The Comparative Study on Clustering Method Using Hospital Facility Data in Jakarta District and Surrounding Areas

Yogi Wahyu Romadon and Devi Fitrianah

***Abstract*—The present study reports the findings of a comparative analysis conducted on health facility research (HFR) data, based on a clustering method with K-Means and K-Medoids algorithm. The K-Means algorithm consist of four steps: specifyng the centroid values, grouping data on the centroid, calculating centroid values, repeating grouping data on the centroid, and calculating the values until the cluster is stable (convergence), The K-Medoids algorithm consists of six steps: medoids initialization, data allocation to the nearest medoid, determininationof new medoids, calculation of data distance with medoid, calculation of deviation, and repeating the determination of new medoids, until the deviation to the convergence cluster is counted. The data utilized in this study are HFR data located in the Jakarta district and surrounding areas, from 2013 to 2018. Our results showed that the execution time from K-Medoids algorithm outperformed the K-Means algorithm. The K-Medoids algorithm speeded up the execution time, and at the same time improved the density value between clusters (silhouette). By utilizing the clustering method, health facilities in hospitals in Jabodetabek can be categorized based on their resources.**

***Index Terms*—Clustering, K-Means, K-Medoids, health facility research.**

I. Introduction

Hospitals provide vitally important inpatient, and outpatient, as well as emergency services [1]. Hospitals are one of the main facilities for all residents, of the Jakarta distric, allowing constant access to health services. There is a need to assure that high quality of health services and appropriate infrastructure, exsist to promote preventive, curative, and rehabilitative activities. The central government, local government and the community assure this quality [2].

Hospital facilities and services grouped into class A consisting of 2.42%, class B consisting of 14.11%, class C consisting of 41.25%, and class D consisting of 21.07%, in 2016; 21.15% have not been grouped [1].

In accordance with the conditions described above, a system is needed to group hospital into a particular data group (cluster), based on certain characteristics, so that the user can find information regarding which hospital offer the set of doctors, facilities, and set staff pertaining to the users' requirement.

Data was collected by the Ministry of Health of the Republic of Indonesia in the form of data on hospital resources obtained through health facility research (HFR), which is available to the public on the website (*sirs.yankes.kemkes.go.id/rsonline)*. However there is still a lack of effort to utilize the data for further analysis. Therefore, the author will perform a comparative study by conducting an analysis, using the K-Means and K-Medoids algorithms and classifying hospital facilities based on the resources owned by a hospital. The data will be grouped into a cluster based on certain similarities with the predeterimined set of clusters. In accordance with these conditions, an algorithm that is suitable for use is employed using K-Means and K-Medoids. These algorithms partition or group data into one or more clusters that have certain similarities based on a predetermined cluster number. The clustering method, using the K-Means and K-Medoids algorithms, will be implemented with TensorFlow, which is an open-source library software for numerical calculations for both machine learning and deep learning [3].

Manuscript received July 19, 2019; revised October 17, 2019.

The authors are with the Department of Informatics, Faculty of Computer Science, Mercu Buana Univerisity, Jakarta 11650, Indonesia (e-mail: 41515010069@student.mercubuana.ac.id, devi.fitrianah@mercubuana.ac.id).

II. Literature Review

A. Clustering

One method for grouping data is clustering. In clustering, existing data are grouped based on the level of similarity data sharing common characteristics with other data are grouped into one cluster, while data that do not have similarities be grouped with other clusters [4]. Clustering methods have been widely used in various fields, including pattern recognition, data analysis, and image processing [5].

Basically, this method has a simpleset of steps that is defining the distance between data elements, so that each data will be centered to a nearest point (called centroid); if the data is not close together it will center to another point [6].

B. K-Means

The K-Means algorithm is one method, used to classify semi-constructed or unstructured datasets. This method is one of the most common and effective methods for grouping data, due to its simplicity and ability to handle productive datasets [7]. The K-Means algorithm is generally implemented to relocate datasets into *k* cluster [8].

The K-Means algorithm has the following main steps: determining the value of *k* randomly as a cluster center; locating existing data so that they forn or produce clusters with *k* as their center (centroid); calculating a new cluster center or calculating centroid; repeatedly generating new

doi: 10.18178/ijmlc.2019.9.6.868

partitions and calculating new clusters until the member is stable (convergence) [9]. These stages can be explained by the following equations.

1) Determine the value *k* randomly as the center of the cluster; determination of *k* can be calculated by following equation (1):

$$Z_1(1),\ Z_2(1),\ \ldots Z_k(k) \tag{1}$$

where $Z_1(1),\ Z_2(1),\ \ldots Z_k(k)$ is the value of each cluster center (centroid).

