Cost-sensitive Global Model Trees applied to loan charge-off forecasting

Marcin Czajkowski a,⁎, Monika Czerwonka, Marek Kretowski b

a Faculty of Computer Science, Białystok University of Technology, Wiejska 45a, 15-351 Białystok, Poland
b Collegium of Management and Finance, Warsaw School of Economics, Al. Niepodległości 162, 02-554 Warsaw, Poland

A R T I C L E   I N F O

Article history:
Received 17 April 2014
Received in revised form 10 February 2015
Accepted 30 March 2015
Available online 9 April 2015

Keywords:
Cost-sensitive regression
Model trees
Evolutionary algorithms
Asymmetric costs
Loan charge-off forecasting

A B S T R A C T

Regression learning methods in real world applications often require cost minimization instead of the reduction of various metrics of prediction errors. Currently in the literature, there is a lack of white box solutions that can deal with forecasting problems where under-prediction and over-prediction errors have different consequences. To fill this gap, we introduced the Cost-sensitive Global Model Tree (CGMT), which applies a fitness function that minimizes an average misprediction cost. Proposed specialized genetic operators improve searching for optimal tree structure and cost-sensitive linear regression models in the leaves. Experimental validation is performed on loan charge-off data. It is known to be a difficult forecasting problem for banks due to the asymmetric cost structure. Obtained results show that specialized evolutionary algorithm applied to model tree induction finds significantly more accurate predictions than tested competitors. Decisions generated by the CGMT are simple, easy to interpret, and can be applied directly.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

A lot of real world problems are cost-sensitive, which means that different types of prediction errors are not equally costly [5]. As a result, typical minimization of prediction errors is not the best scenario. A cost-sensitive term encompasses all types of learning where cost is considered [50,19], and different types of costs (e.g., costs of attributes, cost of instances, and costs of errors) can be distinguished. The current research focuses on a single cost for decision making, however, multiple costs are also investigated [35].

For example, in medical diagnoses, there are several types of costs that can be minimized, such as the cost of misclassification (e.g., overlooking an ill patient can be fatal in contrast to a false positive test) or the cost of treatment (e.g., financial or risk). When speculating on stock exchange, investors directly compare future gains and losses and usually give more weight to losses. Researchers show that potential gains need to be approximately twice as large to offset potential losses [51]. As a consequence, investors tend to realize their gains more often than their losses as they sell winning stocks more readily. There are many other examples for such asymmetry, such as in bankruptcy prediction [57], behavioral finances [45], expected stock returns [2], criminal justice settings [6], physician prognostic behavior [1], product recommendations [32], and so on.

Cost-sensitive regression is still not adequately addressed in the data mining literature, as most existing research in this area deals with classification problems. Conventional algorithms usually operate with symmetric loss functions and minimize absolute or squared errors that do not distinguish differences between under-prediction and over-prediction, as each is weighted equally (the cost of under-prediction and over-prediction is equal). There is a need for solutions with asymmetric loss functions that can successfully forecast cost-sensitive regression problems. Such models also minimize absolute or squared errors, however, the under-predicted and over-predicted instances have different weights that depend on the costs.

Our study makes several important contributions to the literature. First, we propose a new method called the Cost-sensitive Global Model Tree (CGMT), which extends the cost-neutral solution called GMT [17]. By applying evolutionary algorithms (EA) in the model tree induction, we managed to successfully search for optimal tree structure and cost-sensitive regression models in the leaves under different asymmetric loss functions. What is more, CGMT predictions on loan charge-off forecasting data as one of the cost-sensitive problems faced by banks are significantly more accurate than the results of their tested counterparts.

Next, the hierarchical tree structure, in which appropriate tests from consecutive nodes are sequentially applied, closely resembles a human way of decision making. Therefore, the CGMT prediction model is natural and easy to understand and interpret, which is extremely important in financial forecasting. Finally, we propose an improvement of existing algorithms in terms of their performance. In solutions proposed in [5] and [56], the linear tuning function is calculated by the heuristic approach (hill climbing algorithm). We managed to find a direct minimization that returns the exact value of an adjusted regression model. The proposed approach significantly extends upon previously performed research on cost-sensitive extensions for GMT [16].

http://dx.doi.org/10.1016/j.dss.2015.03.009
0167-9236/© 2015 Elsevier B.V. All rights reserved.
particular, proposed solution can work with any convex function. New specialized variants of genetic operators were proposed, the fitness function was improved and mode detailed experimental analysis was performed.

The rest of the paper is organized as follows. The next section presents background information and Section 3 proposes the CGMT approach. The experimental evaluation is performed in Section 4 and the paper is concluded in the last section, in which future work is also discussed.

2. Background

In this section, we want to present some background information about decision trees for regression and asymmetric loss function as well as the problem of loan charge-off forecasting.

2.1. Regression and model trees

The most common predictive tasks in data mining are classification and regression, and the decision tree [42] is one of the most frequently applied prediction techniques. Tree-based approaches are easy to understand, visualize, and interpret. Their similarity to the human reasoning process makes them a powerful tool [29] among data analysts.

