Mid Review Report

Software Development Effort Prediction using Ensemble Model

Submitted by

Sachin Dawar

Roll No. : 197942

Department of Computer Science

National Institute of Technology, Warangal

April 2022

Under the supervision of

Dr. MANJUBALA BISI

Dr. MANJUBALA BISI	Sachin Dawar
Assistant Professor	3rd Year
Dept. of CSE	MCA CSE
NIT Warangal	Reg No.:
mc19156	

CERTIFICATE

This is to certify that the work which is being presented in this mid-term report entitled as **"Software Development Effort Prediction using Ensemble Model" is submitted by Sachin Dawar (Roll no. 197942)** in partial fulfilment for the award of the Degree of Master of Technology in Computer Science and Information Security and submitted to the Department of Computer Science and Engineering of National Institute of Technology, Warangal is an authentic record of his own work carried out under my supervision. The matter represented in this report has not been submitted by him for award of any other degree of this or any other institute/university.

Dr. MANJUBALA BISI

Assistant Professor

Dept. of CSE

NIT Warangal

Abstract

Software development effort prediction is an necessary tool of software engineering for effective planning and delivering successful software projects at Time, Cost and Budget. The main aim of this project is to improve the accuracy of software development effort estimation in order to help software development firms and practitioners. In software engineering, effort estimation would help managers and project team to estimate, forecast, and accurately quote the requirements for schedule, budget and manpower to successfully complete software projects.

Contents

List of Figures

List of Tables

1 Introduction

The process of estimating the amount of work and hours required to develop software is known as software Software Development Effort prediction . It is typically measured in man-hours or man-months. Developing software systems nowadays is both costly and demanding. There are various methods for quantifying a project in software engineering. One of the most critical tasks in the software engineering process is to accurately estimate the software engineering process, which includes cost, effort, and time .These factors all play a role in determining whether the project succeeds or fails. Software effort prediction is very important because Software firm have proper over project and they can plan systematically .We can identify the resources how much, effort, resources, and time it will take to build specific software product, So they will deliver the project at right time. Effort will depend upon productivity.Prediction is based on past data/past experience,available knowledge, assumptions. project manager see the productivity data of the past projects to decide the effort prediction for the new project.

Managers would be able to estimate, forecast, and properly quote the needs for schedule, money, and personnel to effectively finish software projects using *software development effort prediction* in software engineering. The project team has a very difficulty in delivering good-quality software to end users on time and on budget. Several studies have stressed the relevance of the software project manager's role in the success or failure of a project.

1.1 Objective

To predict the software project effort using Software Development Effort Prediction using Ensemble Model so that project will succeed at on right time and right cost.

1.2 Motivation

Nowadays it's a critical task for a software firm to develop a project at the right time and at cost, So motivation behind this software effort estimation is to help managers and project team to predict the project effort using some historical software parameter,So that Software firm will develop the project at the right time and cost and at the right budget.

2 Problem Description

Software project’s success depends primarily on its accuracy in estimating effort. To date, a lot of research has been conducted to estimate the accuracy of software effort using distinctive techniques. In any case, researchers and experts are attempting to determine which estimating technique produces increasingly accurate results on specific data sets and other relevant characteristics.. Failure of a software project indicates recognizable cost, Scope, effort, schedule, or quality failure.

3 Related Work

Software Development Effort Prediction tool is very helpful for software firms . Over the past few years, there is a many work has been performed on various types of Software Development effort estimation techniques and a many models have been proposed to achieve high Software Development effort prediction accuracy . The use of regression analysis; also known as algorithmic estimation is one way to predict software effort.

1. paper, an ensemble model is proposed that incorporated Use Case Points (UCP), expert judgment and Case-Based Reasoning (CBR) techniques to improve the estimation accuracy prediction of software development effort. use case points are a widely used method for measuring software project size and predicting effort.A Case-Based Reasoning (CBR) is the use of an analogy approach to machine learning and is the most investigated method of estimating the effort based on machine learning. CBR's first implementation was proposed in 1997 in software effort estimation [1].

2. Usman, M., et al.[2] compared the COCOMO model using KLOC with COCOMO using Function Point Analysis (FPA). They investigated that the COCOMO model used with FPA has given more accurate results than KLOC using MMRE evaluation measure.

3. Omar Hidmi1, and Betul Erdogdu Sakar [3]Estimate software effort objectively by using machine learning techniques instead of subjective and time consuming estimation methods. Models using two machine

learning techniques which are Support Vector Machine (SVM) and K-Nearest Neighbor (k-NN) separately and combining those together using ensemble learning were tried on two public datasets namely Desharnais[21] and Maxwell dataset[22]. Results show that svm technique outperform k-nn technique, also ensemble learning improves the results.

