Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study

Mahmoud O. Elish and Karim O. Elish
Information and Computer Science Department
King Fahd University of Petroleum & Minerals
Dhahran 31261, Saudi Arabia
{elish kelish}@kfupm.edu.sa

Abstract

There is an increasing interest in more accurate prediction of software maintainability in order to better manage and control software maintenance. Recently, TreeNet has been proposed as a novel advance in data mining that extends and improves the CART (classification and regression trees) model using stochastic gradient boosting. This paper empirically investigates whether the TreeNet model yields improved prediction accuracy over the recently published object-oriented software maintainability prediction models: multivariate adaptive regression splines, multivariate linear regression, support vector regression, artificial neural network, and regression tree. The results indicate that improved, or at least competitive, prediction accuracy has been achieved when applying the TreeNet model.

1. Introduction

Software maintenance has been one of the most difficult and costly tasks in the software development lifecycle [15, 28]. Accurate prediction of software maintainability can be useful to support and guide [5]: software related decision making; maintenance process efficiency; comparing productivity and costs among different projects; resource and staff allocation, and so on. This in turn helps in keeping future maintenance effort under control.

Despite the availability of few maintainability prediction models such as [5, 14, 28], their prediction accuracy is considered low according to the criteria suggested in the literature [4, 17]. Consequently, it is necessary to investigate new models that may yield improved prediction accuracy. Recently, TreeNet has been proposed as a novel advance in data mining that extends and improves the CART (classification and regression trees) model using stochastic gradient boosting [11]. The capability of TreeNet has been evaluated in a broad range of prediction problems in different applications, and has demonstrated promising results. Examples include software reliability [12], fraud detection [19], epidemiology [11], insurance [6], financial distress [20], medical and biological sciences [22]. The main features of TreeNet models include [11, 23]: automatic variable subset selection; ability to handle data without pre-processing; resistance to outliers; automatic handling of missing values; robustness to dirty and partially inaccurate data; high speed; and resistance to over-training. These features of TreeNet in addition to its demonstrated capability in many prediction problems suggest that it can also be a promising novel prediction model for object-oriented software maintainability.

This paper empirically investigates whether TreeNet yields improved prediction accuracy over the recently published object-oriented software maintainability prediction models [28]: multivariate adaptive regression splines, multivariate linear regression, support vector regression, artificial neural network, and regression tree.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 provides an overview of TreeNet. Section 4 discusses the conducted empirical evaluation and its results. Section 5 concludes the paper and outlines directions for future work.

2. Related work

Many research studies have used linear regression models to build software maintainability prediction models. For example, multiple linear regression models were used by Li and Henry [15] to predict maintenance effort; by Fioravanti and Nesi [8] to predict adaptive maintenance effort; by De Lucia et al. [5] to predict corrective maintenance effort; by Misra...
[18] to predict maintainability index; and by Bandi [1]
to predict maintenance time.

Some machine learning models were also applied
in predicting software maintainability. Thwin and
Quah [25] used neural networks to build object-
oriented software maintainability prediction models.
Koten and Gray [14] used Bayesian belief network
(BBN) to predict object-oriented software
maintainability. The limitations of BBN, however,
might make it difficult for practitioners to build a
maintainability model with high prediction accuracy
[16, 26, 28].

Recently, Zhou and Leung [28] used multivariate
adaptive regression splines (MARS) for predicting
object-oriented software maintainability. They
compared the prediction performance of the MARS
model with four models: multivariate linear regression
(MLR), support vector regression (SVR), artificial
neural network (ANN), and regression tree (RT). They
used two datasets (UIMS and QUES) by Li and Henry
[15] for model evaluation. Their results suggest that
MARS can predict maintainability more accurately
than the other four models for the QUES dataset, and
as accurate as the best model for the UIMS dataset. In
our study, the prediction performance of TreeNet is
compared against the five most-recently investigated
prediction models in [28], i.e. MARS, MLR, SVR,
ANN, and RT.

