Comparison and Analysis of Different Software Cost Estimation Methods

Sweta Kumari
Computer Science & Engineering
Birla Institute of Technology
Ranchi India

Shashank Pushkar
Computer Science &Engineering
Birla Institute of Technology
Ranchi India

***Abstract*- Software cost estimation is the process of predicting the effort required to develop a software system. The basic input for the software cost estimation is coding size and set of cost drivers, the output is Effort in terms of Person-Months (PM's). Here, the use of support vector regression (SVR) has been proposed for the estimation of software project effort. We have used the COCOMO dataset and our results are compared to Intermediate COCOMO as well as to MOPSO model results for this dataset. It has been observed from the simulation that SVR outperforms other estimating techniques. This paper provides a comparative study on support vector regression (SVR), Intermediate COCOMO and Multiple Objective Particle Swarm Optimization (MOPSO) model for estimation of software project effort.**

We have analyzed in terms of accuracy and Error rate. Here, data mining tool Weka is used for simulation.

Keywords--- Support vector regression; PM- person-months; MOPSO- Multiple objective particle swarm optimization; COCOMO- Constructive cost estimation; Weka data mining tools.

I. INTRODUCTION

Cost estimation is a process or an approximation of the probable cost of a product, program, or a project, computed on the basis of available information. Accurate cost estimation is very important for every kind of project, if we do not estimate the projects in a proper way; result the cost of the project is very high sometimes it will be reached 150-200% more than the original cost [19]. So in that case it is very necessary to estimate the project correctly. The Cost for a project is a function of many parameters. Size is a primary cost factor in most models and can be measured using lines of code (LOC) or thousands of delivered lines of code (KDLOC) or function points. A number of models have been evolved to establish the relation between size and effort for Software Cost Estimation. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Data mining help us to classify the past project data and generate the valuable information.

Support vector regression (SVR) is a kernel method for regression based on the principle of structural risk minimization [11, 3]. Kernel methods have outperformed more traditional techniques in a number of problems, including classification and regression [11, 3]. Here, the use of SVR has been proposed for the estimation of software project cost and also, it has been found that this technique outperforms the other popular cost estimation procedures in terms of accuracy. The rest of the paper is organized as follows: Literature review refers to some existing estimation methods. Then the basic idea for this new approach for estimation has been discussed. Then the simulated experiment has been mention. We discuss the results and give the concluding remarks.

II. LITERATURE REVIEW

Various effort estimation models have been developed over the last four decades. The most commonly used methods for predicting software development efforts are function Point Analysis and Constructive Cost Model (COCOMO) [10]. Function point analysis is a method of quantifying the size and complexity of a software system in terms of the functions that the system delivers to the user [4]. The function does not depend on the programming languages or tools used to develop a software project [3]. COCOMO is developed by the Boehm [2]. It is based on linear-least-squares regression. Using line of code (LOC) as the unit of measure for software size itself contains so many problems [7]. These methods failed to deal with the implicit non-linearity and interactions between the characteristics of the project and effort [5, 11].

In recent years, a number of alternative modelling techniques have been proposed. They include artificial neural networks, analogy-based reasoning, and fuzzy system and ensemble techniques. Ensemble is used to combine the result of individual methods [12, 17]. In analogy-based cost estimation, similarity measures between a pair of projects play a critical role [16]. This type of model calculates distance between the software project being estimated and each of the historical software projects and then retrieves the most similar project for generating an effort estimate [14]. Further, Lefley and Shepperd [9] applied genetic programming to improve software cost estimation on public datasets with great success. Later, Vinay kumar et al. [15] used wavelet neural networks for the prediction of software cost estimation. Unfortunately the accuracy of these models is not satisfactory so there is always a scope for more accurate software cost estimation techniques.

III. THE BASIC IDEA

Suppose we are given training dataset$\{(x_1, y_1), \ldots, (x_l, y_l)\} \subset \chi \times \mathbb{R}$, where χ denotes the space of the input patterns (e.g. $\chi = \mathbb{R}^d$). The goal of regression is to find the function $f(x)$ that best models the training data. In our case, we are interested in building a regression model based on the training

data to use it subsequently to predict the total effort in man-months of future software projects. In linear regression, this is done by finding the line that minimizes the sum of squares error on the training set.

