Exact Rule Learning via Boolean Compressed Sensing

Dmitry M. Malioutov DMALIOUTOV@US.IBM.COM
Kush R. Varshney KRVARSHN@US.IBM.COM
Business Analytics and Mathematical Sciences, IBM Thomas J. Watson Research Center
1101 Kitchawan Road, Yorktown Heights, NY 10598 USA

Abstract

We propose an interpretable rule-based classification system based on ideas from Boolean compressed sensing. We represent the problem of learning individual conjunctive clauses or individual disjunctive clauses as a Boolean group testing problem, and apply a novel linear programming relaxation to find solutions. We derive results for exact rule recovery which parallel the conditions for exact recovery of sparse signals in the compressed sensing literature: although the general rule recovery problem is NP-hard, under some conditions on the Boolean 'sensing' matrix, the rule can be recovered exactly. This is an exciting development in rule learning where most prior work focused on heuristic solutions. Furthermore we construct rule sets from these learned clauses using set covering and boosting. We show competitive classification accuracy using the proposed approach.

1. Introduction

Organizations in many domains are turning to predictive analytics to support decision making (Davenport & Harris, 2007; Issenberg, 2013). However, with this growth, predictions and other outputs of machine learning algorithms are being presented to users who have limited analytics, data, and modeling literacy. Therefore, it is imperative to develop interpretable machine learning methods in order for predictions to have impact by being adopted and trusted by decision makers (Fry, 2011).

It has been frequently noted that rule sets composed of Boolean expressions with a small number of terms are the most well-received and trusted outputs (Liu & Li, 2005). As an example, IBM's SlamTracker reports keys to winning a tennis match as a conjunctive clause. The predictive decision rule for Federer defeating Murray in the 2013 Australian Open was:

- Win more than 59% of 4 to 9 shot rallies; AND
- Win more than 78% of points when serving at 30-30 or Deuce; AND
- Serve less than 20% of serves into the body.

Federer did not satisfy any of the three conditions and lost the match.

Similarly, in a management setting, the prediction for salespeople voluntarily resigning was presented by Varshney et al. (2012) as a Boolean expression:

- Job Role = Specialty Software Sales Rep; AND
- Base Salary $\leq$ 75,168; AND
- Months Since Promoted $>$ 13; AND
- Months Since Promoted $\leq$ 30; AND
- Quota-Based Compensation = FALSE.

In this paper, motivated by these consumability concerns, i.e. acceptance by users with limited knowledge of predictive modeling, we develop a new approach to interpretable supervised classification through rule learning based on Boolean compressed sensing. As opposed to most 'black-box' classification paradigms such as neural networks and kernel-based support vector machines (SVMs), Boolean rules can be easily interpreted by the practitioner and provide readily recognizable insight into the phenomenon of interest. Additionally, rules indicate actions that may be taken, e.g. in the salesforce example, increasing salary, offering promotion, or changing compensation plan to prevent resignation.

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

Supervised classification has been the subject of active research for decades. The earliest successful approaches to machine learning were decision list approaches that produce Boolean rule sets (Rivest, 1987; Clark & Niblett, 1989; Cohen, 1995), and decision tree approaches, which can be distilled into Boolean rule sets (Quinlan, 1987). Most such approaches attempt to learn rules to maximize criteria such as support, confidence, lift, conviction, Gini impurity, and information gain, some followed by heuristic pruning procedures. These original methods, which are still widely used by analytics practitioners today precisely due to their interpretability, rely on greedy, heuristic training procedures because rule learning is inherently combinatorial and notoriously difficult from the perspective of the theory of computation (Valiant, 1985; Kearns & Vazirani, 1994). Due to their heuristic nature, existing rule learning methods tend to have worse accuracy on many data sets than classification methods like SVMs that are based on optimizing a principled objective function.

There has been a renewed interest in rule learning that attempts to retain the interpretability advantages of rules, but changes the training procedures to be driven by optimizing an objective. Rückert & Kramer (2008) learn individual rules to maximize a quantity they define: margin minus variance, using quadratic programming. ENDER uses empirical risk to both learn rules and ensembles, but learns rules in a greedy manner (Dembczyński et al., 2010). Bertsimas et al. (2012); Letham et al. (2012) attempt to learn decision trees using mixed integer programming techniques. Jawanpuria et al. (2011) propose a hierarchical kernel learning approach. Set covering machines (SCM) formulate rule learning with an optimization objective similar to ours, but find solutions using a greedy heuristic (Marchand & Shawe-Taylor, 2002). Friedman & Popescu (2008) use optimization to combine basic Boolean clauses obtained from decision trees. The goal, despite the combinatorial nature of the problem, is to achieve classification accuracy on par with techniques that are more difficult to interpret; the contribution of our paper is in the same vein.

Compressed sensing and sparse signal recovery is a field where the core problem is also combinatorial, specifically NP-hard (Natarajan, 1995). However, recent dramatic breakthroughs have established that although the general problem is hard, for many specific instances (under conditions based on the restricted isometry property or incoherence), the hard combinatorial problem can be solved via a convex relaxation (Candès & Wakin, 2008). Moreover, recent work has highlighted parallels between sparse signal recovery and Boolean group testing (Gilbert et al., 2008).

