INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078

(Volume 8, Issue 4 - V8I4-1169)
Available online at: https://www.ijariit.com

Software Defect Prediction by Data Science and Machine Learning Approaches

Vishal Thakur
vishalthakur728@gmail.com
University Institute of Information Technology, Himachal Pradesh University, Shimla, Himachal Pradesh

Ashok Kumar Kashyap
ashokuiit2972@gmail.com
International Centre for Distance Edu. and Open Learning, Himachal Pradesh University, Shimla, Himachal Pradesh

Aditi Badhan
aditibadhan@gmail.com
Himachal Pradesh University, Regional Center, Dharamshala, Himachal Pradesh

ABSTRACT

The quality, dependability, and cost of maintenance are all significantly impacted by the existence of software flaws. Bug-free software, especially software that has been thoroughly built, is difficult to obtain because of the many hidden problems that have been found [7, 9]. A key issue for software technology is the creation of a model based on software bug prediction that can identify faulty components in the early stages of development. As a result, this is neither a linear or constant operation. PSO's formal foundations will be briefly discussed in the section that follows. A linear PSO allows the notion of margin maximization to be explained in an extremely easy manner since the score remains direct and constant number everywhere it works. The pattern of precision in graph analysis differs from that of accuracy. Two datasets, JM1 and KC1, show enhanced precision in the case of accuracy.

Keywords: *Data Science, Datamining, Software Defect, Deep Learning ,Machine Learning*

1. INTRODUCTION

The discipline of anticipating bugs in a software system is an essential one in software development. Since 40-60% of software maintenance expenses are already spent, finding defects in earlier stages of development is critical [4, 6]. Faster software adaption and more efficient use of resources are both aided by early identification of software errors. However, a variety of approaches have been proposed to address the issue of bug prediction. Methods of Machine Learning (ML) are the most widely used. In the concept of bug prediction based on previous faulty data, required metrics, and other software computing approaches, machine learning (ML) techniques are widely used for the prediction of bugs or flaws. Various ML approaches might be used to look at the data from different angles and provide developers with useful insights. Vulnerabilities in software databases may be found by classifying and grouping these approaches. When it comes to machine learning and data mining, a classification is an effective strategy [11, 12]. An early project data-based model of classification is used to classify software modules as either faulty or non-default, based on a variety of complexity criteria. It is a nonhierarchical procedure that moves data between clusters until a desired set of clusters or clusters of data is achieved. Clustering methods forecast how much data will be collected. If this theory is right, it should result in a cluster. But it's a simple task to meet everyone's expectations.

1.1 Need for Bug Prediction

If a software issue produces an irregular feature or functionality that does not meet the criteria, it is a software bug. The presence of a problem in any version of the programme, commercial or noncommercial, is unacceptable. During the design and coding phases, errors are most common. More than $1.7 trillion was lost throughout the world in 2017 because to 606 software defects detected by Tricentis, a tech company, according to the sixth annual Software Fail Watch report [14]. Clearly, a more effective method of forecasting software problems would lower the losses associated with international software manufacturing.

2. RELATED WORK

Aslı Sar et al. [1] conducted a thorough review of the literature on CSE. There were 158 primary studies included in the review, as well as 6 follow-up studies. They also reviewed 67 primary studies that met our quality requirements. They came up with ten

© 2022, www.IJARIIT.com All Rights Reserved

research topics and synthesised several approaches for each of the original studies that went into them. CSE Crowdsourcing (CSE) is a business model, resources, methods, and procedures for software production that are all examined in this study. The digital economy is also examined. Crowdsourcing software for coding and reviewing tasks is being studied by a variety of research teams. Using a unique technique, crowdsourcing focuses more on project planning, task description, and implementation than traditional methods. In CSE, there is a lack of research on how to measure effort and its associated costs. Predicting the mission's outcome depends in large part on the type and expected duration of the operation.

