ORIGINAL RESEARCH

Automated selection of a software effort estimation model based on accuracy and uncertainty

Fatih Nayebi*, Alain Abran, Jean-Marc Desharnais

Software Engineering and Information Technologies Department, École de technologie supérieure, Université du Québec, Montreal, Canada

Received: February 1, 2015 **Accepted:** March 26, 2015 **Online Published:** April 28, 2015
DOI: 10.5430/air.v4n2p45 **URL:** http://dx.doi.org/10.5430/air.v4n2p45

Abstract

Software effort estimation plays an important role in the software development process: inaccurate estimation leads to poor utilization of resources and possibly to software project failure. Many software effort estimation techniques have been tried in an effort to develop models that generate optimal estimation accuracy, one of which is machine learning. It is crucial in machine learning to use a model that will maximize accuracy and minimize uncertainty for the purposes of software effort estimation. However, the process of selecting the best algorithm for estimation is complex and expert-dependent. This paper proposes an approach to analyzing datasets, automatically building estimation models with various machine learning techniques, and evaluating and comparing their results to find the model that produces the most accurate and surest estimates for a specific dataset. The proposed approach to automated model selection combines the Bayesian information criterion, correlation coefficients, and PRED measures.

Key Words: Bayesian information criterion, Machine learning, Software effort estimation

1 Introduction

In the software industry, planning the software development process affects scheduling for the software project, the effort and cost involved in the project, and the quality of the software product delivered to users. Software effort estimation is crucial to optimal planning and is important for controlling the software development process: overestimation can lead to the misuse of development resources, while underestimation can lead to a lower quality product. Effort estimates should be both accurate and certain.

There are a number of software effort estimation techniques available, such as expert judgment, the use of historical data, analogy-based estimation, proxy-based estimation, and algorithmic models. Software researchers have been seeking ways to build estimation models since the 1960s.[1] Some studies exploit machine learning techniques in algorithmic models because of their ability to learn continuously and their accuracy,[2] while machine learning-powered models rely on historical data and expert knowledge for learning and estimation. For this reason, the quality of the datasets used and of the feature subsets selected in the construction of these models is important, as is the knowledge of experts.

More accurate estimation models can be built by adding features, such as size, development language, business sector, etc., to machine learning techniques, but this may result in overfitting: an estimation model with too many features can exaggerate any minor fluctuations in the data and may overfit, leading to poor estimation performance in general. To avoid overfitting and to make the model produce reasonable

*Correspondence: Fatih Nayebi; Email: fatih.nayebi.1@etsmtl.net; Address: Software Engineering and Information Technologies Department, École de technologie supérieure, Université du Québec, Montreal, Canada.

estimates on different datasets, it is important to apply the uncertainty criterion during model selection.

Besides, finding the best model for a specific dataset is time-consuming and requires expert knowledge. What is more, even the experts may choose unsuitable criteria and estimation techniques. For these reasons, a more structured estimation process for exploring multiple machine learning techniques, supported by an automated selection subprocess, would be highly beneficial.

This study proposes an approach to determining which machine learning model performs the best in terms of estimating software development effort on a specific dataset. The proposed model will analyze new projects and learn from them on an ongoing basis, keeping the effort estimation process up to date.

Our objective is to develop an automated approach to exploring a number of estimation models to determine which performs better, considering one set of criteria and all the possible combinations of algorithms and feature subsets that could be exploited. We present and test a prototype of this approach in this paper, which we apply in a specific experimental context using 9 different estimation techniques, each of which uses 9 datasets.

The paper is organized as follows: Section 2 presents related work. Section 3 explains the proposed approach. Section 4 presents the automated prototype. Section 5 presents the 9 datasets selected to test the prototype. Section 6 presents the test results. Section 7 concludes the study and proposes future work.

2 Related works

In this section, we review some of the studies in which machine learning techniques are exploited for software effort estimation. Baskeles et al. investigate support vector regression, radial basis functions, regression trees, and multilayer perceptron estimation techniques. Using NASA and USC datasets, they found that regression trees perform the best.[4] Shin and Goel apply radial basis function neural networks to create an alternative to the linear regression approach.[5] Burgess and Lefley examine the capacity of genetic programming to generate accurate effort estimations on Desharnais data. Genetic programming performed better than other algorithms, but more effort was required to set up the estimation model.[6] Fisher applies artificial neural networks and regression trees to a COCOMO dataset for software effort estimation, and compares their results with those of non-machine learning techniques.[37] He found that the artificial neural network and regression tree techniques compare well.[7] Malhotra et al. compare several machine learning techniques using the PROMISE dataset.[18] They found that support vector machines outperform artificial neural networks and bagging, while decision trees outperform support vector machines.[8] Bibi et al. investigate five machine learning methods, examining not only their accuracy, but also their comprehensibility, causality, applicability, sensitivity, and uncertainty, as well as their handling of missing values and dynamic updating. They propose using a decision tree to select the best estimation technique, since the performance of these techniques can change, depending on the dataset and the weights of the model features.[9] Sarcia et al.[33] propose a different approach for evaluating estimation techniques, which is to select the model based on uncertainty instead of accuracy, as they consider that accuracy can vary widely, since it depends on the nature of the test sample, the error measure used, and the error statistics selected (e.g. MMRE (Mean Magnitude of Relative Error), PRED, Mean, and Standard Deviation). In contrast, uncertainty is an invariant evaluation criterion with respect to the error statistics. Also, they extend their technique by defining Bayesian Prediction Intervals for evaluating uncertainty.

