Novel Approach of Software Effort Prediction by Random Forest with Boosting and Bagging Approach

Ritu Saini[1], Er. Maninder Kaur[2]

[12]Doaba Group Of College, PTU, Punjab, India

Abstract- In the field of software development, software project estimation is the most challenging task. If there is no proper and reliable estimation provided in the software development, there will be no proper arrangement as well as control of the project. Software project estimation is necessary to handle underestimates and overestimates in terms of cost, effort etc. Small projects are not difficult to estimate and accuracy can be improved by traditional approach of Expert judgment. As the measure of project size increases i.e. for embedded and large-scale projects, precision and accuracy become important concern. In this thesis feature set is increased by adding Line of code (LOC). By using these features effort information is improved and help in software development. In second part features are selected by using Grey wolf optimization (GWO) algorithm. In both works random forest and sampling with boosting and bagging method is used which improves the random forest training model. In this work Random forest with GWO and Random forest without GWO is compared with parameter Accuracy, Precision and Recall.

Keywords- estimation, software, feature selection

I. INTRODUCTION

To overcome the issues of overestimation and underestimation software project estimation approach is used. If the number of resources is more than required resources it enhances the cost of the project and this condition arise the demand of software project estimation.

In small project it is not difficult to estimate the project and mainly estimated by expert judgment approach but in the embedded and large scale projects accuracy and precision of result matters most and they need effective estimation approach. The estimation process with good reliability is an issue that was faced in the projects. In the software estimation process these are the basic steps that are considered:-

- Estimation of project Size: This factor related to the size of th project and measured in the term of function point and line of codes. The UCP (Use case point) and Story points are another method which also helps to estimate the project size.
- Effort estimation: Effort estimation for the project based on the manpower and their working hours in the terms of person per month and person hours.
- Scheduling estimation: To decide the total time for project development.
- Cost estimation to decide the overall budget.

Effort estimation process starts after the estimation of size of the project. This estimation performed after the complete requirements are defined and size mentioned. The software development process includes the design, develop, and testing of modules and each modules required separate effort to complete it. The coding or development part of software development process takes not more effort than other phases. The writing, documentation, implementation of prototype, and review of document takes more effort.

II. RELATED STUDY

Dragicevic et al. [11] proposed the Bayesian method for the effort estimation of software development. This model is simple and small and it can be used from the initial stage of the software development. This model is able to estimate the parameters automatically and learned them from the dataset. The data collected from the single company a precision of the model calculated by using different metrics. The statistical results show good prediction accuracy. Moosavi, et al. [2] presented a model which is a combination of bird optimization algorithm and adaptive neuro-fuzzy inference system. Optimization algorithm used to adjust the variables. This model is based on the optimized ANFIS which produced the effective accuracy to estimate the effort on wide range of projects. The test function in this model includes the unimodal and multimodal function. The results evaluation of the proposed work is based on the three models which improves the performance of the model.

Masoud, Mohammad, et al. [3] proposed the machine learning algorithm for prediction and estimation. This work is based on the expectation maximization soft clustering method and it is a unsupervised algorithm. This model divides the project into four parts. This project helps to develop enterprise and helps in decision making. COCOMO model is used to test and deploy the model and it provides effective results in effort estimation. Araújo, et al. [4] proposed a multilayer hybrid perceptron for software development effort estimation by using the combination of morphological and linear operator. The proposed model trained by using the gradient descent algorithm and performed the experimental analysis using relevant dataset for effort estimation. The result evaluation is based on the MMRE and PRED25 which shows effective prediction.

Kumar chandan et al. [5] worked on the defect estimation in the software development life cycle. This model based on the Bayesian Belief network which predicts the defect of the requirement analysis, development, coding, and testing. The model developed with the help of expert assessment and qualitative value of software metrics. The model was tested on the 10 project by using qualitative data set. The results of the proposed model were more effective than the existing approach. Puspaningrum, et al. [6] presented the harmony search and Cuckoo optimization algorithm for the software cost estimation. These algorithms optimize the result of the COCOMO model on four coefficients. The experiment performed on the NASA dataset and results evaluated by using magnitude of relative error. The results represent the effectiveness by estimating effort and time of development.