Hyunjoo Kim et al. [2] built an installation cost estimation model based on IFC cost data. An investigation on the expense of fixing walls in office buildings was the focus of this paper. There were two major advantages outlined in the answer. After that, a cost estimate is generated by retrieving and analysing the substitute data from a BIM file and by utilising IFC. Next, with the assistance of CBR, the accuracy is increased by comparing certain cost-related facts, such as contractors and suppliers.

Assia Najm et al. [3] comprehensively map out DT articles according to the following criteria: work methodology, input form, tools utilised in combination with DT methods, as well as identifying platforms and patterns for publication. DT articles. In order to undertake a complete mapping of DT research predominantly focused on SDEE done between 1985 and 2017, an automated search was conducted on five digital repositories. The researchers uncovered 46 studies that they believe are important. Essentially, the data demonstrated that the majority of researchers are dependent on the kind of contribution they make to the approach.

Przemyslaw Pospieszny et al. [4] Achieves a tighter correlation between current research findings and actual operational outcomes by using cutting-edge machine learning delivery and management methodologies that draw on both academic discoveries and the best practises of industry. Smart data planning, an average ensemble of three machine learning algorithms (and cross validation) were used to accomplish this. As a byproduct of this work, firms engaged in the design or integration of information systems will have a decision-making tool.

Ahmed Bani Mustafa et al. [5] Naive Bayes, Logistic Regression and Random Forests are three machine learning approaches that may be used to predict COCOMO NASA pre-processed test data covering 93 projects. In addition to the five folds of cross-validation, the classification accuracy, precision, recall, and AUC of the created models were all evaluated and evaluated. The results of the computation were then compared to COCOMO's results.. It has proven possible to outperform the COCOMO model using any of the strategies that have been tried so far. Nave Bayes and Random Forests, on the other hand, were shown to be the most effective. Nave Bayes surpassed the other two approaches in terms of both the ROC curve and the Recall ranking. The Confusion Index of Random Forests is higher, and the metrics of Identification Accuracy and Precision are higher as well. Results from this study show that data mining in general, as well as machine evaluation methods in particular, is relevant.

Rekha Tripathi et al. [6] compare and contrast classical and machine learning (ML) methodologies. According to the findings, ML techniques provide a more accurate estimation of effort than do traditional methods. Various Machine learning approaches are compared in this article in order to determine whether or not the ML approach is more successful, and under what circumstances.

3. PROBLEM STATEMENT

Software defect prediction results in affecting the quality of the software. This is due to the fact that a defective software module creates a massive impact on the quality of a software module. This in turn increases the cost and time delay of the software, which results in development to a smaller extent. Defect prediction can be a valuable tool for guiding the use of tools for quality assurance. Nevertheless, while a great many research projects covered methods for predicting defects and methodological aspects of prediction research, the real cost - saving potential of predicting defects remains unclear. In this paper, we close the research gap and build a cost model for prediction of software defects. The boundary conditions we derive mathematically are proven to be met by defect model prediction so that a positive benefit is derived when using the defect prediction model. The cost model includes aspects including costs for quality assurance, post-release defect costs, probability of failing to identify expected defects within quality assurance, and the relationship between software artifacts and defects. We initialize the cost model with different assumptions and conduct experiments to explain cost trends in real projects. The broad range of our project includes the use of latest technological fields such as Artificial intelligence, Machine Learning, and feature optimized weight techniques.

The current research project focuses on the following application areas:
Dataset: PROMISE Dataset

