Generalized Linear Rule Models

Dennis Wei [1] Sanjeeb Dash [1] Tian Gao [1] Oktay Günlük [1]

Abstract

This paper considers generalized linear models using rule-based features, also referred to as rule ensembles, for regression and probabilistic classification. Rules facilitate model interpretation while also capturing nonlinear dependences and interactions. Our problem formulation accordingly trades off rule set complexity and prediction accuracy. Column generation is used to optimize over an exponentially large space of rules without pre-generating a large subset of candidates or greedily boosting rules one by one. The column generation subproblem is solved using either integer programming or a heuristic optimizing the same objective. In experiments involving logistic and linear regression, the proposed methods obtain better accuracy-complexity trade-offs than existing rule ensemble algorithms. At one end of the trade-off, the methods are competitive with less interpretable benchmark models.

1. Introduction

Decision rules have served as important building blocks for supervised learning models. They are often combined via logical operations into decision lists (Rivest, 1987; Angelino et al., 2017; Yang et al., 2017) and rule sets (Lakkaraju et al., 2016; Wang et al., 2017), models whose appeal stems from the human-interpretability of their constituent rules.

In this paper, we consider models that are linear combinations of decision rules, also referred to as rule ensembles, within the framework of generalized linear models (GLM) (McCullagh & Nelder, 1989). Rule ensembles retain interpretability while allowing modelling flexibility since rules are able to capture nonlinear dependences and interactions. Our problem formulation accordingly trades off predictive accuracy against the complexity of the rule ensemble, measured in terms of the number of rules as well as their lengths, i.e., the number of elementary conditions on individual variables in the conjunction. We make use of GLMs to address the real-valued prediction tasks of regression and probabilistic classification, where class probability estimates are desired.

The main challenge in learning rule-based models is due to the exponential size of the space of rules, corresponding to all possible conjunctions of the input features. Predominant approaches include pre-selecting a (often large) subset of candidate rules and greedy optimization in which rules are added one by one but not revised. The former includes optimization methods that select from among the candidates (Friedman & Popescu, 2008; Lakkaraju et al., 2016; Wang et al., 2017) while the latter includes sequential covering (Cohen, 1995; Clark & Boswell, 1991; Fürnkranz et al., 2014) and boosting.

Our main contribution herein is to propose an approach that avoids both of the above alternatives. The technique of column generation (CG) is used to intelligently search the space of rules and produce useful ones as needed, as opposed to using a large, fixed set of candidates. Instead of boosting, a GLM is re-fit as rules are generated, which allows existing rules to be reweighted and even discarded. The CG subproblem is formulated as an integer program (not of exponential size) and is solved using either integer programming or a heuristic that targets the same objective. We also discuss a non-CG algorithm that uses only first-degree rules, i.e., those with a single condition, and optionally numerical features as-is. This latter algorithm produces a kind of generalized additive model (GAM) (Hastie & Tibshirani, 1990).

Experiments are presented involving the two most common cases of GLMs, logistic and linear regression. The proposed methods are seen to yield better performance-complexity trade-offs than existing rule ensemble algorithms. At the performance-maximizing end of the trade-off, the methods are competitive with less interpretable benchmark models such as tree ensembles and nonlinear support vector machines (SVM). The trade-off curves also suggest that substantially simpler models are often available at a relatively small cost in performance.

[1]IBM Research, Yorktown Heights, NY, USA. Correspondence to: Dennis Wei <dwei@us.ibm.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

1.1. Related Work

Of the rule ensemble algorithms that have been proposed, the most closely related to the current proposal is RuleFit (Friedman & Popescu, 2008), which also fits linear models to a set of rules. This set however is predetermined by first inducing a large number (hundreds) of decision trees from the data and then extracting rules corresponding to nodes of the trees. As will be seen in Section 5, this approach yields more complex rule ensembles for the same performance. Rückert & Kramer (2006) also propose fitting linear models to an increasing set of rules but the rules are generated in an unsupervised manner.