In this paper, we reformulate a recent combinatorial relaxation developed for Boolean group testing problems that resembles the basis pursuit algorithm for sparse signal recovery in the context of learning classification rules (Malioutov & Malyutov, 2012).[1] The primary contribution of this work is showing that the problem of learning sparse conjunctive clause rules and sparse disjunctive clause rules from training samples can be represented as a group testing problem, and that we can apply the linear programming (LP) relaxation developed by Malioutov & Malyutov (2012) to solve it. Despite the fact that learning single clauses is NP-hard, we also establish conditions under which, if the data can be perfectly classified by a sparse Boolean rule, the relaxation recovers it exactly. To the best of our knowledge, this is the first work that combines compressed sensing ideas with classification rule learning to produce optimal rules.

Single conjunctive clauses have value, as illustrated by the tennis example on the previous page, but they are not as expressive as sets of rules. Single rules are building blocks for more complex rule-based classifiers. Two ways to combine conjunctive clause rules are as follows. The first is by constructing a rule set in disjunctive normal form (DNF), i.e. taking the OR operation of all of the rules, through a set covering approach: the first rule is learned on the entire training data set, the second rule is learned on the subset of data that does not satisfy the first rule, and so on (Rivest, 1987; Cohen, 1995); the recent work of Bertsimas et al. (2012) uses mixed integer programming to construct decision lists of this type. The second way is by treating each rule as a weak learner in an ensemble with the overall classification being a weighted vote of all of the individual rules (Cohen & Singer, 1999); the primary way to train the individual rules is sequentially via boosting.

In this paper, we apply these construction approaches to the individual rules found by our proposed exact rule learning solution, which allows us to handle a very general class of classification problems. On several data sets, we find that the proposed Boolean group testing approach has better accuracy than heuristic decision lists and has similar interpretability, both with single rule learning and rule set induction. We also find accuracy on par with weighted rule set induction via the C5.0 approach and better interpretability. Additionally, accuracy is not far off from the best non-interpretable classifiers, even providing the best accu-

[1] Other approaches to approximately solve group testing include greedy methods and loopy belief propagation; see references in Malioutov & Malyutov (2012).

racy on one data set.

The remainder of this paper is organized as follows. In Section 2 we describe how rule learning can be seen as a form of Boolean group testing. In Section 3, by alluding to ideas from compressed sensing, we present an LP relaxation for rule learning, and show that it can recover rules exactly under certain conditions. In Section 4 we suggest two approaches to combine rules into rule sets giving us a powerful classification algorithm, that we evaluate empirically in Section 5. We conclude with a summary and discussion in Section 6.

2. Clause learning as group testing

We start by discussing the problem of learning AND-clauses and OR-clauses, and use these clauses to build more expressive and powerful rule sets in Section 4. Despite the apparent simplicity of learning conjunctive clauses, this is in fact an NP-hard combinatorial optimization problem. Our approach builds upon the problems of group testing and sparse signal recovery.

2.1. The group testing problem

The group testing problem arose during the second world war when the United States was drafting citizens into the military and needed to test a large population for syphilis at low cost. One option would have been to test each of the n subjects' blood samples individually, but this would have been costly because most individuals did not have the disease. If the blood samples of several subjects were mixed together and then tested, the result on this mixed sample represented the OR of the subjects' disease state. By cleverly coming up with mixtures of subjects, it was possible to isolate the diseased subjects using $m \ll n$ tests. The mixture of subjects can be represented by an $m \times n$ Boolean matrix $\mathbf{A}$, where the rows represent different pools or mixtures and the columns represent different subjects. An entry $\{\mathbf{A}\}_{ij}$ is one if subject j is part of a pool i and zero otherwise.

The true diseased states of the subjects (which are not known when conducting the tests) can be represented by a vector $\mathbf{w} \in \{0,1\}^n$. The group testing then results in a Boolean vector $\mathbf{y} \in \{0,1\}^m$. We summarize the result of all m tests using the following notation which represents a Boolean matrix-vector product:

$$\mathbf{y} = \mathbf{A} \vee \mathbf{w}, \tag{1}$$

i.e.,

$$y_i = \bigvee_{j=1}^{n} \{\mathbf{A}\}_{ij} \wedge w_j. \tag{2}$$

In the presence of measurement errors,

$$\mathbf{y} = (\mathbf{A} \vee \mathbf{w}) \oplus \mathbf{n}, \tag{3}$$

where $\oplus$ is the XOR operator and $\mathbf{n}$ is a noise vector.

Once the tests have been conducted, the objective is to recover $\mathbf{w}$ from $\mathbf{A}$ and the measured $\mathbf{y}$. The recovery can be stated through the following combinatorial optimization problem:

$$\min \|\mathbf{w}\|_0 \quad \text{such that } \mathbf{y} = \mathbf{A} \vee \mathbf{w}, \tag{4}$$

where $\|\cdot\|_0$, the ℓ_0-quasinorm, counts the number of nonzero elements in its argument. The ones in the resulting $\mathbf{w}$ identify the sparse set of diseased subjects.

2.2. Classification rules as collections of diseased subjects

We have described group testing; now we show how the formulation can be adapted to rule-based classification. The problem setup of interest is standard binary supervised classification. We are given m labeled training samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ where the $\mathbf{x}_i \in \mathcal{X}$ are the features and the $y_i \in \{0,1\}$ are the Boolean labels. We would like to learn a function $\hat{y}(\cdot) : \mathcal{X} \to \{0,1\}$ that will accurately generalize to classify unseen, unlabeled feature vectors drawn from the same distribution as the training samples.