Researchers will use different techniques, parameters, and datasets depending on their expertise. In our review of the literature, we found that the best effort estimation model for a particular dataset may change because different criteria are used. That is, a technique might beat all the others when it is used by an expert on some datasets, but it could fail when it is used by another expert in conjunction with different estimation parameters and datasets. Furthermore, in industrial practice, very few experts with sufficient in-depth knowledge of a broad range of estimation techniques are available to analyze a specific dataset, select suitable parameters, build models using different estimation techniques, and, from the results obtained, determine the best estimation technique for that dataset. Our study here proposes a strategy to address these issues, in which some steps of the estimation process are automated and a mechanism for selecting the best model for a specific dataset is included.

3 Proposed approach

We begin by considering a number of machine learning techniques for estimation, comparing their performances on a specific dataset, and deriving a set of criteria for selecting the model that performs the best on that dataset. We apply cross validation to define datasets for training and testing; feature subset selection to eliminate unnecessary features; execution of the available estimation techniques on the dataset to come up with suitable estimation models; analysis of model performances; and comparison of the models and identification of best model for the dataset investigated. All the steps in this approach are explained in greater detail in the subsections below.

3.1 Selection of datasets for training and testing

Our proposed approach cross validates the dataset selected for a project to create training and testing datasets to assess

how accurately an estimation model will perform in practice. It defines a part of the dataset for testing the model in the training phase, with a view to limiting problems such as overfitting and for generalizing the model to different independent datasets.

One of the most common approaches to cross validation is K-Fold Cross Validation. In this approach, in which N observations are validated, K defines the partition of the N observations into K disjoint subsets. Repeated calls return different randomly generated partitions. K-1 subsets are used for training and the final subset is used for testing. This process is repeated K times, leaving out one different subset for evaluation each time. Different values of K can be used in K-Fold Cross Validation. In this study, we take K = 5 (5-Fold Cross Validation), as it provides the best overall result.

3.2 Feature subset selection

The importance of feature subset selection is that it eliminates features that will decrease the accuracy of the estimation model, as well as decreasing the computational cost of the model. One of most powerful feature subset selection algorithms is Wrapper, which evaluates all combinations of features using two different algorithms:

- The learning algorithm for estimation,
- The search algorithm for finding the best combination of features, based on the results of the automated estimation techniques tested.

Various search and learning algorithms are used in Wrapper, but, in general, an exhaustive search algorithm and the K-Nearest Neighbor algorithm outperform the others. In this study, Wrapper is implemented with the feature subset selection settings considered above. It gives superior accuracy, but at the cost of high computational effort.[15] This cost is not a drawback for the proposed selection process, however, because the selection model does not need to run during estimation – and only if there are changes in the training dataset – on a weekly or bi-weekly basis, saving the model parameters to be used later. We combine feature subset selection with Linear Regression (LR) and the Multi Layer Perceptron (MLP) artificial neural network in this study, in order to achieve better accuracy with these algorithms.

3.3 Estimation

Briefly, these are the algorithms that we use in our proposed effort estimation approach.

- Linear Regression (LR)
- LR with Feature Subset Selection (LRFS)
- Least Median Squares (LMS)
- Pace Recognition (PR)
- K-Nearest Neighbor (KNN)
- M5P
- Support Vector Regression (SVR)
- Multi Layer Perceptron (MLP)
- MLP with Feature Subset Selection (MLPFS)

These algorithms are explained in greater detail in the subsections below. In addition, a software prototype is developed to run the algorithms, compare the outcomes, and select the best possible algorithm and parameters combination.

3.3.1 Linear regression

The LR technique is aimed at finding a learning function that maps a feature to an output with a continuous value, which is software effort in this study. LR achieves this by finding a line, which minimizes the sum of the squares on the training dataset. Simple LR is also combined with feature subset selection (LRFS) in this study.

3.3.2 Least median squares

The LMS linear regression method estimates the parameters by solving the nonlinear minimization of the median squares of the error.

3.3.3 Pace regression

PR is a type of linear regression, proposed by Wang,[35] which is aimed at removing the drawbacks of linear regression by calculating the effect of each feature and using clustering to estimate its contribution to the overall regression.[36]

3.3.4 K-nearest neighbor

The KNN algorithm works as follows: first, it computes the distance of a specific instance from all the other instances and finds its k nearest instances, or neighbors. Based on the k nearest neighbors, it estimates the dependent feature, which is software effort estimation in this study. We use the IBK algorithm, which is implemented as the KNN algorithm in the WEKA API, in our proposed model. We also use the IBK with different numbers of neighbors, k.

3.3.5 M5P

The Decision Tree (DT) is one of the most easily understood machine learning techniques in estimation. Basically, the DT consists of a root and internal and leaf nodes. These techniques work recursively, to make splitting decisions according to impurity measures until all the features have been assigned to a particular class. M5P is one of the DT algorithms used in this study. It is implemented in the WEKA API used in the model prototype, and empirical analyses are performed accordingly.[12]

3.3.6 Support vector regression

SVMs can be used in both regression and classification. SMOreg is an implementation of the SVM in regression. Here, we implement SMOreg with different kernels, and the RegSMOImproved algorithm[16] as a learning algorithm.

