Enhancement of the K-means Algorithm for Mixed Data in Big Data Platforms

Koren, Oded; Hallin, Carina Antonia; Perel, Nir; Bendet, Dror

Document Version
Accepted author manuscript

Published in:
Intelligent Systems and Applications

DOI:
10.1007/978-3-030-01054-6_71

Publication date:
2019

License
Unspecified

Citation for published version (APA):
Koren, O., Hallin, C. A., Perel, N., & Bendet, D. (2019). Enhancement of the K-means Algorithm for Mixed Data in Big Data Platforms. In K. Arai, S. Kapoor, & R. Bhatia (Eds.), *Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys). Volume 1* (pp. 1025-1040). Springer. https://doi.org/10.1007/978-3-030-01054-6_71

Enhancement of the K-means Algorithm for Mixed Data in Big Data Platforms

Oded Koren, Carina Antonia Hallin, Nir Perel, and Dror Bendet

Article in proceedings (Accepted version*)

Please cite this article as:
Koren, O., Hallin, C. A., Perel, N., & Bendet, D. (2019). Enhancement of the K-means Algorithm for Mixed Data in Big Data Platforms. In K. Arai, S. Kapoor, & R. Bhatia (Eds.), Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys). Volume 1 (pp. 1025-1040). Cham: Springer. Advances in Intelligent Systems and Computing, No. 686 https://doi.org/10.1007/978-3-030-01054-6_71

This is a post-peer-review, pre-copyedit version of an article published in *Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference*. The final authenticated version is available online at:

DOI: https://doi.org/10.1007/978-3-030-01054-6_71

Enhancement of the K-means Algorithm for Mixed Data in Big Data Platforms

***Abstract*—Big data research has emerged as an important discipline in information systems research and management. Yet, while the torrent of data being generated on the Internet is increasingly unstructured and non-numeric in the form of images and texts, research indicates there is an increasing need to develop more efficient algorithms for treating mixed data in big data. In this paper, we apply the classical K-means algorithm to both numeric and categorical attributes in big data platforms. We first present an algorithm which handles the problem of mixed data. We then utilize big data platforms to implement the algorithm. This provides us with a solid basis for performing more targeted profiling for business and research purposes using big data, so that decision makers will be able to treat mixed data, i.e. numerical and categorical data, to explain phenomena within the big data ecosystem.**

Keywords—big data; mixed data; Hadoop; K-means

I. INTRODUCTION

Every business or organization appears to experience a data-driven revolution in management. Firms adopt big data tools to capture enormous amounts of fine-grained data derived from social media activity, Web browsing patterns, mobile phone usage, video, audio, image, and text message usage, and new formations of data generation like mobile utilizations, messages over the internet, and IOT usages [6]. The analysis of these data promises to produce insights and predictions that will revolutionize managerial decision making [21]. Possibly, the invention of big data is "the most significant "tech" disruption in business and academic ecosystems since the meteoric rise of the Internet and the digital economy" [2].

As big data involves the ability to render into data many aspects of the world that have never been quantified before, also referred to as "datafication" [8], the challenge for businesses is to develop better and more simple algorithms, systems, and processes that can make sense of all of the heterogeneous and fragmented information on the Web. Publications in information and management science on big data are increasingly grappling with such challenges. Top-tier information science journals, such as *Management Science* and *MIS Quarterly*, have commissioned special issues on data science, analytics, and big data, and, recently, journals on big data have been launched [2].

A big data ecosystem includes a platform that is enabled to handle a huge amount of data (in several levels) via a variety of tools. The use of big data technologies is associated with the emergence of new technical skills, such as Apache™ Hadoop®!, MapReduce, Apache Pig!, Apache Hive TM, and Apache HBase™ [27]. The early adaptation of big data tools attracted media attention, such as when Sears started to experiment with Apache™ Hadoop®!, and this was central to the first wave of big data investments. Of course, Sears learned Apache™ Hadoop®! the hard way, through trial and error, since it had only a few outside experts available to guide its work when it introduced the software in 2010 [16].

