On Building Prediction Systems for Software Engineers

MARTIN SHEPPERD  
Empirical Software Engineering Research Group, School of Design, Engineering and Computing, Bournemouth University, Talbot Campus, BH12 5BB, UK  
mshepper@bournemouth.ac.uk

MICHELLE CARTWRIGHT  
Empirical Software Engineering Research Group, School of Design, Engineering and Computing, Bournemouth University, Talbot Campus, BH12 5BB, UK  
mcartwri@bournemouth.ac.uk

GADA KADODA  
Empirical Software Engineering Research Group, School of Design, Engineering and Computing, Bournemouth University, Talbot Campus, BH12 5BB, UK  
gkadoda@bournemouth.ac.uk

Abstract. Building and evaluating prediction systems is an important activity for software engineering researchers. Increasing numbers of techniques and datasets are now being made available. Unfortunately systematic comparison is hindered by the use of different accuracy indicators and evaluation processes. We argue that these indicators are statistics that describe properties of the estimation errors or residuals and that the sensible choice of indicator is largely governed by the goals of the estimator. For this reason it may be helpful for researchers to provide a range of indicators. We also argue that it is useful to formally test for significant differences between competing prediction systems and note that where only a few cases are available this can be problematic, in other words the research instrument may have insufficient power. We demonstrate that this is the case for a well known empirical study of cost models. Simulation, however, could be one means of overcoming this difficulty.

Keywords: Cost estimation, prediction system, empirical evaluation

1. Introduction

This short note summarises some of the issues and opportunities that are opening up for researchers and users of software engineering prediction systems. First, we need to consider what we mean by a prediction system. A prediction system has the following structure:

\[ ps : X \rightarrow \hat{y} \]

\( ps \) is a function that takes \( X \) as an input vector and outputs \( \hat{y} \), the estimated value of \( y \), where \( y \) is the true value. The scalar type of \( \hat{y} \) is very important in determining the nature of \( ps \). Typically a software engineer might wish to predict development effort, which is usually treated as a continuous variable. Another frequent target of prediction systems is defects, which is an example of a discrete variable. In other disciplines categorical outputs are more commonplace, for example we might wish to classify mushrooms or whatever. For this reason the machine learning community often refer to prediction systems as classifiers.

As already indicated two major targets for prediction in software engineering are project effort (and closely related is duration) and defects. Of course there are many other possibilities, reliability being one of them. This note focuses on project effort as a useful example...
that illustrates many wider issues of prediction. It is also an area of considerable research activity.

Researchers have been developing software effort prediction systems for over 40 years. A paper by Benington (1956) is the first published work on the subject that the authors are aware of. Early work generally took the form of collecting data and fitting simple linear or exponential models, for example Walston and Felix (1977) suggested a prediction system \( E = 5.2^* \text{KLOC}^{0.91} \) where \( E \) was effort in person months and KLOC was software size in thousands of lines of code. To derive this prediction system they collected data from 60 software projects from the Federal Systems Division at IBM. Unfortunately such a model only had local significance.

The next development was the trend towards universal prediction systems. These tended to be more complex than the early models with the objective of being more widely applicable. This led to the inclusion of many parameters or drivers to reflect differences in software development environments. COCOMO is probably the best known example of this approach (Boehm, 1984). Unfortunately, despite the increased sophistication of these prediction systems there is little evidence to suggest that they perform well outside of their development environments, in other words they tend to be over adapted to their training (Kemerer, 1987; Kitchenham and Taylor, 1984).

By the beginning of the 1990s researchers were addressing problems such as calibration and the use of statistical methods to build local prediction systems (Jeffery and Low, 1990; Kok, Kitchenham and Kirakowski, 1990). There was also some pioneering work examining the application of machine learning (ML) methods. This included work by our research group at Bournemouth University on the use of analogy in prediction that typically outperformed regression analysis (Shepperd and Schofield, 1997). ML results were often encouraging but not always consistent. For example, Finnie and Wittig (1997) found neural nets to be substantially more accurate than standard regression techniques, whilst Samson (1997) found them to be highly inaccurate. This meant that it has been difficult to construct an overall picture. The problems have been somewhat compounded by investigators employing different accuracy indicators, including \( R \)-squared and mean magnitude of relative error (MMRE). In addition, investigators have adopted differing evaluation procedures ranging from model fitting (i.e. using the entire dataset) to validating with an independent dataset.