Vijay, et al. [7] estimated the effort by using the fuzzy based function point metrics and quality factors. The model developed to resolve the issue of uncertainty in the estimation process and evaluate the accuracy of the software effort estimation. The uncertainty is reduced by using the triangular fuzzy sets and defuzzification by using weighted average approach. The estimated efforts are compared with the existing model by using MMRE and VAF metrics and it gives better results than existing method.Dhaka, V. S., et al. [8] proposed the fuzzy inference system for the effort estimation. This work considered the because the complexity in use cases are high and it takes more time to develop, test and implement. The proposed method provides the reliable results on the use case points and it is produced from actual business process.

Azzeh, et al. [19] proposed model is designed for the classification and prediction stages by using the concept of radial basis neural network and support vector machine. The industrial projects and student projects are used for the construction of observations. This model produced better accuracy from the UCP prediction model. The proposed model gives better accuracy on all datasets by using the environmental factors of UCP to classify and estimate the productivity.Sarro, Federica, et al. [10] introduced the multi-objective effort estimation model which combines the Confidence interval analysis and mean absolute error. The proposed work done by using the PROMISE repository dataset. The statistical analysis of the work shows that this method is significant and gives better accuracy. This model also reduced the uncertainty of the estimation.

III. PROPOSED WORK

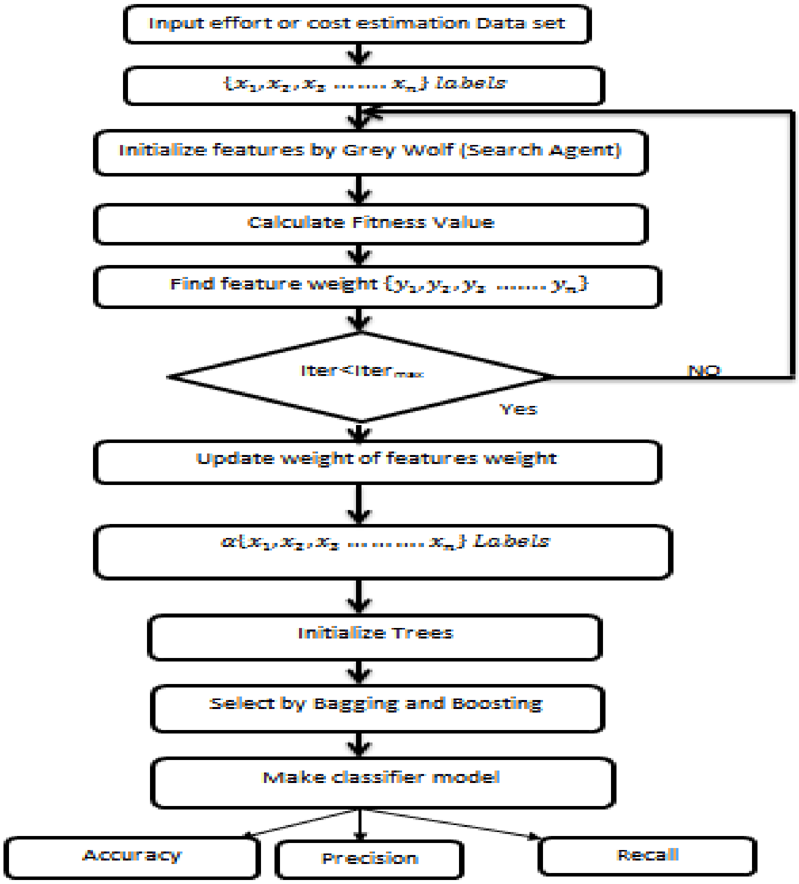
In this section propose methodology of the work is described in detail and flow chart represents the flow of work step by step. In this grey wolf Optimization algorithm used for the optimization and random forest with bagging and boosting for classification.

A) Grey Wolf optimization algorithm is a bio-inspired algorithm which is based on the leadership and hunting behavior of the wolves in the pack. The grey wolves prefer to live in the pack which is a group of approximate 5-12 wolves. In the pack each member has social dominant and consisting according to four different levels.

