EBNO: Evolution of Cost-Sensitive Bayesian Networks

Eman Nashnush and Sunil Vadera[1]

Author version of article to appear in the Expert Systems Journal (accepted Oct 2019).

ABSTRACT

The last decade has seen an increase in the attention paid to the development of cost sensitive learning algorithms that aim to minimize misclassification costs while still maintaining accuracy. Most of this attention has been on cost sensitive decision tree learning, while relatively little attention has been paid to assess if it is possible to develop better cost sensitive classifiers based on Bayesian networks. Hence, this paper presents EBNO, an algorithm that utilizes Genetic Algorithms to learn cost sensitive Bayesian networks; where genes are utilized to represent the links between the nodes in Bayesian networks and the expected cost is used as a fitness function. An empirical comparison of the new algorithm has been carried out with respect to: (i) an algorithm that induces cost-insensitive Bayesian networks to provide a base line, (ii) ICET, a well-known algorithm that uses Genetic Algorithms to induce cost-sensitive decision trees, (iii) use of MetaCost to induce cost-sensitive Bayesian networks via bagging (iv) use of AdaBoost to induce cost-sensitive Bayesian networks and (v) use of XGBoost, a gradient boosting algorithm, to induce cost-sensitive decision trees. An empirical evaluation on 28 data sets reveals that EBNO performs well in comparison to the algorithms that produce single interpretable models and performs just as well as algorithms that use bagging and boosting methods.

Keywords— Cost-sensitive classification, machine learning, data mining

Conflicts of interest: none

1 INTRODUCTION

The induction of classifiers from data sets of pre-classified instances is known to be a major challenge and many algorithms have been introduced to learn decision trees, Bayesian networks, and neural networks. Most of the early machine learning algorithms focused on maximizing accuracy and assumed that costs for misclassification error remain equal, irrespective of the class (Dong & Wu, 2018; Mitchell, 1997). However, several authors have noted that this is not adequate for practical applications (Domingos, 1999; Drummond & Holte, 2000). Hence, in recent years, a significant level of attention has been paid to cost-sensitive learning which aims to minimize the expected cost (Dai, Han, Hu, & Liu, 2016; Ling & Sheng, 2004; Lomax & Vadera, 2017). Historically, most of the cost-sensitive algorithms developed have focused on learning decision trees, with a recent survey comparing over 50 algorithms (Lomax & Vadera, 2013). In contrast, although Bayesian networks (BNs) have proved their effectiveness in a wide range of applications (Heckerman, Mamdani, & Wellman, 1995; Jiang, Li, Cai, & Zhang, 2013), the number of studies exploring the development of algorithms that learn cost-sensitive Bayesian networks is limited to a few studies (Jiang, Li, & Wang, 2014; E. Nashnush & Vadera, 2014, 2017).

In general, a Bayesian network classifier is a probabilistic model that represents variables (continues or discrete) as nodes in a Directed Acyclic Graph (DAG) where edges between nodes represent the direct correlations between variables (Pearl, 1988).

There are two steps to constructing Bayesian networks (Heckerman et al., 1995; Lauritzen & Spiegelhalter, 1988):

1. Learning the graphical structure: this aims to find the relationships between the variables.

[1] Corresponding author is Sunil Vadera, S.Vadera@salford.ac.uk

2. Learning the parameters: this aims to determine the extent of the relationships between the variables and takes the form of a table that represents the conditional probabilities between a node and its parents.

Learning the graphical structure of a Bayesian network is known to be an NP-hard problem and much more challenging than learning the parameters (Cheng & Greiner, 2001; Chickering, Heckerman, & Meek, 2004; Dasgupta, 1999). Although several algorithms have been developed to learn different types of graphical structure (e.g., Chow and Liu(1968); Langley et al. (1992); Friedman et al.(1997)) they focus on maximizing accuracy without taking account of the cost of misclassifications. Hence, this paper explores whether Genetic algorithms (GAs) can be used to evolve the structure of Bayesian networks that minimize the expected cost of misclassification whilst utilizing existing methods for estimating the parameters. The key questions explored in this paper include:

- Is it possible to evolve cost-sensitive Bayesian networks that are better at minimizing cost than algorithms that induce cost-sensitive decision trees?
- Are the resulting cost-sensitive Bayesian networks better at minimizing cost than those obtained from algorithms that aim to maximize accuracy?
- If there are improvements, are they at the expense of reduced accuracy?

The paper is organized as follows: Section 2 presents the context by describing approaches to cost-sensitive learning; Section 3 develops EBNO, an evolutionary algorithm for learning cost-sensitive Bayesian networks; Section 4 presents an empirical evaluation aimed at addressing the questions above and Section 5 concludes the paper.

2 Background on Cost-Sensitive Learning

There have been many different approaches to cost-sensitive learning dating back to the late 1980s. Figure 1 lists some of the major studies to provide a historical context. These algorithms can be categorized under three broad approaches:[2] (i) methods that explicitly modify an algorithm to take account of costs, (ii) methods that utilize bagging and boosting, and (iii) methods that utilize genetic algorithms.

These three approaches have been used mainly to develop algorithms that induce cost-sensitive decision trees but can also be adopted for developing algorithms that induce Bayesian networks, and hence the following subsections summarize some of the key studies. Readers interested in a more detailed and comprehensive account are referred to the survey by Lomax and Vadera (2013). Several authors have also described algorithms for the evolution of Bayesian networks (Campos, Fernandez-Luna, Gámez, & Puerta, 2002; Larrañaga, Poza, Yurramendi, Murga, & Kuijpers, 1996; Zeng, Zhang, Cai, Jiang, & Jiang, 2006) and although these do not take account of cost, they are closely related to the work described in this paper and are also summarized below.

[2] Other categorisations, such as black box and white box (Zadrozny et al. 2003) also exist and the one we use extends this to include algorithms that utilise genetic algorithms.

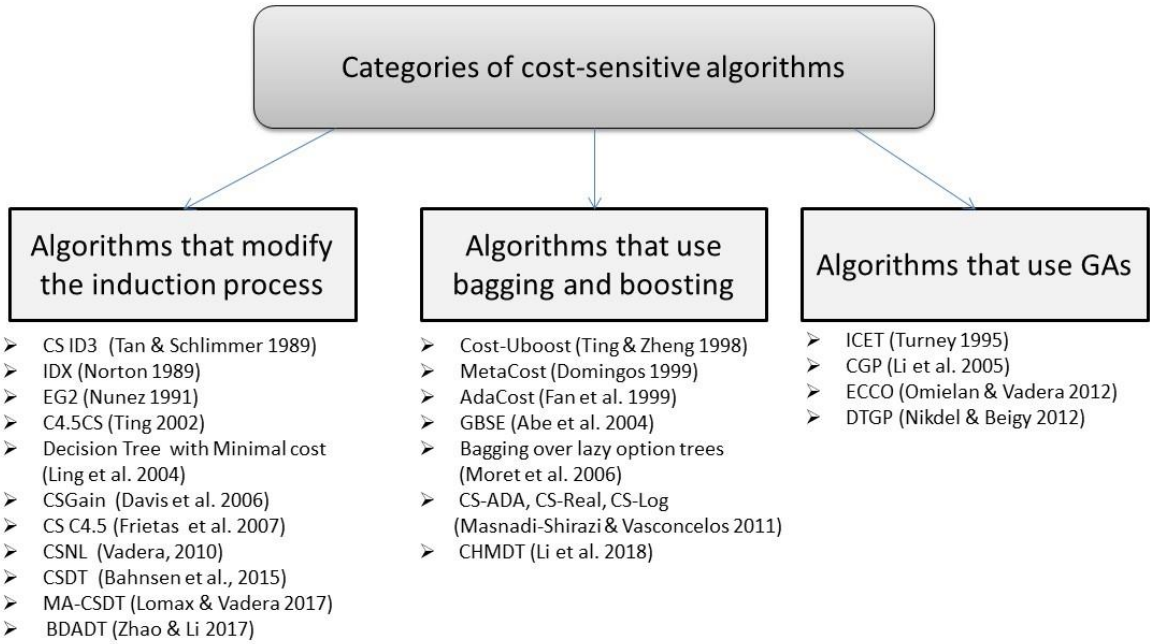

Figure 1: **Categories of cost-sensitive learning algorithms with references to examples of studies under each category**[3]

2.1 Algorithms that modify a classifier

A key step in decision tree learning is selecting the next attribute of the decision tree, which is typically done by using an information theoretic measure such as information gain. This is based on the difference between the entropy of a class label on the current data, D, before splitting and the entropy if an attribute A is used (Quinlan, 1993):

$$\text{Entropy(D)} = \sum_{c \in \text{Class}} -\text{P(c)}.\log_2 \text{P(c)}$$

$$\text{Entropy(A)} = \sum_{a \in A} \text{P(a)} \,.\, \sum_{c \in \text{Class}} -\text{P(a|c)} \;\; . \;\; \log_2 \text{P(a|c)}$$

$$\text{I}_\text{A} \;=\; \text{Entropy(D)} - \text{Entropy(A)} \qquad (1)$$

Where, c is a class value, a is an attribute value and I_A is the information gain.

