge to a number of clusters k_i. We then proceed to go through the previous and current clusters and match the best clusters to each other. I then calculate the cut between the points in the current and previous cluster and count them up, dividing by the total amount of data points. If $k_i = k_{i-1}$ we find the best fitting matches and ignore the rest of the clusters. The method I use for comparing clusters between frames is the mean value of the points in the clusters. We deem the sales large to be the same cluster at some cut off threshold which is the same for both algorithms. We would say that it's quite noticeable when the clusters are not sequential since the sales is very small when two consecutive frames are converged correctly, and a factor of 10 larger when it does not. Since the sampling time is constant, there is upper bound on how far a point can change position between frames.

```
Length of Cluster 1:  458

Centroid Cluster 1: [0.5359751037344395, 6.848132780082986]

Length of Cluster2:  782

Centroid Cluster2:  [0.18595132743362813, 2.1085066371681456]

Length of Cluster 3:  1395

Centroid Cluster 3: [0.1048153618906944, 1.1106942392909938]


Final Mean of Gaussian1:  [-0.2529028734625937, 0.018147916362775733]

Final Amplitude of Gaussian1:  0.5580227982782385

Covariance Matrix:  [4.43918853 2.34465405]

 [2.34465405 3.96456541]

Final Mean of Gaussian 2:  [2.5938744098777353, 1.3009754577208656]

Final Amplitude of Gaussian 2:  0.5728032024750462

Covariance Matrix:  [8.16237026 5.49537361]

 [5.49537361 6.32953227]


Final Mean of Gaussian 3:  [4.560270472056536, 4.269452906123397]

Final Amplitude of Gaussian 3:  0.5143995189260154

Covariance Matrix:  [5.76946192 3.26290271]
```

[3.26290271 5.13100977]

The result of the test is shown in table 2, were we have an averaged percentage (10 iterations) of correct transfers between frames, the method used, what data set was used as well as the values of *k*.

Table 2: Result from the data points tracking

% correct	Method	Data set	*k*
0.834865227719	GMM	Video Game Sales	1
0.754034713764	GMM	Video Game Sales	2
0.768226840163	GMM	Video Game Sales	3
0.911985983716	k-Mean	Video Game Sales	1
0.712328247172	k-Mean	Video Game Sales	2
0.841447317338	k-Mean	Video Game Sales	3

A further application of the test used in table 2 is the method of recording sales [7-9]. In the following sections we can see the result of these recording sales. The sales recording algorithm works as follows: For each cluster in F_i we check if there is a matching cluster in the previous frame F_{i-1}. If there is, we use the sales from the previous frame, otherwise we pick a new sales record. This means that every time we find k_i $f=$ k_{i-1} or $|Cf(P_i)||+Cf(P_{i-1})|$ we change sales. There is however a limited amount of sales, which means that they may reoccur. The images are supposed to work as a complement to table 1 and 2. The sequence is also long enough to produce sales complex to make sense of, so we have mainly used the first frames needed to show the problem areas.

4. Discussion

One of the main issues with the GMM plots is that it could be challenging to find the maximum, given that the function is not monotonously decreasing before we get close to our local maximum. When looking at the GMM for the Video Games Sales data, it's an even more subtle change, as can be seen in figure. To help this search along, making it possible to look for a global maximum of the data set, we have used linear de-trending. This can be seen in figure. This example comes from data set, but a similar improvement can be seen in all data set. Both plots are created by taking the average of all plots produced when running the algorithm. The idea behind adding dimensions is that it could help when there are ambiguities. Say that spatial coordinates would return one cluster, but they have sales with different sign, or different rating. Adding these features could potentially help to separate the clusters. In reality, a reasonable use of these values was: $k = \{1, 2, 3\}$ this is of course a rough estimation, but serves as a benchmark. This was the values used to produce the table 1 and 2. As one can see, we have not used k_3, rating, since it gave a worse score regardless of the values we tried. Running the algorithm using only this parameter gave no indication of the true clusters. All cases benefited from having k_2, sales, but in reality, the score could be improved by adjusting the value. A further exploration could add additional parameters, making it even more unambiguous. One thing to address about the sales in "Data set" is that there seems to be a problem when changing countries. Our initial theory was that it had to do with the sales changing sign, but there was no noticeable effect when excluding sales as a parameter. Further analysis could focus on this since it seems to create a big decrease in performance.

