

# Construction of Deterministic, Consistent, and Stable Explanations from Numerical Data and Prior Domain Knowledge

Katrina Riehl  
Enthought, Inc.  
Austin, Texas 78701  
kriehl@enthought.com

Klaus Truemper  
University of Texas at Dallas  
Richardson, Texas 75080  
truemper@utdallas.edu

## ABSTRACT

In the explanation problem, training records randomly taken from two populations and, possibly, partial prior domain knowledge are given. Two deterministic explanations for the differences between the two populations are to be constructed, or it must be declared that such explanations most likely are not contained in the data. The explanations must be accurate and in conformance with the given prior domain knowledge.

This paper presents a multi-step solution algorithm called EXARP for the explanation problem. A key feature is the use of several alternate random processes (ARPs) which attempt to distort data or otherwise disrupt the solution process. Appropriate counteractions prevent or at least mitigate the negative effects of the ARPs.

EXARP is potentially useful in domains such as bioinformatics, economics, engineering, finance, medicine, and psychiatry. To-date, the algorithm has largely been used to explain medical data. The explanations may be used in several ways, for example, to define tailored or targeted therapies. In one such case, explanations produced by EXARP have led to a new breast cancer treatment that is currently being evaluated and has the potential of saving a large number of lives.

## Categories and Subject Descriptors

I.2.6 [Computing Methodologies]: Artificial Intelligence-Learning

## General Terms

Alternate random process, explanation, learning, logic

## 1. INTRODUCTION

We define the *explanation problem* as follows. Records of two training sets  $A$  and  $B$  taken randomly from two populations  $\mathcal{A}$  and  $\mathcal{B}$ , respectively, are given. The attributes of the records may be numerical or nominal, and some entries may be missing and presumably cannot be obtained for various reasons. Possibly, partial prior domain knowledge is also given. We call such knowledge *prior facts*. Either two explanations in if-then format must be constructed, or it must be declared that explanations most likely are not contained in the data. The two explanations must be in conformance with the given prior facts. The if-conditions must be so simple that persons with minimal mathematical background can understand them. Specifically, the conditional part of each explanation may only contain a few elementary equations and inequalities that involve one attribute each and that are linked by the logic “and” and “or.” The conclusions must be deterministic claims of membership in population  $\mathcal{A}$  for one explanation and in population  $\mathcal{B}$  for the second explanation. The if-conditions of the two explanations must be such that they cannot evaluate simultaneously to *True*. When this condition is satisfied, we say that the explanations are *consistent*. The explanations are not allowed to flip from one conclusion to the opposite one by small random changes of the data. When this condition holds, the explanations are said to be *stable*.

The two explanations must be verified as likely correct by statistical tests using a given additional testing set. For the statistical tests, the following rule is used. If an if-condition evaluates to *True*, and if the conclusion is correct (resp. incorrect), then the case is counted as *correct* (resp. *incorrect*). If both if-conditions evaluate to *False*, then the conclusion is “This record cannot be classified.” The latter case is counted as *undecided*. The error performance is a weighted sum of the incorrect and undecided cases, where the weights are domain-dependent and reflect the consequences of incorrect and undecided classifications.

In an additional optional step, the explanations may be validated by one or more experts using procedures consistent with practice of the domain. The evaluation process may thus range from an experience- and intuition-based judgment of one expert to a formal process involving experts, lab tests, clinical studies, and so on. Expert evaluation is also optional when it is declared that explanations of the desired kind most likely are not contained in the data.

We have two reasons for demanding two explanations instead of just one. First, when explanations are required to be deterministic as done here, then two explanations seem the simplest way to introduce notions of stability and undecided cases. Second, two consistent and stable explanations illuminate and clarify the differences between the two populations from two opposite viewpoints.

The rest of the paper proceeds as follows. Section 2 discusses two typical applications. Sections 3 and 4 compare the explanation problem with the separation problem and subgroup discovery problem, respectively, and review related methods.

Section 5 gives an overview of the construction method. Sections 6–11 provide details of the steps and the validation process. Section 6 deals with discretization. Section 7 covers the derivation of logic formulas from data. Section 8 explains the selection of important factors. Section 9 handles the case where prior facts are part of the input. Section 10 summarizes the method. Section 11 deals with the validation of explanations.

Sections 12–15 address advanced features of the algorithm and various other issues. Section 12 describes how qualitatively specified effects can be explained. Sections 13, 14, and 15 cover overfitting, manual guidance, and computational complexity, respectively. Section 16 describes three example applications. Finally, Section 17 summarizes the results.

## 2. APPLICATION EXAMPLES

We motivate the explanation problem by two examples.

In the first example, a medical expert is puzzled by unexpected results of a preliminary clinical study. For example, he/she may want to find out why a cure is achieved only in some cases. The explanations may be used in several ways, for example, to direct research efforts, or to define tailored or targeted therapies. Section 16.2 covers an instance, where the explanations have led to a new breast cancer treatment that is currently being evaluated and has the potential of saving a large number of lives. If the explanations are to be credible, then they must be consistent and stable as defined above. Indeed, in our experience, domain experts typically mistrust inconsistent or unstable explanations. Also, the construction scheme must be able to recognize situations where explanations of the desired kind cannot be derived from the data. A statement to that effect is preferable to forced explanations that an expert declares to be artificial or contrived.

We interrupt the technical discussion for a moment and mention how a key difficulty of clinical trials is addressed by the method proposed here. Current evaluation techniques used in such trials typically require 500-1,000, sometimes even more, patients to prove effectiveness. This requirement arises from the large number of attribute values per patient, the anticipated variety of responses, and the inability of the techniques to derive conclusive results for smaller sample sizes. Even in such a large-scale study, the techniques may fail to characterize the possibly small subset of the patient population that can be successfully treated. Singer summa-

rizes this key problem using the failed drug trial of Iressa, a potential drug for lung cancer, as an example [56]: “Iressa was heralded as key example of the potential of targeted cancer therapies. But its power may have been lost in the recent large-scale trial. The 1,700 patients were part of the general patient population, rather than the subgroup most likely to respond to the drug. [David] Johnson [President, American Society of Clinical Oncology] says clinical trials must be designed to clarify the true potential of the growing number of targeted therapies. He and others are trying to change the way drugs are evaluated. One solution is to better understand how a drug acts and *why some patients respond* [Emphasis added] before beginning large-scale trials.” The method proposed here is designed to find the reasons for a positive response, as demanded by Singer. The method can accomplish this with small patient data sets, say involving 50-60 patients, and thus can produce the desired insight in the preliminary phases I and II of drug testing. The claim that 50-60 patients suffice is based on evaluation of a number of data sets in various settings within and outside medicine. Two cases are included in Sections 16.1 and 16.2.

In the typical evaluation of clinical data, prior facts may enter in at least two ways. First, such facts may be given up front. Second, an expert’s evaluation of initially obtained explanations may be something like “The explanations are reasonable, but . . . ,” where the “but” is followed by a general rule or consideration that we view as a prior fact. That fact may or may not be implicitly contained in the training data. Regardless of the case, we make the prior fact part of the input data and recompute the explanations, which now are in conformance with that fact.

In the second example, publicly available financial data plus certain index data supplied by a financial services company are to be used to predict investment performance. In a project of this type, the method described later declared that future performance could not be predicted by reliable deterministic explanations derived from the available data. The result is compatible with financial theory, in particular with the conclusions of [21]. But the method was able to construct compact explanations for past performance from the data. These explanations solely relied on the index data and helped identify investment opportunities with strong performance record.

The next section compares the explanation problem and the separation problem, and discusses the extent to which methods for the latter problem apply here.

## 3. EXPLANATION VERSUS SEPARATION

It may seem that the explanation problem is just a minor variation of the separation problem where two populations must be separated by some function or rule. We argue that this is not so. We begin by summarizing two general differences, then analyze in detail whether current separation methods can solve the explanation problem.

First, the main goal of separations is high predictive accuracy. In contrast, explanations are to provide an understanding of the differences between two populations. In the medical example of Section 16.2, the expert wants to know why a cure isn’t always achieved. If explanations supply that

insight with, say, 85% predictive accuracy, then they are preferred to incomprehensible separating rules with, say, 90% accuracy. For example, the explanations may be the take-off point for revision of the treatment or further medical research. Somewhat more accurate but incomprehensible separation rules do not supply insight for these steps.

Second, the explanation problem requires recognition of the situation where consistent and stable explanations most likely are not contained in the training data. No such demand is made in the separation problem. It could be argued that this aspect could be added to the separation problem, and that it could be handled via an accuracy threshold on predictive accuracy. The following example shows that the problem of giving up, so to speak, is more difficult. Let training sets  $A$  and  $B$  have 7 records each. There is just one attribute  $x$ . The values for  $x$  in the 7 records of set  $A$  (resp.  $B$ ) are 1.0, 1.0, 1.0, 1.0, 5.0, 7.0, 10.0 (resp. -7.0, -2.0, 0.0, 0.99, 0.99, 0.99, 0.99). The training records are completely separated by the two consistent rules “If  $x \geq 0.995$ , then in population  $\mathcal{A}$ ” and “If  $x < 0.995$ , then in population  $\mathcal{B}$ .” However, if  $x$  is considered to be subject to random changes, then it is quite appropriate to consider the values 1.0 of  $A$  records and the values 0.99 of  $B$  records to be random variations of each other. This implies that the two separation rules are not stable. In fact, stable explanations cannot exist for this example where a large percentage of the training records of  $A$  is separated from a large percentage of the  $B$  records. In this case, the algorithm presented shortly would give up and declare that explanations likely do not exist.

We now establish the extent to which current separation methods can solve the explanation problem.

The conditions imposed here on explanations generally rule out Bayesian networks, nearest-neighbor techniques, neural nets, and support vector machines as candidate construction methods. The situation is different for schemes constructing decision trees or rules.

### 3.1 Decision Trees

In a decision tree, each path from the root node to any leaf node is readily restated as an explanation. However, current construction techniques for decision trees typically focus on high prediction accuracy and may fail to consistently produce trees that provide valid explanations. For example, in tests of the decision tree method C4.5 [51] reported in Section 16, the explanations are not trustworthy or most likely wrong. In another example, reported in [66], the comparison of four decision tree methods produced a winning scheme whose implied rules were characterized as follows: “The first rules extracted using the intelligent platform [the winning scheme] show some similarities with already existing knowledge.” This is not a strong claim.

### 3.2 Rules

Almost all current construction schemes of rules also concentrate on prediction accuracy and may produce rule sets that, when viewed as explanations, do not seem useful. Here are examples.

1. Reference [1] derives 12 rules for the Wisconsin Breast

Cancer Data [45]. Each rule uses 5 attributes, and the discretization introduces up to 3 intervals per attribute, a rather complex description if used as explanation.

2. Reference [7] combines a construction method of rules with entropy-based discretization techniques. For almost every case discussed in the reference, the number of rules is too large to permit their use as explanations.
3. Reference [58] focuses on the discovery of rule patterns instead of explanations. The methods of the reference typically produce too many rules to be useful as explanations.
4. The STUCCO method of [11] mines contrast sets to exhibit group differences by separating rules. Judging from the application discussed in detail in the reference, the rules may not be explanations of the form demanded here. The application concerns students who chose to, or not to, enroll at UC Irvine. STUCCO produces 26 rules for one class and 19 rules for the other class. The rule sets are not consistent. For example, positive yield rule #13 “Students who are not born in the US are more likely to enroll [at UC Irvine]” clashes with negative yield rule #15 “Students who are Chinese are less likely to enroll [at UC Irvine].”
5. The methods CAEP [19], JEP-C [33], and eJEP [22] have excellent predictive accuracy, but produce rule sets that are too large to serve as explanations. For example, [22] cites rule sets for well-known data sets with the number of rules ranging from 54 to thousands.
6. The LAD method [12, 13] generates patterns by an experiment-guided process and solves set covering problems to obtain explanations. The number of rules of the explanations can be large. For example in [5], the method produces explanations using 42 high risk patterns and 77 low risk patterns. A drawback of the method is the fact that it requires considerable manual guidance. Reference [5] demonstrates this fact.
7. The two versions of the improved CN2 method [15] generate rule sets for well-known data sets with average number of rules ranging from 27 to 109 depending on the version and a certain significance threshold. This is too large to be useful for explanations. For the original CN2 method [16], reference [15] reports that the average number of rules for the same data sets has a much smaller average value of 6 for one of the significance thresholds. On the other hand, reference [36] reports for the original CN2 an average of 18 rules for well-known data sets. The accuracy of the original CN2 is substantially below that of the improved versions for 5 of the 12 data sets, and even on average has worse accuracy.
8. The method RIPPER [17] generates according to [36] an average of 16 rules for well-known data sets. Accuracy is good. We apply RIPPER in Section 16 for comparison purposes.
9. The accurate rule construction method SLIPPER [18] produces rule sets for well-known data sets that, according to the reference, have on average at least 18

rules. Also, the classification by the rules uses weights and thus effectively is probabilistic and not deterministic as demanded here. This may sound like a minor variation. But there is a substantial difference between saying “These are the cases of disease  $X$ ” and “These are the cases when  $X$  is not present” on one hand, and saying that certain rules must be evaluated, and the results weighted and summed, to decide whether  $X$  is present.