The attribute that results in the highest information gain is used as the next attribute in a tree and the process repeated recursively until a stopping condition, such as a certain proportion of examples belonging to the same class is reached. However, this selection measure does not take account of costs.

A natural way of making such algorithms cost-sensitive, which has been attempted by several authors, is to modify the above measure to include misclassification costs (Liu, 2007; Norton, 1989; Tan & Schlimmer, 1990; Zhao & Li, 2017). However, empirical evaluations of this approach have showed mixed results (Lomax, S. and Vadera, 2011; Vadera & Ventura, 2001). Hence, instead of adapting the information gain measure to include costs, other algorithms utilize the cost of misclassification directly as the selection criteria. Examples of algorithms that take this approach include Cost-Minimization (Pazzani et al., 1994), Decision Tree with Minimal Costs (Ling & Sheng, 2004), PM (Liu, 2007), Cost-Sensitive Non Linear Decision trees (CSNL) (Vadera, 2010), and Cost-Sensitive Decision Trees (CSDT) (Bahnsen, Aouada, & Otterssten, 2015).

[3] Appendix A provides an expansion of the acronyms used in Figure 1.

2.2 Algorithms that utilize bagging and boosting

These algorithms make use of a technique known as bagging (Breiman, 1996) which applies a base learner on different samples of training data and combines the outcomes to predict the class of each example that reduces the cost. One of the earliest and most widely cited examples is the MetaCost system (Domingos, 1999), which uses bagging to relabel the data to minimize classification cost and then applies the base learner on the relabeled data to induce a classifier, as summarized in Figure 2. A more recent method, due to Li et al. (2018), proposes an attribute selection measure that is a function of the Gini index, information gain and misclassification cost. This measure is used in an algorithm known as CHMDT (Cost-sensitive and Hybrid attribute Multi-Decision Tree) to rank the attributes, select the top n attributes and induce n trees, where each tree has one of the n attributes as a root node. Classification is then performed by applying the n trees and using majority voting.

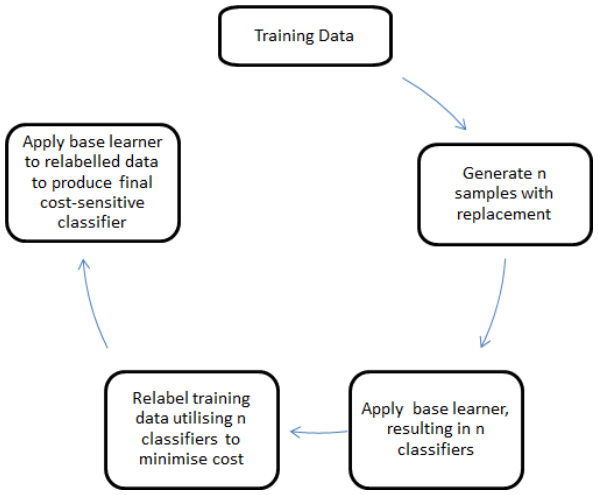

Figure 2: The MetaCost system (Domingos, 1999)

Boosting applies a number of hypotheses and then combines them to form a more accurate composite hypothesis. One of the earliest examples of boosting is AdaBoost (Adaptive Boosting (Freund & Schapire, 1996)) which uses an accuracy based learner to generate an improving sequence of hypotheses. AdaBoost starts the boosting process by assigning unit weights to each example, then in each sequential trial, it increases the weights of misclassified examples and decreases the weights of the other examples. After many sequential trials, it combines these hypotheses to perform a final classification which is based on selecting the class that results in the maximum weighted vote. Several algorithms that build on this concept have been developed in recent years, including XGBoost (Extreme Gradient Boosting) which has been credited with winning several submissions to the Kaggle challenges (Chen and Guestrin , 2016).

There are several studies that use boosting and modify the weighting rules to take account of costs, including AdaCost (Adaptive Boosting with Costs) (Fan et al., 1999), Cost-UBoost (Cost with Unequal instance weights boosting) (Ting & Zheng, 1998), and GBSE (Gradient Boosting with Stochastic Ensembles)
(Abe et al., 2004). For example, AdaCost uses the cost of misclassifications to assign high initial weights to costly examples. It then increases the weights of costly misclassifications and decreases the weights of correct classifications before another round of boosting. Masnadi-Shirazi and Vasconcelos (2011) propose a framework that enables derivation of three cost-sensitive boosting algorithms, CS-Ada, CS-Log and CS-Real, each based on their respective cost-insensitive versions: AdaBoost, LogitBoost and RealBoost. An empirical comparison showed that CS-Ada performed well in comparison to other cost-sensitive adaptations of AdaBoost, such as AdaCost. However, in a more recent paper, "Cost-sensitive boosting algorithms: Do we really need them?", Nikolaou et al. (2016) present a critique of cost-sensitive boosting algorithms from multiple perspectives and conclude that Adaboost performs just as well as other variations of boosting algorithms.

2.3 Algorithms that use Genetic algorithms

Genetic algorithms (GAs) have been utilized by several authors to learn cost-sensitive decision trees (Kretowski & Czajkowski, 2018; Nikdel & Beigy, 2012; Omielan & Vadera, 2012; Turney, 1995). Here we describe one of the earliest and most seminal studies and then summarize studies that use GAs for evolving Bayesian networks given their relationship to the work presented in this paper.

Turney's (1995) ICET system (Inexpensive Classification with Expensive Tests) was one of the first to use GAs to evolve decision trees in order to minimize both test costs and misclassification costs. ICET uses a genetic pool that consists of genes representing the cost of attributes ($Cost_A$), a parameter used to control the amount of weight that should be given to the cost of attributes ($\boldsymbol{\omega}$), and a parameter (CF) used to indicate the level of pruning by C4.5.

These parameters are used in a version of C4.5 to generate trees, where, instead of Equation (1), the following measure, known as the Information Cost Function (ICF) is used to rank attributes:

$$ICF_A = \frac{2^{I_A} - 1}{(Cost_A + 1)^{\omega}} \qquad (2)$$

In ICET, trees are not represented in a genetic pool directly, but are learnt using the genes as parameters for a tree induction algorithm (C4.5) as illustrated in Figure 3. Following this process, the decision trees are evaluated using expected cost as a fitness function, and a new pool is produced using mutation and cross over. This process is repeated 50 times and the fittest tree returned.

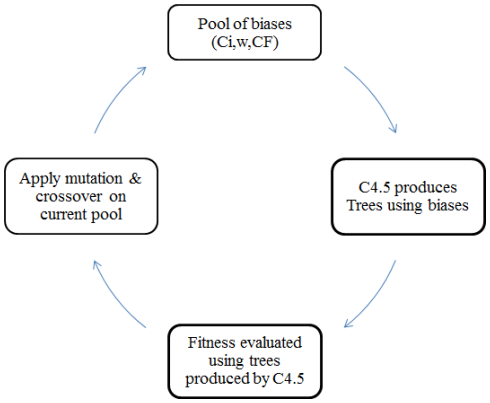

Figure 3: The ICET System (Turney, 1995)

Several authors have also studied the use of GAs for evolving the structure of Bayesian networks. Larrañaga et al.(1996), represent a Bayesian network as an ordered list of nodes in which a node can only have preceding nodes as parents. In contrast to the algorithm developed by Larrañaga et al. (1996), which focuses on evolving directed acyclic graphs, Zeng et al. (2006), evolve an extended Naïve Augmented Bayes network (EANB) in which each attribute can have one other attribute as a parent. Zeng et al. (2006) compare their algorithm with Naïve Bayes (Langley et al., 1992), C4.5(Quinlan, 1993), and Tree Augmented Naïve Bayes (N. Friedman et al., 1997), and conclude that it outperforms these algorithms in terms of accuracy. These algorithms have some similarities with the use of GAs in this paper, though our primary focus is on cost-sensitive Bayesian networks and their relative merits in comparison to cost-sensitive decision trees.

