Numeric Prediction Algorithms for Bridge Corrosion

Yousheng Cheng, Kansas State University, Manhattan, USA, (ych6565@ksu.edu)
Hani G. Melhem, Kansas State University, Manhattan, USA, (melhem@ksu.edu)

Summary
The research reported in this article was conducted to mainly explore the two common numeric prediction techniques, the model tree and the regression tree, when used in conjunction with bagging as a wrapper method. Bagging is used to improve the prediction accuracy of these two algorithms, and results are compared with the ones obtained earlier by the k-nearest neighbor (*KNN*) algorithm. From the conducted experiments, both the bagged regression tree and bagged model tree produce better results than not only their corresponding regression tree and model tree alone, but also the *KNN* with optimal value of k equal to 7. In addition, the bagged model tree yields the lowest prediction errors and a highest correlation coefficient of 0.81. It is demonstrated that it is feasible to use the bagged model tree for engineering applications in prediction problems such as estimating the remaining service life of bridge decks.

1 Introduction
A couple of previous research studies on the application of machine learning techniques to bridge deck deterioration have been conducted and were published earlier. In the first study, the k-nearest-neighbor (*KNN*) method was applied to the prediction of the remaining service life of bridge decks based solely on the degree of corrosion of the reinforcing steel bars (Melhem and Cheng 2003a). The dataset used was extracted from a set of typical deck survey reports of the Kansas Department of Transportation (*KDOT*), and results were compared to those obtained using the C4.5 inductive learning algorithm (programming was done in C++). The remaining service life was treated as a symbolic class value, and was discretized into eleven consecutive values (of the type "between 10 and15 years"). In the second study, the decision tree algorithm with different wrapper methods was used to model general bridge deck deterioration (Melhem and Cheng *et al*. 2003b). The dataset was from the *KDOT* inspection electronic database normally used for the Pontis bridge management system (PONTIS 2001) rather than from field survey reports. The measure of bridge deck deterioration was the Health Index, which was also treated as discrete class value because the decision tree algorithm is used for symbolic class value classification. Experiments with the inductive decision tree algorithm were done using Weka (Frank et al., 2000), which is a collection of machine learning programs developed at the University of Waikato, New Zealand.

The data set used in this study is from the same field survey reports of bridge decks used in the first study. The class/target value is the predicted remaining service life, taken as a numeric value, which is more suitable for the continuous space representation of the number of service years, rather than symbolic value as in the first study. Based on the bagging algorithm, the outcomes over individual predictors are averaged as the final prediction of a true (numeric) target value.

2 Regression tree and model tree
Trees used for numeric prediction are a special type of decision trees that deal with a continuous goal variable. The types of trees are divided into regression tree (RT) and model tree (MT),

according to the way of representing a target/class value at each leaf (Witten and Frank 2000). The difference between RT and MT is that RT stores a target/class value that represents the average (constant) value of the cases that arrive at a leaf, whereas MT uses a linear regression model to predict the class value of the cases that get to the leaf. A linear form is normally assumed for the unknown regression function and the parameters (coefficients) of the model are estimated using a least squares criterion (Draper & Smith 1981). The estimation of the parameters is accomplished by solving a set of linear equations. Therefore, for RT the prediction at each leaf is expressed as:

$$e = \frac{\sum_{i=1}^{n} a_i}{n} \tag{1}$$

where e is the prediction value, n is the number of instances reaching the specific leaf, and a_i is the actual target value of the i^{th} instance at the leaf. For MT, the prediction at each leaf is represented by

$$e = c_0 + \sum_{i=1}^{n} c_i a_i \tag{2}$$

where e, n, and a_i are the same as in Eq. (1), and, c_i ($i=0, 1, \ldots, n$) are the coefficient to be solved for.