In rule-based classifiers, the clauses are made up of individual Boolean terms, e.g. 'months since promoted > 13.' Such a term can be represented by a function $a(\mathbf{x})$. To represent the full diversity and dimensions of the feature space $\mathcal{X}$, we have many such Boolean terms $a_j(\cdot) : \mathcal{X} \to \{0,1\}$, $j = 1, \ldots, n$. For each continuous dimension of $\mathcal{X}$, these terms may be comparisons to several suitably chosen thresholds. Then for each of the training samples, we can calculate the truth value for each of the terms, leading to an $m \times n$ truth table $\mathbf{A}$ with entries $\{\mathbf{A}\}_{ij} = a_j(\mathbf{x}_i)$.

Writing the true labels of the training set as a vector $\mathbf{y}$, we can write the same expression in the classification problem as in group testing (3): $\mathbf{y} = (\mathbf{A} \vee \mathbf{w}) \oplus \mathbf{n}$. In the classification problem, $\mathbf{w}$ is the binary vector to be learned that indicates the relevant features and thresholds. For reasons of interpretability, insight, and generalization, we would like $\mathbf{w}$ to be sparse, i.e. have mostly zero-valued entries. The $\mathbf{w}$ corresponding to the tennis example of Section 1 has three nonzero elements and the salesforce example five. With the desideratum of sparsity, the optimization problem to be solved is also the same as for group testing (4).

The nonzero coefficients thus directly specify a Boolean clause classification rule which can be applied

to new unseen data. This clause is a disjunctive OR-rule. In most of the rule-based classification literature, however, the learning of AND-clauses is preferred. This is easy to handle using DeMorgan's law. If we complement $\mathbf{y}$ and $\mathbf{A}$ prior to the learning, then we have:

$$\mathbf{y} = \mathbf{A} \wedge \mathbf{w} \quad \Leftrightarrow \quad \mathbf{y}^C = \mathbf{A}^C \vee \mathbf{w}. \tag{5}$$

Hence, our results apply to both OR-rules and AND-rules, and we focus on the conjunctive case for the remainder of the paper.

We have now described how to set up the problem of learning AND-clause classifiers. However, the ℓ_0 minimization (4), as we have written it, is a combinatorial optimization problem. The next section shows how $\mathbf{w}$ can be found by relating the problem to Boolean compressed sensing and using a linear programming relaxation. The section also provides theoretical results on exact recovery.

3. Learning AND-clauses through Boolean compressed sensing

Gilbert et al. (2008) and others have pointed out the similarity of group testing to another problem: compressed sensing—a signal processing technique for efficiently measuring and reconstructing signals. Both involve sparse signal recovery, but group testing is under a Boolean algebra instead of the typical algebra of real numbers encountered in compressed sensing. Due to their close connection, we show that it is possible to apply suitably modified, efficient LP relaxations from compressed sensing to solve the group testing problem, and now, due to Section 2, the classification rule learning problem.

3.1. Basis pursuit

In the compressed sensing and sparse signal recovery problem, the most popular technique for getting around the combinatorial ℓ_0 objective is to relax it using a convex alternative, the ℓ_1-norm. This relaxation, known as basis pursuit, results in the following optimization problem:

$$\min \|\mathbf{w}\|_1 \quad \text{such that } \mathbf{y} = \mathbf{A}\mathbf{w}, \tag{6}$$

where $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{A}$ are all real-valued and the product $\mathbf{A}\mathbf{w}$ is the standard matrix-vector product. This optimization problem (6) can be solved efficiently via LP solvers.

It has been shown that under certain conditions on the matrix $\mathbf{A}$, the ℓ_0 solution and the ℓ_1 solution are equivalent, and that it is even possible to use a random matrix for $\mathbf{A}$ and satisfy the conditions with high probability. The work of Malioutov & Malyutov (2012) extends the basis pursuit idea to Boolean algebras as we describe next.

3.2. Boolean compressed sensing

The challenge in compressed sensing is with the non-convexity of the ℓ_0-quasinorm. In the Boolean setting, the equation (1) is also not a linear operation and must also be dealt with if an LP relaxation is to be applied. If a vector $\mathbf{w}$ satisfies the constraint that $\mathbf{y} = \mathbf{A} \vee \mathbf{w}$, then it also satisfies the pair of ordinary linear inequalities:

$$\begin{aligned} \mathbf{A}_{\mathcal{P}}\mathbf{w} &\geq \mathbf{1}, \\ \mathbf{A}_{\mathcal{Z}}\mathbf{w} &= \mathbf{0}, \end{aligned} \tag{7}$$

where $\mathcal{P} = \{i | y_i = 1\}$ is the set of positive tests, $\mathcal{Z} = \{i | y_i = 0\}$ is the set of negative (or zero) tests, and $\mathbf{A}_{\mathcal{P}}$ and $\mathbf{A}_{\mathcal{Z}}$ are the corresponding subsets of rows of $\mathbf{A}$. The vectors $\mathbf{1}$ and $\mathbf{0}$ are all ones and all zeroes, respectively. These constraints can be incorporated into an LP.