3 DEVELOPMENT OF EBNO

Section 2 described various approaches that have been used to develop algorithms that induce cost-sensitive decision trees. Any of these approaches could be adopted for generating Bayesian networks that take account of costs of misclassification, and in a previous study, we reported attempts at using a direct approach as well as changing the distribution of examples to reflect the misclassification cost (Nashnush & Vadera, 2017). In this section we formulate a GA for evolving cost-sensitive Bayesian networks.

As described in Section 1, learning a Bayesian network consists of two parts: a *qualitative part* that learns a structure and a *quantitative part* that learns the parameters of the structure. There are different types of Bayesian networks and learning them is known to be an NP-hard problem (Chickering et al., 2004). Hence, several algorithms have been developed that reduce the size of the search space by limiting the type of topology that is learned. Chow and Liu (1968) developed an algorithm for learning Bayesian trees based on approximating the joint distribution of a set of discrete variables using the products of distributions involving no more than pairs of variables as illustrated in Figure 4(a). Duda and Hart (1973) and Langley et al. (1992) developed an algorithm for learning Naïve Bayes structures, where the attributes are represented as independent nodes that have one parent node which represents the class. A Naïve Bayes classifier, as shown in Figure 4(b), assumes conditional independence of the features given the class. Naïve Bayes is easy to construct and has been used as a classifier for many years, especially where the features are not strongly correlated. Pearl (1988) developed an algorithm to learn singly-connected graphs, which are Directed Acyclic Graphs (DAGs) where any two nodes have at most one unique path between them, as illustrated in Figure 4(c).

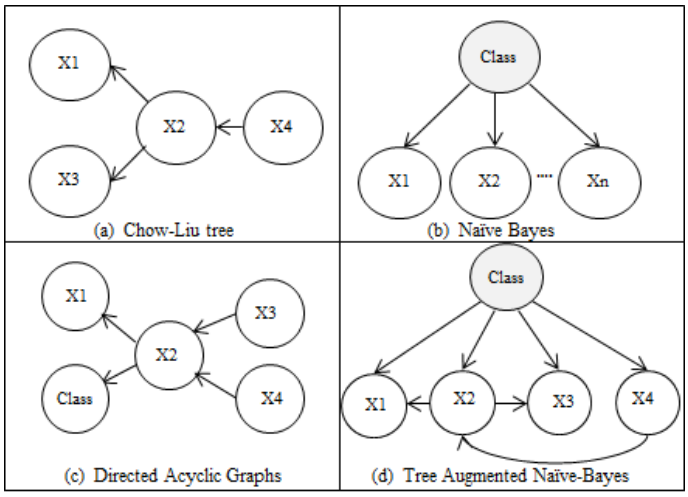

Figure 4: Types of Bayesian Structures

More recently, Friedman et al. (1997) have developed a natural extension to the Naïve Bayes classifier, known as a *Tree Augmented Naïve Bayes* (TAN) structure. In contrast to Naïve Bayes, where the assumption is that all attributes are independent, a TAN can model dependencies between attributes by allowing the attributes to form a tree. Thus, in a TAN structure, the correlations between attributes can be captured by adding additional edges between attributes, as illustrated in Figure 4(d). In this study we adopt TANs, since they make fewer assumptions than Naïve Bayes networks, avoid the computational overhead of DAGs, and have been shown to be effective classifiers (N. Friedman et al., 1997).

The structure of a TAN can be viewed as a directed graph which can be represented by an adjacency matrix A, where an element A(i,j) is set to 1 if node j is a parent of node i, and set to 0 if there is no link between node j and node i. Figure 5 illustrates the idea, where node a_0 has two parents a_2 and a_4, and hence A(0,2)=A(0,4)=1; while there are no links between a_1 and a_3, so A(1,3)=A(3,1)=0.

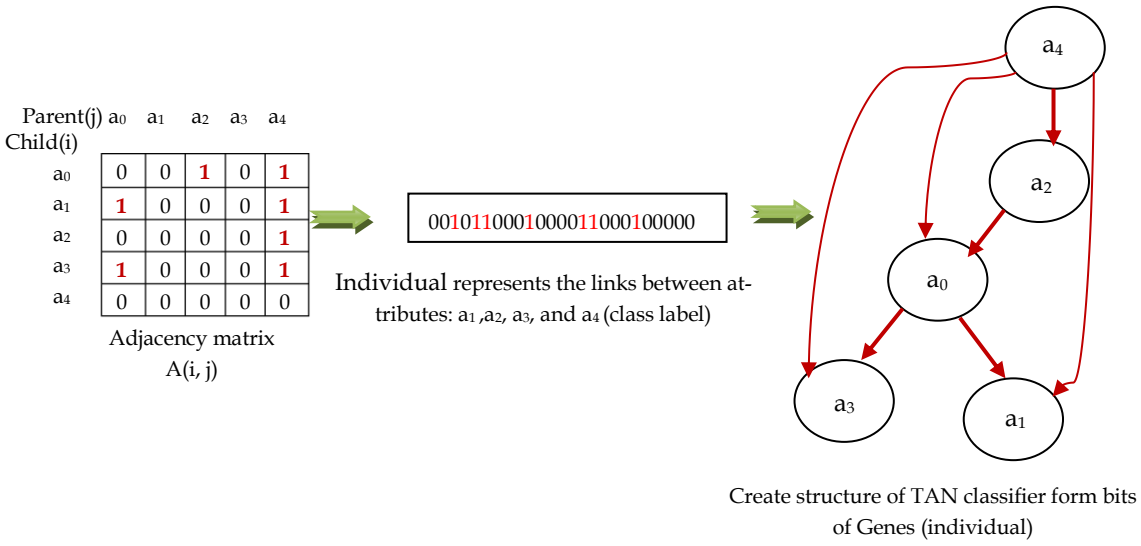

Figure 5: An illustration of how TAN classifiers are represented.

Generating the initial pool of TANs for a GA involves three steps: (i) generating the adjacency matrix randomly, (ii) testing the adjacency matrix to ensure that it represents a valid TAN, and if not, to make it a TAN, and (iii) converting the adjacency matrix to a linear string of bits that can be used by a GA. Figure 6 illustrates the first of these steps, showing a case when an illegal TAN is obtained, either when generating the initial population or following mutation or crossover.

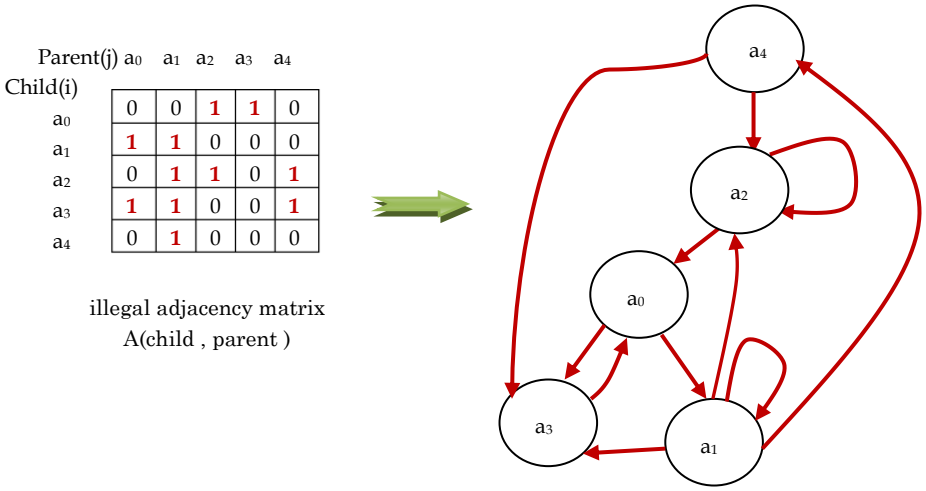

Figure 6: An example of an illegal TAN structure created from an adjacency matrix

The following two steps correct such illegal TANs:

(i) Any circularities are removed: immediate circularities, such as a_2 to a_2 in Figure 6, are removed and paths that are circular are broken by randomly removing a link in the circular chain.

