Evaluation of Partitioning Clustering Algorithms for Processing Social Media Data in Tourism Domain

Shini Renjith
Department of Computer Applications
CUSAT
Kochi, India
shinirenjith@gmail.com

A. Sreekumar
Department of Computer Applications
CUSAT
Kochi, India
sreekumar@cusat.ac.in

M. Jathavedan
Department of Computer Applications
CUSAT
Kochi, India
mjvedan@cusat.ac.in

***Abstract* - Recommender systems are evolving as an essential part of every industry with no exception to travel and tourism segment. Considering the exponential increase in social media usage and huge volume of data being generated through this channel, it can be considered as a vital source of input data for modern recommender systems. This in turn resulted in the need of efficient and effective mechanisms for contextualized information retrieval. Traditional recommender systems adopt collaborative filtering techniques to deal with social context. However they turn out to be computational intensive and thereby less scalable with internet and social media as input channel. A possible solution is to adopt clustering techniques to limit the data to be considered for recommendation process. In tourism context, based on social media interactions like reviews, forums, blogs, feedbacks, etc. travelers can be clustered to form different interest groups. This experimental analysis aims at comparing key clustering algorithms with the aim of finding an optimal option that can be adopted in tourism domain by applying social media datasets from travel and tourism context.**

Keywords* - *Recommender Systems, Clustering Algorithms, Travel and Tourism, Cluster Evaluation

I. INTRODUCTION

In end user perspective, travel and tourism is mostly explorative in nature and repetitive travels to same locations are minimal. So, travelers have to take decisions regarding their destinations and associated facilities to be consumed without adequate prior or personal knowledge. The best option available is to leverage social media and internet, but the amount of time required to extract relevant information is too high. Tourism recommenders are the best solutions in this scenario. Recommender systems helps in terms of automated filtering, processing, personalization and contextualization of the huge volume of data that is available and growing on a daily basis on the internet and the social media.

Traditional recommender systems use collaborative filtering algorithms [1][2] in social contexts to predict probable options for a user based on past traits of similar users. However, with ever-increasing volume of data, collaborative filtering recommender systems are highly computational intensive and thereby not scalable in big data context [3]. Recommender systems based on clustering techniques is a solution to the scalability concern. Restricting the processes of recommendation within comparably small clusters can reduce the need of high computation power. However this demands for the identification and adoption of better clustering mechanisms. In this work, we compared multiple clustering algorithms using three different datasets of varying volumes and evaluated their performance.

Section 2 of this paper explains the concept of clustering and clustering algorithms that are used in this experimental analysis. Sections 3 and 4 detail on the tools and methodology adopted and details of the experiments. Section 5 concludes the paper by sharing our inferences and future plans.

II. ANTECEDENTS

A. Clustering

Clustering in machine learning world is an unsupervised approach of grouping a set of entities together so that the entities in one group are more similar to each other than to the entities in another group. Unsupervised learning is applied while there is input data, but there is no corresponding output variables associated with it. Its goal is to understand and model the underlying distribution of data so as to learn more about it. Clustering has various applications like market segmentation for targeted advertisements and promotional offers, grouping of web contents in a search engines, text summarization, biological applications, astronomy, etc.

Clustering reveals natural and meaningful groups among available data. Clustering algorithms aims to achieve highest intra-cluster similarity and least inter-cluster similarity. The concept of distance measure is used to calculate the similarity between objects. When the distance measure between two entities is very less, they are considered as similar. Based on the data under consideration appropriate distance measure can be chosen for clustering. A few of the most common distance measures include Euclidean, Manhattan, Cosine, Jaccard and Minkowski distances.

Clustering Algorithms can be generally categorized into three groups – partitioning [4], hierarchical and density based clustering. Partitioning clustering is used to categorize observations within a dataset based on their similarity. In this approach, the user has to identify the optimal count of clusters for the dataset in consideration and it need to be mentioned to the algorithm. The common partitioning clustering algorithms are k-means clustering [5][6], k-medoids clustering which is also known as Partitioning Around Medoids (PAM) [7][8], Clustering for Large Applications (CLARA) [8][9][10] and Fuzzy C-means clustering [11].