Thus the Boolean ℓ_1 problem is:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} w_j \\ \text{s.t.} \quad & w_j \in \{0,1\},\ j = 1, \ldots, n \\ & \mathbf{A}_{\mathcal{P}}\mathbf{w} \geq \mathbf{1} \\ & \mathbf{A}_{\mathcal{Z}}\mathbf{w} = \mathbf{0}. \end{aligned} \tag{8}$$

Because of the Boolean integer constraint on the weights, the problem (8) is NP-hard. We can further relax the optimization to the following LP[2]:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} w_j \\ \text{s.t.} \quad & 0 \leq w_j \leq 1,\ j = 1, \ldots, n \\ & \mathbf{A}_{\mathcal{P}}\mathbf{w} \geq \mathbf{1} \\ & \mathbf{A}_{\mathcal{Z}}\mathbf{w} = \mathbf{0}, \end{aligned} \tag{9}$$

which is tractable. If non-integer w_j are found, we set them to one.

Slack variables may be introduced in the presence of errors, when there may not be any sparse rules producing the labels $\mathbf{y}$ exactly, but there are sparse rules that approximate $\mathbf{y}$ very closely. This is the typical

[2]Instead of using LP, one can find solutions greedily, as is done in the SCM, which gives a $\log(m)$ approximation. The same guarantee holds for LP with randomized rounding. Empirically, LP tends to find sparser solutions.

case in the supervised classification problem. The LP is then:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} w_j + \lambda \sum_{i=1}^{m} \xi_i \qquad (10) \\ \text{s.t.} \quad & 0 \le w_j \le 1,\ j = 1, \dots, n \\ & 0 \le \xi_i \le 1,\ i \in \mathcal{P} \\ & 0 \le \xi_i,\ i \in \mathcal{Z} \\ & \mathbf{A}_{\mathcal{P}}\mathbf{w} + \boldsymbol{\xi}_{\mathcal{P}} \ge \mathbf{1} \\ & \mathbf{A}_{\mathcal{Z}}\mathbf{w} = \boldsymbol{\xi}_{\mathcal{Z}}. \end{aligned}$$

The regularization parameter λ trades training error and the sparsity of $\mathbf{w}$.

3.3. Exact recovery guarantees

We now use some tools from combinatorial group testing (Dyachkov & Rykov, 1983; Du & Hwang, 2006) to establish results for exact recovery and recovery with small error probability in AND-clause learning via an LP relaxation. First we introduce some definitions used in group testing.

Definition 1 *We call a measurement matrix* $\mathbf{A}$ K-separating*, if Boolean sums of sets of* K *columns are all distinct.* $\mathbf{A}$ *is called* K-disjunct*, if the union of any* K *columns does not contain any other column.*

Note that the K-separating property for $\mathbf{A}$ is sufficient to allow exact recovery of $\mathbf{w}$ with up to K nonzero entries (Du & Hwang, 2006). However, finding the solution would in general require searching over all K-subsets out of n. The property of K-disjunctness, which can be viewed as a Boolean analog of spark, is a more restrictive condition which allows a dramatic simplification of the search: a simple algorithm that considers rows where $y_i = 0$ and sets all w_j where $\{\mathbf{A}\}_{ij} = 1$ to zero and leaves the other $w_j = 1$, is guaranteed to recover the correct solution. For non-disjunct matrices this simple algorithm finds feasible but suboptimal solutions. Note that any K-disjunct matrix is guaranteed to be K-separating.

We recall the simple proof from Malioutov & Malyutov (2012) that shows that LP relaxation in (9) recovers the correct solution for the group testing problem with K-disjunct $\mathbf{A}$.

Lemma 1 *Suppose there exists* $\mathbf{w}^*$ *with* K *nonzero entries and* $\mathbf{y} = \mathbf{A} \vee \mathbf{w}^*$*. If the matrix* $\mathbf{A}$ *is* K*-disjunct then LP solution* $\hat{\mathbf{w}}$ *in (9) recovers* $\mathbf{w}^*$*, i.e.* $\hat{\mathbf{w}} = \mathbf{w}^*$*.*

Proof. First, due to the K-disjunct property, $\mathbf{w}^*$ is a unique solution to $\mathbf{y} = \mathbf{A} \vee \mathbf{w}^*$ with up to K entries. Also, $\mathbf{w}^*$ is a feasible solution to the LP. Consider rows of $\mathbf{A}$ corresponding to positive tests, and columns of $\mathbf{A}$ which are not eliminated via the zero-rows. There are exactly K such columns, and the K-disjunct property implies that the matrix is full-rank (in the Euclidean sense). For each column $\mathbf{a}_j$ there is at least one nonzero entry i which does not appear in any other column. Since $y_i = 1$, and $\{\mathbf{A}\}_{ij} = 1$, we have $\hat{w}_j \ge 1$. Now $w_j^* = 1$ for all j not eliminated via zero-rows, so $\mathbf{w}^*$ must be the unique optimal solution of the LP. $\diamond$

Now to apply it to rule learning we start with classification problems with binary features. In this setting the matrix $\mathbf{A}$ simply contains the feature values.[3] A simple corollary of Lemma 1 is that if $\mathbf{A}$ is K-disjunct and there is an underlying error-free K-term AND-rule, then we can recover the rule exactly via our LP in (9).