3.3.7 Multi layer perceptron

The MLP is a feed-forward Artificial Neural Network (ANN), which can be applied to regression, classification, and time-series forecasting.[17] We apply the MLP with various numbers of hidden layers, momentum values, and learning rates. It can also be combined with feature subset selection (MLPFS) and is used as such in this study.

3.4 Selection criteria

To evaluate the performance of the various models proposed, a number of criteria are combined and used in this study: PRED(25), the Correlation Coefficient (CORR), and the Bayesian Information Criterion (BIC).

3.4.1 PRED(X)

PRED is a ratio of estimates, which are within X percent of the actual values according to the Magnitude of relative error (MRE). The MRE is calculated as follows:

$$MRE = \frac{|actual_i - predicted_i|}{actual_i} \tag{1}$$

and PRED is calculated as follows:

$$PRED(X) = \frac{100}{N} * \sum_{i=1}^{N} \begin{cases} 1, & if MRE_i \leq X \\ 0, & otherwise \end{cases} \tag{2}$$

Generally, X=25 is the value of X used in the literature. The higher the value of PRED(X), the better. PRED(25) = 60 means that 6 out of 10 estimates are within 25% of the actual values.

3.4.2 Correlation coefficient

The correlation of two variables (the predicted value and the actual value in this case) shows how the variables are linearly related to each other. A CORR can take values between -1 and +1, and perfectly reflects either the negative or the positive linear relationship between the variables.[13]

3.4.3 Bayesian information criterion)

The BIC is a criterion for model selection among a finite set of models, and is based on the likelihood function. It was introduced by Schwarz (1978) as follows:

$$BIC = -2 * \ln L + k * \ln N \tag{3}$$

where N is the sample size, L is the maximized value of the likelihood function of the estimated model, and k is the number of independent features in the estimated model.[31]

Other above considered measures only measure accuracy, and model accuracy can change according to the size of the test dataset and the error statistics. Uncertainty can be a good alternative in estimation model selection, because it is a invariant criterion.[32] The BIC is used to measure the uncertainty of an estimation model, and is not dependent on the size of the test dataset or the error measures.

3.5 Selection technique

In this study, three of the criteria considered above: PRED, CORR, and BIC, are implemented for each model on each dataset individually. As well, separate ranks are calculated for each criteria (predRank, corrRank, and bicRank) for all the models, sorted by best result first. Then, the following formula is used to calculate the final rank of the model:

$$\begin{aligned} algorithmRank &= predWeight * predRank \\ &+ corrWeight * corrRank + bicWeight * bicRank \end{aligned} \tag{4}$$

Note that predWeight, corrWeight, and bicWeight are parametric coefficients, which can be determined prior to the estimation process, according to the organization's estimation goals. Estimation goals should be determined before model selection, since PRED and CORR measure the accuracy of the model, and BIC measures the uncertainty of the model. After calculating final ranks of all the models, the proposed solution selects the model with the highest rank.

4 Experimental prototype

To illustrate and evaluate the proposed approach, a machine learning model was developed in MATLAB.[10] Code was then compiled into dynamic link libraries (DLLs), and a software service was developed in .NET,[11] which exploits the machine learning DLLs developed as an estimation engine. Our proposed automated effort estimation model accomplishes all the steps depicted in Figure 1. Experimentation starts with the Cross Validate Data step, and ends with results analysis and selection of the best model.

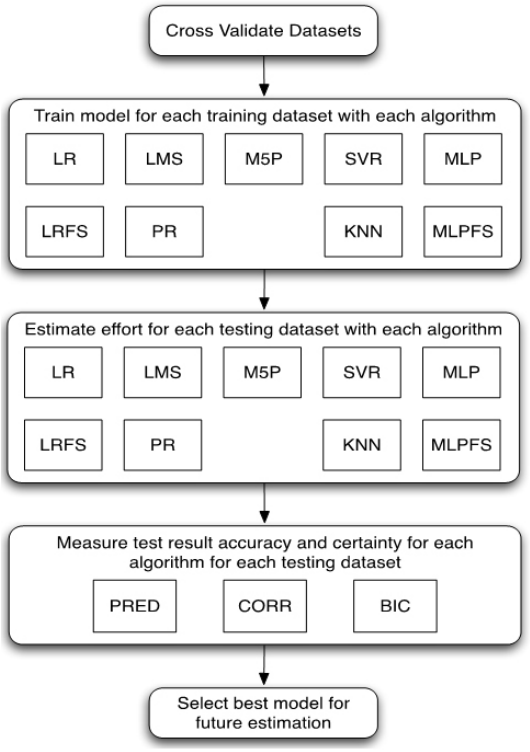

Figure 1: Proposed Approach

5 Empirical evaluation

The datasets used for empirical evaluation are described in this section.

5.1 Datasets

The characteristics of a dataset are considered to be significant in terms of model performance and model evaluation in machine learning.[34] Different datasets from different domains and sources behave differently. To illustrate the generality of the proposed approach, we use 9 different datasets in this study, with different types of measurement methods from the following two sources:

- PROMISE - The PRedictOr Models In Software Engineering data repository,[18, 19] which is a publicly available online data repository.
- ISBSG - The International Software Benchmarking Standards Group data repository. The ISBSG is a non-profit organization which maintains the ISBSG dataset of projects contributed by organizations across the world[20] that have been measured using a recognized functional size measurement method.