B. k-means Algorithm

k-means algorithm [5][6] is the most popular unsupervised machine learning algorithm where a given training set is partitioned into k number of clusters through an iterative process. The algorithm identifies k different clusters from a given dataset with maximum intra cluster similarity among entities and maximum inter cluster dissimilarity. Each cluster is characterized by its centroid which is the mean value of the entities present in the cluster. k-means clustering algorithm focuses on defining clusters by ensuring that the total within cluster variation is kept at the minimum. As per Hartigan-Wong algorithm[6], the standard implementation of k-means clustering, the total within cluster variation is defined as the sum of squared distances between each of the cluster members with the corresponding centroid (1).

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

where x_i is an entity belonging to cluster C_k and μ_k is the mean of all entities forming the cluster C_k. The assignment of

each entity to a given cluster is performed in such a way that the sum of squared distances between the entity and to the centroid of the cluster is minimum. The goodness of the clustering is measured as the total within cluster variation (2) which requires to be kept as lower as possible.

$$Total\ WCV = \sum_{i=1}^{k} WCV(C_k) = \sum_{i=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2 \quad (2)$$

The k-means algorithm is explained in Fig. 1.

1. Determine k, the total number of clusters to be formed
2. Identify k initial centroids to start with. Normally a random sampling is performed within the dataset to identify the initial points
3. For each entity in the dataset
   a. Calculate the distance between the entity and each of the k centroids
   b. Assign the entity to the cluster whose centroid is the closest
4. For each of the k clusters
   a. Update the cluster centroid by re-calculating the mean values of all the entities in the cluster.
5. Iteratively minimize the total within cluster variation. That is, iterate steps 3 and 4 until there is no changes to cluster assignments or the maximum number of iterations are reached

Fig. 1. k-means Algorithm

A simple and easiest measure for closeness of an item with its centroid is Euclidean distance. The advantages of k-means algorithm include ease to understand, flexibility, performance with large number of input attributes. The key challenges include initialization of centroids, selection of k, low computational efficiency and high need of memory to process due to more number of iterations required and high sensitivity to the presence of outliers in test dataset.

C. k-medoids Algorithm

The k-medoids algorithm [7][8] works similar to k-means algorithm, but less sensitive to the presence of noises and outliers in the data set. The key difference is in fitting the centroids. While in k-means algorithm the centroids need not be the actual entities from the dataset, k-medoids algorithm chose an entity from the dataset whose average dissimilarity to all the other entities is minimum as the medoid. The most common k-medoids algorithm is Partitioning around Medoids (PAM). The key activity performed in the PAM algorithm is the search for k number of medoids among the entities in the dataset. Clusters are formed by assigning each entity from the dataset to the nearest medoid.

As the next phase in the algorithm, the objective function is computed by swapping medoid and non-medoid entities. The objective function represents the cumulative dissimilarities of all entities in the dataset to their nearest selected medoids. This step enables the algorithm to improve the quality of clustering. If there is an opportunity exists to reduce the value of objective function by interchanging a selected medoid with another entity, the swap is performed. This process is repeated until there is no opportunity for reducing the objective function so that k number of medoids are identified which has the least cumulative dissimilarities among the entities of the dataset.

The k-medoids algorithm is represented in Fig. 2.

1. Determine k, the total number of clusters to be populated
2. Identify k initial medoids to start with. Medoids are always selected from the members of the dataset.
3. For each entity in the dataset
   a. Calculate the distance between the entity and each of the k medoids
   b. Allocate the entity to a cluster whose medoid is the nearest
4. For each of the k clusters
   a. Scan for an entity that can reduce the average dissimilarity within the cluster.
   b. If an entity is found that can reduce the objective function, swap the medoid and the entity in consideration.
5. If there is any change to at least one medoid in step 4, go to step 3 else end

Fig. 2. k- medoids Algorithm

The common distance measure being used with PAM algorithm are Euclidean distance and Manhattan distance. Thus the underlying calculations for total dissimilarities are similar to that of k-means algorithm. While Euclidean distance refers to the square root of sum of squares of the distances, Manhattan distance consider the sum of absolute distances.