2.1 Constructing the regression tree and model tree

RT and MT are first constructed using a decision tree induction algorithm to generate an initial tree. Decision tree algorithms split the attribute so as to maximize the information gain, whereas the splitting criterion used for RT and MT is based on treating the standard deviation of the class values in the portion L of the learning data that gets to a particular node as a measure of the error at that node, thus evaluating the expected decrease in error resulting from testing each attribute at that node. The best split of the attribute is taken to be the one that maximizes the expected decrease of error. The expected error is measured by standard variance, and the expected error decrease is expressed using standard variance decrease. These are evaluated by Equation (3) and (4), respectively (Witten and Frank 2000):

$$\sigma^2(L) = \frac{\sum_{i=1}^{N} (y_i - \mu)^2}{N} \tag{3}$$

$$svd = \sigma^2(L) - \sum_i \frac{|L_i|}{|L|} \times \sigma^2(L_i) \tag{4}$$

where $\sigma^2(L)$ is the standard variance of the class values of the instances in the learning set L, N is the size of the learning set, y_i denotes the class value of the i^{th} instance, μ is the mean of the class values of the N instances, svd stands for the standard variance decrease, and L_1, L_2,… represent the subsets that result from splitting the node according to the selected attribute. The splitting is recursively done according to the rule of maximizing the expected error decrease, and keeps going till either the class values of the instances arriving at a node change very slightly, i.e., their standard variance is only a small fraction of the standard variance of the original instance set, or only very few instances remain.

2.2 Handling missing values

A revision is made to the standard variance decrease equation to take into consideration any missing values. Including the missing value compensation, Eq.(4) becomes

$$svd = \frac{k}{|L|} \times [\sigma^2(L) - \sum_{i \in \{LT, RT\}} \frac{|L_i|}{|L|} \times \sigma^2(L_i)], \quad (5)$$

where L is the set of instances that reach the node, L_{LT} *and* L_{RT} are sets in the left and right branches, respectively, that result from splitting on the attribute, and k is the number of instances without missing values for that attribute. All splits on attributes are binary.

2.3 Pruning the tree

When a model tree is constructed, a linear model is needed for each interior node of the tree, not just at the leaves. A model is evaluated for each node of the unpruned tree prior to pruning. The model takes the form $c_0 + c_1 a_1 + c_2 a_2 + ... + c_n a_n$, where $a_1, a_2, ..., a_n$ are attribute values. The coefficients $c_0, c_1, c_2, ..., c_n$ are solved using standard regression. However, only the attributes that are tested in the subtree below this node are used in the regression, since the other attributes that influence the predicted value have been considered in the tests that lead to the node. After the linear model is evaluated for each interior node, the tree is pruned back from the leaves, as long as the estimated expected error decreases.

2.4 Evaluating Prediction

The evaluation measures are different for classification (symbolic class values) than for numeric prediction (continuous class values). For the latter the basic quality measure, given by accuracy percentage or error rate, is not appropriate since errors are not simply "hit" or "miss". Let $e_1, e_2, ..., e_n$, stand for the estimated (predicted) values of instances on the test dataset, and $a_1, a_2, ..., a_n$, denote the actual values of the instances. Several methods can be used to evaluate the quality of numeric predictions. For instance, the root mean-squared error, *rmse,* is the principal and most commonly used method, and is given by

$$rmse = \sqrt{\frac{(e_1 - a_1)^2 + ... + (e_n - a_n)^2}{n}} \quad (6)$$

Another alternative is the mean absolute error, *mae*. It considers the average of the individual errors not considering their sign, and is calculated as

$$mae = \frac{|e_1 - a_1| + ... + |e_n - a_n|}{n} \quad (7)$$

Mean-squared error tends to exaggerate the influence of the instances whose prediction error is larger than the others, but the absolute error does not have this effect, since all sizes of error are dealt with equally according to their magnitude. These two methods measure the absolute errors, but sometimes the averages of absolute error will be meaningless. Therefore, the methods measuring relative errors are of importance. The measures commonly used include the root relative squared error, *rrse*, and the relative absolute error, *rae*, which are evaluated by Eq. (8), and (9), respectively,

