A SAT-Based Approach to Learn Explainable Decision Sets ⋆

Alexey Ignatiev[1,3], Filipe Pereira[1], Nina Narodytska[2], and Joao Marques-Silva[1]

[1] LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
{aignatiev,jpms}@ciencias.ulisboa.pt
fcpereira@lasige.di.fc.ul.pt
[2] VMWare Research, CA, U.S.A.
nnarodytska@vmware.com
[3] ISDCT SB RAS, Irkutsk, Russia

Abstract. The successes of machine learning in recent years have triggered a fast growing range of applications. In important settings, including safety critical applications and when transparency of decisions is paramount, accurate predictions do not suffice; one expects the machine learning model to also explain the predictions made, in forms understandable by human decision makers. Recent work proposed explainable models based on *decision sets* which can be viewed as unordered sets of rules, respecting some sort of rule non-overlap constraint. This paper investigates existing solutions for computing decision sets and identifies a number of drawbacks, related with rule overlap and succinctness of explanations, the accuracy of achieved results, but also the efficiency of proposed approaches. To address these drawbacks, the paper develops novel SAT-based solutions for learning decision sets. Experimental results on computing decision sets for representative datasets demonstrate that SAT enables solutions that are not only the most efficient, but also offer stronger guarantees in terms of rule non-overlap.

1 Introduction

Machine learning (ML) has witnessed remarkable progress and important successes in recent years [18, 22, 28]. In some settings, predictions made by machine learning algorithms should provide explanations, preferably explanations that can be interpreted (or understood) by human decision makers. Concrete examples include safety-critical situations, but also when transparency of decisions is paramount. The importance of explainable AI (XAI), i.e. the problem of associating explanations with ML predictions, is underscored by recent research [2, 21, 42], by ongoing research programs [9], by EU-level legislation which is expected to enforce the automated generation of explanations [11], and also by a number of meetings on computing explainable ML models [16, 17, 30].

An often used approach to provide explanations for ML predictions is to resort to some sort of logic-related model, including rule/decision lists, rule/decision sets, and decision trees [2, 21]. These logic-related models can in most cases associate explanations with predictions, represented as conjunctions of literals, that follow from the

⋆ This work was supported by FCT grants SAFETY (SFRH/BPD/120315/2016) and ABSOLV (028986/02/SAICT/2017), and LASIGE Research Unit, ref. UID/CEC/00408/2013.

actual model representation. Clearly, the smaller the model representation, the simpler the explanations are likely to be, and so easier to understand by human decision makers.

Recent approaches include the computation of (smaller or smallest) rule lists [2,42], the computation of decision sets [21], but also the computation of decision trees [3]. Rule lists impose an order of the rules [35], whereas decision sets do not. Clearly, from an interpretability perspective, decision sets are the most appealing since each prediction depends only on the literals associated with each rule. On the negative side, decision sets can exhibit rule overlap, and so may require decisions to be made when more than one class is predicted. Furthermore, even restricted forms of rule learning are well-known to be hard for NP [35].

This paper analyzes recent work on computing interpretable decision sets [21]. The paper highlights a number of drawbacks of the proposed approach, related with rule overlap, the generation of explanations, but also with the scalability of the approach. The paper then investigates three main topics. The first topic is the proposal of a rigorous definition of rule overlap. The paper relates this new definition with earlier work, and conjectures that solving the problem of overlap when learning optimal (in size) decision sets is hard for the second level of the polynomial hierarchy. The paper then proposes a number of variants of learning decision sets with less demanding constraints on overlap, and shows that these variants are instead hard for NP. The second topic is the issue of generating explanations for predictions. The paper shows that different models for learning decision sets provide different forms of computing explanations, thus enabling the generation of explanations in most settings. The third topic is to develop different propositional models for learning optimal decision sets. The proposed models build on earlier work on inductive inference [19], but introduce a number of variants, allowing for multiple classes, and also accommodating different overlap constraints. Moreover, the paper shows that all these models exhibit symmetries in the problem formulation, and so predicates breaking these symmetries can be used for improving performance.

The paper is organized as follows. Section 2 introduces the definitions and notation used in the remainder of the paper. The issue of overlap and explanation generation is investigated in Section 3. Propositional models for learning decision sets subject to different constraints on overlap are proposed in Section 4. Section 5 analyzes the performance of the proposed approach on representative datasets, and compares with earlier work [21]. Section 6 concludes the paper.

2 Preliminaries

This section briefly overviews Boolean Satisfiability (SAT), the classification problem in ML, and the learning of decision sets (DS). Throughout the paper, the notation $[R]$ is used to denote the set of natural numbers $\{1, \ldots, R\}$, moreover, for a point $\mathbf{f}$ in some K-dimensional space, the r^{th} coordinate is given by $\mathbf{f}[r]$.

Boolean Satisfiability (SAT). We assume notation and definitions standard in the area of SAT [4]. Formulas are represented in Conjunctive Normal Form (CNF) and defined over a set of variables $X = \{x_1, \ldots, x_n\}$. A formula $\mathcal{F}$ is a conjunction of clauses, a clause is a disjunction of literals, and a literal is a variable x_i or its complement $\neg x_i$. Where appropriate, formulas are viewed as sets of sets of literals. CNF encodings of car-

Ex.	Vacation (V)	Concert (C)	Meeting (M)	Expo (E)	Hike (H)
e_1	0	0	1	0	0
e_2	1	0	0	0	1
e_3	0	0	1	1	0
e_4	1	0	0	1	1
e_5	0	1	1	0	0
e_6	0	1	1	1	0
e_7	1	1	0	1	1

(a) A classification example

if ¬Meeting **then** Hike
if ¬Vacation **then** ¬Hike

(b) Decision set with some overlap

if Vacation **then** Hike
if ¬Vacation **then** ¬Hike

(c) Decision set with no overlap

Fig. 1: A classification example and its decision set

dinality constraints have been studied extensively, and will be assumed throughout [4]. Moreover, standard clausification techniques are assumed [39].

Classification problems. We follow the notation used in earlier work [3, 21]. We consider a set of features $\mathcal{F} = \{f_1, \ldots, f_K\}$, all of which are assumed to be binary, taking a value in $\{0, 1\}$. When necessary, the fairly standard one-hot-encoding [32] is assumed for handling non-binary categorical features. Numeric features can be handled with standard techniques as well. Since all features are binary, a literal on a feature f_r will be represented as f_r, denoting that the feature takes value 1, i.e. $f_r = 1$, or as $\neg f_r$, denoting that the feature takes value 0, i.e. $f_r = 0$. Hence, the space of features (or *feature space* [14]) is $\mathcal{U} \triangleq \prod_{r=1}^{K} \{f_r, \neg f_r\}$.