The processes on large amounts of data that can be stored in Hadoop Distributed File System (HDFS™) can be executed via MapReduce jobs [9]. Furthermore, there are other functionalities, possibilities, and tools that can enable the analysis of information for various business purposes (such as machine learning algorithms). The ability to combine big data tools with different data analysis functionalities, such as Apache HIVE TM and Apache Pig!, is growing [12], [18], [22], as is the variety of other big data tools designated for handling data (like ETL process and analysis) [28]. Big data is also being studied in relation to machine learning tools such as Apache Mahout™ [19].

The approaches to dealing with the structuring of the massive volumes of data in big data are performed by different capabilities and tools [10], [28]. Apache™ Hadoop®! is a platform that includes the ability to store, manage, read, write, and operate on massive amount of data/files via HDFS™, a system based on the Google File System (GFS) [14] with the capability of analyzing the information for different purposes [28]. Although these approaches have advanced the capabilities of dealing with massive data, they do not offer

algorithms that can structure data effectively for analytical and decision making purposes. For example, IBM's Watson may be on the cutting edge in natural language processing, but it has a long way to go in terms of the system's capability for absorbing and interpreting big data across the Internet [2]. These observations reflect a need to develop new approaches for structuring and categorizing massive amounts of data in an emergent big data ecosystem.

K-means is a popular data clustering method. It is a simple and elegant approach to partitioning a dataset into K distinct clusters. This algorithm was originally described by [20]. First, a value of K is specified, and then the algorithm assigns each observation from the data set to exactly one of the K clusters. The assignment decision is done by minimizing the 'differences' between observations which belong to the same cluster. These differences are commonly measured by squared Euclidean distance, but there are many other possible ways to define this concept. A recent example involving K-means utilizations can be found in [11], where the authors studied how different types of communities may affect the effectiveness of open source software. In addition, [13] used the K-means method to investigate and identify different type of user roles in innovation-contest communities. Reference [25] applied the K-means algorithm to studying time varying effects on the allocation of marketing resources. And finally, [15] used K-means to analyze doctor's profiles.

One of the challenges with using the K-means algorithm has been that the algorithm works well with numeric data, but is not directly applicable to non-numeric, categorical data [4], since the Euclidean distance function is not meaningful when considering categorical values.

This paper presents a novel approach, which overcomes the difficulty of working with mixed data for decision making in big data. We address the question of how K-means algorithms can solve the problem of clustering mixed data in big data.

The performance of the K-means algorithm on categorical data has been studied in the information science literature, which describes how it converts multiple category attributes into binary attributes and then treats them as numeric [24]. However, this method may greatly increase the computational effort, especially when working with big data. Consequently, scholars have applied K-modes algorithms and the K-prototypes algorithm [17]. The K-modes algorithm extends the K-means method to clustering categorical data by defining differences between clusters in terms of frequencies and by considering modes instead of means. The K-prototypes algorithm is a mixture of the K-means and the K-modes algorithms. That is, the definition of a "cluster center" (or representative) allows treating a clustering problem with categorical variables to be a traditional K-means problem [26]. The general method of choosing a representative of a cluster and measuring dissimilarities between clusters is performed by relative frequency-based methods [3] or in studies applying the K-means algorithm on mixed data [3], [29]. However, the latter studies were not performed in a big data environment. For example, the numerical studies presented in [3] considered datasets with at most 690 elements.

Our contribution is to adapt the K-means algorithm on mixed big data. That is, we have used big data platforms (in terms of parallel computation techniques and storage capabilities) in order to explore how the K-means algorithm works on big data with both numeric and non-numeric variables. Since data size expands tremendously, analyzing data on a single machine is inefficient. Therefore, considering parallelism within a distributed computational framework is the most appropriate solution. One of the most common programming frameworks for processing large scale datasets through the utilization of parallelism is MapReduce [9] and the exploitation of the qualities of parallel computing [5], [7].

In this paper, we address two questions: (i) We provide a clustering algorithm which handles both numeric and categorical attributes in big data environments, based on the capabilities of big data tools and the K-means algorithm; (ii) We explore how the results of the algorithm in a big data environment, based on the ability to support complex architectures, can provide the extension of capabilities, such as clustering, profiling, analysis and predictions.