2) After the clusters are determined, the next step is to group and distribute the data into each cluster, so that the cluster pieces are performed with *k* as the center of each cluster (centroid). The grouping and distribution can be formulated with the equation (2),

$$x \in S_i(k) = if\ \|x - z_i(k)\| < \|x - z_i(k)\| \tag{2}$$

where *i* is a value for every *k* cluster that exist, *i* is not equal to *j*, and $S_j(k)$ shown the cluster sample from $Z_j(k)$.

3) The next step after the data has been grouped and distributed into existing cluster, is to calculate the new cluster center, for example $Z_j(k+1)$, $j = 1, 2, \ldots, k$ such that the sum of squared distances from all points $Sj(k)$ to the center of new cluster. The new cluster center $Zj(k+1)$ is calculated to minimize the next index performance given according to the equation (3).

$$J_j = \sum_{x \in S_j(k)} \left\| x - z_j(k+1) \right\|^2,\ j1, 2, \ldots, K \tag{3}$$

4) The last step in this algorithm is if data $Z_j(k+1) = Z_j(k)$ for $j = 1,2,\ldots,k$, the procedure of K-Means algorithm will occur when this process stops and does not repeat to equation (2) and (3), which is called convergence.

C. K-Medoids

The K-Medoids algorithm is a clustering method that functions to break up the dataset into groups. The advantage of this method is, that it is able to overcome the weakness of the K-Means algorithm in regards to the sensitivity to outlier [10].

The difference between the K-Means and the K-Medoids algorithm is that the K-Medoids algorithm uses an object as medoid (cluster center) for each cluster, whereas the K-Means algorithm uses the mean as the center of cluster. The steps of the K-Medoids algorithm are as follows; initialize *k* as much as the number of clusters and cluster centers, allocate each data to the nearest cluster, determine a new medoid, calculate distance between the object and the cluster center, do the determination of the new medoid, calculate the total deviation, and calculate the distance of the object with the cluster.

1) Initialize *k* as many of cluster and as cluster center (medoid), according to the equation (4) dan (5).

$$d(x, y) = \|x - y\| \tag{4}$$

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2};\ 1, 2, 3, \ldots n \tag{5}$$

where *d* is *Euclidian Distance*, x_i and y_i are the distance between points *x* and *y*.

2) Allocate each data to the nearest cluster of each data with the euclidean distance equation, which is shown in the equation (4) then (5).
3) Determine randomly the object on each cluster as new medoid.
4) Calculate each distance between objects in each cluster with a new medoid.
5) Calculate the total deviation (*S*) by means of total distance recently – total long distance. If you get $S < 0$ as a result, then exchange between objects with cluster data, to make the new *k* as new medoid, which shown in the equation (6),

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, o_i) \tag{6}$$

where *E* is the absolute error deviation for each object *p* in the dataset, *n* is the number of objects to be grouped into as many as *k* cluster and o_i is the cluster center (medoid) of the cluster C_i.

6) Repeat steps 3 trough 5, until the medoid does not change, and a cluster and each member are obtained (convergence).

D. Silhouette Coefficient

Silhouette coefficient is one way to perform a cluster evaluation internally. Silhouette coefficient is also used to assess the quality and strength of a cluster, by assessing how precisely an object is placed into a cluster based on synthetic and real-world data [11], [12]. Silhouette coefficient is also used to calculate similarity between one object in one cluster and another object in another cluster [13]. Silhouette coefficient calculation can be determined by the following equation (7).

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{7}$$

where $s(i)$ is silhouette of *i*, with value between -1 to 1, *i* is center cluster are determined, $a(i)$ is mean dissimilarity from points or point cluster one with the other, $b(i)$ is the mean value of smallest dissimilarity between *i* and element other than cluster *i* [14].

E. Rand Index

Rand index is the method for performing an external evaluation of a clustering [15]. Rand index is a way to measure the percentage of decisions that are appropriate to the cluster result obtained, where rand index can be obtained by equation (8) [16].

$$R_i = \frac{TP + TN}{TP + FP + FN + TN} \tag{8}$$

TP is *True Positive* is a result where the model places two similar points in the same cluster, TN is *True Negative* is a result where the model places two similar points into different clusters, FP is *False Positive* where the model places two points which do not have the same resemblance into the same cluster, and FN is *False Negative* state where the model places similar data into different clusters.