To learn a classifier, one starts from given training data (also referred to as examples or samples) $\mathcal{E} = \{e_1, \ldots, e_M\}$. Examples are associated with classes taken from a set of classes $\mathcal{C}$. The paper focuses mostly on binary classification, i.e. $\mathcal{C} = \{c_0, c_1\}$. (We will associate c_0 with 0 and c_1 with 1, for simplicity.) Thus, $\mathcal{E}$ is partitioned into $\mathcal{E}^+$ and $\mathcal{E}^-$, denoting the examples classified as positive ($c_1 = 1$) and as negative ($c_0 = 0$), respectively. Each example $e_q \in \mathcal{E}$ is represented as a 2-tuple (π_q, ς_q), where $\pi_q \in \mathcal{U}$ denotes the literals associated with the example and $\varsigma_q \in \{0, 1\}$ is the class to which the example belongs. We have $\varsigma_q = 1$ if $e_q \in \mathcal{E}^+$ and $\varsigma_q = 0$ if $e_q \in \mathcal{E}^-$. A literal l_r on a feature f_r, $l_r \in \{f_r, \neg f_r\}$, *discriminates* an example e_q iff $\pi_q[r] = \neg l_r$, i.e. the feature takes the value opposite to the value in the set of literals of the example. Moreover, we assume a mapping from feature values to classes, $\mu : \mathcal{U} \rightarrow \mathcal{C}$, i.e. we require consistency in the examples. Alternatively, we could allow for possible inconsistencies in the examples, by associating examples with elements of a relation $\rho \subseteq \mathcal{U} \times \mathcal{C}$.

Details on how to handle the extensions to this basic formulation, including non-binary features, the handling of non-binary classes, and allowing for inconsistent examples, are beyond the scope of this paper but are discussed in later sections. Furthermore, in this paper we assume that all features are specified for all examples; the work can be generalized for situations where the value of some features for some examples is left unspecified.

In the remainder of the paper, we will also consider non-conflicting subsets of $\mathcal{L} \triangleq \cup_{r=1}^{K} \{f_r, \neg f_r\}$, such that a subset of $\mathcal{L}$ is *non-conflicting* if for all features f_r, the literals f_r and $\neg f_r$ do not both occur in that subset. When referring to the actual data points representing the examples in $\mathcal{E}$, we use the notation $\mathbf{f}$, with $\mathbf{f} \in \prod_{r=1}^{K} \{f_r, \neg f_r\}$.

Example 1. Figure 1a shows a simple classification example. The set of binary features is $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$ with $f_1 \triangleq \text{V}$, $f_2 \triangleq \text{C}$, $f_3 \triangleq \text{M}$, and $f_4 \triangleq \text{E}$. Example e_1 is represented by the 2-tuple (π_1, ς_1), with $\pi_1 = (\neg\text{V}, \neg\text{C}, \text{M}, \neg\text{E})$ and $\varsigma_1 = 0$. Moreover, the literals V, C, $\neg$M and E discriminate e_1. For this classification example, we have $\mathcal{U} = \{\text{V}, \neg\text{V}\} \times \{\text{C}, \neg\text{C}\} \times \{\text{M}, \neg\text{M}\} \times \{\text{E}, \neg\text{E}\}$.

The objective of classification is to learn some function $\hat{\phi}$ which matches the actual function ϕ on the training data and generalizes *suitably well* on unseen test data [13, 14, 27, 34]. In this paper, we seek to learn representations of $\hat{\phi}$ corresponding to *decision sets* (DS). Many other representations have been studied, including decision trees [34], rule lists [2], and sums of terms (i.e. DNF) [15, 41], among others. These are of interest, including for XAI, but are beyond the scope of this work.

Related work. Rule learning, as a form of covering problem, can be traced back to the 1960s [26]. Rule learning finds important applications in ML and Data Mining (DM), and it is a standard topic in ML and DM textbooks [13, 14, 27]. Although rule learning has been investigated at the propositional and predicate levels, in different settings, the focus of this paper is the optimal learning of propositional rules. Rules can be organized as lists, being referred to as rule (or decision) lists, or as sets, being also referred to as rule (or decision) sets. The difference between the two representations is that lists impose an order on the rules, and sets do not. It is well-known that learning optimal rule lists is NP-hard [20]. As a result, most algorithms for learning rule lists or sets are heuristic [7, 8, 33, 34], being in general efficient to run, but providing essentially no guarantees in terms of the quality of the computed rules. Recent work has focused on developing small or optimal rule (or decision) lists [2], but also rule (or decision) sets [21]. The focus of the paper is the learning of decision sets, and so we investigate in more detail the recently proposed IDS (*interpretable decision sets*) approach [21].

3 Learning Explainable Decision Sets

This section introduces, but also generalizes, the definitions proposed in earlier work [21] for the problem of learning decision sets.

Definition 1 (Itemset)**.** Given $\mathcal{F}$, an *itemset* π is an element of $\mathcal{I} \triangleq \prod_{r=1}^{K} \{f_r, \neg f_r, \mathfrak{u}\}$, where $\mathfrak{u}$ represents a *don't care* value. Where applicable, an itemset π is also interpreted as the conjunction of the coordinates different from $\mathfrak{u}$, i.e. the *specified* literals of π.

Clearly, an itemset represents a cube in the K-dimensional feature space. Moreover, a K-dimensional point in feature space is also a (completely specified) itemset.

Definition 2 (Clashing itemsets)**.** Given two itemsets $\pi_1, \pi_2 \in \mathcal{I}$, the two itemsets *clash*, written $\pi_1 \cap \pi_2 = \emptyset$, if and only if there exists a coordinate r such that $\pi_1[r] = f_r$ and $\pi_2[r] = \neg f_r$, or $\pi_1[r] = \neg f_r$ and $\pi_2[r] = f_r$.

Definition 3 (Rule)**.** A *rule* is a 2-tuple (π, ς), where $\pi \in \mathcal{I}$ is an itemset, and $\varsigma \in \mathcal{C}$ is a class. Moreover, a rule (π, ς) is to be interpreted as follows:

IF the specified literals in π are true, **THEN** pick class ς

Rules can and have been used in different settings [13, 14, 27]. This paper considers the use of rules as the building block of decision sets.

Definition 4 (Decision Sets)**.** Given a set of (binary) features $\mathcal{F}$, defining a feature space $\mathcal{U}$, and a set of classes $\mathcal{C}$, a *decision set* $\mathbb{S}$ is a finite set of rules.