There are additional rule-based separation methods that for sundry reasons cannot be used, or can be used only in restricted settings, to obtain explanations.

10. The APRIORI-C algorithm [34], which is based on the APRIORI algorithm [3, 4] encodes missing values by representing them as another numerical case. In principle, this may lead to a conclusion when most or all values are actually missing. In medicine, this would be unacceptable. The algorithm has relatively poor accuracy on certain numerical data sets due to the selected discretization scheme. According to [36], the rule sets are of reasonable size.
11. The method ROCCER [49], which selects rules from a larger set created by the APRIORI algorithm [3, 4], does not use the above approach of APRIORI-C for missing values. The reference declares that missing values cannot be handled. Accuracy and rule sizes reported in [49] are attractive. Since these results apply only to data sets without missing entries, one naturally wonders how performance might be when instead of APRIORI a rule generation scheme is used that tolerates missing values. There are indications that the latter restriction does matter. For example, rule sets computed by the two versions of CN2 [15] for the data sets of that reference, which sometimes have missing entries, are much larger than rule sets reported for the same methods in [49] for data sets without missing entries. This fact raises the question of how APRIORI-C would fare if the method was forced to process missing values correctly. Finally, ROCCER sometimes requires significant computing effort. As cited in [49], the run times on the data sets of the reference range from a few seconds to 10 min for most cases, but also can require up to 1.5 hr.
12. The profile construction method of [61] computes characterizations of classes that are too concise and approximate to serve as explanations.

There is an additional aspect. The cited tree and rule construction methods typically employ a discretization for numerical data that often causes the computed explanations to be not stable. Section 16 demonstrates this aspect for C4.5 and RIPPER.

The subgroup discovery problem asks for results that at first glance seem almost identical to those demanded by the explanation problem. The next section discusses this aspect.

## 4. EXPLANATION VERSUS SUBGROUP

The subgroup discovery problem was first defined in [38] and [63]. The goal is the discovery of interesting subgroups in a population  $\mathcal{P}$ . The subgroups are described by patterns or rules that relate explaining variables and a target variable. Interestingness is measured with quality functions balancing conflicting goals [39]. Typical goals are the size of the subgroup, the length of the pattern or rule defining the subgroup, and the significance of the subgroup, which measures the extent to which the subgroup falls within a specified group identified by the target variable.

When the subgroup discovery problem is specialized to the case considered here, then the population  $\mathcal{P}$  is the disjoint union of two populations  $\mathcal{A}$  and  $\mathcal{B}$ , the target variable indicates membership in  $\mathcal{A}$  or  $\mathcal{B}$ , and the explaining variables are the attributes of records. For comparison purposes, we assume that setting in the discussion below. Evidently, in that case the goal of interesting subgroups is similar to that of explanations. But the conditions imposed on subgroups differ from those for explanations. Subgroups must be interesting, while explanations not only must tell key differences between the populations, but also must be consistent and stable. In addition, since a subgroup is allowed to intersect both populations, classification via the defining rules of subgroups necessarily is probabilistic, while explanations must be deterministic. This is a substantial difference, as argued in Section 3.

Typical for subgroup discovery is an iterative investigation involving the steps of data mining, interpretation of results, and evaluation by experts; for a methodology, visualization technique, and example applications, see [9, 27, 28, 29, 42]. The computing process of explanations introduced here is different. To be sure, the process sometimes involves two iterations, particularly when the expert brings up prior facts that either are missing from initial explanations or are not easily recognized as implied. In such a case, the prior facts are added to the explanations, and the computing process is carried out again. A very simple example case is discussed in Section 9. But this is not the usual situation. If the supplied data are correct, then typically the method proposed here either computes explanations that quite often supply insight and are accepted, or it declares that explanations very likely cannot be obtained from the data. In both cases, the process stops.

Prior domain knowledge plays different roles in subgroup discovery and the construction of explanations. For the former task, such knowledge, in varying formats, can be used in a number of ways. For example, to focus the investigation on important patterns; to differentiate between abnormality and normality; to combine patterns; or to improve the handling of missing values [8]. In contrast, prior facts used here must have the tightly specified format of if-then rules. Their inclusion assures that explanations are in conformance with that knowledge and is done in such a way that the role of the prior facts is evident from the explanations; see Section 9.

There are numerous methods for subgroup discovery. We omit a detailed review of the schemes since they do not solve the explanation problem considered here. The first algorithms were EXPLORA [38] and MIDOS [63, 64]. These

methods have been followed by many other schemes. We only cite here the methods CN2-SD [43] and APRIORI-SD [36], since they are constructed from the already discussed induction schemes CN2 [16, 15] and APRIORI-C [34] by a general technique that applies to other induction schemes as well.

## 5. PROPOSED METHOD

This paper proposes a solution method for the explanation problem that involves four steps: discretization, feature selection, derivation of explanations, and validation. Each of the steps is well known in machine learning. However, most details are different from prior methods, as follows. The discretization step relies on intervals of uncertainty that induce stability of explanations. The feature selection step derives the important attributes in one step using a logic minimization process instead of an iterative approach as done in virtually all prior methods. The computation of explanations uses logic minimization instead of rule/branching-variable selection and pruning as typically done in prior methods. The validation step includes testing of consistency and, possibly, a remedial process for inconsistent explanations. Finally, the method accommodates prior facts so that the explanations are in conformance with that information.

The method can run without manual guidance; details are given in Section 14. All steps of the method except the last one rely on alternate random processes described next.

### 5.1 Alternate Random Process

An *alternate random process*, for short ARP, is a postulated random process whose goal is to disrupt or mislead a computing procedure. Analysis of an ARP motivates an algorithmic approach that largely avoids the negative impact of the ARP. An example illustrates the idea. A core problem of discretization is the fact that numerical values just below a cutpoint may move by random variations above the cutpoint, or *vice versa*. The result is a different discretized representation for certain, virtually identical, cases, a bad effect that potentially results in unstable explanations. One ARP of the discretization step models such random movements. Analysis of the impact of the ARP leads to a discretization strategy that declares values close to the cutpoint to be unusable and thereby avoids the potential discretization errors associated with these values. Of course, subsequent steps of the method must process the case of unusable values. We see later how that complication can be handled.

The responses of the various algorithms to ARP actions are so designed that all ARP actions occurring with probability greater than a given value or satisfying an equivalent criterion, are counteracted. For example, in the discretization step, misleading ARP-induced configurations are effectively obliterated by the algorithm if they occur on average at least once.

The notion of ARP is different from the concepts underlying probability or uncertainty models such as Bayesian networks, Naive Bayes classifiers, or fuzzy sets. For an overview of these methods, see [54]. In these settings, probabilistic or uncertainty behavior is postulated in a model, which is then used directly to classify data. Indeed, the model is the main evaluation device. In contrast, each ARP is an antago-

nist whose disruptive actions try to confuse the construction method.

The notion of ARP is related to the concept of *randomization testing* [32], where an empirical distribution of a statistic under a null hypothesis  $H_0$  is constructed by evaluation of random samples. The distribution is used to estimate the probability of a given result under  $H_0$ . The cited reference describes an interactive rule construction method called IRT that is based on randomization testing. In [48], an application of randomization testing called *randomization pruning* reduces excess structure in decision trees for large data sets. Randomization testing and ARPs have in common that they postulate an underlying random process and evaluate it for a certain decision. For example, in [48], the decision is whether to prune a branch of a tree. For the ARPs of the subsequent sections, the decisions concern the variance of a Gaussian convolution step, the intervals of uncertainty around cutpoints, and the importance of learned DNF clauses. Randomization testing and the ARPs proposed here differ in that the former process constructs empirical distributions, while the ARPs used here are probabilistic models that are *a priori* specified except for one parameter whose value is readily available or obtained by simple computation.

How were the ARPs and the corresponding modifications of the computing process formulated? In each case, the trigger for using an ARP was some troubling result produced by computations for sample training sets. We then postulated various ARPs that intuitively seemed to be the cause of the difficulties. For each ARP, we selected a temporary modification of the computing process that, we guessed, would eliminate or at least mitigate the distortion and obfuscation of the ARP. Tests showed which action was most effective. We picked the corresponding ARP, investigated it mathematically in detail, and used that insight to construct a method that eliminated or reduced the negative impact of the ARP actions.

Due to the key role of ARPs, the proposed solution method is called EXARP (EXplanations via ARPs).

We discuss details of EXARP, beginning with the discretization step. To simplify the discussion, we assume that the input consists of training data and does not include any prior facts. The discussion proceeds under this assumption until Section 9, which supplies the required modifications when prior facts are part of the input.

## 6. DISCRETIZATION PROCESS

The discretization step consists of a modified version of the pattern analysis process of [10] and an additional, new step creating intervals of uncertainty that contain the cutpoints. The reference contains a detailed review of prior discretization methods. Here, we only mention that entropy combined with the minimum description length principle [23, 24, 50] and other schemes using information gain have been the most widely used methods, with strong performance with regard to prediction accuracy; see, for example, [6, 7, 20, 40]. While entropy-based methods consider all data of a given attribute simultaneously and thus are global in some sense, the pattern analysis of [10] and the computation of

intervals of uncertainty rely solely on local phenomena. We prefer the latter approach based on the preliminary empirical results of [47] for the original pattern analysis [10] and, more importantly, due to two reasons that we discuss after an overview of the discretization step.

For a given numerical attribute, discretization generally introduces cutpoints and declares the logic variable associated with a cutpoint to have the value *False* (resp. *True*) if the attribute value is less than or equal to (resp. greater than) the cutpoint. The pattern analysis of [10] looks for cutpoints surrounded by patterns that are unlikely to have been produced by a certain random process. Though the reference does not use the term, we consider that random process to be an ARP. If attribute values are considered to be subject to random variations, then we use a second ARP and identify an interval around each cutpoint where we expect that values may randomly cross the cutpoint. We consider the values in that interval to be unusable and encode this fact by assigning to the associated logic variable the value *Unavailable*. Following [60], the same value is used to represent missing values, which by one of the assumptions of Section 1 cannot be obtained.

We pause for a moment and introduce the evaluation rule of [60] for DNF logic formulas when *True/False/Unavailable* values are assigned to variables. First, any DNF clause involving a variable having the value *Unavailable* is replaced by *False*. Then the reduced formula is evaluated in the customary way.

We justify the dual use of *Unavailable* as missing and unobtainable on one hand and as representing an unusable interval value on the other hand. According to the ARP, no value in the uncertainty interval has a reliable *True/False* encoding. Thus, the *True/False* value essentially cannot be decided, and we view this as an instance of a missing logic value that cannot be obtained.

It may seem that the introduction of an interval of uncertainty only moves the problem of randomly occurring contradictory encodings from the cutpoint to the endpoints of the interval. But in the former case, *True* may turn to *False* or *vice versa*, and thus a valid conclusion may become the opposite erroneous one. In the latter case, *True* or *False* may become *Unavailable* or *vice versa*. Under the consistency condition imposed on explanations, where at most one if-condition of the two explanations may evaluate to *True*, it can be proved that the latter change at best has no effect and at worst turns the conclusion “This record is in  $\mathcal{A}$ ” or “This record is in  $\mathcal{B}$ ” into “This record cannot be classified” or *vice versa*. Thus, the uncertainty intervals guarantee stability of explanations. For other, sometimes much more sophisticated treatments of missing values, see the reviews in [2, 65]. Here, we mention only that the main prior techniques are deletion of cases, imputation (= guessing missing values), and specialized handling as done, for example, in several decision tree methods.

