Comprehensible Classification Models – a position paper

Alex A. Freitas
School of Computing
University of Kent
Canterbury, CT2 7NF, UK
A.A.Freitas@kent.ac.uk

ABSTRACT

The vast majority of the literature evaluates the performance of classification models using only the criterion of predictive accuracy. This paper reviews the case for considering also the comprehensibility (interpretability) of classification models, and discusses the interpretability of five types of classification models, namely decision trees, classification rules, decision tables, nearest neighbors and Bayesian network classifiers. We discuss both interpretability issues which are specific to each of those model types and more generic interpretability issues, namely the drawbacks of using model size as the only criterion to evaluate the comprehensibility of a model, and the use of monotonicity constraints to improve the comprehensibility and acceptance of classification models by users.

Keywords

Decision tree, rule induction, decision table, nearest neighbors, Bayesian network classifiers, monotonicity constraint.

1. INTRODUCTION

The vast majority of works on classification model evaluation use predictive accuracy as the only evaluation criterion [34], [36], [64]. However, in real-world applications where the goal is to produce a classification model that is useful to the user, the comprehensibility (interpretability) of the model to the user is also important – see Section 2. Model interpretability was stressed in early machine learning research [57]; but in the last two decades, popular classification methods like ensembles [60], support vector machines (SVM) and kernel-based learning methods [13] have been designed to maximize predictive accuracy only. Indeed, the models produced by SVMs are in general black box models that can be hardly interpreted by users; whilst an ensemble of models tends to be harder to be interpreted by users, by comparison with a single classification model [23], [26], [69].

Despite the dominance of predictive accuracy as the evaluation criterion, there has been significant progress in methods proposed for improving the comprehensibility of classification models [6], [21], [47], [49], [53], [55], [70]. In addition, the importance of comprehensible classification models continues to be emphasized in many application domains, like medicine [5], [46], [55], [59], [76]; credit scoring [4], [35]; churn prediction [47], [70]; and bioinformatics [26], [67]. However, most of these works focus on improving the comprehensibility of just one type of classification model representation (e.g. decision trees or classification rules).

1.1 Goals and Scope of the Paper

This paper has two broad goals: (a) discussing the pros and cons of five types of classification knowledge representation – decision trees, classification rules, decision tables, nearest neighbors, and Bayesian network classifiers, with respect to their interpretability; and (b) discussing how to improve the comprehensibility of classification models in general.

We focus on the comprehensibility of classification models, rather than on the trade-off between predictive accuracy and comprehensibility. Although that trade-off is important, it is out of the scope of this paper since it is extensively discussed in the literature – see e.g. works on multi-objective optimization concepts such as Pareto dominance [25], [38] and lexicographic optimization [25], [42]; the minimum description length principle [30]; and Occam's razor [17], [74]. It is important to note, though, that nearly all works on the accuracy-comprehensibility trade-off measure comprehensibility in terms of model size, an over-simplistic notion of comprehensibility, as discussed in Subsection 4.1. In contrast, this paper discusses comprehensibility in a broader sense, involving several issues related to the interpretability of a model, as discussed in Sections 3 and 4.

Another topic out of the scope of this paper is the extraction of comprehensible classification models (e.g. rule sets or decision trees) from the "black box" models produced by methods such as SVMs and artificial neural networks (ANNs) or from the complex models produced by ensembles of classifiers. This topic is left out of the scope of this paper for two reasons. First, it is already extensively discussed in the literature – see e.g. [8], [39], [3], [28], [14], [69]. Second, the concepts and methods discussed in this paper are applicable to classification models expressed in the aforementioned five types of knowledge representation in general, regardless of whether the model was induced directly from the data (as usual) or extracted from a black box model. This is because this paper focuses on classification *models* (the outputs of classification algorithms), rather than on classification *algorithms*.

1.2 Structure of the Remainder of the Paper

Section 2 reviews the case for discovering comprehensible classification models. Section 3 discuss issues in the interpretation of each of the above five types of classification knowledge representations, and reviews works comparing the interpretability of different types of representations. Section 4 focuses on more generic issues of classification model interpretability: it first discusses the limitations of model size as a single criterion for measuring a model's comprehensibility, and then discusses the use of semantic monotonicity constraints. Section 5 concludes the paper with a summary and discussion of its main points.

2. THE CASE FOR COMPREHENSIBLE CLASSIFICATION MODELS

The importance of comprehensible classification models stems from several issues. First, understanding a computer-induced model is often a prerequisite for users to trust the model's predictions and follow the recommendations associated with those predictions. The need for trusting computational predictions tends

to be particularly strong in medical applications [19], [39], [46], [59], [69], where lives are at stake. Model comprehensibility is also important for the model's acceptance by users in financial applications [15] and in customer churn prediction applications [70]. In bioinformatics, understanding a model's predictions often helps biologists to trust the predictions [16], [26], [67], increasing the chances that users will invest a lot of time and money in the execution of sophisticated biological experiments to try to confirm the model's predictions. Arguably, the ultimate value of a model's predictions for bioinformatics is determined by the cumulative success of the experiments inspired by those predictions [37].

The need for comprehensible models in order to improve the user's trust on the model is also strengthened when the system produces an unexpected model to the user, in which case the user requires good explanations from the system as a requirement for model acceptance [46]. A real-world example of this point (not involving data mining, but relevant to our discussion) is given in [33]. Shortly before a major accident happened in the Three-Mile Island nuclear power plant, the plant's automated system recommended that a human operator shutdown some system, but the operator did not implement the shutdown because she/he did not believe in the system's recommendation.

In addition, in some application domains users need to understand the system's recommendations enough to legally explain the reason for their decisions to other people. For instance, according to the Danish Public Administration Act, an administrative decision is required to be accompanied with its main reasons [40]. In the medical domain, if a doctor makes a decision (say, recommends surgery) based on the prediction of a classification model and that leads to major harm to the patient, the doctor should understand the reason for the model's predictions in order to defend her/his decisions in court if she/he is sued for medical negligence [59]. Legal requirements are also common in credit scoring applications, where a bank often has the legal obligation of explaining why a customer was denied credit [35], [49].

Furthermore, comprehensible classification models can give new insights to users about important predictive relationships in the data, i.e., identifying which attributes are the strongest predictors of the class variable. In some scientific application domains, the analysis of a comprehensible classification model can even lead to the formulation of new theories or hypothesis about the target problem – see e.g. [26], [41] for examples in bioinformatics; and [59], [73] for examples in medical applications.

Of course, model comprehensibility is not always important, and in some applications users might be happy in accepting the predictions of a model based only on its high predictive accuracy, regardless of its comprehensibility. The relative importance of comprehensibility and predictive accuracy is a subjective issue, which depends on the user's interests and the application domain.