Given a decision set $\mathbb{S}$, there may exist points in feature space not covered by $\mathbb{S}$. An often used (optional) solution is to consider a *default* rule, which applies whenever the disjunction of the conjunctions of literals associated with each rule of $\mathbb{S}$ takes value 0.

Definition 5 (Default rule $\mathfrak{D}$)**.** A rule of the form $\mathfrak{D} \triangleq (\emptyset, \varsigma)$ denotes the *default* rule of a decision set $\mathbb{S}$, applicable when all the other rules take value 0 on a given point of feature space. The class selected is ς.

Example 2. Referring back to Example 1, Figure 1b shows an example of a decision set for the dataset of Figure 1a, whereas Figure 1c shows a different decision set. (The difference between the two relates with the notion of overlap to be introduced below.) Moreover, for the first decision set (see Figure 1b), a (necessary) default rule could be $(\emptyset, 0)$. For example, for the feature space point $(\mathrm{V}, \mathrm{C}, \mathrm{M}, \mathrm{E})$ we can now say that the class, due to the default rule, is 0.

In contrast with earlier work [21], we consider generalized forms of cover, subject to subsets of the feature space.

Definition 6 ($\mathcal{X}$-Cover)**.** Given $\mathcal{X} \subseteq \mathcal{U}$ and an itemset, the $\mathcal{X}$-cover of the itemset is the set of feature space points in $\mathcal{X}$ with a non-empty intersection with the itemset. The cover of the default rule $\mathfrak{D}$ is the set of points in feature space not covered by *any* of the other rules of a decision set.

Earlier work [21] considers a less general definition of cover, where $\mathcal{X}$ corresponds to the training data $\mathcal{E}$. Overlap between two rules assesses whether the set of points covered by two rules intersect. Overlap has been investigated recently in the context of learning decision sets [21]. This earlier work focused on overlap *solely* with respect to the *training data*, i.e. the starting set of examples, providing *no* guarantees on any other point of feature space. As a result, and in contrast with earlier work [21], we consider generalized forms of overlap, subject to subsets of the feature space.

Definition 7 ($\mathcal{X}$-overlap)**.** Two rules $r_1 = (\pi_1, \varsigma_1)$ and $r_2 = (\pi_2, \varsigma_2)$ *overlap* in $\mathcal{X} \subseteq \mathcal{U}$ iff,

$$\exists\, \mathbf{f} \in \mathcal{X}.\ \mathbf{f} \cap \pi_1 \neq \emptyset \ \text{ and } \ \mathbf{f} \cap \pi_2 \neq \emptyset \tag{1}$$

Observe that the simpler definition, requiring $\pi_1 \cap \pi_2 \neq \emptyset$, would not enable restricting overlap to specific subsets of $\mathcal{U}$. Furthermore, the definition of overlap considered in earlier work [21] corresponds to $\mathcal{E}$-overlap.

The above definition can be qualified with $\oplus$ or $\ominus$, depending if we are concerned with overlap where the classification agrees ($\oplus$), i.e. all rules whose bodies are not false predict the same class, or disagrees ($\ominus$), i.e. there exist rules whose bodies are not false that do not predict the same class.

More importantly, the proposed formulation of overlap enables investigating the quality of decision sets in points of feature space *not* covered by the initial set of examples. We will be mostly concerned with $\mathcal{U}^{\ominus}$-overlap between pairs of rules with different classifications, aiming to eliminate such overlap. We will be less concerned with $\mathcal{U}^{\oplus}$-overlap, but this can also be deemed of interest [21].

Example 3. With respect to Example 1 and the decision set shown in Figure 1b there is no $\mathcal{E}$-overlap, but there is overlap in feature space. For the point $(\neg\mathrm{V}, \neg\mathrm{C}, \neg\mathrm{M}, \neg\mathrm{E}) \in \mathcal{U}$ we have $\ominus$ overlap. Moreover, the decision set in Figure 1c exhibits no overlap.

Generating succint explanations. For a rule (π, ς), its *explanation* is the conjunction of literals in π. Thus, for *any* point in feature space for which there exists no $\ominus$ overlap,

we can simply pick one of the rules consistent with that point as the explanation for the prediction. In this situation, we refer to the explanation as *offline* (or *explicit*). Moreover, assuming there is no $\ominus$ overlap and that all points in feature space are covered by some rule, then the set of rules provides a succint representation of the explanations of the predictions made. If there exists $\ominus$ overlap, then one can simply pick one of the rules for which the itemset takes value 1, and list the itemset as an explanation. One additional case, is when some point in feature space is not covered by any rule in a decision set. In this case, one resorts to a default rule $\mathfrak{D} = (\emptyset, \varsigma)$, which has no immediate explanation. Nevertheless, it is still possible to provide explanations, albeit the set of justifications can no longer be represented succintly. We consider a point $\mathbf{f}$ in feature space such that no rule is applicable, and so the default rule is used. For each rule $r_i = (\pi_i, \varsigma_i)$, there must exist one literal $l_{i,k}$ that falsifies the itemset π_i. As a result, an explanation for selecting the default rule can be constructed by picking one falsified literal from each itemset of each rule with a class that is *not consistent* with the class associated with the default rule. We refer to these explanations as *online* (or *implicit*). Clearly, the explanation will depend on each point on feature space not covered by the other rules, but we are still able to produce explanations.

Example 4. We consider again Example 1 and the decision set in Figure 1b, assuming a default rule $(\emptyset, 0)$. For the point in feature space $(\mathrm{V}, \mathrm{C}, \mathrm{M}, \mathrm{E})$ the prediction will be 0 (i.e. $\neg$H), due to the default rule. Moreover, since the prediction will be 0, then we pick a 0-valued literal from the rules that would predict a different class, M in this case for the first rule. Thus, we can provide the explanation $\{\mathrm{M}\}$; i.e. any time there is a `meeting`, then we will *not* take the `hike`.

4 Learning Decision Sets with SAT

This section develops different SAT models for learning decision sets. We can associate a Boolean function E^0 with $\mathcal{E}^-$, which takes value 1 for each point in feature space associated with $\mathcal{E}^-$, i.e. each combination of binary features that represents an example in $\mathcal{E}^-$ is a minterm of E^0. Similarly, we associate a Boolean function E^1 with $\mathcal{E}^+$, which takes value 1 for each point in the feature space associated with $\mathcal{E}^+$. Moreover, each combination of binary features that represents an example in $\mathcal{E}^+$ is a minterm of E^1. Clearly, our working hypothesis is that $E^0 \land E^1 \models \bot$, i.e. the examples represent a mapping. As shown below, the minimum decision set problem can be formalized in different ways. This paper considers a general formalization of the minimum decision set problem, in terms of computing two sets of terms F^0 and F^1, i.e. two DNF representations, and is defined as follows:

Definition 8. [MINDSET, MINDS_0] Let $\langle \mathcal{E}^-, \mathcal{E}^+ \rangle$ be a 2-tuple of examples associated with two distinct classes, c_0 and c_1, and each represented by Boolean functions E^0 and E^1, respectively. MINDS_0 is the problem of finding the smallest DNF representations of Boolean functions F^0 and F^1, measured in the number of *terms*, such that: (i) $E^0 \models F^0$; (ii) $E^1 \models F^1$; and (iii) $F^1 \leftrightarrow F^0 \models \bot$.

