Study the Performance of M5-Rules Algorithm and Decision Table Majority Classifier for Modeling of Effort Estimation of Software Projects

Heetika Duggal, Parminder Singh

Abstract—Software effort estimation is the process of predicting the most realistic use of effort required to develop or maintain software based on incomplete, uncertain and/or noisy input. Effort estimates may be used as input to project plans, iteration plans, budgets etc. In this Study, the performance of single conjunctive rule learner, M5-Rules Algorithm and decision table majority classifier is compared for Modeling of Effort Estimation of Software Projects. The performance of the developed models were tested on 93 NASA projects from different centers for projects from 1971-1987.

Keywords—Software cost estimation, Effort estimation, Rules Generation, conjunctive rule learner, decision table, M5-learner.

I. INTRODUCTION

In the last three decades, many quantitative software cost estimation models have been developed. They range from empirical models such as Boehm’s COCOMO models [1] to analytical models such as those in [2, 3, 4]. An empirical model uses data from previous projects to evaluate the current project and derives the basic formulae from analysis of the particular database available. A good software cost estimate should be conceived and supported by the project manager and the development team. It is accepted by all stakeholders as realizable. It is based on a well-defined software cost model with a credible basis. It is based on a database of relevant project experience and it should be defined in enough detail so that its key risk areas are understood and the probability of success is objectively assessed [5].

Typical major models that are being used as benchmarks for software effort estimation are:

- Halstead,
- Walston-Felix
- Bailey-Basili
- Doty (for KLOC > 9).

These models have been derived by studying large number of completed software projects from various organizations and applications to explore how project sizes mapped into project effort. But still these models are not able to predict the Effort Estimation accurately.

In this Study, the performance of single conjunctive rule learner, M5-Rules Algorithm and decision table majority classifier is compared for Modeling of Effort Estimation of Software Projects. The dataset is based on the cost factors in COCOMO II.

A great deal of research over the past several years has studied the performance of M5-Rules Algorithm and Decision Table Majority Classifier.

The remainder of this paper can be described as follows:

Section II outlines the COCOMO Model that is used for the effort and cost estimation. Section III discusses the methodology adopted for generating and comparing a number of models. Section IV highlights results of implementation. It discusses the results of the various models used for the effort estimation and Section V is all about conclusions of this research work.

II. COCOMO MODEL

The COCOMO software cost model measures effort in calendar months of 152 hours (and includes development and management hours). COCOMO assumes that the effort grows more than linearly on software size; i.e. months = a*KLOC^b*c. Here, "a" and "b" are domain-specific parameters; "KLOC" is estimated directly or computed from a function point analysis; and "c" is the product of over a dozen "effort multipliers" i.e. months = a*(KLOC^b)*(EM1* EM2 * EM3 * ...). In COCOMO I, the exponent on KLOC was a single value ranging from 1.05 to 1.2. In COCOMO II, the exponent "b" was divided into a constant, plus the sum of five "scale factors" which modeled issues such as "have we built this kind of system before?". The COCOMO II effort multipliers are similar but COCOMO-II dropped one of the effort multiplier parameters; renamed some others; and added a few more (for "required level of reuse", "multiple-site development", and "schedule pressure"). The effort multipliers fall into three groups: those that are positively correlated to more effort; those that are negatively correlated to more effort; and a third group containing just schedule information. In COCOMO-I, "sced" has a U-shaped correlation to effort; i.e. giving programmers either too much or too little time to develop a system can be detrimental. The actual development effort is...
expressed in months (one month = 152 hours and includes development and management hours). The cost factors are shown in Table I.

<table>
<thead>
<tr>
<th>Cost Factors</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RELY</td>
<td>required software reliability</td>
</tr>
<tr>
<td>DATA</td>
<td>database size</td>
</tr>
<tr>
<td>CPLX</td>
<td>product complexity</td>
</tr>
<tr>
<td>TIME</td>
<td>execution time constraint</td>
</tr>
<tr>
<td>STOR</td>
<td>main storage constraint</td>
</tr>
<tr>
<td>VIRT</td>
<td>virtual machine volatility</td>
</tr>
<tr>
<td>TURN</td>
<td>computer turnaround time</td>
</tr>
<tr>
<td>ACAP</td>
<td>analyst capability</td>
</tr>
<tr>
<td>AEXP</td>
<td>application experience</td>
</tr>
<tr>
<td>PCAP</td>
<td>programmer capability</td>
</tr>
<tr>
<td>VEXP</td>
<td>virtual machine experience</td>
</tr>
<tr>
<td>LEXP</td>
<td>language experience</td>
</tr>
<tr>
<td>MODP</td>
<td>modern programming practice</td>
</tr>
<tr>
<td>TOOL</td>
<td>software tools</td>
</tr>
<tr>
<td>SCED</td>
<td>development schedule</td>
</tr>
</tbody>
</table>

### Table I
THE COST FACTORS IN COCOMO II

- **Halstead Model**

- **Walston-Felix Model**

- **Bailey-Basili Model**

- **Doty Model**

### III. METHODOLOGY PROPOSED

The following steps are proposed for the meeting the above said objectives:

A. **Data Collection**

First, a Survey of the existing Models of Effort Estimation is to be performed and Secondly, Historical Data being used by various existing models for the cost estimation is collected.