$$rrse = \sqrt{\frac{(e_1 - a_1)^2 + ... + (e_n - a_n)^2}{(a_1 - \chi)^2 + ... + (a_n - \chi)^2}} \quad (8)$$

$$rae = \frac{|e_1 - a_1| + \ldots + |e_n - a_n|}{|e_1 - \chi| + \ldots + |e_n - \chi|} \tag{9}$$

$$\chi = \frac{\sum_i a_i}{n} \tag{10}$$

The last method is called the correlation coefficient, which measures the statistical correlation between the actual values and the predictions. The correlation coefficient, *cc*, is computed as

$$cc = \frac{\sigma_{PA}^2}{\sigma_P^2 \sigma_A^2} \tag{11}$$

in which:

$$\sigma_{PA}^2 = \frac{\sum_i (e_i - \lambda)(a_i - \chi)}{n-1} \tag{12}$$

$$\sigma_P^2 = \frac{\sum_i (e_i - \lambda)^2}{n-1} \tag{13}$$

$$\sigma_A^2 = \frac{\sum_i (a_i - \chi)^2}{n-1} \tag{14}$$

$$\lambda = \frac{\sum_i e_i}{n} \tag{15}$$

where χ is the same as in Eq. (8) and (9). The correlation coefficient ranges from 1 for perfect correlation, through 0 for no correlation at all, to –1 when there are completely negatively correlated outcomes. Negative values should not occur for reasonable prediction measures. Good performance results in a large value of the correlation coefficient, and small error rate. All the above methods were used to measure the prediction quality of the learning algorithms reported in this article.

3 Bagging

The idea behind bootstrap aggregating or bagging (Breiman 1994) is to generate multiple predictors and use these to get an aggregated predictor. The aggregation averages over these individual predictors when predicting a real-valued (numeric) outcome, and votes when classifying a discrete-valued (symbolic) target. The multiple predictors are formed as follows: Given a learning set, *L*, consisting of *m* instances, the multiple predictors of size *N* are generated by replicate datasets drawn randomly from the original set *L* with replacement. Each replicate dataset has the same size *m* as the original set *L*, but some instances may not appear while others may appear more than once. A point has been made (Breiman, 1994) that the critical factor is the instability of the prediction method: bagging can improve accuracy if a little disturbance in *L* can lead to different predictors being constructed; in other words, the prediction method should be unstable for bagging to work well.

The computation procedure for bagging is the following:

(1) Given a dataset $D = \{ (x_1, y_1), (x_2, y_2), ..., (x_m, y_m) \}$ of m instances where x_i is an instance, and y_i is the target value relative to x_i. The dataset is divided into a test set T and learning set L at random.

(2) A bootstrap sample L_b is randomly drawn from L, and a tree is generated using L_b and 10-fold cross-validation. This is repeated N times, resulting in the tree predictors $C_1(x), C_2(x), .., C_N(x)$.

(3) For a given instance x, its estimated numeric target value is the average over the $C_1(x), C_2(x), .., C_N(x)$. The differences between the estimated target value and the actual value are called the prediction error.

(4) The random division of the data set is repeated, for example, 10 times, and the average prediction error over the 10 iterations is reported.

4 Description of data

In the bridge engineering domain, the deterioration of bridge decks directly affects the remaining service life of the deck. The bridge deck data and the methodology for determining the remaining service life of concrete elements with respect to corrosion-induced deterioration are described in Melhem and Cheng (2003a). The development of the methodology is given in Nagaraja (1997). The attributes, originally extracted from field bridge deck survey reports from the Kansas Department of Transportation and used in the study, are listed in Table 1.