Boosting algorithms, which do not modify previously added rules, are represented by SLIPPER (Cohen & Singer, 1999), MLRules and ENDER (Dembczyński et al., 2010). Other algorithms include HKL (Jawanpuria et al., 2011), which is also able to effectively optimize over an exponential number of conjunctions but relies on a regularizer with a special group structure. While often cited as a rule ensemble method, Weiss & Indurkhya (2000) actually learn ensembles of disjunctive normal forms (DNF), which are more akin to tree ensembles.

Even though branch-and-bound and CG have been used before in ML, e.g., for boosting (Demiriz et al., 2002) and in particular for rule learning (Angelino et al., 2017; Rudin & Ertekin, 2018), we believe our work is the first application of CG to a nonlinear optimization problem in ML. Our CG approach is inspired by its recent use in learning disjunctive/conjunctive normal form rules (Dash et al., 2018), with similar benefits. While that work applied CG to a linear program, here we have a (convex) nonlinear problem. For nonlinear problems, even though the general framework has been discussed for OR problems (Garcia et al., 2003), there are only a few practical applications, mostly nonlinear variants of the VRP (Borndörfer et al., 2013; Fortz et al., 2010). We also note that whether CG can be successfully applied to MINLPs has been posed as an open problem in a 2018 Dagstuhl Seminar (Bonami et al., 2018).

2. Generalized Linear Rule Models

We consider the standard supervised learning problem of predicting a target variable $Y \in \mathcal{Y}$ using input features $\mathbf{X} = (X_1, \dots, X_d)$, given a training dataset of i.i.d. samples $(\mathbf{x}_i, y_i)$, $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$, $i = 1, \dots, n$. The output space $\mathcal{Y}$ may be discrete or continuous. We assume that all features X_j have been binarized to take values in $\{0, 1\}$. For categorical features, this is achieved through the usual "one-hot" coding into indicators $X_j = x$ for all categories x as well as their negations $X_j \neq x$. Numerical features are binarized through bi-directional comparisons to a set of thresholds, e.g., $X_j \leq 1$, $X_j \leq 2.3$ and $X_j > 1$, $X_j > 2.3$. Further details on binarization are given in Section 5.

In this section, we recall aspects of generalized linear models (GLMs) and introduce notation needed to define them over a feature space of rules. Let $\mathcal{K}$ denote the set of conjunctions of $\mathbf{X}$ to be considered. We defer discussion of the choice of $\mathcal{K}$ to Section 3 but note that it is not necessary to limit its size. Denote by A_k the variable corresponding to conjunction $k \in \mathcal{K}$, and $a_{ik} \in \{0, 1\}$ the value taken by A_k in instance i. Let $k = 0$ be the index of the empty conjunction $A_0 \equiv 1$.

For a GLM, we posit that Y conditioned on $\mathbf{X}$ follows an exponential family distribution given by

$$p_{Y \mid \mathbf{X}}(y \mid \mathbf{x}) = h(y) \exp\left(\eta y - \Phi(\eta)\right), \tag{1}$$

where the canonical parameter η is a linear combination of the conjunctions A_k of $\mathbf{X}$,

$$\eta = \sum_{k \in \mathcal{K}} \beta_k A_k, \tag{2}$$

and $\Phi(\eta)$ is the log-partition function. The terms with nonzero coefficients β_k define an ensemble of rules mapping conjunctions to real values β_k, which are then linearly combined. The distribution may have parameters in addition to η (e.g., the variance in the Gaussian case) but these are either assumed known or their estimation can be separated from that of η. The prediction function is given by the conditional mean of Y,

$$\hat{y}(\mathbf{X}) = \mathbb{E}[Y \mid \mathbf{X}] = \Phi'(\eta), \tag{3}$$

where the second equality holds for (1).

