**International Journal of Electrical and Computer Engineering (IJECE)**
Vol. 9, No. 1, February 2019, pp. 409~416
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i1.pp409-416


# Granularity analysis of classification and estimation for complex datasets with MOA

**Chanintorn**
Jittawiriyanukoon, Graduate School of ELearning, Assumption University, Thailand

**Article Info**

***Article history:***

Received Mar 15, 2018
Revised Jul 26, 2018
Accepted Aug 16, 2018

***Keywords:***

Big data curation
Classification
Estimation
Granularity level
MOA
Parallel processing
Regression based machine learning

**ABSTRACT**

Dispersed and unstructured datasets are substantial parameters to realize an exact amount of the required space. Depending upon the size and the data distribution, especially, if the classes are significantly associating, the level of granularity to agree a precise classification of the datasets exceeds. The data complexity is one of the major attributes to govern the proper value of the granularity, as it has a direct impact on the performance. Dataset classification exhibits the vital step in complex data analytics and designs to ensure that dataset is prompt to be efficiently scrutinized. Data collections are always causing missing, noisy and out-of-the-range values. Data analytics which has not been wisely classified for problems as such can induce unreliable outcomes. Hence, classifications for complex data sources help comfort the accuracy of gathered datasets by machine learning algorithms. Dataset complexity and pre-processing time reflect the effectiveness of individual algorithm. Once the complexity of datasets is characterized then comparatively simpler datasets can further investigate with parallelism approach. Speedup performance is measured by the execution of MOA simulation. Our proposed classification approach outperforms and improves granularity level of complex datasets.



***Corresponding Author:***

Chanintorn,
Jittawiriyanukoon, Graduate School of ELearning,
Assumption University, Thailand.
Email: pct2526@yahoo.com

## 1. INTRODUCTION

Complex datasets can be the prospects and inquiries they affect the data analytics. The complexity of datasets is the indication of difficulty data scientist experiences as curating the insights–a complex dataset is usually more problematic to classify than regular dataset, and generally involves a diverse set of technical approaches to figure so [1]. Complex datasets require increased effort to outline the data prior to visualization and curation. To characterize the complexity of datasets is essential as well as the forthcoming complexity is to be taken into account. Big data represents complex dataset hence massive amount of data slows the high speed computers down close to bottleneck stage in order to calculate and extract insights [2], [3]. Other implications derive from distinctive sources. Various sources can generate disorganized datasets or datasets succeed dissimilar structures. Data must be preprocessed in order to comply with primary repository format.

In order to iron out the bottleneck problem of complex dataset processing, data transformation and refining steps (pre-processing) help reduce processing power and time. Besides data mining approach based upon the integration of knowledge is introduced. The pre-processing steps of business oriented data are opted to form an ontology ambitious information system (OAIS). The knowledge base is then determined to help sort out the post-processing of interpretation. Finally, the integration of objective and subjective criteria in teaching is evaluated to develop an expert knowledge.Pre-processing of datasets incorporates normalization, attribute extraction, noise removal, classification and structure re-configuration. Nawi et al. [4] have presented an artificial neural network based algorithm for data pre-processing. The algorithm has turned out

to be common and becomes an analytical tool for mining pattern recognition and machine learning. Big data has been mined using parallelism approach as introduced in [5]. This mining approach has not mentioned how to discard redundant and messy data which is important in preprocessing steps. The relation between preprocessing and complex dataset with technological approaches has been experimented in [6]. Various frameworks for analytical tools like Flink, Spark and MapReduce are also issued for complex data learning. Insights from big data curation and the infrastructure for analytics at Twitter have been presented by [7]. A dynamic role in assisting data scientists with big data has been emphasized, but comprehensive insights are not available. Data analytics from several algorithms must be aggregated into production system, but they achieve in sharing outputs for academic intellect at Tweeting.