Table 1. Attributes/Class and their Actual Value Ranges

Attributes/Class	**Actual Ranges of Attributes**
YearBuilt	1907-1991
DeckArea	842.8-104860 (sq-ft)
AreaSpall	0-53.06 (sq-ft)
DelaminatedArea	0-190.01 (sq-ft)
Cover	0.67-4.25 (in)
CriticalRebarChloride	0-100 (%)
SurfaceChloride	0.01-0.56 (%)
YearCFS	0-72 (years)
RemainingServiceLife (Class)	1.5 – 65 (Years)

The bridge deck survey reports initially contained a variable number of data points that were averaged to obtain the attribute values for Cover, CriticalRebarChloride, and SurfaceChloride. All of the attributes used are numeric (continuous), and their actual ranges are listed in Table 1. Factors such as concrete type, compressive strength, and water/cement ratio, service conditions (traffic volume, truck loads, etc.), and environmental conditions (temperature and relative humidity/moisture content), which all influence the deterioration of bridge decks, are not included in this study. The reasons for not including such factors are given by Melhem and Cheng (2003a). MT and RT are applied to the domain using only one class, with the mean values (numeric) of the discretized continuous space taken as the final decision (This is shown later in Table 4, in which column (2) gives the original class values and column (3) gives the corresponding class values used in this study).

5 Experiments

Experiments were conducted using Weka (Frank et al., 2000) with the following five algorithms: (1) regression tree, RT, (2) regression tree with bagging (called bagged regression tree or BRT), (3) model tree, MT, (4) model tree with bagging (called bagged model tree or BMT), and (5) *KNN*. Details about *KNN* algorithm were given in (Melhem and Cheng 2003a).

5.1 Exploring the five algorithms

The data set used from the deck survey reports includes 295 instances, and each instance has 8 attributes, all of which are continuous/numeric. The goal consists of predicting the remaining service life of bridge decks. The experiments were done using the following methodology: each test data was selected at random to eliminate any ordering effects, and a ten-fold cross validation evaluation was carried out on the selected data. Five measures (Eq.6 through 11) were recorded in each iteration, and the average over the ten iterations is taken as the final value for each measure. The final values are used to measure the prediction quality of the different algorithms. The pruning feature was applied in all runs for the four tree algorithms. The discussion of the results is presented below.

The research results obtained by the four tree algorithms are summarized in Table 2. It shows that the model of the regression tree over ten-fold cross validation leads to mean absolute error of 6.07, root mean squared error of 9.86, relative absolute error of 67.37%, root relative squared error of 76.79%, and correlation coefficient of 0.65. While combining RT and bagging, the values of the above five measures are slightly but obviously improved. As expected, the results obtained indicate that the larger the correlation coefficient, the smaller the errors. Based on the same dataset and ten-fold cross validation, the MT produces less error values than RT alone. As bagging has shown to be beneficial when combined to RT, the combination of MT and bagging also yields better results than MT alone. This last combination produces the lowest error rate (by all four measurements) and the highest correlation coefficient.

Table 2 Results Obtained Using the Four Individual Models

Measure	Model			
	RT	Bagged RT	MT	Bagged MT
Mean absolute error	6.07	4.80	5.43	4.47
Root mean squared error	9.86	8.03	9.16	7.54
Relative absolute error	67.37%	53.29%	60.35%	49.67%
Root relative squared error	76.79%	62.57%	71.38%	58.78%
Correlation coefficient	0.65	0.79	0.70	0.81

Table 3 shows a summary of the results obtained using *KNN* algorithms by setting a series of k equal to 1 up to 10. It can be seen from Table 3 that all results improve with the increase of k from 1 till 7 (except for $k=5$), and degrade beyond that. This means that the best results are attained at $k = 7$, which is usually called the k's optimal value. With k equal to 7, the *KNN* over ten-fold cross validation results in a mean absolute error of 5.15, a root mean squared error of 8.15, a relative absolute error of 61.02, a root relative squared error of 67.749%, and a correlation coefficient of 0.74.