In cases where users are happy to accept the predictions of a non-comprehensible model based only on its predictive accuracy, one caveat should be mentioned, though. Although the classes of test set instances are *unknown by the algorithm*, the test instances' classes are *known by the user* – i.e., both training and test sets contain data about the "past". The predictions that really matter to the user are the predictions to be made for "future" data, whose classes are *unknown by the user* at the time the classification algorithm is run. We use predictive accuracy in the test set as an *estimate* of the predictive accuracy in future data, but that estimate may not be reliable if the future dataset has a probability distribution significantly different from the past data. This involves the "datashift" problem [58]. Understanding a very accurate classification model might give us more confidence that the model is really capturing the correct patterns in the target domain (rather than patterns valid in past data that would not be valid in "future" data), helping us to cope with the datashift problem. It should be noted, though, that research on methods to cope with the datashift problem typically focus only on predictive accuracy, unfortunately ignoring model comprehensibility issues.

We conclude this section with a story about the effectiveness of an ANN in a military application. It is not clear if this story is based on a real or hypothetical application of ANNs, but it is related to the datashift problem. In this story, the military trained an ANN to classify images of tanks into enemy and friendly tanks. The predictive accuracy of the ANN in in the test set images was very high. However, when the ANN was deployed in the field (corresponding to "future data"), it had a poor accuracy rate. Later, users noted that all photos of friendly (enemy) tanks were taken on a sunny (overcast) day. I.e., the ANN learned to discriminate between the colors of the sky in sunny vs. overcast days! If the ANN had output a comprehensible model (explaining that it was discriminating between colors at the top of the images), such a trivial mistake would immediately be noted.

3. INTERPRETING SPECIFIC TYPES OF CLASSIFICATION MODELS

3.1 Interpreting Decision Trees

The comprehensibility of decision trees is facilitated by several factors [56], [61]. First, a decision tree has a graphical structure. Second, a decision tree typically contains a subset of (rather than all) attributes, helping users to focus their analysis on the most relevant attributes. Third, the hierarchical tree structure provides information about the relative importance of different attributes: broadly speaking, the smaller the depth (the closer to the root) of an attribute, the more relevant the attribute is for classification. Note that an attribute might occur two or more times in the same path from the root to a leaf – due, e.g., to multiple binary partitions on nodes with numerical attributes. In this case we consider the attribute's depth as the shallowest occurrence of the attribute in that path. In the case of multiple occurrences of an attribute in paths with non-overlapping sets of edges, an aggregated measure of the depths of those occurrences (e.g. the mean depth) would need to be computed.

The criterion of assigning greater relevance to attributes with smaller depth has a caveat, however; even in the simplest case of comparing the relevance of two attributes which occur just once in the tree. An attribute *A* might have smaller depth than another attribute *B*, yet *B* might be more relevant in the sense of being used to classify more instances. This suggests that a more intuitive criterion for determining the relevance of an attribute in a decision tree is the number of instances whose classification used a value of that attribute – i.e. the summation of the number of instances assigned to any leaf node whose associated path (from the root to that leaf) includes the attribute in question. This criterion takes into account attributes occurring in multiple paths in the tree, and the measure of relevance (total number of instances classified by that attribute) has a very clear interpretation by users.

Another caveat in decision tree interpretation is that some subtree(s) of the constructed tree can contain irrelevant attributes, even when the data is not noisy. This is because the decision tree structure imposes the rigid requirement that once an attribute is selected to label a tree node, each value of that attribute has to be included in the tree. Hence, some branch(es) may have to be added to the tree just to preserve the structure of a tree, even if that branch is associated with an irrelevant attribute value [9], [10], [24], [29]. Such irrelevant values mislead the user's interpretation of the tree and may lead to overfitting.

For example, suppose a disease has the following patterns: (a) patients who are old and smoke are very likely to get the disease; (b) a positive blood test result is a reasonably good predictor that a patient will get a disease regardless of age and smoking status. In this scenario, it is plausible that a decision-tree induction algorithm would create the tree shown in Figure 1. The leftmost path in the tree captures the above pattern (a). That pattern refers only to the *old* value of age; the values *middle-age* and *young* had to be put as extra branches in the tree just to preserve the tree structure, even though the latter two values are irrelevant. As a result, the information about the above pattern (b) had to be replicated in two different subtrees of the tree shown in Figure 1.

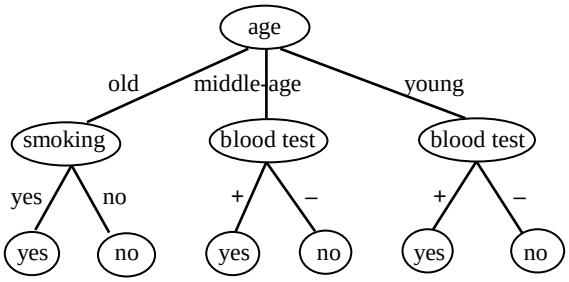

Figure 1: Example of a decision tree illustrating the problem of irrelevant branches and a replicated subtree.

In addition to mislead the user's interpretation of the tree, such replication is associated with the data fragmentation problem, since the instances supporting pattern (b) will be divided between the two subtrees, reducing the number of instances available for attribute selection at each just-created node. Note that in general tree pruning methods do not help to avoid this problem, since the only way to remove the *irrelevant* values *middle-age* and *young* from the tree would be to remove the *age* attribute, but for this we would need to remove the *relevant old* value too.

3.2 Interpreting Classification Rules

Classification rules – of the form: IF (conditions) THEN (class) – are a logics-based representation, like decision trees – since a path from the root to a leaf in a decision tree is equivalent to an IF-THEN rule. However, there are important differences in the interpretation of rules and decision trees.

First, classification rules have a textual representation, rather than the graphical representation of decision trees. The "flat" textual representation gives no direct clue about which attributes are more important than others in a rule, unlike the hierarchical positional information of attributes in a tree. Some positional information related to the relative importance of attributes could be added to rules, by showing rule conditions in the order in which they were added to a rule, in decreasing order of relevance as estimated by a rule induction algorithm. However, this would still leave us with the problem of how to estimate the relevance of an attribute in the entire classification model (with all rules). One solution is the same type of attribute relevance measure proposed for decision trees in Subsection 3.1, I.e., the relevance of an attribute can be measured by the summation of the number of instances classified by any rule whose antecedent includes the attribute in question.

Although the textual rule representation makes it difficult for users to get the "full picture" about the model, it allows users to analyze individual rules as modular pieces of knowledge ("local patterns"), one at a time. It is possible for users to focus on such modular/local patterns in a decision tree, by analyzing separately each path from the root to a leaf node; but it has been argued that converting a decision tree into a set of rules and analyzing the rules separately often improves a model's comprehensibility [56]. Note that after converting a decision tree into a set of rules, one can prune those rules in a way more flexible than the pruning of decision trees. Any condition can be removed from a rule, regardless of the order in which conditions were added to the rule. By contrast, in decision trees, a typical pruning method removes deep attribute tests more easily than removing shallow ones.