3. TreeNet: an overview

TreeNet (also known as multiple additive
regression trees (MART)) is a novel advance in data
mining proposed by Friedman [10, 11] at Stanford
University. TreeNet extends and improves the CART
(classification and regression trees) model using
stochastic gradient boosting [10]. Boosting is a general
method that attempts to “boost” the accuracy of any
given learning algorithm by fitting a series of models
each having a low error rate and then combining into
an ensemble that may perform better [19, 24].

Therefore, TreeNet models inherit almost all of the
advantages of tree-based models, while overcoming
their primary disadvantages, i.e., inaccuracy [11]. The
main features of TreeNet models include [11, 23]:
automatic variable subset selection; ability to handle
data without pre-processing; resistance to outliers;
automatic handling of missing values; robustness to
dirty and partially inaccurate data; high speed; and
resistance to over-training.

A TreeNet model can be thought of as a series
expansion approximating the true functional
relationship [23]:

\[ F(X) = F_0 + \beta_1 T_1(X) + \beta_2 T_2(X) + \ldots + \beta_n T_n(X) \]

where Ti is a small tree. Each tree refines and
improves on its predecessors. TreeNet models are thus
typically composed of hundreds of small trees, each of
which contributes slight refinement to the overall
model.

Technically, TreeNet employs an iterative
algorithm, where at each iteration m, a new regression
tree \( T_m(x; \{ R_{jm} \}_1^J) \) is built to partition the x-space
(predictor variables) into J disjoint regions \( \{ R_{jm} \}_1^J \)
and predict a separate constant value in each one [11]:

\[ T_m(x; \{ R_{jm} \}_1^J) = \sum_{j=1}^J \tilde{y}_{jm} I(x \in R_{jm}) \]

Here, \( \tilde{y}_{jm} = \text{mean}_{x \in R_{jm}} \tilde{y}_{m} \) is the mean of
‘pseudo’-residuals in each region \( R_{jm} \) induced at the
mth iteration. [10, 11]

The current approximation \( F_{m-1}(x) \) is then
separately updated in each corresponding region [10,
11]:

\[ F_m(x) = F_{m-1}(x) + \nu \gamma_{jm} I(x \in R_{jm}) \]

Where,

\[ \gamma_{jm} = \arg \min_{\gamma} \sum_{x \in R_{jm}} L(y, F_{m-1}(x) + \gamma) \]

The ‘shrinkage’ parameter \( 0 < \nu \leq 1 \) controls the
learning rate of the procedure. This leads to the
following TreeNet algorithm for generalized boosting
of regression trees [10, 11]:

1. \( F_0(x) = \arg \min_{\gamma} \sum_{i=1}^N L(y_i, \gamma) \)

2. For \( m = 1 \) to \( M \) do

3. \( \tilde{y}_{jm} = \left[ \frac{\partial L(y, F(x))}{\partial F(x)} \right]_{F(x) = F_{m-1}(x)}, i = 1, N \)
4. \[ \{ R_{jm} \}^j_i = J - \text{terminal node tree } (\{ \tilde{y}_{jm}, x_i \}^y) \]

5. \[ \gamma_{jm} = \arg \min_{\gamma} \sum_{x_j \in R_m} L(y_j, F_{m-1}(x_j) + \gamma) \]

6. \[ F_m(x) = F_{m-1}(x) + \nu \gamma_{jm} I(x \in R_{jm}) \]

7. End For

There are specific algorithms for several loss criteria including \([10, 11]\): least squares, least absolute deviation, and Huber-M.

4. Empirical evaluation

This section discusses the conducted empirical study that evaluates the capability of TreeNet in predicting object-oriented software maintainability.

4.1. Goal

Using GQM template \([2]\) for goal definition, the goal of this empirical study is defined as follows: Evaluate TreeNet for the purpose of predicting object-oriented software maintainability with respect to its prediction accuracy against the five compared models (MARS, MLR, SVR, ANN, and RT) from the point of view of researchers and practitioners in the context of two object-oriented software datasets.