In this research, the performance of several pre-processing models in order to specify granularity level, decrease noisy samples and correct possible error of the training samples is investigated. The main objectives are to confirm accuracy of classification, simplify the computation and to excel preprocess. Bayesian, Boosting, Nearest Neighboring and the proposed classification models are introduced in this research. Additionally, the complex datasets proceed to be executed at post-processing environment. To accelerate the post-processing calculation, the parallel processing system as presented in [8] is employed. The MOA simulation [9] results and speedup performance are summarized. In the simulations, complex datasets obtained from public repository are used. The continuing part is written as follows: Section 2 and 3 expose the theoretical context of complex dataset characteristics and the pre-processing approaches respectively. Section 4 presents the parallel estimation model. Results and analysis finally is established in section 5.

## 2. COMPLEX DATASETS

It is known that there is a debate about “big data”. It is about a complexity per se. The data with difficulty in handling is the matter of size. Enormous effort in making use of big size of data, just to point out where to manipulate is mandatory. Complexity reflects a tedious task. Not to mention, even a trivial dataset can parade complexity causing data scientists hard to mine with current techniques.

Data from various senders, or different datasets from the same sender, is structured dissimilarly. For instance, one unit has few different files–while another unit stores the information on a database. Furthermore, in some of the database instance there is duplicate content which is identical to files content. To make use of data from multiple sources, without duplicating or losing information, necessitates pre-processing task [10].

As a definition of “big data,” the collected data size can upset both processing units and applications used to analyze. Size can be in petabytes (PB)–the taller the dataset is, the more problematic to squeeze them on built-in-memory while processing. Let A denote a given dataset matrix which contains a rows and b columns [Ai1, Ai2, Ai3,…, Ai(b-1), Aib] for each i= 1, 2, 3,…, a. The A matrix is presumed to be a deterministic set. Obviously, state space of the dataset becomes [a, b] and computational cost is O (ab) [11].

The level of granularity is vigorous for development of full report or dashboard and data integration or visualization. It is simpler for developer to drill-down into the latest detail of datasets–nevertheless, this is a balance between data indexing and the computational cost of analytical depth. Data curation which appreciates granular drill-down deals with the involvement of bigger adhoc based amount of data due to the ignorance of data integration, summary and pre-process.

Diverse databases communicate dissimilar query languages. Structural Query Language is the principal communications of querying data from central Relational Database, but if a third party hardware is used then syntax and API have to be interfaced, and additionally communication protocols and the internal database structure must be exploited to access. Analytical tool is to be elastic in order to approve the built-in connection to destined database through API unless a bulky process of extracting data to SQL database/warehouse is invalidated [12].

Processing with multimedia data warehoused in table style (.csv) is a burden, but unstructured massive data is another tedious task, since it is a rich-text oriented dataset plus video and audio streams. Various types of data exhibit diverse rules, and compromising a single type of truth data among all is critical in order to produce decisions making [13].

Disseminated data occurs whenever data is stored in several places, for instance, at work place, in clouds, or different branches. These data is isolated and to collect them all is not easy. Not to mention, after collection–some standardization, normalization and cleansing are compulsory prior to the different datasets can be cross-referenced and manipulated. Location based dataset is gathered regarding to the related objectives and applications [14].

Lastly, not only current data is taken into account but the forthcoming speed of data (growth rate) is also considered. It is altering or rising. If the datasets are often being updated meaning that additional

datasets are being augmented, this beefs up computational resources and boosts the mentioned complexities about type, size and format [15].

In practice, complexity occurs in data then a development of analytical tools is needful and depending on (a) clustering analysis or (b) classification method. Even though such a tool irons out all data analysis problems then a dataset which appears as follows arises. Note that it is not estimated by a straight line nor easily segmentized into clusters. It is complex per se as it demonstrates spherical, recurring or loopy structure. Figure 1 shows examples of complex data traditional techniques cannot classify all characteristics.

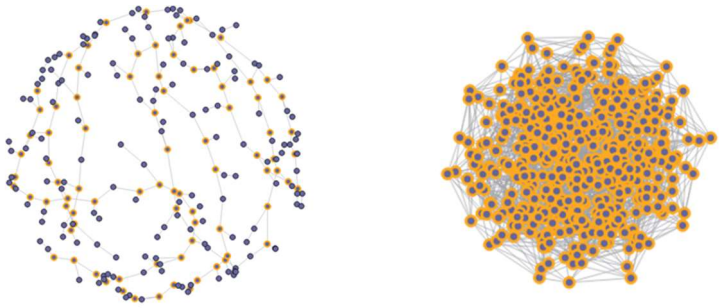

Figure 1. Example structures of complex data

## 3. PREPROCESSING METHODS

In this section, preprocessing approaches are described. Our proposed method which is applicable for complex data, classification algorithms and the comprehensive discussion are given.

