Comparative Analysis on Prediction of Software Effort Estimation Using Machine Learning Techniques

Prof. A. J. Singh, Mukesh Kumar

Department of Computer Science, Himachal Pradesh University, Summer-Hill, Shimla. Himachal Pradesh
Email: mukesh.kumarphd2014@gmail.com

Abstract: Effort Estimation (EE) is a technique for finding the entire effort required to predict the accuracy of a model. It's a significant chore in software application development practice. To find accurate estimation, numerous predictive models have developed in recent times. The estimate prepared during the early stage of a model expansion is inaccurate since requirements at that time are not very clear, but as the model progresses, the accuracy of the estimation increases. Therefore, accurate estimation is essential to choose for each software application model development. Here, Linear Regression (LR), Multi-layer perceptron (MLP), Random Forest (RF) algorithms are implemented using WEKA toolkit, and results shows that Linear Regression shows better estimation accuracy than Multilayer Perceptron and Random Forest.

Keywords: Effort Estimation, Machine Learning Techniques, Prediction, Classification, Random Forest, Linear Regression, Multi-layer perceptron

1. Introduction

Predicting the number of work units compulsory to perform a particular assignment based on an understanding of similar projects and other project features that are supposed to be associated with the effort. The functions of the software application are the input and the effort we want to predict. The processing is used to predict the number of units of work required to perform a particular task. Based on the knowledge of similar projects and other project features that are believed to be related to the effort. It is essential to organize, superiority and success of any software application development. The commonly used efficient categories of effort estimation are expert estimation, algorithmic estimation, and machine learning. In this contribution, comparisons of different machine learning algorithms have performed and which algorithm is more suitable in which situation have discussed.

Accuracy is an indicator of how closely the reality is. Every time you make an estimate, everybody desires to know how close the number is to reality. We want each forecast to be accurate at the time of creation.

Nomenclature

EE	Effort Estimation
LR	Liner Regression
MLP	Multilayer Perceptron
RF	RandomForest
IG	Information Gain
GR	Gain Ratio
CC	Correlation Coefficient
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
RRAE	Root Relative Absolute Error
RSE	Relative Squared Error

2. Literature Resources

The effort estimation techniques needed to develop a software application for making prediction has surveyed here. The research work completed by the different researcher has given. Wittig, G., Finnie, G. et al., in his study "Estimating software development effort with connectionist models" using Neural Network algorithm to estimate total effort need for software application development. Their implementation result was very encouraging as the Neural Network algorithm show the capacity to assess application development effort within 25% of actual attempt more than 75% of the time for substantial industrial dataset. Boetticher et al., in his study "An assessment of metric contribution in the construction of a neural network-based effort estimator". In his work author used four metrics like object, size, complexity, and vocabulary assessed in terms of their contribution in programming effort estimation. Xishi Huang, Danny Ho, Luiz F, Capretz, Jing Ren, in his study "An Intelligent Approach to Software Cost Prediction" used an intelligent approach to calculate software cost estimation prediction. In this research work, they integrated two different techniques like the neuro and fuzzy technology with the COCOMO model. Their research approach makes the best use of both the historical project implementation dataset and expert knowledge. They validate their model by using the industry project dataset. Srinivasan K., and D. Fisher, in his study "Machine Learning, Approaches to Estimating Software Development Effort". In this research, the work author describes different algorithms of machine learning, which we usually used to build the estimation of software application effort from the historical dataset. In their research work, they implemented different models like a neural network, regression tree algorithm, COCOMO and SLIM. They used the COCOMO dataset, which has collected from 63 various software application projects for training the prediction model and tested against the Kemerer COCOMO dataset. In their implementation, they found that the regression tree algorithm performed exceptionally well as compared to COCOMO and the SLIM model. Shivhare J, in his study "Effectiveness of Feature Selection and Machine Learning Techniques for Software Effort Estimation" implemented Neural Network and classification algorithm for calculating effort estimation for application. In the first phase of research, they explained about different features selection algorithms like Information Gain, Rough Set analysis and Rough-reduct to find out the optimal feature set in the dataset. In the second phase of research, they implemented different ML algorithms like Naive Bayes, CART, Feed Forward Neural Network, Functional Link Neural Network, Radial Bias Neural Network, and Support vector machine. In their result, they said that Feed Forward Neural Network and Naïve Bayes give better results as compare to other implemented algorithms. Saini N, Khalid B in his study "Empirical Evaluation of machine learning techniques for software effort estimation". They implemented decision tables, decision trees, radial bias networks, and MLP. In their research, they found that decision tree performance is better than other algorithms in terms of Mean magnitude relative error. Seref B, Barisci N, in his study "Software Effort Estimation Using Multilayer Perceptron and Adaptive Neuro-Fuzzy Inference System". They implemented the multi-layer perceptron classification and adaptive neuro-fuzzy interference system to find effort estimation. In his research work, they used NASA and Desharnais dataset with different projects. The evaluation criteria for their analysis are mean magnitude relative error (MMRE), percentage relative error (PRE). In their result, they said that the adaptive neuro-fuzzy interference system gave better result as compared to multilayer perceptron algorithm. From reviewing some research papers on effort estimation, we found that in most of the cases researcher goes for the ML algorithms to find the accurate result. Mostly used ML algorithms are Artificial Neural Network, Naive Bayes and Random Forest and Support Vector Machine algorithms on different datasets with some good results.