A. Support Vector Regression

In this work we propose to use ε-SVR, which defines the ε-insensitive loss function. This type of loss function defines a band around the true outputs sometimes referred to as a tube, as shown in Fig. 1.

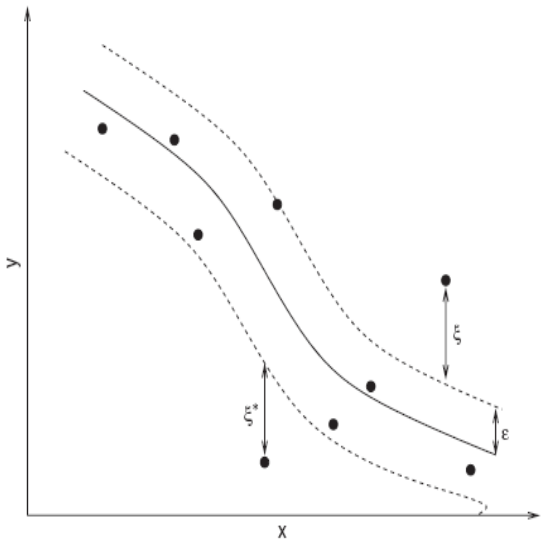

Fig.1 Regression using ε-SVR

The idea is that errors smaller than a certain threshold $\varepsilon > 0$ are ignored. That is, errors inside the band are considered to be zero. On the other hand, errors caused by points outside the band are measured by variables ξ and ξ* as shown in Fig. 1.

In the case of SVR for linear regression, $f(x)$ is given $f(x) = \langle w, x \rangle + b$, with $w \in \chi$, $b \in \mathbb{R}$. $\langle .,. \rangle$ denotes the dot product. For the case of nonlinear regression, $f(x) = \langle w, \phi(x) \rangle + b$, where ϕ is some nonlinear function which maps the input space to a higher (maybe infinite) dimensional feature space. In ε-SVR, the weight vector *w* and the threshold *b* are chosen to optimize the following problem [11]:

$$\text{minimize}_{w,b,\xi,\xi^*} \quad \frac{1}{2}\langle w, w \rangle + C\sum_{i=1}^{l}(\xi i + \xi i^*),$$

$$\text{subject to} \begin{cases} (\langle w, \phi(x_i) \rangle + b) - y_i \le \varepsilon + \xi i, \\ y_i - (\langle w, \phi(x_i) \rangle + b) \le \varepsilon + \xi i^*, \\ \xi i, \ \xi i^* \ge 0 \end{cases} \quad \text{.....(1)}$$

The constant C>0 determines the trade-off between the flatness of *f* and the amount up to which deviations larger than ε are tolerated. ξ and ξ* are called slack variables and measure the cost of the errors on the training points. ξ measures deviations exceeding the target value by more than ε and ξ* measures deviations which are more than ε below the target value, as shown in Fig. 1.

The idea of SVR is to minimize an objective function which considers both the norm of the weight vector *w* and the losses measured by the slack variables (see Eq. (1)). The minimization of the norm of *w* is one of the ways to ensure the flatness of *f* [11].

The SVR algorithm involves the use of Lagrangian multipliers, which rely solely on dot products of $\phi(x)$. This can be accomplished via kernel functions, defined as $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. Thus, the method avoids computing the transformation $\phi(x)$ explicitly. The details of the solution can be found in [11].

IV. Experiments

The regression methods considered in this paper were compared using the well-known COCOMO software project dataset, reproduced in Table I .This dataset consists of two independent variables-Size and EAF (Effort Adjustment Factor) and one dependent variable-Effort. Size is in KLOC (thousands of lines of codes) and effort is given in man-months [1].In this work we are interested in estimating the effort of future projects, where the effort is given in man-months. The simulations were carried out using the Weka tool [13]. In Weka, SVR is implemented using the Sequential Minimal Optimization (SMO) algorithm [6].