AI Intelligence: Searching useful weight features by swarm intelligence

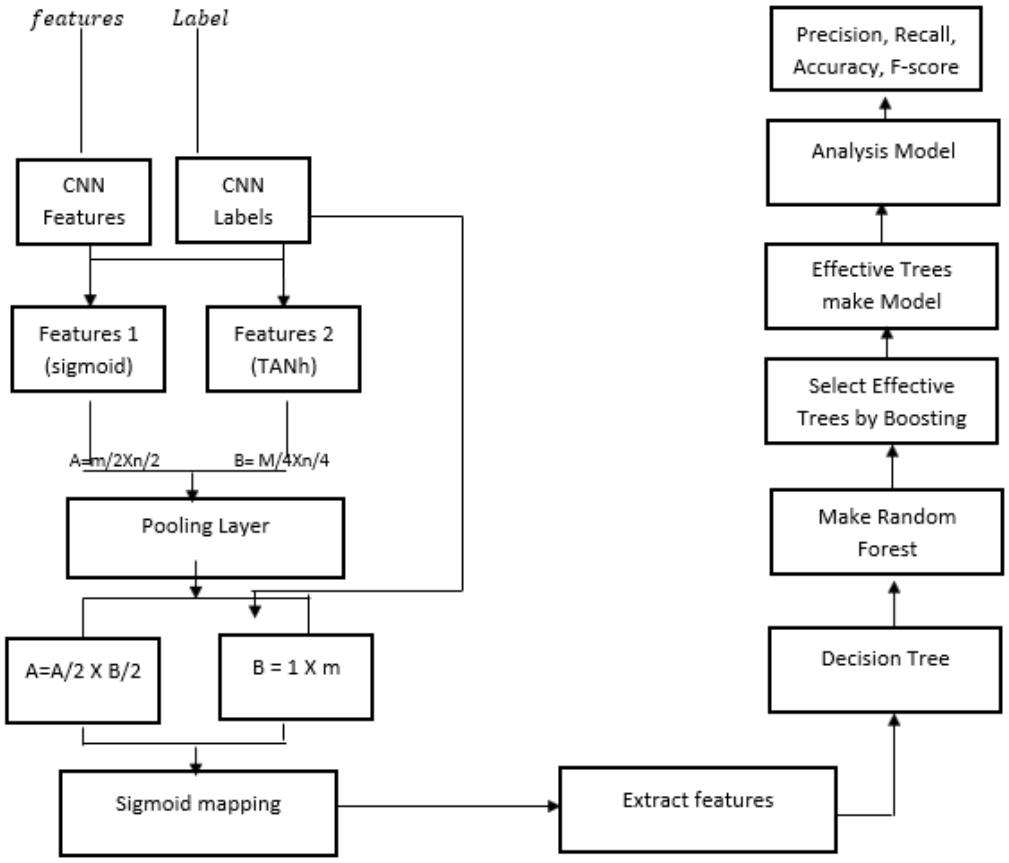
Machine learning: To adapt and detect new patterns

Proposed Framework:

$$m \times n1 \times m$$

$$\begin{bmatrix} f_1 & \cdots & f_n \\ \vdots & \ddots & \vdots \\ f_m & \cdots & f_n \end{bmatrix} \begin{bmatrix} 1 \\ . \\ . \\ m \end{bmatrix}$$

© 2022, www.IJARIIT.com All Rights Reserved



Description:

STEP 1: Features and labels should be extracted from the promise dataset.

STEP 2: For this reason, two activation functions are used; one for convolution layers, and one for sigmoid and TANH activation functions, respectively, to map the features.

STEP 3: After activation, function is mapped by max polling, then merged in matrix A and labelled in Matrix B at last.

STEP 4: Apply the sigmoid function, which gives us the abstract characteristics, after labelling.

STEP 5: Features are learnt using a decision tree that does not create an overlapping forest of features.

STEP 6: Finding relevant trees in the forest using the boosting strategy, then creating the final model and analysing various aspects of it..

Result Analysis:

The proposed system was built on the Python which is a most powerful and open source language. The Screenshots of coding part are as shown in fig 4.4,fig 4.5,fig 4.6, and fig 4.7. Figure 4.4 shows the coding of Convolutional Neural Networks (CNN) with different layers and its activation function and it relates to features of software defect. It also show the impact of layers approach of classification.