3. Traditional Techniques for Effort Estimation

There are lots of methods developed in the past to calculate the total effort required to create an application. Some of the formally used estimation methods are functional point analysis, expert judgment method and estimating by analogy. At some point in time, these listed methods are widespread in use. Below is the brief descriptions of these methods are given:

Functional Point Analysis: It was developed by IBM and promoted by the International Function Point Users Group. It provides a method to size the software product functionally. It is used to measure functionality, software development and maintain independently. The function counts are a linear arrangement of five essential components like external I/O, inquiries, interfaces and logic internal files with three complexities like complex, average and dull.

Expert Judgment Method: In this method, experts of effort estimation are involved, and their advice and experience must have taken into consideration. It is the oldest and commonly used methods for an estimate to arrive at some cost estimation conclusion.

Estimating by Analogy: It is effortless and straightforward estimation technique that works similar to the expert judgment method. Because of effort estimation, experts often go for completed projects to make any final decision or opinion.

These techniques discussed very famously, but now a day, lots of Machine learning techniques are used in place of these techniques and gave a tremendous result. Above listed methods require a lot of manual work. In functional point Analysis, the substantive count is done manually, which requires more time, detailed knowledge and expertise in a specific area. At some point in time, there is no similar project found to estimation the total effort.

4. Research Background

In this section, our discussion is mostly concentrated on the Machine Learning approach to approximate the total effort estimation of any software project. Classification The algorithm classifies data according to the number of classes. These algorithms can have applied to the structured or unstructured dataset. The primary purpose behind the classification algorithms is to identify the level to which the new data will fall. Classifications are known as a supervised learning algorithm with a computer program that could be learned using data. The given data set might be bi-class or multi-class too. There are lots of classification algorithms in uses like DT, NB, ANN, RF and SVM. Below is some mostly used Machine Learning algorithms are discussed in more detail for better understanding of these algorithms before implementation:

Artificial Neural Network Algorithm: ANN statistical data are not linear. The multifaceted relationships among I/Ps and O/Ps can have utilized for discovering templates or data sets patterns. As a NN tool, the data warehouse includes data processing procedure is known as data mining.

Naive Bayes Algorithm: The classification algorithm, Naive-Bayesian classification is a Machine Learning method that depends on Bayes' theorem:

$P \{A|B\} = P \{B|A\} P (A)/ P (B)$

As depicted in equation, A and B are two numerous events, P (A) and P (B) being the probabilities for A and B, respectively. P (A|B) is the possibility of A occurring because B has transpired. This equation (1) has utilized for computing the probability of some variable values.

Random Forest Algorithm: It is a simple and easy-to-use algorithm that generates excellent results in most situations. It is a supervised classification algorithm. This algorithm produces a forest with several trees. In general, the more trees are in the forest; the forest will become stronger. In Random classification forest, larger the quantity of trees in the forest than more accurate is the result.