Table 3 Results Obtained by *KNN*

Measures	*k*									
	1	2	3	4	5	6	7	8	9	10
mae	6.4	5.6	5.4	5.3	5.3	5.2	5.2	5.3	5.3	5.4
rmse	10.5	9.2	8.8	8.6	8.6	8.4	8.2	8.3	8.4	8.5
rae (%)	75.4	65.9	63.8	62.9	63.1	61.9	61.0	63.2	64.0	63.6
rrse (%)	87.6	76.1	73.4	71.4	71.8	69.7	67.7	69.0	70.2	70.2
cc	0.62	0.68	0.70	0.72	0.71	0.72	0.74	0.73	0.72	0.71

Note: *mae*: mean absolute error; *rmse*: root mean squared error; *rae*: relative absolute error; *rrse*: root relative squared error; *cc*: correlation coefficient

5.2 Analysis of the results

For the given dataset used in this study, the metric of mean absolute error is considered more meaningful than any of the others. This can be deduced by comparing the original class/target values and their respective predictions. Table 4 shows the mean absolute error, the original class/target values (continuous spaces), the "true" (or numeric) class/target values used here (averaged over original continuous spaces), and the rough predictions, where predictions are obtained by the numeric target value plus the mean absolute error. The predictions and the *mae* are listed for the models giving the best results, namely the bagged MT and *KNN* with the optimal value of *k* (*k*=7). As seen in Table 4, the larger the class values, the more accurate the predictions are. This is due to the conversion (averaging over the original spaces) and the different intervals of the original spaces represented in this application. Therefore, this may not be true for other engineering problems and cannot be generalized.

Table 4 Comparisons of bagged MT (*BMT*) to *KNN* (*k*=7)

Mean absolute Error (1)	(2)	Original Range (3)	"True" Targets (4)	Predictions By *BMT* (5)=(4)+(1)	Predictions By *KNN* (6)=(4)+(2)
BMT	*KNN* (*k*=7)	0-3	1.5	5.97	6.65
		3-6	4.5	8.97	9.65
		6-10	8	12.47	13.15
		10-15	12.5	16.97	17.65
		15-20	17.5	21.97	22.65
4.47	5.15	20-25	22.5	26.97	27.65
		25-30	27.5	31.97	32.65
		30-40	35	39.47	40.15
		40-50	45	49.47	50.15
		50-65	57.5	61.97	62.65
		65 up	65	69.47	70.15

The best model, bagged MT, was chosen for discussing the prediction capabilities of the algorithm. Figure 1 is part of a pruned training binary model tree with 40 rules generated by the bagged MT scheme. It contains 8 rules/leaves denoted by *LM*. *LMi* is used to represent the prediction stored in the i^{th} leaf. Among the predictions stored in the 8 leaves, only four (*LM2*, *LM6*, *LM7,* and *LM8)* are linear functions. The others (*LM1, LM3, LM4, LM5*) are constant, i.e., all the coefficients in Eq. (2) equals zero except c_0. Note also that not all the 8 attributes are