A critical question is when can we expect our features to yield a K-disjunct matrix? We show that if we have enough samples to guarantee that each $K+1$ subset of features is well-sampled as we define below, then the matrix is K-disjunct. More formally,

Lemma 2 *Suppose that for each subset of* $K+1$ *features, among our* m *samples we find at least one example of each one of the possible binary* $(K+1)$*-patterns, then the matrix* $\mathbf{A}$ *is* K*-disjunct.*

Proof. Note that there are 2^{K+1} possible binary patterns for K features. Suppose that on the contrary the matrix is not K-disjunct. Without loss of generality, K-disjunctness fails for the first K columns covering the $(K+1)$-st one. Namely, columns $\mathbf{a}_1, \dots, \mathbf{a}_{K+1}$ satisfy $\mathbf{a}_{K+1} \subset \cup_{k=1}^{K} \mathbf{a}_k$. This is clearly impossible, since by our assumption the pattern $(0, 0, \dots, 0, 1)$ for our $K+1$ variables is among our m samples. $\diamond$

To interpret the lemma: if features are not strongly correlated, then for any fixed K, for large enough m we will eventually obtain all possible binary patterns. Using a simple union bound, for the case of uncorrelated equiprobable binary features, the probability that at least one of the K-subsets exhibits a non-represented pattern is bounded above by $\binom{n}{K} 2^K (1 - (1/2)^K)^m$. Clearly as $m \to \infty$ this bound approaches zero: with enough samples $\mathbf{A}$ is K-disjunct.

[3] In general it will contain the features and their complements as columns. However, with enough data, one of the two choices will be removed by zero-row elimination beforehand.

3.4. Recovery with small error probability

We now use approximate disjunctness (also known as a weakly-separating design) (Mazumdar, 2012; Malyutov, 1978) to develop less restrictive conditions, when we allow a small probability of error in recovery.

Definition 2 *We call an $m \times n$ measurement matrix $\mathbf{A}$ (ϵ,K)-disjunct, if out of $\binom{n}{K}$ K-subsets of columns of $\mathbf{A}$ at least a $(1-\epsilon)$-fraction of them satisfy the property that their union does not contain any other column of $\mathbf{A}$.*

Lemma 3 *If the matrix $\mathbf{A}$ is (ϵ, K)-disjunct then LP in (9) recovers the correct solution with probability at least $1-\epsilon$.*

The proof is a direct extension of our earlier proof.

We note that by allowing a small probability of error in recovery, i.e. approximate disjunctness, we can dramatically decrease the required number of samples. For example for deterministic designs there are known constructions of K-disjunct matrices of size $O(K^2 \log(n))$, whereas by allowing (ϵ, K)-disjunctness, there exist constructions requiring as few as $O(K^{3/2}\sqrt{\log(n/\epsilon)})$ measurements, and a non-constructive information-theoretic argument suggesting it could go to $O(K \log(n))$ (Mazumdar, 2012).

3.5. Continuous features

Now let us consider the case where we have continuous features. We discretize feature dimension x_j using thresholds $\theta_{j,1} \leq \theta_{j,2} \leq \cdots \leq \theta_{j,D}$ such that the columns of $\mathbf{A}$ corresponding to x_j are the outputs of Boolean indicator functions $I_{x_j \leq \theta_{j,1}}(\mathbf{x}), \ldots, I_{x_j \leq \theta_{j,D}}(\mathbf{x}), I_{x_j > \theta_{j,1}}(\mathbf{x}), \ldots, I_{x_j > \theta_{j,D}}(\mathbf{x})$. One choice of thresholds is empirical quantiles.[4] Note that the columns are not bins between the thresholds, but rather half-spaces defined by all threshold values.

This matrix, as defined above, is not disjunct because, e.g., $I_{x_j > \theta_{j,1}}(\mathbf{x}) \geq I_{x_j > \theta_{j,2}}(\mathbf{x})$. However, without loss of generality, for each feature we can remove all but one of the corresponding columns of $\mathbf{A}$. First, all of the columns that intersect zero-rows in $\mathbf{y}$ can be eliminated. Second, since the columns form a nested set, we can select the remaining column with the most nonzero entries without affecting the optimal value of the objective.[5] Through this reduction we are left with a simple classification problem with binary features, hence the results in Lemmas 1 and 3 carry through for continuous features, where we can recover the rules up to the threshold resolution.

[4] We could also use all the sample values as thresholds: while it would increase complexity, column generation could be used to mitigate it (Demiriz et al., 2002).

[5] Some of the thresholds will be violated by the optimal solution—the ones that are not violated are indistinguishable given the available data and our choice of thresholds.

4. Rule sets

In Section 3, we have shown how to efficiently solve the single AND-clause learning problem from training data by appealing to Boolean group testing and compressed sensing. In this section we discuss how to take these individual rules and combine them into rule sets.

The first rule set induction approach we consider, the set covering approach, also known as separate-and-conquer, is the outer loop of myriad learning algorithms (Fürnkranz, 1999). The differences in the algorithms are in how they learn individual conjunctive rules (our proposal here being to learn the individual rules through Boolean compressed sensing). After the first individual AND-rule is learned, the training samples for which the rule returns one are removed from the training set. Then a second individual rule is learned on the remaining training samples. The training samples classified as one are again removed from the training set, a third rule is learned, and the process continues. The final classification is a DNF, i.e. the OR of all of the learned AND-clauses.