5.1.1 COCOMO 81 Dataset

This dataset consists of 63 TRW projects. Project effort is measured in person-months, and software size is measured in LOC (Lines Of Code). In the COCOMO model – equation (5) – the constants a and b are domain-specific constants, and EM_i are effort multipliers, which can be expressed on any of the six levels of the following ordinal scale: very low, low, nominal, high, very high, and extra high.

$$Effort = a*(kLOC^b)*(EM_1*EM_2*\cdots*EM_{15}) \quad (5)$$

5.1.2 NASA 93 Dataset

This dataset consists of 93 flight or ground system software projects developed for NASA in seven different development centers in the 1970s and 1980s.[19, 23, 24]

5.1.3 COCOMONASA Dataset

This dataset consists of 60 NASA projects from various centers in the 1980s and 1990s.

5.1.4 Desharnais Dataset

This dataset consists of 81 software projects from a Canadian software house collected by J. M. Desharnais.[19, 21, 22] The software size measure for this dataset is the function point (FP).

5.1.5 ISBSG-Telco Dataset

This dataset was extracted from the 2007 version of the ISBSG dataset and contains 132 projects.[26] In previous studies, a breakdown of the ISBSG repository according to business domain has also been used.[25]

5.1.6 China Dataset

This dataset consists of 499 projects and 16 numeric features; however, the duration feature has been removed because it was derived from the effort feature. The software size measure is the FP.[27]

5.1.7 Maxwell Dataset

This dataset consists of 62 software projects and 26 features. The size measure is the LOC. As with the China dataset, the duration feature has been removed, and for the same reason.[28]

5.1.8 Miyazaki 94 Dataset

This dataset consists of 24 projects from the Fujitsu Large Systems Users Group.[29] The software size measure is the LOC.

5.1.9 Finnish v2 Dataset

This dataset contains 38 projects, and its features are size, development type, hardware platform, development language, business sector, and effort.[30]

5.2 Summary of datasets

Table 1 presents a summary of these 9 datasets in terms of their projects and the number of features they contain. In summary, these datasets are diverse in terms of their domain, their sources, and their size, which makes them a relevant basis for testing the proposed approach.

Table 1: Feature and Project Count of Datasets

No	Dataset	No. of Features	No. of Projects
1	COCOMO 81	17	63
2	NASA 93	17	93
3	COCOMONASA	17	60
4	Desharnais	11	81
5	ISBSG-Telco	8	132
6	China	15	499
7	Maxwell	25	62
8	Miyazaki94	22	24
9	Finnish v2	5	38

6 Experimentation results

Nine different experiments, one for each of the 9 datasets, were carried out. For each experiment, the dataset was divided into training and testing datasets using cross validation. Nine different machine learning models were trained on these datasets, and the accuracy and uncertainty of the models were analyzed on the testing datasets.

The best performing model was selected based on the highest score achieved using the accuracy criterion and the lowest score using the uncertainty criterion. The score comprised a combination of the PRED, CORR, and BIC rankings of each model. For this set of experiments, the parametric weights of the ranking mechanism were assigned the values 0.25, 0.25, and 0.5 for PRED, CORR, and BIC respectively. This set of values gives equal importance to accuracy and certainty (considering that PRED and CORR are accuracy criteria, and BIC is an uncertainty criterion). Of course, in an organizational context, these values can be modified depending on the preferences of an organization: for example, it may want to stress accuracy over uncertainty, and so give more weight to PRED and CORR.

Figures 2 to 10 present the estimation results on the testing dataset for the best machine learning estimation model identified for each of the 9 datasets. The actual values and the estimated values are shown on the Y-axis and X-axis respectively.

Figure 2 shows the results for the COCOMO 81 dataset. After cross validation with a ratio of 4:1, 13 test instances remained, which are shown on the X-axis, and the actual values and the values predicted by SVR for these test instances are shown on the Y-axis. SVR is considered to be the best of all the models in this case, with a PRED value of 15.38, a CORR value of 0.93, and a BIC value of 14.47.

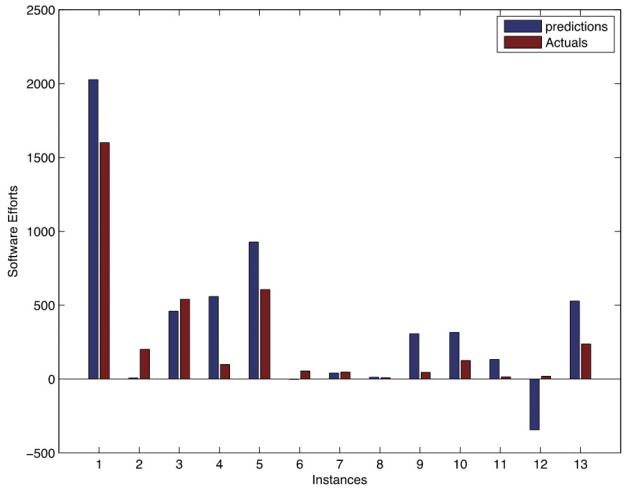

Figure 2: Results for COCOMO 81 Dataset – SVR

Figure 3 shows the results for the NASA 93 dataset. After cross validation with a ratio of 4:1, 19 test instances remained. M5P was found to be the best of the models in this case, with a PRED value of 21.05, a CORR value of 0.97, and a BIC value of 13.