4 Proposed Solution

In proposed approach Ensemble Machine learning has been used. Ensemble Machine learning technique consists of combinations of more than one single technique to estimate the software development effort of a new project using a combination rule i.e. mean, median, Inverse Rank Weighted Mean-IRWM, etc. The estimation of each base model is combined that produced the estimation of an ensemble.

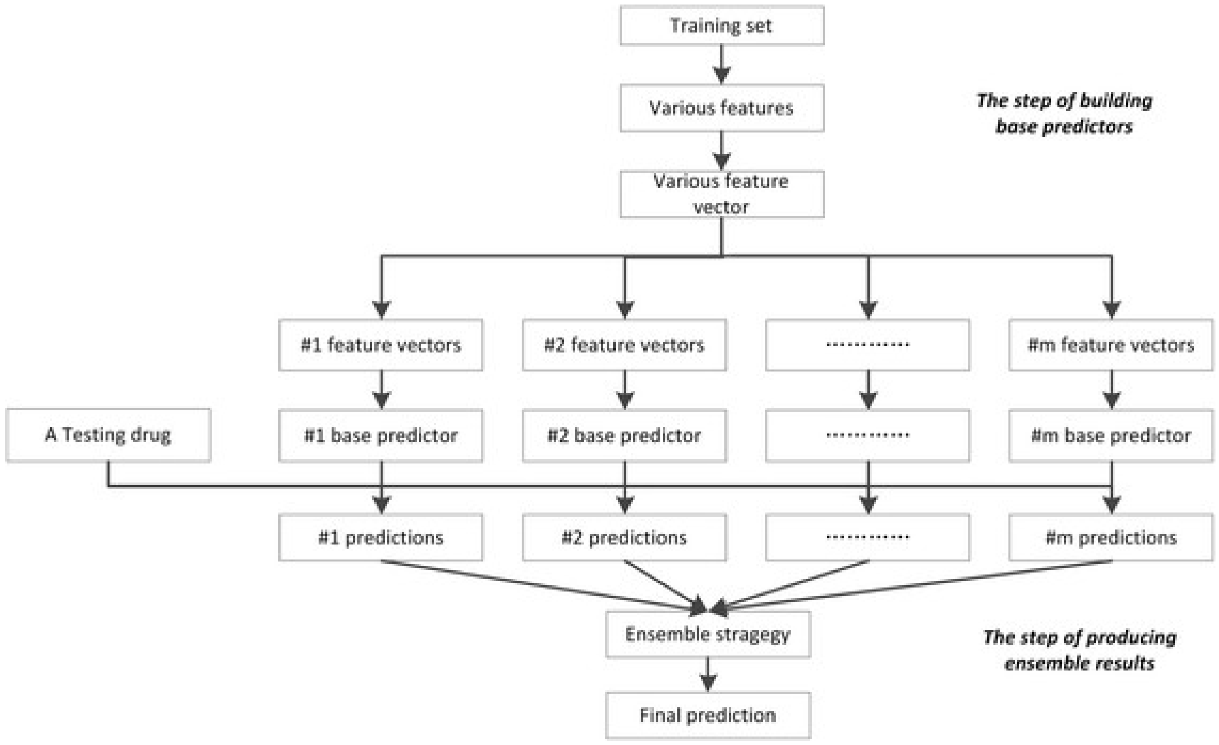


Figure 1: Steps of Ensemble Machine learning Working

4.1 Software firm Dataset Procurement:

For Software development effort prediction Desharnais[21] and Maxwell dataset[22] were used. And we obtained these datasets from Github and Promise Repository. The below table i.e. Table 1 represents all the details

Dataset	Features	Size	Unit	Source
Desharnais	11	81	Hours	Github
Maxwell	26	62	Hours	Github

Table 1: Table 1: Dataset Record

of datasets that we are using for Software Development effort prediction including the number of features, the total number of entries, and software effort measurement units.

4.2 Data Prepossessing:

In this process first step that we have followed is the cleaning of data by removing the unnecessary columns like ’ID’s’ and NAN datatype, and the missing values are being checked.And our target is ’Effort’ which is common in both of the dataset. Scaling of data using MinMax Scalar .We have used MinMax Scalar Module for scaling our data using the scikit learn framework which adjusts the input features value. so we adjusted each feature value between 0 and 1. Scaling the features brings the feature vectors to the same scale so that the model is not negatively impacted by large variations in the data and allowing models to learn more effectively . There should not be any duplicate entries or noise in data.

4.3 Train and Test Dataset:

After prepossessing the data , data separated into two part train and test the ratio of 80% and 20% respectively.This would impact the accuracy of predicting the result. In this step data is ready to Splitted into two parts for the train 80% data and for test 20% data is used .