### 3.1. Bayesian classification

One of the classical predictions is called Bayesian with a simple hypothesis in which all input parameters are assumed to be autonomous [16]. This classification is recognized as a minimum computational cost as well as incomplexity. Let there be m different classes (C1, C2, C3,…, Cm) and the trained Bayesian classifier expects X which belongs to class Ci with high accuracy. The classification model performs as follows: Let each tuple be an n dimensional attribute vector of X (x1, x2 , x3,. . . , xn) be n finite attributes, and suppose xi can take different Ci values, namely, P(Ci/X) > P(Cj/X) for $1 \leq j \leq m$ and $j \neq i$. The Bayesian classifier calculates a probability of Ci as following P(Ci/X) = P(X/Ci) P(Ci) / P(X). The values P(X) and P(X/Ci) are approximated from the training dataset (a dimensional table with tuple). The algorithm obviously accumulates the counts due to taking a new batch of examples. The algorithm of Bayesian classification is described as shown in Figure 2.

**Algorithm Bayesian**

**Require**:Dataset matrix which contains *a* rows (instances) and *b* columns (attributes)
**Ensure**:$[A]_{a \times b}$
**for** *i*= 1 to *a* do
  **for** *j* = 1 to *b* do
  Build a frequency table for all the features against C*i*
  Construct the likelihood table for the features against C*i*
  Compute the conditional probabilities for C*i*
  Compute the maximum probability for C*i*
  **end for**
**end for**

Figure. 2. Bayesian algorithm

### 3.2. Boosting classification

Boosting denotes an algorithm which renovates fragile learners to tough learners. The weighting parameter decomposes the matrix A into two parts equally. First half of the weight (tough) is allocated to the perfect classification part, and the second half is assigned to the misclassified (fragile) part. Poisson distribution for computing the random probability to train the model is employed. The key concept of boosting is to accept a sequence of fragile learners. Weighted parameter is applied to model which was wrongly classified in the previous iteration. Only this time being the weighting parameter alters regarding to

the boosting weight as proceeding through each round of computation in order. The estimation keeps calculating through a weighted sum (regression) or weighted majority (classification) to result the final iteration. The following algorithm listed in Figure 3 explains the iteration of boosting [17].

**Algorithm Boosting**

**Require:**Dataset matrix which contains *a* rows (instances) and *b* columns (attributes)
**Ensure:**$[A]_{a \times b} \rightarrow [A_1]$ and $[A_2]$, N = dimension of [A]
**Set:**Initial weight parameter is $w_n$ (=1/N)
**for** $i = 1$ to a do
  **for** $j = 1$ to b do
  **for** $k = 1$ to $K$ do
  Accept $C_k(x)$ after minimizing error of weight parameter $E_k$
  Compute $E_k = \sum_{n=1}^{N} w_n^{(k)} 1[y_k(x_n) \neq v_n]$
  Compute $\alpha_k = \sum_{n=1}^{N} w_n^{(k)} 1[y_k(x_n) \neq v_n] / \sum_{n=1}^{N} w_n^{(k)}$
  Compute $\beta_k = \frac{1}{2} \ln\left(\frac{1-\alpha_k}{\alpha_k}\right)$
  Randomize through Poisson distribution to update the weight parameter
  $w_n^{(k+1)} = w_n^{(k)} \exp\{\beta_k 1[y_k(x_n) \neq v_n]\}$
  **end for**
  Estimate using final result $Y_K(x) = \text{sgn} \sum_{k=1}^{K} \beta_k y_k(x) \in \{-1, 0, 1\}$
  **end for**
**end for**

Figure 3. Boosting algorithm

### 3.3. Nearest neighboring classification

Nearest with k neighbors (k-NN) used in classification has multiple functions which differs from other algorithms as described above. It is non-parametric which requires no hypotheses about the probability density function of the inputs. In case of unknown input distribution, k-NN is healthier than other parametric algorithms. However, parametric algorithms seem to generate few errors due to considering input probability function. This k-NN is a lazy machine learning algorithm, which analyzes data during the testing phase, rather than in the training period. An advantage of lazy k-NN is that it rapidly adjusts to any changes as it does not take a common dataset from the beginning. But a major disadvantage is the huge computational cost occurs during testing period. In k-NN classification, an input is classified by its majority of the k neighbors. The algorithm is presented in [18].