Support Vector Machine Algorithm: It's a supervised ML algorithm for classifying problems and invented by Vapnik. The kernel concept uses the classification limit to find the units below. Based on the boundaries, it differentiates between data. It could have utilized in different categories, in space points, and to map data. Linear SVM have used for solving multiclass classification work. Linear SVM provides excellent precision. SVM is the primary classification method for running the classification tasks in a multi-dimensional space for the construction of a hypermarket that classifies case studies in different classes. It has two types of functions: regression in addition to the classification tasks. Besides, it can manage certain and stable variables. SVM has recently been essential in the specific field of the model, as well as a machine learning class.

Effort Estimation Parameters: The present state of effort estimation has satisfactorily implemented. In the last few years, lots of evaluation and estimation methods have developed for estimation. But most of them lack in sound conceptualization, theoretical bases, statistically significant experimental validation. Most of the estimation metrics developed has developed by individuals and implemented or tested in a minimal environment. In this particular section, we try to briefly explain different estimation metrics like SSE, MSE, RMSE, MMRE, RAE, RRSE and MAE. These are popular and mostly used parameters for effort estimation.

The tool used: In the recent past, many software tools are developed for effort estimation by ML algorithms. These tools are WEKA, MATLAB, Rapid Miner, Orange, KNIME, TANAGRA, XLMiner, Python, Kaggle, Teradata, DataMelt and KEEL. But in our study, we found that only a few of them are used randomly used like WEKA, MATLAB, Rapid Miner and KEEL. Here, we used WEKA for implementing ML algorithms for evaluation of effort estimation. It almost provides support to all types of ML algorithm like clustering, classification, and association rule mining.

Feature Selection Methods: It is a pre-processing technique used in the computer learning process to eliminate irrelevant and excessive attributes to improve the learning accuracy. These options are not just about reducing cardinality, but also an optional attribute that can have based on lack of interaction among attributes and classification. Table 1- lists different feature selection techniques used in Machine Learning algorithms:

Table 1- Different feature selection techniques

Feature Selection	Different Techniques used under which methods
Filter Methods	Information gain (IG), Gain Ratio (GR), Symmetric Uncertainty (SU), Correlation-based Feature Selection (CFS), Markov blanket Filter (MBF), Fast Correlation-based Feature Selection (FCBF), Minimum Redundancy Maximum Relevance (MRMR)
Wrappers Methods	Sequential selection algorithms, Heuristic search algorithms

Data selection for implementation: This dataset is available on PROMISE Software Engineering Repository. Desharnais dataset consists of a total of eighty-one instances and twelve attributes. The dataset has downloaded in tabular form "https://www.kaggle.com/toniesteves/desharnais-dataset". Table 2 - lists different attributes of the selected dataset used for this study:

Table 2- List of attributes be in the dataset with their description

Dataset Attributes	Attributes Description
id	Project ID Number
Project	Project information
TeamExp	Team experience of the project team measured in years
ManagerExp	Project Manager experience measured in years
YearEnd	Project end year
Length	Duration of the project in months
Transactions	Count of total basic logical transactions in the project
Entities	Count of the total number of Entities in the project
PointsNonAdj	Size of the project measured in adjusted function points
Adjustment	Adjustment points in the project model
PointsAjust	Size of the project measured in unadjusted Points
Language	Type of language used in the project expressed as 1, 2 or 3
Effort	Actual Effort of the project has measured in person-hours

5. Implementation of Machine Learning Algorithms

We are using the WEKA tool kit and evaluate selected dataset using 10-fold cross-validation method. Here, we have chosen three mostly used Machine Learning algorithms like Linear Regression, Multilayer Perceptron and Random Forest for implementation using 10-fold cross-validation method. The implementation results of all these algorithms have given in table 3 below:

This experiment involves the Desharnais dataset (with all the 12 attributes) to calculate the performance of ML algorithms. Measurements of 10-fold cross-validation ML performance have given in Table 3. The comparison clearly shows that the Linear Regression algorithm works well because of high correlation coefficients. The values of the CC, MAE, RMSE, RRAE and RSE for linear regression are 0.7519, 2135.2805, 2906.7096, 66.9452%, 65.5568%.

Table 3- Performance Comparisons of ML Algorithms

Evaluation Criterions	LR	MLP	RF
CC	0.7519	0.6351	0.6098
MAE	2135.2805	3020.911	2257.2178
RMSE	2906.7096	4563.0947	3503.0415
RRAE	66.9452 %	94.7114 %	70.7682 %
RSE	65.5568 %	102.9143 %	79.0063 %

Fig.1- shows the power measurements of the Desharnais dataset with ML algorithms. The figure shows that LR algorithm gives improved results.