4.4 Data Fitting in Model:

In this project data is fitted into various of models like Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and Ensemble Model learning like Bagging, Boosting. And trained the models using the training dataset.

4.5 SVR:

Support Vector Regression is a supervised learning algorithm. The same technique as SVMs is used in Support Vector Regression. Support Vector Regression is used to predict discrete values. The idea behind SVR is to find the best fit line that is a hyperplane. hyperplane has the maximum number of points.Support Vectors are the points that are closest to the hyper plane. Unlike other Regression models, SVR also tries to minimize the error between the actual and predicted value.

4.6 Decision Tree Regressor :

The Decision Tree Regressor is a Supervised Machine Learning technique in which a data set is divided into small parts and a tree-structure is formed based on specific splitting conditions. It is made up of a number of decisions. There are two types of nodes: decision nodes and leaf nodes. The root node is the first node, and it may be split into further nodes. Leaf nodes store the conclusion or decision, while decision nodes are responsible for data splitting.

For node splitting, Variance Reduction technique used because target variable is continuous.

Variance Reduction : It is measure for deciding the condition on which node is split in further nodes

4.7 Random Forest Model:

Random Forest utilise begging ensemble technique .It is supervised learning technique which makes use of ensemble learning to perform both a regression and classification task .Ensemble learning which is a process of combining multiple algorithm to solve a complex problem and to improve the performance of the model. Random forest constructs multiple decision trees to predict the class of the data set. Trees in Random forest run parallel at same time .

Steps of Random forest algorithm:

- Step 1: Data-set is divides into N number of random records or subset and each having k number of records.
- Step 2: for each subset individual decision trees are constructed.
- Step 3: Every decision tree will give an output .
- Step 4: Final Results is considered based on Averaging of each decision tree's ouput.

4.8 Bagging meta-estimator:

A bagging meta-estimator is an ensembling algorithm used for both classification and regression problems.For our project we have to use Bagging Regressor . It follows the bagging technique to make predictions. In the bagging or bootstrap aggregating technique we create subsets from the original dataset with replacements, that subsets are called multiple datasets or bootstrap samples. A base model is created on each of these subsets and then we have to train these base models using these samples. All base models are run in parallel and are independent of each other. The final predictions are determined by combining the results we get from all the models.

steps for the bagging meta-estimator algorithm:

- First, we create Random subsets from the original dataset.
- This subset includes all features of the dataset.
- To These subsets, a base estimator has to be fitted
- At the end, we combine the results of each model to get a final result.

4.9 Gradient Boosting :

A Gradient Boosting or GBM (Gradient Boosting Machine) is an ensembling algorithm used for both classification and regression problems. Gradient boosting is combining multiple machine learning models (mainly decision trees) and each of this decision tree model is used for predication For our project we have to use Boosting regressor . It follows the Boosting technique to make predictions. Boosting tack care the situation if a data point is predicted wrong by the first model then the next will combine the predicted result to provide correct results. Boosting is a sequential process, where each model tries to correct the errors of the previous model. The success of models is dependent on the each of previous models.

Steps of Boosting working

- firstly, equal weights are given to all data points.
- Subsets are created from the original dataset(subsets are called multiple datasets).
- A base model is created on each of these subsets and then base models train using these samples.
- Each of these models is used to make predication on the complete dataset.
- The actual and predicted values are used to calculate the errors.. (models which give the error or predict incorrect results are weak learners)
- Incorrectly predicted instance gives higher weights and then these instance values give to the next model with new instance values.
- New Model will try to correct the error from the previous model.
- Then similarly, multiple models are created, each of the models corrects the previous model's errors.

- At the last final model which is (strong learner) is the weighted mean or averaging of all the models (weak learners).

4.10 Performance Measures:

To check how the developed model or experiment is performing, we need to use some performance measures to keep track of the model's performance.we have used some performance measures which are RMSE(root mean squared error), MSE(mean squared error), Adj-R2(Adjusted R-squared), MAPE(mean absolute percentage error) .

4.10.1 Mean square error (MSE)

The mean square error (MSE) is a measure of the error between the actual and predicted values of the same data. It is nothing but arithmetic average of square of errors.

$$MSE = \frac{\sum_{i=1}^{N}(y_i - x_i)}{N} \tag{1}$$

Where,

y_i : $Predicted\ value$

x_i : $Actual\ value$

N : $no.\ of\ sample$

4.10.2 Root mean square error (RMSE)

The root mean square error (RMSE) is a measure of the error between the actual and predicted values of the same data. It is nothing but arithmetic average of root of square of errors.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - x_i)}{N}} \quad (2)$$

Where,

y_i : *Predicted value*

x_i : *Actual value*

N : *no. of sample*

4.10.3 Adjusted R-squared (Adj-R2)

Adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases when the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected.