### 3.4. Proposed classification

The proposed method is a logistic regression based learner which incorporates classifications in order to maximize the probability of monitored values. At base level of calculation, there are diverse learning algorithms that are trained individually based upon a perfect training set. This is unlike other algorithms that opt the sample values that minimize the sum of squared errors. The proposed method involves the combination of preprocessing techniques for a post-processing of the output at deep learning level. Note that the original learners are not customized while the proposed mechanism aims at obtainable higher accuracy in classification and higher performance on complex datasets. The proposed model is trained on the meta-outputs from base level of calculation. The algorithm is depicted in Figure 4.

**Proposed Algorithm**

**Require**:Dataset matrix which contains *a* rows (instances) and *b* columns (attributes)
**Ensure**:$[A]_{a \times b}$, M classifiers, N = dimension of [*A*]
**for** $i = 1$ to *a* do
**for** $j = 1$ to *b* do
  **for** $k = 1$ to $P$ do /** Base level calculation **/
  Learner $M_k$ with dataset *A*
  **end for**
  **for** $q = 1$ to $N$ do /** Maximize probability based on regression **/
  $Am = \{a'_q, b_q\}$, where $a'_q = m_0 + m_1 a_q + m_2 a_q + \ldots + m_P a_q$
  **end for**
  Apply learner M with *Am* /** Deep level calculation **/
  Restore *M*
**end for**
**end for**

Figure. 4. Proposed algorithm

### 3.5. Granularity and performance

In a preprocessing approach, the number of classes observed for the process designates a diverse distribution of the dataset. As far as the performance is concerned, it implies the dispersion of the original dataset among the classifiers. Granularity is used to measure the level of hierarchy (in decision tree), the relative size, the detailed level, depth of penetration and scale in a dataset. Regarding to this, the performance for any classifications differs based on the number of selected classes. One reason is that the capability of learning algorithms creates fewer rational to data shortage. However, higher granularity develops the structure of a healthy model, regarding to the detail of the state space. In this research, the following focuses are fulfilled. Firstly, the dependency of the granularity level in complex datasets is investigated. The classifiers in an experimental learner with complex datasets are chosen. Secondly, these training results list the benefit of a higher granularity for all datasets. Lastly, the robust model in terms of the data granularity is further analyzed by high processing power in order to examine a speedup performance and the efficiency. The following metrics are concerned to evaluate the performance of the proposed technique. The accuracy means the number of acceptable classifications according to the total number of instances. The processing time consumed by individual classifier is quantified for the efficiency comparison. The speed-up reflects the performance of a parallel processing system in comparison with a slower version. The speed-up can be computed by sequential time over parallel reference time.

## 4. ESTIMATION METHOD

The open-source based simulation tool called MOA is employed for the analytics. Four complex datasets have been selected and the granularity analysis of preprocessing methods has been accumulated. The execution has been run on a Fujitsu Windows 8 with Intel® Core ™ i5 CPU, 2.67 GHz Processor and 8 GB RAM on board. The datasets have been selected in order that they are different in number of attributes, instances, details and size. Datasets 1, 2, 3, and 4 are run on a single server (M/M/1), and each dataset is divided into 4 subtasks to be independently processed on four parallel processors (M/M/4). The parallel processing is inclusive of splitting time and re-assembling time. Splitting is based upon software developed by [19] and the simulation model is shown in Figure 5. Performance evaluation of parallel processing for reducing of problem complexity and time is also presented in [20]. The simulation results run on one and four processing units are depicted in Table 10.