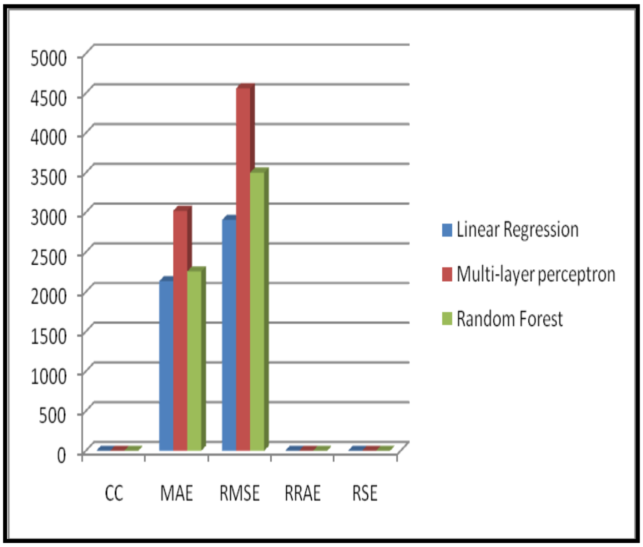

Fig.1- Comparison Graph of above table

The second experiment involves the Desharnais dataset (with only seven selected attributes) to calculate the performance of machine learning algorithms. Measurements of 10-fold cross-validation machine learning performance have given in Table 4. The comparison clearly shows that the Linear Regression algorithm works well because of high correlation coefficients. The values of the CC, MAE, RMSE, RRAE and RSE for linear regression are 0.7673, 2013.7987, 2824.5728, 63.1365 %, 63.7044 %.

In an experiment, supervised attribute filter that can be used to select attributes from the given dataset to improve the overall accuracy of the selected algorithms.

CfsSubsetEval evaluator: Evaluates the value of a subset of attributes taking into account the individual predictive ability of each character and the degree of redundancy between them.

BestFirst search: Searching the subset of attributes in space by a greedy hill-climbing augmented with a function of backtracking.

Table 4- Performance Comparisons of ML Algorithms after feature selection

Evaluation Criterions	LR	MLP	RF
CC	0.7673	0.6843	0.6496
MAE	2013.7987	2742.0907	2148.8052
RMSE	2824.5728	3493.4392	3344.8877
RRAE	63.1365 %	85.9699 %	67.3692 %
RSE	63.7044 %	78.7897 %	75.4394 %

Fig. 2 - shows the power measurements of the Desharnais dataset with ML Algorithms. The figure shows that the Linear Regression algorithm gives better results.

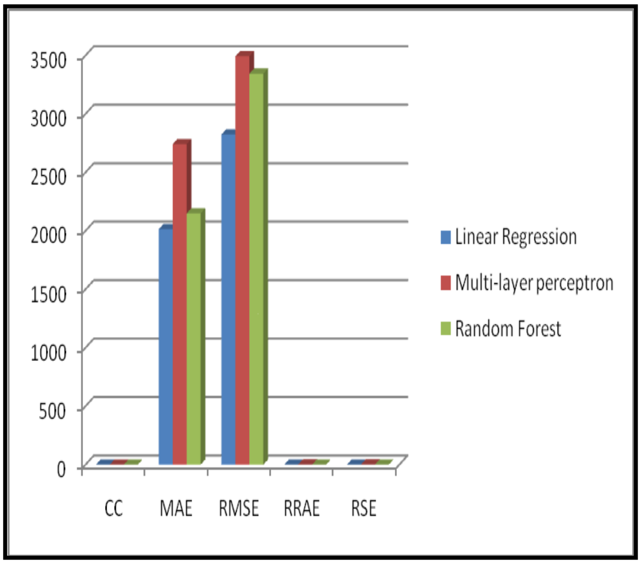

Fig. 2 - Comparison Graph of above table

Comparison of Datasets: WEKA offering an extensive range of ML algorithms. Here, we use WEKA is used to compare the performance metrics of different classifiers such as LR, MLP and RF for both datasets execution and analysis. The comparison determine most excellent effort estimation algorithm for the dataset. The performance measures in Table 5 - show that the LR algorithm gives better results for the Desharnais dataset (With seven selected attributes) compared to Desharnais dataset (With all 12 characteristics).