TABLE I. COCOMO Dataset.

Project No.	Size	EAF	Effort
1	46	1.17	240
2	16	0.66	33
3	4	2.22	43
4	6.9	0.4	8
5	22	7.62	107
6	30	2.39	423
7	18	2.38	321
8	20	2.38	218
9	37	1.12	201
10	24	0.85	79
11	3	5.86	73
12	3.9	3.63	61
13	3.7	2.81	40
14	1.9	1.78	9
15	75	0.89	539
16	90	0.7	453
17	38	1.95	523
18	48	1.16	387
19	9.4	2.04	88
20	13	2.81	98
21	2.14	1	7.3

The following section describes the experimentation part of work, and in order to conduct the study and to establish the affectivity of the models from COCOMO dataset were used. We calculated an

Intermediate COCOMO effort by using the following equations:

$$\text{Effort} = a*(size)^b * EAF \quad (2)$$

where a and b are the set of values depending on the complexity of software (for organic projects a=3.2, b=1.05, for semi-detached a=3.0, b=1.12 and for embedded a=2.8, b=1.2) and the MOPSO model effort[18]is calculated by using following equations:

$$\text{Effort} = a*(size)^b * EAF + C \quad (3)$$

where a and b are cost parameters and c is bias factor. a=3.96, b=1.12 and c=5.42.The performance measures considered in our work are Mean Absolute Relative Error (MARE) and Prediction (25). The MARE is given by the following equation:

$$\text{MARE} = \frac{1}{n} \sum_{i=1}^{n} |fi - yi| \quad (4)$$

Pred (25) is defined as the percentage of predictions falling within 25% of the actual known value, Pred (25). *fi* is the Estimated and *yi* is the Actual value respectively, *n* is the number of data points.

We have carried out simulations considering estimating the SVR effort using both independent variables (Size and EAF). The results of our simulations are shown in Table II.

TABLE II. ESTIMATED EFFORTS OF DIFFERENT TYPES OF MODELS

Project No.	Size	EAF	Measured Effort	COCOMO Effort	MOPSO Effort	SVR Effort	COCOMO Error	MOPSO Error	SVR Error
1	46	1.17	240	208.56	342.84	239.66	31.44	102.84	0.34
2	16	0.66	33	38.82	63.74	88.51	5.82	30.74	55.51
3	4	2.22	43	30.45	46.95	42.32	12.55	3.95	0.68
4	6.9	0.4	8	9.73	19.2	43.21	1.73	11.2	35.21
5	22	7.62	107	626.11	967.39	174.9	519.11	860.39	67.9
6	30	2.39	423	271.97	432.46	174.31	151.03	9.46	248.69
7	18	2.38	321	158.41	245.41	115.85	162.59	75.59	205.15
8	20	2.38	218	176.93	275.46	126.52	41.07	57.46	91.48
9	37	1.12	201	158.85	258.53	201.34	42.15	57.53	0.34
10	24	0.85	79	76.52	123.71	135.91	2.48	44.71	56.91
11	3	5.86	73	59.43	84.85	72.27	13.57	11.85	0.73
12	3.9	3.63	61	48.49	71.43	55.16	12.51	10.43	5.84
13	3.7	2.81	40	35.52	53.59	53.51	4.48	13.59	13.51
14	1.9	1.78	9	11.17	19.88	37.39	2.17	10.88	28.39
15	75	0.89	539	336.18	449.18	391.82	202.82	89.82	147.18
16	90	0.7	453	324.32	433.52	465.13	128.68	19.48	12.13
17	38	1.95	523	284.42	459.45	219.21	238.58	63.55	303.79
18	48	1.16	387	216.23	356.28	263.17	170.77	30.72	123.83
19	9.4	2.04	88	68.64	104.78	80.45	19.36	16.78	7.55
20	13	2.81	98	132.89	202.22	105.03	34.89	104.22	7.03
21	2.14	1	7.3	7.12	14.71	38.13	0.18	7.41	30.83

