Predicting Effort for a Software Project using Classification Algorithms.

1st Dr.V.Vignaraj Ananth
Assistant Professor,Dept of Computer Science and Engineering, Thiagarajar College of Engineering Madurai, TamilNadu, India

2nd Dr. N.Shiva Kumar
Under graduate Student,Dept of Computer Science and Engineering, Thiagarajar College of Engineering Madurai, TamilNadu, India

3rd Mr.S.Aditya
Under graduate Student,Dept of Computer Science and Engineering, Thiagarajar College of Engineering Madurai, TamilNadu, India

---***---

Abstract— *In software effort estimation lot of algorithmic and prediction methods are available. Though various methods are available, the researches try to get a better effort estimation technique, since it has a vital role in software engineering. Mostly, the industries use the expert's judgements for assessing the effort. The experts may use some methodologies for their references. So, still the algorithmic and prediction methods for software effort estimation needs to be improved. A predictive model is required to be accurate and stable in order to give the confidence in software development. One of the important problems in predictive model is the highly distributed training data set. Because of the data distribution, the performance of the prediction methods is affected. From the literature survey, there are two classification methods were chosen which tackle that problem. First, Extra tree Classifier method that predicts the effort with the help of top k features and Random forest method that predicts effort by classification of random features . In this paper, these two methods are experimented and evaluated using various performance measures. The experimental result shows that Extra Tree Classifier outperforms other methods. It not only improves the accuracy of effort estimation and also it achieves stable results in the software effort estimation.*

Index Terms—Analogy based estimation, Datamining techniques, Data partitioning, Ensemble methods, Software effort estimation, Regression.

1. INTRODUCTION

SOFTWARE development effort estimation is the process of estimating the realistic amount of effort required to develop or maintain software based on incomplete, uncertain and noisy input. Effort expressed in terms of person-hours or person-months. Effort estimation is essential for many people and different departments in an organization. Also, it is needed at various points of a project lifecycle. In order to plan a project and inform the project owners about deadlines and milestones we have to know how much effort the job requires.

Effort estimation accuracy depends on available information. Usually, we have less information before we start the project and we have more information while working on the project. Most of the times, we can have more accurate effort estimation after requirement analysis. However, initial effort estimation at early project stages is sometimes more important. There are a number of methods that are used for effort estimation. All of them have pros and cons.

Expert Judgment: An expert on the subject gives judgment on effort based on his experience on that subject.

Formal estimation model: Using a proper model you feed the system with the proper data to get some estimation.

Combination-based estimation: The estimation arrives with a mixture of both expert estimation and formal estimation procedures.

Two types of formal estimation models are as follows, *Algorithmic models*: Algorithmic models which are made by any of the mathematical models or equations. This type of models are used when the data set is enough to train a model. Ex., Function point model, COCOMO model.

Prediction models: Prediction system models which are useful if the available training data set is not efficient to train an algorithmic model that is when the data is sparse then prediction system model is used. Ex., Analogy based estimation, Machine learning based methods like Neural Network.

For predictive models, data set is important. One of the important problems in predictive model is the highly distributed training data set. Because of the data distribution, the performance of the prediction methods are affected.

Most of the papers gathered the data set from International Software Benchmarking Standard Group(ISBSG) and PRedictOr Models In Software Engineering (PROMISE) repository.

ISBSG is an Australian company that collects the information related to software projects from all over the world. PROMISE and ISBSG contains larger number of different data sets.

The data sets which are frequently used are desharnais, cocomo, telecom, kemerer, albrecht, nasa, maxwell, sdr, finnish, miyazaki and china. Data set contains effort estimation data, such as the project type, team members size, LOC etc.,

Section 1 provides the introduction to software effort estimation. Section 2 defines the Literature Survey of this paper. Section 3 discusses the basic concepts involved in this paper. Section 4 describes the techniques of the model. Section 5 discusses the experiments and results and Section 6 defines the conclusion and future work that can be extended from the prior work.

2. LITERATURE SURVEY

In [2] Vu Nguyen et al. (2018) have used window based COCOMO calibration for estimating the software effort model using historical project data sets. As software development practices change over time, a model based on past data may not make accurate predictions for a new project (Vu Nguyen et al. 2018). They suggested a windowing approach to calibrate COCOMO and assess the performance of the proposed method using windows. Their study provides the way to support the use of small windows of completed projects to calibrate models when COCOMO-like data is available. Moreover, when the change happens in software development over time is rapid, the use of windows is more justifiable for improving estimation accuracy.