4.2. Datasets

This study uses two popular object-oriented maintainability datasets published by Li and Henry \([15]\): UIMS and QUES datasets. These datasets were chosen mainly because they have been recently used to evaluate the performance of some machine learning models in predicting object-oriented software maintainability \([28]\), and hence we want to be able to compare our results against this recently published work.

The UIMS dataset contains class-level metrics data collected from 39 classes of a user interface management system, whereas the QUES dataset contains the same metrics collected from 71 classes of a quality evaluation system. Both systems were implemented in Ada. Both datasets consist of eleven class-level metrics: ten independent variables and one dependent variable. The independent variables are five Chidambar and Kemerer metrics \([3]\): WMC, DIT, NOC, RFC, and LCOM; four Li and Henry metrics \([15]\): MPC, DAC, NOM, and SIZE2; and one traditional lines of code metric (SIZE1). The dependent variable is a maintenance effort surrogate measure (CHANGE), which is the number of lines in the code that were changed per class during a 3-year maintenance period. A line change could be an addition or a deletion. A change in the content of a line is counted as a deletion and an addition.

Table 1 defines each metric in the datasets. Data values for the NOC metric in the QUES dataset is not available, and thus is excluded from the following analysis.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMC (Weighted Methods per Class)</td>
<td>The sum of McCabes's cyclomatic complexities of all local methods in a class</td>
</tr>
<tr>
<td>DIT (Depth of Inheritance Tree)</td>
<td>The depth of a class in the inheritance tree where the root class is zero</td>
</tr>
<tr>
<td>NOC (Number of Children)</td>
<td>The number of child classes for a class</td>
</tr>
<tr>
<td>RFC (Response For a Class)</td>
<td>The number of local methods plus the number of nonlocal methods called by local methods</td>
</tr>
<tr>
<td>LCOM (Lack of Cohesion of Methods)</td>
<td>The number of disjoint sets of local methods. Each method in a disjoint set shares at least one instance variable with at least one member of the same set</td>
</tr>
<tr>
<td>MPC (Message Passing Coupling)</td>
<td>The number of messages sent out from a class</td>
</tr>
<tr>
<td>DAC (Data Abstraction Coupling)</td>
<td>The number of instances of another class declared within a class</td>
</tr>
<tr>
<td>NOM (Number of Methods)</td>
<td>The number of methods in a class</td>
</tr>
<tr>
<td>SIZE1 (Lines of code)</td>
<td>The number of lines of code excluding comments</td>
</tr>
<tr>
<td>SIZE2 (Number of properties)</td>
<td>The total count of the number of data attributes and the number of local methods in a class</td>
</tr>
<tr>
<td>CHANGE (Number of lines changed)</td>
<td>The number of lines added and deleted in a class, change of the content is counted as 2</td>
</tr>
</tbody>
</table>

Table 1. Metrics definition
Table 2 and Table 3 provide descriptive statistics for the UIMS and QUES datasets respectively. It can be noticed that all data points for NOC are zeros in the QUES dataset, and thus NOC was removed from the QUES dataset. Both datasets have low medians and means for DIT, which indicate that the use of inheritance in both systems is limited. The similar medians and means for NOM and SIZE2 in both datasets suggest that both systems have similar class sizes at the design level. However, there is a significant difference in SIZE1. In addition, the medians and means for RFC and MPC in the QUES dataset is larger than those in the UIMS dataset. This indicates that the coupling between classes in QUES is higher than those in the UIMS. However, both systems have similar cohesion as suggested by the similar medians and means for LCOM in both datasets. Furthermore, the median and mean for CHANGE in the QUES dataset is larger than those in the UIMS.