We use a simple example to illustrate the pattern analysis. We also compare the method with entropy-based cutpoint selection and argue why we prefer our approach. Let example training sets  $A$  and  $B$  have just one integer attribute

$x$ . Set  $A$  has 8 records with values  $x = 0, 1, 2, 3, 4, 5, 6, 8$ , while set  $B$  has 7 records with values  $x = 7, 9, 10, 11, 12, 13, 14$ . Since all values  $x \leq 6$  occur in  $A$  records, while all values  $x \geq 9$  occur in  $B$  records, the two sets can be almost perfectly separated. The mixing of the two sets, so to speak, occurs with  $x = 7$  in a  $B$  record and  $x = 8$  in an  $A$  record. Thus, there are three reasonable choices for cutpoints:  $c_1 = 6.5$ ,  $c_2 = 7.5$ , and  $c_3 = 8.5$ . Suppose we confine ourselves to just one cutpoint. The pattern analysis of EXARP selects a cutpoint in the center of the region where the two sets  $A$  and  $B$  mix, and thus chooses  $c_2 = 7.5$ . In contrast, entropy minimization picks the cutpoint  $c_1 = 6.5$ , which lies at one end of the mixing region. Let’s change the data a bit at the tail ends of the sequences of integers. Specifically, we delete the  $x = 0$  record from set  $A$  and add an  $x = 15$  record to set  $B$ , getting revised training sets  $A'$  and  $B'$ , respectively. The change from  $A$  and  $B$  to  $A'$  and  $B'$  has virtually no impact on the pattern analysis of EXARP, and the same cutpoint  $c_2 = 7.5$  is selected. In contrast, the cutpoint  $c_1 = 6.5$  no longer minimizes entropy, but  $c_3 = 8.5$  does. The example demonstrates that the cutpoint selection via entropy is global in nature and may shift a cutpoint if a few values far from it are changed. On the other hand, the pattern analysis of EXARP is local and not affected by such changes. This stable behavior is one reason for our choice.

For discussion of a second reason, we ignore the case of  $A'$  and  $B'$  and focus on the sets  $A$  and  $B$ . We apply the following evaluation rule to the cutpoint  $c_2 = 7.5$  selected by EXARP and the cutpoint  $c_1 = 6.5$  minimizing entropy: Points below (resp. above) a cutpoint are declared to be in population  $\mathcal{A}$  (resp.  $\mathcal{B}$ ). Then it is easily checked that the center cutpoint  $c_2 = 7.5$  has accuracy 87% on the training data, while the entropy-selected cutpoint  $c_1 = 6.5$  produces a higher accuracy of 93%. This makes a case for entropy-selected cutpoints. But let us consider random changes of  $x$ . For that situation, the explanations implied by the cutpoints are not stable. As sketched above, EXARP eliminates that problem via an ARP that models the behavior of values  $x$  near the cutpoint. The analysis turns out to produce the uncertainty interval  $[7, 8]$  surrounding the cutpoint  $c_2 = 7.5$ . The uncertainty interval leads to the consistent and stable explanations “If  $x < 7$ , then in  $\mathcal{A}$ ” and “If  $x > 8$ , then in  $\mathcal{B}$ .” It so happens that the explanations have the previous accuracy of 87% on the training data. For the entropy case, the analysis of local uncertainty is more complicated because, intuitively speaking, entropy-based cutpoints are based on global and not local analysis. If we simplistically ignore this fact and, say, shift the interval  $[7, 8]$  surrounding  $c_2 = 7.5$  so that it becomes an uncertainty interval  $[6, 7]$  for  $c_1 = 6.5$ , then the use of the latter interval drops the accuracy on the training data to 80%, which is below the 87% accuracy attained with the interval  $[7, 8]$ . Thus, a more complicated approach is needed; for several approaches, see [31, 62], where uncertainty at decision tree nodes is treated. Here, we prefer the ARP-based interval selection, since it is conceptually consistent with the pattern analysis and has proved to be effective in the applications treated to-date.

We are ready to discuss details of the discretization step.

## 6.1 Selection of Cutpoints

We begin with a sketch of the method of [10], since otherwise the description of the modifications of that method is incomprehensible. The original scheme of [10] treats one attribute at a time, as follows. The numerical values of the attribute occurring in the training records of the sets  $A$  and  $B$  are sorted, and each value is labeled as coming from  $A$  or  $B$ . This step creates a *label sequence*, say of length  $N$ . If a value occurs only in  $A$  (resp.  $B$ ), then it is replaced by 1 (resp. 0). Values occurring in both  $A$  and  $B$  are replaced by the ratio of the number of cases occurring in  $A$  divided by the total number of occurrences of the value in  $A$  and  $B$ . Gaussian convolution is applied to the resulting sequence of values. A cutpoint is selected at the position where the smoothed values change by the largest amount. The cutpoint value is the average of the two neighboring original attribute values.

Suppose we view the cutpoint as a marker inserted into the original label sequence. Intuitively speaking, the cutpoint tends to have locally on one side mostly labels of one kind and locally on the other side mostly labels of the other kind or a mixture of labels.

The key for effectiveness of the method is an appropriate selection of the standard deviation  $\sigma$  of the Gaussian convolution process. For that decision, an ARP is formulated that creates sequences of  $A$  and  $B$  labels of length  $N$  by randomly selecting one a label at a time. The probability  $p$  of an  $A$  label is the number of  $A$  labels divided by  $N$ . The probability of a  $B$  label is  $q = 1 - p$ . We do not want to select cutpoints where the abrupt change in local pattern is producible by the ARP with reasonable probability. We achieve that goal indirectly by a standard deviation  $\sigma$  so that the Gaussian convolution obliterates all patterns that the ARP on average would produce at least once in sequences of length  $N$ .

The method becomes more complicated when additional cutpoints are to be selected, since then the method must also decide which attributes receive additional cutpoints. For the choice, a measure of *attractiveness* of cutpoints is defined that compares the abruptness of the value change at the cutpoint with that of an ideal pattern producing a very abrupt change of values. The attractiveness measure is multiplied with a *relevance count* that measures how much a cutpoint potentially may reduce a certain measure of separation of the two training sets. The cutpoint associated with the highest product is then introduced. The selection process repeats until the measure of separation has been reduced to 0 or no additional cutpoints can be found.

The reader interested in the mathematical details of the above steps should see [10].

The original method has significant shortcomings. First, it may try to introduce ever more cutpoints when full separation is not achievable at all or only with a large number of cutpoints. This mistake has been corrected by giving up on addition of cutpoints when the separation measure is not reduced by the most recently selected cutpoint. In addition, we severely limit the maximum number of cutpoints per attribute. To-date, we never have allowed more than two cutpoints per attribute. This limit differs significantly from the limit of 6 proposed in [10].

Second, the Gaussian convolution step effectively precludes correct evaluation of patterns near the lowest and highest attribute values. This causes problems when data are processed where important patterns may occur near extreme attribute values, as for example in protein chip data. For the remedy, we reduce the standard deviation  $\sigma$  of the Gaussian convolution step by two empirically derived rules that have proved to be effective. The first rule bounds  $\sigma$  by the minimum of  $\lceil Np/4 \rceil$  and  $\lceil Nq/4 \rceil$ . The second rule bounds  $\sigma$  by  $\lfloor \sqrt{Npq} \rfloor$ . As a result, patterns near the extreme values are correctly analyzed for possible cutpoints.

## 6.2 Selection of Uncertainty Intervals

We denote an arbitrary interval of uncertainty for a given attribute by  $U$ . Generally,  $U$  includes one cutpoint  $c$  that has already been determined as described in Section 6.1. We always define  $U$  via some subset of the values  $z$  of the attribute. That is,  $U$  is the smallest closed interval containing the selected  $z$  values. It is convenient that we call the number of attribute values defining  $U$  to be the *size* of  $U$ .

We define an ARP for the selection of  $U$ . For any positive number  $x$ , declare a *jump of size  $x$*  to be a change of the attribute value  $z$  so that it passes  $\lceil x \rceil$  other values. The ARP only considers jumps whose size is a multiple of  $\log_2 N$ , where  $N$  is the total number of records of the training sets  $A$  and  $B$ . We motivate that jump size later. The ARP postulates that the probability of a jump of size  $l \cdot \log_2 N$ , for  $l = 0, 1, 2, \dots$ , decays exponentially with increasing  $l$  and is equal to  $(\frac{1}{2})^{l+1}$ . Ignoring boundary effects, the probabilities sum to 1, since a jump can be to a smaller or larger value, and since  $\frac{1}{2} + \sum_{i=1}^{\infty} 2(\frac{1}{2})^{i+1} = \sum_{i=0}^{\infty} (\frac{1}{2})^i = 1$ .

Suppose we tentatively define  $\log_2 N$  to be the size of the interval  $U$ . We evaluate the effect using the ARP. Any jump of an attribute value  $z$  from below  $U$  to above  $U$  or *vice versa* changes the encoding from *False* to *True* or *vice versa*, and thus is undesirable. Since the interval  $U$  has size  $\log_2 N$ , such a jump of  $z$  near  $U$  must have size  $l \cdot \log_2 N$ , for some  $l \geq 2$ . The probability that this event occurs is approximately equal to  $\sum_{i=2}^{\infty} (\frac{1}{2})^{i+1} = (\frac{1}{2})^2 = 0.25$ . On the other hand, a jump of  $z$  from outside  $U$  into  $U$  may result in the undesirable conclusion "This record cannot be classified." The probability of such a jump depends on the distance of the value  $z$  from  $U$ . The largest probability occurs when  $z$  is within distance  $\log_2 N$  of the closest endpoint of  $U$ . The probability for that case is equal to  $(\frac{1}{2})^{l+1}$  with  $l = 1$ , and thus is  $(\frac{1}{2})^{1+1} = 0.25$ . Evidently, the selected size of  $U$  balances the probabilities of the two undesirable events. For this reason, we now choose that size for  $U$ . Of course, one could introduce cost considerations for erroneous classification versus no classification to refine the selection of  $U$ . But it seems that the probabilistic model employed here is too simple to support that refinement.

In the implementation, we use the standard deviation  $\sigma$  of Section 6.1 instead of  $\log_2 N$  for the size of  $U$ , since  $\sigma$  is roughly equal to  $\log_2 N$  when  $A$  and  $B$  are approximately of equal size, and since  $\sigma$  becomes larger when  $A$  and  $B$  differ substantially in cardinality and thus provides a concomitant increase of  $U$  that intuitively seems desirable.

Let  $u_1$  and  $u_2$  be the lower and upper bounds, respectively,

of  $U$ . Thus,  $U = [u_1, u_2]$ . We call  $u_2 - u_1$  the *width* of  $U$ . The computation of  $u_1$  and  $u_2$  is done via a parameter  $k$  so that approximately  $\sigma/2$  values below and  $\sigma/2$  values above the cutpoint fall into  $U$ . Specifically, if  $\sigma = 1$ , define  $k = 1$ , and if  $\sigma \geq 2$ , define  $k = \lfloor \sigma/2 \rfloor$ . Suppose the ordered attribute values in the  $A$  and  $B$  records are  $z_1 \leq z_2 \leq \dots \leq z_N$ , and assume that the cutpoint  $c$  has been determined as  $c = (z_{i^*} - z_{i^*+1})/2$ . Then  $u_1 = z_{i^*-k+1}$  and  $u_2 = z_{i^*+k}$  are the desired bounds.

Upon closer examination, the definition of the interval  $U$  of uncertainty via the jump size  $\log_2 N$  may seem unreasonable. Indeed, suppose we get additional records and have  $N' = 2N$  values for the attribute. An associated interval  $U'$  of uncertainty is defined by the jump size  $\log_2 N' = 1 + \log_2 N$ . Since the number of additional values in  $U$  roughly matches the number of original values in that interval, the width of  $U'$  may be equal to about half the width of  $U$ . Yet, a jump across  $U'$  is claimed to occur with the same probability, 0.25, as originally for  $U$ . On the surface, this seems to prove the ARP to be inappropriate. We argue that this is not so. When additional values become available, then the abrupt pattern changes at the selected cutpoints become more pronounced and more precise in the sense that they correspond more and more to significant facts separating  $A$  records from  $B$  records. Put differently,  $A$  or  $B$  values near the cutpoint have then a reduced probability of jumping across the cutpoint by a given number of values. The ARP acknowledges this fact in the definition of the probability for jumps. We have investigated this issue empirically. The tests confirmed that the choice of  $U$  via  $\sigma$  is reasonable.

