IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 1, Ver. IX (Feb. 2014), PP 34-38
www.iosrjournals.org

Empirical Evaluation of machine learning techniques for software effort estimation

NehaSaini[1] ,Bushra Khalid[2]
1(Department of Software Engineering, Delhi Technological University,India)
2(Department of Software Engineering, Delhi Technological University, India)

Abstract: *Accurate estimation of software development effort is a very difficult job.Both under estimation as well as over estimation can lead to serious consequences. So its very important to find a technique which can yield accurate results for software effort estimation. Here in our paper we have evaluated various machine learning techniques for software effort estimation like bagging, decision trees, decision tables, multilayer perceptron and RBF networks. Two different datasets i.e. heiatheiat dataset and miyazaki94 dataset have been used in our research. Decision trees outperform all other models in term of MMRE value. Results of machine learning algorithms can be different from dataset to dataset.*

Keywords: *bagging, decision trees, decision table ,effort estimation, , multilayer perceptron, Radial basis function.*

I. Introduction

Software effort estimation is very important for the successful completion of any project.Moreover, the project is easy to maintain and control if software effort estimation is done in accurate way.There has been a lot of work done in the area of software effort estimation. Effort estimation of software projects can be done basically in 3 ways:

- Judgement based on experts: It calculates effort by measuring the degree of similarity our project carries with the past historical project .It is not so accurate as accuracy depends on similarity with past historical project.
- Algorithmic: They are based on some mathematical formula. They are not so accurate as they are unable model all the sets of relationship between attributes on which our project depends.
- Machine learning methods: Here effort estimation is based on applying various machine learning algorithms like neural networks, fuzzy, neuro-fuzzy, genetic algorithms. This is the best one as all set of relationships between effort and independent variables can be modeled using machine learning techniques.

In this paper we have done empirical evaluation of different machine learning techniques on miyazaki94 and heiatheiat dataset. Various methods being used in our research are bagging, decision trees, decision table, multilayer perceptron and radial basis function. These methods are the current trends in the area of software effort estimation. Their ability to model complex set set of relationships have lead to their popularity.

The research work carried out by me has been divided in to various sections .In section 2 the literature survey done for the research have been explained very briefly. In section 3 the research methodology and various machine learning techniques for effort estimation has been discussed. In section 4, whole research background has been explained. First of all the 2 datasets used in the research work have been explained. After that the method used for cross validation has been described. The tool used for calculation of results have been explained. In section 5, the results after application of different algorithms on the dataset have been explained. In section 6 final conclusion and future work have been explained. Finally all the references used in the research have been mentioned.

II. Related work

There has been a lot of work done in the area of software effort estimation. Albrecht et. al. [1] in his paper has measured effort by taking functional points in to account.Malhotraet. al. [7] in 2010 has proposed many models for estimating effort. Various methods used in the paper are least square regression, linear regression, MSP, M5 Rules, RBF, SVM, Pace regression and REP Tree. Malhotraet. al. [13] in 2011 compared various methods like Linear Regression, Artificial Neural Network, Decision Tree, Support Vector Machine and Bagging on a renowned software project dataset. The dataset used consisted of 499 projects. Performance measures being used in the project are MMRE, RMSE, RAE, RRSE, PRED(25). Oliveira et. al. has focused on the importance of neurocomputing for software effort estimation.[2]The paper by Finnie and Wittig [4][5], examined the potential of two artificial intelligence approaches i.e. artificial neural networks (ANN) and case

based reasoning (CBR) for creating development effort estimation models. They concluded that performance of CBR and ANN depends on the dataset being used. The paper by Elish[6] empirically evaluates the potential and accuracy of MART as a novel software effort estimation model when compared with recently published models, i.e. radial basis function (RBF) neural networks , linear regression , and support vector regression models with linear and RBF kernels. The comparison is based on a well-known and respected NASA software project dataset. Burgess and Lefley [3],evaluated the potential of genetic programming (GP) in software effort estimation and comparison was made with the Linear LSR, ANN etc. The results showed dependence on the fitness function used.Saxenaet. al. exploredNeuro-fuzzy techniques to design a suitable model to utilize improved estimation of software effort for NASA software projects[9]. Federica Sarroet. al. used search based techniques for software effort estimation. These approaches include a variety of meta-heuristics, such as local search techniques (e.g., Hill Climbing, Tabu Search, Simulated Annealing) or Evolutionary Algorithms (e.g., Genetic Algorithms, Genetic Programming)[10]. This approach was a suitable solutions to problems characterized by large search space. Ali BouNassif and Mohammad used a decision tree forest (DTF) model and compared it to a traditional decision tree (DT) model, as well as a multiple linear regression model (MLR) [11]. In the paper by Yeong-SeokSeo , the clusters are built on the basis of MRE values. Then a multiple LSR model is build on each of these clusters[12]. After this effort is calculated using each of these models for their respective clusters. Finally, the cumulative effort is calculated.