Observe that condition (iii) above ensures that a decision set is computed (1) exhibiting no $\mathcal{U}^{\ominus}$-overlap and (2) covering the complete feature space $\mathcal{U}$. This should be compared with the substantially less demanding constraint of $\mathcal{E}$-overlap investigated in earlier work [21]. Moreover, the cost of the DNF representation could be measured

in terms of the number of literals. The paper considers the cost in terms of the number of terms (or rules), but it is straightforward to extend to considering also literals. Alternatives to take the number of literals into account are investigated in Section 5.

Lemma 1. For any decision set respecting Definition 8, it holds that (i) $F^0 \land E^1 \models \bot$; and (ii) $F^1 \land E^0 \models \bot$.

Proposition 1. The decision version of MINDS_0 is in Σ_2^{p}.

Proof. (Sketch) Given some size threshold T, simply guess the terms of the two DNFs, F^0 and F^1, using no more than T terms, and then check that, for every assignment, the values of F^0 and F^1 differ. Clearly, this can be encoded as a 2QBF formula. □

Furthermore, the decision version of the MINDS_0 problem is apparently hard for Σ_2^{p}. For example, if we minimize E^0 to a DNF F^0, then computing the smallest DNF F^1 subject to F^0 is a well-known Σ_2^{p}-hard problem [40].

Conjecture 1. MINDS_0 is hard for Σ_2^{p}.

The proof (or disproof) of this conjecture is left as future work. Given the above, we can envision the following optimization problems, studied in the remainder of the paper, which result from relaxing the constraint $F^1 \leftrightarrow F^0 \models \bot$ of MINDS_0, thus achieving hardness for NP:

1. MINDS_4: Minimize F^0, given $F^1 \equiv E^1$ constant, and such that (i) $E^0 \models F^0$; and (ii) $F^0 \land E^1 \models \bot$.
2. MINDS_3: Same as above, but for F^1 given $F^0 \equiv E^0$ constant.
3. MINDS_2: Minimize both F^0 and F^1, such that (i) $E^0 \models F^0$; (ii) $E^1 \models F^1$; (iii) $F^0 \land E^1 \models \bot$; and (iv) $F^1 \land E^0 \models \bot$.
4. MINDS_1: Minimize F^0 and F^1, such that (i) $E^0 \models F^0$; (ii) $E^1 \models F^1$; and (iii) $F^1 \land F^0 \models \bot$.

Observe that all of the above problems are weakened versions of MINDS_0, the main difference being the constraints on the functions associated with E^0 and E^1. Among MINDS_i, $i \neq 0$, MINDS_1 imposes the most severe constraint, ensuring no $\mathcal{U}^{\ominus}$-overlap takes place, although there may be points for which both F^0 and F^1 take value 0.

Proposition 2. The decision versions of the optimization problems MINDS_1, MINDS_2, MINDS_3 and MINDS_4 above are complete for NP.

Proof. (Sketch) The simplest solution is to use earlier results [36, 40] to argue that the decision versions of MINDS_3 and MINDS_4 are complete for NP. (Earlier work [19] claims NP-hardness, but citing references that do not actually prove the result.)
It is easy to reduce MINDS_3 or MINDS_4 to MINDS_1 or MINDS_2; i.e. simply ignore the other computed function. Moreover, we show below that the decision versions of MINDS_1 and MINDS_2 are in NP, by reducing these problems to SAT. Thus, completeness of MINDS_1 and MINDS_2 follows. □

Example 5. With respect to Example 1, the decision set shown in Figure 1b respects MINDS_2, MINDS_3 and MINDS_4, whereas the decision set of Figure 1b also respects MINDS_1 and MINDS_0.

In the sections below we investigate SAT-based models for computing decision sets, under one of the relaxed optimization models MINDS_1, MINDS_2, MINDS_3 or MINDS_4. Moreover, we investigate symmetry breaking properties of the problem formulation, which can be used for constraining any of these models.

4.1 SAT Models for MINDS$_3$ & MINDS$_4$

This section details SAT models for solving MINDS$_3$. With minor modifications, similar models can be devised for MINDS$_4$. The purpose of MINDS$_3$ is to find a minimum-size representation of F^1, subject to a non-$\mathcal{U}^{\ominus}$-overlap constraint with respect to E^0. To solve this problem, a number of propositional models can be envisioned. We first investigate a model proposed in the literature [19,37,38]. Afterwards, we detail a new model, aiming at better performance when using SAT solvers. Both propositional models encode the decision problem of MINDS$_3$: can F^1 be represented with N terms? The model considers a grid of N by K entries, each row of K entries denoting the representation of the condition of a rule or, alternatively, a term in the DNF representation of F^1, for a total of N terms. Throughout this section, it holds that $1 \leq j \leq N$ and $1 \leq r \leq K$, with q associated with some example e_q from $\mathcal{E}$, $\mathcal{E}^-$ or $\mathcal{E}^+$.

An existing SAT model. One model, proposed by Kamath et al. [19], assumes the representation of a Boolean function in terms of K-dimensional points describing the functions ON-set and the OFF-set, respectively E^1 and E^0 in our case.

The variables used in the propositional representation are:

- $p_{jr} = 1$ iff x_i not included in term j.
- $p'_{jr} = 1$ iff $\neg x_i$ not included in term j.
- sl^q_{jr}: replace either with p'_{jr} if feature f_r occurs positively in $e_q \in \mathcal{E}^+$, or with p_{jr} if feature f_r occurs negatively in $e_q \in \mathcal{E}^+$.
- $cr_{jq} = 1$ iff rule j covers $e_q \in \mathcal{E}^+$.