If at least one of the sets  $A$  and  $B$  is very small, then the interval  $U$  likely assigns the value *Unavailable* to a significant percentage of the attribute values of the small set(s). To prevent that unfortunate event, we compute intervals of uncertainty as described above only if both  $A$  and  $B$  are not very small. In the implementation, we require that  $|A| \geq 6$  and  $|B| \geq 6$ .

Even when random changes are not postulated, we embed each cutpoint into a small interval of uncertainty that contains the cutpoint but no value of the training data. That choice eliminates the customary asymmetric treatment of cutpoints where, for example, values less than or equal to the cutpoint would be coded as *False* and values above the cutpoint as *True*. This concludes the discussion of intervals of uncertainty.

So far we have ignored nominal values, which generally are elements of finite sets. We use the discretization method for nominal values described in [10], where some cases involve an intermediate transformation to numerical values.

We should mention that the pattern analysis can be extended so that pairs of values of numerical attributes are considered in the Euclidean plane and certain regions of pairs are identified. The analysis also defines unusable regions, which analogously to the one-dimensional case are encoded by *Unavailable*. The outcome of the analysis is an additional numerical attribute that represents the regions and that is added to the training records of the sets  $A$  and

$B$ . Recursive application of the process and subsequent discretization of the additional attributes effectively achieves discretization of points in  $k$ -dimensional Euclidean space, for any  $k \geq 3$ . Details are provided in [53].

## 7. COMPUTATION OF FORMULAS

From now on we suppose that the training sets  $A$  and  $B$  contain only logic data. We call them *logic training sets*, for short. We find several separating DNF formulas for  $A$  and  $B$  with a scheme that is based on a subroutine of the induction method of [60]. The subroutine was proposed first, for classification purposes, in [25, 26]. The classification schemes of [35, 58, 59] are also related but use different solution techniques. Here, some of the DNF formulas are employed for the selection of important factors, while others are used in the explanations. The subsequent sections provide details of these roles.

The scheme uses all records, including those with *Unavailable* entries. We call the method BASIC LEARNING. Recall from Section 6 the evaluation of DNF formulas when variables may have the value *Unavailable*. That is, any clause with a variable having that value evaluates to *False*, and the correspondingly reduced formula evaluates as usual.

We sketch the construction of formulas by Algorithm BASIC LEARNING, since otherwise we cannot explain their roles in the subsequent sections. The method computes either 2 or 20 separating DNF formulas where each clause has a minimum number of literals. Later, the 20 formulas are used for the selection of important factors, and the 2 formulas are inserted into the if-conditions of the explanations.

In the case of 2 DNF formulas, one of them, called *Amin*, attains the value *True* on the records of  $A$ , while the other min formula, called *Bmin*, does so for the records of  $B$ . Each of the 2 DNF formulas is produced by the well-known sequential covering process that selects one clause at a time. Reference [58] calls it the OCAT (one clause at a time) method. For example, the first clause selected for *Amin* evaluates to *True* on the records of a maximum subset  $A'$  of  $A$  and to *False* on all records of  $B$ . Thus, the clause covers the maximum subset  $A'$ . The second clause covers the records of a maximum subset  $A''$  of  $A - A'$ . The process continues in this fashion until each record of  $A$  has been covered by a clause. The order in which the clauses are determined implies a *ranking of coverage* that is utilized in Section 8.

The case of 20 DNF formulas is a bit more complicated. The 20 formulas can be viewed as 10 sets of pairs of formulas. Each pair behaves as described above, except that the formulas are guaranteed to evaluate to *True* on just 60% of the records of  $A$  or  $B$ , whichever applies. Without risk of confusion, we also label the formulas of a pair as *Amin* and *Bmin*, being aware that the value *True* is guaranteed only for 60% instead of 100% of the relevant set. For the computation of the 20 DNF formulas, each of the sets  $A$  and  $B$  is divided into 10 subsets of essentially equal size. Let's call each of these subsets a *chunk*. The chunks of  $A$  and  $B$  are viewed as a circular file, and 10 times a subset of 6 consecutive chunks of  $A$  and the analogous subset of  $B$  are used to learn 2 DNF formulas. The 10 ways produce a total of 20 DNF formulas. The selection of the 6 consecutive chunks is

an optimal compromise between two opposing goals, where one goal desires training on largest subsets of  $A$  and  $B$ , while the second goal wants largest subsets not used in training and thus available for evaluation of the formulas on unseen data. Details are described in [25].

One may view the computation of the 20 formulas as a deterministic version of bagging [14] with 10 pairs of training subsets. But the 20 DNF formulas learned from the 10 pairs of subsets are put to different use. In bagging, the formulas would act as an ensemble and would produce a vote total for classification. As an aside, references [25, 26, 60] employ the above 20 formulas together with certain other formulas in that way. Here, each clause of the 20 DNF formulas is evaluated separately via an ARP to obtain an importance value. These values are then used for feature selection. Details are provided in Section 8.

It is possible that 2 or 20 logic formulas separating  $A$  and  $B$  as described above do not exist. When Algorithm BASIC LEARNING detects this situation, it outputs all pairs of  $A$  and  $B$  records causing this conclusion. The algorithm then deletes enough records so that the reduced sets can be separated. The deletions are so selected that the expected misclassification cost caused by them is minimized. For the decision, a misclassification cost of  $A$  records and the analogous cost for  $B$  records must be supplied. If the deletion process removes so many records that a reasonable separation is no longer possible, the algorithm outputs that information and stops. This is an early termination case where EXARP claims that explanations most likely are not contained in the data.

## 8. IMPORTANT FACTORS

For both supervised and unsupervised learning tasks, the problem of selecting attributes is called the *feature selection problem*. Existing feature selection methods for supervised learning tasks can be grouped into *filter* methods, *wrapper* methods, and *hybrid* methods combining both filter and wrapper approaches. Since the prior work is extensive, we cannot review it in detail. Instead, we give a terse summary and refer the reader to the reviews of [30, 41, 44] for details and references. *Filter* methods typically rely on statistical techniques and do not involve data mining algorithms. The technique essentially filters out unimportant features before the data mining process begins. *Wrapper* methods use the selected separation algorithm in the search for important features. The aim is a subset of features that will improve the mining algorithm’s performance. The technique tends to be more computationally expensive than filter methods due to the overhead incurred by the separation algorithm. *Hybrid* methods use both statistical measurements and data mining algorithms to determine important features. Common to virtually all methods is an iterative search for the desired subset of important features. In fact, the *generalized hybrid algorithm* defined in [44], which subsumes all prior methods, is of this form.

We propose a different approach where all important features are identified in one step. Key tools are: (1) 10  $A$ min and 10  $B$ min formulas learned from the training data by Algorithm BASIC LEARNING, and (2) an ARP whose action may make unimportant features look important.

We motivate the approach by an intuitive discussion. First, each of the 20 formulas is computed according to Section 7 from 60% of the training data. Thus, application of such a formula to all training data indicates how well it generalizes. Second, the bias of clause selection inherent in the sequential covering approach of Algorithm BASIC LEARNING is counterbalanced by the selection of the various 60% subsets. Thus, we may view the 20 formulas as 20 different ways to separate the set  $A$  from the set  $B$ , where for each formula we also find out how well it generalizes. Third, we use an ARP to judge how unusual each clause of the 20 formulas is, and estimate from that information the significance of the attributes.

We discuss a small example that illustrates the main ideas of the method. Suppose one clause  $C$  of one of the  $A$ min formulas is  $x \wedge \neg y$ . Let  $p_x$  (resp.  $p_{\neg y}$ ) be the ratio of the number of records of  $A \cup B$  where the attribute  $x$  (resp.  $y$ ) has the value *True* (resp. *False*), divided by the total number of records of  $A \cup B$ . Consider an ARP that randomly constructs  $A$  records. In particular, the ARP selects *True/False/Unavailable* values for attribute  $x$  and  $y$  so that  $x = \text{True}$  and  $y = \text{False}$  occur simultaneously with probability  $q = p_x p_{\neg y}$ . If the ARP manages to construct many records with values  $x = \text{True}$  and  $y = \text{False}$ , then the clause  $C$ , which evaluates to *True* for these records, will seem, but actually may not be, important. The number of records created by the ARP with the specified values for  $x$  and  $y$  follows the binomial distribution with parameters  $q$  and  $N = |A|$ . In particular, the mean value is  $qN$ , and the variance is  $q(1 - q)N$ . Suppose the set  $A$  actually has  $n$  records with the specified value pair. Using the normal distribution, we estimate the probability  $p_C$  that the ARP produces at most  $n$  records having the value pair  $x = \text{True}$  and  $y = \text{False}$ . If  $p_C$  is large, then we conclude that the ARP unlikely has produced the  $n$  cases. Accordingly, the clause  $C$  likely has identified a nonrandom feature of the  $A$  records and thus is important. For this reason, we declare  $p_C$  to be the *significance value* of the clause. In the actual computations, we do not use the count specified above for  $n$ , but instead let  $n$  be the number of records for which the clause  $C$  evaluates to *True* when applied in the ranking-of-coverage sequence defined in Section 7. Thus, a record with the specified value pair is not counted in  $n$  if and only if a higher-ranked clause already evaluates to *True* for that record.

In the general case, a clause  $C$  of an  $A$ min formula has any number of literals. For each literal  $z$ , we define  $p_z$  to be the relative frequency with which  $z$  evaluates to *True* in the records of  $A \cup B$ . For some integer  $k > 0$  specified momentarily, define  $l$  to be the minimum of  $k$  and the number of literals of clause  $C$ . Let  $q$  be the product of the  $l$  smallest  $p_z$  defined by the literals  $z$  of the clause  $C$ . We postulate that the ARP creates with probability  $q$  *True/False/Unavailable* entries in the records so that clause  $C$  evaluates to *True*. The remaining argument is as above, and it results in the significance value  $p_C$  for clause  $C$ .

It may seem simplistic that  $q$  is estimated as a product of  $l$  relative frequencies  $p_z$ . If we view the latter values as probabilities, then the process implies independence of the corresponding  $l$  attributes of the  $A$  records. On the surface, this assumption is quite severe. But the clause comes from

an Amin formula, and due to the implied minimization of clause size, we can hope that there is at least some degree of independence of the attributes corresponding to the selected  $l$   $p_z$  values. We still must define  $k$ . So far, we have found that  $k = 2$  works well and thus have used it in all applications.

With the above process, we obtain a significance value  $p_C$  for each clause  $C$  of each Amin formula. The analogous process with the set  $B$  and the Bmin formulas results in significance values  $p_C$  for each clause  $C$  of each Bmin formula. Once the significance values  $p_C$  are at hand for all clauses of all Amin and Bmin formulas, we select the important factors as follows. For each literal  $z$ , we compute the significance of  $z$  in a 4-step procedure. First, for each Amin formula  $F$ , we compute a significance value  $p_{F,z}$  for  $z$  relative to  $F$  as follows. If the literal  $z$  does not occur in any clause of formula  $F$ , then  $p_{F,z} = 0$ . Otherwise,  $p_{F,z}$  is equal to the largest  $p_C$  value of the clauses  $C$  of  $F$  containing the literal  $z$ . Second, we sum the  $p_{F,z}$  values of the 10 Amin formulas and divide that sum by 10, thus getting an average significance value for  $z$  relative to the 10 Amin formulas. Third, we carry out the above two steps once more, this time using the 10 Bmin formulas instead of the 10 Amin formulas, and get an average significance value for the literal  $z$  relative to the 10 Bmin formulas. Fourth, the larger of the two average significance values is declared to be the significance value of the literal  $z$ .

Suppose that, for given attribute  $x$ , the significance values for both literals  $z = x$  and  $z = \neg x$  are at hand. We define the significance of the attribute  $x$ , denoted by  $s_x$ , to be the maximum of the two significance values of the two literals. We consider  $s_x$  to be an estimate of the probability that attribute  $x$  is important for explaining the differences between  $A$  and  $B$ .

We define the attributes  $x$  for which  $s_x \geq 0.6$  to be the *important factors*. The threshold value 0.6 is based on a number of empirical tests where it proved to be a reliable and consistent criterion for importance. When at least one but not more than  $m$  attributes pass the threshold test, where  $m$  is very small, then we lower the threshold to a rounded 0.6 value, which effectively means that the attributes with  $s_x \geq 0.55$  are considered to be the important factors. To-date, we have almost always used the threshold value  $m = 2$ . If no important factors are determined, the algorithms outputs that information and stops. This is an early termination case.