Another important difference between decision tree and rule representations is that in a tree the set of leaves represent mutually exclusive and exhaustive class predictions, i.e., each instance in the test set is classified by exactly one leaf node. This is not the case with rules, where some instances may satisfy the antecedent of several rules predicting conflicting classes for that instance. This characteristic of rule sets has the disadvantage of requiring an extra method for resolving conflicting class predictions [9], [12], [51], [72] – which is not needed in a decision tree. Many rule induction systems cope with this problem by outputting an *ordered* rule *list* (decision list), rather than an *unordered* rule *set*. In such lists, a test instance is given to each rule in the list in turn, until the instance's attribute values match a rule's antecedent. Then the instance is assigned the class predicted by that rule, ignoring the other rules in the list. However, an ordered rule list makes it harder for users to interpret rules which are not at the start of the list, since such rules only make sense in the context of all previous rules in the list [12], [51], [72].

Although the simplicity of a rule list is usually measured by the total number of conditions in all rules, this is not a fair measure of simplicity since different test instances require different numbers of rules to be evaluated in the ordered rule list. A fairer measure of an ordered rule list's simplicity is the average number of rule conditions that were evaluated in the rule list in order to classify an instance, where the average is computed over all test instances [51]. This is a measure of the "prediction-explanation size" of a rule list, rather than a measure of the rule list's size, considering that the rule conditions evaluated in order to classify an instance are an explanation of that class prediction to the user.

The fact that rules do not have to make mutually exclusive and exhaustive predictions facilitates the representation of models containing only relevant conditions, by comparison with the less flexible decision tree representation – as discussed earlier. Re-visiting the example of a tree with irrelevant branches in Figure 1, the below rule set represents the correct patterns in that example:

IF (*age = old*) AND (*smoking = yes*) THEN (*disease = yes*)
IF (*blood test = positive*) THEN (*disease = yes*)
IF none of the above rules is satisfied THEN (*disease = no*)

This rule set is similar to the tree in Figure 1, but the rule set has only 3 rules, rather than the 6 paths from the root to a leaf in that tree – although the rule set requires a method for resolving conflicting class predictions by different rules, as mentioned earlier. Note that this rule set does not have any replicated information, since the positive *blood test* pattern appears only once in the rule set (whilst it appears twice in the tree), and the rules do not represent irrelevant conditions (whilst the tree has irrelevant branches referring to the *middle-age* and *young* values of *age*). An explanation is that decision tree induction algorithms typically select one *attribute* at a time when expanding the tree, whilst rule induction algorithms typically select one *attribute-value* at a time when expanding a classification rule [24]. Of course, selecting an entire attribute would be more appropriate if all the attribute's values were relevant in the target dataset.

Finally, it should be noted that, when interpreting a discovered rule set (or ordered rule list), it is often worth to analyze not only the rules themselves, but instances which are exceptions of the rules – i.e. instances that satisfy a rule antecedent but have a class different from the class predicted by the rule. Although this kind of rule exception analysis is rarely performed, it can lead to further insight about the application domain– see e.g. [52].

3.3 Interpreting Decision Tables

Decision tables are a tabular knowledge representation where the state of a set of conditions jointly determines one or more outcomes [47]. When used as classification models, the conditions are attribute-value conditions and each set of conditions (table cell) is associated with a class prediction. We focus on single-hit tables, with mutually exclusive rules (cells), since they do not have redundancy and facilitate the interpretation of decision tables by users [48]. To classify a new instance, the table cell matching that instance is found, and then the instance is assigned the most frequent class among all instances matching that table cell. If there are no instances matching that cell, it is common to assign to the new instance the most frequent class in the training set, but other approaches can be used [44].

Decision tables can be induced from data in different ways. For instance, they can be induced by performing attribute selection with a wrapper [43] or a (faster) filter approach [44]. In [47] a decision table is extracted from an induced decision tree; whilst in [4] first rules were extracted from an ANN, and then decision tables were constructed from those rules with a semi-automated approach. Note that generating decision tables from a rule set may lead to a very large number of columns in the table, requiring a table contraction method to produce a table with near-optimal condition order and size – i.e., the smallest possible table size that is logically equivalent to the rule set where the table was derived from. In [4] exhaustive search was used, but much larger rule sets would require search methods like branch-and-bound or heuristic search. In experiments reported in [44], although decision tables used much fewer attributes than C4.5 in most datasets, in many cases the tables still had hundreds or thousands of cells.

3.4 Interpreting Nearest Neighbors

Although nearest neighbor algorithms in general do not discover an explicit, abstract classification model, they still provide an explanation for the classification of each new instance. That explanation consists of the attribute values in the nearest neighbors, which are used to predict the class of the new instance.

There are two caveats, though. First, the details of the explanation (the attribute values of the nearest training instances) are normally different for each new test instance being classified. This is in contrast with decision trees and classification rules, where a single path to a leaf node or a single rule can provide an explanation for the classification of a (potentially large) set of instances.

Secondly, the conventional explanation for each test instance's classification consists of the set of values of all attributes in the data. In datasets with a very large number of attributes, the concept of "nearest neighbors" seems intuitively unsatisfactory as an explanation for the classification of a new instance, because even the nearest training instance will probably differ from the new one being classified in a very large number of attributes [49].

Two approaches for improving the interpretability of nearest neighbors are as follows. First, instead of using the entire training set in the distance computations to find the nearest neighbors, the system can produce a small set of prototypes (typical instances), and then search for nearest neighbors computing the distance between a test instance and the prototypes only. The test instance is then assigned the most frequent class among its nearest prototype(s). In this case the set of prototypes represent an abstract, summarized form of knowledge which can be directly shown to users, analogous to an abstract model like a decision tree or rule set. Several methods for selecting prototypes have been proposed [63], [66], [75]. However, usually prototypes have been selected with the purpose of increasing predictive accuracy, rather than improving interpretability as suggested here.

A second approach to improve the comprehensibility of nearest neighbor predictions involves computing attribute weights [71], where the weight of an attribute is proportional to its predictive power. This is a natural approach to try to improve predictive accuracy anyway, because an unweighted-attribute nearest neighbor algorithm uses the strong simplifying assumption that all attributes are equally relevant for predicting the class, which is unlikely to be the case in most applications. A set of attribute weights could be considered a comprehensible form of knowledge by itself, helping to identify the most relevant attributes. In addition, attribute weights can improve the comprehensibility of explanations for the classifications of test instances, as follows. When the user asks to see the attribute values in the nearest neighbors for a given test instance being classified (as an explanation for that classification), the system could show the attributes in decreasing order of weight (relevance). Hence, users could focus their attention on analyzing the neighbors' attribute values for the most relevant attributes only.

3.5 Interpreting Bayesian Network Classifiers

Unlike the interpretation of decision trees, which contain a subset of relevant attributes, interpreting the model produced by Naïve Bayes requires the user to analyze the probabilities associated with all attributes. However, in some cases users might prefer to interpret the probabilities computed by Naïve Bayes for all attributes, rather than focusing on just the set of attributes in a decision tree. This can occur e.g. in medical applications [45], [46], [76]. To cite from [46]: "*One of the main advantages of this* [Naïve Bayes] *approach, which is appealing to physicians, is that all the available information is used to explain the decision; such an explanation seems to be 'natural' for medical diagnosis and prognosis.*" Note, however, that in these cases the medical doctors' preference for the use of all original attributes seems

biased by the fact that the original attributes had been carefully selected by experts, which is not the case in many real-world data mining applications.