The coefficients β_k in (2) are determined by minimizing the negative log-likelihood corresponding to (1) on the training data. Since $\mathcal{K}$ is potentially large, a sparse solution is essential and is obtained through ℓ_1 regularization. The optimization problem is therefore

$$\min_{\beta} \; \frac{1}{n} \sum_{i=1}^{n} \left[\Phi\left(\sum_{k \in \mathcal{K}} \beta_k a_{ik} \right) - y_i \sum_{k \in \mathcal{K}} \beta_k a_{ik} \right] + \sum_{k \in \mathcal{K}} \lambda_k |\beta_k|. \tag{4}$$

The factor $h(y)$ in (1) is not a function of β and is omitted. Each regularization parameter λ_k depends on the number of literals of conjunction k as specified later. It is a property of exponential families that the log-partition function $\Phi(\eta)$ is convex. By affine composition property of convex functions the problem (4) is therefore convex.

We specialize the foregoing to the two most common cases of GLMs, logistic and linear regression. For logistic regression, the log-partition function $\Phi(\eta) = \log(1 + e^{\eta})$. Substituting this into (4), the quantity in square brackets becomes the familiar expression

$$\log\left(1 + \exp\left((-1)^{y_i} \sum_{k \in \mathcal{K}} \beta_k a_{ik}\right)\right), \tag{5}$$

where $y_i \in \{0, 1\}$. For linear regression, $\Phi(\eta) = \eta^2/2$ and the bracketed quantity becomes (after adding back $y_i^2/2$)

$$\frac{1}{2}\Big(y_i - \sum_{k \in \mathcal{K}} \beta_k a_{ik}\Big)^2. \tag{6}$$

3. Model Instantiations

We discuss two instantiations of generalized linear rule models (GLRM). In Section 3.1, the set of conjunctions $\mathcal{K}$ is restricted to first-degree or singleton conjunctions, i.e., those with a single condition on an individual feature. Section 3.2 considers the general case with no restriction on $\mathcal{K}$.

3.1. Generalized Additive Model Using First-Degree Rules

In the first case, the conjunctions A_k correspond to the binarized features X_j themselves. In terms of the original unbinarized features, conditions are placed on only one feature at a time and so the resulting GLRM is free of interaction terms. On the other hand, if features are binarized as discussed at the beginning of Section 2, then the GLRM is a type of generalized additive model (GAM), i.e., a sum of univariate functions. For numerical features, first-degree rules correspond to step functions, which can be linearly combined into arbitrary piecewise-constant functions with discontinuities at the binarization thresholds. For categorical features, any function can be realized.

Friedman & Popescu (2008) discuss a similar type of model obtained by restricting their decision trees to depth 1 (decision stumps). There are two differences however. First, the singleton rules herein are systematically enumerated as opposed to generated by a randomized tree induction procedure. Second, from every pair of complementary singleton rules (e.g., $X_j \leq 1$, $X_j > 1$), we remove one member as otherwise the pair together with the empty conjunction $A_0 \equiv 1$ are collinear. The results in Section 5 suggest that this removal of linearly dependent rules contributes toward sparser, simpler rule ensembles.

In addition to first-degree rules, following Friedman & Popescu (2008) we may also include in the feature space any numerical features as they are, without binarization. This model variant is also evaluated in Section 5.

3.2. General Rule Ensemble Using Column Generation

We now let $\mathcal{K}$ be the set of all possible conjunctions of $\mathbf{X}$ to obtain rule ensembles with no restrictions. Since $\mathcal{K}$ is now exponentially large, it is intractable even to enumerate the variables in problem (4) (unless the feature dimension d is very small). We exploit the technique of column generation (CG) to tackle this problem.

Column generation was originally developed to solve linear programs (LPs) with a very large number of columns (Gilmore & Gomory, 1961; Conforti et al., 2014). Using the fact that optimal solutions of LPs are sparse, the main idea is to first solve a restricted problem with a small number of candidate columns and then generate some of the missing columns based on the optimal dual solution of this restricted problem. Using the dual solution, one can compute the marginal benefit (or, partial derivative) of introducing a missing column to the restricted problem. If partial derivative for the most promising missing column is non-negative, then the procedure terminates. The crucial component of this approach is to formulate a column generation problem that can search through all of the missing columns without complete enumeration. We next describe how to adopt this idea to solve the generalized model (4). To derive the column generation subproblem, it is helpful to express β_k as $\beta_k = \beta_k^+ - \beta_k^-$ for $\beta_k^+, \beta_k^- \geq 0$. Problem (4) becomes