The problem of learning the optimal model tree is known to be NP-complete [39]. Consequently, practical decision-tree inducers are based on heuristics such as the greedy approach, where locally optimal decisions are made in each node. This process is known as recursive partitioning [41]. Two main variants of decision trees can be distinguished by the type of problem they are applied to. Tree predictors can be used to classify existing data (classification trees) or to approximate real-valued functions (regression trees). In each leaf, the classification tree assigns a class label (usually a majority class of all instances that reach that particular leaf), while the regression tree holds a constant value (usually an average value for the target attribute). The model tree can be seen as an extension of the regression tree. The most important difference is that the constant value in each leaf of the regression tree is replaced in the model tree by the linear (or nonlinear) regression function. An example of classification, regression, and model tree induced by the top-down greedy approach is illustrated in Fig. 1.

To limit the negative effects of greedy induction, multiple authors have proposed various techniques [36,54]. However, the true global approach for decision tree induction was possible with evolutionary computation. In the literature, there are attempts to apply the evolutionary approach for the induction of decision trees [4], but only a few solutions concern the regression problem. In TARGET [20], the authors propose to evolve a CART-like regression tree with simple operators and the Bayesian Information Criterion (BIC) [43] as a fitness function.

Evolutionary induced regression trees with linear models in the leaves were proposed in a solution called E-Motion [3]. The authors applied a standard 1-point crossover and two different mutation strategies (shrinking and expanding). The algorithm optimizes the tree error

Fig. 1. An example of top-down induction of classification, regression, and model tree.
(RMSE, MAE) and size (number of leaves) using a weight formula or lexicographic analysis as a fitness function. A more complex solution called GMT is proposed in [15,17], where specialized EA is used to induce model trees as well as the linear regression models in the leaves.

Most of the systems that perform global searches in the field of candidate solutions compete successfully with popular greedy methods. Evolutionary induced trees are usually smaller and have higher prediction accuracy [4]. The price for better and more accurate predictions is the slow induction time of decision trees, but this can be partially mitigated by parallel implementations or combining EA with memetic algorithms [11,14].

Although, the forecasting performance of regression and model trees may be slightly lower than more complex solutions like random forests or neural networks, the explanation ability of the model trees and the transparency of predictions are significantly higher.

2.2. Asymmetric loss function

The loss function is a part of every forecasting model. Let dependent variable $y$ be predicted based on a vector of independent variables ($X$), and let $f(X)$ be a function of the regressors minimizing a particular loss function $L(y, f(X))$. Typical examples of loss functions are the squared error, such as used in least squares methods:

$$L(y, f(X)) = (y - f(X))^2 \quad (1)$$

and the absolute error:

$$L(y, f(X)) = |y - f(X)|. \quad (2)$$

Both loss functions are symmetric in that for every $y$ and $k$, the $L(y + k, f(X)) = L(y - k, f(X))$. Symmetric loss functions dominate in statistics and data mining. The Nobel Laureate Clive Granger stated that the obvious problem with the choice of the right loss function is that it is a symmetric function, whereas actual loss functions are often asymmetric [24].

The pioneer in the field of asymmetric costs in prediction was Varian [53], who proposed the $LinLin$ loss function. This loss function, which was approximately exponential on one side and linear on the other, became a popular alternative to least squares procedures. It was later extended to the asymmetric linear quadratic loss functions [10] presented in Fig. 2.

However, in machine learning, there are not many propositions of algorithms that handle asymmetric costs. In the most recent survey of cost-sensitive decision tree induction algorithms [34], the regression or model trees were not mentioned, and only the problem of classification was discussed. In the induction of cost-sensitive classification trees, three techniques are popular:

- Transforming the traditional decision tree into a cost-sensitive one. This requires incorporating costs into splitting criteria and pruning [8];
- Post hoc tuning (e.g., MetaCost [18] or cost instance weighting [47]). Both methods are universal and transform traditional algorithms into cost-sensitive;
- Using the evolutionary approach for cost-sensitive decision tree induction [30].

The vast majority of data mining algorithms are applied only to classification problems [48], while cost-sensitive regression is not really studied outside the field of statistics [5]. There are, however, few papers in the literature that deal with regression (e.g., in [12], the authors propose a modified back-propagation neural network (NN) that applies the $LinLin$ cost function). As for the regression and model trees, there are two different techniques: the application of EAs for model trees [16] and post hoc tuning [5,56]. There also exist other regression solutions, but they are focused on different types of costs, such as feature evaluation costs [23] and rare extreme values [49].

Post hoc tuning methods [5] for regression are similar to the ones in cost-sensitive classification. The solution proposed by Bansal et al. [5] minimizes average misprediction cost post hoc by adjusting the final prediction by a certain value. The algorithm is tested for traditional regression methods under an asymmetric cost structure. This system was later extended [56] and introduced polynomial functions as a model adjustment. This way, the algorithm could work with any convex cost function, such as $LinLin$ and $QuadQuad$.

The first cost-sensitive extensions for the global induction of model trees were introduced in [16]. Authors have successfully extended the evolutionary algorithm for model tree induction with an asymmetric $LinLin$ loss function that minimizes the average misprediction cost. Two variants of mutation operators that search for cost-sensitive linear regression models in the leaves were proposed. The first operator adjusted the regression model by a certain amount, like in the post hoc tuning method [5]. The second operator calculated a new regression model based on a subset of instances in the node.

2.3. Loan charge-off forecasting

Loan loss reserves are determined by banks based on their predictions of future loan charge-off amounts. This data is characterized by asymmetric costs on misprediction errors because the under-prediction of loan charge-off is more costly than over-prediction. This problem was also evaluated in the papers [5,56,16].