The second rule set construction approach we consider is boosting. The result is a weighted rule set in which the individual rules vote to produce the final classification. The learning procedure is sequential, like set covering, but instead of removing training samples that are classified as one on each round, all samples are included on each round but weights are given to emphasize incorrectly classified samples. For a given round t, the objective of the LP becomes:

$$\min \quad \sum_{j=1}^{n} w_j + \lambda \sum_{i=1}^{m} d_{t,i}\xi_i, \tag{11}$$

where $d_{t,i} \geq 0$ is the weight applied to sample i and $\sum_i d_{t,i} = 1$. Cohen & Singer (1999) propose a form of boosting for rule sets that results in a voting that is interpretable and similar to set covering rule sets. We adopt the same boosting updates, as presented in Algorithm 1.

5. Empirical results

In this section, we first discuss some implementation notes and show an illustrative example of the proposed

Algorithm 1 Boosting Decision Rules

initialize: $d_{1,i} = \frac{1}{m}$.
for $t = 1$ **to** T **do**
  $\mathbf{w}^* \leftarrow$ LP with objective (11)
  $s_{\mathrm{TP}} \leftarrow \sum_{\{i|\hat{y}_{\mathbf{w}^*}(\mathbf{x}_i)=1, y_i=1\}} d_{t,i}$
  $s_{\mathrm{FP}} \leftarrow \sum_{\{i|\hat{y}_{\mathbf{w}^*}(\mathbf{x}_i)=1, y_i=0\}} d_{t,i}$
  $s_{\mathrm{T}} \leftarrow \sum_{\{i|y_i=1\}} d_{t,i}$
  $s_{\mathrm{F}} \leftarrow \sum_{\{i|y_i=0\}} d_{t,i}$
  if $\left(\sqrt{s_{\mathrm{TP}}} - \sqrt{s_{\mathrm{FP}}}\right)^2 > \left(\sqrt{s_{\mathrm{T}}} - \sqrt{s_{\mathrm{F}}}\right)^2$ **then**
    $\hat{C}_t \leftarrow \frac{1}{2} \ln\left(\frac{s_{\mathrm{TP}}+\epsilon}{s_{\mathrm{FP}}+\epsilon}\right)$
    $\hat{y}_t(\mathbf{x}) \leftarrow \hat{C}_t \hat{y}_{\mathbf{w}^*}(\mathbf{x})$
  else
    $\hat{C}_t \leftarrow \frac{1}{2} \ln\left(\frac{s_{\mathrm{T}}+\epsilon}{s_{\mathrm{F}}+\epsilon}\right)$
    $\hat{y}_t(\mathbf{x}) \leftarrow \hat{C}_t$
  end if
  $d_{(t+1),i} \leftarrow d_{t,i} / \exp((2y_i - 1)\hat{y}_t(\mathbf{x}))$
  normalize so that $\sum_i d_{(t+1),i} = 1$
end for
output: $\hat{y}(\mathbf{x}) = \begin{cases} 0, & \sum_t \hat{y}_t(\mathbf{x}) \leq 0 \\ 1, & \sum_t \hat{y}_t(\mathbf{x}) > 0 \end{cases}$

approach. Then we provide a comparative study of classification accuracy and rule set complexity.

5.1. Implementation notes

As discussed in Section 3.5, continuous features are approached using indicator functions on thresholds in both directions of comparison; in particular we use 10 quantile-based thresholds per continuous feature dimension. To solve the LP (10), we use SDPT3 version 4.0. We do not attempt to optimize the regularization parameter λ in this work, but leave it fixed at $\lambda = 1000$. We also do not attempt to optimize the number of rounds of boosting, but leave $T = 5$.

5.2. Illustrative example

We illustrate the types of sparse interpretable rules that are obtained using the proposed rule learner on the iris data set. We consider the binary problem of classifying iris versicolor from the other two species, setosa and virginica. Of the four features, sepal length, sepal width, petal length, and petal width, the rule that is learned, plotted in Fig. 1, involves only two features and three Boolean expressions:

- petal length $\leq$ 5.350 cm; AND
- petal width $\leq$ 1.700 cm; AND
- petal width $>$ 0.875 cm.

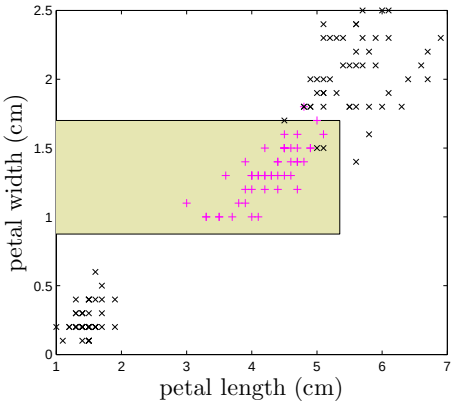

Figure 1. Decision rule learned to classify the iris species versicolor (magenta + markers) from the other two species (black × markers).