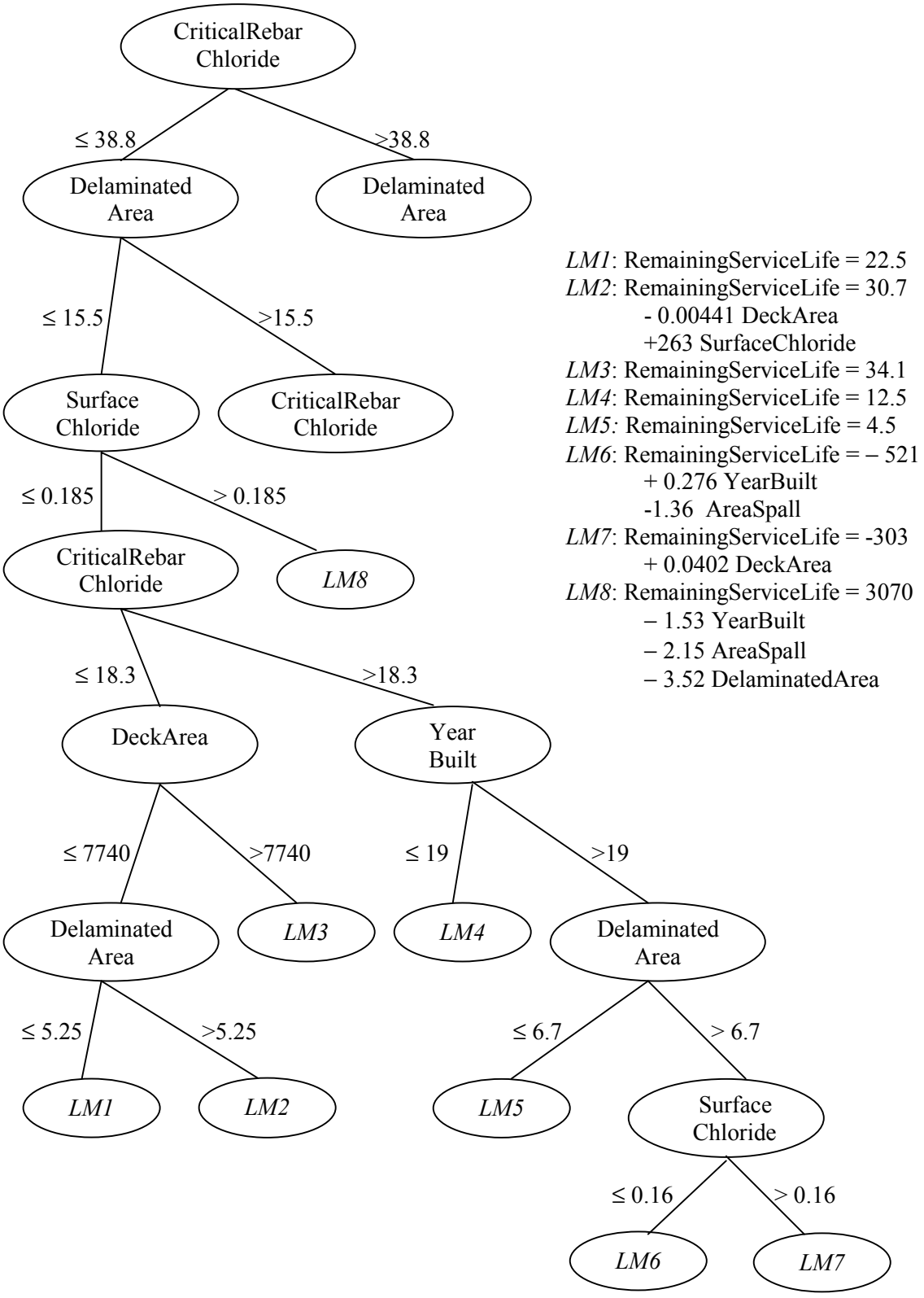

Figure 1. Part of a Pruned Bagged Model Tree

included in the prediction functions of *LM2*, *LM6*, *LM7*, and *LM8*. In other words, even for the linear functions, some of the coefficients in Eq. (2) are equal to zero.

The selected model tree shown in Fig. 1 can be used as a model for predicting of the remaining service life of a bridge deck. For example, a certain bridge deck from the data set used has the following values of some attributes: "CriticalRebar Chloride" = 0.00%, "Delaminated Area" = 0.00, and "Surface Chloride"= 0.22%. According to Figure 1, one would traverse down the tree starting from the root node "CriticalRebar Chloride", and due to its value of 0.00%, which is less than 38.8%, take the left branch and reach the second level node "Delaminated Area". With a value of 0.00, which is smaller than 15.5, one will further go again along the left branch, and arrive at the third level node "Surface Chloride", whose value (0.22%) is larger than 0.185%, and therefore take the right branch and get to the leaf called *LM8*. This leaf gives the predicted remaining service life of the bridge deck (according the formula shown to the right of the Figure for *LM8*).