Figure 4.4 Software Defect by CNN

© 2022, www.IJARIIT.com All Rights Reserved

Figure 4.5 Software Defect by CNN

In fig 4.5 show the coding Of CNN with different layers and its activation function and it relates classification layer with different optimization.

Figure 4.6 Software Defect by Ensemble learning

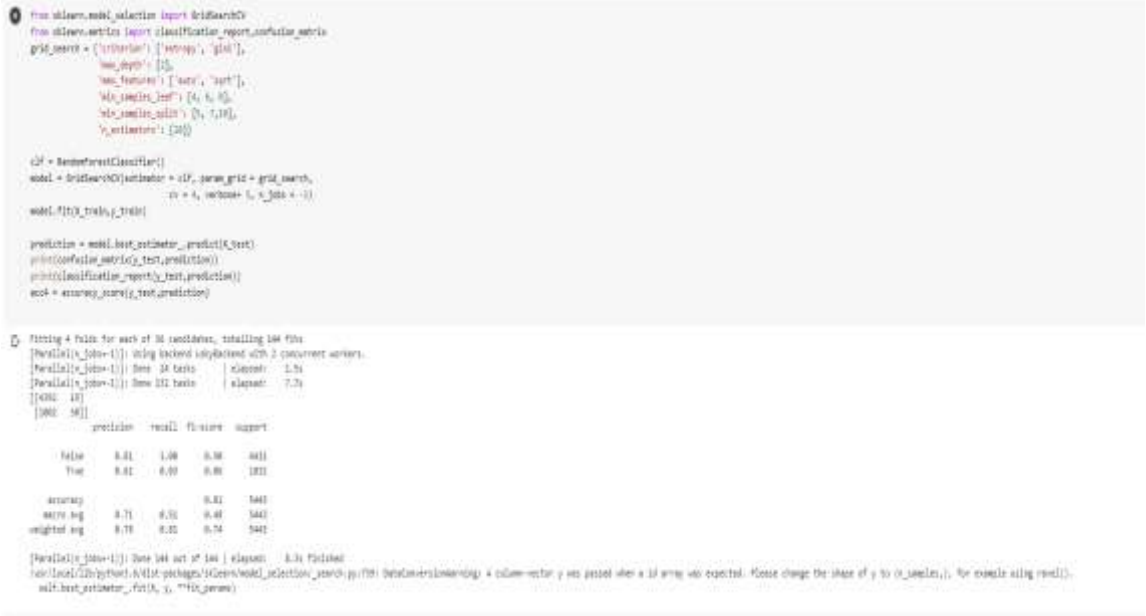


Figure 4.7 Performance parameters

© 2022, www.IJARIIT.com All Rights Reserved

Fig 4.6 show the coding of Ensemble learning using Random forest with help of decision tree and other classifier and fig 4.7 show the coding Of Ensemble learning using Random forest with help of decision tree and other classifier and different performance parameters.

4. ALL CLASS ANALYSIS

This part includes the details of the experiment based on different classifiers as represented below:

Table 4.1 Analysis of the different dataset on the existing and proposed approach

Dataset	Accuracy (CNN)	Accuracy (Ensemble)	Precision (CNN)	Precision (Ensemble)	Recall (CNN)	Recall (Ensemble)
CM1	96.34	94.04333333	96.45	95.28333333	95.52916667	94.27916667
JM1	97.45	93.74666667	98.45	93.91333333	95.89	94.64
KC1	98.34	93.23	97.67	92.34	97.67	93.34
KC2	94.34	90.45	94.56	89.12	96.34	98.56
PC1	98.56	96.56	99.12	95.12	98.99	98.78

Table 4.1 analyses the proposed approach and the existing approach performance on different datasets using comparison metrics. The results of the experiment are shown in table 4.1 and the graphical representation is presented via figure 4.8, 4.9 and 4.10 based on accuracy, precision and recall comparison, respectively. The comparison of the results is shown in two aspects. The first aspect is different dataset and the other aspect is the proposed and the existing approach. In the first comparison, accuracy of different dataset varies from 94% - 98% in the proposed approach and the existing approach varies from 90% - 96%.