(ii) By definition of a TAN, the class node must have no parents, and all the other nodes must have the class node as a parent. In addition, each node, except one node which is labelled the root, has one other parent that is chosen from the other nodes. If this is not the case, then this is corrected by making sure the class node is added as a parent, and one of the other nodes is chosen randomly as the other parent.

Once an adjacency matrix representing a valid TAN is obtained, it can be converted to a string of bits by arranging it row by row as illustrated in Figure 5. As well as the representation, there are two more ingredients required to use a GA, namely a fitness function and the operators required for generating offspring. To generate the offspring, the standard selection, mutation and crossover operators are used together with the above steps for correcting illegal offspring. The fitness function used in EBNO is the expected cost:

$$Fitness\ function = \sum_{i=1}^{k}\sum_{j=1}^{k} Cij.Pij \qquad (4)$$

where, k is the number of classes, Pij represents the probability of classifying an example of class i as j and Cij is the cost of classifying an example of class i as class j.

Given the above representation and fitness function, Figure 7 summarizes the EBNO algorithm.

Input:

Training data D
Misclassification cost matrix Cij
Number of generations of evolution N

Output: A Tree Augmented Network (TAN)

1. Randomize D, and divide it into 3 parts:
   a. Dp: 50% for parameter learning,
   b. Df: 25% for the fitness function,
   c. Dt: 25% for testing.
2. Generate an initial population of 50 individuals, {P_1, …,P_{50}} where each individual represents a network of connections.
3. Check that the individuals represent valid TANs: namely that there are no circular paths, each node should have just one parent, and the class label is the main parent for all nodes.
4. Create 50 well-formed TANs, Ti, one from each individual Pi.
5. Learn the parameters for each of the 50 TANs, Ti, using the data in Dp.
6. Evaluate the fitness of each TAN Ti using the data Df, and cost matrix Cij using Equation (4)
7. Produce the next generation as follows:
   a. P_1 is set to the individual that has minimum cost in the previous generation.
   b. P_2,..,P_{25} are obtained by using the standard mutation and crossover operators on the best 25 individuals in the previous generation.
   c. P_{26},..,P_{50} are generated randomly.
8. Repeat from step 3 for N cycles and then return the minimum cost TAN as the output.

Figure 7: The EBNO Algorithm

4 EMPIRICAL EVALUATION AND RESULTS

This section presents an empirical evaluation of the EBNO algorithm using 28 data sets from the UCI repository which are listed in Table 1 (Appendix B) together with their characteristics (Dua & Taniskidou, 2017). The experimental methodology involves using 75% of the data for training and 25% for testing. The 75% of training data is further subdivided so that 50% is used for learning the parameters and 25% is used for assessing the fitness. All experiments are repeated with 10 random trials and the results report the average cost, accuracy, precision, recall and F measure. For consistency, the same misclassification costs were used for all the data sets, with the experiments performed using 16 different cost ratios for the two classes: [4:1, 4:2, 4:3, 4:4, 3:1, …, 2:4,1:4].

Given the questions raised in the introduction, the evaluation is carried out with respect to the following algorithms:

(i) The ICET algorithm to enable comparison with the results from a well-known cost-sensitive decision tree learning algorithm that is known to perform well (Lomax, S. and Vadera, 2011; Vadera & Ventura, 2001).

(ii) MetaCost with TAN as the base classifier to enable comparison with an algorithm that produces cost-sensitive Bayesian networks. In this study, our choice of MetaCost was influenced primarily by the fact that it generates a model which can be viewed and interpreted by a user. It also remains an option available in toolkits such as Weka, Scikit learn and R, so the results should be of interest to practitioners.

(iii) The original TAN learning algorithm in order to assess the extent to which EBNO makes a difference in comparison to an algorithm that focuses on maximizing accuracy of TANs.

(iv) Use of two boosting algorithms: AdaBoost to induce cost-sensitive TANs and use of XGBoost to induce cost-sensitive decision trees. Although AdaBoost is one of the earliest boosting algorithms, as the study by Nikolaou et al. (2016) concludes, it remains an important method for boosting and cost-sensitive learning. XGBoost is included primarily because it is a more recent innovation and has resulted in some of the best results in Kaggle competitions.

The WEKA implementations of MetaCost and algorithm for learning TANs were utilized for the evaluations, and for consistency, EBNO was also implemented in WEKA, with default settings for the probability of cross-over rate (0.6), mutation (0.033) and generations (20).[4] For ICET, an implementation that has been used in other work and verified as faithful was used (Lomax, S. and Vadera, 2011; Vadera & Ventura, 2001). For AdaBoost and XGBoost, we used the implementations provided by the Scikit learn package in R.

Tables 2 and 3 (Appendix B) present the average misclassification cost, accuracy, precision, recall and F measure when each of the six methods is applied to the 28 data sets. To compare the performance of the algorithms, we follow the recommendations by Demšar (2006) who carries out an extensive study of different parametric and non-parametric methods for comparison of machine learning algorithms, and concludes by advocating the use of non-parametric methods. More specifically, Demšar (2006) recommends the use of a test introduced by Friedman (1937) to determine if there is a difference amongst the algorithms and if so, to follow up with the use of the Nemenyi test (1963) to assess if one method is significantly better than another. Figure 8 shows a box plot of the misclassification costs for each algorithm, giving a visual indication of the differences.

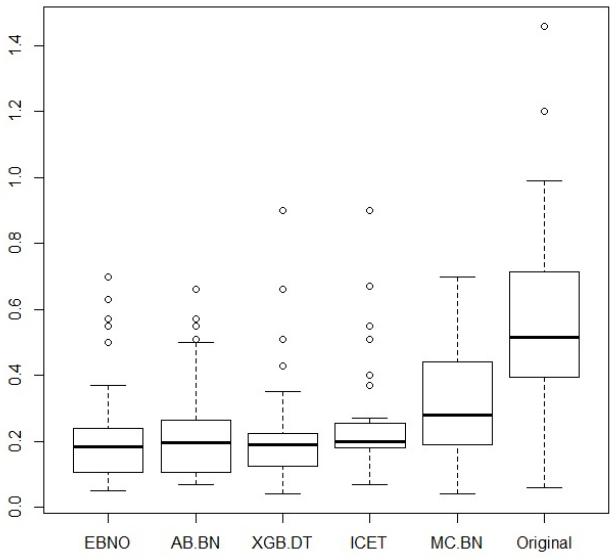

Figure 8: Box plot of the misclassification cost for each algorithm

[4] The default settings were used to allow easy repeatability of the experiments and for consistency with ICET which also adopts default values.

Carrying out a Friedman test on the misclassification costs results in a p-value of 5.5e-14, confirming a significant difference amongst the algorithms. Table 4 presents the results of a Nemenyi test between pairs of algorithms, showing that there is a significant difference between EBNO and the other algorithms that also produce a single interpretable model Figure 9 summarizes the relative performance of the algorithms using a Critical Difference (CD) diagram, where the CD is defined as the minimum distance that must be exceeded to reject the null hypothesis that two algorithms are the same.