Since the use of agile models in software development is increasing, the problem for effort estimation remains a challenge due to the lack of standard performance metrics that are used for predicting the effort. In [8], Srdjana Dragicevic et al. (2017) applied Bayesian network model for effort prediction in agile models. With Simple and small and with the inputs that can be easily gathered, the suggested model has no practical impact on agility. This proposed model can be used during the planning stage as early as possible. The data sets are elicited from completed agile projects from a software company. This proposed model described the various performance statistics used to assess the precision of the model by using Mean Magnitude of Relative Error (MMRE), prediction at level m , accuracy, mean absolute error, root mean squared error, relative absolute error and root relative squared error. The obtained results indicated that the proposed model has very good prediction accuracy compared to other algorithmic models.

In [11] Zeynab Abbasi Khalifelu et al. (2012) proposed data mining techniques to estimate the software costs. The results of each technique are evaluated and analyzed. They used NASA's data set to train and test each of these techniques. The results indicated that data mining techniques improve the estimation accuracy of the models with better accuracy rate.

In [9] Saka Nanda et al. (2016) used a hybrid technique which is a combination of both Particle Swarm Optimization (PSO) and Adaptive Neuro Fuzzy Inference System (PSO-ANFIS) for estimating the effort (Saka Nanda et al. 2016). In this method, they introduced a Neuro-fuzzy optimized model with Particle Swann Optimization (PSO) to estimate effort using NASA dataset. Parameter cost driver which consists of 17 features will be optimized using PSO techniques to improve prediction accuracy rate. The results obtained using optimization technique was used to train the Neurofuzzy model.

The performance of the proposed estimation model was assessed using the performance metrics such as Mean Standard Error (MSE), Mean Magnitude of Relative Error (MMER), and Level Prediction (Pred). This method produces better accuracy rate compared to ANFIS approach. However, it may produce imprecise results due to presence of noise in the data set.

The performance of the proposed estimation model was assessed using the performance metrics such as Mean Standard Error (MSE), Mean Magnitude of Relative Error (MMER), and Level Prediction (Pred). This method produces better accuracy rate compared to ANFIS approach. However, it may produce imprecise results due to presence of noise in the data set.

In [3] Ricardo et al. (2017) propose the dilation-erosion-linear perceptron which is a hybrid morphological neuron to solve some prediction problems. Since solving tasks with complex input-output nonlinear relationships, a particular class of hybrid multilayer perceptrons, called the Multilayer Dilation-Erosion-Linear Perceptron (MDELP) was proposed to deal with software development effort estimation problems. Each processing unit of the proposed model is a composition between a hybrid morphological operator and a linear operator. According to Pessoa and Maragos's ideas, a descending gradient-based learning process was introduced to train the proposed model. An experimental analysis was done using datasets of software effort estimation from software projects and the results obtained are analyzed and compared using the performance criteria MMRE and PRED25 measures.

To overcome the weaknesses such as insufficient data in single estimation techniques for prediction analysis, ensemble methods have been proposed for Software Effort Estimation (SDEE). An Ensemble Effort Estimation (EEE) technique combines several single/classical models. In [12] Ali Idri et al. (2016), proposed techniques based on the six viewpoints that include single models used to construct ensembles, ensemble estimating the effort accuracy, rules used to merge single estimates, accuracy comparison between the techniques and with single models, accuracy comparison between ensemble techniques and methodologies that used to construct ensemble methods. It is found that ensemble techniques are separated into two types namely homogeneous and heterogeneous, and that the machine learning single models are the most commonly engaged in constructing ensemble estimation techniques. They also found that ensemble techniques usually yield acceptable estimation accuracy, and in fact are more accurate than single models.

Software Effort Estimation (SEE) using machine learning (ML) techniques that aimed to improve the estimation accuracy and to implement the process that are used to generate these estimates. Among the ML techniques, decision tree-based models are widely used for most estimation cases. However, a few studies have done to use of Random Forest (RF) in software effort estimation. In [1] Zakrani Abdelali et al. (2019) proposed a RF model and optimized by varying its values by their key parameters. The performance of the RF is compared with a classical Regression Tree (RT). The evaluation was implemented and analyzed through the 30% hold-out validation method using the datasets such as ISBSG R8, COCOMO. To analyze the most accurate techniques, three widely known performance accuracy measures: Pred(0.25), MMRE and MdMRE are used. The result showed that the optimized RF algorithm outperforms the regression trees model based on all evaluation criteria and improves the accuracy rate.