$$\overline{R}^2 = 1 - (1 - R^2)\left[\frac{n-1}{n-(k+1)}\right] \quad (3)$$

where,

k : number of features

n : number of samples

R : coefficient of determination of the prediction

4.10.4 Mean absolute percentage error (MAPE)

The mean absolute percentage error (MAPE) is the mean or average of the absolute percentage errors of forecasts. Error is defined as actual or observed value minus the forecasted value.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{A_i - P_i}{A_i}\right| \tag{4}$$

Where,

P_i : *Predicted value*

A_i : *Actual value*

N : *no. of sample*

4.11 Test the data and result

We have fitted the 20% of data on Desharnais[21] and Maxwell dataset[22] into various of models and trained the models using the training dataset. We have measured the performance of models SVM, Decision Tree , random forest, boosting and bagging Ensemble Model for predicting the Software development efforts by using several performance measures like Adj-R2, RMSE, MSE, and MAPE and results are shown in below table.

Model	MSE	RMSE	MAPE	Adj-R2
SVM	0.0067	0.0819	74.1125	0.8683
Decision Tree	0.0018	0.0428	21.3688	1.0
Random Forest	0.0002	0.0158	0.5465	0.020
Bagging	0.0094	0.0969	168.9803	0.7372
Boosting	0.0119	0.1091	7.9216	0.7206

Table 2: Performance measures On Desharnais dataset[21]

Model	MSE	RMSE	MAPE	Adj-R2
SVM	0.0118	0.1090	704.3764	0.7124
Decision Tree	0.0553	0.2352	109.8431	1.0
Random Forest	0.0054	0.0735	120.2785	0.0215
Bagging	0.02310	0.1519	162.921	0.7530
Boosting	0.014	3.098	0.99	0.028

Table 3: Performance measures On Maxwell dataset[22]

Comparison of actual values and predicted values plot on graph after applying different algorithms .

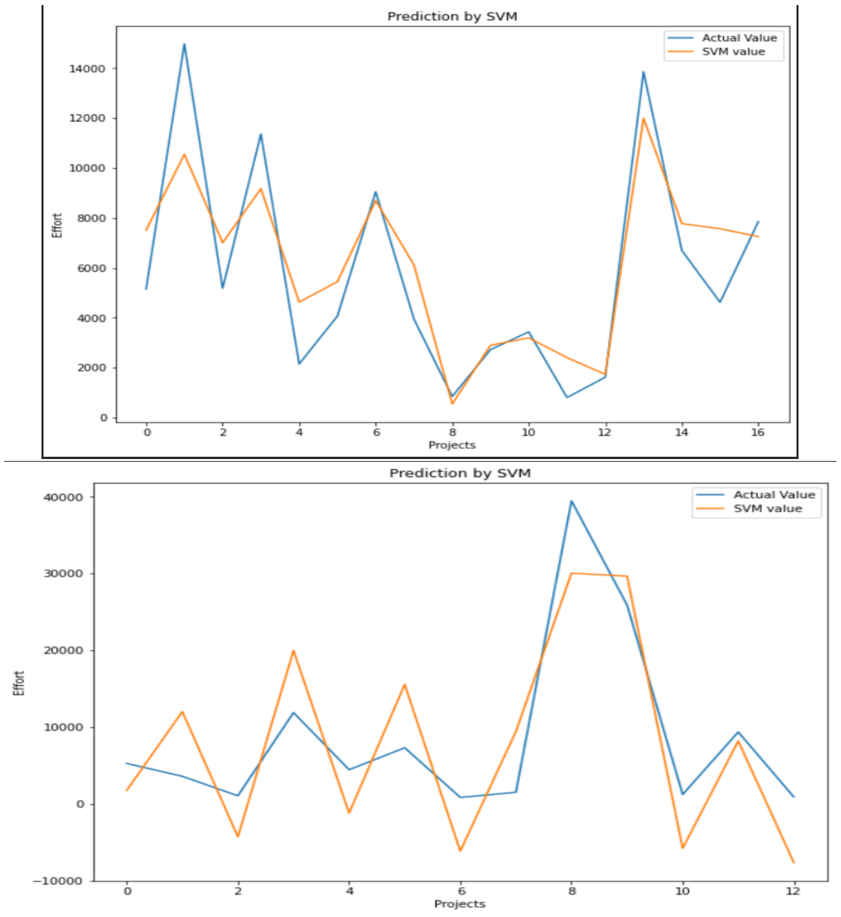
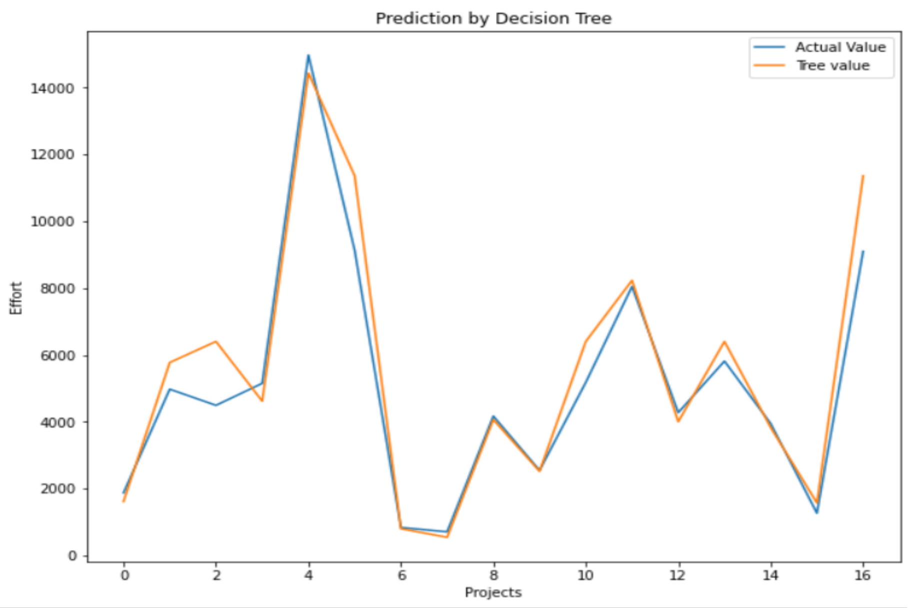