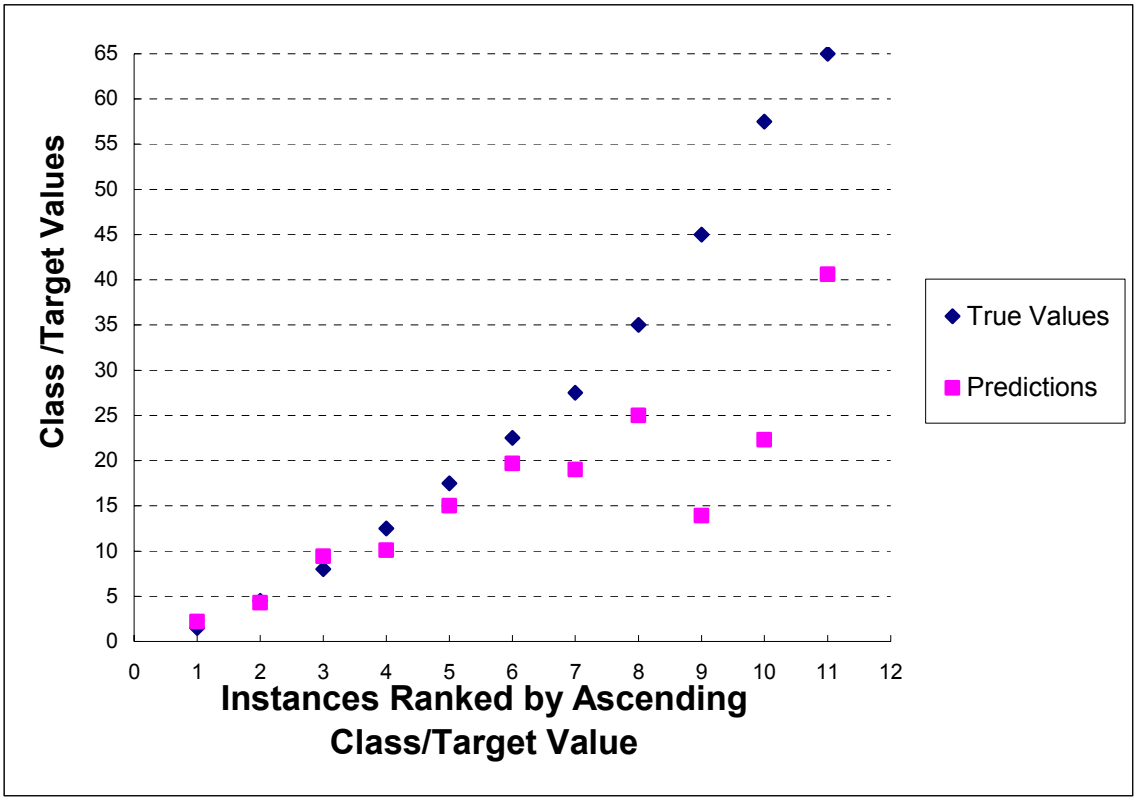

Figure 2. Comparison of Predictions with "True" Values for 11 Typical Instances

To analyze the predictions generated by the bagged MT, eleven instances with different target values were selected for testing. The predictions and numeric target values of these 11 instances are shown in Figure 2 where the horizontal axis represents instances ranked by ascending true (numeric) class/target values, and vertical axis denotes the predictions and actual target values, where the "square" stands for the former, and the "diamond" denotes the latter. It can be seen that the predictions for the instances with target values below 27.5 are very close to the true values, while the predictions for the instances with values beyond 27.5 (inclusive) are far from their true class values. This is due to the fact that the instances (number of bridge decks) with class values (remaining service life) larger than or equal to 27.5 are very few in the data set. Among the 295 instances, the

number of instances with the target values of 27.5, 35, 45, 57.5, and 65 is 7, 10, 3, 4, and 5, respectively, for a total of 29 cases, which make up about 9.8 percent of the data set.