We use the above rules whenever EXARP runs without supervision as described in Section 14. In interactive use of EXARP, we have allowed a reduction of the bound 0.60 and have labeled the resulting explanations as tentative. Except for rare cases, explanations so obtained have turned out to be useless.

The selection criteria for important factors are so severe that few attributes survive the test. As a result, when the selected attributes are used to compute 2 formulas Amin and Bmin for the explanations, then the sequential covering process of Algorithm BASIC LEARNING introduces at most a small bias into the clause selection.

## 9. PRIOR FACTS

Up to this point, we have assumed that the input of EXARP consists just of training sets and does not include prior facts. In this section, we remove that restriction and discuss the required modification of the algorithm when prior facts are supplied with the training sets. Thus, we have an instance of *analytical-inductive learning* [46], where incomplete domain knowledge and an incomplete set of training instances are used to construct hypotheses. On the other hand, this is not *explanation-based learning* [46], where a complete domain theory is used to construct explanations for a given set of training instances.

Various schemes have been proposed for analytical-inductive learning, for example, backpropagation, neural nets, and search; see [46] for example techniques. For the use of prior facts in Adaboost, see [55]. We describe an approach where the prior facts are represented by certain records that are added to the training data sets.

We require that each prior fact is an if-then statement that concludes membership in one of the populations. The if-condition is a DNF clause, with the format the same as for explanations. Here is an example.

“If  $(x > \alpha)$  and  $(y < \beta)$ , then the record is in population  $\mathcal{A}$ .”

We use the example to develop the method for inserting prior facts into the computing process of EXARP.

Let numerical training sets  $\tilde{A}$  and  $\tilde{B}$  be given. A correct representation of  $\tilde{A}$ ,  $\tilde{B}$ , and the above prior fact is achieved in the following 3-step process.

1. Discretize  $\tilde{A}$  and  $\tilde{B}$  as described in Section 6. This produces any number of cutpoints.
2. Add to the set of cutpoints for attribute  $x$  (resp.  $y$ ) the cutpoint  $c^* = \alpha$  (resp.  $d^* = \beta$ ). Discretize  $\tilde{A}$  and  $\tilde{B}$  again using the expanded list of cutpoints. Let  $A$  and  $B$  be the resulting logic training sets. Thus, each entry of any record of  $A$  or  $B$  is equal to *True*, *False*, or *Unavailable*.

We need some analysis before we can state the third step. Suppose that the cutpoints for attribute  $x$  (resp.  $y$ ) are  $c_1 < \dots < c_k = c^* < \dots < c_l$  (resp.  $d_1 < \dots < d_m = d^* < \dots < d_n$ ). We denote the cutpoints of the attributes  $z \neq x, y$  by  $e_{z,h}$ , with suitably defined indices  $h$ . As a matter of convenience, we use the cutpoint labels also as the logic attributes in the encoding of the logic training data.

Define  $\tilde{A}'$  to be the infinite set  $\tilde{A}' = \{\text{record } r \mid r_x > \alpha, r_y < \beta, r_z = \text{arbitrary}, z \neq x, y\}$ . Since all  $r_x$  and  $r_y$  cases for which the rule concludes membership in population  $\mathcal{A}$  are represented, the set  $\tilde{A}'$  correctly represents the rule. Suppose we discretize  $\tilde{A}'$  using the list of cutpoints defined in Step 2, getting  $A'$ . Evidently, the records of  $A'$  have the entries  $c_1 = \dots = c_k = c^* = \text{True}$ ,  $c_i = \text{True/False}$ ,  $k < i \leq l$ ,

$d_m = d^* = \dots = d^n = \text{False}$ ,  $d_j = \text{True/False}$ ,  $1 \leq j < m$ , plus *True/False* values for the cutpoints  $e_{z,h}$ , for all  $z$  and  $h$ .

By the derivation of the set  $A'$ , any two formulas  $A_{\min}$  and  $B_{\min}$  computed from the sets  $A \cup A'$  and  $B$  produce explanations that are in conformance with the prior fact. However, parts of the explanations may be unattractive or confusing when compared with the prior fact. This happens whenever a clause of  $A_{\min}$  uses *True/False* values of some  $c_i$ ,  $k < i \leq l$ , or  $d_j$ ,  $1 \leq j < m$ , or  $e_{z,h}$  to separate one or more records of  $A'$  from  $B$ . In that case, the prior fact is hidden in an incomprehensible clause of  $A_{\min}$ . We avoid such confusing  $A_{\min}$  clauses by declaring that none of the *True/False* values of  $c_i$ ,  $k < i \leq l$ , or  $d_j$ ,  $1 \leq j < m$ , or  $e_{z,h}$  may be used to achieve the separation of  $A'$  from  $B$ . We claim that the corresponding restriction on  $B_{\min}$  for the separation of  $B$  from  $A'$  is trivially satisfied. Indeed, since each clause of  $B_{\min}$  evaluates to *False* for all records of  $A'$ , and since any  $c_i$ ,  $k < i \leq l$ , or  $d_j$ ,  $1 \leq j < m$ , or  $e_{z,h}$  has *True* as well as *False* in the records of  $A'$ , any literals in the clause defined from these variables cannot evaluate to *False* for all records of  $A'$ . Thus, each clause must have at least one literal of the variables  $c_i$ ,  $1 \leq i \leq k$ , or  $d_j$ ,  $m \leq j \leq n$ , where the literal evaluates to *False* due to the value *True* of  $c_i$  or *False* of  $d_j$ .

Relative to the evaluation of  $A_{\min}$  and  $B_{\min}$ , and for any logic attribute  $w$ , the statement  $S_1 =$  “The *True/False* value of  $w$  is known but cannot be used for the evaluation of formulas” imposes the same restriction as the statement  $S_2 =$  “The *True/False* value of  $w$  is unknown but can be obtained.” Indeed, both statements effectively say that the *True/False* of  $w$  exists but for some reason cannot be used for the evaluation of formulas. In [60], the situation of  $S_2$  is encoded by assigning the value *Absent* to the attribute  $w$ . Accordingly, we replace in each record of  $A'$  each *True/False* value of  $c_i$ ,  $k < i \leq l$ , and  $d_j$ ,  $1 \leq j < m$ , and  $e_{z,h}$ , all  $z$  and  $h$ , by *Absent*. The change converts each record of  $A'$  to the record with  $c_1 = \dots = c_k = c^* = \text{True}$  and  $d_m = d^* = \dots = d^n = \text{False}$ , and with *Absent* assigned to all other logic attributes. We call that record the *truncated discretization record*, for short *truncated record*, of the prior fact.

Algorithm BASIC LEARNING can handle records with *True/False/Absent/Unavailable* entries and thus can deal with the truncated record. Details are provided in [60]. We only mention that the algorithm relies on the following evaluation rule for DNF formulas when the entries are *True/False/Absent/Unavailable*. The formula has the value *True* if there is a clause where all variables have *True/False* values such that the clause evaluates to *True* with these values. The formula has the value *False* if for each clause, there is a variable with value *Unavailable*, or with a *True/False* value such the literal of the variable evaluates to *False*. In the remaining case, the formula has the value *Undecided*.

Suppose we add the truncated record to the logic training set  $A$  and apply Algorithm BASIC LEARNING to compute formulas  $A_{\min}$  and  $B_{\min}$  separating the expanded set  $A$  and  $B$ . By the above construction of the truncated record, the formulas are comprehensible relative to the prior fact, in

the sense that we can tell by inspection how the prior fact manifests itself in the explanations. Specifically, the formula  $A_{\min}$  must have at least one clause whose literals form a subset of the literals  $c_i$ ,  $1 \leq i \leq k$ , and  $\neg d_j$ ,  $m \leq j \leq n$ . Generally, the clause is at least as strong as the prior fact, and any strengthening is made possible by the training sets  $\tilde{A}$  and  $\tilde{B}$ . Moreover, each clause of  $B_{\min}$  has at least one literal of the form  $\neg c_i$ ,  $1 \leq i \leq k$ , or  $d_j$ ,  $m \leq j \leq n$ . These literals imply limits of possible strengthening of the prior fact, based on the training sets  $\tilde{A}$  and  $\tilde{B}$ .

We are ready to list the third step of the algorithm handling the prior fact.

3. Derive the truncated record of the prior fact, and add it to the logic training set  $A$ . Run Algorithm BASIC LEARNING to compute  $A_{\min}$  and  $B_{\min}$ . The resulting formulas  $A_{\min}$  and  $B_{\min}$  are in conformance with the prior fact.

Suppose EXARP, using the above steps, computes consistent explanations for the enlarged set  $A$  and  $B$ . Then the explanations correctly classify all training instances of  $\tilde{A}$ ,  $\tilde{B}$  as well as any instance satisfying the prior fact. Of course, other outcomes are possible. In particular, EXARP may detect a conflict between  $\tilde{A}$ ,  $\tilde{B}$ , and the truncated record. EXARP has a diagnostic subsystem that localizes the problem and supports corrective action.

The above representation typically induces instability into the explanations with respect to the prior fact. To eliminate that effect, we embed the cutpoints  $c^*$  and  $d^*$  into suitably selected uncertainty intervals. The selection of these intervals is not based on the pattern analysis of Section 6, but rather reflects the precision with which the prior fact holds. For example, suppose  $x$  represents the age of a patient in years. Now on the birthdate, the age jumps by 1. Assume that the boundary value  $\alpha$  for the inequality involving  $x$  is integer. Then the uncertainty interval is  $[\alpha, \alpha]$ . Since the precision with which attribute values are known can be established prior to the execution of EXARP, we demand that information whenever a prior fact is specified. As a result, the explanations are stable with respect to the prior fact.

The construction of the additional cutpoints, uncertainty intervals, and the truncated record is readily adapted when any number of inequalities or equations occur as literals in the prior fact, or when membership in population  $\mathcal{B}$  instead of  $\mathcal{A}$  is concluded. We omit the straightforward details and mention only that, if the prior fact concludes membership in population  $\mathcal{A}$  (resp.  $\mathcal{B}$ ), then the corresponding truncated record is added to  $A$  (resp.  $B$ ). When a number of prior facts are available, then each fact produces a truncated record. The rules are adapted in the expected way when Algorithm BASIC LEARNING computes 20 formulas instead of just 2 formulas. We then add each truncated record to each 60% subset of  $A$  or  $B$ . Thus, all truncated records participate in each of the 20 cases.

We should mention that the above evaluation rule for *True/False/Absent/Unavailable* entries is useful when explanations are to be evaluated with partial knowledge where

numerical values are known for some attributes, while for the remaining attributes a value cannot be obtained or is presently unknown but can be obtained. The latter cases are encoded by *Unavailable* or *Absent*, respectively. The evaluation of the if-conditions of consistent explanations either produces a conclusion of membership in one of the two populations, or it declares the situation to be undecided. In the latter case, solution of a certain problem called Q-ALL SAT [60] tells whether additional numerical values replacing the *Absent* entries can possibly lead to a conclusion of membership. Despite the fact that Q-ALL SAT is at the second level of the polynomial hierarchy, an effective solution algorithm is available [52]. Evaluation via Q-ALL SAT is important in diagnostic systems where, due to cost considerations, information is obtained in a stepwise fashion until either the case is settled or it is discovered via solution of a Q-ALL SAT instance that a diagnosis is impossible.

We conclude the section with a simple example case. A risk profile for possible heart disease is to be computed from a very small data set of patients with chest pain complaints. EXARP says, via the significance values of attributes, that the data do not have enough information for the desired explanations. We force EXARP to go on, since this is a demonstration project that is to motivate a larger study, and since these data are all we can get at this point. EXARP computes the following explanations 1 and 2, where each if-condition corresponds to one DNF clause. We purposely have not simplified explanation 2 to simplify the comparison with an expanded version given later.

1. If the patient takes medication for cholesterol control, or if age  $\geq 52$  years and systolic blood pressure  $\geq 134$ , then heart disease must be considered.
2. If the patient does not take medication for cholesterol control and age  $\leq 47$  years, or if the patient does not take medication for cholesterol control and systolic blood pressure  $\leq 124$ , then heart disease need not be considered.

The explanations are consistent and stable. The expert declares the explanations to be somewhat reasonable. “But,” he says, “a patient of age 70 who does not take cholesterol medication and has systolic blood pressure  $\leq 124$  is classified by explanation 2 as ‘heart disease need not be considered,’ which no physician would ever agree to.” Actually, the data observe this rule, but the explanations do not display any version of it. We should not fault EXARP for not detecting this. The method said at the outset that the data do not suffice for reliable explanations.