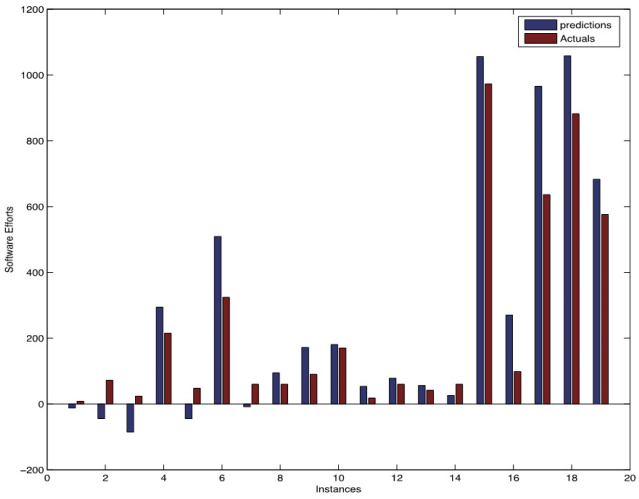

Figure 3: Results for NASA 93 Dataset – M5P

Figure 4 shows the results for the COCOMONASA dataset. After cross validation with a ratio of 4:1, 12 test instances remained. MLPFS was considered to be the best of the models in this case, with a PRED value of 33.33, a CORR value of 0.94, a BIC value of 13.88, and an overall ranking of 8.25.

Figure 5 shows the actual and predicted values for the Desharnais dataset. This dataset had been split into 65 training and 16 testing instances by cross validation. LRFS was selected as the best model by the automated estimation model, with a total score of 8.5.