In [13] Sultan Aljahdali et al.(2015) used Regression, Support Vector Machine (SVM) and Artificial Neural Network(ANN) to predict an accurate effort. They proposed two models for software effort estimation and first model represents function points using Linear Regression (LR), Support Vector Machines (SVM) and Artificial Neural Networks (ANN). The proposed models have 'n' number of inputs and a single output. The first model utilizes the Source Line Of Code (KLOC) as inputs while the second model utilize the information domain characters such as the number of Inputs, number of Outputs, External interface Files, Internal logical files and User Inquiries to estimate the Function Point (FP). The proposed SVM and ANN models shows better estimation accuracy compared to linear regression models. Moreover, the results suggest that ANN produces accurate effort estimation in comparison with COCOMO II

In [14] Andres Garcia and Alain Abran(2018) have used Support Vector Regression model(SVR) for dealing with an unstructured data(Andres Garcia& Alain Abran 2018).They analyzed the prediction accuracy of two types of support vector regression (ε-SVR and υ-SVR) methods for software effort estimation. Both types of support vector regression uses linear, polynomial and radial basis function. Prediction accuracies for ε -SVR and υ -SVR are compared with statistical regressions, neural networks, association rules, and decision trees algorithms. These models are trained and tested with five data sets from the International Software Benchmarking Standards Group (ISBSG). Each data set was selected on the basis of data quality, platform and programming language generation. The result suggested that the polynomial kernel ε -SVR (PK ε -SVR) is more accurate when compared with statistical regressions, neural networks, association rules and decision trees.The disadvantage of this model is its high computational time for training the data set.

3. BASIC CONCEPTS

A. Overfitting:

In overfitting, a statistical model describes random error or noise instead of the underlying relationship. Overfitting occurs when a model is excessively complex, such as having too many parameters relative to the number of observations. A model that has been overfit has poor predictive performance, as it overreacts to minor fluctuations in the training data.

B. Tree pruning:

Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances. Pruning reduces the complexity of the final classifier, and hence improves predictive accuracy by the reduction of overfitting.

C. Tree smoothing:

It aims to connect the sharp discontinuities between adjacent linear models at the leaves caused during pruning

4. TECHNIQUES

As mentioned in Section 2. Various techniques can be applied to estimate accurate software effort estimate. As the aim of this study is to assess some of the methods incorporated with regression technique. The following technique is considered:

- Extra Classifier Tree
- Extra Classifier Tree

Extra Classifier Tree Algorithm:

To obtain extra tree classifier technique-based effort estimation model, the steps presented underneath are taken into consideration.

1) The dataset is preprocessed
2) All the available features in the given dataset is collected.
3) To select training set for the tree ,a random sample of n cases, from the original data of all N accessible training cases is chosen. The rest is taken as test dataset.
4) To rank the features for finding top k features of all the available features we use the below formula

   $$Entropy(S) = \sum_{i=1}^{c} -p_i log_2(p_i)$$

   where c is the number of unique class labels

   and p_i is the proportion of rows with output label i.
5) With the help of entropy the gain(information gain)of each feature is calculated.

   Now the k features with highest information gain will be selected as top k features

   $$Gain(S,A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$
6) Now we will find the top k features for recent years of dataset by repeating step 3 and 4 separately for recent years dataset. If this feature is not included in the already derived k features , then this feature id merged along with the previously derived features.Else, this step is skipped.
7) Final effort is predicted.