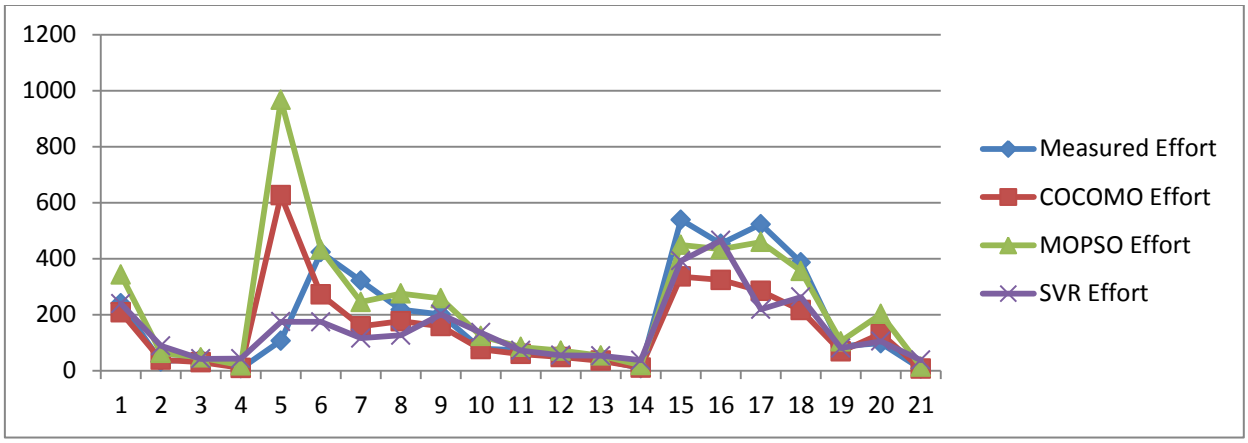

Fig.2: Measured Effort Vs Estimated Effort of various Models.

Figure 2 shows the graph of measured effort versus estimated effort of Intermediate COCOMO, MOPSO and SVR model.

From the figure 2, one can notice that the SVR estimated efforts are very close to the measured effort.

V. Results And Discussions

The results are tabulated in Table III. It was observed that the SVR gives better results in comparison with Intermediate COCOMO and MOPSO model. The MARE and Prediction accuracy is good. These results suggest that using data mining and machine learning techniques into existing software cost estimation techniques can effectively improve the accuracy of models.

TABLE III: PERFORMANCE AND COMPARISONS

Results	Intermediate COCOMO	MOPSO	SVR
MARE	85.62	77.74	68.72
Prediction (25%)	38.09	42.86	47.62

The following figure 3 shows the performance measures of Intermediate COCOMO, MOPSO and SVR model.

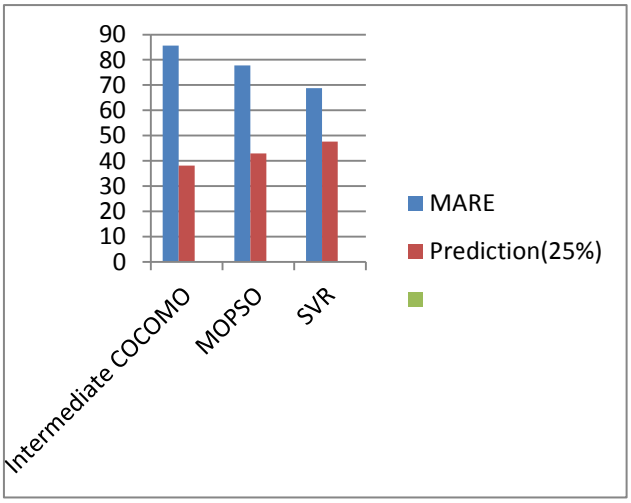

Fig.3. Performance Measure

VI. Concluding Remarks

This paper provides the use of Support Vector Regression for estimation of software project effort. We have carried out simulations using the COCOMO dataset. We have used weka tools for simulations because it consist of different-different machine learning algorithms that can be help us to classify the data easily.