D. Clustering for Large Applications (CLARA) Algorithm

CLARA (Clustering for Large Applications) algorithm [8][9][10] is formulated as an extension to k-medoids algorithm to deal with datasets containing very large number of entities (when it ranges in several thousands). This is to overcome the scalability challenges of k-medoids algorithm like the needs for high computing time and large memory requirements. This is achieved by means of sampling approach. The algorithm takes into account of a small sample from the dataset and applies PAM algorithm to identify an optimal set of medoids for the sample. The quality of the chosen medoids is measured against the complete dataset by checking the average dissimilarity between all entities in the dataset and the medoids. This sampling and clustering process is repeated in order to reduce the sampling bias.

The CLARA algorithm is explained in Fig. 3.

1. Split the dataset randomly into multiple subsets having a fixed size
2. Apply PAM algorithm on each subset and choose the corresponding k medoids
3. For each entity in the complete dataset
   a. Calculate the distance between the entity and each of the k medoids
   b. Allocate the entity to a cluster whose medoid is the nearest
4. For each subset
   a. Calculate the average dissimilarities of the entities to their closest medoid (quality or goodness of the clustering)
5. Select sub-dataset with the minimum average dissimilarity

Fig. 3. CLARA Algorithm

E. Fuzzy C-Means Clustering

Fuzzy C-means algorithm [11] is considered as a soft clustering algorithm where each entity in a dataset can belong to all clusters to a certain extent. The belonging of an entity to a cluster is calculated based on the distance of the entity from the center of the cluster – an entity which is located near to the centroid of the cluster will have a higher degree of membership to the particular cluster whereas an entity which is far from the centroid of a cluster will have a lower degree of membership in the particular cluster. The cumulative membership across all clusters for any given data point will be 1 or 100% (3). Mathematically the membership of an element can be represented as

$$\sum_{k=1}^{c} \mu_k(x_i) = 1 \quad (3)$$

where c is total number of clusters and μ_k is the degree of membership of a data point $x_i$in a cluster k. The centroid of cluster can be calculated as (4).

$$C_k = \frac{\sum_{i=1}^{n}[\mu_k(x_i)]^m x_i}{\sum_{i=1}^{n}[\mu_k(x_i)]^m} = \frac{\sum_{i=1}^{n}[u_{ik}]^m x_i}{\sum_{i=1}^{n}[u_{ik}]^m} \quad (4)$$

where $x_i$is a data point among the n number of items belonging to the dataset, m is the fuzziness coefficient with $m \in [1, \infty]$ and u_{ik} is the degree of membership for x_i in the cluster k having centroid C_k. On each iteration, membership matrix is updated using the equation (5).

$$u_{ik} = \mu_k(x_i) = \frac{\left[\frac{1}{d_{ik}}\right]^{1/(m-1)}}{\sum_{k=1}^{c}\left[\frac{1}{d_{ik}}\right]^{1/(m-1)}} \quad (5)$$

where d_{ik} represents the Euclidean distance between x_i and C_k. The goal of fuzzy c means algorithm is to iteratively minimize the objective function (6).

$$\mathrm{J}(U, c_1, c_2, .., c_k) = \sum_{k=1}^{c}\sum_{j=1}^{n}(u_{ik})^m (d_{ik})^2 \quad (6)$$

The fuzzy c means algorithm can be represented as in Fig. 4.