We settle on the prior fact “If age  $> 67$ , then heart disease must be considered.” When this prior fact is added to the training data as described above, with uncertainty interval  $U = [67, 67]$ , the following explanations results.

- 1'. If the patient takes medication for cholesterol control, or if age  $\geq 52$  years and systolic blood pressure  $\geq 134$ , or if age  $\geq 68$  years, then heart disease must be considered.

- 2'. If the patient does not take medication for cholesterol control and age  $\leq 47$  years, or if the patient does not take medication for cholesterol control and systolic blood pressure  $\leq 124$  and age  $\leq 66$  years, then heart disease need not be considered.

In explanation 1, the prior fact is explicitly stated by the condition “age  $\geq 68$ ”. In explanation 2, the prior fact is implicitly accounted for by “age  $\leq 47$ ” in the first if-statement, and explicitly by “age  $\leq 66$ ” in the second one. The condition “age  $\leq 47$ ” implies a limit on possible strengthening of the prior fact when cholesterol medication is not taken.

The expert declares the revised explanations 1 and 2 to be sufficiently interesting and reasonable to motivate the larger study.

## 10. CONSTRUCTION OF EXPLANATIONS

We use the above schemes as follows to construct explanations. We discretize the training data, represent prior facts by truncated records, compute important factors, restrict the training data to the important factors, and finally compute two DNF formulas  $A_{min}$  and  $B_{min}$  for the if-conditions of the desired two explanations.

## 11. VALIDATION

We validate the explanations in a 2-step or, optionally, 3-step process.

### 11.1 Consistency

In the first validation step, we check if the two explanations are consistent. Recall that this property requires that the two if-conditions of the explanations cannot evaluate simultaneously to *True*. We confirm this by solving an instance of the propositional satisfiability problem SAT that involves the two if-conditions and certain logic clauses implied by the discretization step and described below. The two explanations are consistent if and only if the SAT instance is unsatisfiable.

We describe the logic clauses implied by the discretization step. Recall that a DNF clause evaluates to *False* if any variable has the value *Unavailable*. A simple argument using this fact establishes that we only need to consider satisfiability of SAT instances where all logic variables of the two if-conditions have *True/False* values. Thus, the case of *Unavailable* assignments can be ignored.

We consider the two basic discretization cases. In the first situation,  $k$  cutpoints have been introduced for a numerical attribute. If  $k = 1$ , no additional conditions are needed. If  $k \geq 2$ , let  $y_1, y_2, \dots, y_k$  be the logic variables corresponding to the cutpoints in sorted ascending order. In any encoding of a numerical value, either all  $y_i$  have the value *False*, or, for some  $j$ , all  $y_i, i < j$ , have the value *False*, and the remaining  $y_i$  have the value *True*. The clauses  $\neg y_{i-1} \vee y_i, 1 < i \leq k$ , constrain the *True/False* assignments exactly to these cases and thus are added to the SAT instance. In the second case, a nominal attribute has been encoded by  $l \geq 2$  logic variables  $z_1, z_2, \dots, z_l$ . Then always exactly one of the variables has the value *True*. The clauses  $y_1 \vee y_2 \vee \dots \vee y_l$  and  $\neg y_i \vee \neg y_j, 1 \leq i < j \leq l$ , restrict the *True/False* assignments

to these cases and thus are added to the SAT instance. As an aside, if  $l$  is large, then a more compact representation of linear complexity is possible; see [60] for details.

### Remedy of Some Inconsistent Cases

Suppose the SAT instance is satisfiable, so the if-conditions are inconsistent. We check if that negative outcome can be avoided by a change of the  $A_{min}$  and  $B_{min}$  clauses making up the two explanations. This option exists since Algorithm BASIC LEARNING of Section 7 not only produces two minimal  $A_{min}$  and  $B_{min}$  formulas for the explanations, but also generates companion formulas that are obtained from  $A_{min}$  and  $B_{min}$  by adding a maximum number of literals to the clauses under the requirement that the resulting formulas, called  $A_{max}$  and  $B_{max}$ , perform identically to  $A_{min}$  and  $B_{min}$  on the training data. By the derivation of  $A_{max}$  and  $B_{max}$ , any formulas  $A_{bet}$  and  $B_{bet}$  derived from  $A_{max}$  and  $B_{max}$  by deleting literals not occurring in  $A_{min}$  and  $B_{min}$ , perform identically to  $A_{min}$  and  $B_{min}$  on the training data. We use this fact next.

Still assume that  $A_{min}$  and  $B_{min}$  produce inconsistent explanations. If  $A_{max}$  and  $B_{max}$  also result in inconsistency, then we declare the inconsistency to be *unfixable* and reject the explanations. Otherwise, we determine  $A_{bet}$  and  $B_{bet}$  formulas that are consistent and contain a minimum number of additional literals beyond those of  $A_{min}$  and  $B_{min}$ . In the typical case, this is a simple problem solved by inspection. But in general, this is a potentially very difficult optimization version of a problem at the second level of the polynomial hierarchy. In [60], the optimization problem is called Q-MINFIX UNSAT. The reference supplies a fast heuristic that has proved to be effective [57].

### Unfixable Inconsistent Cases

If the explanations based on  $A_{min}$  and  $B_{min}$  are inconsistent, and if the inconsistency is unfixable, then we reject the explanations. Of course, the computation of the DNF formulas  $A_{min}$  and  $B_{min}$  described in Section 7 could be modified so that consistency of formulas is guaranteed. We prefer the present approach where consistency is checked as part of the validation step, since in all applications to-date, we have had unfixable inconsistency only when the explanations were not trustworthy or wrong. Thus, we consider unfixable inconsistency to be an important criterion for the rejection of explanations.

### 11.2 Accuracy

In the second validation step, we apply the explanations to the testing data and evaluate the accuracy using standard statistical methods. The statistical significance should be at or very close to the gold standard  $p < 0.0001$ . If this is not the case, we reject the explanations. We note that verification via schemes such as leave-one-out or  $k$ -fold cross-validation may not be acceptable, as we learned first-hand while working with researchers in medical areas.

### 11.3 Expert Evaluation

Suppose we have determined a high level of significance for the explanations. In an optional third step, one or more domain experts evaluate the explanations in a process consistent with practice of the domain.

In each case of early termination or rejection of the explanations in one of the first two validation steps, the data may also be discussed with domain experts. If they say that the existence of explanations seems very unlikely, then we consider the negative conclusion by EXARP to be validated by the experts.

Until recently, we carried out expert validation on all applications. During that time, EXARP never produced explanations that an expert declared to be outright incorrect. Furthermore, whenever EXARP declared likely nonexistence of explanations, then a domain expert confirmed that existence of explanations was highly speculative and that most likely there was none. This does not mean that explanations by EXARP will always be valid, or that absence of explanations is always correctly claimed. But it indicates that EXARP is fairly reliable. For this reason, we now consider expert validation to be optional.

There is another good reason for not mandating expert validation of explanations or of conclusions of their nonexistence. In ongoing research, EXARP is used as a search tool in systems that in one run process a large number of training sets. The search is akin to looking for a few needles in a haystack. Thus, in the preponderance of cases, EXARP claims that there is no explanation. Furthermore, the explanations produced by EXARP constitute intermediate information for additional processing steps. In that setting, expert validation of the voluminous EXARP output is, practically speaking, not possible.

## 12. EXPLANATION OF EFFECTS

Up to this point, we have assumed that the training sets  $A$  and  $B$  are given. In certain situations, we have instead a set  $S$  of records, each with an associated effect  $e$ . For example, the effect  $e$  may be the time-to-progression (TTP) of a disease. In such cases, one may want to learn logic formulas that explain why the effect is low or high.

### 12.1 Imprecise Effects

In the general case, the effect  $e$  is not known exactly. Instead, an *effect interval*  $E$  is given that contains the actual value. We assume, reasonably, that  $E$  is an open interval of the form  $(-\infty, e_1)$ ,  $(e_1, e_2)$ , or  $(e_2, \infty)$ , for finite  $e_1$  and  $e_2$ . Moreover, for the definition of low and high effects, a closed *threshold interval*  $[d_1, d_2]$  with finite  $d_1$  and  $d_2$  is given. If an effect interval  $E$  is contained in the open interval  $(-\infty, d_1)$  (resp.  $(d_2, \infty)$ ), then the effect is considered to be low (resp. high). For the explanation of low and high effects, we place into a set  $A$  (resp.  $B$ ) the records of the set  $S$  that according to the preceding definition have low (resp. high) effect. We then proceed as described in the preceding sections. The records of  $S$  whose effect intervals  $E$  do not fall into  $(-\infty, d_1)$  or  $(d_2, \infty)$  are ignored. A more complicated use of effect intervals is discussed next.

### 12.2 Importance Functions for Effects

Suppose we have a set  $S$  of records with associated effect intervals as before, but we are not given a threshold interval as assumed in Section 12.1. Nevertheless, we want to explain why the effect is low or high. We obtain the desired explanations via *importance functions*, as follows. Let

$e_{\min}$  (resp.  $e_{\max}$ ) be the minimum (resp. maximum) of the finite endpoint values specifying the effect intervals. We partition the interval  $[e_{\min}, e_{\max}]$  into 10 evenly sized subintervals with successive endpoints given by  $f_0 = e_{\min}$  and  $f_i = f_{i-1} + (e_{\max} - e_{\min})/10$ , for  $i = 1, 2, \dots, 10$ . For  $i = 2, 3, \dots, 6$ , we combine these subintervals by taking two successive intervals at a time, thus getting the intervals  $[f_i, f_{i+2}]$ . We view the latter intervals as threshold intervals and carry out the earlier described analysis. Thus, for each interval  $[f_i, f_{i+2}]$  and each attribute  $x$ , we obtain a significance value  $s_x^i$ . For each attribute  $x$ , we define the *importance function*  $F_x$  to be the function that, for  $i = 2, 3, \dots, 6$ , maps the index  $i$  to the significance value  $s_x^i$ .

Whenever  $F_x(i) \geq 0.6$ , then attribute  $x$  is considered important for telling *low effects* falling into the interval  $(-\infty, f_i)$  from *high effects* falling into the interval  $(f_{i+2}, \infty)$ . This information, by itself, can be useful. For a detailed analysis of the low/high effects, we take the threshold intervals  $[f_i, f_{i+2}]$  for which at least one  $F_x(i) \geq 0.6$ , and find explanations for the corresponding training sets  $A^i$  and  $B^i$ . We may further narrow the selection of the threshold intervals  $[f_i, f_{i+2}]$  by using large  $\sum_x F_x(i)$  values as secondary criterion. If no threshold interval satisfies the above criteria, then the algorithm states that explanations of the effect very likely cannot be derived from the given data, and stops. This is yet another early termination case.

### 13. OVERFITTING

We examine the well-known issue of overfitting, which occurs when a formula represents the training data well but performs worse than another, simpler formula on the entire population. In EXARP, we avoid overfitting as follows. First, we use Gaussian convolution in the discretization step to eliminate structures that superficially look significant but may actually be the result of random effects. Second, we define intervals of uncertainty around cutpoints to lessen the detrimental impact of random variations of attribute values near cutpoints. Third, we remove attributes that likely have little or no significance for separating formulas. Fourth, we search for DNF formulas whose clauses have a minimum number of literals. The first three parts are handled by the analysis of ARPs. The fourth part is handled by logic minimization. Taken together, these steps have proved to be fairly effective tools for avoiding overfitting. To be sure, formulas that do not overfit the data may nevertheless be inconsistent or untrustworthy. Thus, avoidance of overfitting, by itself, does not guarantee that explanations are valid.

### 14. MANUAL GUIDANCE

EXARP can run without manual interference or guidance once the following information is supplied for each attribute of the training/test records: whether the attribute is nominal or numerical; in the nominal case, whether the attribute values have an implied linear ordering; in the numerical or linearly-ordered nominal case, whether random changes should be assumed. A possible part of the input are prior facts, including uncertainty intervals for bounding constants. The intervals reflect the precision with which attribute values are measured. Finally, estimates of the misclassification cost for  $A$  records and the analogous cost for  $B$  records are needed.