5.3. Classification performance comparisons

As an empirical study, we consider several interpretable classifiers: the proposed Boolean compressed sensing-based single rule learner (1Rule), the set covering approach to extend the proposed rule learner (RuSC), the boosting approach to extend the proposed rule learner (RuB), the decision lists algorithm in SPSS (DList), the C5.0 Release 2.06 algorithm with rule set option in SPSS (C5.0), and the classification and regression trees algorithm in Matlab's classregtree function (CART).[6]

We also consider several classifiers that are not interpretable: the random forests classifier in Matlab's TreeBagger class (TrBag), the k-nearest neighbor algorithm in SPSS (kNN), discriminant analysis of the Matlab function classify, and SVMs with radial basis function kernel in SPSS (SVM).

The data sets to which we apply these classification algorithms come from the UCI repository (Frank & Asuncion, 2010). They are all binary classification data sets with real-valued features. (We have not considered data sets with categorical-valued features in this study to allow comparisons to a broader set of classifiers; in fact, classification of categorical-valued features is a setting in which rule-based approaches excel.) The specific data sets are: Indian liver patient dataset (ILPD), Ionosphere (Ionos), BUPA liver disorders (Liver), Parkinsons (Parkin), Pima Indian diabetes (Pima), connectionist bench sonar (Sonar), blood transfusion service center (Trans), and breast cancer Wisconsin diagnostic (WDBC).

[6] We use IBM SPSS Modeler 14.1 and Matlab R2009a with default settings.

Table 1. Tenfold cross-validation test error on various data sets.

	1Rule	RuSC	RuB	DList	C5.0	CART	TrBag	kNN	Discr	SVM
ILPD	0.2985	0.2985	0.2796	0.3654	0.3053	0.3362	0.2950	0.3019	0.3636	0.3002
Ionos	0.0741	0.0712	0.0798	0.1994	0.0741	0.0997	0.0655	0.1368	0.1425	0.0541
Liver	0.4609	0.4029	0.3942	0.4522	0.3652	0.3768	0.3101	0.3101	0.3768	0.3217
Parkin	0.1744	0.1538	0.1590	0.2513	0.1641	0.1282	0.0821	0.1641	0.1641	0.1436
Pima	0.2617	0.2539	0.2526	0.3138	0.2487	0.2891	0.2305	0.2969	0.2370	0.2344
Sonar	0.3702	0.3137	0.3413	0.3846	0.2500	0.2837	0.1490	0.2260	0.2452	0.1442
Trans	0.2406	0.2406	0.2420	0.3543	0.2166	0.2701	0.2540	0.2286	0.3369	0.2353
WDBC	0.0703	0.0562	0.0562	0.0967	0.0650	0.0808	0.0422	0.0685	0.0404	0.0228

Table 1 gives tenfold cross-validation test errors for the various classifiers. Table 2 gives the average number of rules across the ten folds needed by the different rule-based classifiers to achieve those error rates.

It can be noted that our rule sets have better accuracy than decision lists on all data sets and our single rule has better accuracy than decision lists in all but one instance. On about half of the data sets, our set covering rule set has fewer rules than decision lists. Taking the number of rules as an indication of interpretability, we see that our set covering rule set has about the same level of interpretability as decision lists but with better classification accuracy. (We did not optimize the number of rules in boosting.) Even our single rule, which is very interpretable, typically has better accuracy than decision lists with more rules.

Compared to the C5.0 rule set, our proposed rule sets are much more interpretable because they have many fewer rules on average across the data sets considered. The accuracy of C5.0 and our rule sets is on par, as each approach has better accuracy on half of the data sets. The best performing algorithms in terms of accuracy are SVMs and random forests, but we see generally quite competitive accuracy with the advantage of interpretability by the proposed approach. On the ILPD data set, our boosting approach has the best accuracy among all ten classifiers considered.

Table 2. Tenfold average number of conjunctive clauses in rule set.

	1Rule	RuSC	RuB	DList	C5.0
ILPD	1.0	1.2	5.0	3.7	11.7
Ionos	1.0	4.1	5.0	3.7	8.4
Liver	1.0	3.5	5.0	1.1	15.3
Parkin	1.0	3.1	5.0	1.2	7.3
Pima	1.0	2.3	5.0	5.0	12.0
Sonar	1.0	3.9	5.0	1.0	10.4
Trans	1.0	1.2	5.0	2.3	4.3
WDBC	1.0	4.1	5.0	3.2	7.4

6. Conclusion

In this paper, we have developed a new optimization approach for learning decision rules based on compressed sensing ideas. The approach leads to a powerful rule-based classification system whose outputs are easy for human users to trust and draw insight from. In contrast to typical rule learners, the proposed approach is not heuristic. We prove theoretical results showing that exact rule recovery is possible through a convex relaxation of the combinatorial optimization problem under certain conditions.

In addition, through an empirical study, we have shown that the proposed algorithm is practical and leads to a classifier that has better accuracy than SPSS's decision lists, similar accuracy as C5.0's weighted rule set as well as classification and regression trees, and accuracy not far from SVMs and random forests. In fact, on the ILPD data set, the boosting version of the rule set provides the best accuracy overall. This accuracy is achieved while maintaining similar interpretability as SPSS's decision lists and having better interpretability than all other methods. Even our single rule classifier without a full rule set can be used in practical situations.