At 0.05 level of significance, we performed spearman rank order correlation analysis between CHANGE and each of the independent variables. Table 4 reports the results for each dataset in terms of the correlations coefficients and p-values. All the metrics except DIT and NOC are significantly correlated with CHANGE in the UIMS dataset. However, only RFC, MPC and SIZE1 metrics are significantly correlated with CHANGE in the QUES dataset. Clearly, the correlations in the UIMS dataset are different from the correlations in the QUES dataset. In addition to the observations obtained from Table 2 and Table 3, this means that the characteristics of the UIMS dataset are different from the QUES dataset. Accordingly, and similar to [28], these two datasets are considered heterogeneous and a separate maintainability prediction model is built for each dataset.

Table 2. Descriptive statistics: UIMS dataset

<table>
<thead>
<tr>
<th>Metric</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMC</td>
<td>0</td>
<td>69</td>
<td>11.38</td>
<td>15.90</td>
</tr>
<tr>
<td>DIT</td>
<td>0</td>
<td>4</td>
<td>2.15</td>
<td>0.90</td>
</tr>
<tr>
<td>NOC</td>
<td>0</td>
<td>8</td>
<td>0.95</td>
<td>2.01</td>
</tr>
<tr>
<td>RFC</td>
<td>2</td>
<td>101</td>
<td>23.21</td>
<td>20.19</td>
</tr>
<tr>
<td>LCOM</td>
<td>1</td>
<td>31</td>
<td>7.49</td>
<td>6.11</td>
</tr>
<tr>
<td>MPC</td>
<td>1</td>
<td>12</td>
<td>4.33</td>
<td>3.41</td>
</tr>
<tr>
<td>DAC</td>
<td>0</td>
<td>21</td>
<td>2.41</td>
<td>4.00</td>
</tr>
<tr>
<td>NOM</td>
<td>1</td>
<td>40</td>
<td>11.38</td>
<td>10.21</td>
</tr>
<tr>
<td>SIZE1</td>
<td>4</td>
<td>439</td>
<td>106.44</td>
<td>114.65</td>
</tr>
<tr>
<td>SIZE2</td>
<td>1</td>
<td>61</td>
<td>13.97</td>
<td>13.47</td>
</tr>
<tr>
<td>CHANGE</td>
<td>2</td>
<td>253</td>
<td>42.46</td>
<td>61.18</td>
</tr>
</tbody>
</table>

Table 3. Descriptive statistics: QUES dataset

<table>
<thead>
<tr>
<th>Metric</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMC</td>
<td>1</td>
<td>83</td>
<td>14.96</td>
<td>17.06</td>
</tr>
<tr>
<td>DIT</td>
<td>0</td>
<td>4</td>
<td>1.92</td>
<td>0.53</td>
</tr>
<tr>
<td>NOC</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>RFC</td>
<td>17</td>
<td>156</td>
<td>54.44</td>
<td>32.62</td>
</tr>
<tr>
<td>LCOM</td>
<td>3</td>
<td>33</td>
<td>9.18</td>
<td>7.31</td>
</tr>
<tr>
<td>MPC</td>
<td>2</td>
<td>42</td>
<td>17.75</td>
<td>8.33</td>
</tr>
<tr>
<td>DAC</td>
<td>0</td>
<td>25</td>
<td>3.44</td>
<td>3.91</td>
</tr>
<tr>
<td>NOM</td>
<td>4</td>
<td>57</td>
<td>13.41</td>
<td>12.00</td>
</tr>
<tr>
<td>SIZE1</td>
<td>115</td>
<td>1009</td>
<td>275.58</td>
<td>171.60</td>
</tr>
<tr>
<td>SIZE2</td>
<td>4</td>
<td>82</td>
<td>18.03</td>
<td>15.21</td>
</tr>
<tr>
<td>CHANGE</td>
<td>6</td>
<td>217</td>
<td>64.23</td>
<td>43.13</td>
</tr>
</tbody>
</table>