Furthermore, the constraints proposed in [19] can be translated as follows:

1. One of p_{jr} and p'_{jr} must be true:
$$(p_{jr} \lor p'_{jr}) \qquad j \in [N] \land r \in [K] \tag{2}$$
2. Each negative example $e_q \in \mathcal{E}^-$, with a set of positive features P_q and a set of negative features N_q, must be discriminated by every term:
$$\left(\bigvee\nolimits_{r \in P_q} \neg p'_{jr} \lor \bigvee\nolimits_{r \in N_q} \neg p_{jr}\right) \qquad j \in [N] \land e_q \in \mathcal{E}^- \tag{3}$$
3. Each positive example must be covered:
   - Constraint for a term not covering a positive example:
   $$(sl^q_{jr} \lor \neg cr_{jq}) \qquad j \in [N] \land r \in [K] \land e_q \in \mathcal{E}^+ \tag{4}$$
   - Each positive example must be covered by some term:
   $$\left(\bigvee_{j=1}^{N} cr_{jq}\right) \qquad e_q \in \mathcal{E}^+ \tag{5}$$

Analysis of the constraints yields the following:

Proposition 3. The model uses $\mathcal{O}(N \times M \times K)$ clauses and literals.

An alternative model. In contrast with the model of Kamath et al. [19], we propose a model with a different semantics for some of the variables, and a few additional clauses, to elicit propagation. As shown by the experimental results, the motivation has been to devise a model for which the computed solutions are (heuristically) easier to interpret, by specifying fewer literals.

The sets of variables to use are the following:

- s_{jr}: whether for rule j, a literal in feature r is to be skipped.

- l_{jr}: literal on feature r for rule j, in the case the feature is *not* skipped.
- d^0_{jr}: whether feature r of rule j discriminates value 0.
- d^1_{jr}: whether feature r of rule j discriminates value 1.
- cr_{jq}: whether (*used*) rule j covers $e_q \in \mathcal{E}^+$.
  (Observe that this variable is also used in the existing model [19].)

The constraints encoding MINDS$_3$ are:

1. Each term must have some literals:
$$\left(\bigvee_{r=1}^{K} \neg s_{jr}\right) \qquad j \in [N] \tag{6}$$
2. One must be able to account for which literals are discriminated by which rules:
$$\begin{aligned} d^0_{jr} &\leftrightarrow \neg s_{jr} \land l_{jr} \qquad && j \in [N] \land r \in [K] \\ d^1_{jr} &\leftrightarrow \neg s_{jr} \land \neg l_{jr} \qquad && j \in [N] \land r \in [K] \end{aligned} \tag{7}$$
3. In addition, one must be able to discriminate all the negative examples in each term. Let $e_q \in \mathcal{E}^-$ be a negative example, and $\sigma(r,q)$ denote the sign of feature f_r for e_q. Then,
$$\left(\bigvee_{r=1}^{K} d^{\sigma(r,q)}_{j,r}\right) \qquad j \in [N] \land e_q \in \mathcal{E}^- \tag{8}$$
4. We must also ensure that each positive example is covered by some rule, associated with its class.
   - First, define whether a rule covers some specific positive example:
   $$cr_{jq} \leftrightarrow \left(\bigwedge_{r=1}^{K} \neg d^{\sigma(r,q)}_{j,r}\right) \qquad j \in [N] \land e_q \in \mathcal{E}^+ \tag{9}$$
   - Second, each $e_q \in \mathcal{E}^+$ must be covered by some rule. This corresponds to (5).

Proposition 4. The propositional encoding uses $\mathcal{O}(N \times M \times K)$ clauses and literals.

4.2 SAT Models for MINDS$_1$ & MINDS$_2$

The models analyzed in the previous section, MINDS$_3$ and MINDS$_4$, learn one function for one class, e.g. F^1 for c_1. For the other class, e.g. c_0, only the original minterms are available, and a default rule that may opt to pick this other class for points of feature space not covered by F^1. It is in general possible to have more accurate representations of the two classes, by considering some of the models described earlier in this paper, concretely MINDS$_2$ and MINDS$_1$. This section develops propositional models for MINDS$_2$ and MINDS$_1$.

The case of MINDS$_2$. It is immediate to generalize MINDS$_3$ (or MINDS$_4$) to the case of MINDS$_2$. Essentially, the constraints for discriminating classes and for covering classes must be replicated for the target classes [1].

The case of MINDS$_1$. We consider a grid of N by K entries, each row of K entries denoting the organization of a rule. The (basic) sets of variables to use are the same as for MINDS$_3$, with the addition of c_j, representing a class variable, which is 0 if the class of rule j is false (or negative), and 1 otherwise. Moreover, the constraints encoding MINDS$_1$ are:

1. Every term *must* be used. This constraint corresponds to (6).

[1] The generalization from MINDS$_3$ to MINDS$_2$ is straightforward, and omitted due to space constraints. Moreover, the model for MINDS$_1$ follows a similar approach.

2. We must also be able to account for which literals are discriminated by which rules. This constraint corresponds to (7).
3. In addition, we must be able to discriminate positive examples in rules of the negative class and vice-versa. Let $e_q \in \mathcal{E}^+$ be a positive example, and $\sigma(r,q)$ be defined as above. Then,
$$\begin{aligned} \neg c_j &\rightarrow \left(\bigvee_{r=1}^{K} d_{j,r}^{\sigma(r,q)} \right) \qquad j \in [N] \land e_q \in \mathcal{E}^+ \\ c_j &\rightarrow \left(\bigvee_{r=1}^{K} d_{j,r}^{\sigma(r,q)} \right) \qquad j \in [N] \land e_q \in \mathcal{E}^- \end{aligned} \tag{10}$$
4. We must also ensure that each example is covered by some rule, associated with its class.
   - First, the constraint for a rule to cover some example:
$$\begin{aligned} cr_{jq} &\leftrightarrow \left(\neg c_j \land \bigwedge_{r=1}^{K} \neg d_{j,r}^{\sigma_{r,q}} \right) \qquad j \in [N] \land e_q \in \mathcal{E}^- \\ cr_{jq} &\leftrightarrow \left(c_j \land \bigwedge_{r=1}^{K} \neg d_{j,r}^{\sigma_{r,q}} \right) \qquad j \in [N] \land e_q \in \mathcal{E}^+ \end{aligned} \tag{11}$$
   - Second, all examples no matter the class must be covered, and so we generalize (5) to get:
$$\left(\bigvee_{j=1}^{N} cr_{jq} \right) \qquad e_q \in \mathcal{E} \tag{12}$$
   Thus, every element is covered.
5. Finally, two terms associated with different classes *must not* exhibit $\mathcal{U}^{\ominus}$-overlap:
$$\neg(c_i \leftrightarrow c_j) \rightarrow \left(\bigvee_{r=1}^{K} \neg s_{ir} \land \neg s_{jr} \land \neg(l_{ir} \leftrightarrow l_{jr}) \right) \qquad i, j \in [N] \land i < j \tag{13}$$

4.3 Breaking Symmetries

The propositional models proposed in earlier sections essentialy capture (unordered) sets of terms. The lack of order reveals a symmetry. If the number of terms is large, this can impact performance significantly. A standard technique to eliminate such symmetries in the problem formulation is to impose an order in the representation. The approach we take is to sort the terms, such that the number of each feature is inverse to the weight of the feature in the binary representation of the number associated with the term. Unspecified features have the largest weight. Clearly, imposing an order on the terms does *not* affect correctness of the propositional model.

