ISSN: 0974-5823 Vol. 6 No. 3 December 2021

International Journal of Mechanical Engineering

Performance Comparison of Various Machine Learning Algorithms during Software Effort Estimation

Anil Kumar[1,2] and B.D.K. Patro [3]

[1]Dr. A.P.J. Abdul kalam Technical University, Lucknow, UP, India
[2]Department of Computer Science & Engineering, School of Computing and Information Technology, Manipal University Jaipur, Rajasthan, India, anil_kpawar@yahoo.com
[3]Department of Computer Science & Engineering, Rajkiya Engineering College, Kannauj, UP, India bdkpatro@rediffmail.com

***Abstract* - The handling of Software developing process is a tedious task as it is equipped with multiple tasks that demand to be done at the same time. One important parameter of the whole process is software effort estimation that requires lot of attention in low cost and low manpower. Over or under estimation of a project can ruin hard work in terms of time and cost. Choosing a machine learning technique for regression with a compatible dataset is a big issue, as different dataset gives different answers at times. To accurately predict effort, this paper presents a comparison between seven regression models that have been evaluated on China data sets. The resulting parameters are MEA, R-square error and RMSE and comparison proves that ANFIS Regressor and linear regression model gives least error and better stabilization.**

Index Terms - Robust Regression, Linear Regression, Artificial Neural Network, Support Vector Machine, Neuro-fuzzy, Random Forest, Stochastic gradient Descent.

INTRODUCTION

Software effort estimation (SEE) is a process that helps in estimating the total effort in terms of cost, manpower and time etc. in advance. If estimation goes wrong at any point of time the complete project can fail and investor might deal with loss. On the other hand, an accurate approximation, not only save time, resources and manpower effort but also helps in designing fault and error-free robust system. The initial inputs are project planning and pricing, investment analysis and financial plans etc. On the basis of inputs, a predicted approximations done on how the software is designed and what could be pro and cons. For accurate estimation, it is highly desirable to have better information of resources, time management and proper planning.

Estimators generally don't differentiate between important and irrelevant input information. The most important priority of client is the low cost that eventually reduces working hours which impacts bad on whole project. Estimation also relies on types of projects and its application. It can be categorized in simple organic, embedded, and semidetached complex projects.

The research is based on basic evaluation, recent framework, and advancement of solution. For correct evaluation, following two points should be included in research: first, estimates should include the project success or failure chances with respect to resources. Second, report on accuracy, collected resources plus fresh estimates excluding historic and wrong evaluations should be evaluated. The estimates should be comparable with actual work and required time. Contingency is always considered because risk and uncertainties can occur any time so extra emergency hours should be always desirable. For high accuracy, documentation for all possibilities and assumption is highly recommended. A better estimation can refine decision-making quality plus it gives a chance to manage risk within deadlines. Also, uncertainty like contingencies, variance, known and unknown risk should be considered and included while making a rough estimate. Estimation can be classified into (i) Machine Learning (ii) algorithmic and non- algorithmic. Machine learning in other words computational algorithms that enhance its own performance by learning and use of data. It builds model, train it with variety of data to make assumption and predictions. It comprises with neural network, fuzzy, SVM and Bayesian network etc. On other hand, Algorithmic SEE applied mathematical and statistics analysis. Non-algorithmic model uses interpretation and historic data. The fixed and rigid approach of algorithmic model can be overcome by ML.

There are many models have already been available in market on various topics like Robust Regression, Artificial Neural Network, Support Vector Machine, Neuro-fuzzy, Stochastic gradient Descent, k-NN and others. Each model works differently, and lacks result in some features. In this study seven popular model are compared through results and give detailed analysis on which is better and give good outcome. Recently, hybrid and meta-heuristic techniques are high in demand as two combined model used several cycles and factors that works better in terms of accuracy and error.

Other than Introduction, the paper is structured as follow. Section 2 is consisting of recent and related literature work. It is then followed by section 3 that explains experimental set up and proposed structure. Section 4 is covering results and relative tables. Last section 5 concludes the study and focused on future work.

RELATED WORK

Estimation of effort is part of Software development process. It can be based on non-agile and agile process. Non-agile is the conventional method that combines multiple lifecycles from planning, requisition, testing, updating and maintenance. Agile method is versatile and iterative in nature. Limited information resources and similar prerequisites affect whole process of effort estimation. An average of 6% of total budget of software making is spending on evaluation which results into limited expert manpower and low overhead. Required data, robustness, informative power, high accuracy and reliability are the basic key factors that kept in mind while choosing estimation method. Some related research papers are lined up below that been utilized in the entire study.

Fernández-Diego et al. [1] studied 73 different papers of Agile Software Development (ASD) from 2014 to 2020. This paper identifies Scrum, Xtreme programming and other four agile methods to calculate accuracy and effect size. The mean magnitude of relative error (MMRE), MER and Balanced Relative Error (BRE) are parameters that have been used for comparison.

A. Trendowicz et al. [2] presented a review study of effort evaluation for the field of industrial areas. It explains the traditional estimation methods of effort. To maintain demand of software, more resources, improved functionality and regular updatesare mandatory within balanced cost.

E. Karunakaran et al. [3] reviewed software sizing model and effort evaluation for recent years. Study is based on component size, Proxy size, SEER-SEM and software life cycle model (SLIM) etc. Most of the model are fuzzy based while regression-based models are not that much explored in comparison.