Table 4. Spearman correlation results

<table>
<thead>
<tr>
<th>Metric</th>
<th>UIMS dataset</th>
<th></th>
<th>p-value</th>
<th></th>
<th>QUES dataset</th>
<th></th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMC</td>
<td>0.738*</td>
<td>0.000</td>
<td>0.084</td>
<td>0.486</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIT</td>
<td>-0.088</td>
<td>0.594</td>
<td>-0.041</td>
<td>0.734</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOC</td>
<td>0.307</td>
<td>0.058</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFC</td>
<td>0.464*</td>
<td>0.000</td>
<td>0.379*</td>
<td>0.001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCOM</td>
<td>0.760*</td>
<td>0.000</td>
<td>-0.048</td>
<td>0.691</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPC</td>
<td>0.714*</td>
<td>0.000</td>
<td>0.547*</td>
<td>0.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAC</td>
<td>0.486*</td>
<td>0.002</td>
<td>-0.191</td>
<td>0.110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOM</td>
<td>0.631*</td>
<td>0.000</td>
<td>0.049</td>
<td>0.684</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIZE1</td>
<td>0.776*</td>
<td>0.000</td>
<td>0.621*</td>
<td>0.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIZE2</td>
<td>0.581*</td>
<td>0.000</td>
<td>-0.001</td>
<td>0.995</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Significant correlation at the 0.05 level

4.3. Prediction accuracy measures

In this study, we used de facto standard and commonly used prediction accuracy measures [5, 9, 13, 21]: mean magnitude of relative error (MMRE) and prediction at level q (Pred(q)) measures. MMRE over a dataset of n observations is calculated as follows:

\[ MMRE = \frac{1}{n} \sum_{i=1}^{n} MRE_i \]

where \( MRE_i \) is a normalized measure of the discrepancy between the actual value (\( x_i \)) and the predicted value (\( \hat{x}_i \)) of observation i. It is calculated as follows:
\[ MRE_i = \frac{|x_i - \hat{x}_i|}{x_i} \]

Pred\( (q) \) is a measure of the percentage of observations whose MRE is less than or equal to \( q \). It is calculated as follows:

\[ \text{Pred}(q) = \frac{k}{n} \]

where \( k \) is the number of observations whose MRE is less than or equal to a specified level \( q \), and \( n \) is the total number of observations in the dataset. In this study, we used Pred\( (0.25) \) and Pred\( (0.30) \) because they are commonly used in the literature \([4, 5, 13, 17, 28]\).

### 4.4. Cross validation

A leave-one-out cross-validation procedure was used to evaluate the performance of the TreeNet model and the other five compared models. We used this procedure because \([21, 27]\): (i) it is closer to a real world situation than \( k \)-cross validation from a practitioner’s point of view (ii) it is deterministic (no sampling is involved); and (iii) it ensures the largest possible amount of data is used for training. In addition to the above reasons, this procedure was also used in \([28]\) and hence we will be able to compare our results against their results. The only disadvantage of this procedure, however, is that it is computationally intensive, which is not a problem in this study due to the relatively small dataset size.

In this procedure, one observation is removed from the dataset, and then each prediction model is built with the remaining \( n-1 \) observations and evaluated in predicting the value of the observation that was removed. The process is repeated each time removing a different observation. For instance, given that there are 39 observations in the UIMS dataset, all models are trained using 38 observations and then their prediction accuracy is tested on the withheld observation. This process is repeated for each observation. After that, the prediction accuracy measures are computed for each model.

### 4.5. Models construction

In this study, the TreeNet models were constructed and applied using TreeNet software by Salford Systems\(^1\). We tried to optimize the parameters of TreeNet, but in most cases the impact of varying these parameters was small and we resorted to using default parameters. The parameter settings for TreeNet are shown in Table 5.

The other five compared models (MARS, MLR, SVR, ANN, and RT) were built in \([28]\) as follows. The MARS models were built with MARS 2.0 tool using its default settings. The MLR models were built with SPSS 13.0 using stepwise selection procedure (p-value entry criterion \( \leq 0.05 \), and p-value exit criterion \( \geq 0.10 \)). The ANN, SVR, and RT models were built with WEKA 3.4.6 under its default settings.