1. Determine k, the total number of clusters to be populated
2. Identify k initial centroids to start with
3. Initialize the membership matrix for all the elements in dataset against each cluster so that the cumulative value of membership of an element is 100%
4. For each of the k clusters
   a. Calculate the centroid considering all memberships of the cluster
5. For each entity in the dataset
   a. Calculate the dissimilarity between each entity and the centroids
   b. Update the membership matrix
6. Iteratively minimize the objective function. That is, iterate steps 4 and 5 until there is no changes to cluster centroids or the maximum number of iterations are reached

Fig. 4. Fuzzy C means Algorithm

III. METHODOLOGY

A. Tools Used

The main tools used for this analysis and study is R programming language [12][13], which is a free open source platform for statistical computing and data representation and RStudio [14], an integrated development environment for R. There are multiple R packages available with standard implementations for various machine learning algorithms. The key R packages used in this study are listed in Table 1.

TABLE I. R PACKAGES USED FOR ANALYSIS

R Package	Purpose
stats	Implementation of k-means algorithm
Cluster [15]	Implementation of PAM and CLARA algorithms
NbClust [16]	To decide the optimal count of clusters
Factoextra [17]	For data visualization

B. Data Cleansing and Normalizing

Most of the times, the input data for clustering process is impacted with presence of error values, missing values and noises. Due to this, data cleaning is considered as an essential activity that need to be performed prior to cluster analysis. There may be multiple attributes associated with entities in every datasets and each of these attributes may be measured using different scales. Assume there are two attributes, one where the differences between candidate cases is large and the other small, then the former attribute may act as the only driver while measuring the distance between entities. This leads to the need of normalizing the attribute values prior to clustering so as to ensure none of the attribute is getting an overweight.

There are multiple techniques being adopted for data cleansing. In R, the function str() is used to analyze all attributes and summary() to analyze individual ones. In some scenarios error data can be considered as missing data and one can apply similar processing logics. One of the approaches in handling missing values is to ignore them. But if the volume of such cases is high, it can be replaced with NA values. Typical method to handle missing values is to replace those with the mean value. But in the case of categorical values like gender attribute of a user, a new attribute can be added to identify records containing missing values.

Scaling or normalizing of the attributes is a common practice that need to be performed in clustering scenarios. This step ensures that all attributes are in the same range or scale sp that the attributes having values in large range does not dominate other attributes. We used R functions scale() along with lapply() to standardize all observations based on Z score.

C. Estimating the Optimal Number of Clusters

Partitioning clustering requires to specify the value of k, the number of clusters in the dataset. There are multiple methods to predict the near optimal cluster count in the dataset like Elbow method, GAP statistic method, Average Silhouette method, etc. The basic requirement of partitioning algorithms are to identify clusters in such a way that the total within cluster sum of squares is minimal. We leveraged the R package NbClust to decide the optimal count of clusters in the data set. This package apply 30 different indices and proposes the optimal cluster count that can be adopted.

D. Clustering

The cleansed data is used as input to clustering operation. Among multiple clustering available, we have evaluated the clustering techniques described in Section 2. Implementation and verification of clustering algorithms are performed using the R functions mentioned in Table 1.

E. Cluster Evaluation

The objective of a clustering algorithm is to accomplish minimum inter-cluster similarity among clusters and maximum intra-cluster similarity within each cluster [18][19]. The extent to which this objective is met can be considered as a criterion for measuring the quality of clustering. This is achieved by analyzing intrinsic characteristics of the clusters

formed. The common measures for internal evaluation of clusters include Silhouette analysis [20] and Dunn Index [21]. Silhouette analysis is a measure which indicates how well the dataset is clustered by calculating the average distance between the clusters formed [Fig. 5]. The function silhouette() available in the package cluster can be leveraged for the same.

1. For each item in dataset,
   a. Calculate the average dissimilarity Avg.Dissn between item n and all other items of the cluster to which item n belongs to.
   b. For all other clusters C, to which item n does not belong to
      i. Calculate the average dissimilarity Avg.Diss(n,C) of item n to all observations of C.
      ii. The smallest of these Avg.Diss(n,C) is defined as Avg.Neigh.Dissn = minC Avg.Diss(n,C). i.e., Avg.Neigh.Dissn is the dissimilarity between item n and the nearest cluster to which it does not belong to.
   c. Silhouette Width of the item n is defined as
2. Sil.Widthn = (Avg.Neigh.Dissn – Avg.Dissn) / max(Avg.Neigh.Dissn , Avg.Dissn).