Figure 2: Support Vector Regressor(Graph)



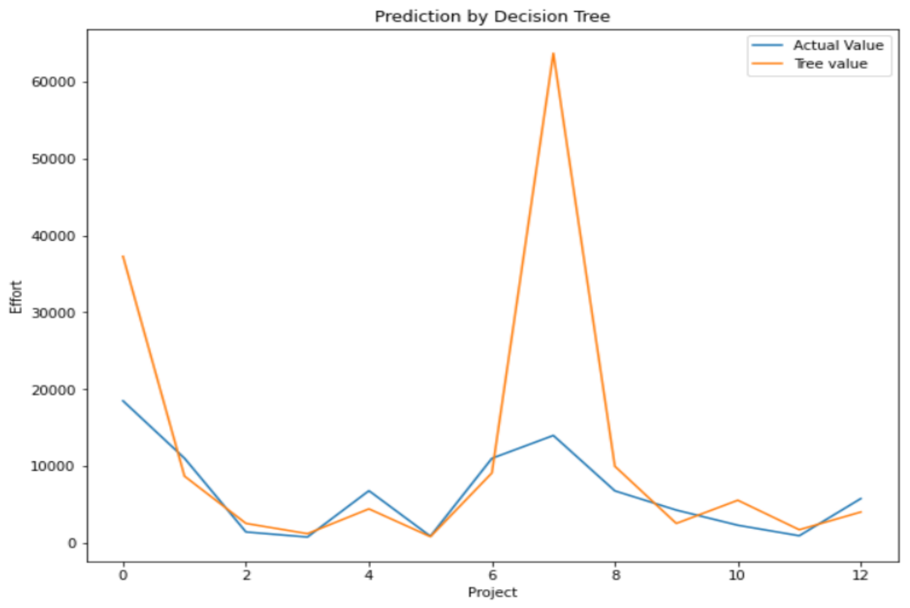


Figure 3: Decision Tree Regressor(Graph)