6 Conclusions

This paper explored the numeric prediction of the regression tree and model tree algorithms, investigated whether their performance can be improved by bagging, and compared their performance to that of the k-nearest neighbor (*KNN*) algorithm. The following conclusions can be made in accordance with the experiments conducted:

- Generally speaking, the model tree gave better predictions than the regression tree. When bagging is used with either of the basic tree models, the bagged model tree remains better than the bagged regression tree regardless of which error measure is used.
- The *KNN* algorithm achieves the optimal performance at k=7. It is more efficient than the model tree and regression tree, but worse than the bagged model tree and bagged regression tree.
- Bagging consistently improves the numeric prediction. It leads to a decrease of 1.99 in the mean absolute error and an increase of 0.14 in the correlation coefficient when combined to the decision tree, and a decrease of 0.96 in the error and an increase of 0.11 in the correlation when combined to the model tree. Bagging made the regression tree (which was not as good) more efficient than the model tree.
- For problems with class values that are numeric in nature such as the number of remaining service years investigated in this study, the measure of mean absolute error is preferred over all others since the original target value (the remaining service life) should actually be a continuous space. The "true" (numeric) value used here is the numeric average of the initially discretized ranges. Consequently, the final results from the prediction plus/minus the mean absolute error either exactly fall into, or are very close to the numeric range.
- The best algorithm is the bagged model tree, which yields the highest correlation coefficient of 0.81, and the lowest mean absolute error of 4.47.

It should be noted that neither the best correlation coefficient nor the lowest mean absolute error obtained in this study is ideal. Future research emphasis is placed on two things: First is the enhancement of the dataset by both increasing the size of the dataset and including some other important attributes. Second is the improvement of the numeric prediction algorithms. Two or more methodologies may be integrated rather than combining the individual predictions. For example, the k-nearest-neighbor, or the Kernel model may be used instead of the linear model in the model tree algorithm. Also, other regression methods such as fitting exponential and quadratic regression may be used in the model tree instead of the linear regression.

7 References

Breiman, L. (1994). "Bagging predictors." *Technical Report No. 421*, Department of Statistics, University of California, Berkeley, California 94720.

Draper, N. R., and Smith, H. (1981). *Applied Regression Analysis*. 2nd edition, John Wiley.

Mahoui, A., Seewald, A.K, Kibriya, A.M., Pfahringer, B., Frank, E., Schmidberger, G., Witten, I.H., Lindgren, J., Boughton, J., Wells, J., Trigg, L., Coelho, L.S., Ware, M., Hall, M.,

Bouckaert, R., Kirkby, R., Butler, S., Legg, S., Inglis, S., Roy, S., Voyle, T., Xu, X., Wang, Y., and Wang, Z. (2000). *Weka 3 - Machine Learning Software in Java*. Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Melhem, H. G. and Cheng, Y. (2003a). "Prediction of remaining service life of bridge decks using machine learning." *Journal of Computing in Civil Engineering,* 17 (1), 1-9.

Melhem, H. G., Cheng, Y., Kossler, D., and Scherschligt, D. (2003b). "Wrapper Methods for Inductive Learning: Example Application to Bridge Decks." *Journal of Computing in Civil Engineering,* 17 (1), 46-57.

PONTIS (2001), *Pontis Bridge Management*, Release 4, Technical Manual, American Association of State Highway and Transportation Officials, Washington, D.C.

Nagaraja, S. (1997). "Bridge deck rebar-corrosion knowledge based decision system development using machine learning techniques." PhD dissertation, Kansas State University, Manhattan, Kansas.

Witten, I.H., and Frank, E. (2000). *Data Mining*. Morgan Kaufmann Publisher, Inc.