Let’s assume that the bank needs to predict the future loan charge-off ($y$) using a vector of independent variables ($X$). If the regression model in the bank over-predicts its future loan charge-off ($f(X) > y$), the worst that could happen is a reduction in the bank’s income because extra funds will remain in the loan loss reserves and in some cases, it will also result in a lower credit score from financial analysts. Under-prediction ($f(X) < y$) means that the bank did not prepare sufficient provisions for its loan losses and does not have enough reserves. It causes regulatory problems and a significant downturn of its credit rating, which is much more dangerous to the bank.

As it was pointed out in [5], it is important to penalize under-predictions more heavily than over-predictions by discouraging banks from having less than adequate amounts as reserves. This requires using asymmetric loss functions in prediction models like $LinLin$ and $QuadQuad$.

Fig. 2. An example of $LinLin$ and $QuadQuad$ cost functions.
3. Cost-sensitive Global Model Tree

In this section, we present how to induce the cost-sensitive model tree with the evolutionary approach. At first, we briefly describe a solution called Global Model Tree (GMT) [17]. Next, we illustrate how to efficiently convert this cost-neutral model tree inducer into the cost-sensitive one and propose our solution, the Cost-sensitive Global Model Tree (CGMT). We also refer to the solutions proposed in [16, 5, 56] and show how they can be improved.

3.1. Global Model Tree schema

Evolutionary algorithms [37] belong to a family of meta-heuristic methods. They represent techniques for solving a wide variety of difficult optimization problems. GMT follows a traditional framework of EA inspired by biological mechanisms of evolution. The algorithm (see Fig. 3) operates on individuals that compose a current population. Each individual represents a candidate solution to the target problem. Individuals are assessed using a fitness function that measures their performance. Next, individuals with higher fitness usually have a higher probability of being selected for reproduction. Genetic operators such as mutation and crossover influence individuals, thereby producing new offspring. This guided random search (offspring usually inherits some traits from its ancestors) is stopped when some convergence criteria is satisfied.

Model trees are represented in their actual form as traditional univariate trees, so each split in the internal node is based on a single attribute. If the attribute is nominal, at least one value is associated with each branch (inner disjunction). In case of a continuous-valued attribute, the typical inequality tests are applied. To predict instances associated with each leaf, a multivariate linear regression model [40] is constructed. Initial individuals are induced using greedy strategies on a random subsample of the training data, and the tests in internal nodes are searched on a random subset of attributes. Each tree leaf contains a multiple linear regression model using the standard regression technique [40] that is constructed with instances associated with that leaf. A dependent variable \( y \) is explained by the linear combination of multiple independent variables \( X = \{ x_1, x_2, \ldots, x_m \} \):

\[
y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_q x_q.
\]

\[ \text{Initialize population of } P \text{ individuals with semi-random top-down strategy} \]

\[ \text{Evaluate solutions in the population (fitness)} \]

\[ \text{Perform ranking selection} \]

\[ \text{Apply genetic operators to generate new solutions} \]

where \( q \) is the number of independent variables and \( \beta_0 \ldots \beta_q \) are fixed coefficients that minimize the sum of the squared residuals of the model.

Tree-based representation requires developing specialized genetic operators corresponding to classical mutation and crossover. Application of the operators can modify the tree structure, tests in internal nodes, and models in the leaves. The crossover operator attempts to combine elements of two existing individuals (parents) to create a new solution. An example of crossover where two individuals exchange all subtrees is illustrated in Fig. 4. The mutation operator makes random changes in some places of selected individuals. Several variants of crossover and mutations were proposed [13, 15, 17]:

- Modify the test in internal nodes (shift threshold, replace tested attribute);
- Prune the internal node and transform it into the leaf with a new multivariate linear regression model;
- Expand the leaf into the internal node;
- Replace one of the following: subtree, branch, node, or test between two affected individuals;
- Modify linear regression models in the leaves (add, remove, or change attributes).

The selection mechanism is based on the ranking linear selection [37] with the elitist strategy, which copies the best individual founded so far to the next population.

Selection acts as a force that increases the quality of the population. For this reason, the mechanism of selection usually requires a quantitative measure to assess individuals in the population. Such an objective function is called a fitness function and in practice, it is a very important and sensitive element of the evolutionary system. It drives the evolutionary search process by measuring how good a single individual is in terms of meeting the problem objective. In the context of decision trees, a direct minimization of the prediction performance measured on the learning set leads to the over-fitting problem and poor performance on unseen testing instances. Therefore, there is a need to balance the predictive error and the complexity of the tree. In GMT, the authors adapt the Bayesian Information Criterion (BIC) [43] as a fitness function. The BIC fitness is equal to:

\[
\text{Fit}_{BIC} (T) = -2 \cdot \ln (L(T)) + \ln (n) \cdot k(T),
\]

where \( L(T) \) is the tree \( T \) maximum of likelihood function, \( k(T) \) is the complexity term, and \( n \) equals the number of instances. The function \( L(T) \) is common for regression models [21] and equals:

\[
\ln (L(T)) = -0.5n \cdot [\ln (2\pi) + \ln (SSe(T)/n) + 1].
\]

The term \( SSe(T) \) is the sum of squared residuals of the tree \( T \) (on the training set).