We describe next the constraints for the alternative model proposed in Section 4.1. For the other models, a similar solution is used. The additional variables used are the following:

- $eq_{j,r} = 1$ iff term j equals term $j-1$ until feature r.
- $gt_{j,r} = 1$ iff term j is greater than term $j-1$ by feature r.

For the constraints below $j \in [N]$ and $r \in [K]$. The constraints for $eq_{j,r}$ are the following, with $eq_{j,0} = 1$:
$$eq_{j,r} \leftrightarrow eq_{j,r-1} \land \left(s_{j-1,r} \land s_{j,r} \lor d_{j-1,r}^{1} \land d_{j,r}^{1} \lor d_{j-1,r}^{0} \land d_{j,r}^{0} \right) \tag{14}$$
The constraints for gt_{jr}, with $gt_{j0} = 0$, are the following:
$$gt_{j,r} \leftrightarrow gt_{j,r-1} \lor eq_{j,r-1} \land \neg s_{j-1,r} \land s_{j,r} \lor eq_{j,r-1} \land d_{j-1,r}^{1} \land d_{j,r}^{0} \tag{15}$$

Observe that distinguishing a positive literal corresponds to accepting a negative literal. Clearly, additional variables can be introduced to enable clausification [39]. Finally, for the last feature, each term must be greater than the preceeding one:

$$\bigwedge_{j=2}^{N} (gt_{j,K}) \tag{16}$$

5 Experimental Results

This section evaluates the ideas studied in the paper given a variety of datasets.

5.1 Experimental Setup

The proposed models were implemented in a prototype as a Python script instrumenting calls to the MiniSat 2.2 SAT solver [10]. More precisely, the weakened models MINDS_i, $i \in [4]$, were implemented. Although all these models target binary classification, most of the practical benchmark datasets require non-binary classification. Therefore, the implemented prototype supports non-binary classification as well. As a result, we deem interesting for the evaluation to check the performance of models MINDS_2 and MINDS_1 generalized to an arbitrary number of classes, and so we *do not* test models MINDS_3 and MINDS_4, as they are expected to be easier to deal with. Also note that the implementation supports both encodings of MINDS_3 (and, thus, of generalized MINDS_2) studied in the paper: (1) the existing encoding [19] and (2) the alternative encoding proposed above. In the following, the novel encoding of MINDS_2 is simply called MinDS_2 while the encoding from [19] is referred to as MP92. Additionally and for testing how helpful the proposed symmetry breaking predicates (SBPs) are, the basic models were augmented with SBPs resulting in the following configurations: MinDS_2+SBP, MinDS_1+SBP, and MP92+SBP. Finally, IDS[2], a recent approach [21] based on *smooth local search* [12], was also tested in the evaluation. IDS uses the Apriori algorithm [1] for generating candidate itemsets, with the default support threshold[3] equal to 0.2. For simplifying the problem solved by IDS, we increased this value to 0.5, which resulted in two configurations of IDS to run: IDS-supp0.2 and IDS-supp0.5.

The experiments were performed on a subset of datasets of the PMLB repository[4] [31]. The number of samples in the selected datasets varies from 87 to 49621[5] ($\approx$ 1651.1 on average) while the number of original (i.e. non-binary) features varies from 4 to 59 ($\approx$ 15.1 on average). Applying the *one-hot encoding* results in 6 to 2232 binary features ($\approx$ 353.1 on average). The total number of selected datasets is 49.

All the conducted experiments were performed in Ubuntu Linux on an Intel Xeon E5-2630 2.60GHz processor with 64GByte of memory. The time limit was set to 600s and

[2] https://github.com/lvhimabindu/interpretable_decision_sets/

[3] A support threshold parameter ϵ in the Apriori algorithm ensures that the candidate itemsets are present in at least ϵ data points.

[4] https://github.com/EpistasisLab/penn-ml-benchmarks/

[5] Some of the PMLB datasets are inconsistent, i.e. they have multiple occurrences of the same samples marked by different labels. Since the proposed models assume consistent data, the datasets were replaced by their largest consistent subsets. The number of samples shown above corresponds to the size of the resulting consistent datasets.

Table 1: Number of solved instances per model (out of 49 in total).

MP92	MP92+SBP	MinDS_2	MinDS_2+SBP	MinDS_1	MinDS_1+SBP	IDS-supp0.2	IDS-supp0.5
42	45	42	45	6	6	0	2

the memory limit to 10GByte for each individual process to run. The experimental evaluation was divided into two parts detailed below.

5.2 Testing Scalability

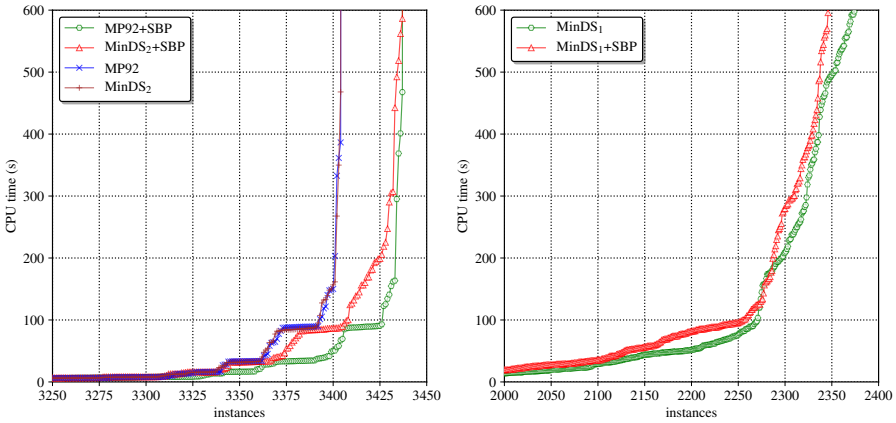
The number of benchmarks solved by each competitor is shown in Table 1. Given the large number of binary features in the datasets, the performance of both MP92 and MinDS_2 can be regarded as quite positive. As expected, symmetry breaking improves it further: MP92+SBP and MinDS_2+SBP solve all but 4 instances. Observe that MinDS_1 and MinDS_1+SBP perform significantly worse: these models can solve only 6 instances. However, this is not surprising given that MinDS_1 targets computing decision sets exhibiting no $\mathcal{U}^{\ominus}$-overlap, which is in general significantly harder to solve.