Our algorithm enables the application of the K-means algorithms to both numerical and non-numerical data. The empirical evidence is broadly supportive of the two issues we seek to address. We first create a procedure that "flattens" all the data from categorical and numerical data to pure numerical data. We then filter all the categorical classes into distinct groups, based on the categorical combinations, which allows us to analyze each group separately (since we are dealing with big data, the grouping process and the K-means process are performed via big data platforms). That is, we perform the K-means algorithm only on the remaining numeric variables. Lastly, we collect all the groups' analysis outcomes, which can serve as a basis for further analysis, in order to support the organization requirements and business needs.

The implication of our study involves the presentation of a method for treating mixed data in big data which was not previously possible. The approach advances the capabilities of dealing with massive data, such as in decision making, since profiling, forecasting, and other analyses can be performed in a more targeted manner.

Recent studies have discussed the relation between big data and theory. For example, in [23], the author states that big data and theory can be synergistic for exploring phenomena or problem solving by using the big data platforms and tools to generate theoretical insights and by not starting with a preconceived theory. Furthermore, in [1], the author indicates that "big data has potentially important implications for theory." On the one hand, theory can be replaced by patterns derived from data. On the other hand, data without theory lacks order, sense, and meaning. We have adopted the concept presented in these studies. That is, we present a method for analyzing data in big data environment, where this method can be applied for any relevant theoretical issue.

The rest of the paper is organized as follows. Under Model Development, we present our new alternative procedure for performing the K-means algorithm with mixed data in a big data environment. We then turn to an implementation example of the proposed procedure on a generated dataset of approximately 1GB, while in the last part we discuss and conclude the paper.

II. MODEL DEVELOPMENT

We argue that K-means applied to mixed data can enhance decision making within the big data ecosystem and allows decision makers to treat massive amounts of data. The current study thus analyzes the impact of K-means applied to both numerical and categorical (non-numerical) data in big data platforms. The model assumes a dataset which includes m categorical variables and n quantitative variables, and that categorical variable j may have $a_j \geq 2$ different states.

The K-Means Algorithm Procedure:

Claim 1: Non-numeric data in big data can be assigned values.

Proof: We first perform the K-means algorithm on our dataset by adopting the following steps:

1. Create $\prod_{j=1}^{m} a_j$ different types of groups, which differ by their values of the categorical variables. Each record is assigned to its group, according to its categorical values.
2. Each group generated in step 1 is a file (or other storage format) in the big data platform (this will enable parallel computing in the next steps).
3. Perform a parallel K-means algorithm on all groups according to the numeric variables.
4. Aggregate all the clusters (K clusters from each group) from step 3 to one outcome for further analysis.

As a simple example, assume we have two categorical variables and three numeric variables, as follows: Gender – male/female; marital status – single/married/divorced; income; age and number of children. Table 1 presents the first step in the data mining process, which is to create six groups that differ by the values of their categorical variables.

TABLE 1: THE DATA MINING PROCESS OF CREATING GROUPS WITH NUMERICAL AND CATEGORICAL DATA

Group	Categorical variables
Group 1	Male and single
Group 2	Male and married
Group 3	Male and divorced
Group 4	Female and single
Group 5	Female and married
Group 6	Female and divorced

At step 2, each one of these groups is saved as a file containing all the records with the same categorical attributes. At step 3, the K-means algorithm is performed (according to the numeric variables) simultaneously on all groups, so that, for each group, we get K clusters (the K may be different for each group, depending on the decision needs and requirements, and the number of records in each group). Step 4 is optional, and is performed according to the needs of the research.

III. IMPLEMENTATION EXAMPLE OF THE ALGORITM

The following section presents an end-to-end implementation example.