TABLE 4

Results from the pairwise Nemenyi test

	EBNO	AB.BN	XGB.DT	ICET	MC.BN
AB.BN	0.988				
XGB.DT	0.992	1.000			
MC.BN	0.004	0.036	0.029		
ICET	0.135	0.452	0.406	0.863	
Original	1.1e-10	1.1e-10	4.9e-09	0.014	0.0002

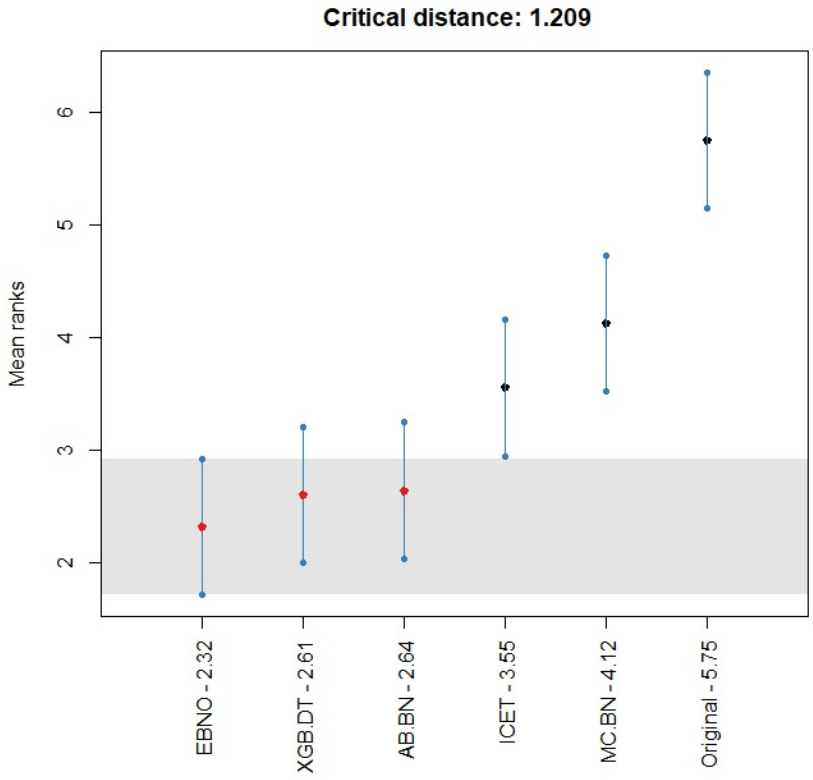

Figure 9: Critical Difference (CD) diagram with α= 0.15

Thus, based on the above results we can surmise that, in general:

(i) As expected, use of cost-sensitive algorithms such as EBNO, AB.BN, XGB.DT, ICET and MC.BN results in more cost-effective decisions than the use of the original cost-insensitive Bayesian learning algorithm (Figures 8 and 9).

(ii) There is little to distinguish between EBNO and the use of boosting to generate cost-sensitive models. It is however, worth noting that EBNO generates an explicit model that can be viewed by a user; something that is likely to become an important consideration as greater attention is paid to ensuring the fairness of the models learned.

(iii) EBNO is better at minimizing cost than the other algorithms that also aim to produce explicit models (ICET, MC.BN, Original) More specifically, EBNO is ranked first for 19 out of the 28 data sets. For the remaining 9, where EBNO is ranked 2nd best, it is outperformed by ICET on 5 data sets (Breast Cancer, Cylinder Band, Haberman, Sonar, and SPECT-Heart) and by MC.BN on 4 data sets (German Credit, Kr vs Kp,Musk, Sick). Given that both ICET and EBNO utilize GAs, this suggests that evolving cost-sensitive BNs has some merit over evolving cost-sensitive decision trees.

(iv) EBNO performs very well in terms of accuracy, recall, precision and F measure (Tables 2 and 3, Appendix B) in comparison to the other cost-sensitive algorithms. It even remains competitive in terms of accuracy with the use of cost-insensitive Bayesian networks which aim to maximize accuracy. However, the precision and recall measures (Table 3, Appendix B) reveal that the improvements in cost sensitivity are due to improving recall rates over precision when compared to the use of cost-insensitive Bayesian networks (Table 3, Appendix B).

These improvements do, of course, come at a price, since use of evolution can be computationally more expensive than greedy algorithms such as Friedman et al.'s(1997) algorithm for learning TANs. Equally, it can also be argued that this is worthwhile given there is little difference in computational cost once a classifier is deployed.

5 CONCLUSION AND FUTURE WORK

Throughout the past decade, the problem of developing algorithms that induce cost-sensitive classifiers has become a significant challenge. Most of the research on cost-sensitive learning algorithms has focused on induction of decision trees with either direct amendments to existing algorithms or use of indirect methods such as bagging. Bayesian networks have been shown to be an effective classifier, and hence an obvious question is whether they can result in classifiers that perform better than decision trees when it comes to minimizing costs of misclassification.

This research has aimed to address this question by developing an evolutionary algorithm, EBNO, for learning Bayesian networks that aim to minimize costs of misclassification. An empirical evaluation of the algorithm relative to other algorithms shows that:

- As one would expect, EBNO outperforms a cost-insensitive version of an algorithm that learns Bayesian networks.
- An empirical comparison with ICET, which also adopts GAs, shows that there are merits in using Bayesian networks over decision trees for cost-sensitive learning.
- EBNO, which generates an explicit model, performs as well as the use of AdaBoost and XGBoost, in terms of cost-sensitive classification, and hence should be considered where an application benefits from the transparency of a single model that can be viewed by a user.
- Perhaps more surprisingly, on many of the cases, EBNO did not compromise accuracy, because the use of a GA led to exploration of a wide range of potential solutions.

In conclusion, given the results of this study, the evolution of cost-sensitive Bayesian networks should be considered a serious alternative to cost-sensitive decision trees. One direction for future work is to investigate how to extend EBNO so it can take account of other types of costs, such as test costs. Our empirical evaluation used the default parameters for the GA used in EBNO and some experimentation and tuning of these parameters might lead to even better performance. Use of evolution can be computationally expensive, particularly as the number of features increases, so another direction of work is to explore the use of parallel implementations of

GAs (Harding & Banzhaf, 2007) so that EBNO can be applied to very large data sets.

ACKNOWLEDGEMENTS

The authors are grateful to the reviewers for their comments which have led to several improvements to this paper. The authors acknowledge that the work presented in this paper is based on the first author's Doctoral thesis titled "Development of new cost-sensitive Bayesian network learning algorithms" (Eman Nashnush, 2015) carried out under the supervision of the second author. An e-copy of the thesis is available from the University of Salford Repository at http://usir.salford.ac.uk.

REFERENCES

Abe, N., Zadrozny, B., & Langford, J. (2004). An iterative method for multi-class cost-sensitive learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '04).* (pp. 3–11). ACM New York, NY, USA.

Bahnsen, A., Aouada, D., & Ottersten, B. (2015). Example-dependent cost-sensitive decision trees. *Expert Systems with Applications*, *42*(19), 6609–6619.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Campos, L. M. De, Fernandez-Luna, J. M., Gámez, J. A., & Puerta, J. M. (2002). Ant colony optimization for learning Bayesian networks. *International Journal of Approximate Reasoning*, *31*(3), 291–311.

Cheng, J., & Greiner, R. (2001). Learning Bayesian belief network classifiers: Algorithms and System. In *Advances in Artificial Intelligence* (pp. 141–151). Berlin Heidelberg: Springer.

Chickering, D. M., Heckerman, D., & Meek, C. (2004). Large sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research*, *5*, 1287–1330.

Chow, C., & Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, *14*(3), 462–467.

Dai, J., Han, H., Hu, Q., & Liu, M. (2016). Discrete particle swarm optimization approach for cost sensitive attribute reduction. *Knowledge-Based Systems*, *102*, 116–126. https://doi.org/10.1016/j.knosys.2016.04.002

Dasgupta, S. (1999). Learning Polytrees. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence* (pp. 134–141).

Davis, J. V., Ha, J., Rossbach, C. J., Ramadan, H. E., & Witchel, E. (2006). Cost-sensitive decision tree learning for forensic classification. In *17th European Conference on Machine Learning* (pp. 622–629).

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Domingos, P. (1999). MetaCost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 155–164).

Dong, S., & Wu, Y. (2018). A diversity-based method for class-imbalanced cost-sensitive learning. In *International Conference on Mathematics and Artificial Intelligence* (pp. 51–55). Chengdu, China: ACM New York, NY, USA.

Drummond, C., & Holte, R. C. (2000). Exploiting the cost (in) sensitivity of decision tree splitting criteria. In *International Conference on Machine Learning* (pp. 239–246).

Dua, D., & Taniskidou, K. E. (2017). UCI Machine Learning Repository. Retrieved June 10, 2018, from https://archive.ics.uci.edu/ ml

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley.

Fan, W., Stolfo, S. J., Zhang, J., & Chan, P. K. (1999). AdaCost: Misclassification Cost-Sensitive Boosting. In I. Bratko & S. Dzeroski (Eds.), *Proceedings of the Sixteenth International Conference on Machine Learning* (pp. 97–105). Morgan Kaufmann Publishers Inc.