Furthermore, in some applications users might prefer the natural probabilistic interpretation of naïve Bayes, rather than the Boolean logic-based nature of classifiers like decision trees, classification or decision tables. E.g., it has been argued that modeling probabilities is very important in clinical practice [5].

However, Naïve Bayes makes the assumption that attributes are independent from each other given the class. This assumption reduces the number of conditional probabilities to be interpreted by the user, but it misses the opportunity of detecting interesting correlations among attributes. Although that assumption is usually unrealistic, it might be realistic in some cases where predictor attributes are created by medical doctors, who tend to "think linearly" and create independent attributes [45], [50].

More sophisticated types of Bayesian Network Classifiers (BNCs) do not have the Naïve Bayes' strong assumption of attribute independence; they allow the modeling of attribute dependencies, represented by edges between attributes (nodes) in a network [27], [11]. There are two broad approaches to construct a BNC. In the class-constrained approach, the class attribute is given a special treatment, being inserted at the root of the network, with edges pointing from it to other nodes in the network. In the unconstrained approach, no special treatment is given to the class attribute, which can be inserted anywhere in the network.

An example of a class-constrained BNC method is the Bayesian Network-Augmented Naïve Bayes (BAN) illustrated in Figure 2. In a BAN there are two types of edges: (a) an edge from class C to each predictor attribute A_i, $i = 1,\ldots,N$ (where N = number of predictor attributes) – these edges are also used by Naïve Bayes; and (b) edges representing dependencies among attributes – these edges are not used by Naïve Bayes. In order to avoid overfitting or to improve the simplicity of the model, a BAN-building algorithm often has a user-defined parameter K specifying the maximum number of parents for a node, not counting the class node (which is a parent of every other node). If K is set to 1 we have a Tree Augmented Naïve Bayes (TAN) [27].

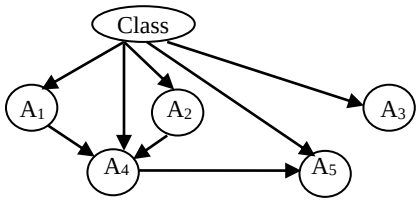

Figure 2: Bayesian network-Augmented Naïve Bayes (BAN)

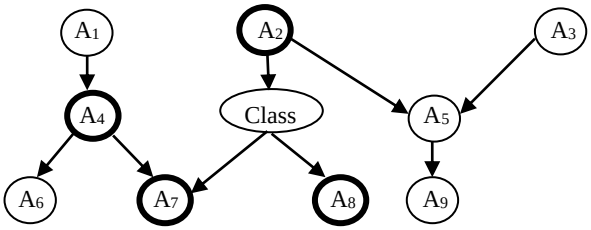

Figure 3: General Bayesian Network (GBN), with the Class' Markov blanket shown in boldface

An example of the unconstrained approach for building a BNC is the General Bayesian Network (GBN) classifier, illustrated in Figure 3, where the class attribute has both incoming and outgoing edges. In order to classify test instances with a GBN, we only need to use the Markov blanket of the class node, which is the set of attributes which the class variable depends on [11], and consists of set of nodes that are parents, children or parents of the children of the class node in the GBN. In Figure 3 the set of nodes in the Markov blanket of the class node is shown in boldface.

The comprehensibility of a BNC depends partially on its graphical model's size, since a BNC with a very large number of edges would be difficult to interpret. This is particularly the case for BANs. In the case of GBNs, it is the size of the Markov blanket of the class node that matters for interpreting the model. The Markov blanket of a class node tends to be much smaller than a full BAN network. Actually, the Markov blanket of the class node can be considered a form of discovered knowledge by itself, analogous to the knowledge represented by a subset of relevant attributes selected by an attribute selection method.

However, size by itself is not enough to evaluate the comprehensibility of a BNC. Even relatively small BNCs might be somewhat confusing to users, unless they have the proper training to interpret such classifiers. In particular, the use of directed edges to represent dependencies can be confusing to users, since users usually think of probabilistic dependencies as undirected (or "bidirectional") dependencies. Note also that most statistical tests of attribute dependence are symmetric with respect to the attributes, i.e., for a given pair of attributes A, B and a dependence measure Dep, usually we have $Dep(A,B) = Dep(B,A)$. Heckerman et al. [32] give an example of the situation where users get confused by unidirectional edges in Bayesian networks, based on their experience of showing Bayesian networks to users, as follows. Consider a Bayesian network where attributes *gender* and *age* are parents of the attribute *income*. Users understand that *gender* and *age* are predictors of *income*, but they also think that *income* is a predictor of *gender* and *age* and wonder why there are no edges pointing from *income* to those two attributes.

To avoid this and other problems in the interpretation of Bayesian networks, Heckerman et al. [32] proposed the use of dependency networks, which represent bidirectional dependencies among attributes by using a cyclic directed graph (where a set of directed edges can form a cycle). The use of dependence networks in classification is relatively rare, and this type of model seems worth more attention and research than it has received so far.

3.6 User-based Experiments Comparing the Comprehensibility of Different Types of Classification Model

There have been few experiments directly comparing the *subjective* comprehensibility of different types of classification models *to the user*, as summarized next.

In [35], an experiment measured the accuracy, response time and confidence with which a user answered questions about classification models expressed in different representations, namely: single-hit decision tables, binary decision trees, rule lists with univariate conditions and oblique rules (with multivariate conditions). The experiments involved only the credit scoring domain, and the models were induced only for the two-class problem of accepting or rejecting a credit application. The

experiment involved 51 non-expert users with no prior experience with any of the knowledge representations or credit scoring. Most users found decision tables the most comprehensible, easiest to use type of representation. This was attributed mainly to the physical conciseness of the decision table format, and also partly attributed to the fact that in each column of a decision table the attributes are considered in the same order [35]. The latter is in contrast with a decision tree, where different paths from the root to a leaf consist of attributes in different orders, which seems to slow down users when they had to do a fast search within the tree in order to answer classification questions. In addition, most users found oblique rules very difficult to use and the least comprehensible representation. Decision trees with oblique splits were also considered less comprehensible than decision trees with univariate splits in a bioinformatics problem [31].

In [1], 100 non-expert users were asked to compare the understandability of decision trees and rule lists induced from two small datasets from the UCI ML repository – namely, Contact Lenses and Labor. Decision trees were in general deemed by users to be more understandable than rule lists for both datasets.

A different type of experiment, not directly related to data mining, compared decision trees and tables in the context of a computer game where users had to interpret trees and tables to make investment decisions that maximized their profit in the game [65]. Among a group of 67 non-expert users, decision trees were overall found to be more comprehensible than decision tables. The greater comprehensibility of decision trees was attributed to their ability in graphically revealing the patterns in the data and the ease with which users can follow a tree path until its leaf node.