Table 5 - Comparisons of Machine Learning Techniques

EC	Desharnais dataset (With all 12 attributes)			Desharnais dataset (With seven selected attributes)		
	LR	MP	RF	LR	MP	RF
CC	0.7519	0.6351	0.6098	0.7673	0.6843	0.6496
MAE	2135.28	3020.91	2257.21	2013.79	2742.09	2148.80
RMSE	2906.70	4563.09	3503.04	2824.57	3493.43	3344.88
RRAE	66.94%	94.71%	70.76%	63.13%	85.96%	67.36%
RSE	65.55%	102.91%	79.00%	63.70%	78.78%	75.43%

6. Conclusion and Future Work

A software vendor association strive to estimate the effort required to build software as precisely as possible. Estimating effort required is essential for a successful software development process. Nowadays, development process becomes more multifaceted and significance of research in software development approaches has always increased. So an accurate evaluation is a prime goal for risk-free projects. The results by applying LR, MLP and RF algorithms with the 12 attributes showed that Linear Regression computed superior estimation results than other ML techniques. Using performance matrices such as Correlation Coefficients, MAE, RMSE, RRAE and RSE for Linear Regression is 0.7519, 2135.2805, 2906.7096, 66.9452% and 65.5568%. When these techniques had used on the Desharnais dataset with only seven attributes selected, results showed that Linear Regression allowed a better estimate than Multilayer Perceptron and Random Forest. Using performance measures such as Correlation Coefficients, MAE, RMSE, RRAE and RSE for linear regression is 0.7673, 2013.7987, 2824.5728, 63.137%, 63.7044%. Many applications will have designed with less size or effort, which will reduce the complexity of the software. Future work involves investigating new methods and models for estimating software effort that could help us comprehend software effort estimation process easily. The research work can have continued by choosing a permutation of different ML techniques that gives the enhanced result.

Acknowledgements

I am grateful to Dr. Prasenjit Das, Mr. Sankar Aggarwal, Mr. Manik Gupta for all help and valuable suggestion provided by them during the study.

References

Wittig, G., Finnie, G., (1997). Estimating software development effort with connectionist models. Information and Software Technology, pp. 469- 476.

Boetticher, G., (2001). An Assessment of Metric Contribution in the Construction of a Neural Network-Based Effort Estimator. Second Int. Workshop on Soft Computing Applied to Soft. Engineering.

Xishi Huang, Danny Ho, Luiz F, Capretz, Jing Ren, (2003). An Intelligent Approach to Software Cost Prediction. 18th International Forum on COCOMO and Software Cost Modeling, Los Angeles.

K. Srinivasan, D. Fisher, (2005)Machine Learning Approaches to Estimating Software Development Effort. IEEE Transactions on Software Engineering, vol.21.

Jyoti Shivhare, (2014). Effectiveness of Feature Selection and Machine Learning Techniques for Software Effort Estimation.

Neha Saini, Bushra Khalid, (2015). Empirical Evaluation of machine learning techniques for software effort estimation. Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661

Berna Seref, Necaattin Barisci, (2014). Software Effort Estimation Using Multilayer Perceptron and Adaptive Neuro-Fuzzy Inference System. International Journal of Innovation, Management and Technology, Vol. 5, No. 5.

G. R. Finnie and G. E. Wittig. (1997). A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks Case-Based Reasoning and Regression Models. J.SYSTEMS SOFTWARE, pp.281-289.

Rekha Tripathi, Dr. P. K. Rai, (2016). Comparative Study of Software Cost Estimation Technique. International Journal of Advanced Research in Computer Science and Software Engineering Volume 6, Issue 1.

Nayar, Nandini, Sachin Ahuja, and Shaily Jain. (2019). Swarm intelligence and data mining: a review of literature and applications in healthcare. Proceedings of the Third International Conference on Advanced Informatics for Computing Research.

M. Ruchika and J. Ankita, (2011). Software Effort Prediction using Statistical Machine Learning Methods. International Journal of Advanced Computer Science and Applications, vol. 2, no.1.