3.2. Cost-sensitive transformation

There are a few steps to transform the regular model tree into a cost-sensitive one. First, the authors adapt the average misprediction cost (AMC) [5] as a measurement for assessing the performance of the proposed method. Consider a dependent variable, \( y_i \), that is predicted based on a vector of independent variables \( x \). The regression method learns the prediction model, \( f: x \rightarrow y \) from the training data \( S = \{ \langle x_i, y_i \rangle | i = 1, 2, \ldots, N \} \). AMC can be defined as:

\[
\text{AMC} = \frac{1}{N} \sum_{i=1}^{N} C(f(x_i) - y_i),
\]

where \( C(e) \) is a function that characterizes the cost and \( e \) is a prediction error. AMC measurements can be viewed as a more general version of
the mean absolute error (MAE). When the costs for under-prediction and over-prediction are equal, the loss function is symmetric (Eq. (2)) and \( AMC = MAE \).

In CGMT, we adapted the cost-sensitive BIC-like fitness function proposed in [16]. The authors replaced \( SSe(T) \) from Eq. (5) with \( AMC \) from Eq. (6). This way, the fitness function will measure how good a single individual is in terms of meeting the cost-sensitive problem objective. The complexity term, \( k(T) \), works as a penalty for over-parameterization of the tree \( T \). The penalty term:

\[
k(T) = 2 \times (Q(T) + M(T)) \tag{7}
\]

where \( Q(T) \) is the number of internal nodes and \( M(T) \) is the number of attributes that build models in the leaves. In this paper, we updated the parameter \( k(T) \) in comparison to [16]. Performed experiments showed that this higher value of complexity term works better with CGMT because it reduces the tree size and improves the generalizability.

With the all cost-sensitive transformations proposed in this paper, the final form of the fitness function for CGMT is as follows:

\[
Fit_{CGMT}(T) = n[\ln(2\pi) + \ln(AMC(T)/n) + 1] + 2\ln(n) + (Q(T) + M(T)) \tag{8}
\]

where in the first element of equation depends on the tree average misprediction cost measured on the learning set and the second part reflects the tree complexity.

To allow CGMT to work with any convex cost function (e.g., LinLin, QuadQuad, LinEx, and SquarEx) we had to develop a universal way to search for cost-sensitive models in the leaves of the tree. We proposed an adjustment of regression models and different variants of mutation operator together with modified fitness function, thereby efficiently converting cost-neutral model trees into cost-sensitive ones.

3.2.1. Model adjustment for any convex cost function

One of the ways to modify the regression model in the leaf into a cost-sensitive one is to shift it toward a smaller \( AMC \) error. We will denote the desired shift as \( \theta \). This strategy works well in the post hoc tuning methods proposed in [5] and later in [56]. To find regression model adjustment, the authors used a heuristic approach such as the hill climbing algorithm. A similar idea was proposed in [16], where each regression model was shifted by a certain amount using EA. For convex cost functions like QuadQuad, LinEx, and SquarEx, \( \theta \) cannot be calculated directly and needs to be approximated. In CGMT, we simply checked between which two instances the \( AMC \) error was minimized. Next, the algorithm selects \( \theta \) randomly as long as two instances are on the different sides of the regression line (one instance should be under-predicted and the other should be over-predicted). After calculating the adjustment, the new regression model in the leaf is equal to:

\[
f_{\text{new}}(x) = f(x) + \theta \tag{9}
\]

where \( \theta \) is the shift. Fig. 5 shows an example of actual \( f_0(x) \) and the shifted regression model \( f_{\text{new}}(x) \).

In CGMT, the adjustment of the regression model in the leaf is applied after any modifications of the regression model. Changes in regression models can be caused either by mutation or crossover operators.

3.2.2. Model adjustment for LinLin

Finding adjustment \( \theta \) with EA is time consuming as besides searching for the optimal tree structure, splits in internal nodes, and regression models in the leaves, EA also looks for the value that each model should be shifted. However, in case of LinLin cost function, the adjustment \( \theta \) can be calculated directly.
Let $x^-$ denote under-predicted instances and $x^+$ denote over-predicted instances by an actual regression model in the leaf. The cost of under-prediction and over-prediction equals $C^-$ and $C^+$, respectively. If the $AMC^+$ denotes the average misprediction cost of over-predicted instances ($x^+$) and $AMC^-$ the under-predicted ones ($x^-$), then:

$$AMC = AMC^+ + AMC^-.$$ (10)

In $LinLin$ loss function, which minimizes the absolute error (Eq. (2)), the problem of searching for $\theta$ is reduced to finding appropriate regression quantile that will split instances according to their costs and cardinality [28]. In our case, we know the weights given to positive (over-predicted) and negative (under-predicted) instances. To minimize Eq. (10), we need to have equal weight differences between over-predicted and under-predicted instances; therefore, we need to solve:

$$N^+ * C^+ - N^- * C^- = 0,$$ (11)

where $N^+$ and $N^-$ denote the number of over-predicted and under-predicted instances in the adjusted model, respectively. Considering that $N^+ + N^- = N$, we can estimate the number of instances that should be over-predicted and under-predicted:

$$\hat{N}^+ = \frac{n * C^-}{C^- + C^+}, \quad \hat{N}^- = \frac{n * C^+}{C^- + C^+}.$$ (12)

The final step is to calculate model adjustment, which equals:

$$\theta = y_k - f(x_k),$$ (13)

where $k$ is the $N^+$-th instance in order starting from the most under-predicted instances to the most over-predicted. The instance $k$ can also be calculated as the $N^+$-th one in order, sorted from the most over-predicted instances to the most under-predicted, as this would return the same instance. However, the estimated value $\hat{N}^+$ from Eq. (12) does not have to be an integer; therefore, we have to consider two variants:

- If $\hat{N}^+$ is an integer, then $N^+ = \hat{N}^+$,
- If $\hat{N}^+$ is not an integer, then: $N^+ = \left\{ \begin{array}{ll} \text{floor}(\hat{N}^+) & \text{if } C^- > C^+ \\ \text{ceil}(\hat{N}^+) & \text{if } C^- < C^+. \end{array} \right.$

With knowledge of the exact value of over-predicted and under-predicted instances in the adjusted model, we can calculate the shift from Eq. (13). It can be observed that when $N^+ = \hat{N}^+$, the $\theta$ has the same value in the range $\langle y_k - f(x_k), y_{k+1} - f(x_{k+1}) \rangle$.

### 3.2.3. Random model changes

Shifting regression models in the leaves to make them cost-sensitive is one of the options. Another one is to modify the coefficients in linear regression models and leave the search for the optimal one to the EA. We randomly selected a single coefficient and modified it (increased or decreased it) by a random percent from 0 % to 100 %. Next, we adjusted the regression model with the above-mentioned algorithm. Such a technique, together with a mutation variant that added, removed, or changed the attributes in the models in the leaves, allowed us to find completely new and cost-sensitive regression models. It can also work with any convex cost function like $LinLin$, QuadQuad, LinEx, SquarEx, etc. Fig. 5 illustrates actual $f_0(x)$ and the new, randomly modified regression model denoted as $f_2(x)$.

### 3.3. Discussion of the improvements

In the paper, we have extended the GMT solution [17] to work as a cost-sensitive learner. The original GMT is a cost-neutral algorithm with symmetric loss function and therefore, couldn’t be applied to the data with different prediction costs. To transform GMT to the cost-sensitive learner, the new fitness function that involves costs of the under-predictions and over-prediction was defined. Next, new variants of mutations were proposed that improve the search of the cost-sensitive regression models in the leaves. Modifications of the initialization the individuals in the population was also performed as only the long dipolar strategy [16] was applied to search for the splits in the internal nodes. Experimental results presented in the next section show that there is a huge gap between GMT and CGMT when cost-sensitive data is analyzed.

In the literature, there is also an attempt to transform GMT to the cost-sensitive learner denoted as $GMT_{CS}$ [16]. There are, however, significant differences between proposed $CGMT$ solution and the $GMT_{CS}$. In particular:

- An evolutionary search of the regression model adjustments for the $LinLin$ cost function in $GMT_{CS}$ was replaced by the analytical formula. This increased the convergence of evolutionary algorithms and significantly improved the prediction accuracy of the regression models;
- New mutation operators that perform random changes in the regression models in the leaves were introduced. This allowed $CGMT$ to
explore a larger solution space and fit more accurate cost-sensitive linear regression models into the instances in the leaves;

- The previous GMT algorithm works only with the LinLin cost function. The CGMT solution works with any convex cost functions like LinLin, QuadQuad, LinEx, SquareX, etc.;
- The fitness function of the CGMT was improved to increase the algorithm’s generalization ability;
- More detailed experimental analysis of loan charge-off predictions was performed, and LinLin and QuadQuad loss functions were investigated. Presented results in the next section show that CGMT managed to significantly outperform \( P \text{ value} < 0.0001 \) in terms of error reduction the GMT solution.

4. Experiments

In this section, we present an experimental validation of the proposed approach called CGMT and its competitors on a loan charge-off forecasting problem.

4.1. Datasets

The proposed solution is applied to a loan charge-off forecasting problem that is characterized by asymmetric costs, as over-prediction is less costly than under-prediction. We focus on financial institutions in the USA from 2004 to 2010 and use the data from Wharton Research Data Services [55]. The average number of banks analyzed in this study was 7,695 — from 6,992 to 8,315. In each quarter of the year, the banks needed to forecast the amount of loan charge-off.

In the experiments, we preprocessed the data in the same way as other authors that have dealt with this forecasting problem [5,56,16]. The instances with missing values were removed and the natural logarithm transformation was performed on the dependent variable to reduce the extent of skewness.

Each quarter of the year was predicted separately with 14 attributes related to the current financial information of the bank. To predict the loan charge-off in the following quarter, the attributes presented in Table 1 were used. From the table, we can observe that one of the independent variables is the loan charge-off of the previous quarter. One can also notice that no e.g., macroeconomic variables and unemployment information were used. These attributes could have had an impact on the amount of charge-off loans, however, to compare CGMT with previous research, we used the same set of independent variables.

From 28 quarters, 27 datasets were generated, as for the newest quarter the predicted charge-off value for 1 2011 was not accessible in WRDS. Each algorithm was trained on a single (training) quarter and then tested on the following (testing) quarter. In this way, 26 independent training and testing sets were used in the experiments.

4.2. Setup

We followed the experimental validation performed in [56]. Thanks to Prof. Huimin Zhao, who provided us with the source code of his tuning methods denoted as BSZ [5] and Linear [56], we were able to confront the prediction performance of all tested algorithms. We attached the average misprediction cost (AMC) for three base regression models: standard least-squares linear regression (LR), the M5 model tree [52], and the back-propagation neural network (NN) [44], which were also tested in [5,56,16].