In fig 4.8 the analysis of accuracy is done and it shows the accuracy pattern as same in the proposed and existing approach. In figure 4.4, higher accuracy in KC1 is obtained and minimum accuracy is obtained in KC2. On comparing with the proposed and existing approach, CNN based proposed approach improves accuracy significantly. In table 4.1, another parameter is precision ranging from 94% - 99% in the proposed approach and the existing approach ranges from 89% - 95%. Figure 4.8 shows the comparison of precision.

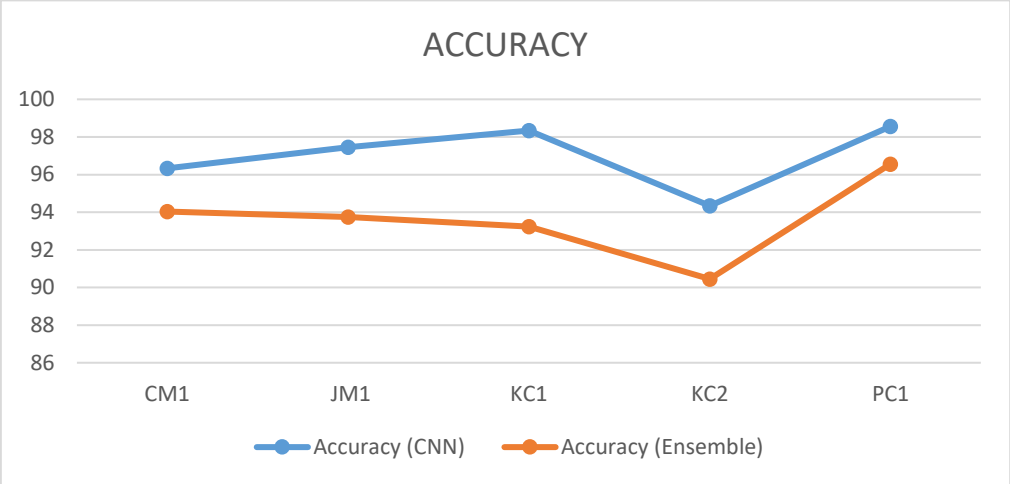


Figure 4.8 Accuracy analysis of the different dataset on the existing and proposed approach

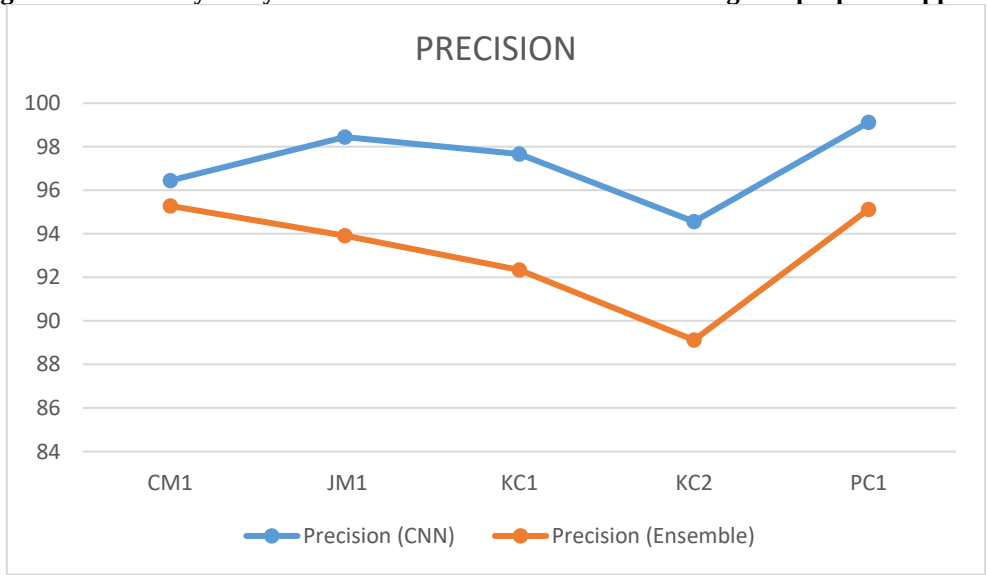


Figure 4.9 Precision analysis of the different dataset on the existing and proposed approach