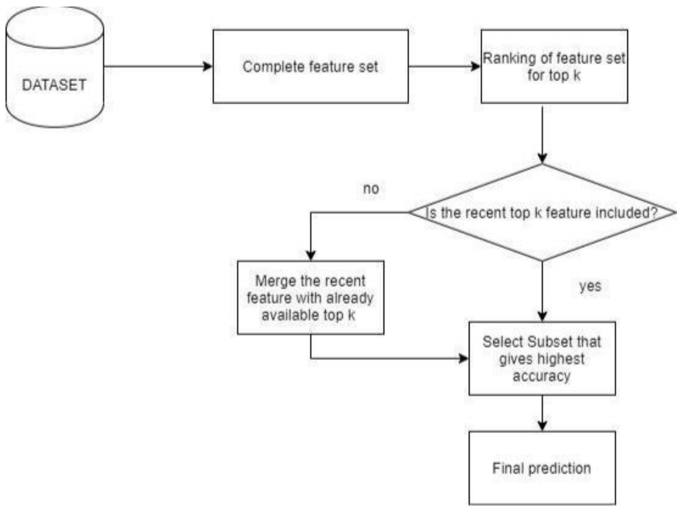

DATASET
Complete feature set
Ranking of feature set for top k
Is the recent top k feature included?
no
yes
Merge the recent feature with already available top k
Select Subset that gives highest accuracy
Final prediction

Figure 3.1.EXTRA TREE CLASSIFIER ALGORITHM

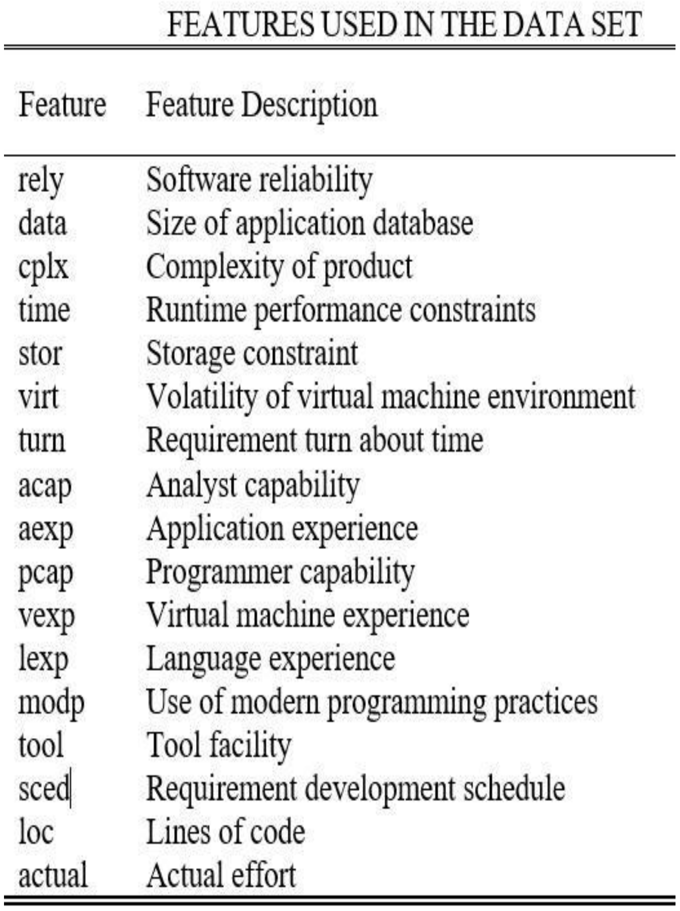
5. EXPERIMENT AND RESULTS

A. Dataset Collection:

The accuracy of the method is assessed using various Our study uses seven data sets. These data set contains the NASA projects which are mostly in the form of COCOMO approach. These data sets are collected from different NASA development centers around the United States.

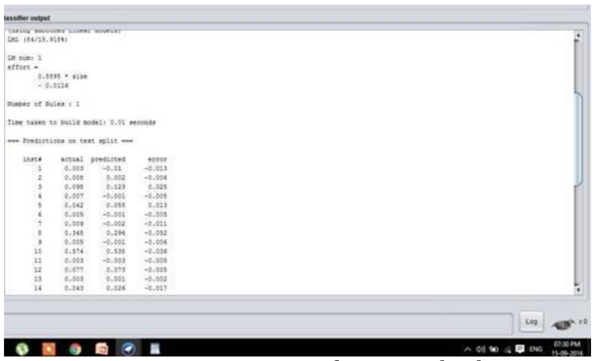
Three other data sets(cocomo81e,cocomo81o,cocomo81s) are mostly from southern California aerospace companies. Data sets have 17 features which are specified in table 2.These data set values are specified in the nominal form(low, very low, high, very high and so on).