III. Research Methodology

In this paper, the modern machine learning techniques available in weka tool have been used. We have used Linear regression and some machine learning techniques like Decision tree, Radial basis network, Multilayer perceptron and Bagging methods that have been successfully applied in various areas.

3.1. Bagging

It was introduced by Breiman in 1996. The most important feature of bagging is that it combines multiple predictors. The machine learning algorithms which are used for statistical classification and regression are improved by bagging.It improves the algorithm by improving their stability and accuracy. Bagging reduces variance and avoids over fitting.

3.2. Decision trees and Decision tables

Decision tree is a kind of tool to come out with a decision on the basis of some conditions and their possible consequences. Decision trees can be thought of as a way to graphically represent some algorithm. Decision trees are mostly used in the area of research where we need to find out the best strategy or move to reach up to our required goal. Mostly decision trees are used in the field of classification and regression. In order to partition a particular dataset in to a particular class, decision trees make use of the concept of breadth first search and depth first search algorithms .Decision tables use same approach like decision trees but they make use of relations to make any decision.

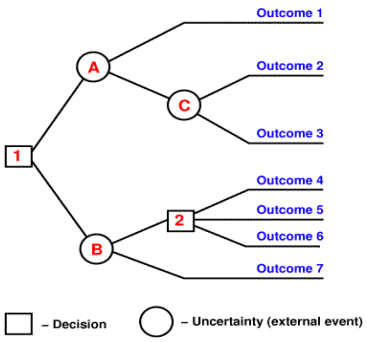

Figure 1: Example of decision tree

3.3. Multilayer Perceptron

Basically a multilayer perceptron is a feed forward model .Its function is to map the set of inputs to particular set of outputs. In MLP , as the name says we have multiple layers of neurons with each layer fully connected to other layers of network and the network is like a mesh like network. Each node is a neuron leaving the input nodes and each node has a non linear activation function corresponding to it. MLP utilizes a supervised learning technique called backpropagation for training the network [16][17]. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable[18].

3.4. Radial Basis Function Network

The theory of function approximation has given birth to the idea of RBF networks. We have already seen how MLP approximates functions above.The approach taken by RBF for function approximation differs slightly from MLP. Its main features are:

- RBF network make use of 2 layer feed forward networks.
- Radial basis functions like Gaussian function are implemented in the hidden nodes.
- Similar to MLP, here also output nodes implement linear summation activation functions.
- The training of RBF networks is done in two phases. Firstly the weights are determined from input to hidden layers and in the second phase the weights are determined from hidden to output layers.
- They can be trained very fast and their learning power is also very good.
- Interpolation power of RBF networks is also very good

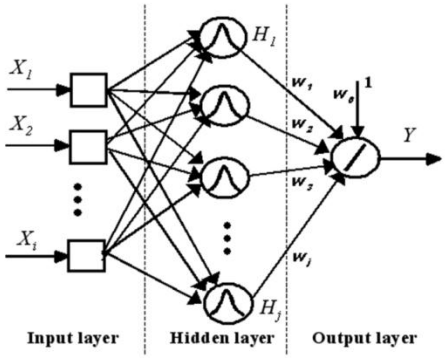

Figure 2: A Radial Basis Function Network

IV. Research Background

For our research we have used2 datasets namely HeiatHeiat Dataset and Miyazaki94 Dataset. Description of datasets is as follows:-

4.1. Dataset 1:Miyazaki94 Dataset: It consists of 9 attributes and 48 instances. Here 70% of the dataset is used for training and 30% of the dataset is used for validation It can be obtained from [14].

4.2. Dataset 2:HeiatHeiatDataset: It consists of two attributes i.e. size of project and effort. Here 70% of the dataset is used for training and 30% of the dataset is used for validating the data.It can be obtained from[15].