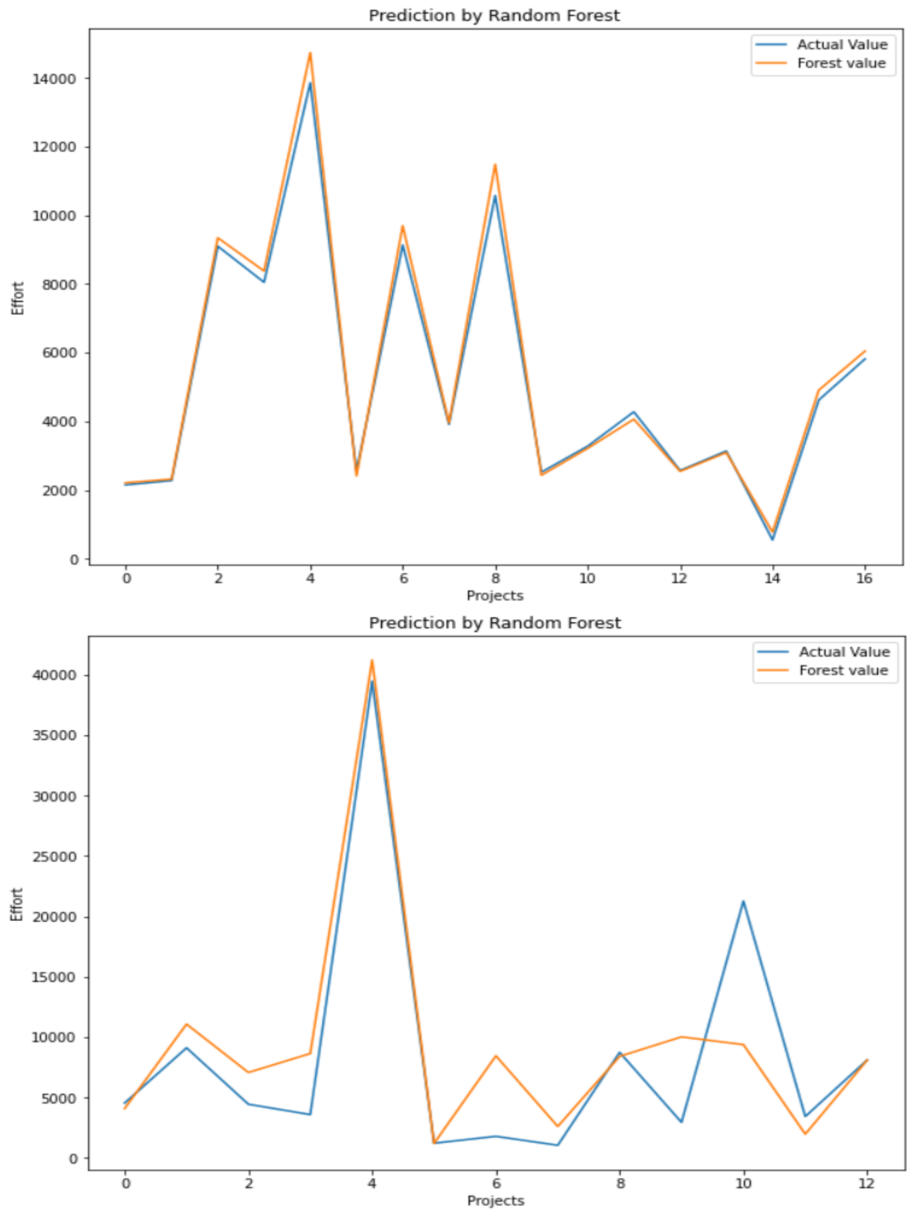
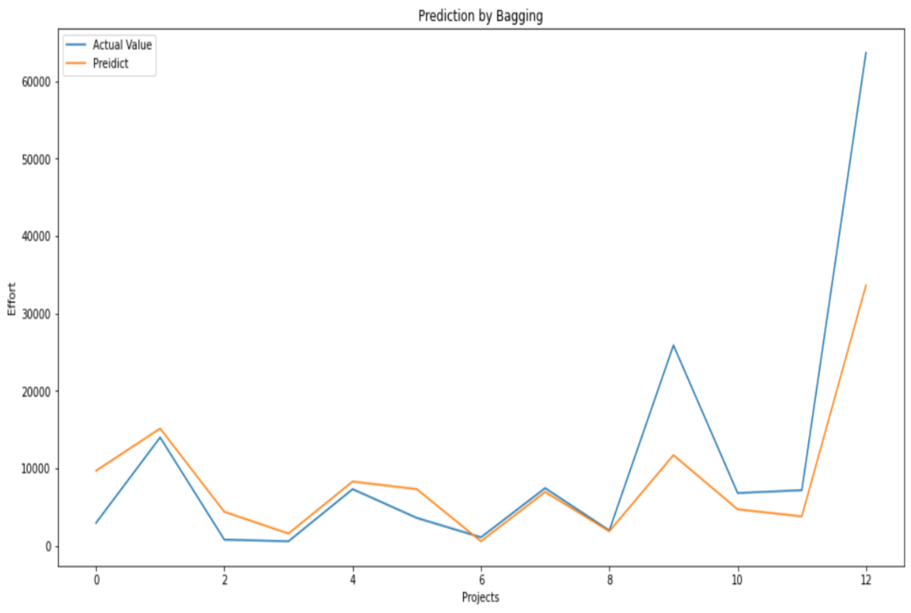


Figure 4: Random Forest Regressor(Graph)