B. RESULTS



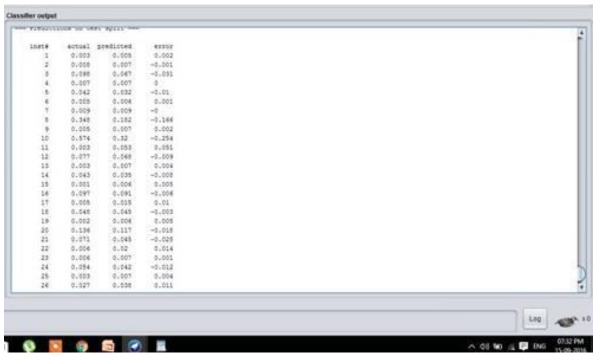
FEATURES USED IN THE DATA SET

Feature	Feature Description
rely	Software reliability
data	Size of application database
cplx	Complexity of product
time	Runtime performance constraints
stor	Storage constraint
virt	Volatility of virtual machine environment
turn	Requirement turn about time
acap	Analyst capability
aexp	Application experience
pcap	Programmer capability
vexp	Virtual machine experience
lexp	Language experience
modp	Use of modern programming practices
tool	Tool facility
sced	Requirement development schedule
loc	Lines of code
actual	Actual effort



```
(using smoothed linear models)
LM1 (64/18.918%)

LM num: 1
effort =
        0.0095 * size
        - 0.0114

Number of Rules : 1

Time taken to build model: 0.01 seconds

=== Predictions on test split ===

    inst#     actual  predicted      error
        1      0.003      -0.01     -0.013
        2      0.008      0.002     -0.006
        3      0.098      0.123      0.025
        4      0.007     -0.001     -0.008
        5      0.042      0.055      0.013
        6      0.005     -0.001     -0.005
        7      0.009     -0.002     -0.011
        8      0.348      0.296     -0.052
        9      0.005     -0.001     -0.006
       10      0.574      0.538     -0.036
       11      0.003     -0.003     -0.005
       12      0.077      0.073     -0.005
       13      0.003      0.001     -0.002
       14      0.043      0.026     -0.017
```

Fig.1 Output Screen of ECT method



```
=== Predictions on test split ===

 inst#     actual  predicted      error
     1      0.003      0.005      0.002
     2      0.008      0.007     -0.001
     3      0.098      0.067     -0.031
     4      0.007      0.007      0
     5      0.042      0.032     -0.01
     6      0.003      0.004      0.001
     7      0.009      0.009     -0
     8      0.348      0.182     -0.166
     9      0.005      0.007      0.002
    10      0.574      0.32      -0.254
    11      0.003      0.053      0.051
    12      0.077      0.068     -0.009
    13      0.003      0.007      0.004
    14      0.043      0.035     -0.008
    15      0.001      0.006      0.005
    16      0.097      0.091     -0.006
    17      0.005      0.015      0.01
    18      0.048      0.045     -0.003
    19      0.002      0.008      0.005
    20      0.136      0.117     -0.018
    21      0.071      0.045     -0.025
    22      0.006      0.02       0.014
    23      0.006      0.007      0.001
    24      0.054      0.042     -0.012
    25      0.003      0.007      0.004
    26      0.027      0.038      0.011
```