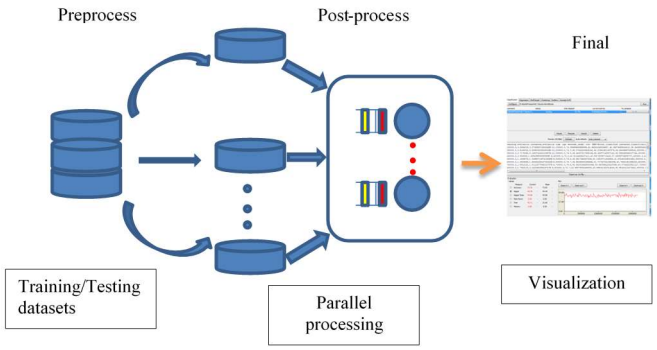


Figure 5. Simulation model

## 5. RESULTS AND ANALYSIS

Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and simulation runtime for four datasets are tabularized in Table 1. Granularity and completeness of these four datasets can be found as shown in Table 2-5. It is obviously seen that dataset 2-4 are complete datasets while only dataset 1 is containing high percentage of missing and considerable as incomplete dataset.

Performance of preprocessing methods described in section 3 lists out all metrics, such as Area Under the Receiver Operating Characteristic curve (AUROC), Classification Accuracy (CA) and precision. Preprocessing performance evaluations for each dataset are shown in Table 6-9. In all cases proposed method outperforms marginally. Then the proposed preprocessing time in msec is taken into account in order to compute for the parallel processing (post-processing) in the simulation model as shown in Figure 5. In order to compare to other research, the Naïve Bayes (NB) in Spark pre-processing mechanism is considered. Note

that NB-Spark results only AUROC as depicted in Table 10. The speed-up metric for these four datasets is calculated from simulation result as displayed in Table 11. In case of dataset #3 and #4, preprocessing time improves speed-up as it differs significantly from post-processing time.

Table 1. MOA Simulation Results

| | Dataset | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| MAE | 0.1 | 57.4 | 0.09 | 0.03 |
| RMSE | 0.23 | 67.6 | 0.14 | 0.06 |
| RunTime(msec) | 80 | 270 | 450 | 180 |

Table 2. Granularity of Dataset #1

| | Dataset #1 | |
|---|---|---|
| | Attr 1 | Attr 2 |
| Skewness | 14.57 | 7.5 |
| Kurtosis | 213.8 | 88.68 |
| Dispersion | high | 751,271 |
| Missing (%) | 61 | 0.5 |

Table 3. Granularity of Dataset #2

| | Dataset #2 | | |
|---|---|---|---|
| | Attr 1 | Attr 2 | Attr 3 |
| Skewness | 14.57 | -2.69 | 0.97 |
| Kurtosis | 213.8 | 7.58 | 1.88 |
| Dispersion | high | 378.1 | 4,606.08 |
| Missing (%) | 0 | 0 | 0 |

Table 4. Granularity of Dataset #3

| | Dataset #3 | | | | | | |
|---|---|---|---|---|---|---|---|
| | Attr 1 | Attr 2 | Attr 3 | Attr 4 | Attr 5 | Attr 6 | Attr 7 |
| Skewness | -0.5 | -0.59 | -0.01 | 0.38 | 1.44 | 0 | 2.36 |
| Kurtosis | -1.39 | 1.88 | 0.95 | 0.87 | 4.23 | -0.09 | 7.99 |
| Dispersion | 7,131.6 | 1.35 | 0.69 | high | high | 0.02 | 0.02 |
| Missing (%) | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 5. Granularity of Dataset #4

| | Dataset #4 | | | | |
|---|---|---|---|---|---|
| | Attr 1 | Attr 2 | Attr 3 | Attr 4 | Attr 5 |
| Skewness | 0.2 | 0.3 | 0.42 | -0.23 | 0.045 |
| Kurtosis | -1.11 | -0.86 | -1.58 | -0.89 | -0.6 |
| Dispersion | 2.68 | high | high | 0 | 0 |
| Missing (%) | 0 | 0 | 0 | 0 | 0 |