5. Conclusion

The K-means output presents fixed clusters depicting hard membership of a data point to a cluster. Gaussian mixture model (GMM) determines these clusters without associating each sample with a cluster. It brings out a probabilistic approach of soft membership of each data-points to the dataset. The first thing we can conclude is that none of the algorithms produced a result that was sufficient for the task. That is, there would be little use of a clustering with as low as 50% success rate in any real-life scenario. What we can see is however that these rather primitive algorithms do produce a non-trivial result. Even this low score does indicate that there is potential, and maybe given a non-ambiguous initiation something like conditioning on previous frames, adding peripheral clusters, or extending the dimensionality might improve the score as to make if relevant. The real time or online use of the algorithm is also something that needs to be explored more. K-means is faster than Gaussian Mixture Model clustering. But there is a chance that since the maximum number of clusters are so low, both might be relevant. Since tracking for individual digit is already in place, the clustering might not need to be done at every frame, but at a given interval or when changes happen. To comment on the scores in table 1 and 2, we were expecting an even better performance from the Gaussian Mixture Model clustering. As to my knowledge Gaussian Mixture Model clustering should perform better on these types of data sets. We might however have underestimated the spherical nature of data. One explanation for the lack of performance is we did not find a way of determining k that could rival approximation. That is, the clustering might be better if k was given. If one would to venture outside of algorithms who are easy to implement, more modern algorithm is available. One example would be Gaussian mixture model, which is a density-based algorithm who can process clusters of arbitrary shaped, does not need a supplied k and is deterministic [6,16]. Another is the GMM method which is based in randomized search [17-22]. Both are focused on large data set but are also more sophisticated. Another noticeable feature is that both are also faster than k-means. Another algorithm which we have considered is the k-means algorithm [23-25] which works by starting with k_{min} clusters and alternating between converging using k-means and then splitting clusters into two or more. Each iteration they use approximation to evaluate k. This way there is no need to redo the whole algorithm when trying out different k, but, they have sales with different sign, or different rating. Adding these features could potentially help to separate the clusters.

References

[1] Marcel R Ackermann, Johannes Blömer, Daniel Kuntze, and Christian Sohler. Analysis of agglomerative clustering. *Algorithmica*, 69(1):184–215, 2014.

[2] Michael R Anderberg. Cluster analysis for applications. monographs and textbooks on probability and mathematical statistics, 1973.

[3] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[4] Ian Davidson and SS Ravi. Agglomerative Gaussian Mixture Model clustering with constraints: Theoretical and empirical results. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 59–70. Springer, 2005.

[5] William HE Day and Herbert Edelsbrunner. Efficient algorithms for agglomerative Gaussian Mixture Model cluster- ing methods. *Journal of classification*, 1(1):7–24, 1984.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[7] Abdulshaheed, H.R., Binti, S.A., and Sadiq, I.I., 2018. A Review on Smart Solutions Based-On Cloud Computing and Wireless Sensing. *International Journal of Pure and Applied Mathematics*, 119 (18), pp.461–486.

[8] Limin Fu and Enzo Medico. Flame, a novel fuzzy clustering method for the analysis of dna microarray data. *BMC bioinformatics*, 8(1):1, 2007.

[9] Anil K Jain and HC Martin. Law, data clustering: a user's dilemma. In *Proceedings of the First international conference on Pattern Recognition and Machine Intelligence*, 2005.

[10] Ismo Kärkkäinen and Pasi Fränti. *Dynamic local search algorithm for the clustering problem.*

University of Joensuu, 2002.

[11] Robert E Kass and Larry Wasserman. A reference bayesian test for nested hypotheses and its relationship to the schwarz criterion. *Journal of the american statistical association*, 90(431):928–934, 1995.

[12] Jon Kleinberg and Eva Tardos. *Algorithm design*. Pearson Education India, 2006.

[13] Godfrey N Lance and William T Williams. Computer programs for Gaussian Mixture Model polythetic classification (“similarity analyses”). *The Computer Journal*, 9(1):60–64, 1966.

[14] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[15] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. In *International Workshop on Algorithms and Computation*, pages 274–285. Springer, 2009.

[16] Raymond T Ng and Jiawei Han. E cient and e ective clustering methods for spatial data mining. In *Proc. of*, pages 144–155, 1994.

[17] Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering*, 14(5):1003–1016, 2002.

[18] Dan Pelleg, Andrew W Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML*, volume 1, 2000.

[19] Abdulshaheed, H.R., Binti, S.A., and Sadiq, I.I., 2018. Proposed a Smart Solutions Based-on Cloud Computing and Wireless Sensing. International Journal of Pure and Applied Mathematics, 119 (18), pp.427–449.

[20] Cor J. Veenman, Marcel J. T. Reinders, and Eric Backer. A maximum variance cluster algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1273–1280, 2002.

[21] Cai W., Chen S., and Zhang D., “Fast and Robust Fuzzy Cmeans Clustering Algorithms Incorporating Local Information for Segmentation,” Pattern Recognition, vol. 40, no. 3, pp. 825-838, 2007

[22] Tolias A. and Panas M., “On Applying Spatial Constraints in Gaussian Clustering using a expectation maximization Based System,” IEEE Signal Processing Letters, vol. 5, no. 10, pp. 245-247, 1998.

[23] D. Petrovic, Basic, H., Durakovic, B., and Prodanovic, S., “Science-Technology Park Ilidža as a Generator of Innovation Potential and SME’s Development in Bosnia and Herzegovina”, Periodicals of Engineering and Natural Sciences, vol. 1, 1 vol., no. 2, pp. 51-55, 2013.

[24] B. Durakovic, “Emerging Issues, Trends and Challenges for Sustainable Engineering”, The Sixth Regional Conference on Soft Computing 2017. 2017.

[25] M. Inalpolat and Durakovic, B., “Implementation of Advanced Automated Material Handling Systems in Manufacturing Environment”, European Conference of Technology and Society - EuroTecS. 2013.