Freitas, A., Costa-Pereira, A., & Brazdi, P. . (2007). Cost-Sensitive Decision Trees Applied to Medical Data. In `Song I.Y, E. J., & N. T.M (Eds.), *Lect. Notes Comput. Sci.*, (Vol. 4654, pp. 303–312). Springer Berlin Heidelberg.

Freund, Y., & Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 148–156). Morgan Kaufmann Publishers.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, *32*, 675–701.

Friedman, N., Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*(2–3), 131–163.

Harding, S., & Banzhaf, W. (2007). Fast genetic programming on GPUs. In *Proceedings of the European Conference on Genetic Programming* (pp. 90–101). Springer Berlin Heidelberg.

Heckerman, D., Mamdani, A., & Wellman, M. P. (1995). Real-world applications of Bayesian networks. *Communications of the ACM*, *38*(3), 24–26.

Jiang, L., Li, C., Cai, Z., & Zhang, H. (2013). Sampled Bayesian network classifiers for class-imbalance and cost-sensitive learning. In *25h IEEE International Conference on Tools with Artificial Intellligence (ICTAI)* (pp. 512–517).

Jiang, L., Li, C., & Wang, S. (2014). Cost-sensitive Bayesian network classifiers. *Pattern Recognition Letters*, *45*, 211–216.

Kretowski, M., & Czajkowski, M. (2018). Evolutionary Algorithms for Global Decision Tree Induction. In *Encyclopedia of Information Science and Technology* (4th ed., pp. 2132–2141). Hershey, PA: IGI Global.

Langley, P., Iba, W., & Thompson., K. (1992). An analysis of Bayesian classifiers. In *Proceedings of the tenth national conference on Artificial intelligence (AAAI'92)* (pp. 223–228).

Larrañaga, P., Poza, M., Yurramendi, Y., Murga, R. H., & Kuijpers, C. M. (1996). Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(9), 912–926.

Lauritzen, S. L., & Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 157–224.

Li, F., Zhang, X., Zhang, X., Du, C., Xu, Y., & Tian, Y.-C. (2018). Cost-sensitive and hybrid -attribute measure multi-decision tree over imbalanced data sets. *Information Sciences*, *422*, 242–256.

Li, J., Li, X., & Yao, X. (2005). Cost-sensitive classification with genetic programming. In *IEEE Congress on Evolutionary Computation,* (pp. 2114–2121). https://doi.org/doi: 10.1109/CEC.2005.1554956

Ling, C. X., & Sheng, V. S. (2004). Cost-sensitive learning. In *Encyclopedia of Machine Learning* (pp. 231–235). Springer.

Liu, X. (2007). A New Cost-Sensitive Decision Tree with Missing Values. *Asian Journal of Information Technology*, *6*(11), 1083–1090.

Lomax, S. and Vadera, S. (2011). An empirical comparison of cost-sensitive decision tree induction algorithms. *Expert Systems: The Journal of Knowledge Engineering*, *28*(3), 227–268.

Lomax, S., & Vadera, S. (2013). A survey of cost-sensitive decision tree induction algorithms. *ACM Computing Surveys*, *45*(2), Article 16:1--35.

Lomax, S., & Vadera, S. (2017). A Cost-Sensitive Decision Tree Learning Algorithm Based on a Multi-Armed Bandit Framework. *The Computer Journal*, *60*(7), 941–956. https://doi.org/10.1093/comjnl/bxw015

Masnadi-shirazi, H., & Vasconcelos, N. (2011). Cost-Sensitive Boosting. *IEEE Transaction Pattern Analysis and Machine Intelligence*, *33*(2), 294–309.

Mitchell, T. M. (1997). Does machine learning really work? *AI Magazine*, *18*(3), 11.

Moret, S., Langford, W., & Margineantu, D. (2006). Learning to predict channel stability using biogeomorphic features. *Ecol. Modell*, *191*(1), 47–57.

Nashnush, E., & Vadera, S. (2014). Cost-sensitive Bayesian network learning using sampling. In *Advances in Intelligent Systems and Computing* (Vol. 287, pp. 467–476).

Nashnush, E., & Vadera, S. (2017). Learning cost-sensitive Bayesian networks via direct and indirect methods. *Integrated Computer-Aided Engineering*, *24*(1), 17–26. https://doi.org/10.3233/ICA-160514

Nashnush, Eman. (2015). *Development of new cost-sensitive Bayesian network learning algorithms, PhD Thesis*. University of Salford.

Nemenyi, P. B. (1963). *Distribution-free multiple comparisons, PhD thesis,*. Princeton University.

Nikdel, Z., & Beigy, H. (2012). A genetic programming-based learning algorithms for pruning cost-sensitive classifiers. *International Journal of Computational Intelligence and Applications*, *11*(2), 1–16.

Nikolaou, N., Narayanan, E., Kull, M., Flach, P., & Brown, G. (2016). Cost-sensitive boosting algorithms: Do we really need them? *Machine Learning*, *104*(2–3), 359–384.

Norton, S. W. (1989). Generating better decision trees. In *IJCAI'89 Proceedings of the 11th International Joint Conference on Artificial Intelligence* (Vol. 1, pp. 800–805).

Núñez, M. (1991). The use of background knowledge in decision tree induction. *Machine Learning*, *6*, 231–250.

Omielan, A., & Vadera, S. (2012). ECCO: A New Evolutionary Classifier with Cost Optimisation. In *Intelligent*

Information Processing VI (pp. 97–105). Berlin Heidelberg: Springer.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing misclassification costs. In *Proceedings of the 11th International Conference on Machine Learning* (pp. 217–225). New Brunswick, New Jersey, USA.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, California: Morgan Kaufmann Publishers.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.

Tan, M., & Schlimmer, J. (1990). Two Case Studies in cost-sensitive concept acquisition. In *Proceedings AAAI* (pp. 854–860). AAAI. Retrieved from http://www.aaai.org/Papers/AAAI/1990/AAAI90-128.pdf

Tan, M., & Schlimmer, J. C. (1989). Cost-sensitive concept learning of sensor use in approach and recognition. In *Proceedings of the sixth international workshop on Machine learning* (pp. 392–395). Morgan Kaufmann Publishers.

Ting, Kai Ming. (2002). An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering*, *14*(3), 659–665.

Ting, K.M., & Zheng, Z. (1998). Boosting cost-sensitive trees. *Discovery Science*, 244–255.

Turney, P. D. (1995). Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 369–409.

Vadera, S. (2010). CSNL: A cost-sensitive non-linear decision tree algorithm. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *4*(2), 1–25.

Vadera, S., & Ventura, D. (2001). Comparison of Cost-Sensitive Decision Tree Learning Algorithms. In *Proceedings of the Second European Conference on Intelligent Management Systems in Operations* (pp. 79–86). Operational Research Society, UK.

Zeng, D., Zhang, S., Cai, Z., Jiang, S., & Jiang, L. (2006). Augmented Naive Bayes Based on Evolutional Strategy. In *Proceedings of the sixth IEEE International Conference on Intelligent Systems Design and Applications* (pp. 446–450).

Zhao, H., & Li, X. (2017). A cost sensitive decision tree algorithm based on weighted class distribution with batch deleting attribute mechanism. *Information Sciences*, *378*, 303–316. https://doi.org/10.1016/J.INS.2016.09.054

APPENDIX A

List of Acronyms in Figure 1 together with their meanings.