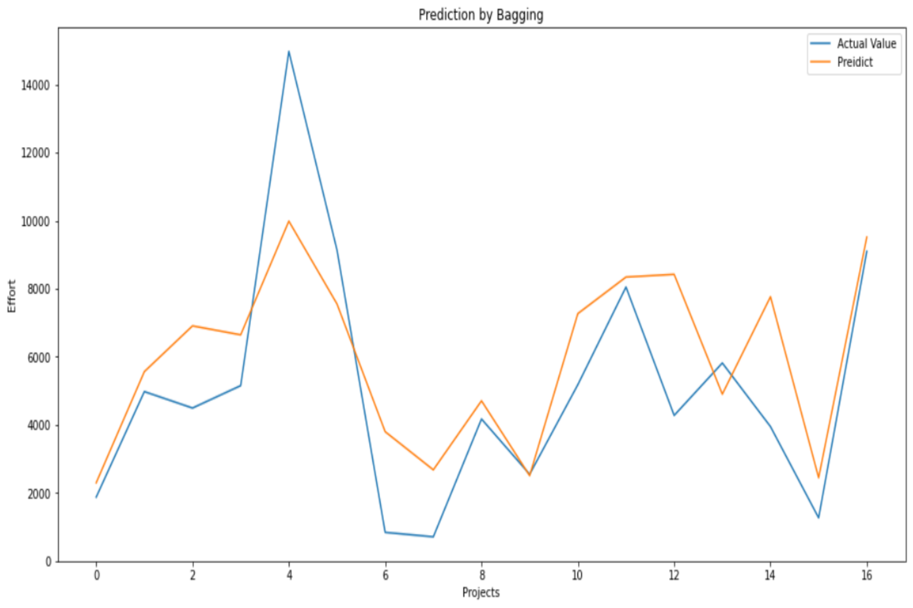
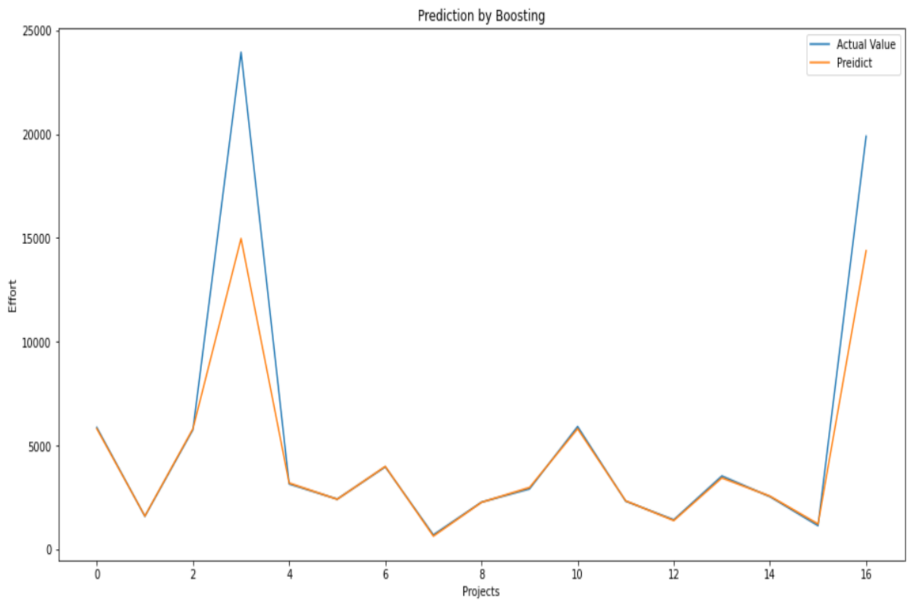


Figure 5: Bagging(Graph)



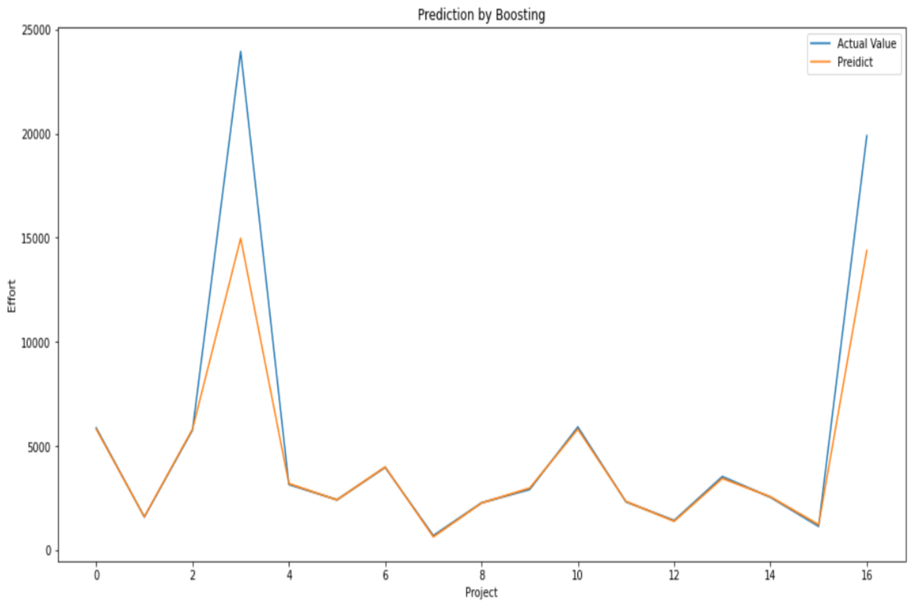


Figure 6: Boosting(Graph)

So we found that among these models Boosting Ensemble Model is performing better in predicting Software development efforts than other models . Seeing the graph we can also observe that actual and predict values lines are very close to each other in Boosting Ensemble algorithm .