1. Upload the data set and categorical files to the HDFS™ (in Apache™ Hadoop®![1]).

   Pre-set: each of the $\prod_{j=1}^{m} a_j$ possible combinations of the values of each categorical variable is in a separated file. Each file contains the records with the corresponding categorical values. This is a mandatory step due to the fact that there is a need to create all combinations of the available states based on the definition/business requirements. Note that there might be empty files (groups) if there are no records with the corresponding categorical values.
2. Multiplication of all the files (from step 1) can create multiple lines. Each line describes a unique combination. All lines are stored in a file in HDFS™ (in Apache™ Hadoop®!) for parallel analysis (in big data platform).
3. Filter the dataset for each unique file (form step 2) and send the relevant quantitative variables to the relevant file.
4. Run (via bash script) K-means (Apache Mahout™[2]) on each file that is located in a separated directory (from step 3) with the following parameters:
   i. A configurable parameter x for the number of iterations (in this use case we use 5 iterations for all K-means runs)
   ii. A Number of clusters (K), which is influenced by the number of records per each unique file (from step 3). The number of clusters K increases when the number of records per file grows.
5. Gather all the clusters to one defined structure for additional analysis (compare between clusters, order, analysis, etc.).

Note that steps 1 to 3 were implemented and tested on a single-node environment. The Apache Pig! code operation includes (see next section for a detailed description):

1. Loading the full dataset.
2. Creating all the categorical variables combinations (3 categorical variables with different states – total of 36 groups in this use case example).
3. Filtering the relevant Categorical variables and creating the groups/files (per each combination) with the relevant filter quantitative variables (5 variables in this use case example).

The total run duration time of steps 1 to 3 in our example was 3:21 minutes in average using a single-node environment. We neglected to include this amount of time with the total time, since it is relatively small compared with the total K-means running time. The procedure flow diagram is presented in Figure 1, while Table 2 describes the implementation steps and guidelines.

[1] http://hadoop.apache.org/

[2] http://mahout.apache.org/

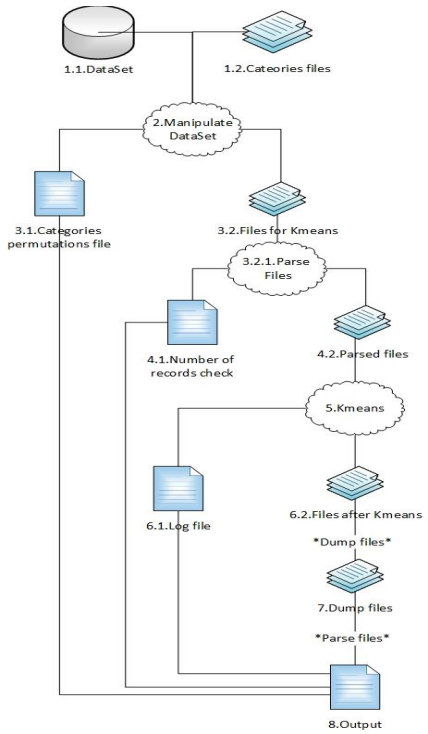

Fig. 1 *Procedure flow*

TABLE 2 PROCEDURE IMPLEMENTATION GUIDELINES

Step	Name	Deception	Input	Output	Implantation note
1.1	DataSet	Data set file			Load the DataSet to Apache™ Hadoop®! (HDFS™)
1.2	Categories files	Unique separated file for each category with all states			Load the DataSet to Apache™ Hadoop®! (HDFS™)
2	Manipulate DataSet		1.1.DataSet 1.2.Categories files	3.1.Categories files 3.2.Files for K-means	APACHE PIG!
3.1	Categories permutations file	All permutations of the categories states			Result of cross (Cartesian product) on 1.2.Categories files
3.2	Files for K-means	Make files to run K-means on them			
3.2.1	Parse files	Manipulate 3.2 files for K-means files	3.2.Files for K-means	4.1.Length check 4.2.Parsed files	Delete the category column
4.1	Number of records check	Checks the number of records			Inner validation – uses in 8.Output
4.2	Parsed files	Files are ready to run in K-means (in directories)			
5	K-means	Run K-means on each directory	4.2 Parsed files	6.1.Log file 6.2.Files after K-means	Apache Mahout™. K driven from number of records. Max of iterations is predefined.
6.1	Log file	Record the time for each run			
6.2	Files after K-means	Result of K-means algorithm			
7	Dump files	Dump of the result of 6.2 Files after K-means			Apache Mahout™ Apache™ Hadoop®!
8	Output	Aggregate all dump files to csv/excel			Parsing 7.Dump files and aggregate them to csv/excel file

IV. DATA STRUCTURE

A. *Before Manipulation*

To evaluate the performance of our K-means procedure, we tested a fictive sample in the big data ecosystem. The following nine variables were used as part of the sample data set implementation of a use case: 5 quantitative variables and 3 categorical variables. Table 3 presents the names and values of the 3 categorical variables.