B. **Develop the different Statistical Models of the Effort Estimation**

The following modeling approaches are used for effort dataset:

- Halstead Model
- Walston-Felix Model
- Bailey-Basili Model
- Doty Model
- Single Conjunctive Rule Learner,
- M5-Rules Algorithm

- Decision Table Majority Classifier

In addition to single conjunctive rule learner, M5-Rules Algorithm and decision table majority classifier the different existing models: Halstead Models, Walston-Felix Model, Bailey-Basili Model and Doty Model are also used for the comparison of results. The equations for the models are as under:

\[
\text{Halstead Effort} = 5.2(KLOC)^{1.50} \\
\text{Walston-Felix Effort} = 0.7(KLOC)^{0.91} \\
\text{Bailey-Basili Effort} = 5.5 + 0.73(KLOC)^{1.16} \\
\text{Doty (for KLOC > 9) Effort} = 5.288(KLOC)^{1.047}
\]

C. **Comparison of Models**

The comparison of the results is made on the basis of the following criteria:

i) **Mean absolute error**: Mean absolute error, MAE, is the average of the difference between predicted and actual value in all test cases; it is the average prediction error (Challagulla, 2005). The formula for calculating MAE is given in equation shown below:

\[
\frac{1}{n} \sum_{i=1}^{n} |a_i - c_i|
\]

Assuming that the actual output is a, expected output is c.

ii) **Root Mean-Squared Error**

RMSE is frequently used measure of differences between values predicted by a model or estimator and the values actually observed from the thing being modeled or estimated (Challagulla, 2005). It is just the square root of the mean square error as shown in equation given below:

\[
\sqrt{\frac{1}{n} \sum_{i=1}^{n} (a_i - c_i)^2}
\]

The mean-squared error is one of the most commonly used measures of success for numeric prediction. This value is computed by taking the average of the squared differences between each computed value and its corresponding correct value. The root mean-squared error is simply the square root of the mean-squared-error. The root mean-squared error gives the error value the same dimensionality as the actual and predicted values.
The mean absolute error and root mean squared error is calculated for each machine learning algorithm.

D. Conclusions

The conclusions are made on the basis of the comparison made in the previous section.

IV. Results & Discussion

The publicly available PROMISE Software Engineering Repository data set is used for the experimentation. It consists of 93 instances each with 23 input attributes and one output attribute named as effort. The single conjunctive rule learner, M5-Rules Algorithm and decision table majority classifier are run in the WEKA environment and formulae of the other models is calculated in the MATLAB environment. The performance of the various algorithms measured in RMSE and MAE values is shown in Table II.

<table>
<thead>
<tr>
<th>Models Used</th>
<th>Conjunctive Rule</th>
<th>Decision Table</th>
<th>M5Rules</th>
<th>Halstead Model</th>
<th>Halstead-Felix Model</th>
<th>Bailey-Basili Model</th>
<th>Doty Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAE</td>
<td>695.31</td>
<td>536.26</td>
<td>377.35</td>
<td>6814</td>
<td>583</td>
<td>472.2</td>
<td>416.99</td>
</tr>
<tr>
<td>RMSE</td>
<td>1246.63</td>
<td>1127.37</td>
<td>801.09</td>
<td>18963</td>
<td>1097.2</td>
<td>954.37</td>
<td></td>
</tr>
</tbody>
</table>

When the M5Rules algorithm is run the following is the snapshot of the rule generated by the algorithm:

```
act_effort = 393.1026 * node=embedded + 361.0229 * data=vl,xh,h + 1703.6997 * time=xh + 3.9232 * equivphyskloc - 45.7682 [93/59.9614]
```

V. Conclusion

In this study, single conjunctive rule learner, M5-Rules Algorithm and decision table majority classifier are experimented to estimate the software effort for projects. The performances of the developed models were tested on NASA software project data presented in [1] and results are compared with the Halstead, Walston-Felix, Bailey-Basili, Doty and GA Based models mentioned in the literature. The proposed M5Rule based rule generation model is able to provide good estimation capabilities as compared to other models in the study. It is suggested to use of M5Rule technique to build suitable model structure for the software effort.

REFERENCES