Such experiments comparing the users' preferences for different types of knowledge representation are useful and should be done more often, but note that the results of such experiments are naturally biased by the background of the users [65], [35]. For instance, some users might be familiar with sophisticated mathematical equations and find them more comprehensible than graphical or linguistic models like decision trees or rules. E.g., in an application in the Earth Sciences [62], where users were familiar with differential equations, the authors suggested data mining should produce that kind of model, to facilitate users' acceptance of the model.

Overall, however, it is usually agreed that representations like decision trees, classification rules or decision tables tend to be more comprehensible to most users than mathematical equations (like linear combinations of attributes in oblique decision trees) and non-linear models like ANN or SVM [26], [49], [70].

4. INTERPRETING CLASSIFICATION MODELS: GENERIC ISSUES

4.1 The Drawbacks of Model Size as the Single Measure of Comprehensibility

In the vast majority of papers where the comprehensibility of a classification model is evaluated, that evaluation is done in an over-simplistic way, by measuring only the size of the model – e.g. the number of nodes in a decision tree, or the number of edges in a Bayesian network classifier. The assumption is that the smaller the model is, the more comprehensible it would be to the user. However, there are several problems with that assumption.

First of all, the size of a model is just a syntactical aspect of that model; it does not capture any aspect of its semantics. Clearly, the comprehensibility of a model depends strongly (and subjectively) on the actual "contents" of the model, i.e. the attributes in the decision tree, the attribute-value conditions in classification rules, the parent attributes of each attribute in a Bayesian network classifier, etc.. It is quite possible that a larger decision tree be more comprehensible to the user than a short one, because the larger tree uses attributes which make more sense to the user.

For example, in [46] medical experts preferred *larger* trees over shorter ones, because the shorter trees contained fewer informative attributes and described the patient too poorly for supporting the medical doctor's decisions. In [1], in experiments with 100 non-expert users who subjectively evaluated the understandability of decision trees and rule lists induced from the UCI Labor dataset, in general the larger a model was, the more *understandable* it was considered by the users. This was attributed possibly to larger models providing more classification-relevant information to the user, and also to the fact that the small size of the Labor dataset did not lead to very large models.

In addition, extreme simplicity is not acceptable for users. For instance, a trivial decision tree with a single leaf node predicting the most frequent class in the training set for all instances in the test set can hardly be accepted by users. Also, a model can be non-trivial and still be considered too simple to be accepted by users. An example of this point in the medical domain is given in [20], where the only internal node of the tree is the attribute *Fever*. Medical doctors cannot accept such an over-simplistic explanation. To cite from [20]: "*Humans by nature are mentally opposed to too simplistic representations of complex relations*."

Intuitively, the size of a model can have a significant effect on its comprehensibility, particularly when the model is "too large", since in such case the user could not even have time to analyze the full model. One caveat, though, is that maybe the user could make time to analyze part of the model (e.g. a rule subset or a subtree of a decision tree) and still find that part comprehensible or useful. In addition, the definition of "too large" varies hugely among users. For instance, in [62] a rule induction system discovered 41 rules, and that number was considered an "overwhelming" number of rules by the user. By contrast, in [68] apparently a user patiently analyzed a set of 29,050 rules – which led to the identification of a small subset of 220 (< 1%) interesting rules.

Despite the difficulty in predefining a maximum model size that would allow a user's interpretation of a model, some authors have suggested to impose strict user-defined size constraints on the classification model, in order to improve the model's comprehensibility [62], [69]. For instance, in [62] it is suggested that "*a useful feature for future machine learning algorithms would be the ability to directly specify the maximum number of rules in the model as a parameter to the learning algorithm*". Although this approach might be useful in some specific cases – when more sophisticated approaches would not be cost-effective – we would not recommend this approach in general, since it is not flexible and seems over-simplistic. We favor instead a more principled approach for coping with the accuracy-comprehensibility trade-off, such as multi-objective optimization based on Pareto dominance [25], [38] or lexicographic optimization [25], [42].

The assumption that smaller models are more comprehensible to users was also empirically tested in [35] in experiments with 51 non-expert users. Overall, the users' comprehensibility of induced models tended to increase with the models' sizes, for different types of knowledge representation – namely: decision tables, decision trees and classification rules. Another work reports the results of experiments where two expert users (neurologists) subjectively evaluated many classification rules predicting whether a patient is normal or has early signs of dementia [53]. Although the results showed that there was some correlation between a rule's size and its acceptance by users, that correlation was overall quite weak. A stronger correlation existed between the rule's acceptance by users and the number of rule conditions that violate the user's background knowledge: the smaller this number, the more likely the rule is to be accepted by users. Such violations of background knowledge will be discussed in Subsection 4.2.

4.2 Incorporating Semantic Monotonicity Constraints in Classification Models

Users are more like to trust and accept classification models when they are built by respecting monotonicity constraints provided by the users or experts in the application domain [49], [55], [70]. A monotonic relationship between a numerical predictor attribute and the class attribute occurs when increasing the value of the attribute tends to either monotonically increase or monotonically decrease the probability of an instance's membership to a class.

Monotonicity constraints seem to be quite common in real-world applications [6]. For instance, when predicting whether or not a customer will buy a product, the probability of class "*buy = yes*" tends to decrease monotonically with an increase in the product's price [21]. Also, the probability of class "*cancer = yes*" is usually expected to increase monotonically with the patient's age.

When such monotonicity constraints are provided by the user or an expert, they can be incorporated in a classification algorithm in several different ways. Let us consider first the case of decision trees, where a tree is said to be "monotone" if it respects all the predefined monotonicity constraints, i.e. if the tree has no pair of leaves leading to a violation of some monotonicity constraints for some instances classified by those two leaves.

In [6], the attribute selection criterion of a decision-tree algorithm was modified to consider a combination of entropy and a measure of the degree to which monotonicity constraints are respected. Strictly enforcing all monotonicity constraints during tree construction is difficult because monotonicity is a global property of a full tree [21]; whilst typical decision tree algorithms treat the selection of an attribute test for a given tree node as independent from the test selection in other tree paths. Hence, in [22] a standard (not necessarily monotone) tree is first built, and then the tree is pruned with a method that guarantees that the pruned tree is monotone. The monotone trees turned out to be slightly more accurate and much smaller than standard trees.

Note that strictly enforcing all monotonicity constraints, as hard constraints, is not always desirable, since there is a trade-off between maximizing accuracy and respecting monotonicity constraints [6], [7]. Considering the monotonicity constraints as soft constraints, which may be violated if this increases the model's accuracy, may be more suitable sometimes. In some cases, though, strictly enforcing monotonic constraints does not lead to loss of accuracy, and may even improve accuracy [47], [2].

So far we have discussed monotonic relationships for numerical attributes, but monotonic relationships may involve nominal attributes too. The latter case occurs when one or more of the attribute's values either increases or decreases the probability of an instance belonging to a class. In an extreme case, a nominal value indicates membership in a given class at the exclusion of other classes. An example of this case is shown in the rule below [55], where the underlined conditions are counter-intuitive to the user, since the rule predicts that the patient is *normal*.