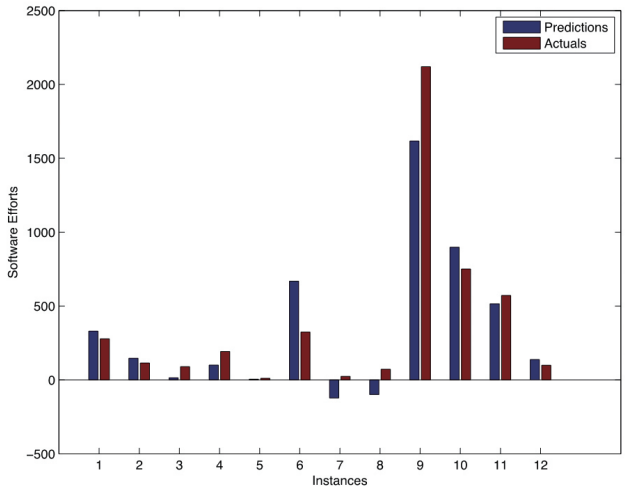

Figure 4: Results for COCOMONASA Dataset - MLPFS

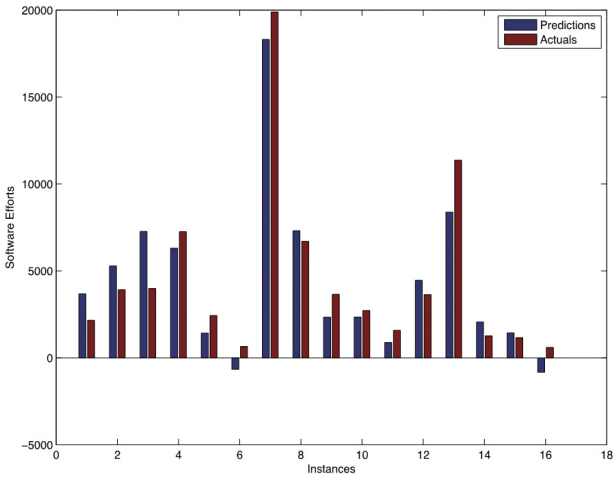

Figure 5: Results for Desharnais Dataset - LRFS

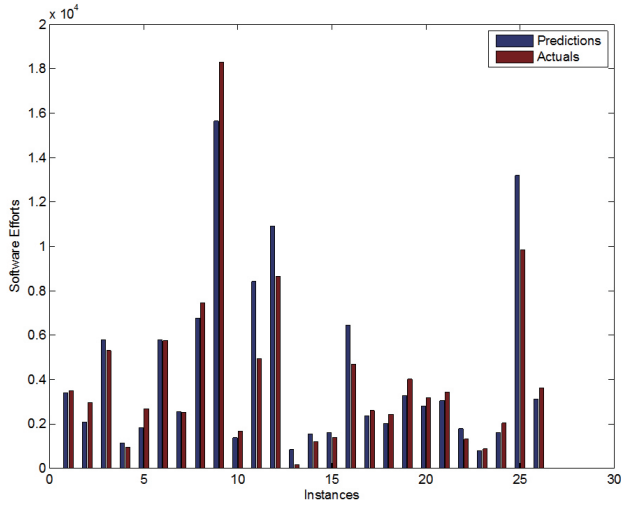

Figure 6: Results for ISBSG-Telco Dataset – M5P

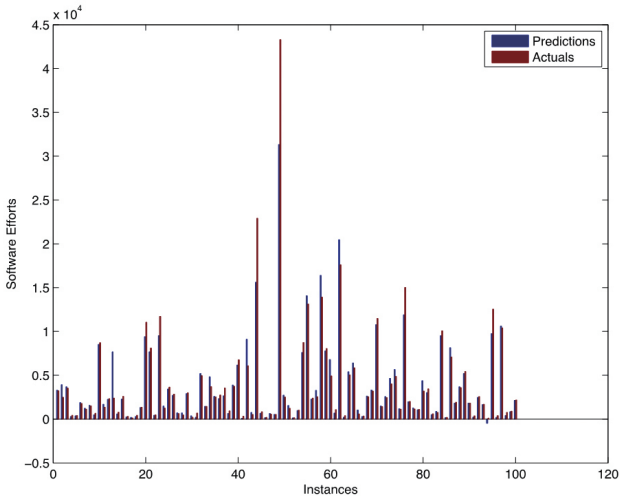

Figure 7: Results for China Dataset – M5P

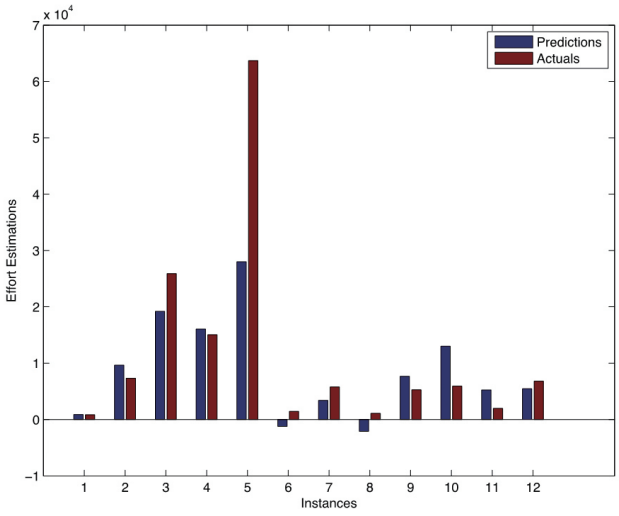

Figure 8: Results for Maxwell Dataset - SVR

Figure 6 shows the results for the ISBSG-Telco dataset. After the cross validation with ratio of 8:2, 26 test instances remained for validation of the candidate models. M5P was selected as the best of the models, with a PRED value of 65.38, a CORR value of 0.94, and a BIC value of 15.33.

Figure 7 shows the actual and predicted values for the 100 test instances of the China dataset. M5P outperformed the other models, with a PRED value of 72, a CORR value of 0.94, a BIC value of 16.63, and an overall ranking of 9 (out of 9). The highest PRED value in this study was obtained by the China dataset. A large number of instances in the training and testing datasets may be the reason for its superiority.

Figure 8 shows the results for the Maxwell dataset. After cross validation with a ratio of 4:1, 12 test instances remained.

SVR is considered to be the best of the models in this case,

with a PRED value of 38.46, a CORR value of 0.68, and a BIC value of 21.52. The highest BIC scores were obtained by the Maxwell dataset, which means that this dataset obtained the most uncertain results. Figure 9 shows the results for the Miyazaki94 dataset. The nine test instances on the X-axis were obtained by cross validation, with a ratio of 8:2. LRFS was selected by the automated estimation model, with a PRED value of 55.56, a CORR value of 0.97, and a BIC value of 7.2. The least uncertain results were obtained by this dataset, as its BIC score was the lowest of all the datasets.

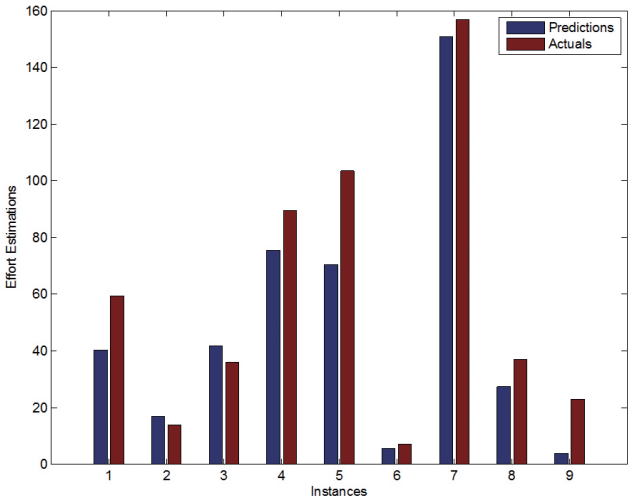

Figure 9: Results for Miyazaki94 Dataset - LRFS

Figure 10 shows the results for the Finnish v2 dataset. After cross validation with a ratio of 8:2, 30 training and 8 testing instances remained. M5P performed the best of all the models, with a PRED value of 62.5, a CORR value of 0.94, and a BIC value of 16.02.

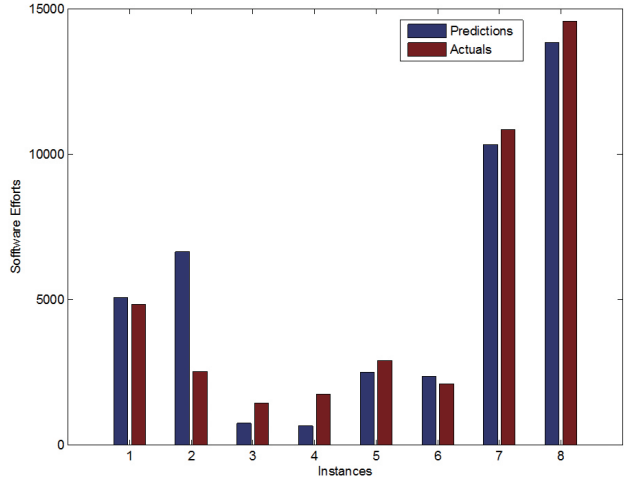

Figure 10: Results for Finnish v2 Dataset - LRFS

Table 2 presents the best model selected for each of the 9 datasets. For 4 of the 9 datasets, the M5P model outperformed the other models, in terms of accuracy and uncertainty, with LRFS receiving the next highest score.