Assessing the performance of IDS [21]. As shown in Table 1, and in contrast to the SAT-based models studied, IDS [21] performs quite poorly in practice. With the default support threshold 0.2, IDS is unable to solve (within 600s) any instance, and it can solve only 2 instances if the support threshold is increased to 0.5. Moreover, and although IDS aims at maximizing the number of covered training samples and minimizing the rule overlap, the rules produced by IDS exhibit significant overlap, even on examples taken from the training data[6]. Given the poor performance of IDS and the weak guarantees in terms of rule overlap, the rest of this section focuses solely on the SAT-based models.

Performance on subsampled datasets. To investigate the performance of the models further, we (1) discarded the 6 instances solved by all models and (2) subsampled the remaining 43 $(49-6)$ benchmarks in the following way. For each dataset, we randomly selected 5%, 10%, 20%, and 50% of training samples and repeated this procedure 20 times for each percentage value. This resulted in 80 randomly subsampled datasets for each of the 43 benchmarks. The total number of subsampled benchmarks is 3440.

Figure 2 depicts the performance of the proposed models on the subsampled benchmarks. As shown in Figure 2a, MP92 and MinDS_2 demonstrate almost the same performance and solve successfully 3404 benchmarks. Enabling symmetry breaking proves itself helpful allowing them to solve 33 more instances, i.e. 3437 overall. This is, however, not the case for MinDS_1, which solves 2374 instances (2346 instances, resp.) if SBPs are disabled (enabled, resp.). This can be explained by the benchmarks' nature as they have a large number of classes and training samples while the solutions are not large enough for SBPs to pay off. Note that although MinDS_1 performs significantly worse than MP92 and MinDS_2, it can still solve $\approx 70\%$ of the subsampled benchmarks. These results should be regarded as significant given the size and the properties of the tested datasets, as well as the fact that MinDS_1 targets no $\ominus$ overlap on the *com-*

[6] These surprising results motivated in part our detailed analysis of overlap. It should be noted that the authors of IDS [21] have been informed of IDS's poor performance and poor ability to avoid rule overlap, but have been unable to justify the results of IDS.



(a) MinDS_2 and MP92 (b) MinDS_1

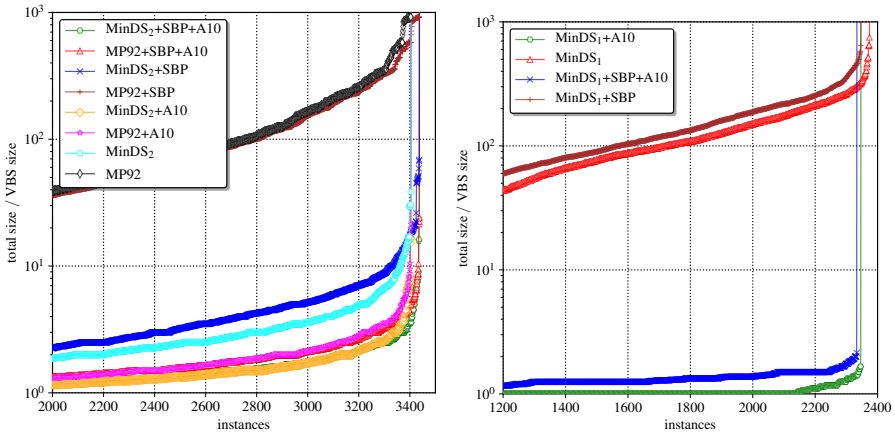
Fig. 2: Performance of the considered models on subsampled datasets.

plete feature space. To our best knowledge, these results are far beyond the state of the art [21], and enable solving to optimality a whole new range of challenging datasets.

5.3 Assessing Quality

Observe that the proposed models target minimizing the number of rules in the target decision sets, rather than their total size, i.e. the total number of literals used. Hence, it is of interest to compare the "quality" of solutions reported by MP92 and the novel model MinDS_2. One option is to simply compare the number of literals in the decision sets reported by the two models. Alternatively, one can try to minimize the number of literals in the resulting decision sets, by applying Boolean lexicographic optimization (BLO) [23], as soon as a decision set with the smallest number of rules is computed. For this, as soon as the number of rules in the decision set is minimized (i.e. the corresponding CNF formula is satisfiable), a simple MaxSAT problem can be devised by augmenting the formula with unit soft clauses, which force all literals of the decision set to be unused. This can be applied to any model MINDS_i, $i \in [4]$. Afterwards, the minimum number of literals can be computed by a standalone MaxSAT solver or approximated with the use of an MCS (minimal correction subset) enumerator [24]. While the former approach is exact, it is often outperformed by the latter one. For the purpose of the evaluation, we tried both options with every model considered. The MaxSAT solver used for computing exact solutions was MSCG [29] while the approximation of MaxSAT solutions was done by computing first 10 MCSes with the LBX algorithm [25].

Evaluation of the quality of solutions was done in the following way. Additionally to the tested configurations of MinDS_2 and MP92, all of them were ran in the BLO mode with literal minimization done by (1) a MaxSAT solver and (2) an MCS enumerator. Among all configurations, a *virtual best solver* (VBS) was constructed w.r.t. the total number of literals in the solution. Afterwards, we measured how much larger the decision sets for each tested configuration are w.r.t. the VBS, i.e. given the number of literals L in the solution produced by the configuration and the number of literals L^* in the VBS solution, we considered value L/L^*. A similar study was done for MinDS_1.



(a) MinDS_2 and MP92 (b) MinDS_1

Fig. 3: Quality of solutions computed with the considered models.

Figure 3 shows the quality of solutions for all tested models. (Y-axis here is scaled logarithmically.) Here, the configurations marked by **+A10* compute 10 MCSes to approximate the solution. All configurations that use a MaxSAT solver represent a constant $f(x) = 1$ and, thus, are omitted in Figure 3. However, note that they participate in the VBS. In general, our experiments suggest that the MaxSAT-based literal minimization is expensive and results in only $\approx 85\%$ of the instances solved. One surprising observation is how much worse the quality of MP92's solutions is when compared to MinDS_2 (see Figure 3a). In some cases, decision sets learned by MP92 have 3 orders of magnitude more literals than the VBS decision sets and 1-2 orders of magnitude more literals than solutions computed by MinDS_2. On the other hand, these results indicate that approximate literal number minimization after learning a target decision set is feasible and does not degrade the performance of the overall procedure if done by enumerating a fixed number of MCSes. This is confirmed for the case of MinDS_1 (see Figure 3b). Note that efficient minimization of the total number of literals in the target decision sets is crucial given the requirement that they must be interpretable.