Fig. 5. Silhouette Analysis

Dunn Index is defined as the ratio of minimum separation to maximum intra cluster distance [Fig. 6]. It is measured in R using the functions cluster.stats() from package fpc.

1. For each cluster,
   a. Calculate the distance between each item within the cluster and the items in other clusters
   b. Determine inter-cluster separation (min.separation) which is the minimum of the pairwise distance
2. For each cluster,
   a. Calculate the distance among each member items.
   b. Determine the intra-cluster compactness (max.diameter) which is the maximum value of intra-cluster distances
3. Dunn Index = (min.separation) / (max.diameter).

Fig. 6. Calculation of Dunn Index

IV. Experimental Analysis

We considered three real-time social media datasets from tourism domain for the analysis and captured the results. The datasets corresponds to user interest information accrued from reviews, feedbacks on different types of point of interests and ratings on attractions visited from three different geographies. Table 2 depicts more details of the datasets used.

TABLE II. Details of Datasets

Dataset 1	User interest information populated from destination reviews (Source [1]) across South India published on holidayiq.com till October 2014.	Contains 249 user records with 6 interest attributes derived from more than 1500 reviews
Dataset 2	User's average feedback/rating information on 10 categories of attractions in East Asia captured from tripadvisor.com.	Contains 980 user records with 10 feedback attributes inferred from numerous destination reviews
Dataset 3	User's average rating information on 24 types of attractions across Europe captured from Google reviews.	Contains 5456 user records with 24 rating attributes

Fig. 7 depicts the result of the estimation of optimal number of clusters using NbClust() function and results plotted using fviz_nbclust() function.

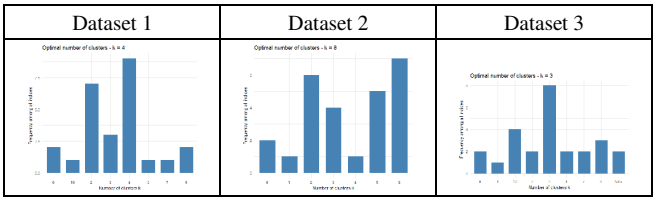

Fig. 7. Estimation of optimal number of clusters

Clustering is performed using k-means, k-medoids (portioning around medoids), clustering for large applications (CLARA) and fuzzy c-means approaches and the resultant clusters are plotted using fviz_cluster() function in Fig. 8.