TABLE 3 STATES OF CATEGORIES

#	Variable	Number of values	Values
1	Marital status	3	Single, Married, Divorcee
2	Age range (selection)	3	20-34, 35-49, 50-64
3	Academic degree	4	Non, First degree, Second degree, Third degree

The example illustrates the challenge of using, in some cases, the average of categorical values in

case of K-means. Suppose that K-means clustering algorithm finds the marital status average of 1.5. What does it mean? Half single? Almost Married?

Next, we combined a sample of quantitative variables with the categorical data. Table 4 presents the list of the 5 quantitative variables.

TABLE 4 QUANTITATIVE VARIABLES

#	**Variable**
1	Salary (amount)
2	Clothing spending (month)
3	Distance from work (km)
4	Working hours (average day)
5	Food spending (month)

TABLE 5 RAW DATA EXAMPLE BEFORE MANIPULATION

Age	Acad Degr.	Marital status	Salary	Clothing spending per month	Distance from work (km)	Working hours (average day)	Food spending (month)
20-34	Second degree	Married	4,801	677	106	5	322
20-34	First degree	Divorcee	5,244	2,396	87	6	4,388
35-49	First degree	Single	5,566	3,958	28	9	2,236
20-34	Non	Single	1,776	637	61	8	1,680

B. After Manipulation

To create different states and make reactive the averages in the categorical variables, we performed a Cartesian product between all categorical variables, so that we had $3\times3\times4=36$ distinct groups as follows:

$$\boldsymbol{3}_{\text{Marital status 3 states}} * \boldsymbol{3}_{\text{Age range 3 states}} * \boldsymbol{4}_{\text{ Academic degree 4 state}} = \boldsymbol{36\ distinct\ groups} \quad (1)$$

The data set was transformed into the following new data set that includes all the categorical permutations. An example of a recode is presented in Table 6.

TABLE 6 CATEGORICAL PERMUTATIONS EXAMPLE

Filed	**Value**
Category	20-34_First-degree_Divorcee
Salary	1015
Clothing spending (month)	4274
Distance from work (km)	68
Working hours (average day)	11
Food spending (month)	2466

Table 7 presents the size and capacity of the dataset that was used for this implementation example.

TABLE 7 DATA CAPACITY

Parameter	**Size**
Size	~1GB (975MB)
Total Number of records	19,600,000
Number of groups/files* (see algorithm description)	36

Meanwhile, Table 8 presents the entire list of all 36 files/groups permutations (from the full dataset) and their capacity (number of records and size in Kbytes).

TABLE 8 GROUP FILE PERMUTATIONS AND CAPACITY

#	Name/ description	Size (KBytes)	Number of records
1	20-34_First-degree_Divorcee	11,364	537,432
2	20-34_First-degree_Married	11,312	516,264
3	20-34_First-degree_Single	11,260	573,888
4	20-34_Non_Divorcee	11,148	520,576
5	20-34_Non_Married	11,144	575,848
6	20-34_Non_Single	11,048	553,896
7	20-34_Second-degree_Divorcee	11,036	536,256
8	20-34_Second-degree_Married	11,024	546,056
9	20-34_Second-	11,004	571,536