F. Related Works Regarding to Data Mining on Hospital Facilities

There are several related studies in grouping hospital facilities. In 2015, Bichen Zheng [17] implemented Neural Network, Random Forest, and Support Vector Machine to classify facility data in hospitals in the United States, and readmission data to classify existing patients. The results obtained showed an average accuracy value of 78.4%, after the completion of the tuning parameters process, increased to 97.3%.

Then, Abd Elrazek in 2017 [18] applied the Naïve Bayes algorithm with medical care system data including facilities, time, and cost in Egypt, which aimed to produce more efficient analysis compared to traditional statistical analysis in terms of controlling costs and improving patient care.

Next, in 2016 Magesh Vivek, Sundar Franco, and Thomas Greefin [19] applied the ANFIS method (neural-fuzzy is system which contains both Neural Network and fuzzy system), which aimed to improve the quality of health services for tuberculosis and cardiovascular patients.

III. Methodology

In this paper, five consecutive methodological steps were taken.: data collection, preprocessing, clustering with K-Means and K-Medoids, validation, and result analysis. An overview of the methodology is displayed on Fig. 1.

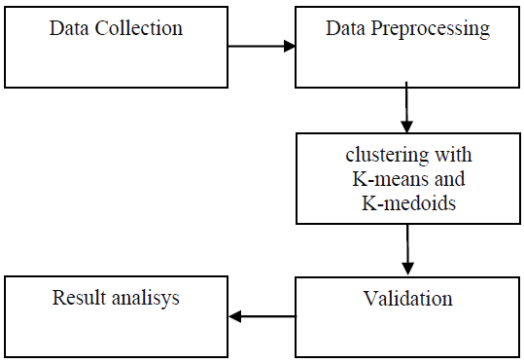

Fig. 1. Block diagram of methodology.

The following are details of the methodology stage:

A. Data Collection

Data was obtained, using the Scraping or Web Scraping Technique. This technique is one of the possible ways to collect content that is available on each page of a website [20]. This technique is implemented, using the python programming language, and using the Scipy module with spyder as the Scientific Python Development EnviRonment on the Ministry of Health of the Republic Indonesia website *sirs.yankes.kemkes.go.id/rsonline*.

The data collected from the scraping process were HFR data conducted by the Ministry of Health of the Republic Indonesia. They consisted of 37,518 data points, which had not been preprocessed, and contained all the resources in the hospitals in throughout Indonesia from 2013 to 2018. The data consisted of various attributes, for example number of rooms in the hospital, number of medical personnel, and equipment data in the hospital.

The dataset used in this study included resource data on hospitals in the Jakarta district (Jakarta, Bogor, Depok, Tangerang, and Bekasi), in the period from 2013 to 2018, consisting of 337 registered hospital data.

B. Preprocessing

Preprocessing is a process that aims to improve the quality of the data mining process [21], [22]. At the preprocessing stage, several processes are carried out: data cleaning, data reduction, data transformation, and data integration [23].

The data obtained from the scraping process were preprocessed, using data cleaning, data reduction, and data transformation.

1) Data cleaning: data that were still incomplete or empty were supplemented with appropriate data to avoid incomplete data; empty or non-filled attributes were filled with integer data, which corresponded to the attributes of the HFR data.
2) Data reduction, at this step there are data with attributes that are not needed in the clustering process will be reduced or eliminated, while the 113 other attributes in the dataset will be simplified to 3, that is bed in hospital, medical personnel, and list of hospital equipment.
3) Data Transformation: in this process data other than integer type were converted into an integer type, for example in the dataset there were data in the form of string type that were 'existing-function' and 'non-existent'; the data were converted into an integer by converting 'existing-function' into 1 while 'non-existent' was changed to 0. Another transformation was, that using TensorFlowLearn, this process aimed to convert integer data type into data that can be understood by TensorFlow (Tensor data type), which is `tf.int32`. Pseudo code for transforming data into Tensor data type is shown in Fig. 2.

```
# Pseudo code for preprocessing data

def normalize(v):
    norm =
tflearn.data_preprocessing.DataPreprocessing(v)
    if norm == 0:
        return v
    return v / norm
```

Fig. 2. Pseudo code for preprocessing data.

C. Clustering with K-Means and K-Medoids

1) K-Means

In the K-Means algorithm, the first thing to do is to determine the random value initialization *k* as the cluster center (centroid). The *k* value is determined for as many as four points different from each other, which are used as the center of each existing cluster.