In future work we plan to conduct a detailed study of the practical aspects of our algorithm, such as optimizing the number of rounds of boosting and setting the error parameters, to further improve its performance. Furthermore, we plan to evaluate the performance of our approach on much larger data sets from real-world projects, compare both the accuracy and the interpretability, and evaluate feedback from typical users. We also plan to investigate the generalization performance of our approach from the statistical learning theory perspective: as it is based on principled convex relaxations, such analysis may be more amenable than for heuristic rule learning approaches.

References

Bertsimas, D., Chang, A., and Rudin, C. An integer optimization approach to associative classification. In *Adv. Neur. Inf. Process. Syst. 25*, pp. 269–277. 2012.

Candès, E. J. and Wakin, M. B. An introduction to compressive sampling. *IEEE Signal Process. Mag.*, 25(2):21–30, Mar. 2008.

Clark, P. and Niblett, T. The CN2 induction algorithm. *Mach. Learn.*, 3(4):261–283, Mar. 1989.

Cohen, W. W. Fast effective rule induction. In *Proc. Int. Conf. Mach. Learn.*, pp. 115–123, Tahoe City, CA, Jul. 1995.

Cohen, W. W. and Singer, Y. A simple, fast, and effective rule learner. In *Proc. Nat. Conf. Artif. Intell.*, pp. 335–342, Orlando, FL, Jul. 1999.

Davenport, T. H. and Harris, J. G. *Competing on Analytics: The New Science of Winning*. Harvard Business School Press, Boston, MA, 2007.

Dembczyński, K., Kotłowski, W., and Słowiński, R. ENDER: A statistical framework for boosting decision rules. *Data Min. Knowl. Disc.*, 21(1):52–90, Jul. 2010.

Demiriz, A., Bennett, K. P., and Shawe-Taylor, J. Linear programming boosting via column generation. *Mach. Learn.*, 46(1–3):225–254, Jan. 2002.

Du, D.-Z. and Hwang, F. K. *Pooling Designs and Nonadaptive Group Testing: Important Tools for DNA Sequencing*. World Scientific, Singapore, 2006.

Dyachkov, A. G. and Rykov, V. V. A survey of superimposed code theory. *Probl. Control Inform.*, 12(4): 229–242, 1983.

Frank, A. and Asuncion, A. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2010.

Friedman, J. H. and Popescu, B. E. Predictive learning via rule ensembles. *Ann. Appl. Stat.*, 2(3):916–954, Sep. 2008.

Fry, C. Closing the gap between analytics and action. *INFORMS Analytics Mag.*, 4(6):4–5, Nov./Dec. 2011.

Fürnkranz, J. Separate-and-conquer rule learning. *Artif. Intell. Rev.*, 13(1):3–54, Feb. 1999.

Gilbert, A. C., Iwen, M. A., and Strauss, M. J. Group testing and sparse signal recovery. In *Asilomar Conf. Signals Syst. Comp. Conf. Record*, pp. 1059–1063, Pacific Grove, CA, Oct. 2008.

Issenberg, S. How president Obama's campaign used big data to rally individual voters. *MIT Tech. Rev.*, 116(1):38–49, Jan./Feb. 2013.

Jawanpuria, P., Nath, J. S., and Ramakrishnan, G. Efficient rule ensemble learning using hierarchical kernels. In *Proc. Int. Conf. Mach. Learn.*, pp. 161–168, Bellevue, WA, Jun.–Jul. 2011.

Kearns, M. J. and Vazirani, U. V. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, 1994.

Letham, B., Rudin, C., McCormick, T. H., and Madigan, D. Building interpretable classifiers with rules using Bayesian analysis. Technical Report 609, Dept. Stat., Univ. Washington, Dec. 2012.

Liu, J. and Li, M. Finding cancer biomarkers from mass spectrometry data by decision lists. *J. Comp. Bio.*, 12(7):971–979, Sep. 2005.

Malioutov, D. and Malyutov, M. Boolean compressed sensing: LP relaxation for group testing. In *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pp. 3305–3308, Kyoto, Japan, Mar. 2012.

Malyutov, M. The separating property of random matrices. *Math. Notes*, 23(1):84–91, 1978.

Marchand, M. and Shawe-Taylor, J. The set covering machine. *J. Mach. Learn. Res.*, 3:723—746, Dec. 2002.

Mazumdar, A. On almost disjunct matrices for group testing. In *Proc. Int. Symp. Alg. Comput.*, pp. 649–658, Taipei, Taiwan, Dec. 2012.

Natarajan, B. K. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24(2):227–234, 1995.

Quinlan, J. R. Simplifying decision trees. *Int. J. Man-Mach. Studies*, 27(3):221–234, Sep. 1987.

Rivest, R. L. Learning decision lists. *Mach. Learn.*, 2 (3):229–246, Nov. 1987.

Rückert, U. and Kramer, S. Margin-based first-order rule learning. *Mach. Learn.*, 70(2–3):189–206, Mar. 2008.

Valiant, L. G. Learning disjunctions of conjunctions. In *Proc. Int. Joint Conf. Artif. Intell.*, pp. 560–566, Los Angeles, CA, Aug. 1985.

Varshney, K. R., Rasmussen, J. C., Mojsilović, A., Singh, M., and DiMicco, J. M. Interactive visual salesforce analytics. In *Proc. Int. Conf. Inf. Syst.*, Orlando, FL, Dec. 2012.