The remainder of this note focuses upon issues of how to compare prediction systems in a systematic fashion and what kind of evidence is required in order to make a judgement regarding the relative performances of prediction systems A and B. This is an important topic to tackle if we wish to be able to make significant progress in developing and evaluating software engineering prediction systems. Or to put it another way, if we cannot satisfactorily compare competing prediction systems we will be unable to make meaningful progress.

2. Comparing Prediction Systems

From the earliest days, investigators have been concerned with the accuracy of prediction systems. Unfortunately, over the years many different indicators have been proposed and utilised.
Table 1. Summary of main accuracy indicators.

<table>
<thead>
<tr>
<th>Accuracy indicator</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TotalError = [ \sum_{i=1}^{n} (x_i - \hat{x}_i) ]</td>
<td>This sum of errors or residuals makes comparisons between different datasets difficult. Also over and under estimates can mask one another.</td>
</tr>
<tr>
<td>TotalAbsoluteError = [ \sum_{i=1}^{n}</td>
<td>(x_i - \hat{x}_i)</td>
</tr>
<tr>
<td>TotalRelativeError = [ \sum_{i=1}^{n} \left( \frac{x_i - \hat{x}_i}{x_i} \right) ]</td>
<td>One major drawback of the total error indicator is its lack of consideration of the scope of the values being estimated.</td>
</tr>
<tr>
<td>MMRE = [ \frac{1}{n} \sum_{i=1}^{n} \left( \frac{</td>
<td>\hat{x}_i - x_i</td>
</tr>
<tr>
<td>BalancedMMRE = [ \frac{1}{n} \sum_{i=1}^{n} \left( \frac{</td>
<td>\hat{x}_i - x_i</td>
</tr>
<tr>
<td>Pred(n)</td>
<td>Another widely used prediction quality indicator is Pred(n), which is simply the percentage of estimates that are within n% of the actual value. Typically n = 25%.</td>
</tr>
<tr>
<td>MeanSquareError = [ \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 ]</td>
<td>The argument for using the mean square error is that if you are risk averse it penalises large errors more than small errors, however, it suffers from the problem of it being difficult to compare values across datasets.</td>
</tr>
<tr>
<td>[ R^2 = 1 - \frac{\sum_{i=1}^{n} (x_i - \hat{x}<em>i)^2}{\sum</em>{i=1}^{n} (x_i - \bar{x})^2} ]</td>
<td>Strictly this is more a measure of the goodness of model fit rather than predictive accuracy. Where a regression equation embodies multiple independent variables the adjusted R-squared indicator should be used.</td>
</tr>
</tbody>
</table>

Table 1 contains details of eight prediction system accuracy indicators, however, others may be found in the literature, see for example (Conte, Dunsmore and Shen, 1986; Kemerer, 1987). We not aware of these other indicators being used to any significant degree.

The problem of many indicators is compounded by the fact that they do not necessarily move together as the following example illustrates. We compare three different regression techniques for building software project effort prediction systems and evaluate them using the Desharnais dataset (1989). The dataset comprises 77 complete cases (projects) each of
Table 2. A comparison of different accuracy indicators.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>PS1</th>
<th>PS2</th>
<th>PS3</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R^2 )</td>
<td>0.28</td>
<td>0.25</td>
<td>0.26</td>
</tr>
<tr>
<td>MMRE</td>
<td>0.78</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td>Pred(25)</td>
<td>45%</td>
<td>35%</td>
<td>35%</td>
</tr>
<tr>
<td>Balanced MMRE</td>
<td>0.84</td>
<td>0.78</td>
<td>0.77</td>
</tr>
</tbody>
</table>

Table 3. Estimation strategies and accuracy indicators.