Algorithms that modify the induction process	
CS ID3 (Tan & Schlimmer 1989)	Cost-Sensitive Iterative Dichotomiser 3
EG2 (Nunez 1991)	Economic Generalizer 2
IDX (Norton 1998)	Iterative Dichotomiser X
C4.5CS (Ting 2002)	C4.5 Cost Sensitive
CSGain (Davis et al. 2006)	Cost Sensitive Gain
CS C4.5 (Frietas et al. 2007)	Cost Sensitive C4.5
CSNL (Vadera, 2010)	Cost Sensitive Non-Linear
CSDT (Bahnsen et al., 2015)	Cost Sensitive Decision Tree
MA-CSDT (Lomax & Vadera 2017)	Multi-Armed Bandit - Cost Sensitive Decision Tree
BDADT (Zhao & Li 2017)	Batch Deleting Attribute Decision Tree
Algorithms that use bagging and boosting	
Cost-Uboost (Ting & Zheng 1998)	Cost with Unequal instance weights boosting
MetaCost (Domingos 1999)	Meta algorithm for Cost sensitive learning
AdaCost (Fan et al. 1999)	Adaptive Boosting with Costs
GBSE (Abe et al. 2004)	Gradient Boosting with Stochastic Ensembles
CS-ADA, CS-Real, CS-Log (Masnadi-Shirazi & Vasconcelos 2011)	Cost Sensitive-AdaBoost, Cost-Sensitive LogitBoost and Cost-Sensitive RealBoost
CHMDT (Li et al. 2018)	Cost-sensitive and Hybrid attribute measure Multi- Decision Tree
Algorithms that use GAs	
ICET (Turney 1995)	Inexpensive Classification with Expensive Tests
CGP (Li et al. 2005)	Constrained Genetic Programming cost-sensitive classifier
ECCO (Omielan & Vadera 2012)	Evolutionary Classifier with Cost Optimisation
DTGP (Nikdel & Beigy 2012)	Decision Trees pruning with Genetic Programming

APPENDIX B

TABLE 1. Characteristics of the data sets used for empirical evaluation.

Dataset	Class distribution	Instances	Attributes	Type of attributes
Adult	(76 : 24)	48842= (37155,11687)	14	5 continuous
Australian Credit	(56 : 44)	690=(383, 307)	15	5 continuous
Bank	(54 : 46)	600=(362, 274)	11	2 continuous
Breast Cancer	(70 : 30)	286=(201,85)	9	All nominal
Bupa liver disorder	(58 : 42)	345=(200, 145)	7	6 continuous
Cleveland disease	(54 : 46)	303=(165,138)	13	5 continuous
Crx	(56 : 44)	689=(382,307)	16	6 continuous
Cylinder Band	(58 : 42)	540=(312,228)	39	17 continuous
German credit	(70 : 30)	1000=(700,300)	20	7continuous
Gymexamg	(70 : 30)	2500=(1755,745)	20	11 continuous
Haberman	(74 : 26)	306=(225,81)	3	2 continuous
Hepatitis	(97 : 23)	155=(32, 123)	19	6 continuous
Horse Colic	(63 : 37)	368=(214,152)	22	14 continuous
Horse	(66:34)	370=(215,153)	28	8 continuous
IonoSphere	(64 : 36)	351=(225,126)	34	23 continuous
kr-vs-kp	(52 : 48)	3196=(1669,1527)	36	All nominal
Labor	(65 : 35)	57=(37,20)	16	8 continuous
Musk	(52 : 48)	476=(207,269)	168	166 continuous
Pima_diabetes	(57 : 43)	768=(500,268)	8	All continuous
Sick	(94 : 6)	2800=(171, 2629)	29	7 continuous
Sonar	(53 : 47)	280=(111,97)	60	All continuous
Spambase	(61 : 39)	4601=(2788,1813)	57	All continuous
SPECT Heart	(59 : 41)	267=(157,110)	22	All nominal
Supermarket	(64 : 36)	4627=(2948,1679)	216	All nominal
Tic-Tac-Toe	(65 : 35)	958=(626,332)	9	All nominal
Vote	(61 : 39)	435=(267,168)	16	All nominal
Weather	(64 : 36)	14=(9,5)	5	All continuous
Wisconsin Cancer	(66 : 34)	699=(458,241)	10	All continuous

TABLE 2. Average Cost and Accuracy. Columns labeled Cost are presented in the format Average Cost± Standard Error.

Dataset	EBNO algorithm		ICET		MetaCost.BN		Original BN		Adaboost BN		XGBoost	
	Cost	Accuracy	Cost	Accuracy	Cost	Accuracy	Cost	Accuracy	Cost	Accuracy	Cost	Accuracy
Adult	0.25±0.17	0.84	0.27±0.12	0.80	0.29±0.44	0.80	0.38±0.12	**0.86**	**0.23±0.12**	0.84	0.25±0.1	0.81
Australian Credit	0.19±0.12	0.85	0.22±0.15	0.85	0.25±0.38	0.84	0.43±0.17	**0.86**	**0.11±0.18**	0.83	0.22±0.33	0.84
Bank	**0.20± 0.19**	0.74	**0.20± 0.11**	0.66	0.25±0.31	0.74	0.53±0.37	**0.76**	0.22±0.2	0.74	**0.20± 0.9**	0.66
Breast Cancer	0.23±0.21	**0.73**	**0.15±0.99**	0.70	0.29±0.30	**0.73**	0.41±0.36	0.70	0.3±0.18	**0.73**	**0.15±0.79**	0.72
Bupa liver disorder	**0.22±0.80**	0.61	0.24±0.73	0.62	0.37±0.39	0.60	0.43±0.49	0.60	**0.22±0.74**	**0.71**	0.23±0.66	0.61
Cleveland disease	0.13±0.15	0.69	0.22±1.44	0.65	0.19±0.18	0.70	0.43±0.12	**0.86**	**0.11±0.19**	0.67	0.15±1.37	0.66
Crx	**0.10±0.58**	**0.86**	0.19±1.63	0.83	0.21±0.14	0.81	0.5±0.053	**0.86**	0.19±1.6	0.82	0.19±1.61	0.85
Cylinder Band	0.57±0.26	**0.76**	**0.51±0.44**	0.72	0.69±0.33	0.74	0.73±0.88	0.74	0.57±0.34	**0.76**	**0.51±0.1**	0.72
German credit	0.20±0.30	0.73	0.20±0.21	0.62	**0.19±0.28**	0.73	0.56±0.29	**0.74**	0.2±0.38	0.73	**0.19±0.27**	**0.74**
Gymexamg	**0.13±0.11**	0.66	0.19±0.21	0.63	0.23±0.32	0.60	0.91±0.42	**0.70**	0.15±0.39	0.62	0.15±0.21	0.69
Haberman	0.7±0.13	0.72	**0.67±0.38**	**0.73**	0.70±0.21	0.68	0.99±0.37	0.70	0.66±0.26	0.71	**0.66±0.1**	**0.73**
Hepatitis	**0.10±0.10**	**0.90**	0.55±0.20	0.78	0.11±0.78	0.88	0.7±0.15	**0.90**	**0.10±0.10**	**0.90**	0.15±0.7	0.82
Horse Colic	**0.20±0.28**	0.66	**0.20±0.28**	0.65	0.37±0.34	0.65	0.58±0.34	0.63	**0.20±0.24**	**0.68**	**0.20±0.1**	**0.68**
Horse	0.12±0.19	**0.79**	0.18±0.23	**0.79**	0.66±0.54	0.70	0.70±0.35	0.78	**0.09±0.59**	0.70	0.11±0.3	**0.79**
IonoSphere	0.14±0.27	**0.94**	0.18±0.35	0.83	0.35±0.11	0.88	0.29±0.08	0.91	**0.11±0.33**	**0.94**	0.12±0.5	0.90
Kr-vs-Kp	0.37±0.37	0.95	0.37±0.37	**0.96**	**0.32±0.44**	0.83	0.50±0.11	0.87	0.39±0.22	**0.96**	0.35±0.24	**0.96**
Labor	**0.11±0.37**	**0.86**	0.18±0.17	0.77	0.27±0.54	0.70	0.33±0.13	**0.86**	**0.11±0.67**	0.85	0.12±0.17	0.79
Musk	0.19±0.21	0.79	0.22±1.67	0.65	**0.17±0.30**	0.83	0.46±0.16	**0.84**	0.22±1.67	0.65	0.20±1.4	0.64
Pima diabetes	**0.11±0.41**	**0.77**	0.18±0.90	0.68	0.57±0.67	0.75	0.78±0.80	**0.77**	0.55±0.6	**0.77**	0.18±0.33	0.68
Sick	0.05±0.45	**0.97**	0.07±0.29	0.93	**0.04±0.40**	0.94	0.06±0.50	**0.97**	0.08±0.2	0.96	**0.04±0.8**	**0.97**
Sonar	0.63±0.45	0.66	0.18±0.61	0.68	0.63±0.45	0.66	0.68±0.44	**0.74**	**0.11±0.61**	0.68	0.2±0.6	0.66
Spambase	**0.18±0.22**	**0.93**	0.21±0.58	0.77	0.21±0.34	0.91	0.20±0.43	0.92	0.21±0.55	0.77	0.20±0.59	0.89
SPECT Heart	0.50±1.40	0.64	**0.40±2.4**	0.64	0.51±1.40	0.64	0.80±0.30	**0.68**	0.51±1.39	**0.68**	0.43±2.4	0.62
Supermarket	0.55±3.41	**0.64**	0.90±2.17	0.64	0.68±3.22	0.63	1.46±5.66	0.63	**0.50±3.41**	**0.64**	0.90±2.11	0.64
Tic-Tac-Toe	0.10±0.98	0.69	**0.1±1.01**	0.61	0.33±1.00	0.60	1.20±1.90	**0.74**	**0.1±1.01**	0.66	**0.1±0.9**	0.66
Vote	**0.05±0.12**	**0.96**	0.14±0.17	0.95	0.15±0.33	**0.96**	0.15±0.15	0.93	0.07±0.17	0.95	0.1±0.17	0.95
Weather	0.10±0.20	**0.60**	0.18±0.52	**0.60**	0.18±0.52	**0.60**	0.60±0.33	**0.60**	**0.09±0.28**	**0.60**	0.12±0.51	**0.60**
Wisconsin Cancer	**0.10±0.81**	0.96	0.14±1.11	0.97	**0.10±0.81**	**0.98**	0.23±0.7	0.97	**0.10±0.81**	**0.98**	0.13±0.9	0.97
Average	**0.23**	0.78	0.27	0.74	0.33	0.75	0.57	**0.79**	**0.23**	0.77	**0.23**	0.76