1. The wolves on the first level are called alpha wolves (α) and they are leaders in the hierarchy. Wolves at this level are the guides to the hunting process in which other wolves seek, follow and hunt and work as a team. Decision making is the main task that is performed by the alpha wolves and the order by the alpha wolves is followed by all members of the pack.
2. Second level wolves are called beta (β). These wolves are called subordinates and advisors of alpha nodes. The beta wolf council helps in decision making. Beta wolves transmit alpha control to the entire packet and transmit the return to alpha.
3. The wolves of the third level are called Delta wolves (δ) and called scouts. Scout wolves at this level are responsible for monitoring boundaries and territory. The sentinel wolves are responsible for protecting the pack and the guards are responsible for the care of the wounded and injured.
4. The last and fourth level of the hierarchy are called Omega (ɷ). They are also called scapegoats and they must submit to all the other dominant wolves. These wolves follow the other three wolves.

B) Random forest is a learning method for classification, regression and generating the multitude of decision trees. It generates the multitude at the time of training and output of the class. It provides the high accuracy and learning is very fast in it. It works very effectively on the large size database. It easily handles the large size input variables without variable deletion.

1. Input the effort or cost estimation Data set.
2. Initialize the features by Grey wolf search agent.
3. Calculate the fitness value.
4. Find the features weight.
5. Check the Iter < Iter Max if yes go to next step otherwise go to step 4.
6. Update the weight of the features.
7. Initialize the tree after labeling.
8. Select by Bagging and Boosting and make the model for the classification.
9. Analysis the accuracy, precision and recall.


Input effort or cost estimation Data set
$\{x_1, x_2, x_3 \ldots\ldots\ x_n\}$ labels
Initialize features by Grey Wolf (Search Agent)
Calculate Fitness Value
Find feature weight $\{y_1, y_2, y_3 \ldots\ldots\ y_n\}$
Iter<Iter$_{max}$
NO
Yes
Update weight of features weight
$\alpha\{x_1, x_2, x_3 \ldots\ldots\ x_n\}$ Labels
Initialize Trees
Select by Bagging and Boosting
Make classifier model
Accuracy
Precision
Recall

IV. RESULTS AND DISCUSSIONS

This section describes the result and discussion in the graphical form. The result of different classifiers used for the comparison and discussed for evaluation. The results evaluation based on the precision, recall and accuracy of the classifiers.

V. RESULTS OF CLASSIFICATION

Table 4.1 Result of Classification

Classification	Accuracy	Precision	Recall
Random forest + Boost	62	52	69
Random forest + Boost+ GWO	71	93	94
Random forest +Bagging+ GWO	69	68	58
Random forest + Bagging	35	92	97

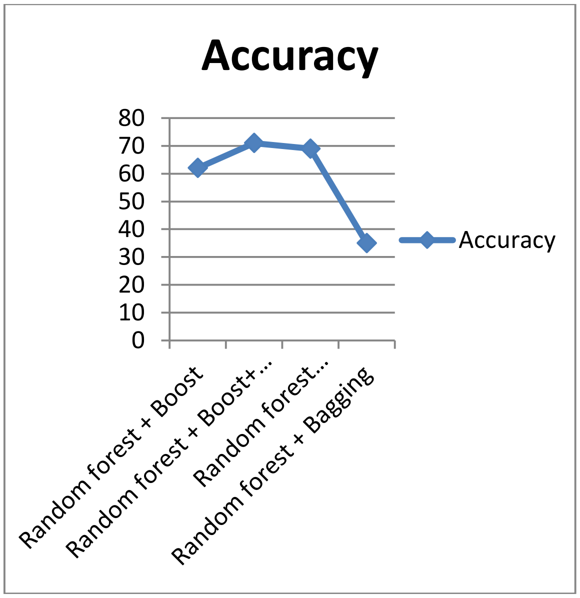


Fig.1: Accuracy of classifiers

Figure 4.1 depicts the accuracy of the Random forest + Boost, Random forest + Boost+ GWO, Random forest +Bagging+ GWO and Random forest+Bagging classifiers. The highest accuracy 93% in graph shown by Random forest + Boost+ GWO and minimum by Random forest + Bagging classifier that is 52%.