Different models like Analogy based estimation (ASEE), Regression model, Neural Network (NN), Fuzzy, Grey relational Analysis (GRA) estimation model are compared by P. Kumar et al. [4] to find out a best possible model for higher prediction.

H. Rastogi et al. [5] subsumed multiple techniques like PSO, ANN, SVM and others hybrid model with respective merits and demerits on COCOMO II data set. Study explains hybrid model like ANN-PSO, and Neuro-Fuzzy model performs better than conventional models.

S. K. Sehra et al. [6] developed a hybrid model to combine fuzzy based machine learning called fuzzy analytic hierarchy process (FAHP) to rank the features for selection purpose. FAHP has combined with kernel least square SVM for calculation of effort.

Dukka K. K. Reddy and H. S Behera [7] presented the non-algorithmic approach for PSO to study outcome for problems like uncertainty, off base and other constraints. Multiple models using PSO are compared on NASA data set. Chaos theory [14] with PSO is integrated that applies tent mapping on COCOMO I and MARE data set. The absolute relative error is up to 0.079% which is remarkable.

F. Samadzadegan et al. [8] provide a solution for multi-class problems through Genetic algorithm and SVM. This paper explains how to optimize the SVM parameter and apply Kernel in binary classification. The results show that proposed model improves classification accuracy.

Particle Swarm optimization (PSO) is an optimization technique that has been widely used in optimizing or tuning global or local best values. A research on PSO infused with GA [9] is done for quick and stabilize tuning in multi modal functions. In a recent research of 2020, P. Suresh Kumar and H. S. Behera et al. [10] presented a comparison on KNN, SVM, NN, RF and back propagation model through Orange data mining tool. On a fixed learning rate of 0.5, MMRE is calculated to conclude results.

K.Langsari et al. [11] optimized effort parameters using Fuzzy and Multi objective PSO on COCOMO II dataset. The results are optimized on MMRE that reduce error rate to 11.89% to 8.08%. The model although has a drawback of lower accuracy. S. Grimstad et al. [13] explains problems arises in effort approximation. How is it done without clarity of input availability? Reliable estimation is still a big question. Proper testing time and scheduled overrun and minor estimation error can't be ignored. Different Regression Fuzzy model [17] is assembled named MLR, Mamdani, Sugano constant and linear for three input variables i.e. resource data, software and team size to analyze performance. Comparison of multiple models also explain the need of change in heteroscedastic behavior of data set.

Using COCOMO II data set, an approach on learning with recursive feature elimination [12]is presented. The effort estimation is done using recursive feature ranking and selection method and later seven ML model are developed. During feature selection, main requirement is to selection some important feature while ignoring the rest. It not only increases accuracy but also reduce the computational time. Features like reliability, LOC, software tools, actual cost, analysts and programmer capabilities and memory constraints and other are included in any dataset, but for ranking and selection point of view only few parameters are taken in to account. Traditionally tuned parameters mostly perform inaccurate during evaluation, so hyperparameters are tuned according to model to reduce the overall error. To solve this problem S. K. Palaniswamy and R. Venkatesan [15] presented two approaches first PSO and second GA to tuned parameters. Effort estimation for multiple Agile and Non-agile software using DBN-ALO is explained by A.

Kaushik et.al [16]. Deep belief network is a sub-part of neural network. DBN model is composed with two layers using Boltzmann machine in which one is hidden and other is visible. Carbonera C. E. et. al. [19] reviewed recent work in effort approximation through various items like application, cost, pricing, productivity, and maintenance etc. Outliner's detection method [20] using weighted approach for dataset is a popular technique. This detection method also relies on interquartile range, error, and absolute deviation. Mahmood Y et al. [21] presented a detailed review on software effort estimation (SEE) using different case point. The study is based on findings of current research, accuracy parameters, availability, and usages of dataset.

There are multiple ways for estimation methods like analogy-based technique, parametric approach, and De/Re composition. Many have been discussed above. The proposed study aims to give a good comparison between different model's performance. The next section explains the proposal and methodology of model.

EXPERIMENTAL SETUP AND PROPOSED ALGORITHM

The setup is developed to measure the performance that composed with a module, extractor, data processor and performance measuring parameters. Dataset utilized for proposed algorithm is divided in to testing and training sets for learning and evaluation purpose. Figure 1 show the block diagram of experimental setup and Following steps are taken while evaluation of different model.

Step 1: Extract dataset D of F feature from a software model.

Step 2: Do data pre-processing of D.

Step 3: Break dataset into two-parts, one for training D_{tr}(75% of D) and other for testing D_{ts} (25% of D).

Step4: Train and test the given ML model on given data set.

Step5: Evaluate the performance of ML models by calculate R square, MAE, RMSE.

IMPLEMENTED MODEL DESCRIPTION

Linear Regression: the statistical method to calculate parameters that remove noise and co-linearity. It is utilized for reliable prediction and calculation. First data is analyzed for linearity and sample data is collected for model. The ultimate model is then used for making prediction. The equation for LR model is shown below.

$$Y = \beta_0 + \beta_1.X + \epsilon \quad (1)$$

Where X is independent, and Y is dependent variable. β_0 *and* β_1 *are respective intercept coefficient for* Y *and* X *a* and ϵ is error.