After the four clusters, the next step is to classify and distribute the data into each cluster, so that four clusters are formed with *k* as the center of each cluster (centroid). The grouping and distribution data can be displayed in a pseudo code (Fig. 3).

where `_kmeans_clustering_model_fn` is the function that is in `tf.contrib.KMeansClustering`, which serves to classify each data with each cluster that has been defined on *class* `KMeansClustering`.

```
"""Pseudo code for grouping each data into each
cluaster"""

def _kmeans_clustering_model_fn(features,
labels, mode, params, config):
  assert labels is None, labels
  (all_scores,model_predictions,losses,is_init
  ialized,init_op,training_op) =
  clustering_ops.KMeans(_parse_tensor_or_dict(
  features), params.get('num_clusters'),
  initial_clusters=params.get('training_initia
  l_clusters'),
  distance_metric=params.get('distance_metric'
  ),
  use_mini_batch=params.get('use_mini_batch'),
  mini_batch_steps_per_iteration=params.get('m
  ini_batch_steps_per_iteration'),
  random_seed=params.get('random_seed'),
  kmeans_plus_plus_num_retries=params.get('kme
  ans_plus_plus_num_retries'))
  ...
```

Fig. 3. Pseudo code for grouping each data into each cluster.

The next step after the data has been grouped and distributed into 4 existing clusters, is to calculate the new cluster center.

The last step in this algorithm is to determine, whether the data move cluster after the calculation of a new cluster. The process can be displayed in Fig. 4, key is the result of calculation will be checked that data is experiencing movement or not.

```
"""Pseudo code for check the data are change
cluster or not"""

def transform(self, input_fn=None,
as_iterable=False):
  key = KMeansClustering.ALL_SCORES
      results = super(KMeansClustering,
  self).predict(
          input_fn=input_fn,
          outputs=[key],
          as_iterable=as_iterable)
      if not as_iterable:
        return results[key]
      else:
        return results
```

Fig. 4. Pseudo code for check the data are change cluster or not.

2) K-Medoids

In the K-Medoids algorithm six steps are carried out, which include initialization of *k* as many clusters and as a cluster center (medoid).

The first step is to determine a medoid randomly; it shoukd not overlap with the other centroids which is four medoid pieces.

The second step is to allocate and distribute each existing data into the medoid by using the euclidean distance equation. Before the data are distributed to each medoid, euclidean distance must be calculated for each predetermined medoid.

After euclidean distance is known, data are allocated to each medoid based on the specified euclidean distance. Pseudo code for distributing or allocating each data into an existing medoids is shown in Fig. 5.

```
"""Pseudo code to allocate or distribute all data
into each medoid"""

def clustering_K_Medoids(instance, medoids=4):
    row_count = len(medoids.index)
    distances = np.zeros((row_count),
    dtype=float)

    for index, row in medoids.iterrows():
        distances[index] =
euclidean_distance(instance,row)
        result = [0, 0]
    result[0] = np.argmin(distances)
    result[1] = distances[np.argmin(
                        distances)]
    return result
```

Fig. 5. Pseudo code to allocate or distribute all data into each medoid.

where `clustering_K_Medoids` is a function of grouping or allocating data in the dataset into each medoid.

```
"""Pseudo code to determine random object as new
medoids using previous function"""

def k_medoids(tf, k, max_iterations):
    row_count = len(tf.index)
    col_count = len(tf.columns)
    medoids = tf.sample(k)
    medoids = medoids.reset_index(
    drop=True)

    ...

    for index in range(0,k):
        membership.append([])

    # First time classify
    prev_medoids = medoids.copy()
    pred = np.zeros(
    row_count).astype(int)

    for index, row in tf.iterrows():
        tmp_array = clustering_K_Medoids(
         row, prev_medoids)
        pred[index] = tmp_array[0]
         membership[tmp_array[0]].append(
         index)
        error += tmp_array[1]

    best_error = error
    best_pred = np.copy(pred)
    best_medoids = medoids.copy()
     ...
```

Fig. 6. Pseudo code to determine random object as new medoids.

The third step was to determine an object randomly as new medoids, in order to replace previous medoids. This step is displayed in pseudo code in Fig. 6.

In the fourth step, after the new medoid is formed, the total deviation (*S*) of each object is calculated and the object becomes the center of the cluster.

The fifth step is to redetermine the object as a new medoid, to calculate the distance between each of the different objects in each cluster, and to calculate the total deviation, until the medoid does not change (convergence), and a cluster and each member are obtained. This step is displayed on the pseudo code on Fig. 7.

```
"""Pseudo code for calculate total deviation and
find new medoid"""

...
  for index, row in tf.iterrows():
       tmp_array = clustering_K_Medoids(
       row, prev_medoids)
       pred[index] = tmp_array[0]
       membership[tmp_array[0]].append(
       index)
       error += tmp_array[1]
       ...
```

```
"""stop condition when the distance deviation
does not change """

       if(error >= best_error):
           iteration += 1
       else:
           iteration = 0
           best_error = error
           best_pred = np.copy(pred)
           best_medoids = medoids.copy()
       i += 1
       if(iteration == max_iterations):
            is_convergence = True
```

Fig. 7. Pseudo code for calculate total deviation and find new medoid.