4.12 Future Work

For Software Development Effort Prediction , many algorithms have been tested, some of them are mentioned above.And We Measures the performance of each algo. using **RMSE** and **MAPE** has lower values which indicates that it can predict Software project effort is more accurately. Comparison of those algorithms **gradient boosting algorithm** performed well .Our Next target is to apply remaining advance ensemble algorithms Stacking, Blending, AdaBoost, GBM, XGB, CatBoost to predict software development effort.

5 References:

5.1 Papers:

1. 21. Shepperd, M.J. and C. Schofield, Estimating Software Project Effort Using Analogies. IEEE Trans. Software Eng., 1997. 23: p. 736-743.

2. Usman, M., et al., S-26 Developing and using checklists to improve software effort estimation: A multi-case study. Journal of Systems and Software, 2018. 146: p. 286-309.

3. Omar Hidmi1, and Betul Erdogdu Sakar 2 ,Software Development Effort Estimation Using Ensemble Machine Learning,Int'l Journal of Computing, Communications Instrumentation Engg. (IJCCIE) Vol. 4, Issue 1 (2017) ISSN 2349-1469 EISSN 2349-1477.

4. Bardsiri, V.K., et al., LMES: A localized multi-estimator model to estimate software development effort. Engineering Applications of Artificial Intelligence, 2013. 26(10): p. 2624-2640.

5. Medina, A. and A.J. Francis, What are the characteristics that software development project team members associate with a good project manager? Project Management Journal, 2015. 46(5): p. 81-93.

6. Lehtinen, T.O.A., et al., Perceived causes of software project failures -An analysis of their relationships. Inf. Softw. Technol., 2014. 56(6):p.623-643.

7. Kocaguneli, E., et al., Exploiting the Essential Assumptions of

Analogy-Based Effort Estimation. IEEE Transactions on Software Engineering, 2012. 38(2): p. 425-438

8. Minku, L.L. and X. Yao, Ensembles and locality: Insight on improving software effort estimation. Information and Software Technology, 2013. 55(8): p. 1512-1528
9. Pai, D.R., K.S. McFall, and G.H. Subramanian, Software Effort Estimation Using a Neural Network Ensemble. Journal of Computer Information Systems, 2013. 53(4): p. 49-58
10. Yurdakurban, V. and N. ErdoĞan. S-7 Comparison of machine learning methods for software project effort estimation. in 2018 26th Signal Processing and Communications Applications Conference (SIU). 2018
11. Grimstad, S. and M. Jørgensen, S-31 Inconsistency of expert judgmentbased estimates of software development effort. Journal of Systems and Software, 2007. 80(11): p. 1770-1777.
12. Wu, D., J. Li, and C. Bao, Case-based reasoning with optimized weight derived by particle swarm optimization for software effort estimation. Soft Computing, 2018. 22(16): p. 5299-5310.
13. Jørgensen, M., S-30 A review of studies on expert estimation of software development effort. Journal of Systems and Software, 2004. 70(1-2): p. 37-60.
14. Jorgensen, M. and K. Moløkken-Østvold, A preliminary checklist for software cost management. 2003. 134-140.

15. Kocaguneli, E., T. Menzies, and J.W. Keung, On the value of ensemble effort estimation. IEEE Transactions on Software Engineering, 2012. 38(6): p. 1403-1416

16. Wohlin, C. and A. Aurum, Towards a decision-making structure for selecting a research design in empirical software engineering. Empirical Software Engineering, 2015. 20(6): p. 1427-1455.

17. Usman, M., et al., S-26 Developing and using checklists to improve software effort estimation: A multi-case study. Journal of Systems and Software, 2018. 146: p. 286-309.

18. Shepperd, M.J. and C. Schofield, Estimating Software Project Effort Using Analogies. IEEE Trans. Software Eng., 1997. 23: p. 736-743.

19. Desharnais dataset

    `https://www.kaggle.com/datasets/toniesteves/desharnais-dataset/`

20. Maxwell dataset

    `https://github.com/dr-bigfatnoob/effort/blob/master/datasets/csv/maxwell.arff.csv/`