## 15. COMPLEXITY OF EXARP

The most complicated step of EXARP outside validation involves the solution of certain linear programming (LP) instances in Algorithm BASIC LEARNING. In the validation step, a SAT instance must be solved to check consistency of the explanations, and a Q-MINFIX UNSAT instance must possibly be solved in the remedial step for inconsistency. The LP instances are solvable in polynomial time [37]. But the SAT problem is  $\mathcal{NP}$ -complete, and Q-MINFIX UNSAT is an optimization version of a problem at the second level of the polynomial hierarchy. Except for the validation step, a suitable implementation of EXARP is guaranteed to be polynomial. In our implementation, we solve the LP instances with the simplex method, which is not polynomial but solves reasonably sized instances rapidly and reliably. In practice, the SAT instances are very small and are readily solved by standard techniques. The Q-MINFIX UNSAT instances can be solved approximately by the fast heuristic of [60]. As a result, EXARP is fast on reasonably sized instances. For example, on a PC with 2.8GHz processor, training sets with several hundred records are handled in less than 10 sec.

## 16. EXAMPLE APPLICATIONS

We describe the results of EXARP for three real-world example applications. Each time, the algorithm is used without manual interference or guidance. For comparison purposes, we also apply the decision tree method C4.5 [51] and the rule construction method RIPPER [17] to the examples. We use the C4.5 and RIPPER implementations supplied by the University of Waikato on the Weka website. Both C4.5 and RIPPER are run with the default settings. The EXARP results for the real-world applications have also been examined by domain experts, and we summarize their evaluation.

We should insert a comment about the difficulty of obtaining data of real-world applications and having results evaluated by experts. There are many data sets for machine learning, for example, those of the UC Irvine Repository, that have been extensively used for comparing induction methods. For some of these data sets, obtaining explanations of the form demanded here is irrelevant or useless, while for the rest we do not have access to experts who would be agreeable to help carry out an evaluation of explanations. So instead, we present the results for three applications of our research program where experts were available. In each of the three application cases, we were given the data without interpretation or hypotheses, and were simply asked to apply EXARP and report the resulting explanations. In each case, we were not provided, and did not search for, any information about the nature of the attributes and their meaning. In addition, we applied the standard version of EXARP and performed no manual guidance. Thus, the three cases represent a blind test of EXARP running without supervision. Once the explanations had been obtained and forwarded to a domain expert for evaluation, we did gain insight into the underlying processes by the feedback. Finally, we applied C4.5 and RIPPER to the data sets for comparison purposes.

The comparison of results for three cases is meant to be illuminating and obviously involves too small a sample to support conclusions. Presently, EXARP is used in several ongoing projects where expert validation is available, and

over the next several years a reasonably accurate performance profile should emerge.

## 16.1 Charité Cervical Cancer Data

The data set was supplied by the Frauenklinik of the Charité, Berlin, Germany. The data cover 57 patients. We split them randomly into 31 for training and 26 for testing, guessing in advance that 26 testing cases will suffice to achieve statistical significance. This guess is based on the tail probabilities of the binomial distribution with  $p = 0.5$  and on the optimistic assumption that the explanations will be quite accurate. The 31 training cases are made up of 19 cervical cancer cases of type FIGO I-III, and of 12 cases of cervical cancer recurrence. The 26 testing cases consist of 14 FIGO I-III cases and 12 recurrence cases. The records have 14 attributes covering various lab tests. Each attribute is numerical, and the values are subject to random variations.

Application of the decision tree method C4.5 and subsequent conversion of root-to-leaf paths produces the following two explanations.

1. If  $[\text{ENDOSTATIN} \leq 124]$  or  $[124 < \text{ENDOSTATIN} \leq 156 \text{ and } \text{VEGFD} \leq 254]$ , then FIGO I-III case.
2. If  $[\text{ENDOSTATIN} > 156]$  or  $[\text{ENDOSTATIN} > 124 \text{ and } \text{VEGFD} > 254]$ , then recurrence case.

Recall that two explanations concluding membership in population  $\mathcal{A}$  and  $\mathcal{B}$ , respectively, are *consistent* if, for any record, at most one of the two if-conditions of the two explanations can evaluate to *True*. The two explanations are *stable* if small random changes of the data do not flip one conclusion to the opposite one.

The above explanations are consistent. Consider a case of  $\text{ENDOSTATIN} = 156$  and  $\text{VEGFD} \leq 254$ , which would be classified by explanation 1 as a FIGO I-III case. A small change of the  $\text{ENDOSTATIN}$  value, say to 157, would cause explanation 2 to conclude a recurrence case. Thus, the explanations are not stable. The two explanations predict 19 of the 26 test cases correctly, for an accuracy of 73%. This corresponds to a significance of  $p < 0.18$ , which causes rejection of the explanations.

RIPPER produces for the data set the following explanations:

1. If  $\text{ENDOSTATIN} \geq 129.0$ , then recurrence case.
2. Else FIGO I-III case.

When we reverse the roles of the training sets  $A$  and  $B$  and apply RIPPER again, the same explanations result. This need not be the case, as we shall see later. At that time, we discuss that aspect in detail. Applied to the 26 test cases, the explanations classify 21 cases correctly, which equates to 80.7% accuracy. The statistical significance is  $p < 0.002$ .

The two explanations are consistent. We have asked an expert to evaluate them. To get an unbiased response, we did

not tell the expert the source of the explanations and simply said that we had computed them. The expert stated that the explanations may be useful, but also expressed concern that the explanations flip to the opposite conclusions when an  $\text{ENDOSTATIN}$  value of 128.0 changes to 129.0 or *vice versa*. In the terminology used here, the expert is concerned about instability of the two explanations.

When EXARP is applied to the data, it first determines the following important factors.

### Important Factors for Charité Cervical Cancer Data

Attribute	Significance Value
ENDOSTATIN	0.79
ENDOGLIN	0.79
M2PK_PLASMA	0.75

Here are the explanations.

1. If  $\text{ENDOSTATIN} < 123.0$  or  $\text{M2PK_PLASMA} < 18.8$ , then FIGO I-III case.
2. If  $\text{ENDOSTATIN} > 136.0$  and  $\text{M2PK_PLASMA} > 21.8$ , then recurrence case.

The two explanations are consistent and stable. Applied to the 26 test cases, the explanations classify 22 cases correctly, which equates to 85% accuracy. The statistical significance is  $p < 0.0002$ , which is almost the gold standard  $p < 0.0001$  of medical research. A domain expert has declared the explanations to be interesting and useful. Unfortunately, it is not known at this time how the insight produced by the explanations can be used for more effective treatment.

Summary: C4.5 produces consistent and moderately accurate explanations that are unstable and rejected due to a significance  $p < 0.18$ . RIPPER generates consistent and quite accurate explanations that, though unstable, may be useful. EXARP generates consistent and stable explanations that have good accuracy and may be useful.

## 16.2 Bochum Breast Cancer Data

This data set was supplied by the Frauenklinik, Universität Bochum, Germany. Data are given for 14 patients. Each record has values for 35 attributes covering various clinical, lab, and treatment results. Of the 35 attributes, 7 are numerical and can be expected to vary randomly. The remaining attributes are nominal. The records also have various missing entries. Each patient has already been treated for breast cancer, and breast cancer has recurred or metastases have been found. The study was carried out to see if palliative treatment with the two drugs Herceptin and Xeloda can help in these rather desperate situations. Three of the 14 patients achieve astonishingly large time-to-progression (TTP) values, which measure the interval from the beginning of the Herceptin/Xeloda treatment to the point where the cancer progresses again. Explanations are to be found for this effect. Let  $\text{TP\_TISSUE}$  denote the level of Thymidine Phosphorylase in cancerous tissue at the time of the original surgery.

The method C4.5 computes the following two explanations, which are consistent but not stable.

1. If liver metastases occur and if  $TP\_TISSUE > 4$ , then the patient has high TTP.
2. If liver metastases do not occur, or if liver metastases occur and  $TP\_TISSUE \leq 4$ , then the patient has low TTP.

One implication of explanation 2 is that absence of liver metastases triggers low TTP. That odd implication does not agree at all with current understanding of the interaction of Herceptin and Xeloda with cell processes. Moreover, even though there are just 14 records in total, the accuracy of the above explanations on the training data is 93%. Taken together, these facts show that the explanations most likely are wrong.

RIPPER computes no explanations for the data, or rather it declares each case to have low TTP. The same conclusion results when we reverse the roles of the training sets  $A$  and  $B$ . The accuracy implied by this choice is 79%. Needless to say, this output is not useful.

EXARP determines the following important factors, where COX2 is the Cyclooxygenase-2 value and K18 is the Keratin-18 value.

Important Factors for Bochum Breast Cancer Data

Attribute	Significance Value
TP_TISSUE	0.70
COX2	0.60
K18	0.57

The two explanations are:

1. If  $TP\_TISSUE \geq 6$ , and if  $COX2 \leq 2$  or  $K18 \geq 9$ , then TTP is high.
2. If  $TP\_TISSUE \leq 4$ , or if  $COX2 \geq 4$  and  $K18 \leq 8$ , then TTP is low.

The explanations are consistent and correct for 100% of the training data. At present, no additional testing data are available, and splitting of the already very small data set into training and testing data is unreasonable. Thus, the explanations should be treated as consistent hypotheses. Note that the condition  $TP\_TISSUE \geq 6$  of explanation 1 and the related condition  $TP\_TISSUE \leq 4$  of explanation 2 introduce stability with respect to the  $TP\_TISSUE$  value. The same effect is evident for the two conditions  $COX2 \leq 2$  and  $COX2 \geq 4$ . However, we also have the unstable conditions  $K18 \geq 9$  and  $K18 \leq 8$ .

A domain expert has declared that the explanations are in full agreement with knowledge of the interaction of Xeloda with cell processes, and that they may very well be correct.

Indeed, the expert believes that the explanations point to an exciting new treatment option. At the time of this writing (early 2007), an effort has been started for a clinical test of the new treatment for patients where primary and palliative treatments have failed. If that test is successful, then a substantial number of patients can be saved.

Summary: C4.5 generates consistent, unstable explanations that most likely are wrong. RIPPER does not produce usable explanations. EXARP generates consistent and partially stable explanations that may very well be correct.

## 16.3 Munich Dementia Data

This data set was supplied by the Technische Universität München, Germany. The records contain data about conjectured predictors for dementia. Explanations for near-term onset of dementia are to be found. There are 230 training records and 227 testing records. The slightly uneven split results from an even split of the original patient files, since each file may produce one or more records. Each record has a total of 29 attributes, of which 5 are nominal and the remaining 24 are numerical. Of the latter attributes, 16 are considered subject to random changes.

At the outset, EXARP is used as described in Section 12.2 to determine an effect interval and thus a numerical value of “near-term” for which useful explanations are likely to exist. We omit details and mention only that the resulting set  $A$  contains the cases where dementia occurred within 660 days of the day where record data were obtained, while set  $B$  has the cases where at least 940 days elapsed before dementia was detected, if at all. Below we refer to the cases of set  $A$  as *near-term dementia* and to those of set  $B$  as *late/no dementia*.

C4.5 produces two consistent explanations consisting of 6 and 9 rules, respectively. The largest rule has 7 terms. The rules are too complex to be included here, and they seem contrived. Also, they are not stable. The accuracy on the testing data is 84% and turns out to be highest for the three methods.

RIPPER computes two sets of rules; one for the sets  $A$  and  $B$ , and the second one for the case where the roles of  $A$  and  $B$  are reversed. The two rule sets are not consistent, and we elect to use each of the rules to construct two explanations in the obvious way. Indeed, the RIPPER output lists the rules in that form. In the first case, we get the following explanations.

1. If  $SKTSCORE \geq 8$ , or if  $CDRWERT \geq 1$  and  $SKT8RW \geq 8$ , or if  $SKT4RW \geq 26$  and  $SKT3RW \leq 8$  and  $SKT5RW \geq 23$ , then near-term dementia.
2. Else late/no dementia.

The second set of rules produces the following two explanations.

3. If  $CDRWERT \geq 1$  and  $SKT9RW \geq 2$  and  $MMS \leq 26$ , or if  $CDRWERT \geq 1$  and  $SKT9RW \geq 1$  and  $SKT3RW$

$\leq 8$ , or if  $MMS \leq 22$ , or if  $HBEFIND2 = 2$  and  $SKT2RW \geq 7$ , or if  $SKT1RW \geq 19$  and  $GDSSCORE \geq 4$ , then late/no dementia.