Moayed, Ghani, Mojtaba, (2007). Comparing between Web Application Effort Estimation Methods. Fifth International Conference on Computational Science and Applications.

L. Radlinki and W. Hoffmann, (2010). On Predicting Software Development Effort Using Machine Learning Techniques and Local Data. International Journal of Software Engineering and Computing, vol. 2, pp.123-136.

Srinivasan, K., and D. Fisher. (1995). Machine Learning Approaches to Estimating Software Development Effort. IEEE Trans. Software Engineering. , Pp. 126-137.

Girish Chandrashekar, Ferat Sahin, (2014). A survey on feature selection methods. Computers and Electrical Engineering.

R. Malhotra, A. Jain, (2011). Software Effort Prediction using Statistical and Machine Learning Methods. International Journal of Advanced Computer Science and Applications, vol.2, No.1.

Bibi Stamatia and Stamelos Ioannis, (2006). Selecting the Appropriate Machine Learning Techniques for Predicting of Software Development Costs. Artificial Intelligence Applications and Innovations, vol. 204, pp.533-540, 2006.

G. R. Finnie and G.E. Wittig, (1997). A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case Based Reasoning and Regression Models. Journal of Systems and Software, vol.39, pp.281-289.

M. O. Elish, (2009). Improved Estimation of Software Project Effort using Multiple Additive Regression Tree. Expert Systems with Applications, vol.36, pp. 10774-10778.

K. Srinivasan and D. Fisher, (1995). Machine Learning Approaches to Estimating Software Development Effort. IEEE Transactions on Software Engineering, vol.21.

Parag C. Pendharkar, (2010). Probabilistic estimation of software size and effort. An International Journal of Expert Systems with Applications, vol. 37, pp.4435-4440.

G. R. Finnie and G.E. Wittig, (1996). AI Tools for Software Development Effort Estimation," Proceedings of the International Conference on Software Engineering: Education and Practice.

L. Radlinki and W. Hoffmann, (2010). On Predicting Software Development Effort Using Machine Learning Techniques and Local Data. International Journal of Software Engineering and Computing, vol. 2, pp.123-136.

G. Boetticher, T. Menzies and T. Ostrand, (2007). PROMISE Repository of Empirical Software Engineering data http://promisedata.org/repository, West Virginia University, Department of Computer Science.

I. Attarzadeh and Siew Hock Ow, (2009). Software Development Effort Estimation Based on a New Fuzzy Logic Model. International Journal of Computer Theory and Engineering, Vol. 1, No. 4, pp.1793-8201.

C. L. Martin, J. L. Pasquier and Cornelio Y M and Agustin G. T., (2005). Software Development Effort Estimation using Fuzzy Logic: A Case Study. Proceedings of the Sixth Mexican International Conference on Computer Science (ENC'05), IEEE Software.

C. Mair, G.Kadoda, M. Lefley, K.P.C.Schofield, M. Shepperd and Steve Webster, (1999). An Investigation of Machine Learning-Based Prediction Systems. Empirical Software Engineering Research Group, Bournemouth University, U.K.

S. Malathi and Dr S. Sridhar, (2012). Analysis of size matrics and effort performance criterion in software cost estimation" Indian Journal of Computer Science and Engineering (IJCSE) Vol. 3 No. 1.

M. O. Elish, (2009). Improved estimation of software project effort using multiple additive regression trees. Expert Systems with Applications pp. 10774–10778.

Rekha Tripathi, Dr P. K. Rai, (2016). Comparative Study of Software Cost Estimation Technique. International Journal of Advanced Research in Computer Science and Software Engineering Volume 6, Issue 1.

Yogesh Singh, Pradeep Kumar Bhatia and Om Prakash Sangwan, (2015). A Review of Studies on Machine Learning Techniques" International Journal of Computer Science and Security, Volume (1): Issue (1).

Prabhakar and Maitreyee Dutta, (2013). Prediction of Software Effort Using Artificial Neural Network and Support Vector Machine. International Journal of Advanced Research in Computer Science and Software Engineering, pp.40-46.

Nayar, Nandini, Sachin Ahuja, and Shaily Jain. (2019). Swarm Intelligence for Feature Selection: A Review of Literature and Reflection on Future Challenges." Advances in Data and Information Sciences. Springer, Singapore. 211-221.