Robust Regression: it is designed to remove outliers (external noise) from data. Robust regression calculates optimal parameters from data while removes the outliers. For robust modelling, the RANSAC algorithm is applied on data considering total data items are M and parameters are calculated over N data items only. While estimating parameters, the goal is found out the maximum number of data items fit for the proposed model. The value of fit data items and relative structure should be as much as possible for a successful model designing. The number of iterations for model is assuming as L, to calculate the value of number of loops following equation is desirable.

$$L = \frac{\log(P_f)}{\log(1-(P_s)^N)} \quad (2)$$

Where P_f is probability for good fit data items and P_s is the probability for randomly selected data items.

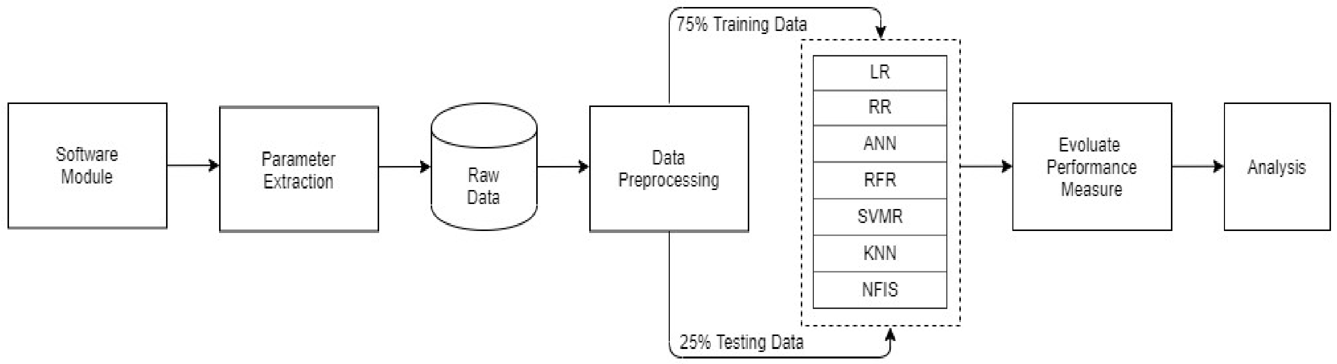


FIGURE 1. Block Diagram of Proposed Model.

Artificial Neural Network: In machine learning algorithms it is most acceptable model for classification and evaluation. It is a feed forward or feedback multilayer model that works as human brain. Neural network prefers because it can handle big dataset and higher variables. The model can correlate variable and patterns quickly. A basic block diagram for NN model is shown below. It comprised with one input layer, one output layer and some hidden layers between input and output.

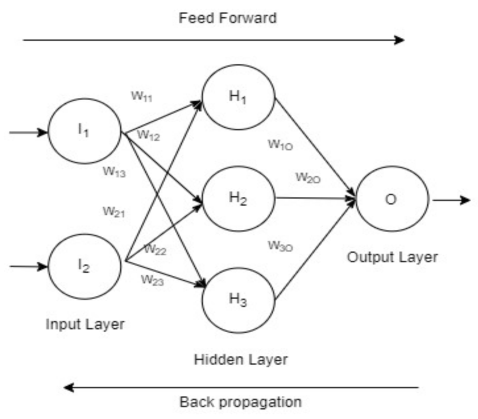


FIGURE 2. Basic architecture for Neural Network

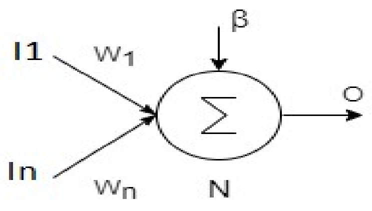


FIGURE 3. Simple neuron structure

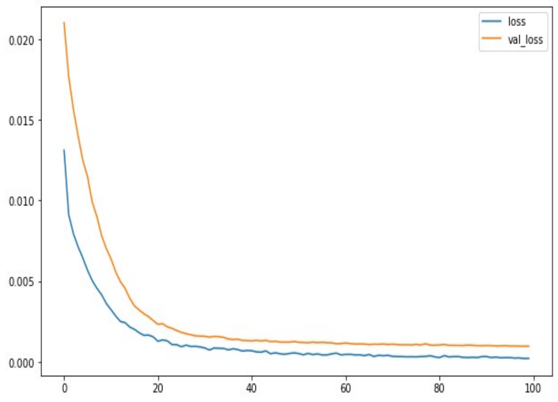


FIGURE 4. Error loss in NN during iteration

Where W is weight of link between one layer to another. The outputis calculated in terms of weight, batch size and learning rate etc. A transfer function is the weighted summation of input and bias.

$$O = \emptyset\left(\sum_{i=1}^{n} W_i * I_i + \beta\right) \tag{3}$$

Here W_i is weight of neuron's link, I_i is input to neuron, β is bias and $\emptyset$ is activation function. In this experiment 'relu' activation function is used and loss is calculated in term mean square error (MSE). Figure 4 show how loss (error) is reduced per iteration.

Random Forest Regression: It combines various predictions from multiple ML techniques for accurate. It is based on ensemble learning that works in following steps; first select any random k data items for training. Second design decision tree accordingly and lastly tree repeats the loop up to any number let's say N in this case. New data items are assigned to sum up all predicted values.