4. Else near-term dementia.

While explanations 3 and 4 seem contrived—indeed, they are rejected by the expert—explanations 1 and 2 are compact and at first glance appear to be reasonable. Hence, we select explanations 1 and 2. This is a fortunate choice. The accuracy of the explanations 1 and 2 on the testing data turns out to be 82%, compared with a lower 75% for the more complex explanations 3 and 4. Thus, the latter explanations are an example of overfitting. Part of the third condition [ $SKT4RW \geq 26$  and  $SKT3RW \leq 8$  and  $SKT5RW \geq 23$ ] of explanation 1 seems artificial. Indeed,  $SKT3RW$  measures the skill of reading numbers, and small values are associated with good performance. Thus, that portion of explanation 1 predicts near-term dementia if, in part, the skill of reading numbers is good. This does not make sense. Since the values of the attributes  $SKT_jRW$ ,  $j > 0$ , are considered to change randomly, explanations 1 and 2 are not stable.

EXARP determines the following factors.

Important Factors for Munich Dementia Data

Attribute	Significance Value
CDRWERT	0.88
PERSEV	0.84
SKT4RW	0.65

The two explanations are:

1. If  $CDRWERT \geq 1$  and [ $SKT4RW \geq 28$  or  $PERSEV = 0$ ], then near-term dementia.
2. If  $CDRWERT = 0$  or [ $SKT4RW \leq 25$  and  $PERSEV \geq 1$ ], then late/no dementia.

The explanations are consistent. Since  $SKT4RW$  is considered to be subject to random changes, while  $CDRWERT$  and  $PERSEV$  are not, the explanations are stable. The explanations classify 79% of the testing records correctly. An expert has examined the explanations and has supplied the following evaluation. The explanations make sense except that the appearance of the terms involving  $PERSEV$  is questionable. The  $PERSEV$  values tell how often names of a given collection, say of the collection of animals, are brought up repeatedly by a patient when as many names as possible are to be recalled. The expert says that the  $PERSEV$  values are possibly independent of early/late onset of dementia.

For all three methods, the significance of the explanations is better than the gold standard  $p < 0.0001$ .

Summary: C4.5 produces consistent and unstable explanations that seem artificial and too complex to be useful. The accuracy is high. RIPPER generates two pairs of consistent

but unstable explanations. One pair seems contrived. The other pair is simple, easy to understand, and has high accuracy. However, one of the rules contradicts common sense. EXARP provides two consistent and stable explanations. An expert agrees with the conclusions, except that he questions relevance of the terms involving  $PERSEV$ . Accuracy is somewhat less than that of C4.5 or RIPPER.

## 17. SUMMARY

This paper introduces a demanding definition of the explanation problem. A solution algorithm called EXARP is proposed. Essential for the algorithm is the notion of alternate random process (ARP). Such a process tends to distort data or otherwise disrupt computing steps. Appropriate counteractions by the algorithm guard against the detrimental effects of various ARPs. Use of the algorithm is demonstrated for three applications. For these cases, EXARP proves to be quite reliable. Partial results for ongoing work, not included here, give further support to that conclusion.

When we tested early versions of the algorithm, we were surprised that it very often constructed explanations as if it had prior domain knowledge. Yet this was accomplished without any such knowledge and without any manual guidance. That result occurred even when the training sets were small. We have the following intuitive explanation for this performance. If key differences exist between the records of two populations, then these differences must occur even in small training sets. Of course, the differences may be cluttered up by random variations. Nevertheless, the differences must be present. Algorithm EXARP effectively removes the random clutter via postulated ARPs, and thus is able to bring the differences to the surface and express them in salient explanations. On the other hand, if key differences do not exist between the records of two populations, then the removal of random clutter via ARPs leaves nothing of substance, and EXARP stops with that conclusion.

## 18. REFERENCES

- [1] S. Abidi and K. Hoe. Symbolic exposition of medical data-sets: A data mining workbench to inductively derive data-defining symbolic rules. In *Proceedings of the 15th IEEE Symposium on Computer-based Medical Systems (CBMS'02)*, 2002.
- [2] E. Acuná and C. Rodriguez. The treatment of missing values and its effect in the classifier accuracy. In *Classification, Clustering and Data Mining Applications*. Springer, 2004.
- [3] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 1993.
- [4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings 20th Int. Conf. Very Large Data Bases (VLDB)*, 1994.
- [5] S. Alexe, E. Blackstone, P. Hammer, H. Ishwaran, M. Lauer, and C. P. Snader. Coronary risk prediction by logical analysis of data. *Annals of Operations Research*, 119:15–42, 2003.
- [6] A. An. Learning classification rules from data. *Computers and Mathematics with Applications*,

- 45:737–748, 2003.
- [7] A. An and N. Cercone. Discretization of continuous attributes for learning classification rules. In *Proceedings of the Third Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-99)*, 1999.
- [8] M. Atzmueller, F. Puppe, and H.-P. Buscher. Exploiting background knowledge for knowledge-intensive subgroup discovery. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.
- [9] M. Atzmueller, F. Puppe, and H.-P. Buscher. Subgroup mining for interactive knowledge refinement. In *Proceedings of the 10th Conference on Artificial Intelligence in Medicine (AIME 05)*, 2005.
- [10] S. Bartnikowski, M. Granberry, J. Mugaň, and K. Truemper. Transformation of rational and set data to logic data. In *Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques*. Springer, 2006.
- [11] S. Bay and M. Pazzani. Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery*, 5:213–246, 2001.
- [12] E. Boros, P. Hammer, T. Ibaraki, and A. Kogan. A logical analysis of numerical data. *Mathematical Programming*, 79:163–190, 1997.
- [13] E. Boros, P. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik. An implementation of logical analysis of data. *IEEE Transactions on Knowledge and Data Engineering*, 12:292–306, 2000.
- [14] L. Breiman. Bagging predictors. *Machine Learning*, 26:123–140, 1996.
- [15] P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Proceedings Fifth European Working Session on Learning*, 1991.
- [16] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.
- [17] W. W. Cohen. Fast effective rule induction. In *Machine Learning: Proceedings of the Twelfth International Conference*, 1995.
- [18] W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1999.
- [19] G. Dong, X. Zhang, L. Wong, and J. Li. CAEP: Classification by aggregating emerging patterns. In *Proceedings of the Second International Conference on Discovery Science*, 1999.
- [20] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Machine Learning: Proceedings of the Twelfth International Conference*, 1995.
- [21] E. Fama and J. McBeth. Risk, return, and equilibrium: Empirical tests. *Journal of Political Economy*, 91:607–636, 1973.
- [22] H. Fan and K. Ramamohanarao. An efficient single-scan algorithm for mining essential jumping emerging patterns for classification. In *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD '02)*, 2002.
- [23] U. Fayyad and K. Irani. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8:87–102, 1992.
- [24] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, 1993.
- [25] G. Felici, F. Sun, and K. Truemper. Learning logic formulas and related error distributions. In *Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques*. Springer, 2006.
- [26] G. Felici and K. Truemper. A MINSAT approach for learning in logic domain. *INFORMS Journal of Computing*, 14:20–36, 2002.
- [27] D. Gamberger and N. Lavrač. Expert-guided subgroup discovery: Methodology and application. *Journal of Artificial Intelligence Research*, 17:501–527, 2002.
- [28] D. Gamberger, N. Lavrač, and G. Krstacić. Active subgroup mining: a case study in coronary heart disease risk group detection. *Artificial Intelligence in Medicine*, 28, 2003.
- [29] D. Gamberger, N. Lavrač, F. Železný, and J. Tolar. Induction of comprehensible models for gene expression datasets by subgroup discovery methodology. *Journal of Biomedical Informatics*, 37, 2004.
- [30] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [31] E. Hernández and J. Recasens. A general framework for induction of decision trees under uncertainty. In *Modelling with Words (LNAI 2873)*. Springer, 2003.
- [32] D. Jensen. *Induction with Randomization Testing: Decision-Oriented Analysis of Large Data Sets*. PhD thesis, Washington University, 1992.
- [33] K. R. Jinyan Li, Guozhu Dong. Making use of the most expressive jumping emerging patterns for classification. *Knowledge and Information Systems*, 3:131–145, 2001.
- [34] V. Jovanoski and N. Lavrač. Classification rule learning with APRIORI-C. In *Progress in Artificial Intelligence Knowledge Extraction, Multi-agent Systems, Logic Programming, and Constraint Solving*. Springer, 2001.
- [35] A. Kamath, N. Karmarkar, K. Ramakrishnan, and M. Resende. A continuous approach to inductive inference. *Mathematical Programming*, 57:215–238, 1992.
- [36] B. Kavsek, N. Lavrač, and V. Jovanoski. APRIORI-SD: Adapting association rule learning to subgroup discovery. In *Advances in Intelligent Data Analysis V*. Springer, 2003.
- [37] L. G. Khachiyan. A polynomial-time algorithm for linear programming. *Soviet Math. Dokl.*, 20:191–194, 1979.
- [38] W. Klösgen. EXPLORA: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*. AAAI Press, 1996.
- [39] W. Klösgen. Subgroup discovery. In *Handbook of Data Mining and Knowledge Discovery*. Morgan Kaufmann,

- 2002.
- [40] R. Kohavi and M. Sahami. Error-based and entropy-based discretization of continuous features. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.
- [41] D. Koller and M. Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, 1996.
- [42] N. Lavrač, B. Cestnik, D. Gamberger, and P. Flach. Decision support through subgroup discovery: Three case studies and the lessons learned. *Machine Learning*, 57, 2004.
- [43] N. Lavrač, B. Kavsek, P. Flach, and L. Todorovski. Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, 5:153–188, 2004.
- [44] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17:491–502, 2005.
- [45] O. Mangasarian and W. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23:1–18, 1990.
- [46] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [47] J. Mugan and K. Truemper. Discretization of rational data. In *Proceedings of MML2004 (Mathematical Methods for Learning)*. IGI Publishing Group, 2007.
- [48] T. Oates and D. Jensen. Large datasets lead to overly complex models: An explanation and a solution. In *Knowledge Discovery and Data Mining*, 1998.
- [49] R. Prati and P. Flach. ROCCER: an algorithm for rule learning based on ROC analysis. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.
- [50] J. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.
- [51] J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [52] A. Remshagen and K. Truemper. An effective algorithm for the futile questioning problem. *Journal of Automated Reasoning*, 34:31–47, 2005.
- [53] K. Riehl. *Data Mining Logic Explanations from Numerical Data*. PhD thesis, Department of Computer Science, University of Texas at Dallas, 2006.
- [54] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall/Pearson Education, Inc., 2003.
- [55] R. E. Schapire, M. Rochery, M. Rahim, and N. Gupta. Incorporating prior knowledge into boosting. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002)*, 2002.
- [56] E. Singer. Iressa's fall from grace points to need for better clinical trials. *Nature Medicine*, 11:107, 2005.
- [57] J. Straach and K. Truemper. Learning to ask relevant questions. *Artificial Intelligence*, 111:301–327, 1999.
- [58] E. Triantaphyllou. *Data Mining and Knowledge Discovery via a Novel Logic-based Approach*. Springer, to appear.
- [59] E. Triantaphyllou, L. Allen, L. Soyster, and S. Kumara. Generating logical expressions from positive and negative examples via a branch-and-bound approach. *Computers and Operations Research*, 21:185–197, 1994.
- [60] K. Truemper. *Design of Logic-based Intelligent Systems*. Wiley, 2004.
- [61] R. E. Valdes-Perez and V. P. and F. Pereira. Concise, intelligible, and approximate profiling of multiple classes. *International Journal of Human Computer Systems*, 53:411–436, 2000.
- [62] L. Wehenkel. On uncertainty measures used for decision tree induction. In *Info. Proc. and Manag. of Uncertainty in Knowledge-Based Systems (IPMU'96)*, 1996.
- [63] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proceedings of First European Conference on Principles of Data Mining and Knowledge Discovery*, 1997.
- [64] S. Wrobel. Inductive logic programming for knowledge discovery in databases. In *Relational Data Mining*, 2001.
- [65] S. Zhang, Z. Qin, C. Ling, and S. Sheng. 'Missing is useful': Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 17:1689–1693, 2005.
- [66] M. Zorman, P. Kokol, M. Lenic, P. Povalej, B. Stiglic, and D. Flisa. Intelligent platform for automatic medical knowledge acquisition: Detection and understanding of neural dysfunctions. In *Proceedings of the 16th IEEE Symposium on Computer-based Medical Systems (CBMS'03)*, 2003.