6 Conclusions & Research Directions

Decision (or rule) sets represent a promising approach for providing explanations in different ML settings. This paper shows that learning optimal decision sets raises a number of difficulties, related with overlap of rules, especially when the rules are associated with different classes. The paper conjectures that the exact solution for the learning problem of decision sets, while ensuring no overlap, is hard for the second level of the polynomial hierarchy. Moreover, the paper proposes a number of alternative problem formulations, all of which are shown to be hard for NP, and develops SAT-based solutions, relating with earlier work [19]. The experimental results, obtained on representative datasets, confirm the relevance of the approach, and yield a number of conclusions. Compared with earlier work [21], that exploits a variant of local search, the proposed SAT-based approach is not only far more accurate, but also remarkably more efficient. The results provide evidence that SAT-based learning of optimal decision sets

can handle practical datasets of interest, when the goal is to devise ML models that associate explanations with predictions.

The promising results in the paper motivate a number of lines of work, including proving (or disproving) the paper's main conjecture, developing more efficient propositional encodings, but also to consider other approaches that enable finding an optimal solution to the learning problem for decision sets. Also and as mentioned on the paper, the proposed approach can be adapted to study the problem from another perspective, i.e. by minimizing the total number of literals in a decision set instead the number of rules, or alternatively refer to multi-objective optimization. This approach may result in smaller and, thus, better interpretable solutions, in which case it would be appealing to compare it again heuristic rule-based classifiers targeting this same problem, e.g. CN2 [6, 7] and PRISM [5] among others. One additional natural line of work will be to extend the work to rule lists [2], but also to more expressive function representation languages, while preserving the ability to provide explanations for predictions.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.
2. E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, and C. Rudin. Learning certifiably optimal rule lists. In *KDD*, pages 35–44, 2017.
3. C. Bessiere, E. Hebrard, and B. O'Sullivan. Minimising decision tree size as combinatorial optimisation. In *CP*, pages 173–187, 2009.
4. A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*. IOS Press, 2009.
5. J. Cendrowska. PRISM: an algorithm for inducing modular rules. *International Journal of Man-Machine Studies*, 27(4):349–370, 1987.
6. P. Clark and R. Boswell. Rule induction with CN2: some recent improvements. In *EWSL*, pages 151–163, 1991.
7. P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.
8. W. W. Cohen. Fast effective rule induction. In *ICML*, pages 115–123, 1995.
9. DARPA. DARPA explainable Artificial Intelligence (XAI) program. https://www.darpa.mil/program/explainable-artificial-intelligence, 2016.
10. N. Eén and N. Sörensson. An extensible SAT-solver. In *SAT*, pages 502–518, 2003.
11. EU Data Protection Regulation. Regulation (EU) 2016/679 of the European Parliament and of the Council. http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679&from=en, 2016.
12. U. Feige, V. S. Mirrokni, and J. Vondrák. Maximizing non-monotone submodular functions. *SIAM J. Comput.*, 40(4):1133–1153, 2011.
13. J. Fürnkranz, D. Gamberger, and N. Lavrac. *Foundations of Rule Learning*. Springer, 2012.
14. J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques, 3rd edition*. Morgan Kaufmann, 2012.
15. J. R. Hauser, O. Toubia, T. Evgeniou, R. Befurt, and D. Dzyabura. Disjunctions of conjunctions, cognitive simplicity, and consideration sets. *Journal of Marketing Research*, 47(3):485–496, 2010.
16. ICML WHI Workshop. ICML workshop on human interpretability in machine learning. https://sites.google.com/view/whi2017/home, August 2017.
17. IJCAI XAI Workshop. IJCAI workshop on explainable artificial intelligence (XAI). http://home.earthlink.net/~dwaha/research/meetings/ijcai17-xai/, August 2017.

18. M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.
19. A. P. Kamath, N. Karmarkar, K. G. Ramakrishnan, and M. G. C. Resende. A continuous approach to inductive inference. *Math. Program.*, 57:215–238, 1992.
20. M. J. Kearns, M. Li, and L. G. Valiant. Learning boolean formulas. *J. ACM*, 41(6):1298–1328, 1994.
21. H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *KDD*, pages 1675–1684, 2016.
22. Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
23. J. Marques-Silva, J. Argelich, A. Graça, and I. Lynce. Boolean lexicographic optimization: algorithms & applications. *Ann. Math. Artif. Intell.*, 62(3-4):317–343, 2011.
24. J. Marques-Silva, F. Heras, M. Janota, A. Previti, and A. Belov. On computing minimal correction subsets. In *IJCAI*, pages 615–622, 2013.
25. C. Mencía, A. Previti, and J. Marques-Silva. Literal-based MCS extraction. In *IJCAI*, pages 1973–1979, 2015.
26. R. S. Michalski. On the quasi-minimal solution of the general covering problem. In *International Symposium on Information Processing*, pages 125–128, 1969.
27. T. M. Mitchell. *Machine learning*. McGraw-Hill, 1997.
28. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.
29. A. Morgado, A. Ignatiev, and J. Marques-Silva. MSCG: Robust core-guided MaxSAT solving. *JSAT*, 9:129–134, 2015.
30. NIPS IML Symposium. NIPS interpretable ML symposium. http://interpretable.ml/, December 2017.
31. R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining*, 10(1):36, Dec 2017.
32. F. Pedregosa and et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
33. J. R. Quinlan. Generating production rules from decision trees. In *IJCAI*, pages 304–307, 1987.
34. J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kauffmann, 1993.
35. R. L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.
36. M. Schaefer and C. Umans. Completeness in the polynomial-time hierarchy: A compendium. *SIGACT news*, 33(3):32–49, 2002.
37. E. Triantaphyllou. Inference of a minimum size boolean function from examples by using a new efficient branch-and-bound approach. *J. Global Optimization*, 5(1):69–94, 1994.
38. E. Triantaphyllou. *Data mining and knowledge discovery via logic-based methods: theory, algorithms, and applications*. Springer, 2010.
39. G. S. Tseitin. On the complexity of derivation in propositional calculus. *Studies in Constructive Mathematics and Mathematical Logic, Part II*, pages 115–125, 1968.
40. C. Umans, T. Villa, and A. L. Sangiovanni-Vincentelli. Complexity of two-level logic minimization. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(7):1230–1246, 2006.
41. T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille. Or's of And's for interpretable classification, with application to context-aware recommender systems. *CoRR*, abs/1504.07614, 2015.
42. T. Wang, C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl, and P. MacNeille. A bayesian framework for learning rule sets for interpretable classification. *Journal of Machine Learning Research*, 18:70:1–70:37, 2017.