Table 2: Best Model Selected for each of the 9 Datasets

Dataset	Model	PRED	CORR	BIC	Score
COCOMO 81	SVR	15.38	0.93	14.47	8.75
NASA 93	M5P	21.05	0.97	13	8.75
COCOMONASA	MLPFS	33.33	0.94	13.88	8.25
Desharnais	LRFS	37.5	0.95	16.36	8.5
ISBSG-Telco	M5P	65.38	0.94	15.33	8.75
China	M5P	72	0.94	16.63	9
Maxwell	SVR	38.46	0.68	21.52	7.25
Miyazaki94	LRFS	55.56	0.97	7.20	8.75
Finnish v2	M5P	62.5	0.94	16.02	8.75

Combining three different measures in the model selection process improves the estimation result. For instance, as shown in Table 3, LMS outperformed the other models for the COCOMONASA dataset in terms of the PRED measure, but it was not selected as the best model owing to its high BIC and low CORR. So, MLPFS, which had the highest overall score, was selected because of its high CORR and low BIC. Figure 4 shows that the MLPFS result is satisfactory.

Table 3: Model Scores in COCOMONASA Dataset

No	Model	PRED	CORR	BIC	Score
1	LR	33.33	0.93	14.06	5.25
2	LRFS	33.33	0.93	14.05	6.25
3	LMS	58.33	0.9	14.95	3.25
4	PR	33.33	0.93	14.32	5.5
5	KNN	16.67	0.82	14.89	1.5
6	M5P	25	0.9	14.62	2.75
7	SVR	41.67	0.91	14.51	5
8	MLP	33.33	0.94	13.88	7.25
9	MLPFS	33.33	0.95	13.87	8.25

7 Summary and future work

An approach has been proposed in this research for the use of multiple machine learning estimation techniques and evaluation of their outcomes for the selection of the one that performs best on a specific dataset. A prototype has been developed to implement this approach, which has been tested using 9 machine learning techniques and evaluated on 9 different datasets. This is in contrast to previous studies in the literature, in which machine learning models are compared and the results analyzed using a single dataset.[38, 39]

Furthermore, previous studies have compared machine learning models only in terms of accuracy.[13] However, accuracy may change with the data and with accuracy statistics, and so relying on accuracy alone may result in inefficient models. The ranking mechanism that we use in our proposed approach considers not only the PRED and CORR accuracy criteria, but also the uncertainty criterion.

In addition, the proposed estimation model combines the measures considered above based on the weights assigned by the organization. Clearly, the goals of any organization, or estimation researcher, can vary. One organization may be more interested in some results than in others, while another organization may aim for greater accuracy. Consequently, if an organization is interested in estimation results that are more certain, it can increase the weight of the BIC in the model. Furthermore, the proposed estimation model performed well for all the datasets, and the results of the model are highly acceptable in terms of uncertainty and accuracy. For these datasets, the M5P decision tree algorithm and linear regression with feature subset selection outperformed the other models, but artificial neural networks and support vector machines did not perform well. The results also show that feature subset selection enhances model performance, especially for linear regression.

In future work, it will be beneficial to replicate this study for the industrial context. Moreover, in this study, feature subset selection is combined only with the multi layer perceptron artificial neural network and linear regression, and it would be useful to combine feature subset selection with the other algorithms as well. Also, the performance of our proposed estimation model could be improved by making further refinements in terms of including parameter optimizations for each machine learning algorithm. A parameter-optimized model would consider the entire parameter space in an attempt to find the best combination of parameters for the machine learning method before estimating the effort for the specific dataset.

References

[1] Jørgensen M. Forecasting of software development work effort: Evidence on expert judgement and formal models. International Journal of Forecasting. 2007; 23(3): 449-462. http://dx.doi.org/10.1016/j.ijforecast.2007.05.008

[2] Singh Y, Bhatia PK, Sangwan O. A review of studies on machine learning techniques. International Journal of Computer Science and Security. 2007; 1(1): 70-84.

[3] Jørgensen M, Kjetil MO. Reasons for software effort estimation error: impact of respondent role, information collection approach, and data analysis method. IEEE Transactions on Software Engineering. 2004; 30(12): 993-1007. http://dx.doi.org/10.1109/TSE.2004.103

[4] Baskeles B, Turhan B, Bener A. Software effort estimation using machine learning methods. 22nd international symposium on Computer and information sciences. IEEE. 2007: 1-6.

[5] Shin M, Goel AL. Empirical data modeling in software engineering using radial basis functions. IEEE Transactions on Software Engineering. 2000; 26(6): 567-576. http://dx.doi.org/10.1109/32.852743

[6] Burgess CJ, Lefley M. Can genetic programming improve software effort estimation? A comparative evaluation. Information and Software Technology. 2001; 43(14): 863-873. http://dx.doi.org/10.1016/S0950-5849(01)00192-6

[7] Srinivasan K, Fisher D. Machine learning approaches to estimating software development effort. IEEE Transactions on Software Engineering. 1995; 21(2): 126-137. http://dx.doi.org/10.1109/32.345828

[8] Malhotra R, Jain A. Software effort prediction using statistical and machine learning methods. International Journal of Advanced Computer Science and Applications (IJACSA). 2011; 2(1).