The results were compared to both Intermediate COCOMO and MOPSO models. The accuracy of the model is measured in terms of its error rate. It is observed from the results that SVR gives better results. On testing the performance of the model in terms of the MARE and Prediction the results were found to be useful. The future work is the need to investigate some more data mining algorithms that can be help to improve the process of software cost estimation and easy to use.

ACKNOWLEDGMENT

The author would like to thank the anonymous referees for their helpful comments and suggestions.

REFERENCES

[1] J.W. Bailey, V.R. Basili, A meta model for software development resource expenditure, in: Proceedings of the Fifth International Conference on Software Engineering, San Diego, California, USA, 1981, pp. 107–116.

[2] B.W.Boehm, "Software Engineering Economics," Prentice- Hall, Englewood Cliffs, NJ, USA, 1981.

[3] A.J. Albrecht and J.E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation," IEEE Transactions on Software Engineering, 1983, pp. 639–647.

[4] J.E. Matson, B.E Barrett and J.M. Mellichamp, "Software development cost estimation using function points," IEEE Transactions on Software Engineering, 1994, pp. 275–287.

[5] A.R. Gray,"A simulation-based comparison of empirical modelling techniques for software metric models of development effort," In: Proceedings of ICONIP, Sixth International Conference on Neural Information Processing, Perth, WA, Australia, 1999, pp. 526–531.

[6] G.W. Flake, S. Lawrence, Efficient SVM regression training with SMO, Mach. Learn. 46 (1–3) (2002) 271–290.

[7] A.Idri, T.M. Khosgoftaar and A. Abran, "Can neural networks be easily interpreted in software cost estimation," World Congress on Computational Intelligence, Honolulu, Hawaii, USA, 2002, pp. 12–17.

[8] X.Huang, L.F.Capetz,J. Ren and D.Ho, "A neuro-fuzzy model for software cost estimation," Proceedings of the third International Conference on Quality Software, 2003, pp. 126-133 .

[9] M. Lefley and M. J. Shepperd, "Using Genetic Programming to Improve Software Effort Estimation Based on General Data Sets", LNCS, Genetic and Evolutionary Computation — GECCO 2003, ISBN: 978-3-540-40603-7, page-208.

[10] B. Kitchenham, L.M. Pickard, S. Linkman and P.W. Jones, "Modelling software bidding risks," IEEE Transactions on Software Engineering, 2003, pp. 542–554.

[11] A.J. Smola, B. Scholkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222.

[12] K.K. Aggarwal, Y. Singh, P.Chandra and M.Puri, "An expert committee model to estimate line of code," ACM New York, NY, USA, 2005, pp. 1-4.

[13] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufmann, San Francisco, 2005.

[14] N.H. Chiu and S.J.Huang, "The adjusted analogy-based software effort estimation based on similarity distances," System and Software, 2007, pp.628-640.

[15] K. Vinaykumar, V. Ravi, M. Carr and N. Rajkiran, "Software cost estimation using wavelet neural networks," Journal of Systems and Software, 2008, pp. 1853-1867.

[16] Y.F. Li, M. Xie and T.N. Goh, "A study of project selection and feature weighting for analogy based software cost estimation," Journal of Systems and Software, 2009, pp. 241–252.

[17] K. Vinay Kumar, V. Ravi and Mahil Carr, "Software Cost Estimation using Soft Computing Approaches," Handbook on Machine Learning Applications and Trends: Algorithms, Methods and Techniques, Eds. E. Soria, J.D. Martin, R. Magdalena, M.Martinez, A.J. Serrano, IGI Global, USA, 2009.

[18] Prasad Reddy P.V.G.D, Hari CH.V.M.K and Srinivasa Rao, "Multi Objective Particle Swarm Optimization for Software Cost Estimation," International Journal of Computer Applications, 2011, Vol.-32.

[19] Narendra Sharma and Ratnesh Litoriya, "Incorporating Data Mining Techniques on Software Cost Estimation: Validation and Improvement, " International Journal of Emerging Technology and Advanced Engineering, 2012, vol.-2