### Table 5. Parameter settings for TreeNet

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn rate</td>
<td>Auto</td>
</tr>
<tr>
<td>Subsample fraction</td>
<td>1.00</td>
</tr>
<tr>
<td>Influence trimming factor</td>
<td>0.10</td>
</tr>
<tr>
<td>M-regression breakdown</td>
<td>0.99</td>
</tr>
<tr>
<td>Regression loss criterion</td>
<td>Huber-M</td>
</tr>
</tbody>
</table>

### 4.6. Results and discussion

Table 6 shows the results of the prediction accuracy measures achieved by the TreeNet model and each of the five compared models for the UIMS dataset within leave-one-out cross-validation. The results of the five compared models are obtained from \([28]\). It can be observed that the TreeNet model has achieved the best MMRE value (1.57). The SVR model has achieved the second best MMRE, while the RT model has the worst MMRE. In addition, the TreeNet model has the best Pred\( (0.25) \) and Pred\( (0.30) \) values, i.e. 31% and 41% respectively. The SVR has also achieved 31% in Pred\( (0.25) \). However, the TreeNet model has yielded 5% improvement in Pred\( (0.30) \) over the best of the other five models, which is the SVR model.

### Table 6. Prediction accuracy: UIMS dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>MMRE</th>
<th>Pred( (0.25) )</th>
<th>Pred( (0.30) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>TreeNet</td>
<td>1.57</td>
<td>0.31</td>
<td>0.41</td>
</tr>
<tr>
<td>MARS</td>
<td>1.86</td>
<td>0.28</td>
<td>0.28</td>
</tr>
<tr>
<td>MLR</td>
<td>2.70</td>
<td>0.15</td>
<td>0.21</td>
</tr>
<tr>
<td>SVR</td>
<td>1.68</td>
<td>0.31</td>
<td>0.36</td>
</tr>
<tr>
<td>ANN</td>
<td>1.95</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>RT</td>
<td>4.95</td>
<td>0.10</td>
<td>0.10</td>
</tr>
</tbody>
</table>

To further analyze the results, we sorted the observations of the UIMS dataset in the ascending order of actual CHANGE (i.e., the dependent

---

\(^1\) http://www.salfordsystems.com/TreeNet.php
Figure 1. Plots of actual and predicted values for each model: UIMS dataset

variables), and then plotted these actual values and predicted values by each model. Figure 1 shows the plots for the 39 observations in the UIMS dataset. This reveals some interesting observations. First, the plotted line of predicted values by the TreeNet model is smoother and less fluctuated than the other five models. Second, the TreeNet model has slightly overestimated the observations with low actual values, and slightly underestimated the observations with high actual values. Third, the predicted values by the TreeNet model are very close to the actual values of Case 1 to Case 34, but are poor for the outlier cases (Case 35 to Case 39). Finally, out of the five compared models, the plotted line of predicted values by the RT model is the most similar to the plotted line by the TreeNet model. Clearly, the TreeNet model improves the accuracy of the RT model.

Table 7 shows the results of the prediction accuracy measures achieved by the TreeNet model and each of the five compared models for the QUES dataset within leave-one-out cross-validation. The results of the five compared models are obtained from [28]. Of all the models, the TreeNet model has achieved the best Pred(0.25) and Pred(0.30) values, and the second best MMRE value. It yields 10% and 6% improvement in Pred(0.25) and Pred(0.30),
and less fluctuated than the other five models. Observations in the QUES dataset after sorting the TreeNet model is just 0.10 more than the MMRE that has been achieved by the MARS model.

Similarly, Figure 2 shows the plots for the 71 observations in the QUES dataset after sorting the observations in the ascending order of actual CHANGE. It can be also observed that the plotted line of predicted values by the TreeNet model is smoother and less fluctuated than the other five models.