The last step in the K-Medoids algorithm is to repeat the determination of the new medoid until the calculation of the total deviation of the medoid does not change (convergence).

D. Validation

In this study the K-Means algorithm and the K-Medoids algorithm were compared, using the silhouette index and the rand index. Details of the comparison are explained bellow:

1) Silhouette index

Silhouette index is used to assess the quality and strength of a cluster, by assessing how precisely an object is placed into a cluster based on synthetic data and real-world data [13].

After clustering was implemented with the K-Means and the K-Medoids algorithm, the results regarding the best cluster were calculated. The test was done with the same data, namely HFR data and with different algorithms with the value of k = 4 in each algorithm. Silhouette Index was calculated using `best_pred` (the best result of clustering method using K-Means and K-Medoids models). All data were fitted, using `StandardScaler` on Scikit-Learn Python module, the best result of silhouette index within K-Means and K-Medoids algorithm was 0.4461 from K-Medoids algorithm.

2) Rand index

Rand index is a measure of similarity between data in one cluster. This validation was calculated by `labels_true` and `labels_pred` (on the K-Means and K-Medoids model), using Scikit-Lern Python module; the best result of the rand index was 0.6124 from the K-Medoids algorithm. Fig. 8 shows the silhouette index and rand index of the K-Means and K-Medoids algorithms.

Table I shows more details on clustering results from the K-Means and K-Medoids algorithms< k refers to the result of grouping hospitals based on resources the z owned.

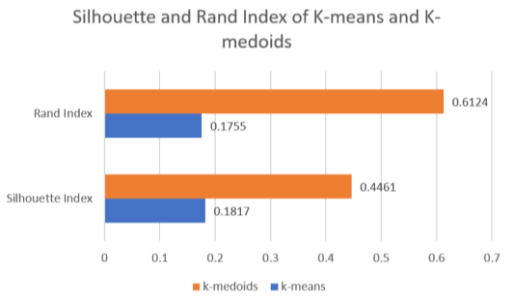

Fig. 8. The result of Silhouette and Rand Index of K-Means and K-Medoids.

TABLE I: THE COMPARISON OF SILHOUETTE INDEX AND RAND INDEX BETWEEN K-MEANS AND K-MEDOIDS

Clusuter Evaluation	Value of k	Result	
		K-Means	K-Medoids
Silhouette Index	4	0.1817	.4461
Rand Index	4	0.1755	0.6214

Table I shows that the K-Medoids algorithm outperform the K-Means algorithm is the K-Medoids. This is indicated by the higher values of the silhouette index and the rand index (close to positive 1). That is because in the K-Medoids algorithm the data at the center of each cluster are randomly determined and then iterated, whereas in the center of K-Means algorithm, is the average result of each cluster.

TABLE II: THE COMPARISON OF EXECUTION TIME BETWEEN K-MEANS AND K-MEDOIDS

Algorithm	Value of k	Execution Time (sec)
K-Means	4	2.96
K-Medoids	4	1.15

Table II shows that K-Medoids had better execution time compared with the K-Means algorithm. That is because the cluster center of K-Means is determined by calculating all the averages in the data of every cluster, so it required more time to calculate the average. Whereas in K-Medoids only initialization data were used as cluster center.

E. Result Analysis

From the studies between the K-Means and the K-Medoids algorithms, the time execution of the K-Medoids algorithm is better than that of the K-Means algorithm. Furthemore, the result of the cluster evaluation with the silhouette index and the rand index showed that the K-Medoids algorithm is better than the K-Means algorithm.

The execution time result of K-Means was 2.96 seconds and the execution time of K-Medoids was 1.15 seconds. From that result K-Medoids algorithm showed better time execution time and outperformed the silhouette index and rand index values, because the K-Medoids used robustness of medoid (use medoid as a cluster center). Therefore, the K-Medoids algorithm is more robust to outliers than arithmetic mean. Ths the hospital resources data can be better grouped using K-Medoids, rather than K-Means.