In all reported experiments, the algorithms ran with their default settings. Algorithms LR, M5, and NN were tested using the Weka system [25]. The settings of CGMT were as recommended in GMT [17]: the population size was 50, the probability of the mutation single node was 0.8, and the probability of the crossover between two individuals was 0.2.

We also presented the results of the cost-neutral GMT algorithm and proposed the CGMT solution. Tests were performed for two different cost functions; LinLin and QuadQuad, and different cost ratios (CR) for under-prediction to over-prediction were as follows: 10:1, 20:1, 50:1, and 100:1. For the results with LinLin cost function, we also included the GMT results [16] (the algorithm does not work with the QuadQuad loss function). Table 2 illustrates an overview of the performed experiments, algorithms, and tested settings.

### Table 1

<table>
<thead>
<tr>
<th>No.</th>
<th>Code</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RCFD1400</td>
<td>Total loans and leases; gross</td>
</tr>
<tr>
<td>2</td>
<td>RCFD1403</td>
<td>Total loans and lease finance receivables:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nonaccrual</td>
</tr>
<tr>
<td>3</td>
<td>RCFD1407</td>
<td>Total loans and lease financing receivables:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>past due 90 days or more and still accruing</td>
</tr>
<tr>
<td>4</td>
<td>RCFD2013</td>
<td>Intangible assets</td>
</tr>
<tr>
<td>5</td>
<td>RCFD2017</td>
<td>Total assets</td>
</tr>
<tr>
<td>6</td>
<td>RCFD3103</td>
<td>Goodwill</td>
</tr>
<tr>
<td>7</td>
<td>RCFD3200</td>
<td>Subordinated notes and debentures</td>
</tr>
<tr>
<td>8</td>
<td>RCFD3210</td>
<td>Equity capital, total</td>
</tr>
<tr>
<td>9</td>
<td>RIAD4010</td>
<td>Interest and fee income on loans, total</td>
</tr>
<tr>
<td>10</td>
<td>RIAD4079</td>
<td>Total noninterest income</td>
</tr>
<tr>
<td>11</td>
<td>RIAD4180</td>
<td>Expense of federal funds purchased and securities sold under agreements to repurchase</td>
</tr>
<tr>
<td>12</td>
<td>RIAD4340</td>
<td>Net income (loss)</td>
</tr>
<tr>
<td>13</td>
<td>RCFD4223</td>
<td>Risk-weighted assets (net of allowances and other deductions)</td>
</tr>
<tr>
<td>14</td>
<td>RIAD4635</td>
<td>Charge-offs on allowance for loan and lease loans during the calendar year-to-date.</td>
</tr>
</tbody>
</table>
4.3. Comparison results

Table 3 summarizes the AMC results for CGMT, three state-of-the-art regression solutions, and the cost-neutral model tree denoted as GMT. In Table 3 we also show the impact of two post hoc cost-sensitive tuning methods: BSZ [5] and Linear [56]. In the experiments, four cost ratio (CR) and two different loss functions (LinLin and QuadQuad) are applied. We report an average value that algorithms achieved on 26 independent testing sets. In case of CGMT, the AMC result equals an average value of 20 runs; therefore, the standard deviation is included.

From the results in Table 3, we see a huge difference between cost-neutral and cost-sensitive algorithms. Post hoc tuning methods significantly reduce AMC, and the Linear variant is almost always better than BSZ. We can observe that the best performance of CGMT competitors is achieved by GMTCS and NNlinear. The poor performance of the MS model tree is a result of over-fitting and the biases of the linear models in the leaves for outliers in the analyzed datasets. It is especially visible when the QuadQuad loss function is applied. An additional study of MS performance showed that errors on a few under-predicted instances was responsible for such poor results (on one of the instances, the AMC was near 1,000,000, which equals under-prediction of the loan charge-off for QuadQuad loss function with Cost ratio = 100). When those outliers were eliminated, M5 managed to obtain slightly better results than LR. In Table 3, we can see that the post hoc tuning of CGMT (algorithms CGMTBSZ and CGMTLinear) has no impact on the results. It proves that there is no more space for improvement for CGMT with examined tuned methods.

The average AMC reduction between the best post hoc tuned algorithm from Table 3 and CGMT varies between 11.8 % and 28.3 %. The real improvement is much higher because the results are on a natural logarithm scale that varies between 44.6 % and 99.9 %. We believe that such error reduction could not be ignored by the institutions struggling with the loan charge-off forecasting. The QuadQuad loss function and cost ratio equals 100, and the AMC for NNlinear equals 14,831,216,212 ($e^{44.6}$), where for CGMT, the AMC is almost four orders of magnitude smaller (9, 530, 426).

It can be expected that searching for cost-sensitive models directly during induction results in finding better solutions. The first attempt of CS-extensions for GMT applied to LinLin (algorithm GMTCS) [16] managed to decrease the AMC and outperformed all tuned base regression models under every cost ratio. However, our algorithm goes even further — it significantly decreases the AMC in comparison to all tested solutions. The CGMT solution can work with any convex function (in contrast to GMTCS). The comparison results are enclosed in Table 4. The average cost reduction between the GMTCS and CGMT is in the range of 9.4 % to 15.4 % when the cost values are on a natural log scale. Therefore, the real cost reduction on the original scale is in the range of 26.7 % to 55.5 %. Unfortunately, we could not compare GMTCS with CGMT using the QuadQuad loss function. However, the AMC reduction for CGMT is expected to be higher than for GMTCS.