4.3. Estimation accuracy measures

In this paper, we have used the standard and most effective estimation accuracy measures i.e.Mean Magnitude of Relative Error (MMRE) and Prediction at level 0.25. 0.50 and 0.75 respectively. Previous studies done in this area havealso used these two measures. MMRE value for a dataset consisting of m observations can be calculated in following manner[6]:

$$MMRE = \frac{1}{m}\sum_{i=1}^{m} MRE_i$$

Where MRE_i is a normalized measure of the discrepancy between the actual value (x_i) and the estimated value (x_i) of observation i. It is calculated as follows[6]:

$$MRE_i = \frac{|X_i - \hat{X}_i|}{X_i}$$

$$Pred(25) = \ m/n$$

here m is the number of observations whose MRE is less than orequal to 0.25, and n is the total number of observations in a particular dataset[6].

$$Pred(50) = \ m/n$$

here m is the number of observations whose MRE is less than or equal to 0.50, and n is the total number of observations in a particulardataset[6].

$Pred(75) = \ m/n$

here m is the number of observations whose MRE is less than or equal to 0.75, and n is the total number of observations in a particulardataset[6].

4.4. Cross validation

10-cross validation method

In this method, 10 equal size subsamples are obtained by partitioning the original sample randomly. Among these 10 subsamples, 9 are used as training data and 1 is used for testing the model. This method is repeated 10 times and each of the 10 sub samples are used once as validation data for testing. The result obtained from these 10 repetitions is averaged to get single value. The main advantage of this method is that all the sub samples are used for both training and validation.

4.5. Tool used for result calculation Weka tool has been used in our research work. Weka tool has embedded machine learning algorithms and it provides automatic preprocessing of our data and make them more suitable as a input to our machine learning algorithms. Other prominent features provided by weka tool are:-----

- Classification
- Clustering
- Association rule extraction

V. Results and discussion

The results that were obtained after the application of different machine learning algorithms on the two datasets have been explained below with the help of tables and after that a detailed discussion about the results has been done.

5.1. Analysis using Miyazaki94 dataset

TECHNIQUE USED	MMRE	MRE	PRE(25)	PRED(50)	PRED(75)
Bagging	0.669	0.5618	0.333	0.663	0.666
Decision tree	0.0496	-0.0336	0.9967	1	1
Linear Regression	0.2352	0.149	0.722	0.888	0.888
Multilayer perceptron	0.151	0.0829	0.8765	0.909	1
Radial Basis Function	0.255	0.06034	0.4	0.765	1
Decision Table	0.0539	-0.035	0.989	1	1

Table1 : Results with Miyazaki94 dataset

5.1.1. Discussion of results with Miyazaki94 dataset

As can be seen from above table, decision tree model outperforms all other models in term of MMRE value. It is having the lowest MMRE value of 0.0436. The value MRE is also competitive with the decision table.The MRE value for decision tree is -0.036. In terms of PRED value also Decision tree outperforms all other models having the highest pred(25) value of 0.9967, pred(50) value of 1 and pred(75) value of 1.

After this multilayer perceptron also shows good performance. The worst performance has been shown by the bagging method with highest MMRE value of 0.691.The pred values were also not satisfactory being 0.333 at pred(25) and 0.663 and 0.669 at pred(50) and pred(75) respectively.

5.2. Analysis of results with HeiatHeiat dataset

TECHNIQUE USED	MMRE	MRE	PRED(25)	PRED(50)	PRED(75)
Bagging	.141259	.0231	.8	1	1
Decision tree	.1074	-0.01845	.909	1	1
Multilayer perceptron	.110865	0.006078	0.919	1	1
Radial Basis Function	0.140978	0.02323	0.909	1	1
Decision Table	.1260	-0.05959	.8181	1	1

Table 2: Results with HeiatHeiat dataset

5.2.1. Discussion of results with HeiatHeiat dataset

As can be seen from above table Decision tree outperforms all other models in term of MMRE value. It is having the lowest MMRE value of 0.1074. The MRE value for decision tree is -0.00019. In terms of PRED value multilayer perceptron outperforms all other models having the highest pred(25) value of 0.919, pred(50) value of 1 and pred(75) value of 1.