IF *the years of education of the patient is* > 5
AND *the patient does not know the date*
AND *the patient does not know the name of a nearby street*
THEN *the patient is* "*normal*"

To avoid the discovery of such rules, in [55], if a nominal attribute takes a value that is indicative of membership in a given class at the exclusion of other classes, that attribute value is not considered when building a rule to predict those other classes. In the above example rule, "*patient does not know the date*" is indicative of membership in the class *mentally impaired*, and so it should not be used in a rule predicting class *normal*.

In [2], monotonic constraints were considered only when learning the parameters of Bayesian networks, not when learning a network structure. Moreover, an algorithm for checking if decision tables violate monotonicity constraints is presented in [47]. In addition, in [18] the training data is first relabeled to make it monotone, and then a nearest-neighbor algorithm is modified to predict class labels in a way that respects the monotonicity constraints.

So far we focused on monotonicity constrains provided by expert users, but those constraints can be automatically derived from the data – e.g., using the concept of globally predictive tests (conditions) [54]. A test T_i is globally predictive of a given class C_j iff $P(C_j|T_i) > P(C_j)$ – e.g., T_i = (*age* > 70), C_j = (*cancer = yes*). Rule induction algorithms often create rules with tests that are locally (not globally) predictive, in the local context of a rule. T_i is locally predictive of C_j in context *Ctx* iff $P(C_j|T_i,Ctx) > P(C_j|Ctx)$, where *Ctx* is a Boolean combination of tests. Using such probabilities, computed from the data, two modified versions of a rule induction algorithm were proposed in [54]. One version is allowed to add, to a rule antecedent, only tests that are globally predictive of the class in the rule consequent – a hard constraint. The other version can add to a rule antecedent a test that is not globally predictive if it is significantly better than the best globally predictive test – a soft constraint.

Automatically deriving monotonicity constraints from the data has the advantage of not requiring expert users, but the disadvantage of relying on the high quality of the data. It has been argued that, even when there are monotonic constraints in the application domain, real-world datasets usually have some non-monotonic noise, with purely monotonic datasets being rare [7]. For researchers who need benchmarking datasets for evaluating classification algorithms that respect monotonicity constraints, information on monotonic constraints for five UCI datasets is given in [2]. For a further discussion about how to incorporate monotonicity constraints in classification models, see [49].

5. SUMMARY AND DISCUSSION

Different knowledge representations have different pros and cons, in terms of their interpretability by users, and none of them can be considered the "most comprehensible" in all applications. The

choice of a knowledge representation should be made by taking into account the user's background and subjective preferences, and characteristics of the target dataset.

For instance, contrasting the interpretability of decision trees and classification rules, if the dataset contains many attributes having a single value that is relevant for class prediction, decision trees would have the drawback of including irrelevant values. This would mislead the user's interpretation of the tree and may lead to overfitting. In such datasets, a rule-based model could be more suitable, since rule induction algorithms can select a single relevant attribute value, instead of selecting all values of an attribute. On the other hand, if the dataset contains many attributes where all of their values are relevant, a decision tree could represent those patterns more naturally than a rule set/list.

We suggested that, when analyzing decision trees and rule sets or lists, a simple yet effective measure of the relevance of an attribute in those models would be the number of test instances whose classification used a value of that attribute. The interpretation of decision trees and rule sets or lists could also include an analysis of the exceptions (or counter-examples) associated with some rules – i.e., instances which satisfy a rule's antecedent (or decision tree path from root to leaf) but do not have the class predicted by the rule. Such counter-example analyses could lead to more insight about the application domain.

Concerning graphical models, although both Bayesian network classifiers and decision trees are graphical models, the correct interpretation of the former seems more difficult to users, although we are not aware of any experimental study comparing their interpretability by users. It has been pointed out that dependence networks may represent more comprehensible probabilistic graphical models than Bayesian network classifiers [32], since the former (unlike the latter) can represent cyclic dependences among attributes, which are often intuitive to users. The use of both dependence networks and the class attribute's Markov blanket as types of comprehensible classification models seems relatively under-explored in the literature.

Although nearest neighbor algorithms usually do not produce an explicit, abstract classification model, the attribute values of the found nearest neighbor(s) can sometimes be used to explain the classification of new instances. That explanation could be improved by showing attributes to users in decreasing order of relevance as computed by an attribute-weighting method, and perhaps using prototypes of training instances.

Turning to evaluation issues, there have been few experiments where users subjectively evaluated the comprehensibility of classification models expressed in different types of knowledge representations. There is a need for more experiments, e.g. along two directions: including user evaluations of Bayesian network classifiers, since previous experiments have focused mainly on decision trees, classification rules and decision tables; and involving more users who are experts in the application domain – previous experiments have used mainly students as "users".

We have also noted that the widely used assumption that smaller models are more comprehensible than larger models is problematic. First, in some studies users found larger models to be *more* comprehensible than smaller models [1], [35], [46], apparently because larger models provided more classification-relevant information to users. Second, users are unlikely to accept models that are very simple when the underlying concept being learned by the algorithm is considered complex [20]. In addition, although a model that is "too large" can hardly be interpreted by a user, it might be possible for a user to focus on the analysis of an interesting subset of the model – e.g. a substree of the decision tree. Note also that it is very hard to define *a priori* which model size would be considered "too large" to be analyzed by a user. Hence, instead of specifying the maximum size of a classification model as a parameter of a classification algorithm, we prefer a more principled approach to cope with the accuracy-comprehensibility trade-off, such as a multi-objective approach based on Pareto dominance or lexicographic optimization [25].

We also discussed the use of monotonicity constraints in classification models, to improve their comprehensibility and acceptance by users. Such constraints involve a semantic aspect of classification models, instead of just considering a syntactical aspect such as model size. Also, monotonicity constraints seem to have a good degree of generality: they seem to exist in many (but not all) real-world applications, and they can be incorporated in all the types of classification model representations discussed earlier – although each classification algorithm has to be adapted to consider monotonicity constraints in a specific way.

Monotonicity constraints can be considered as hard constraints (which can never be violated during model induction) or soft constrains (which can be violated if this leads to significantly higher predictive accuracy), depending on the application domain and the user's interests. Monotonicity constraints can be specified by an expert on the application domain or automatically extracted from (high-quality) data, depending on the application. More research is needed to determine to what extent an automated approach to extract semantic monotonicity constraints from data leads to improved classification models.

6. REFERENCES

[1] Allahyari, H., and Lavesson, N. User-oriented assessment of classification model understandability. *Proc. 11th Scandinavian Conf. on Artificial Intelligence*. IOS, 2011.

[2] Altendorf, E.E., Restificar, A.C., and Dietterich, T.G. Learning from sparse data by exploiting monotonicity constraints. *Proc. 21st Annual Conf. on Uncertainty in Artificial Intelligence (UAI-05)*, 18-26. AUAI, 2005.