![Figure 2. Plots of actual and predicted values for each model: QUES dataset](image)

Table 7. Prediction accuracy: QUES dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>MMRE</th>
<th>Pred(0.25)</th>
<th>Pred(0.30)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TreeNet</td>
<td>0.42</td>
<td>0.58</td>
<td>0.65</td>
</tr>
<tr>
<td>MARS</td>
<td>0.32</td>
<td>0.48</td>
<td>0.59</td>
</tr>
<tr>
<td>MLR</td>
<td>0.42</td>
<td>0.37</td>
<td>0.41</td>
</tr>
<tr>
<td>SVR</td>
<td>0.43</td>
<td>0.34</td>
<td>0.46</td>
</tr>
<tr>
<td>ANN</td>
<td>0.59</td>
<td>0.37</td>
<td>0.45</td>
</tr>
<tr>
<td>RT</td>
<td>0.58</td>
<td>0.41</td>
<td>0.45</td>
</tr>
</tbody>
</table>
When excluding the TreeNet model and considering the results achieved by the other five models (MARS, MLR, SVR, ANN, and RT), it can be observed that the SVR model outperforms the other four models in the UIMS dataset only (see Table 6), whereas the MARS model outperforms the other four models in the QUES dataset only (see Table 7). On the other hand, the TreeNet model outperforms the other five models in both UIMS and QUES datasets in Pred(0.25), and Pred(0.30). Moreover, the TreeNet model has achieved the best MMRE in the UIMS dataset, and the second best MMRE in the QUES dataset. This suggests that the TreeNet model has good generalization capabilities.

As suggested criteria [4, 17], an effort prediction model is considered accurate if its Pred(0.25) ≥ 0.75 or its Pred(0.30) ≥ 0.70. However, it is also reported that the prediction accuracy of software maintenance effort prediction models is often low and thus it is very difficult to satisfy the criteria [5]. It can be noticed from Table 6 and Table 7 that none of the prediction models satisfy the criteria. However, the TreeNet model has achieved improved Pred(0.25) and Pred(0.30) over the other models in both datasets, and its results in the QUES dataset are closer to the criteria.

We also calculated the percentage of observations that have been underestimated and those that have been overestimated by each model for each dataset. Figure 3 illustrates the results. In the UIMS dataset, it has been found that the TreeNet, MARS, SVR, and ANN models tend to slightly overestimate; and the MLR and RT models tend to significantly overestimate. In the QUES dataset, it has been found that the TreeNet model tends to slightly underestimate; the MARS, SVR, ANN, and RT models tend to slightly overestimate; and the MLR model tends to significantly overestimate. This suggests that the TreeNet model does not have a significant tendency toward either underestimation or overestimation.

5. Conclusion

This paper has empirically evaluated the capability of TreeNet in predicting object-oriented software maintainability. It has applied TreeNet using two datasets and compared its prediction performance against recently published object-oriented software maintainability prediction models (MARS, MLR, SVR, ANN, and RT). The results indicate that improved, or at least competitive, prediction accuracy has been achieved when applying the TreeNet model. The TreeNet model has achieved improved prediction accuracy in terms of Pred(0.25) and Pred(0.30) in both datasets. Furthermore, the TreeNet model has achieved the best MMRE in one dataset, and the second best MMRE in the other dataset. The results therefore reveal the effectiveness of TreeNet in predicting object-oriented software maintainability, and thus suggest that it can be a useful and practical addition to the framework of software quality prediction.

One direction of future work would be conducting additional empirical studies with other datasets to further support the findings of this paper, and to realize the full potential and possible limitation of TreeNet. We are currently investigating the application of TreeNet in software fault prediction as an extension to our pervious work in [7]. We are also planning to investigate the capability of TreeNet in other software engineering classification and regression problems such as effort and cost estimation.
Acknowledgement

The authors would like to acknowledge the support of King Fahd University of Petroleum and Minerals in the development of this work.

References