<table>
<thead>
<tr>
<th>Estimation Strategy</th>
<th>Accuracy Indicators</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>risk averse</td>
<td>Mean Square Error, (adjusted) ( R^2 )-squared</td>
<td>spread</td>
</tr>
<tr>
<td>error minimisation</td>
<td>MMRE, Balanced MMRE, Absolute Total Error, Mean Absolute Total Error</td>
<td>spread</td>
</tr>
<tr>
<td>portfolio based</td>
<td>Total Error, Total Percentage Error</td>
<td>centre</td>
</tr>
</tbody>
</table>

which is characterised by 9 independent variables. Three different regression techniques are applied to develop the prediction systems PS1, PS2 and PS3.

Table 2 reveals the results of using four accuracy indicators to make comparisons between PS1, PS2 and PS3. The striking feature is that there is no consensus. In other words, the choice of indicator determines the preferred prediction system. This is unsatisfactory, if we wish to be able to systematically compare different techniques for developing systems.

We believe the reason for this conflict to be that the accuracy indicators are best regarded as different descriptive statistics for errors or residuals (i.e. the \( i \)-th residual is \( \epsilon_i = \hat{y}_i - y_i \) and the error ratio \( z \) is \( \epsilon_i / \hat{y}_i \)). One might use the latter if a proportionate view is required or the researchers wish to compare across different datasets. Such statistics can be classified as either describing central tendency (e.g. mean and median) or spread (e.g. variance and skewness). We can also determine that some indicators are most robust to the presence of extreme outliers, that is robust indicators are less influenced by extreme prediction errors. Such considerations depend upon the needs of the estimator.

There are also a variety of prediction strategies. Some accuracy indicators seem more closely aligned with some goals than others (see Table 3). Here we see three common strategies. Risk averse indicates a higher tolerance of overall error in order to avoid occasional catastrophic errors. Error minimising suggests a concern to minimise overall errors irrespective of how they are distributed. A portfolio type strategy (Kitchenham and Linkman, 1997) is often overlooked by researchers but can be appropriate if a number of projects or prediction elements have to be managed. Here errors can be tolerated as long as they are compensated by other errors. Thus, the goal is to have an unbiased prediction system. Clearly this property is best described by an accuracy indicator related to central tendency such as Total Error and rather poorly described by an indicator pertaining to spread such as MMRE.
The observation that there are multiple estimation goals and that these are best assessed by different indicators does not entirely resolve the problem of whether we should prefer prediction system PS1, PS2 or PS3, particularly in a research setting where there are no estimators. Since the estimation objectives will not be known to researchers, Pickard et al. (1999) have used boxplots to show both the central tendency and spread of the residuals. However, some differences in indicators can be relatively small, for example, Table 2 shows the \( R \)-squared values ranging from 25 to 28%. Are such differences significant?

Building on the ideas of Stensrud and Myrtevit (1998) we used formal tests of significance when comparing the residuals from the competing prediction systems. In the Stensrud and Myrtevit work they used paired t-tests, however, we prefer non-parametric tests. Where the residuals are naturally paired we use a Wilcoxon Signed Rank test and otherwise a Mann-Whitney U test. The reason for this is that we normally analyse absolute residuals and assume that estimators are indifferent between over and under estimates, consequently the distribution must be skewed. Failure to use absolute residuals results in positive and negative errors cancelling each other out. This may help reveal bias but is not informative as regards accuracy.

We believe this approach provides a formal basis for evaluating prediction systems and it expect it to be widely adopted by future researchers in this field. It also offers a further interesting test in that researchers can make a formal comparison with a non-prediction system such as just using the sample mean or even using the previous case as the prediction for the new case. It is incumbent upon researchers to show that their techniques are significantly better than not predicting.

### 3. A Simple Example

To underline the point that evaluating prediction systems requires a more formal basis we now consider a pioneering and well known study by Kemerer (1987) in which he compared four different prediction systems. The study used a dataset of \( n = 15 \). He reported widely varying performance which he described using the widely applied MMRE accuracy indicator as ranging from 85% to 771%.

Figure 1 shows the distributions of the absolute residuals for four prediction systems (SLIM, basic and intermediate COCOMO and function points) and a non-prediction system using the sample mean. Note that the Y axis is somewhat truncated, so a SLIM extreme outlier is not depicted.