© 2022, www.IJARIIT.com All Rights Reserved

In figure 4.9, the precision analysis of different PROMISE datasets is done. The graph shows the precision pattern which is not the same as in the case of accuracy. In precision cases, increased precision can be observed in the two datasets i.e. JM1 and KC1. But in this case, KC2 reduces. Whereas, in the case of accuracy, increment & reduction occur only for a single dataset i.e. KC1 and KC2. But in fig 4.9, the precision of JM1 and KC1 is an approximate value. Similarly, precision and its value are significantly high as compared to the existing approach based on PSO. The last section of this chapter analyses the reason for the performance-based increment of the proposed approach. Table 5.1 also analyses the recall parameter which varies from 95% - 98% in the proposed approach and 93% - 98% in the existing approach. In fig 4.10, the analysis of the recall pattern comes after the experiment and also the graphical representation of the proposed and existing approach is done.

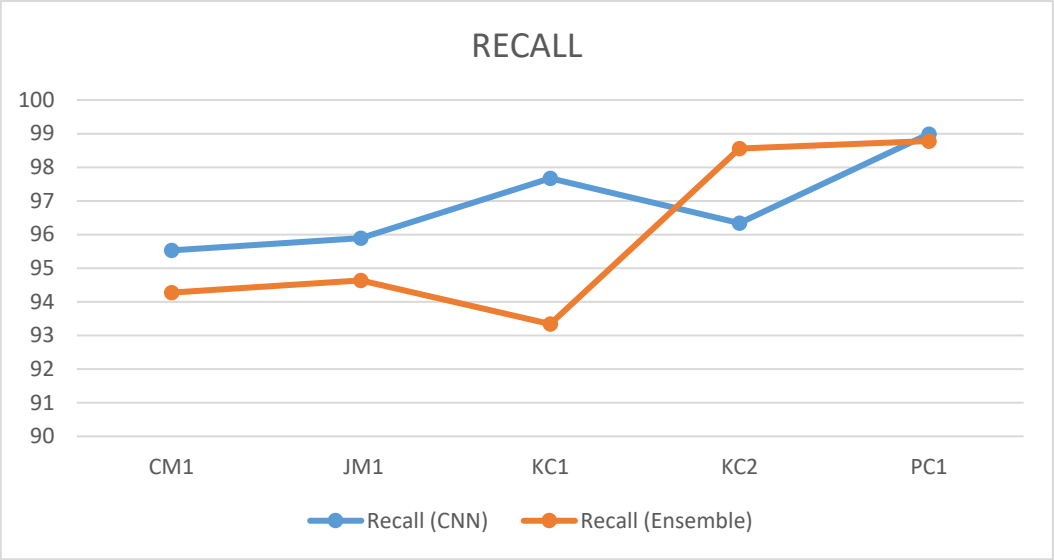


Figure 4.10 Recall analysis of the different dataset on the existing and proposed approach