IV. RESULT AND DISCUSSION

A. K-Means Result

Experiments were carried out, using the clustering method

with K-Means algorithm, on HFR data from 2013 to 2018 consisting of 337 lists of hospitals in Jabodetabek district. Three attributes were preprocessed first, and the data were grouped into 4 clusters (cluster 0, cluster 1, cluster 2, and cluster 3). In cluster 0 is a cluster with resources in the hospital with excellent quality and completeness, if the data is in cluster 1, the hospital has good quality and complete resources, then if a data is included in cluster 2 then the hospital have enough quality and complete resources, the last one is cluster 3 which has deficient resources and quality.

The clustering model with the K-Means algorithm is created using the TensorFlow framework. The results are shown in the graph.

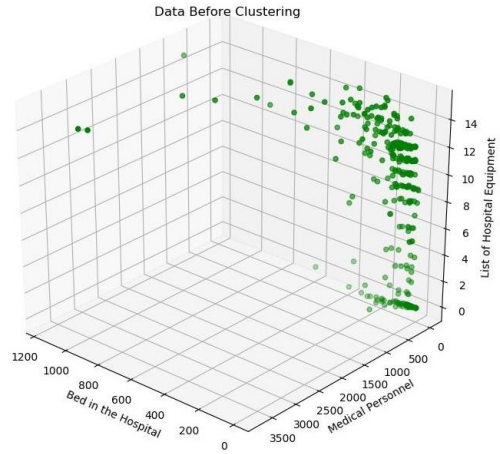

Fig. 9. Data before clustering using K-Means.

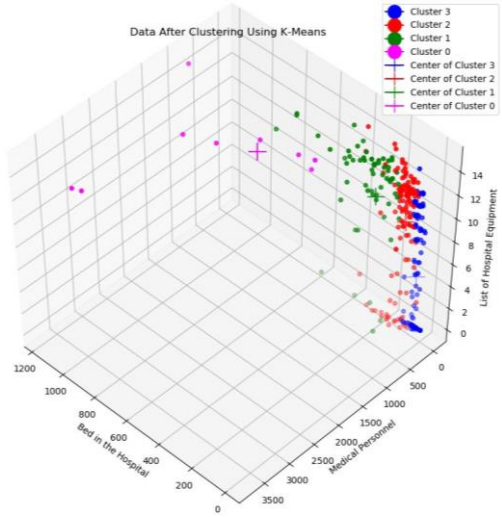

Fig. 10. Clustering result using K-Means with k=4.

Fig. 9 shows the HFR dataset from 2013 to 2018 before experiencing the clustering process with the K-Means algorithm. The dataset is visualized into a plot with a green point symbol, which is the data in the HFR. The data consist of three attributes, namely beds in the hospital attribute (the data on the number of beds in the hospital), medical personnel (data that contains the number of medical personnel available at the hospital), and list of medical equipment (data of equipment information in the hospital).

Fig. 10 displays the results of the K-Means algorithm, which produce four clusters. The cluster0 (magenta color) represent a group of hospitals with the most complete resources, so that it belongs to the group of hospitals with excellent predicate; in cluster1 (green cluster) a group hospitals that have complete resources with good predicate; in cluster2 (red cluster color) are a group of hospitals with enough predicate; while in cluster3 (blue color) are a hospital group with deficient resource predicate.

B. K-Medoids Result

After experiments on HFR data, using the K-Means algorithm, the next step was to test the same data with a different algorithm, namely the K-Medoids algorithm; the number of cluster centers (medoids) was set at 4 as during the testing with the K-Means algorithm.

In this experiment, the K-Medoids model was constructed using the TensorFlow framework and with the help of the pandas module. The results of this experiment are shown in the following graph.

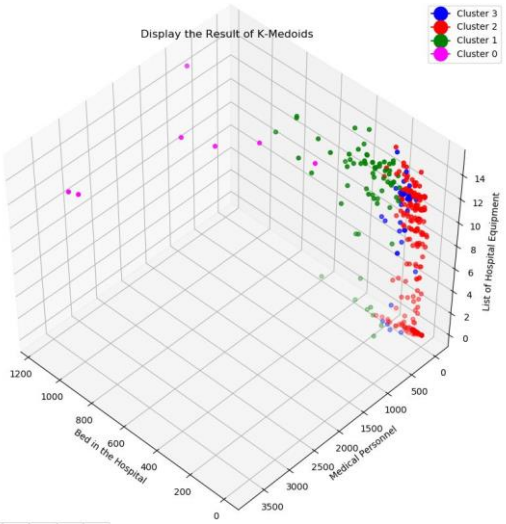

Fig. 11. Clustering result using K-Medoids with *medoids*=4.