The Wilcoxon signed rank test between CGMT and GMTCS proved that the differences between the algorithms are statistically significant (P value < 0.0001). In addition, we have also tested within Weka software two nonlinear regression models: RBF Network [9] that implements a normalized Gaussian radial basis function network and SMOreg [46] that is the support vector machine for regression (with polynomial kernel). Both tuned algorithms perform similar to GMTCS and NNlinear and together with the rest of tested methods, they were statistically worse than the CGMT solution.

5. Discussion

In Tables 3 and 4, we showed the average error results for 26 datasets. To prove that the CGMT algorithm is consistent in reducing the AMC error, we present, in detail, the performance of the top three algorithms: NNlinear, GMTCS, and CGMT. In the first experiment in Tables 3 and 4, the LinLin loss function was used, and the cost ratio was equal to 10. With these settings, the AMC reduction for CGMT was the lowest due to a small penalty for wrong predictions. It is, however, still significant. The results illustrated in Fig. 6 show that the proposed solution is almost always better and (never worse) than NNlinear and GMTCS on all 26 tested datasets. The cost values are on a natural log scale, so the real differences between CGMT and other solutions are much higher.

We can also observe that the results of all algorithms strongly depend on the quarter of the year. One of the reasons for poor predictions at the beginning of each year is that the banks changed the cost-ratio for this particular l quarter. The new year is always a difficult time to forecast financial data, as all institutions need to publish their annual macroeconomic data. In other words, the results were consistent in terms of the cost ratio impacts on the errors.

Table 2
An overview of GMT experimental validation.

<table>
<thead>
<tr>
<th>Model</th>
<th>Tuned BSZ</th>
<th>Tuned Linear</th>
<th>Ratio</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>10:1</td>
<td>7.41</td>
<td>10</td>
<td>LinLin</td>
</tr>
<tr>
<td>M5</td>
<td>10:1</td>
<td>7.29</td>
<td>10</td>
<td>LinLin</td>
</tr>
<tr>
<td>NN</td>
<td>10:1</td>
<td>8.16</td>
<td>10</td>
<td>LinLin</td>
</tr>
<tr>
<td>GMT</td>
<td>10:1</td>
<td>7.07</td>
<td>10</td>
<td>LinLin</td>
</tr>
<tr>
<td>GMTCS</td>
<td>5:1</td>
<td>2.98</td>
<td>5:1</td>
<td>LinLin</td>
</tr>
<tr>
<td>GMT</td>
<td>5:1</td>
<td>2.98</td>
<td>5:1</td>
<td>LinLin</td>
</tr>
<tr>
<td>± (stdev)</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
</tr>
</tbody>
</table>

Table 3
Average misprediction cost (AMC) for CGMT, GMT, and three popular regression algorithms with and without post hoc tuning.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CR</th>
<th>LinLin loss function</th>
<th>QuadQuad loss function</th>
</tr>
</thead>
<tbody>
<tr>
<td>No tuning</td>
<td>10:1</td>
<td>7.41</td>
<td>3.81</td>
</tr>
<tr>
<td>M5</td>
<td>10:1</td>
<td>7.29</td>
<td>3.88</td>
</tr>
<tr>
<td>NN</td>
<td>10:1</td>
<td>8.16</td>
<td>3.57</td>
</tr>
<tr>
<td>GMT</td>
<td>10:1</td>
<td>7.07</td>
<td>3.65</td>
</tr>
<tr>
<td>GMTCS</td>
<td>5:1</td>
<td>2.98</td>
<td>2.98</td>
</tr>
<tr>
<td>± (stdev)</td>
<td>-</td>
<td>0.16</td>
<td>0.16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CR</th>
<th>LinLin loss function</th>
<th>QuadQuad loss function</th>
</tr>
</thead>
<tbody>
<tr>
<td>No tuning</td>
<td>20:1</td>
<td>14.06</td>
<td>4.42</td>
</tr>
<tr>
<td>M5</td>
<td>20:1</td>
<td>13.78</td>
<td>4.67</td>
</tr>
<tr>
<td>NN</td>
<td>20:1</td>
<td>15.60</td>
<td>4.62</td>
</tr>
<tr>
<td>GMTCS</td>
<td>20:1</td>
<td>3.43</td>
<td>3.43</td>
</tr>
<tr>
<td>± (stdev)</td>
<td>-</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>No tuning</td>
<td>50:1</td>
<td>34.02</td>
<td>6.24</td>
</tr>
<tr>
<td>M5</td>
<td>50:1</td>
<td>33.23</td>
<td>6.03</td>
</tr>
<tr>
<td>NN</td>
<td>50:1</td>
<td>37.92</td>
<td>5.69</td>
</tr>
<tr>
<td>GMT</td>
<td>50:1</td>
<td>32.96</td>
<td>6.40</td>
</tr>
<tr>
<td>GMTCS</td>
<td>50:1</td>
<td>4.02</td>
<td>4.02</td>
</tr>
<tr>
<td>± (stdev)</td>
<td>-</td>
<td>0.25</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Table 4
Average misprediction cost (AMC) for GMTCS and CGMT. Algorithms used the LinLin loss function.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>CR = 10</th>
<th>CR = 20</th>
<th>CR = 50</th>
<th>CR = 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMTCS</td>
<td>3.29</td>
<td>3.85</td>
<td>4.66</td>
<td>5.27</td>
</tr>
<tr>
<td>CGMT</td>
<td>2.98</td>
<td>3.43</td>
<td>4.02</td>
<td>4.46</td>
</tr>
</tbody>
</table>
these uncertain times, banks increase their reserves to minimize the risk of not covering their loan losses. However, seasonality affects quarterly data for accounting as well as for economic reasons. For example, in [33], the authors show that loan provisions are often delayed to the IV fiscal quarter when the audit occurs. This factor would explain the constant bad performance of algorithms in the IV quarter, especially when prediction models are built using the previous values of Total Loan Charge-offs (RIAD4635).