[3] Augusta, M.G., and Kathirvalavakumar, T. Reverse engineering the neural networks for rule extraction in classification problems. *Neural Processing Letters* 35(2): 131-150, April 2012.

[4] Baesens, B., Mues, C., De Backer, M., and Vanthienen, J. Building intelligent credit scoring systems using decision tables. In: *Enterprise Information Systems V*, 131-137. Kluwer, 2004.

[5] Bellazzi, R., and Zupan, B. Predictive data mining in clinical medicine: current issues and guidelines. *International Journal of Medical Informatics* 77(2): 81-97, Feb. 2008.

[6] Ben-David, A. Monotonicity maintenance in information-theoretic machine learning algorithms. *Machine Learning* 19(1)*:* 29-43. 1995.

[7] Ben-David, A., Sterling, L., and Tran, T. Adding monotonicity to learning algorithms may impair their accuracy. *Expert Systems with Applications* 36(3): 6627-6634. April 2009.

[8] Boz, O. Extracting decision trees from trained neural networks. *Proc. 8th ACM SIGKDD Int. Conf. on Knowledge*

Discovery and Data Mining (KDD'02), 456-461. ACM, 2002.

[9] Bramer, M. *Principles of Data Mining*. Springer, 2007.

[10] Cendrowska, J. PRISM: an algorithm for inducing modular rules. *International Journal of Man-Machine Studies* 27(4): 349-370. 1987.

[11] Cheng, J., and Greiner, R. Learning Bayesian belief classifiers: algorithms and system. *Proc. 14th Biennial Conference of Canadian Society on Computational Studies of Intelligence (AI'01)*, 141-151. Springer, 2001.

[12] Clark, P., Boswell, R. Rule induction with CN2: some recent improvements. In: *Machine Learning – Proc. Fifth European Conf. (EWSL-91)*, 151-163. Springer, 1991.

[13] Cristianini, N., and Shawe-Taylor, J. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[14] Dejaeger, K., Goethals, F., Giangreco, A., Mola, L., and Baesens, B. Gaining insight into student satisfaction using comprehensible data mining techniques. *European Journal of Operational Research*, 218(2): 548-562, 2012.

[15] Dhar, V., Chou, D., and Provost, F. Discovering interesting patterns for investment decision making with GLOWER – a genetic learner overlaid with entropy reduction. *Data Mining and Knowledge Discovery* 4(4): 251-280, 2000.

[16] Doderer, M., Yoon, K., Salinas, J., and Kwek, S. Protein subcellular localization prediction using a hybrid of similarity search and error-correcting output code techniques that produces interpretable results. *In Silico Biology* 6(5): 419-433, 2006.

[17] Domingos, P. Occam's two razors: the sharp and the blunt. *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD'98)*, 37-43. AAAI, 1998.

[18] Duivesteijn, W., and Feelders, A. Nearest neighbour classification with monotonicity constraints. *Proc. ECML PKDD 2008, Part I, LNAI 5211*, 301-316. Springer, 2008.

[19] Elazmeh, W., Matwin, W., O'Sullivan, D., Michalowski, W., and Farion, W. Insights from predicting pediatric asthma exacerbations from retrospective clinical data. In: *Evaluation Methods for Machine Learning II – Papers from 2007 AAAI Workshop*, 10-15. *Technical Report WS-07-05*. AAAI, 2007.

[20] Elomaa, T. In Defense of C4.5: Notes on learning one-level decision trees. *Proc. 11th Int. Conf. on Machine Learning (ICML-94)*, pp. 62-69. Morgan Kaufmann, 1994.

[21] Feelders, A.J. Prior knowledge in economic applications of data mining. *Proc. European Conf. on Principles and Practice of Knowledge Discovery and Data Mining (PKDD-2000), LNAI 1910*, 395-400. Springer, 2000.

[22] Feelders, A., and Pardoel, M. Pruning for monotone classification trees. *Proc. Intelligent Data Analysis (IDA) Conf., LNCS 2810*, 1-12. Springer, 2003.

[23] Ferri, C., Hernandez-Orallo, J., and Ramirez-Quintana, M.J. From ensemble methods to comprehensible models. *Proc. 5th Int. Conf. on Discovery Science (DS-2002), LNCS 2534*, 165-177. Springer, 2002.

[24] Freitas, A.A. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer, 2002.

[25] Freitas, A.A. A critical review of multi-objective optimization in data mining: a position paper. *ACM SIGKDD Explorations*, 6(2): 77-86. ACM, Dec. 2004.

[26] Freitas, A.A., Wieser, D.C. and Apweiler, R. On the importance of comprehensible classification models for protein function prediction. *ACM/IEEE Transactions on Computational Biology and Bioinformatics* 7(1): 172-182, Jan.-Mar. 2010.

[27] Friedman, N., Geiger, D., and Goldszmidt, M. Bayesian network classifiers. *Machine Learning* 29(2-3): 131-163, Nov./Dec. 1997.

[28] Fung, G., Sandilya, S., and Rao, R.B. Rule extraction from linear support vector machines. *Proc. 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD-2005)*, 32-40. ACM, 2005.

[29] Furnkranz, J. Separate-and-conquer rule learning. *Artificial Intelligence Review* 13(1): 3-54. 1999.

[30] Grunwald, P.D. *The Minimum Description Length Principle*. MIT Press, 2007.

[31] Hayete, B., and Bienkowska, J.R. GOTrees: predicting GO associations from protein domain composition using decision trees. *Proc. Pacific Symp. on Biocomput. 10*, 127-138, 2005.

[32] Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., and Kadie, C. Dependency networks for inference, collaborative filtering and data visualization. *Journal of Machine Learning Research* 1: 49-75, 2000.

[33] Henery, R.J. Classification. In: Michie, D., Spiegelhalter, D.J., and Taylor, C.C. *Machine Learning, Neural and Statistical Classification*, 6-16. Ellis Horwood, 1994.

[34] Huang, J., and Ling, C. Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 17(3): 299-310, 2005.

[35] Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., and Baesens, B. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems* 51(1): 141-154. 2011.

[36] Japkowicz, N., and Shah, M. *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[37] Jiang, T., and Keating, A.E. AVID: an integrative framework for discovering functional relationships among proteins. *BMC Bioinformatics* 6:136, 2005.

[38] Jin, Y. (Ed.) *Multiobjective Machine Learning*. Springer, 2006.

[39] Johansson, U., and Niklasson, U. Evolving decision trees using oracle guides. *Proc. 2009 IEEE Symp. on Computational Intelligence and Data Mining (CIDM 2009)*, 238-244. IEEE Press, 2009.

[40] Karpf, J. Inductive modelling in law: example based expert systems in administrative law. *Proc. 3rd Int. Conf. on Artificial Intelligence in Law*, 297-306. ACM, 1991.

[41] Karwath, A., and King, R.D. Homology induction: the use of machine learning to improve sequence similarity searches. *BMC Bioinformatics* 3:11, 2002.