In Fig. 11 the clustering results using the K-Medoids algorithm are shown four clusters, were produced. Cluster0 (presented with magenta color) cluster is the group of hospitals with the highest number of beds, medical personnel, and hospital equipment; therefore, Cluster1 (presented with green color) is a group of hospitals that have complete resources with a good predicate. In cluster2 (with red colors) are hospitals with enough predicate. Cluster 2 has the same shape as cluster3. The difference is the number of data on the attributes of medical personnel and beds in the hospitals. Cluster 3 (presented with blue) is a group of hospitals with predicate resources that is deficient due to the lowest number of beds in hospitals, medical personnel, and hospital equipment.

C. Comparative Analysis

The findings, obtained from this study, are the time of execution, or the time needed to do one-time clustering process with the K-Means algorithm and the K-Medoids algorithm, and the number of each cluster (member of clusters). The validation of the findings was by conducting cluster evaluation both in terms of internal evaluation and external evaluation.

The calculation of time is done by counting, beginning when the initial process starts and ending when the process stops and produces the cluster. The time obtained was different, depending on the algorithm used; namely, in the K-Means algorithm the time was 2.96 seconds, in the

K-Medoids algorithm the time obtained (was faster) with a value of 1.15 seconds. Other results obtained are the number of each cluster. When clustering with K-Means, cluster 0 consisted of 9, cluster 1 of 124, cluster 2 of 65, and cluster 3 of 139 data, respectively. When clustering with K-Medoids, cluster 0 consisted of 29 data, cluster 1 consisted of 129 data, cluster 2 consisted of 6 data, and cluster 3 consisted of 173 data.

The validation conducted in this study, is using the cluster evaluation method both internally and externally. The results obtained from the validation, namely in the K-Means algorithm, obtained the silhouette index value of 0.1817 and the rand index value of 0.1755. In the K-Medoids algorithm, the silhouette index value is 0.4461 and the rand index value is 0.6214.

V. Conclusion

The clustering method, using the K-Means algorithm and the K-Medoids algorithm, has been successfully applied to data HFR. The results obtained show that the K-Medoids algorithm outperformed the K-Means algorithm in terms of the execution time and cluster generated. Time execution for K-Medoids is better than for K-Means (2.96 seconds for execution on K-Means and 1.15 seconds for execution on K-Medoids, respectively). In addition to the time differences in execution, K-Medoids also maintained the accuracy of grouping results. This can be seen from the value of the silhouette index and the rand index, which approached positive 1.

This study showed that the clustering method can be applied to group hopitals based on beds in the hospital, medical personnel, and list of hospital equipment. Adjustments to the data used before the clustering method is applied, is very influential on the cluster results. In this case the adjustment was performed with data cleaning, which filled the incomplete data, carried out data reduction by reduction of needed attributes, and performed data transformation by changing non-integer into integer data type. From the comparison results between two clustering algorithms with K-Means and K-Medoids, it was concluded that the K-Medoids algorithm was better than the K-Means, supported by the cluster evaluation results and execution time. Hospital resources were grouped in the form of cluster 0 with excellent predicate, cluster 1 with good predicate, cluster 2 with enough predicate, and cluster 3 with deficient predicate.

References

[1] Kementerian Kesehatan Republik Indonesia, *Profil Kesehatan Indonesia*, 2016.

[2] R. I. K. Kesehatan. (2014). Peraturan Menteri Kesehatan Republik Indonesia Nomor 75 Tahun 2014 tentang Pusat Kesehatan Masyarakat. 1–24. [Online]. Available: http://aspak.yankes.kemkes.go.id/beranda/wp-content/uploads/downloads/2015/03/PMK-No.-75-ttg-Puskesmas.pdf

[3] A. A. K. Gulli, *TensorFlow 1.x Deep Learning Cookbook*, 1st ed., BIRMINGHAM-MUMBAI: Packt Publishing Ltd, 2017.

[4] R. Liu, X. Li, L. Du, S. Zhi, and M. Wei, "Parallel implementation of density peaks clustering algorithm based on spark," *Procedia Comput. Sci.*, vol. 107, pp. 442–447, 2017.

[5] J. Han, M. Kamber, and J. Pei, "Data mining concepts and techniques," *Data Mining*, 3rd ed., Morgan Kaufmann Publishers, 2011, pp. 1–38.

[6] R. Bonnin, *Building Machine Learning Projects with TensorFlow*. Birmingham: Packt Publishing Ltd., 2016.

[7] H. I. Arumawadu, R. M. K. T. Rathnayaka, and S. K. Illangarathne, *K-Means Clustering For Segment Web Search Results*, vol. 2, no. 8, pp. 79–83, 2015.

[8] Z. S. Younus *et al.*, "Content-based image retrieval using PSO and k-means clustering algorithm," *Arab. J. Geosci.*, vol. 8, no. 8, pp. 6211–6224, 2015.