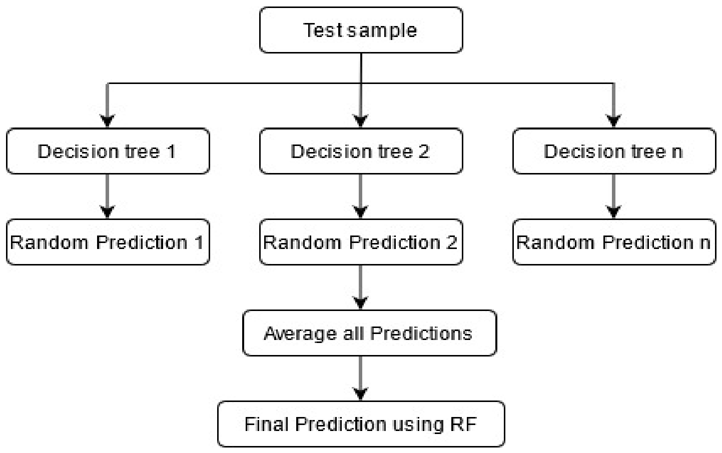


FIGURE 5. Model for random forest regression

SVM Regression: Support Vector Machine [8] works on static theory and fixed results. It provides the optimal solution through hyper plane feature that first train input values. It is first introduced by Vapnik that employs kernel learning. For regression purpose, a maximal margin is set to approximate the value of SVM. The output is a real number, so estimation lies in infinite outcomes which make the process very complex. SVR is basically used to minimize error by maximizing margin.

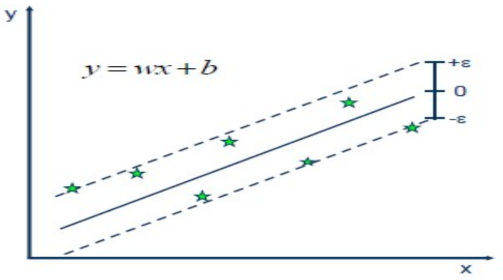


FIGURE 6. Hyper plane show regression or classification of inputs

For Linear SVR case, W is a normal vector for function f(x). The optimization equation can be written as

$$\min \frac{1}{2}|W|^2 \tag{1}$$

$$\text{constrain } \; y_i - wx_i - b \leq \epsilon$$
$$wx_i + b - y_i \leq \epsilon$$
$$y = wx + b$$

Where, ϵ is error that can be ignored until the value is lesser than ϵ.Here RBF kernel is used during SVM regression.

KNN: K- nearest neighbor regression is a non-parametric method that average same nearby observation. The near-by distance is the distance between new point to training point. It can be calculated from Euclidian, Manhattan or via Hamming distance. The size of observations (K) should be selected by analyst in such a way that minimize mean square error. Steps for KNN are following; first, the data should be initialized, and value of K must be chosen. Secondly, the distance will be calculated from training point to next point. Third, arrange the distance in ascending order to select K entries. For regression, K is selected through mean value. Here K=8 neighbors is utilized to implement this model. It is found that at K=8 it gives best result.

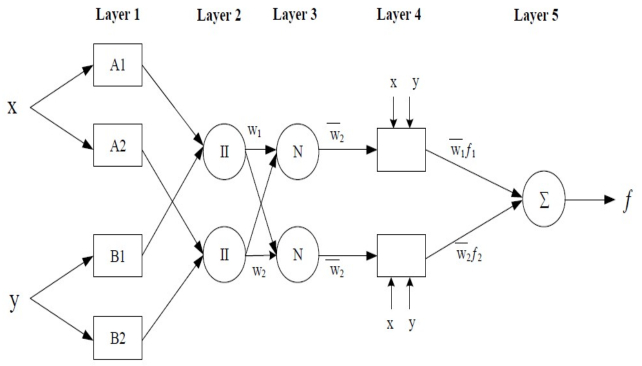


FIGURE 7. Basic architecture of ANFIS method.

Neuro-Fuzzy Regression: A technique that combine neural network and fuzzy logic is known as ANFIS (Adaptive neuro-fuzzy inference system). The basic architecture of ANFIS is shown in figure 7 that contains 5 layers feed forward NN based on constructing rule system. Input **x** and **y** are given to layer 1 that is a fuzzifier layer which is used to grade the input values. Layer 2 contains strength rule which is a product of different x and y members. Layer 3 calculate weighted ratio individually and lastly sum up all rules. Layer 4 is defuzzifier and last layer is a single node summer that add all input values. ANFIS is based on Takagi-Sugano-Kang rule for linear function.

ANFIS construction is based on fuzzy while neural network uses it adaptability of learning. Rather than using sigmoid function for neural network, ANFIS convert the function into fuzzy numerically normalized values lies between 0 and 1.