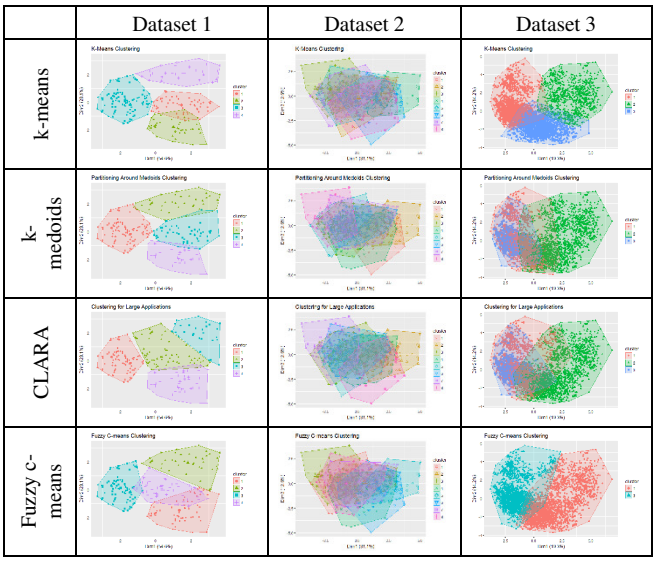

Fig. 8. Clusters generated via different clustering techniques

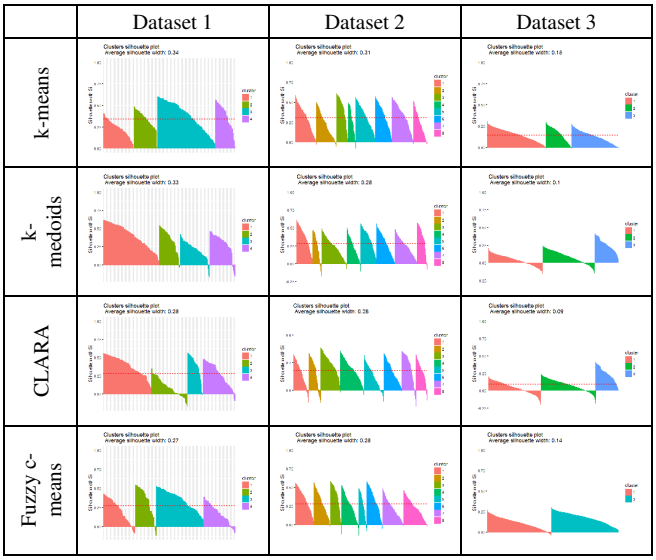

Fig. 9. Results from Silhouette Analysis

For internal evaluation of clusters formed, Silhouette analysis is performed and results are plotted using fviz_silhouette() function [Fig. 9]. We have used cluster.stats() function to capture average silhouette width (avg.silwidth), Dunn index (dunn), Dunn 2 index (dunn2 = min. avg. dissimilarity among two clusters / max. avg. within cluster dissimilarity), entropy of the distribution of items

within a cluster (entropy) and wb.ratio (average.within / average.between) [Table 3].

TABLE III. RESULTS FROM ANALYSIS USING CLUSTER.STATS()

Dataset	Measure	k-means	k-medoids (pam)	CLARA	Fuzzy c-means
Dataset 1	avg.silwidth	0.342257	0.330981	0.281170	0.274475
	dunn	0.068746	0.020773	0.047632	0.056989
	dunn2	1.430012	1.384847	1.316865	1.248009
	entropy	1.294179	1.306847	1.318452	1.345672
	wb.ratio	0.525884	0.531002	0.557269	0.552480
Dataset 2	avg.silwidth	0.305087	0.282551	0.284243	0.280955
	dunn	0.009681	0.010317	0.017442	0.012449
	dunn2	0.902415	0.814189	0.857498	0.808759
	entropy	2.033419	2.026279	2.050770	2.037319
	wb.ratio	0.307399	0.307633	0.316814	0.308307
Dataset 3	avg.silwidth	0.145402	0.095386	0.092687	0.143102
	dunn	0.021640	0.012780	0.005191	0.001034
	dunn2	1.170973	0.974290	0.970604	1.131867
	entropy	1.046666	1.040005	1.035243	0.692863
	wb.ratio	0.811936	0.870084	0.871499	0.852627

V. CONCLUSION

Clustering is the process of populating similar items into clusters with high intra-cluster and low inter cluster similarity. In this study we evaluated the most common partitioning clustering algorithms using three different dataset gathered from travel and tourism domain. As per literature, no single partitioning clustering algorithm is considered as superior for all clustering requirements. From the results gathered during this analysis, k-means algorithm outperformed other partitioning clustering algorithms for the datasets used. So, it is required to choose among algorithms depending on the dataset being processed. As the next step of this study, we are planning to compare performance of hierarchical and density-based clustering against partitioning clustering. We also target to compare time complexity for each of these algorithms for varying sizes of datasets. Further we aim to apply dimensionality reduction techniques to achieve better performance while clustering. For this, we plan to leverage deep learning and auto encoders instead of the standard approaches like principal component analysis.

Clustering can help to propose most relevant solutions to customers based on their profiles. Any information that reflect customer traits can become an input to clustering process. In this work, we considered user reviews, feedbacks and rating information captured from forums and social media. However travel managers have diverse opportunities to capture user traits and interests by tracking the types of queries coming to them, taking direct feedback via questionnaires or surveys, keeping track of the user transactions and monitoring the reviews on travel forums and portals. Depending on the data volume and data distribution pattern in consideration, they can adopt appropriate clustering algorithms to segment their customer base so that targeted marketing strategy and/or travel solutions can be offered.