[9] A. K. Jain, "Jain, Lansing - 2009 - Data Clustering 50 Years Beyond K-Means 1 Anil K . Jain Michigan State University," in *Proc. 19th Int. Conf. Pattern Recognit.*, 2010, pp. 651–666.

[10] D. F. Pramesti, M. T. Furqon, and C. Dewi, "Implementasi Metode K-Medoids clustering untuk pengelompokan data potensi kebakaran hutan / lahan berdasarkan persebaran titik panas (hotspot)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 1, no. 9, pp. 723–732, 2017.

[11] M. Anggara, H. Sujiani, and N. Helfi, "Pemilihan distance measure pada K-Means clustering untuk pengelompokkan member di alvaro fitness," *J. Sist. dan Teknol. Inf.*, vol. 1, no. 1, pp. 1–6, 2016.

[12] M. A. Masud, J. Z. Huang, C. Wei, J. Wang, I. Khan, and M. Zhong, "I-nice: A new approach for identifying the number of clusters and initial cluster centres," *Inf. Sci. (Ny).*, vol. 466, pp. 129–151, 2018.

[13] D. Fitrianah, A. N. Hidayanto, H. Fahmi, J. L. Gaol, and A. M. Arymurthy, "ST-AGRID: A spatio temporal grid density based clustering and its application for determining the potential fishing zones," *Int. J. Softw. Eng. its Appl.*, vol. 9, no. 1, pp. 13–26, 2015.

[14] F. Gargiulo, S. Silvestri, and M. Ciampi, "A clustering based methodology to support the translation of medical specifications to software models," *Appl. Soft Comput. J.*, vol. 71, pp. 199–212, 2018.

[15] C. C. Yeh and M. S. Yang, "Evaluation measures for cluster ensembles based on a fuzzy generalized Rand index," *Appl. Soft Comput. J.*, vol. 57, pp. 225–234, 2017.

[16] M. Hoffman, D. Steinley, and M. J. Brusco, "A note on using the adjusted Rand index for link prediction in networks," *Soc. Networks*, vol. 42, pp. 72–79, 2015.

[17] B. Zheng, J. Zhang, S. W. Yoon, S. S. Lam, M. Khasawneh, and S. Poranki, "Predictive modeling of hospital readmissions using metaheuristics and data mining," *Expert Syst. Appl.*, vol. 42, no. 20, pp. 7110–7120, 2015.

[18] A. E. A. Elrazek, "How can data mining improve health care?" *Appl. Math. Inf. Sci.*, vol. 11, no. 2, pp. 585–588, 2017.

[19] V. S. Magesh and T. G. Franco, *Improving Indian Healthcare Using Data Mining*, 2016, pp. 598–607.

[20] C. F. Zanon *et al.*, "Web scraping computer program for the estomato web software: A potential tool for oral medicine practice and research," *Oral Surg. Oral Med. Oral Pathol. Oral Radiol.*, vol. 124, no. 2, p. e141, 2017.

[21] M. Sadikin and F. Alfiandi, "Comparative study of classification method on customer candidate data to predict its potential risk," *Int. J. Electr. Comput. Eng.*, vol. 8, no. 6, pp. 4763–4771, 2018.

[22] F. Gürbüz, L. Özbakir, and H. Yapici, "Data mining and preprocessing application on component reports of an airline company in Turkey," *Expert Syst. Appl.*, vol. 38, no. 6, pp. 6618–6626, 2011.

[23] A. Idri, H. Benhar, J. L. Fernández-Alemán, and I. Kadi, "A systematic map of medical data preprocessing in knowledge discovery," *Comput. Methods Programs Biomed.*, vol. 162, pp. 69–85, 2018.

Yogi Wahyu Romadon was born in Cilacap, Indonesia, in 1998. He studied the bachelor's degree in informatics engineering from the Mercu Buana University, Jakarta, Indonesia, in period from 2015 to 2019.

He is currently a student in the Faculty of Computer Science. His research interest include data mining, machine learning, and deep learning.

Devi Fitrianah was born in Jakarta, Indonesia, in 1978. She received the bachelor's degree in computer science from Bina Nusantara University, West Jakarta, Indonesia, in 2000, and the master's degree in information technology and Ph.D. degree in computer science from the Universitas Indonesia, Depok, Indonesia, in 2008 and 2015, respectively.

She is currently a faculty member with the Department of Computer Science, Universitas Mercu Buana, where she is the head of Research Center.

Her research interests include image procesing, data mining, applied remote sensing, and geographic information system.