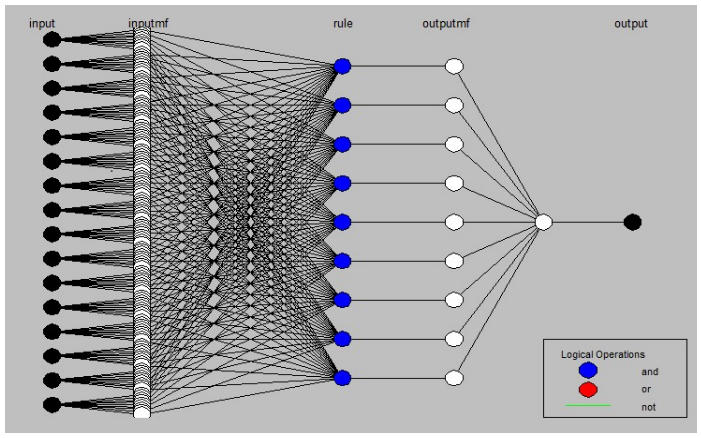


FIGURE 8. Structure diagram of ANFIS Model

During implementation of ANFIS, grid partitioning is utilized. Model is train upto 100 iterations. Basic architecture of five layers of ANFIS is shown in figure 7 and implemented structure of ANFIS is shown in Figure 8.

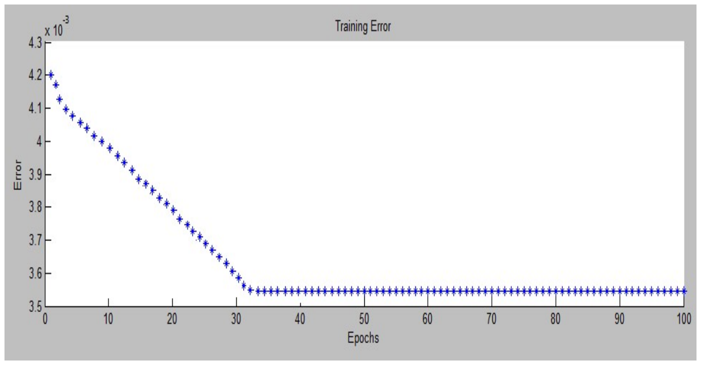


FIGURE 9. ERROR LOSS AT EACH EPOCH

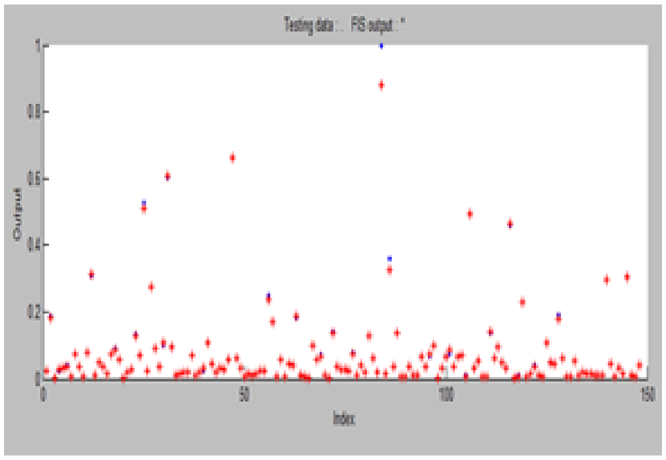


FIGURE 10. Data plots during regression testing

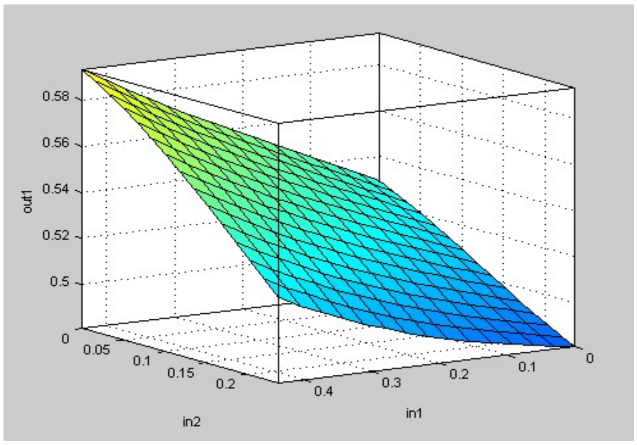


FIGURE11. Surface plot of ANFIS model

FIGURE12. Fuzzy rules and their membership functions

Various performance graphs of Error loss at each epoch, data plot for regression testing and surface plot are shown in consecutive Figure 9-12. Here, epoch is a measure of time at a particular instant, data plot relates the data between input and output variable and surface plot is a three-dimensional plot that interrelate the functionality between dependent and two interdependent variables.

PERFORMANCE ANALYSIS

In this section, parameters like MEA, RMSE etc. have been discussed on which result is simulated. These three parameters are evaluated in two modes: testing and training mode. Respective outcomes, scatter plots and graph are attached in later section. Performance of various algorithms are analyzed on behalf of following measurements. The parameters are explained below:

MEAN ABSOLUTE ERROR (MAE)

It is the average value of error for continuous values. It is a linear value as the weightage is equal for all.

$$MAE = \frac{\sum_{i=1}^{n} | y_i - y_i' |}{n} \qquad (5)$$

y_i' *is true value.*

Where y_i is the prediction values and The difference between prediction and true value is termed as absolute error?

ROOT MEAN SQUARE ERROR (RMSE)

It is the standard deviation of prediction error or average magnitude of the error. RMSE explains that how error values are spreading out and weather the data is lying around fitted regression line or not. It can be done in three steps;

- Square the difference.
- Calculate the average for all values and,
- Lastly taking the square root of final value.

$$RMSE = \frac{\sum_{i=1}^{n} (y_i - y_i')^2}{n} \qquad (6)$$

Where n is the sample size. RMSE is desirable for small error only.