[9] Bibi S, Stamelos I. Selecting the appropriate machine learning techniques for the prediction of software development costs. Artificial Intelligence Applications and Innovations. Springer US. 2006: 533-540.

[10] MATLAB R2015a, The MathWorks, Inc., Natick, Massachusetts, United States. Available from: http://www.mathworks.com/products/matlab/

[11] Microsoft .Net Framework and C Sharp Programming language. Available from: http://www.microsoft.com/net

[12] Holmes G, Donkin A, Witten IH. Weka: A machine learning workbench. Proceedings of the Second Australian and New Zealand Conference on Intelligent Information Systems. IEEE. 1994: 357-361.

[13] Menzies T, Chen ZH, Hihn J, et al. Selecting best practices for effort estimation. IEEE Transactions on Software Engineering. 2006; 32(11): 883-895. http://dx.doi.org/10.1109/TSE.2006.114

[14] Moon TK. The expectation-maximization algorithm. Signal processing magazine. IEEE. 1996; 13(6): 47-60.

[15] Gutlein M, Frank E, Hall M, et al. Large-scale attribute selection using wrappers. IEEE Symposium on Computational Intelligence and Data Mining. IEEE. 2009: 332-339.

[16] Shevade SK, Keerthi SS, Bhattacharyya C, et al. Improvements to the SMO algorithm for SVM regression. IEEE Transactions on Neural Networks. 2000; 11(5): 1188-1193. PMid:18249845. http://dx.doi.org/10.1109/72.870050

[17] Hykin S. Neural Networks: A Comprehensive Foundation. Printice-Hall. Inc., New Jersey (1999).

[18] Boetticher G, Menzies T, Ostrand T. PROMISE Repository of empirical software engineering data. West Virginia University, Department of Computer Science (2007).

[19] Menzies T, Caglayan B, Kocaguneli E, et al. The promise repository of empirical software engineering data. West Virginia University, Department of Computer Science (2012).

[20] Lokan C, Wright T, Hill PR, et al. Organizational benchmarking using the ISBSG data repository. IEEE Software. 2001; 18(5): 26-32. http://dx.doi.org/10.1109/52.951491

[21] Desharnais JM. Analyse Statistique de la Productivitie des Projets Informatique a Partie de la Technique des Point des Fonction. Masters thesis, UQAM. 1989

[22] Desharnais dataset. http://promisedata.org/repository/desharnais

[23] Menzies T, Port D, Chen ZH, et al. Validation methods for calibrating software effort models. In Proceedings of the 27th international conference on Software engineering. ACM. 2005: 587-595.

[24] NASA93 dataset. https://code.google.com/p/promisedata/source/browse/trunk/effort/nasa93/

[25] Kudo M, Sklansky J. Comparison of algorithms that select features for pattern classifiers. Pattern recognition. 2000; 33(1): 25-41. http://dx.doi.org/10.1016/S0031-3203(99)00041-2

[26] Kocaguneli E. Better methods for configuring case-based reasoning systems. Master's thesis. Computer Engineering. Bogazici University (2010).

[27] China dataset. https://code.google.com/p/promisedata/source/browse/trunk/effort/china

[28] Li YF, Xie M, Goh TN. A study of the non-linear adjustment for analogy based software cost estimation. Empirical Software Engi-

neering. 2009; 14(6): 603-643. http://dx.doi.org/10.1007/s10664-008-9104-6

[29] Miyazaki Y, Terakado M, Ozaki K, et al. Robust regression for developing software estimation models. Journal of Systems and Software.1994; 27(1): 3-16. http://dx.doi.org/10.1016/0164-1212(94)90110-4

[30] Finnish-v2 dataset. promisedata.org/repository/data/finnish/finnish-v2.arff

[31] Burnham KP, Anderson DR. Multimodel inference understanding AIC and BIC in model selection. Sociological methods & research. 2004; 33(2): 261-304. http://dx.doi.org/10.1177/0049124104268644

[32] Sarcia SA, Basili VR, Cantone G. Using uncertainty as a model selection and comparison criterion. In Proceedings of the 5th International Conference on Predictor Models in Software Engineering. ACM. 2009. p18.

[33] Sarcia SA. An Approach to improving parametric estimation models in the case of violation of assumptions based upon risk analysis. PhD diss. 2009.

[34] Bakır A, Turhan B, Bener AB. A new perspective on data homogeneity in software cost estimation: a study in the embedded systems domain. Software Quality Journal. 2010; 18(1): 57-80. http://dx.doi.org/10.1007/s11219-009-9081-z

[35] Wang Y. A new approach to fitting linear models in high dimensional spaces. PhD diss. The University of Waikato, 2000.

[36] Wang Y, Witten IH. Pace regression. University of Waikato. Dept. of Computer Science. 1999. p27.

[37] Boehm BW. Software engineering economics. Vol. 197. Englewood Cliffs (NJ): Prentice-hall, 1981.

[38] Finnie GR, Wittig GE, Desharnais JM. A comparison of software effort estimation techniques: using function points with neural networks, case-based reasoning and regression models. Journal of Systems and Software. 1997; 39(3): 281-289. http://dx.doi.org/10.1016/S0164-1212(97)00055-1

[39] Elish MO. Improved estimation of software project effort using multiple additive regression trees. Expert Systems with Applications. 2009; 36(7): 10774-8. http://dx.doi.org/10.1016/j.eswa.2009.02.013