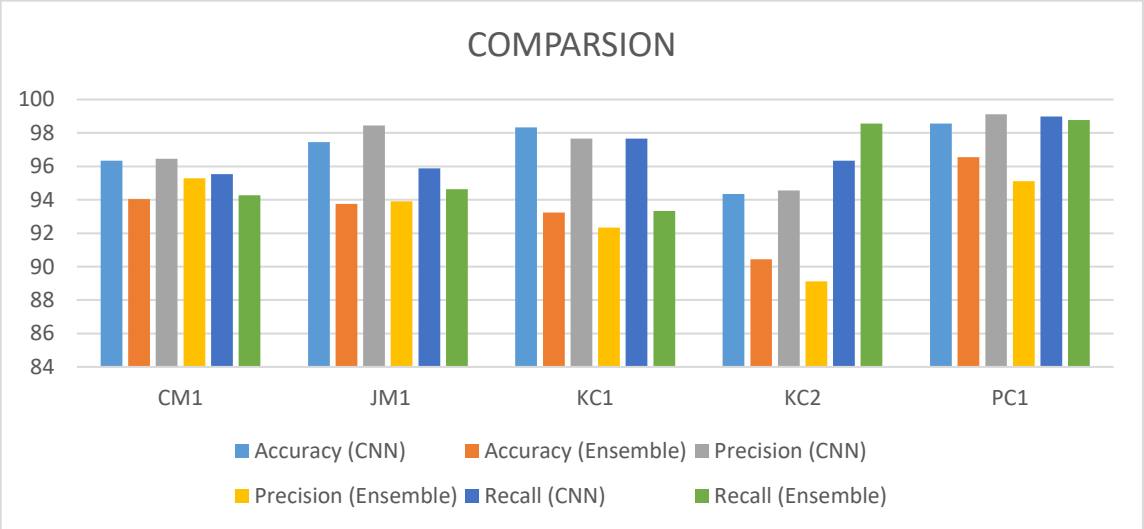


Figure 4.11 Comparative Analysis of different dataset on existing and proposed Approach

Fig 4.10 and fig 4.11 show the recall and comparison analysis respectively. But in fig 4.10, the analysis represents recall not always but shows signs in the proposed approach in case of KC2 dataset.

The average performance of all five datasets recalls improves effectively in the proposed approach as compared to the existing approach.

5. CONCLUSION

As a novel approach for binary classification problems, CNN with Random forest may be a new technique that may prove suitable for non-parametric applied statistics, neural networks and machine learning. PSO uses a score price, which might be an operation of specified money ratios, to classify an organisation as solvent or bankrupt, much as traditional approaches. As a result, this is neither a linear or constant operation. PSO's formal foundations will be briefly discussed in the section that follows. A linear PSO allows the notion of margin maximisation to be explained in an extremely easy manner since the score remains direct and constant number everywhere it works. The pattern of precision in graph analysis differs from that of accuracy. Two datasets, JM1 and KC1, show enhanced precision in the case of accuracy. KC2, on the other hand, exhibits a decrease in accuracy. Furthermore, just one dataset, i.e. KC1 and KC2, has an effect on the accuracy. However, the accuracy of JM1 and KC1 in fig 5.2 is just approximate. When compared to the current technique, the accuracy and value of our approach are substantially higher (PSO). The last portion of this chapter explains why the recommended technique has improved its performance. Table 5.1 also examines the recall parameter, which ranges from 95% to 98% in the proposed technique and 93% to 98% in the current approach.

© 2022, www.IJARIIT.com All Rights Reserved

6. REFERENCES

[1] [1] Sarı Aslı, Ayşe Tosun and GL Alptekin, "A systematic literature review on crowd sourcing in software engineering", Journal of Systems and Software , 153, pp. 200-219, 2019.

[2] [2] kim, Hyunjoo and Jonghyeob Kim, "A Case-Based Reasoning Model for Retrieving Window Replacement Costs through Industry Foundation Class", Applied Sciences, 9(22), pp. 1-15, 2019.

[3] [3] Najm, Assia, Abdelali Zakrani, and Abdelaziz Marzak, "Decision Trees Based Software Development Effort Estimation: A Systematic Mapping Study", International Conference of Computer Science and Renewable Energies (ICCSRE*)*, pp. 1-6, 2019.

[4] [4] Pospieszny, Przemyslaw, Beata Czarnacka-Chrobot and Andrzej Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms", Journal of Systems and Software, 137, pp. 184-196, 2018.

[5] [5] Bani Mustafa, Ahmed, "Predicting software effort estimation using machine learning techniques", 8th International Conference on Computer Science and Information Technology (CSIT), pp. 249-256, 2018.