Fig.2 Output Screen of Random forest method

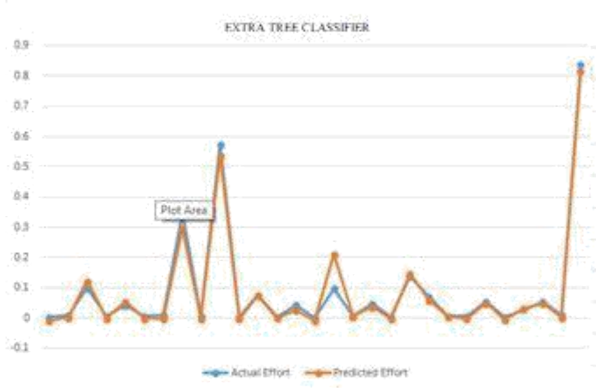


Fig.3 (ECT) Actual Effort vs. Predicted Effort Chart

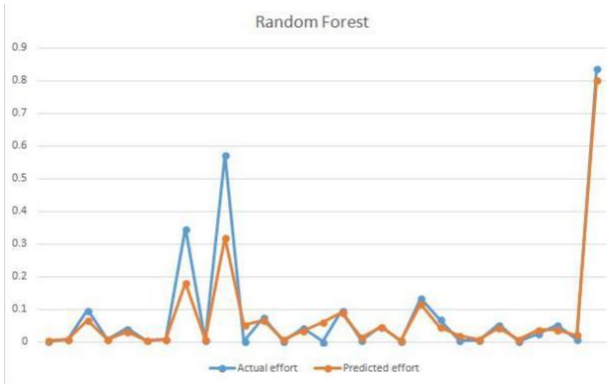


Fig.4 (Random forest) Actual Effort vs. Predicted Effort Chart

C. PERFOMANCE ANALYSIS:

prediction accuracy PRED = $(1 - \sum(x_i - y_i) / N) * 100$ % where,

x_i is the actual effort

y_i is the predicted effort and

N is the total number observations in the dataset For Random forest PRED = 97.52 %

For ECT PRED = 98.5 %

6. CONCLUSION

For successful project accomplishment, accurate estimation of software effort is essential in the software industry. In this paper, among different estimation techniques, we choose ECT and Random Forest methods. ECT technique performed better than Random Forest technique in terms of prediction accuracy and also this eliminates overfitting. Hence, it improves predictive accuracy by the reduction of overfitting.

FUTURE WORK

In the future, ECT can be contrasted with other approximate effort techniques like different machine learning based techniques and to analyse the generalization of results, apply ECT to software project data sets from various domains

VII. REFERENCES

[1] Mustapha, H., &Abdelwahed, N. (2019). Investigating the use of random forest in software effort estimation. *Procedia computer science*, *148*, 343-352.

[2] Nguyen, V., Boehm, B., & Huang, L. (2018). Determining relevant training data for effort estimation using Window-based COCOMO calibration. Journal of Systems and Software, 147, 124-146.

[3] Araujo, R. D. A., Oliveira, A. L., & Meira, S. (2017). A class of hybrid multilayer perceptrons for software development effort estimation problems. Expert Systems with Applications, 90, 1-12.

[4] Wu, D., Li, J., &Bao, C. (2018). Case-based reasoning with optimized weight derived by particle swarm optimization for software effort estimation. *Soft Computing*, *22*(16), 5299-5310.

[5] Zakrani, A., Hain, M., &Namir, A. (2018). Software Development Effort Estimation Using Random Forests: An Empirical Study and Evaluation. *International Journal of Intelligent Engineering and Systems*, *11*(6), 300-311.

[6] Gharehchopogh, F. S., Maleki, I., &Khaze, S. R. (2014). A novel particle swarm optimization approach for software effort estimation. *International Journal of Academic Research*, *6*(2), 69-76.

[7] Rijwani, P., & Jain, S. (2016). Enhanced software effort estimation using multi layered feed forward artificial neural network technique. *Procedia Computer Science*, *89*, 307-312.

[8] Dragicevic, S., Celar, S., & Turic, M. (2017). Bayesian network model for task effort estimation in agile software development. Journal of systems and software, 127, 109-119.

[9] Nanda, S., & Soewito, B. (2016, July). Modeling software effort estimation using hybrid PSO-ANFIS. In 2016 International Seminar on Intelligent Technology and Its Applications (ISITIA) (pp. 219-224). IEEE.

[10] Araujo, R. D. A., Oliveira, A. L., & Meira, S. (2017). A class of hybrid multilayer perceptrons for software development effort estimation problems. Expert Systems with Applications, 90, 1-12.

[11] Khalifelu, Z. A., & Gharehchopogh, F. S. (2012). Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation. Procedia Technology, 1, 65-71.

[12] Ali Idri, Ibtissam Abnanea & Alain Abranb, 2016, 'Missing data techniques in analogy-based software development effort estimation', The Journal of Systems and Software,vol.117, pp.595-611

[13] Aljahdali, S., Sheta, A. F., & Debnath, N. C. (2015, November). Estimating software effort and function point using regression, Support Vector Machine and Artificial Neural Networks models. In 2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA) (pp. 1-8). IEEE.

[14] García-Floriano, A., López-Martín, C., Yáñez-Márquez, C., & Abran, A. (2018). Support vector regression for predicting software enhancement effort. Information and Software Technology, 97, 99-109.

[15] Gharehchopogha, F. S., & Khalifehlou, Z. A. (2012). A new approach in software cost estimation using regression based classifier. Global Journal on Technology, 2.