	degree_Single		
10	20-34_Third-degree_Divorcee	10,940	565,264
11	20-34_Third-degree_Married	10,924	522,928
12	20-34_Third-degree_Single	10,912	558,600
13	35-49_First-degree_Divorcee	10,888	554,680
14	35-49_First-degree_Married	10,828	552,328
15	35-49_First-degree_Single	10,788	559,776
16	35-49_Non_Divorcee	10,784	539,392
17	35-49_Non_Married	10,740	543,704
18	35-49_Non_Single	10,720	526,848
19	35-49_Second-degree_Divorcee	10,700	558,208
20	35-49_Second-degree_Married	10,680	530,376
21	35-49_Second-degree_Single	10,664	531,944
22	35-49_Third-degree_Divorcee	10,656	558,208
23	35-49_Third-degree_Married	10,624	536,648
24	35-49_Third-degree_Single	10,604	565,656
25	50-64_First-degree_Divorcee	10,588	542,136
26	50-64_First-degree_Married	10,576	541,352
27	50-64_First-degree_Single	10,568	546,840
28	50-64_Non_Divorcee	10,524	536,648
29	50-64_Non_Married	10,492	539,392
30	50-64_Non_Single	10,480	545,664
31	50-64_Second-degree_Divorcee	10,432	529,592
32	50-64_Second-degree_Married	10,400	551,544
33	50-64_Second-degree_Single	10,316	508,032
34	50-64_Third-degree_Divorcee	10,272	548,800
35	50-64_Third-degree_Married	10,196	532,728
36	50-64_Third-degree_Single	9,996	540,960
Total		386,632	19,600,000

V. OUTPUT STRUCTURE

The outcomes of the implementation use case include the following products:

- Running log
- Output file – union of all outcomes clusters for future analysis and additional insights
- Performance example

A. Running Log

The running log in Table 9 was designed for the implementation use case and is a text structure file that includes the starting time and the ending time for the entire process. The log also contains the following values per each file/category (permutation).

- Time (start/initiate)
- Time (start for each iteration)
- Number of rows/records
- Number of max iterations
- Number of selected K (clusters)
- Time (finish/end)

TABLE 9 EXAMPLE OF LOG OUTCOMES (PARTIAL LOG)

```
. . .
20-34_First-degree_Divorcee - new terminal
- 2017-06-01--04:01:14.680
20-34_First-degree_Divorcee - before -
2017-06-01--04:01:14.757 n = 1371 k = 10
num of iter = 2
20-34_First-degree_Married - new terminal -
2017-06-01--04:01:17.832
20-34_First-degree_Married - before - 2017-
06-01--04:01:17.912 n = 1317 k = 10 num of
iter = 2
. . .
. . .
. . .
35-49_Third-degree_Single - new terminal -
2017-06-01--04:03:23.431
35-49_Third-degree_Single - before - 2017-
06-01--04:03:23.575 n = 1443 k = 10 num of
iter = 2
50-64_First-degree_Divorcee - new terminal
- 2017-06-01--04:03:29.319
50-64_First-degree_Divorcee - before -
2017-06-01--04:03:29.506 n = 1383 k = 10
num of iter = 2
. . .
. . .
```

B. Clusters Log

The clusters log includes the K-means results per each permutation (group/file). To ensure the possibility of advanced/additional analysis, machine learning capabilities, different AI functionalities, and different BI opportunities, we decided to gather the entire K-means results (from all groups/files) into a structure that allows us to identity the specific

categorical permutation. Note that each group can have between 0 to K clusters. The value of K per each group is in the log file (see Running log).

The head line of the clusters log contains the list of variables designated as the categorical, cluster length, c type (vector of the mean values of the centroid) or r type (vector of cluster's radius).

Each record in the clusters log presents the K-means results per each group in the order of the variables as described in the head line of the Clusters log. Remember that the K-means algorithm is performed only on the quantitative variables after the partition of all records into their corresponding groups.

Please also note that the Clusters log enables advanced BI, AI, and additional machine learning algorithm functionalities on the results (from simple queries, such as sorting, ordering, and selecting, to more complicated and sophisticated possibilities of comparing between clusters, running additional machine learning algorithms on the Clusters log, and other functionalities).