[6] [6] Tripathi, Rekha, and P. K. Rai, "Machine Learning Methods of Effort Estimation and its Performance Evaluation Criteria", International Journal of Computer Science and Mobile Computing, 6(1), pp. 61-67, 2017.

[7] [7] Bansal, A., B. Kumar, and R. Garg, "Multi-criteria decision-making approach for the selection of software effort estimation model", Management Science Letters, 7(6), pp. 285-296, 2017.

[8] [8] Munialo,Samson Wanjala, and Geoffrey Muchiri Muketha, "A review of agile software effort estimation methods", 2016.

[9] [9] Badampudi, Deepika, ClaesWohlin and Kai Petersen, "Software component decision making: In-house, OSS, COTS or outsourcing-A systematic literature review", Journal of Systems and Software, 121, pp. 105-124, 2016.

[10] [10] Vale, Tassio, Ivica Crnkovic, Eduardo Santana De Almeida, Paulo Anselmo Da Mota Silveira Neto, YguaratãCerqueira Cavalcanti and Silvio Romero de Lemos Meira, "Twenty-eight years of component-based software engineering", Journal of Systems and

[11] Software, 111, pp. 128-148, 2016.

[12] Yang, Ye, and Linda Laird, "Teaching software estimation through LEGOS", IEEE 29th International Conference on Software Engineering Education and Training (CSEET), pp. 56-65, 2016.

[13] Sathya, R. and P. Sudhakar, "Improve Software Quality using Defect Prediction Models", International Journal of Engineering and Management Research (IJEMR), 6(6), pp. 24-29, 2016.

[14] Agrawal Vidisha, and Vishal Shrivastava, "Performance evaluation of software development effort estimation using neuro-fuzzy model", Int. J. Emerg. Res. Manag. Technology, 4, pp. 193-199, 2015.

[15] Fehlmann, Thomas, "4.4 When to Use COSMIC FFP? When to Use IFPUG FPA? A Six Sigma View", COSMIC Function Points: Theory and Advanced Practices, 260, 2016.

[16] Sarro, Federica, Alessio Petrozziello, and Mark Harman, " Multi-objective software effort estimation", In proceedings of IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp. 619-630, 2016.

[17] Ochodek, Mirosław, and BatuhanOzgok, "Functional and Non-functional Size Measurement with IFPUG FPA and SNAP—Case Study", In Software Engineering in Intelligent Systems, Springer, Cham, pp. 19-33, 2015.

[18] Saroha, Meenakshi, and Shashank Sahu, "Tools & methods for software effort estimation using use case points model—A review", In International Conference on Computing, Communication & Automation, IEEE, pp. 874-879, 2015.

[19] Du, Wei Lin, Luiz Fernando Capretz, Ali Bou Nassif, and Danny Ho, "A hybrid intelligent model for software cost estimation", arXiv preprint arXiv:1512.00306, 2015.

[20] Patil, Lalit V., Sagar K. Badjate, and S. D. Joshi, "Develop Efficient Technique of Cost Estimation Model for Software Applications", International Journal of Computer Applications, 87, pp. 1-14, 2014.

[21] Nerkar, L. R., and P. M. Yawalkar, "Software Cost Estimation using Algorithmic Model and Non-Algorithmic Model a Review", International Journal of Computer Applications, 2 pp. 4-7, 2014.

[22] Yiğit, Ferruh, and ÖmerKaanBaykan, "A new feature selection method for text categoriz-

[23] -ation based on information gain and particle swarm optimization", In 2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems, pp. 523-529. 2014.

[24] Madheswaran, M., and D. Sivakumar, "Enhancement of prediction accuracy in COCOMO model for software project using neural network", In Fifth International Conference on Computing,Communications and Networking Technologies (ICCCNT), pp. 1-5, 2014.

© 2022, www.IJARIIT.com All Rights Reserved