[42] Kaufmann, K.A., and Michalski, R.S. Learning from inconsistent and noisy data: the AQ18 approach. *Foundations of Intelligent Systems (Proc. ISMIS-99). LNAI 1609*, 411-419. Springer, 1999.

[43] Kohavi, R. The power of decision tables. *Proc. 1995 European Conf. on Machine Learning (ECML-95), LNAI 914*, 174-189. Springer, 1995.

[44] Kohavi, R., and Sommerfield, D. Targeting business users with decision table classifiers. *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining (KDD-98)*, 249-253. AAAI, 1998.

[45] Kononenko, I. Inductive and Bayesian learning in medical diagnosis. *Applied Artificial Intelligence* 7(4): 317-337, 1993.

[46] Lavrac, N. Selected techniques for data mining in medicine. *Artificial Intelligence in Medicine* 16(1): 3-23, May 1999.

[47] Lima, E., Mues, C., and Baesens, B. Domain knowledge integration in data mining using decision tables: case studies in churn prediction. *Journal of the Operational Research Society* 60: 1096-1106, 2009.

[48] Maes, R., and Van Dijk, J.E.M. On the role of ambiguity and incompleteness in the design of decision tables and rule-based systems. *The Computer Journal* 31(6): 481-489. 1988.

[49] Marteens, D., Vanthienen, J., Verbeke, W., and Baesens, B. Performance of classification models from a user perspective. *Decision Support Systems* 51(4): 782-793. 2011.

[50] Michie, D., Spiegelhalter, D.J., and Taylor, C.C. (Eds.) *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994.

[51] Otero, F.E.B. and Freitas, A.A. Improving the interpretability of classification rules discovered by an ant colony algorithm. *Proc. 2013 Genetic and Evolutionary Computation Conference (GECCO'13)*, 73-80. ACM, 2013.

[52] Pappa, G.L., Baines, A.J., and Freitas, A.A. Predicting post-synaptic activity in proteins with data mining. *Bioinformatics* 21(Suppl. 2): ii19-ii25, 2005.

[53] Pazzani, M. Comprehensible Knowledge Discovery: Gaining Insight from Data. *Proc. First Federal Data Mining Conf. and Exposition*, 73-82. Washington, D.C., 1997.

[54] Pazzani, M.J. Learning with globally predictive tests. *Proc. Discovery Science (DS'98), LNAI 1532*. Springer, 1998.

[55] Pazzani, M.J., Mani, S., and Shankle, W.R. Acceptance of rules generated by machine learning among medical experts. *Methods of Information in Medicine*, 40(5): 380-385, 2001.

[56] Quinlan, J.R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[57] Quinlan, J.R. Some elements of machine learning. *Proc. 16th Int. Conf. on Machine Learning (ICML-99)*, 523-525. Morgan Kaufmann, 1999.

[58] Quinonero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N.D. (Eds.) *Dataset Shift in Machine Learning*. MIT Press, 2009.

[59] Richards, G., Rayward-Smith, V.J., Sonksen, P.H., Carey, S., and Weng, C. Data mining for indicators of early mortality in a database of clinical records. *Artificial Intelligence in Medicine* 22(3): 215-231, June 2001.

[60] Rokach, L. *Pattern Classification Using Ensemble Methods*. World Scientific, 2010.

[61] Rokach, L. and Maimon, O. *Data Mining with Decision Trees: theory and applications*. World Scientific, 2008.

[62] Schwabacher, M., and Langley, P. Discovering communicable scientific knowledge from spatio-temporal data. *Proc. 18th Int. Conf. on Machine Learning (ICML-2001)*, 489-496. Morgan Kaufmann, 2001.

[63] Sen, S., and Knight, L. A genetic prototype learner. *Proc. 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*. 1995.

[64] Sokolova, M., and Lapalme, G. A systematic analysis of performance measures for classification tasks. *Information Processing and Management* 45(4): 427-437, July 2009.

[65] Subramanian, G.H., Nosek, J., Raghunathan, S.P., and Kanitkar, S.S. A comparison of the decision table and tree. *Communications of the ACM* 35(1): 89-94, Jan. 1992.

[66] Suri, N.R., Srinivas, V.S. and Murty, M.N. A cooperative game theoretic approach to prototype selection. *Proc. 2007 European Conf. on Machine Learning (ECML 2007), LNAI 4701*, 556-564. Springer, 2007.

[67] Szafron, D., Lu, P., Greiner, R., Wishart, D.S., Poulin, B., Eisner, R., Lu, Z., Anvik, J., Macdonell, C., Fyshe, A., and Meeuwis, D. Proteome analyst: custom predictions with explanations in a web-based tool for high-throughput proteome annotations. *Nucleic Acids Research* 32(Supp. 2): W365-W371, 2004.

[68] S. Tsumoto. Clinical knowledge discovery in hospital information systems: two case studies. *Proc. Europ. Conf. on Principles and Practice of Knowledge Discovery and Data Mining (PKDD-2000), LNAI 1910*, 652-656. Springer, 2000.

[69] van Assche, A., and Blockeel, H. Seeing the forest through the trees: learning a comprehensible model from an ensemble. *Proc. 2007 European Conf. on Machine Learning (ECML 2007), LNAI 4701*, 418-429. Springer, 2007.

[70] Verbeke, W., Marteens, D., Mues, C., and Baesens, B. Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert Systems with Applications* 38(3): 2354-2364. 2011.

[71] Wettschereck, D., Aha, and D.W., Mohri, T. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. In: Aha, D.W. (Ed.) *Lazy Learning*, 273-314. Kluwer, 1997.

[72] Witten, I.H., Frank, E., and Hall, M.A. *Data Mining: practical machine learning tools and techniques. 3rd Ed.* Morgan Kaufman, 2011.

[73] Wong, M.L., and Leung, K.S. *Data Mining Using Grammar-Based Genetic Programming & Applications*. Kluwer, 2000.

[74] Zahalka, J., and Zelesny, F. An experimental test of Occam's Razor in classification. *Machine Learning* 82(3): 475-481, March 2011.

[75] Zhang, J. Selecting typical instances in instance-based learning. *Proc. 9th Int. Workshop on Machine Learning (ML-92)*, 470-479. 1992.

[76] Zupan, B., Demsar, J., Kattan, M.W., Beck, J.R., and Bratko, I. Machine learning for survival analysis: a case study on recurrence of prostate cancer. *Artificial Intelligence in Medicine* 20(1): 59-75, Aug. 2000.

About the author:

Alex A. Freitas is a Professor of Computational Intelligence at the University of Kent, UK. He received his PhD in Computer Science from the University of Essex, UK, in 1997. His publications include three (co)-authored research-oriented books and about 50 peer-reviewed journal papers and 110 peer-reviewed papers in conference proceedings. His publications have over 7,000 citations in Google Scholar, with an H-index of 43. His main research interests are classification (supervised learning) methods and their application to the life sciences (including biology and pharmaceutical sciences) – with a special interest in applying classification methods to the biology of ageing.