R SQUARED SCORE

It is also known as coefficient of determination for multiple regressions. It measures weather data points are spreading out or lying around fitted line? It is measured in percentage from 0 to 100 %. If the R square value is 1 that means differences between the observed data and the fitted values is zero. It can be calculated by dividing sum of square of regression model by total sum of square of error.

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total variance}} \quad (7)$$

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2} \quad (8)$$

k-fold cross validation scheme is used during analyzing of these ML model. Here whole dataset is divided into 10 parts and each time one part is utilized for testing and of testing, it is 0.9816, that clarifies that the fitted value is nearly zero and accuracy is high. The absolute mean error (0.0172 for training and 0.0139 for testing) and root mean square error (0.0035for training and 0.0031 for testing) is low.

Table 1 Values of MAE, RMSE and R square during training and testing.

	Training			**Testing**		
Model	**MAE**	**RMSE**	**R Square**	**MAE**	**RMSE**	**R Square**
Linear Regression	0.0057	0.0123	0.9863	0.0053	0.0108	0.9847
Robust Regression	0.0045	0.017	0.9741	0.0051	0.0113	0.9841
Artificial Neural Network	0.0069	0.0117	0.9476	0.0154	0.0384	0.9335
Random Forest Regressor	0.0027	0.0094	0.954	0.0071	0.0262	0.9391
SVM Regressor	0.008	0.019	0.9099	0.0094	0.041	0.8606
KNN Regressor	0.0218	0.0495	0.8359	0.0248	0.0584	0.8176
ANFIS Regressor	0.0172	0.0035	0.9884	0.0139	0.0031	0.9816

remaining 9 part is availed for training. We take mean of accuracy and error generated during each fold.

Table 1 show the performance of various ML algorithms in term of MAE, RMSE and R2 square. In this table it is clearly visible that in the term of MAE, Random Forest Regressor gives best result compared to other. On other hand ANFIS give best result in term of RMSE and R2 square score.

Figure 13 show the comparative performance of various ML algorithm in term of R2 square which show ANFIS is the best Model compared to other. It has high value of R-square and low values for both error parameter that makes it suitable for the proposal. Radom Forest regression gives good results (RSME = 0.0262 and R square = 0.9391) as well in comparison to other models.

Figure 14 shows the scatter graph of various algorithms, which show relationship between expected outcome and the output, comes from various algorithms. In this graph we can see that e and f perform worst compared to other algorithm and ANFIS gives best regression relation almost identical.

In regression it is important that model should be generalized, means during training and testing model should give same amount of error. So, to show generalized behavior of different ML model three graphs (figure 14) are constructed. These graphs show different error and R2 square value during training and testing times.

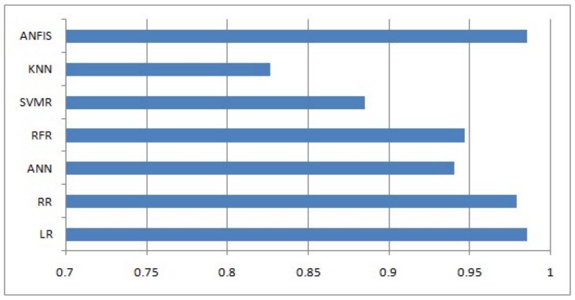
During the analysis, from Table 1, it is found that the ANFIS model shows most promising results than other models. During training the R square is 0.9884 and at time

CONCLUSION AND FUTURE WORK

Effort evaluation is done by using machine learning method are high in demand nowadays. In this paper a detailed comparison of these popular ML techniques is done to analyze the performance. Seven regression models have been analyzed for the sake of better outcome. Although

result directly depends on how many and which features have been selected for experiment. China data set is used for study. The measured parameters are MEA, RMSE and R-square error. The ANFIS model that combine fuzzy and neural network shows the best result and give least error. Future work can be modified by incorporating more feature and big dataset. Research questions for further development could be: which dataset is good and universal; Which parameters should be chosen for feature extraction; how conventional models can apply meta-heuristic approach to upgrade itself for the estimation process?