The multilayer perceptron also shows good performance on this dataset having MMRE value of 0.11065 and pred(25) value of 0.461, pred(50) of 1 and pred(75) of 1.
Among these 5 models bagging and radial basis function performs the worst on the basis of MMRE with value as high as 0.14 approx.

VI. Conclusion and Future Work

After analyzing the results with different datasets and using different machine learning algorithms on the datasets we can conclude different datasets show different results with different algorithms. The result that comes out depends on the type of data to a great extent. Also, we can deduce that decision trees have shown good performance for 2 datasets i.e. HeiatHeiat dataset and Miyazaki94 dataset. So, as per our research Decision trees are good for estimating the software effort. The MMRE value of decision trees for HeiatHeiat and Miyazaki94 dataset are 0.1074 and 0.0436 respectively.

In the future, we can perform this entire study for some industrial software. Also we can use evolutionary algorithms like genetic algorithm, particle swarm optimization and bacterial foraging on the same data and check out whetherthere is any improvement in performance or not. We can also check out whether our project is economically feasible or not by estimating the cost based on predicted effort.

References

[1] A. Albert and J.E. Gaffney, "*Software Function Source Lines of Code and Development Effort Prediction: A Software Science Validation*", IEEE Trans. Software Engineering, *vol. 9*, pp. 639-648, 1983.

[2] A.L.I. Oliveira*,"Estimation of software effort with support vector regression Neurocomputing", vol. 69*, pp. 1749-1753, Aug. 2006.

[3] C.J. Burgess and M.Lefley, *"Can genetic programming improve software effort estimation? A comparative evaluation, Information and Software Technology", vol. 43*, pp. 863-873, 2001.

[4] G.R. Finnie and G.E. Wittig, "*A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models*",Journal of Systems and Software, *vol. 39*, pp. 281-289, 1997.

[5] G.R. Finnie and G.E. Wittig, "*AI Tools for Software Development Effort Estimation*", in Proc. SEEP '96 , 1996, International Conference on Software Engineering: Education and Practice (SE:EP '96).

[6] M.O. Elish,"*Improved estimation of software project effort using multiple additive regression trees*",Expert Systems with Applications, *vol. 36*, pp. 10774-10778, 2009.

[7] R.Malhotra, A.Kaur and Y.singh,"*Application of Machine Learning Methods for Software Effort Prediction*",in Newsletter ACM SIGSOFT Software Engineering Notes , *vol. 35*, May 2010.

[8] Weka. Available: http://www.cs.waikato.ac.nz/ml/weka/

[9]U rvashi Rahul Saxena, S. P. Singh, "*Software Effort Estimation Using Neuro-Fuzzy Approach*", CSI 6th International Conference, pp. 1-6, Sept. 2012.

[10] "*Search-Based Approaches for Software Development Effort Estimation*" Federica Sarro University of Salerno.

[11] "*A comparison between decision trees and decision tree forest models for software development effort estimation",*by Ali BouNassif, Mohammad Azzeh, Luiz Fernando Capretz, Danny Hoin proceeding of 2013 Third International Conference on Communications and Information Technology (ICCIT), At Beirut, Lebanon.

[12] ***"****Improving the Accuracy of Software Effort Estimation Based on Multiple Least Square Regression Models by Estimation Error-Based Data Partitioning ", by* Yeong-seokSeo, Kyung-a Yoon, Doo-hwanBae in Asia-Pacific Software Engineering Conference - APSEC , pp. 3-10, 2009.

[13] R.MalhotraA.Jain, "*Software Effort Prediction using Statistical and Machine Learning Methods*",in International Journal of Advanced Computer Science and Applications, *Vol. 2*, No.1, January 2011 .

[14] http://softwareengineeringwiki.wikispaces.com/file/links/miyazaki94.arff

[15] https://code.google.com/p/open-parametrics/source/browse/open-parametrics/datasets/heiat.arff

[16] Rosenblatt, Frank. x. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC, 1961

[17] Rumelhart, David E., Geoffrey E. Hinton, and R. J. Williams " *Learning Internal Representations by Error Propagation*". David E. Rumelhart, James L. McClelland, and the PDP research group. (editors), Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations. MIT Press, 1986.

[18] Cybenko, G. 1989 "*Approximation by superpositions of a sigmoidal function Mathematics of Control, Signals, and Systems*"2(4), 303–314.