Upon re-analysis, when we apply a Mann-Whitney U Test to the absolute residuals we are unable to demonstrate that any system is more accurate than merely using the sample mean (i.e. that median residual from the prediction system is lower than from simply guessing based on the sample mean). The probabilities in Table 4 are the likelihood that the null hypothesis (i.e. the non-prediction system is no worse than the prediction system) is true. The best prediction system, function points, is not close to the 95% confidence limit so the null hypothesis stands in all cases. In addition, we can see that there is no evidence to substantiate other claims such as basic COCOMO outperforms intermediate COCOMO \( (p = 0.3414) \).
Figure 1. Boxplots of absolute residuals from Kemerer evaluation.

Table 4. Likelihood that a prediction system is predicting better than the sample mean.

<table>
<thead>
<tr>
<th>Prediction system</th>
<th>( p(PS \leq nonPS) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLIM</td>
<td>( p = 0.9996 )</td>
</tr>
<tr>
<td>Intermediate COCOMO</td>
<td>( p = 0.9997 )</td>
</tr>
<tr>
<td>Basic COCOMO</td>
<td>( p = 1.0000 )</td>
</tr>
<tr>
<td>Function points</td>
<td>( p = 0.3414 )</td>
</tr>
</tbody>
</table>

4. Summary

In this keynote it has been argued that prediction systems are important for software engineers as evidenced by the fact that they have been the topic of research for more than four decades. It has also been observed that there are certain difficulties in making systematic comparisons between competing approaches, due to many different accuracy indicators having been employed. We have demonstrated how these indicators do not always behave consistently. Further, the choice of indicator is to some extent governed by the estimator’s prediction strategy. An additional problem is that it is not easy to determine the significance of small differences between accuracy indicators. Consequently, an alternative approach has been proposed and demonstrated for comparing prediction systems based upon absolute residuals. This enables formal tests of significance to be made using non-parametric tests such as Mann-Whitney U and Wilcoxon Signed Ranks. Finally we have shown that this approach highlights problems with small sized validation sets. Put simply, they do not have sufficient power to detect many of the claimed effects. Given the difficulties of collecting large industrial datasets we recommend consideration be given to simulation techniques, particularly where one wishes to explore the relationship between the prediction environment and prediction system.
Acknowledgments

The research described in this invited paper was partly funded by grant (Grant GR/L37298) from the UK Engineering and Physical Sciences Research Council and the Defence Evaluation Research Agency. The authors are also indebted to Barbara Kitchenham, Steve MacDonell, Ingunn Myrtveit and Erik Stensrud for many of the insights and ideas contained within this paper. Any remaining errors are, of course, due to ourselves!

Notes

1. A version of this paper was first presented as an invited keynote at the 4th Intl. Conf. on Empirical Assessment & Evaluation in Software Engineering April 17–19, 2000.
2. Clearly, when evaluating prediction systems, there are many characteristics to consider, for example, sensitivity to input error and availability of inputs. Nevertheless, most research has focused on accuracy.
3. The Estimacs prediction system is excluded since only 9 out 15 predictions were provided.

References

Prof. Martin Shepperd has the Chair of Software Engineering at Bournemouth University, UK. He received a PhD in computer science from the Open University in 1991. He has published more than 70 refereed papers and three books in the field of empirical software engineering. He has served on many programme committees including the European Software Engineering Conference, the IEEE International Metrics Symposium and the Software Quality Workshop. He is the editor of the journal Information & Software Technology and an editorial board member for IEEE Transactions on Software Engineering. He is a council member of the Centre for Software Reliability and a member of the UK, Engineering and Physical Sciences Research Council Computing College. His research interests include software metrics and empirical software engineering.

Dr Michelle Cartwright is a research fellow in the Empirical Software Engineering Group, in the school of Design, Engineering and Computing, Bournemouth University, UK. She received a BSc degree (honours) in Computer Science from the University of Wolverhampton, and a PhD from Bournemouth University. Her research interests include software metrics for object-oriented systems, empirical software engineering, software prediction and empirical software engineering for internet technology.

Dr Gada Kadoda is a research fellow in the Empirical Software Engineering Group, in the school of Design, Engineering and Computing, Bournemouth University, UK. She received a BSc degree (honours) in Computer Science from the University of Khartoum (Sudan), an MSc in Information Systems and Technology from City University, and a PhD from Loughborough University. Her research interests include computer-aided teaching, formal methods, and software prediction systems.