FIGURE 13: Performance comparison of all ML models in term of R square



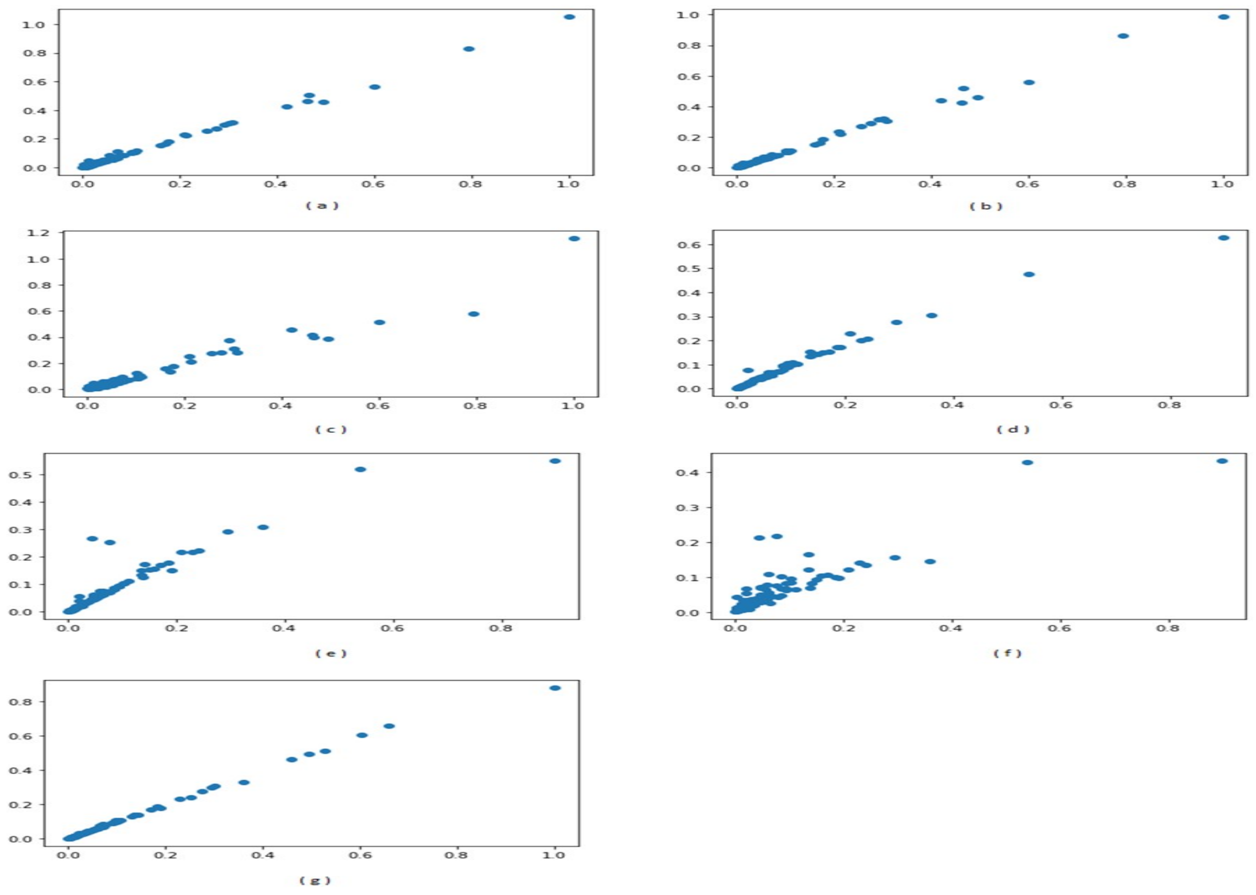


FIGURE 14. Scatter plot of expected and real outcome values. a) LR, b) RBR, c) ANN, d) RFR, e) SVMR, f) KNN, g) ANFIS

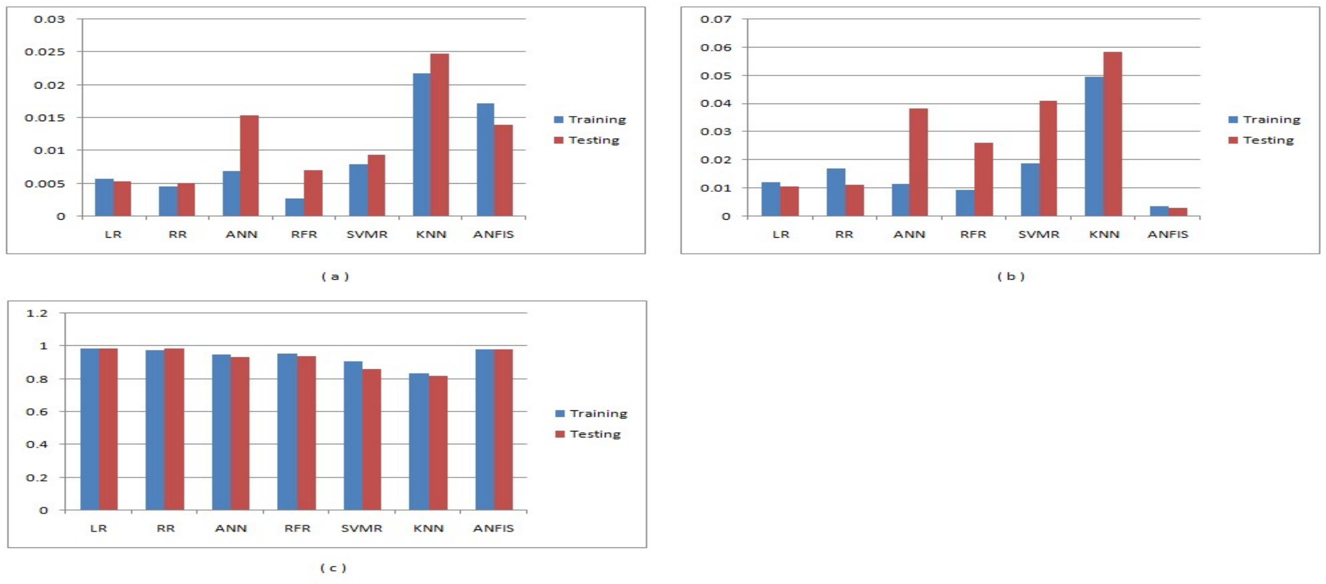


FIGURE 15. Comparison of outcome during training by considering a) MAE, b) RMSE, c) R-square