TABLE 10 CLUSTER LOG OUTCOMES (PARTIAL CLUSTERS LOG)

```
Categories, n, c:Salary, c:Clothing-spending-
(month), c:distance-from-work-(km),
c:working-hours-(avrage day), c:food-
spending-(month), r:Salary, r:Clothing-
spending-(month), r:distance-from-work-(km),
r:working-hours-(avrage day), r:food-
spending-(month)
20-34_First-
degree_Divorcee,154,8839.682,3392.682,71.799,
7.922,3145.383,779.165,696.034,40.717,2.062,1
060.447
20-34_First-
degree_Divorcee,99,7546.01,1480.99,75.121,8.0
61,730.192,1087.673,864.4,43.121,1.963,341.53
2
20-34_First-
degree_Divorcee,158,2615.892,1613.797,78.918,
7.975,3403.715,880.222,938.229,42.779,1.949,6
99.134
20-34_First-
degree_Divorcee,114,3329.649,3898.719,79.518,
8.289,2801.044,924.086,423.515,40,2.003,1053.
317
20-34_First-
degree_Divorcee,158,8847.247,1897.728,75.146,
7.848,2225.57,747.829,907.461,43.077,1.991,10
99.612
```

C. Performance Example - Running Times

Table 11 presents 5 runs of steps 1-3 in the above implementation procedure:

TABLE 11 DATASET MANIPULATION RUN TIME

Pig Runtime			
Average	***Running time***	***Ending time***	***Starting time***
00:03:21	00:03:18	23:59:10	23:55:52
	00:03:21	00:05:50	00:02:29
	00:03:21	00:10:56	00:07:35
	00:03:21	00:15:13	00:11:52
	00:03:22	00:19:44	00:16:22

In Table 12, we present the running times of step 5 (as described in the procedure flow) for the above example, where we executed the process in big data multi-node environment. Table 13 presents the results for a single-node environment.

TABLE 12 K- MEANS IN MULTI-NODE ENVIRONMENT

Total runtime	**Run #**
00:26:39.339	M1
00:24:53.432	M2
00:26:01.409	M3
00:28:46.028	M4
00:24:44.687	M5
00:24:03.577	M6
00:25:35.413	M7
00:26:05.337	M8
00:24:04.655	M9
00:24:23.075	M10

Table 13 K- MEANS SINGLE-NODE ENVIRONMENT

Total runtime	**Run #**
03:21:08.783	S1
03:20:44.000	S2
03:15:14.948	S3
03:16:03.084	S4
03:21:31.207	S5

The differences between the running times in Tables 12 and 13 are well observed. Indeed, working in a multi-node environment allows us to perform tasks in parallel, and therefore is much more efficient.

VI. IMPLICATIONS OF THE K-MEANS ALGORITHM IMPLEMENTATION IN BIG DATA

A. Limitations

While we found that the implementation of the K-means algorithm worked well in the runs, the complexity analysis of our suggested procedure must be tested in future studies. However, we argue that the complexity of our process is better when comparing it to the complexity of a regular K-means algorithm that runs on the full dataset, due to the ability to reduce the size of the dataset. The procedure will run on subsets that possess less recodes per group. This will influence the number of K-means iterations per group.

Also note that in a big data environment, all the K-means calculations can be done in parallel, i.e. in a different datanodes. Therefore, we believe that the complexity will be mostly influenced by the size of the largest group that will be generated.

B. Implications for Theory

We presented a method for analyzing data in big data environment, where this method can be applied for any relevant theoretical question in a big data environment. For example, exploring a certain phenomenon, derive patterns from data, improve decision making methods and run predictions.

C. Implications for Practice

This paper presents a new approach, which overcomes the difficulty of working with mixed data for decision making in a big data environment. The power of clustering and narrowing down the profiles to targeted groups, based on the business needs, improves the decision making process.

REFERENCES

[1] A. Abbasi,, S. Sarker, and R. H. Chiang, “Big data research in information systems: toward an inclusive research agenda,” Journal of the Association for Information Systems, 17(2), 2016.

[2] R. Agarwal, and V. Dhar, “Editorial—big data, data science, and analytics: the opportunity and challenge for IS research,” Information Systems Research, 25(3), 2014, pp. 443-448.

[3] A. Ahmad, and L. Dey, "A K-mean clustering algorithm for mixed numeric and categorical data," Data & Knowledge Engineering, 63(2), 2007, pp. 503-527.

[4] P. Berkhin, "A survey of clustering data mining techniques," Grouping multidimensional data. Springer: Berlin Heidelberg, 2006, pp. 25-71.