ACKNOWLEDGMENT

We acknowledge support from the Department of Computer Applications, Cochin University of Science and Technology for all guidance, reviews, valuable suggestions and very useful discussions..

REFERENCES

[1] Renjith, Shini, and C. Anjali. "A personalized mobile travel recommender system using hybrid algorithm." In Computational Systems and Communications (ICCSC), 2014 First International Conference on, pp. 12-17. IEEE, 2014.

[2] Renjith, Shini, and C. Anjali. "A personalized travel recommender model based on content-based prediction and collaborative recommendation." International Journal of Computer Science and Mobile Computing, ICMIC13 (2013): 66-73.

[3] Jiang, Shuhui, Xueming Qian, Tao Mei, and Yun Fu. "Personalized travel sequence recommendation on multi-source big social media." IEEE Transactions on Big Data 2, no. 1 (2016): 43-56.

[4] Estivill-Castro, Vladimir. "Why so many clustering algorithms: a position paper." ACM SIGKDD explorations newsletter 4, no. 1 (2002): 65-75.

[5] MacQueen, James. "Some methods for classification and analysis of multivariate observations." In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, no. 14, pp. 281-297. 1967.

[6] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means clustering algorithm." Journal of the Royal Statistical Society. Series C (Applied Statistics) 28, no. 1 (1979): 100-108.

[7] Kaufman, Leonard, and Peter Rousseeuw. Clustering by means of medoids. North-Holland, 1987.

[8] Kaufman, Leonard, and Peter J. Rousseeuw. Finding groups in data: an introduction to cluster analysis. Vol. 344. John Wiley & Sons, 2009.

[9] Park, Hae-Sang, and Chi-Hyuck Jun. "A simple and fast algorithm for K-medoids clustering." Expert systems with applications 36, no. 2 (2009): 3336-3341.

[10] Wei, Chih-Ping, Yen-Hsien Lee, and Che-Ming Hsu. "Empirical comparison of fast clustering algorithms for large data sets." In System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on, pp. 10-pp. IEEE, 2000.

[11] Bezdek, James C., Robert Ehrlich, and William Full. "FCM: The fuzzy c-means clustering algorithm." Computers & Geosciences 10, no. 2-3 (1984): 191-203.

[12] Team, R. Core. "R: A language and environment for statistical computing." (2013).

[13] Ihaka, Ross, and Robert Gentleman. "R: a language for data analysis and graphics." Journal of computational and graphical statistics 5, no. 3 (1996): 299-314.

[14] Studio, R. "RStudio: integrated development environment for R." RStudio Inc, Boston, Massachusetts (2012).

[15] Maechler, Martin, Peter Rousseeuw, Anja Struyf, Mia Hubert, and Kurt Hornik. "Cluster: cluster analysis basics and extensions." R package version 1, no. 2 (2012): 56.

[16] Charrad, M., N. Ghazzali, V. Boiteau, and A. Niknafs. "NbClust: An examination of indices for determining the number of clusters. R package version 1.4 2012."

[17] Kassambara, Alboukadel, and Fabian Mundt. "Factoextra: extract and visualize the results of multivariate data analyses." R package version 1, no. 3 (2016).

[18] Maulik, Ujjwal, and Sanghamitra Bandyopadhyay. "Performance evaluation of some clustering algorithms and validity indices." IEEE Transactions on Pattern Analysis and Machine Intelligence 24, no. 12 (2002): 1650-1654.

[19] Kovács, Ferenc, Csaba Legány, and Attila Babos. "Cluster validity measurement techniques." In 6th International symposium of hungarian researchers on computational intelligence. 2005.

[20] Rousseeuw, Peter J. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." Journal of computational and applied mathematics 20 (1987): 53-65.

[21] Dunn, Joseph C. "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters." (1973): 32-57.