Table 6. Preprocessing Performance of Dataset #1

| | Runtime (msec) | Dataset #1 | | |
|---|---|---|---|---|
| | | AUROC (%) | CA (%) | Precision (%) |
| Boost | 30 | 94 | 94.6 | 93 |
| NN5 | 197 | 95.6 | 93.3 | 93.4 |
| NN15 | 206 | 96 | 92.4 | 92.4 |
| Bay | 60 | 98.6 | 94 | 90.6 |
| Proposed | 74 | 95.2 | 95.1 | 93.7 |

Table 7. Preprocessing Performance of Dataset #2

| | Runtime (msec) | Dataset #2 | | |
|---|---|---|---|---|
| | | AUROC (%) | CA (%) | Precision (%) |
| Boost | 20 | 91.8 | 92.4 | 88.5 |
| NN5 | 73 | 98 | 95.5 | 93.8 |
| NN15 | 85 | 99 | 94.7 | 94 |
| Bay | 10 | 99.3 | 94.2 | 94.2 |
| Proposed | 46 | 97 | 96 | 94.7 |

Table 8. Preprocessing Performance of Dataset #3

| | Runtime (msec) | Dataset #3 | | |
|---|---|---|---|---|
| | | AUROC (%) | CA (%) | Precision (%) |
| Boost | 10 | 83.9 | 75 | 84.5 |
| NN5 | 6.7 | 88.2 | 74.7 | 77.2 |
| NN15 | 6.9 | 90.5 | 77.3 | 76.7 |
| Bay | 0 | 98.7 | 89 | 90 |
| Proposed | 8.9 | 94.2 | 90 | 94.2 |

Table 9. Preprocessing Performance of Dataset #4

| | Runtime (msec) | Dataset #4 | | |
|---|---|---|---|---|
| | | AUROC (%) | CA (%) | Precision (%) |
| Boost | 80 | 81.8 | 92.4 | 87.5 |
| NN5 | 60 | 98 | 95.5 | 93.8 |
| NN15 | 80 | 79.5 | 76.7 | 44.4 |
| Bay | 10 | 95.3 | 94.4 | 92.2 |
| Proposed | 80 | 97 | 96 | 94.7 |

Table 10. Preprocessing Performance Comparison

| | Runtime (msec) | Dataset #4 | | |
|---|---|---|---|---|
| | | AUROC (%) | CA (%) | Precision (%) |
| Boost | 80 | 81.8 | 92.4 | 87.5 |
| NN5 | 60 | 98 | 95.5 | 93.8 |
| NN15 | 80 | 79.5 | 76.7 | 44.4 |
| Bay | 10 | 95.3 | 94.4 | 92.2 |
| NB Spark | N/A | 71 | N/A | N/A |
| Proposed | 80 | 97 | 96 | 94.7 |

Table 11. Results Comparison for One and Four Processing Units (Pre:Post)

| Residual Time (msec) | Dataset | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| M/M/1 | 74:109 | 46:60 | 8.9:286 | 80:1591 |
| M/M/4 | 74:42.7 | 46:18 | 8.9:109 | 80:459 |
| Speed-up | 1.56 | 1.65 | 2.5 | 3.1 |

## 6. CONCLUSION

In parallel processing system, several processing units are connected in parallel fashion with each other and this combined structure is filled with a complex dataset. Since the complexity of dataset exists, preprocessing techniques are compulsory. The proposed algorithm for preprocessing is introduced and outperforms for both CA and precision analysis compared to other existing methods. The proposed classification method outperforms and improves granularity level of complex datasets. In the end, parallel processing is employed to measure the post-processing time and speed-up metrics. It is clear that Dataset complexity and pre-processing time reflect the effectiveness of each algorithm. Speedup is based on the runtime of MOA simulation. The future research considers the approximation technique in order to lessen the processing time complexity issued by simulation. The next publication touches a concept of optimizing both CA and precision in preprocesses.