REFERENCES

[1] Marta Fernandez Diego, Erwin R. Mendez, Frenando Gonzalez lardon-de-Gueva, "An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review" DOI 10.1109/ACCESS.2020.3021664, IEEE Access VOLUME 1, 2020.
[2] A. Trendowicz, J. Münch, and R. Jeffery, "State of the Practice in Software Effort Estimation: A Survey and Literature Review" CEE-SET 2008, LNCS 4980, pp. 232–245, 2011.
[3] E. Karunakaran, N. Sreenath, "Survey on Software Effort Estimation Technique – A Review" International Journal of Scientific & Engineering Research, Volume 6, Issue 12, December-2015, ISSN 2229-5518.
[4] P. Kumar, M. B. Reddy, L. Lavanya, "A Survey Report on Software Estimation models and techniques" International Research Journal of Computer Science (IRJCS) ISSN: 2393-9842, Issue 12, Volume 3 (December 2016).
[5] HimaniRagtogi, S. Dhankhar, Misha Kakkar, "A Survey on Software Effort Estimation Techniques" 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), DOI: 10.1109/CONFLUENCE.2014.6949367. 2014 IEEE.
[6] Sumeet Kaur Sehra, Yadwinder Singh Brar, Navdeep Kaur, Sukhjit Singh Sehra, "Software effort estimation using FAHP and weighted kernel LSSVM machine" Soft Computing (2019) 23:10881–10900, https://doi.org/10.1007/s00500-018-3639-2.
[7] Dukka K. Karan Reddy, H. S Behera, "Software effort estimation using Particle Swarm Optimization: Advances and Challenges" 2020 https://doi.org/10.1007/978-981-15-2449-3_20.
[8] F. Samadzadegan, A. Soleymani and R. Ali Abbaspour, "Evaluation of Genetic Algorithms for Tuning SVM Parameters in Multi-Class Problems" CINTI 2010, 11th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary.
[9] Soumya D. Mohanty (2012) Particle Swarm Optimization and regression analysis – I, Astronomical Review, 7:2, 29-35; https://doi.org/10.1080/21672857.2012.11519700.
[10] P. Suresh Kumar and H. S. Behera, "Estimating Software Effort Using Neural Network: An Experimental Investigation" 2020 https://doi.org/10.1007/978-981-15-2449-3_14.
[11]Kholed Langsari1, Riyanarto Sarno2, "Optimizing Effort and Time Parameters of COCOMO II Estimation using Fuzzy Multi-Objective PSO" Proc. EECSI 2017, 978-1-5386-0549-3/17/$31.00 ©2017 IEEE.
[12] K. Eswara Rao1, G. Appa Rao2, "Ensemble learning with recursive feature elimination integrated software effort estimation: a novel approach" Evolutionary Intelligence 2020 https://doi.org/10.1007/s12065-020-00360-5.
[13] Stein Grimstad, MagneJørgensen, KjetilMoløkken-Østvold,"Software effort estimation terminology: The tower of Babel" 2006 doi: 10.1016/j.infsof.2005.04.004.
[14] Zahra A. Dizaji, K. Khalilpour. Particle swarm optimization and chaos theory-based approach for software cost estimation. International Journal of Academic Research Part A; 2014; 6(3), 130-135. DOI: 10.7813/2075-4124.2014/6-3/A.18.
[15] Sampath Kumar Palaniswamy, R. Venkatesan, "Hyperparameters tuning of ensemble model for software effort estimation" 2020 Journal of Ambient Intelligence and Humanized Computing https://doi.org/10.1007/s12652-020-02277-4.
[16] A. Kaushik, Devendra Kr. Tayal, Kalpana Yadav, "A Comparative Analysis on Effort Estimation for Agile and Non-agileSoftware Projects Using DBN-ALO" Arabian Journal for Science and Engineering 2019, https://doi.org/10.1007/s13369-019-04250-6.
[17] Ali Bou Nassif,Mohammad Azzeh,Ali Idriand Alain Abran, "Software Development Effort Estimation Using RegressionFuzzy Models" Computational Intelligence and Neuroscience, Volume 2019, Article ID 8367214, 17 pageshttps://doi.org/10.1155/2019/8367214.
[18] Saurabh Bilgaiyan, Samaresh Mishra, Madhabananda Das, "Effort estimation in agile software development usingexperimental validation of neural network models" 2018, https://doi.org/10.1007/s41870-018-0131-2.
[19] Carlos Eduardo Carbonera, Kleinner Farias, Vinicius Bischoff, "Software development effort estimation: asystematic mapping study" IET Softw., 2020, Vol. 14 Iss. 4, pp. 328-344, 2020doi: 10.1049/iet-sen.2018.5334.
[20] Petr Silhavy, Radek Silhavy and ZdenkaProkopova, "Outliners Detection Method for SoftwareEffort Estimation Models" CSOC 2019, AISC 984, pp. 444–455, 2019, https://doi.org/10.1007/978-3-030-19807-7_43.
[21] Mahmood Y, Kama N, Azmi A. "A systematic review of studies on use case points and expert-based estimation ofsoftware development effort." J SoftwEvol Proc. 2020;e2245. https://doi.org/10.1002/smr.2245.