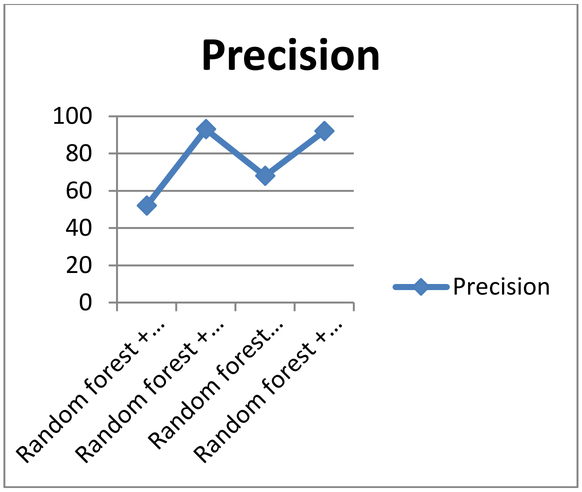


Fig.2: Precision of classifiers

Figure 4.2 depicts the precision of the Random forest + Boost, Random forest + Boost+ GWO, Random forest +Bagging+ GWO and Random forest + Bagging classifiers. The high precision 94 % in graph shown by Random forest + Boost+ GWO, Random forest + Bagging classifier and minimum by Random forest + Boost classifier that is 52%.

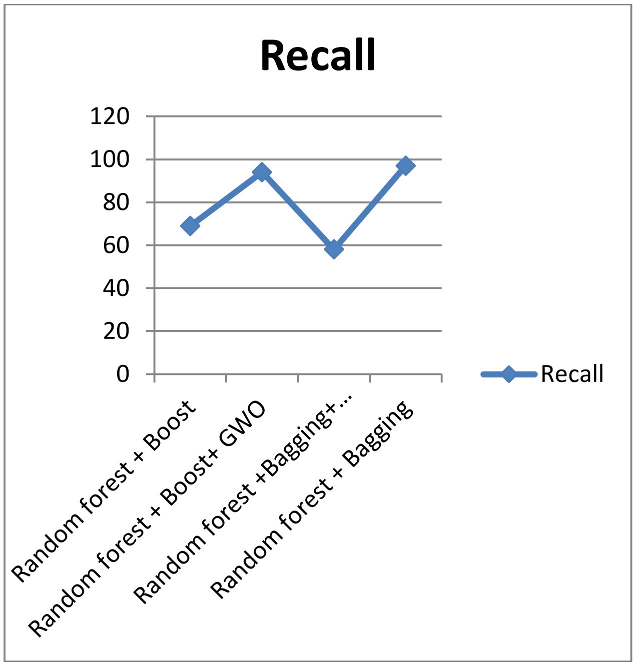


Fig.3: Recall of classifiers

Figure 4.3 depicts the recall of the Random forest + Boost, Random forest + Boost+ GWO, Random forest +Bagging+ GWO and Random forest + Bagging classifiers. The high recall 97 % in graph shown by Random forest + Boost+ GWO, Random forest + Bagging classifier and minimum by Random forest + Bagging+ GWO classifier that is 58%.

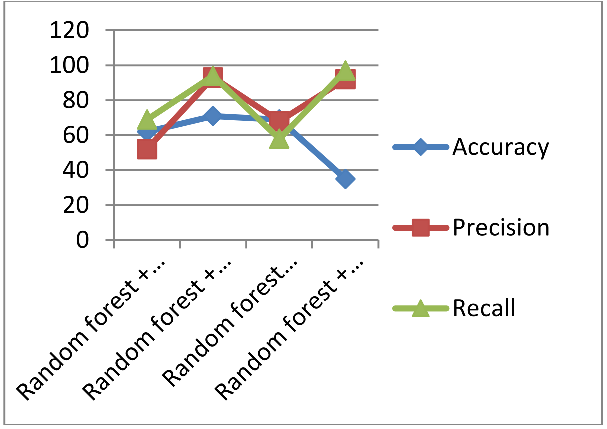


Fig.4: Comparison of classifiers

Figure 4.4 depicts the comparison of the Random forest + Boost, Random forest + Boost+ GWO, Random forest +Bagging+ GWO and Random forest + Bagging classifiers. The effective result shown by Random forest + Boost+ GWO classifier. The red blue curve in the graph represents the accuracy of the different classifiers, Red curve in the graph represents the precision, and green curve represents the recall of the classifier.