## REFERENCES

[1] C. C. Aggarwal, "Data Mining: The Textbook,"Berlin, Germany: Springer, 2015.
[2] D. Laney, "3D Data Management: Controlling Data Volume, Velocity and Variety," Retrieved January 12, 2018, from https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf, 2001.
[3] X. Wu, X. Zhu, G. Q. Wu and W. Ding, "Data Mining with Big Data," *IEEE Trans Knowledge Data Engineering*, vol. 26, no.1, pp. 97–107, 2014.
[4] N. M. Nawi, W. H. Atomi and M. Z.Rehman, "The Effect of Data Pre-Processing on Optimized Training of Artificial Neural Networks," *Procedia Technology*, vol. 11, pp. 32-39, 2013.
[5] C.–F. Tsai, W.–C. Lin and S.–W. Ke, "Big Data Mining with Parallel Computing: A Comparison of Distributed and MapReduce Methodologies," *Journal of Systems and Software*, pp. 83–92, 2016.
[6] S. García, S. Ramírez-Gallego, J. Luengo, et al. "Big Data Analytics 1:9," Retrieved February 15, 2018, from https://doi.org/10.1186/s41044-016-0014-0, 2016.
[7] L. Brisson and M. Collard, "How to Semantically Enhance a Data Mining Process," *Enterprise Information Systems, Springer Berlin Heidelberg*, vol. 19, pp. 103–116, 2010.
[8] C. Jittawiriyanukoon and V. Srisarkun, "An Approximation Method of Regression Analysis in Concurrent Big Data Stream," *International Journal of Technology (IJTech),* vol. 9, no. 1, pp. 192-200, 2018.
[9] Bifet, R. Kirkby, G. Holmes and B. Pfahringer, "MOA: Massive Online Analysis," *Journal of Machine Learning Research 11*, pp. 1601–1604, 2010.
[10] W. Liu, L. Ma, B. Qiu, M. Cui and J. Ding, "An Efficient Depth Map Preprocessing Method Based on Structure-Aided Domain Transform Smoothing for 3D View Generation," *PLoS ONE*, vol. 12, no. 4, pp. 1-20, 2017.
[11] G. Madhu and G. Nagachandrika, "A New Paradigm for Development of Data Imputation Approach for Missing Value Estimation," *International Journal of Electrical and Computer Engineering*, vol. 6, pp. 3222-3228, 2016.

[12] G. Jakobson, G. Piatetsky-Shapiro, C. Lafond, M. Rajinikanth and J. Hernandez, "CALIDA: A System for Integrated Retrieval from Multiple Heterogeneous Databases," *Proceedings of the Third International Conference on Data and Knowledge*, pp. 1-18, 2014.
[13] M. Saqib, M. Arshad, M. Ali, N. U. Rehman and Z. Ullah, "Improve Data Warehouse Performance by Preprocessingand Avoidance of Complex Resource Intensive Calculations," *International Journal of Computer Science (IJCSI),* vol. 9, no. 2, pp. 202-206, 2012.
[14] S. Christa, V. Suma and L. Maduri, "An Effective Data Preprocessing Technique forImproved Data Management in a DistributedEnvironment," *International Journal of Computer Applications on Advanced Computing and Communication Technologies for HPC Applications*, pp. 25-29, 2012.
[15] S. Río, J. Benítez and F. Herrera, "Analysis of Data Preprocessing Increasing the Oversampling Ratio for Extremely Imbalanced Big Data Classification," *Proceedings of IEEE Trustcom/BigDataSE/ISPA Conference 2015*, pp. 180-185, 2015.
[16] H. Y. Mussa, J. B. Mitchell and R. C. Glen, "Full Laplacianised Posterior Naive Bayesian Algorithm," *Journal of Cheminformatics,* pp. 1-6, 2013.
[17] W. Hu, W. Hu and S. Maybank, "AdaBoost-Based Algorithm for NetworkIntrusion Detection," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 38, no. 2, pp. 577-582, 2008.
[18] R. Thirumahal and Patil A. Deepali, "KNN and ARL Based Imputation to Estimate Missing Values," *Indonesian Journal of Electrical Engineering and Informatics,* vol. 2, pp. 119-124, 2014.
[19] G.D.G. Software SARL, Retrieved July 14, 2017, from http://www.gdgsoft.com, 2016.
[20] S. Krishnamurthy and R. Tzoneva, "Decomposition-Coordinating Method for Parallel Solution of a Multi Area Combined Economic Emission Dispatch Problem," *International Journal of Electrical and Computer Engineering*, vol. 6, no. 5, pp. 2048-2063, 2016.