TABLE 3. Precision Recall and F Score.

Dataset	Precision						Recall						F SCORE					
	EBNO	ICET	MC. BN	BN	AB. BN	XGB. DT	EBNO	ICET	MC. BN	BN	AB. BN	XGB. DT	EBNO	ICET	MC. BN	BN	AB. BN	XGB. BN
Adult	0.67	0.56	0.58	**0.71**	0.67	0.59	0.67	**0.83**	0.66	0.65	0.67	0.66	**0.67**	**0.67**	0.62	0.68	**0.67**	0.62
Australian Credit	0.78	0.78	0.79	**0.84**	0.78	0.79	0.94	0.91	0.87	0.86	**0.95**	0.87	0.85	0.84	0.83	0.85	**0.86**	0.83
Bank	0.67	0.61	0.68	**0.81**	0.67	0.61	**0.86**	0.71	0.84	0.61	0.85	0.71	**0.75**	0.66	**0.75**	0.70	**0.75**	0.66
Breast Cancer	**0.61**	0.56	**0.61**	0.60	0.61	0.60	0.59	**0.66**	0.57	0.34	0.58	**0.66**	0.60	0.60	0.59	0.44	0.59	**0.63**
Bupa liver disorder	0.54	0.56	0.53	0.54	**0.63**	0.54	0.59	0.51	0.49	0.35	**0.60**	0.51	0.56	0.54	0.51	0.43	**0.61**	0.52
Cleveland disease	0.61	0.60	0.62	**0.96**	0.61	0.60	0.92	0.75	0.92	0.72	**0.94**	0.89	0.73	0.67	0.74	**0.83**	0.74	0.72
Crax	0.87	0.82	0.78	**0.89**	0.82	0.86	**0.81**	0.79	0.79	0.77	0.79	0.78	**0.84**	0.81	0.79	0.83	0.80	0.82
Cylinder Band	0.77	0.70	0.73	**0.81**	0.77	0.70	0.62	0.60	**0.64**	0.50	0.62	0.60	**0.69**	0.65	0.68	0.62	**0.69**	0.65
German credit	0.54	0.40	0.55	**0.57**	0.54	**0.57**	0.68	0.53	**0.71**	0.51	0.68	0.70	0.60	0.46	0.62	0.54	0.60	**0.63**
Gymexamg	0.42	0.33	0.25	**0.50**	0.32	0.44	**0.29**	0.25	0.15	0.07	0.28	0.27	**0.34**	0.28	0.19	0.12	0.30	0.33
Haberman	0.47	**0.50**	0.41	0.29	0.46	**0.50**	0.43	**0.48**	0.43	0.10	0.47	**0.48**	0.45	**0.49**	0.42	0.14	0.46	**0.49**
Hepatitis	0.73	0.50	0.70	**0.86**	0.73	0.54	**0.89**	0.67	0.78	0.67	**0.89**	0.71	**0.80**	0.57	0.74	0.75	**0.80**	0.61
Horse Colic	**0.49**	0.48	0.47	0.46	**0.49**	0.46	**0.68**	**0.68**	0.58	0.52	**0.68**	**0.68**	**0.57**	0.56	0.52	0.48	**0.57**	0.55
Horse	0.71	0.74	0.59	**0.75**	0.71	0.71	0.71	0.66	0.66	0.60	**0.75**	0.70	0.71	0.70	0.62	0.67	**0.73**	0.70
IonoSphere	0.89	0.70	0.78	0.86	**0.90**	0.86	0.97	0.94	0.94	0.91	**0.99**	0.97	0.93	0.81	0.85	0.88	**0.94**	0.91
kr-vs-kp	0.95	**0.96**	0.81	0.88	0.95	**0.96**	**0.95**	**0.95**	0.85	0.84	**0.95**	**0.95**	0.95	**0.96**	0.83	0.86	0.95	0.95
Labor	0.76	0.62	0.55	**0.91**	0.74	0.70	**0.87**	**0.87**	0.80	0.67	**0.87**	0.86	**0.81**	0.72	0.65	0.77	0.80	0.77
Musk	0.71	0.57	0.74	**0.84**	0.57	0.55	0.88	0.81	**0.92**	0.79	0.83	0.81	0.79	0.67	**0.82**	0.81	0.68	0.66
Pima_diabetes	0.64	0.53	0.61	**0.68**	0.63	0.53	0.75	0.69	**0.76**	0.63	**0.76**	0.69	**0.69**	0.60	0.68	0.65	**0.69**	0.60
Sick	**0.73**	0.48	0.53	0.72	0.72	0.70	0.81	0.88	0.86	0.84	0.79	**0.89**	0.77	0.62	0.65	0.77	0.75	**0.78**
Sonar	0.61	0.63	0.61	**0.79**	0.64	0.62	0.76	0.80	0.76	0.60	**0.81**	0.78	0.68	0.70	0.68	0.68	**0.72**	0.69
Spambase	0.90	0.66	0.91	**0.95**	0.68	0.72	**0.93**	0.84	0.87	0.83	0.86	0.78	**0.92**	0.74	0.89	0.88	0.76	0.75
SPECT Heart	0.54	0.53	0.54	**0.62**	0.62	0.54	0.79	**0.82**	0.75	0.57	0.80	0.81	0.64	0.65	0.63	0.59	**0.70**	0.65
Supermarket	**0.58**	0.51	0.46	0.00	**0.58**	0.51	0.07	0.05	0.07	0.00	**0.08**	0.06	0.13	0.09	0.12	0.00	**0.14**	0.11
Tic-Tac-Toe	0.55	0.47	0.46	**0.73**	0.52	0.54	**0.78**	0.74	0.73	0.41	0.77	0.74	**0.64**	0.57	0.56	0.53	0.62	0.62
Vote	0.95	0.93	**0.98**	0.91	0.94	0.93	**0.95**	**0.95**	0.93	0.91	0.94	**0.95**	**0.95**	0.94	**0.95**	0.91	0.94	0.94
Weather	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	**1.00**	**1.00**	**1.00**	0.50	**1.00**	**1.00**	**0.67**	**0.67**	**0.67**	0.50	**0.67**	0.67
Wisconsin Cancer	0.90	0.92	0.95	0.97	**0.98**	0.92	**1.00**	0.98	0.98	0.93	1.00	0.98	0.95	0.95	0.97	0.95	**0.99**	0.95
Average	0.68	0.61	0.63	**0.71**	0.67	0.65	**0.76**	0.73	0.73	0.60	**0.76**	0.73	**0.70**	0.65	0.66	0.64	**0.70**	0.67