Besides prediction accuracy, there is one more important factor in loan charge-off forecasting problems. Business analysts need a solution that can provide new insights into underlying process that they need to understand. Fig. 7 illustrates an output tree for CGMT for the first quarter of 2004 with a LinLin loss function and cost ratio equal to 10.

The output tree of the CGMT algorithm is compact and can be easily analyzed and interpreted, even in this form. Internal node tests consider previous values of Total Loan Charge-offs (RIAD4635) and Total Loans Not Accruing (RCFD1403). Models in the leaves also consider Noninterest Income (RIAD4079) and Loans 90 + Days Late (RCFD1407). To calculate the real value of the searched loan charge-off, the exponential value of prediction should be taken as the loan charge-off is on a natural log scale. In addition, all attributes (except for RIAD4635, which is on a natural log scale) represent the amount in thousands of dollars. Therefore, for example, the first split checks if the loan charge-off is greater than $391, 505, and if that is true, the predicted loan charge-off for the tested bank can be calculated from the equation:

\[
y = e^{0.94 - \text{RIAD4635} - 1.43}.
\]

The AMC of this particular tree equals 1.545 and the tree has five linear models with an average of 1.8 attributes in each linear model. For this particular dataset (I 2004), the M5linear output tree has an AMC equal to 1.80, and the output tree is larger (19 leaves) and uses an average of 12.1 attributes (from available 14) in each linear model. To top it all off, the post hoc Linear tuning method must be applied; therefore, each prediction of new instances must be multiplied by 0.83, and a value of 1.88 must be added. In this way, the M5 algorithm, which is generally considered as a "white box" solution becomes a model that is really difficult to understand, analyze, and interpret.

A small number of leaves and attributes in the linear models is crucial to understanding the relationships in the data. It should be noticed that the whole tree (tests in internal nodes and models in the leaves) was designed to solve a cost-sensitive problem in the learning phase by EA. The fitness function of CGMT, with its penalty term for overgrown (number of leaves) and complicated (number of attributes in models) trees, keeps output solutions as small and simple as possible. Although induced trees by the CGMT system are different for each quarter, there are similarities and patterns between them that could be helpful for bank analysts. In all performed experiments with different cost ratios and loss functions, the trees induced by CGMT have an average of 8.5 leaves with 1.47 attributes in each linear model. The output trees for CGMT are in general three times smaller and have eight times fewer attributes in linear regression models than the M5 solution.

Finally, the evolutionary approaches are not the fastest ones, and the CGMT is not an exception. The average training time for the proposed solution was from a few minutes to an hour for each dataset which, considering the amount of data, is not a bad result. The execution of the CGMT solution on test instances is very fast.

6. Conclusion and future works

In this paper, we proposed a cost-sensitive solution for model trees. Specialized evolutionary algorithms that can work with any convex cost
function incorporated into the training process successfully dealt with forecasting problems where under-prediction and over-prediction errors had different consequences. With the memetic operators and cost-sensitive fitness function, we managed to efficiently evolve simple model trees that minimized the average misprediction cost.

Experimental validation was performed on 26 independent datasets and over 200,000 instances were tested. We proved that the proposed solution, CGMT, is significantly better than all post hoc tuned methods, as well as other evolutionary approaches, such as GMTC. The true reduction of cost error varies between 26.7% and 99.9%. However, the improvement was not only in the prediction accuracy. Equally if not more important is the fact that the decisions and models induced by CGMT can have direct applicability. They are simple, can be easily understood, and consider costs of errors during the learning phase.

We see many promising directions for future research. In particular, we are focused on extending CGMT to work with multiple costs, such as the cost of attributes. We also want to apply CGMT to different real-life forecasting problems.

Acknowledgments

The authors want to thank Huimin Zhao, Atish P. Sinha, and Gaurav Bansel for providing the implementation of BSZ and Linear tuning methods. This project was funded by the Polish National Science Centre allocated on the basis of decision no. 2013/09/N/ST6/004083.

References


Marcin Czajkowski obtained his Ph.D. with honors in Computer Science from Bialystok University of Technology, in 2015. He has published papers in multiple peer-reviewed journals and conferences. His current research interests are data mining, machine learning and bio-inspired computation.

Monika Czerwona is an assistant professor in the Institute of Finance at Warsaw School of Economics. She received her M.A. in 2003 and her Ph.D. in economics from Warsaw School of Economics in 2007. Her fields of specialization are: Behavioral Finance, Capital Markets, Socially Responsible Investing, and Cultural Finance.

Marek Kretowski is an associate professor in the Faculty of Computer Science, Bialystok University of Technology. He received his Ph.D. in 2002 from the University of Rennes I and Bialystok University of Technology. In 2009 he obtained his habilitation degree from the Institute of Computer Science, Polish Academy of Science. His current research interests are data mining, evolutionary computation and biomedical application of computer science.