[5] X. Cai, F. Nie, and H. Huang, "Multi-View K-Means clustering on big data," IJCAI, 2013.

[6] Cisco, “The Zettabyte era: trends and analysis,” 2016, White paper

[7] X. Cui, P. Zhu, X. Yang, K. Li, and C. Ji, "Optimized big data K-means clustering using MapReduce." The Journal of Supercomputing 70(3), 2014, pp. 1249-1259.

[8] K. Cukier, and V. Mayer-Schoenberger, “The rise of big data: how it’s changing the way we think about the world,” Foreign Affairs, 92(3), 2013, pp. 28-40.

[9] J. Dean, and S. Ghemawat, "MapReduce: simplified data processing on large clusters." Communications of the ACM 51(1), 2008, pp. 107-113.

[10] Y. Demchenko, C. Ngo, and P. Membrey, "Architecture framework and components for the big data ecosystem." Journal of System and Network Engineering, 2013, pp. 1-31.

[11] D. Di Tullio, and D. S. Staples, "The governance and control of open source software projects." Journal of Management Information Systems 30(3), 2013, pp. 49-80.

[12] G. Engelberg, O. Koren, and N. Perel, "Big data performance evaluation analysis using Apache Pig," International Journal of Software Engineering and Its Applications, 10(11), 2016, pp. 429-440.

[13] J. Füller, K. Hutter, J. Hautz, and K. Matzler. "User roles and contributions in innovation-contest communities." Journal of Management Information Systems 31(1), 2014, pp. 273-308.

[14] S. Ghemawat, H. Gobioff and S.T. Leung, “The Google file system,” ACM SIGOPS operating systems review, Vol. 37, 2003, pp. 29-43..

[15] S. Guo, X. Guo, Y. Fang, and D. Vogel. "How doctors gain social and economic returns in online health-care communities: a professional capital perspective." Journal of Management Information Systems 34(2), 2017, pp. 487-519..

[16] D. Henschen, “Why Sears is going all-in on Hadoop.” InformationWeek, 2012.

[17] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," Data mining and knowledge discovery Volume 2, 1998, pp. 283-304.

[18] D. Kendal, O. Koren, and N. Perel, "Pig vs. hive use case analysis," International Journal of Database Theory and Application 9(12), 2016, pp. 267-276.

[19] S. Landset, T. M. Khoshgoftaar, A. N. Richter, and T. Hasanin. "A survey of open source tools for machine learning with big data in the Apache™ Hadoop®! ecosystem." Journal of Big Data, 2(1), 2015, p 24.

[20] J. B. MacQueen, “Some methods for classification and analysis of multivariate observations,” Proceedings of the 5th

Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[21] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh and A. H. Byers, “Big data: The next frontier for innovation, competition, and productivity,” McKinsey Global Institute, 2011.

[22] R. A. Preethi, and J. Elavarasi. "Big data analytics using Hadoop tools—Apache Hive vs Apache Pig,” International Journal of Emerging Technology in Computer Science & Electronics, 24(3), 2017.

[23] A. Rai, “Synergies between big data and theory,” Management Information Systems Quarterly, 40(2), 2016, pp. iii-ix.

[24] H. Ralambondrain, "A conceptual version of the K-means algorithm," Pattern Recognition Letters 16(11), 1995, 1147-1157.

[25] A. R. Saboo, V. Kumar, and I. Park, “Using big data to model time-varying effects for marketing resource (re) allocation,” MIS Quarterly, 40(4), 2016.

[26] O. M. San, V.-N. Huynh, and Y. Nakamori, "An alternative extension of the k-means algorithm for clustering categorical data," International Journal of Applied Mathematics and Computer Science, 14(2), 2004, pp. 241-248.

[27] P. Tambe, “Big data investment, skills, and firm value,” Alok Gupta, 2014, pp. 1452-1469.

[28] T. White, "Hadoop: The Definitive Guide", 4th edition, OReilly Media: Sebastopol, 2015.

[29] R. Xu, and D. Wunsch, "Survey of clustering algorithms," IEEE Transactions on neural networks, 16(3), 2005, pp. 645-678.