Random Forest Regression

Table 4.2 Random Forest Regression

Random Forest Regression	Accuracy
RF+ GWO	79
RF	52.10

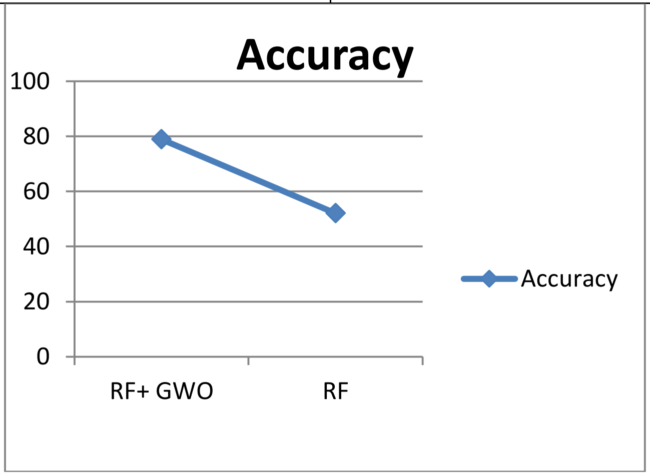


Fig.5: Accuracy of the classifier

In figure 4.5 accuracy comparison is shown with Random forest and Random forest with GWO. The x axis of graph represents the classifiers and y axis of graph represents the random values of accuracy. The accuracy of the Random forest with GWO is better than random forest.

VI. CONCLUSION

Software effort estimation is a challenging issue in the software development process. There are various methods that are proposed by the researchers to solve this issue. In this thesis accuracy of the prediction is improved by feature selection and Machine Learning approach. In this work features selection approach is done by using Grey wolf optimization algorithm. GWO algorithm is used to select the effective weighted feature. The result is shown by the analysis process.

VII. REFERENCES

[1]. Dragicevic, Srdjana, Stipe Celar, and MiliTuric. "Bayesian network model for task effort estimation in agile software development." *Journal of Systems and Software* 127 (2017): 109-119.

[2]. Moosavi, Seyyed Hamid Samareh, and VahidKhatibiBardsiri. "Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation." *Engineering Applications of Artificial Intelligence* 60 (2017): 1-15.

[3]. Masoud, Mohammad, et al. "Software Project Management: Resources Prediction and Estimation Utilizing Unsupervised Machine Learning Algorithm." *International Conference on Engineering, Project, and Product Management*. Springer, Cham, 2017.

[4]. Araújo, Ricardo de A., Adriano LI Oliveira, and Silvio Meira. "A class of hybrid multilayer perceptrons for software development effort estimation problems." *Expert Systems with Applications* 90 (2017): 1-12.

[5]. Kumar, Chandan, and Dilip Kumar Yadav. "Software defects estimation using metrics of early phases of software development life cycle." *International Journal of System Assurance Engineering and Management* 8.4 (2017): 2109-2117.

[6]. Puspaningrum, Alifia, and RiyanartoSarno. "A hybrid cuckoo optimization and harmony search algorithm for software cost estimation." *Procedia Computer Science* 124 (2017): 461-469.

[7]. Vijay, J. Frank. "Enrichment of accurate software effort estimation using fuzzy-based function point analysis in business data analytics." *Neural Computing and Applications* (2018): 1-7.

[8]. Dhaka, V. S., et al. "Software Project Estimation Using Fuzzy Inference System." *Proceedings of International Conference on ICT for Sustainable Development*. Springer, Singapore, 2016.

[9]. Azzeh, Mohammad, and Ali BouNassif. "A hybrid model for estimating software project effort from Use Case Points." *Applied Soft Computing* 49 (2016): 981-989.

[10]. Sarro, Federica, AlessioPetrozziello, and Mark Harman. "Multi-objective software effort estimation." *Software Engineering (ICSE), 2016 IEEE/ACM 38th International Conference on*. IEEE, 2016.