$$\min_{\beta^+,\beta^- \geq 0} \; \frac{1}{n}\sum_{i=1}^{n}\Bigg[\Phi\Bigg(\sum_{k\in\mathcal{K}}(\beta_k^+ - \beta_k^-)a_{ik}\Bigg) - y_i\sum_{k\in\mathcal{K}}(\beta_k^+ - \beta_k^-)a_{ik}\Bigg] + \sum_{k\in\mathcal{K}}\lambda_k(\beta_k^+ + \beta_k^-). \tag{7}$$

Suppose that a restricted version of (7) has been solved for a subset $\mathcal{S} \subset \mathcal{K}$ of the set of conjunctions, yielding $\beta_k^\pm = (\beta_k^\pm)^*$ for $k \in \mathcal{S}$. We extend this to a solution for (7) by setting $\beta_k^\pm = 0$ for $k \notin \mathcal{S}$ and wish to determine the optimality of the extended solution. Since (7) is also a convex problem with non-negativity constraints, a necessary and sufficient condition of optimality is for the partial derivatives of the objective with respect to $\beta_k^\pm$ to be zero if $\beta_k^\pm > 0$ and non-negative if $\beta_k^\pm = 0$. This condition is true for $k \in \mathcal{S}$ due to optimality for the restricted problem. For $k \notin \mathcal{S}$, $\beta_k^\pm = 0$ and we are thus required to check non-negativity of the derivatives.

The partial derivative w.r.t. β_k^+ of the objective in (7) is

$$\frac{1}{n}\sum_{i=1}^{n}\Bigg[\Phi'\Bigg(\sum_{k'\in\mathcal{K}}\beta_{k'}a_{ik'}\Bigg)a_{ik} - y_i a_{ik}\Bigg] + \lambda_k = \frac{1}{n}\sum_{i=1}^{n}\big(\hat{y}(\mathbf{x}_i) - y_i\big)a_{ik} + \lambda_k = \frac{1}{n}\sum_{i=1}^{n} r_i a_{ik} + \lambda_k,$$

using (2) and (3) to obtain the first equality and defining the prediction *residual* $r_i = \hat{y}(\mathbf{x}_i) - y_i$ in the second. The partial derivative with respect to β_k^- is the same except that the residuals are negated. Non-negativity of all partial derivatives can thus be determined by solving the pair of problems

$$\min_{k\in\mathcal{K}} \pm\frac{1}{n}\sum_{i=1}^{n} r_i a_{ik} + \lambda_k. \tag{8}$$

If both optimal values in (8) are non-negative, then all derivatives are indeed non-negative and we conclude that the extended solution is optimal for (7). On the other hand, if the objective in (8) is negative for some $k \notin \mathcal{S}$ and the '$+$' sign (say), then the partial derivative w.r.t. β_k^+ is negative and β_k^+ can be added to the restricted problem (i.e., k added to $\mathcal{S}$) to potentially improve the current solution.

We now use the fact that $\mathcal{K}$ is a set of conjunctions to avoid solving (8) by enumeration. This involves encoding a conjunction by the set of participating features and relating the values a_{ik} to the feature values x_{ij}. Let $z_j \in \{0,1\}$ represent whether feature j is selected in a conjunction. We assume that the regularization parameter λ_k is an affine function of the degree of the conjunction, $\lambda_k = \lambda_0 + \lambda_1 \sum_j z_j$ with $\lambda_0,\ \lambda_1 \geq 0$ (other affine functions of z_j are possible). Define $\bar{x}_{ij} = 1 - x_{ij}$, $\mathcal{I}_+ = \{i : r_i > 0\}$, and $\mathcal{I}_- = \{i : r_i < 0\}$. Then (8) can be reformulated as

$$\begin{aligned} \min_{a,z} \quad & \pm \frac{1}{n} \sum_{i=1}^{n} r_i a_i + \lambda_0 + \lambda_1 \sum_{j=1}^{d} z_j \\ \text{s.t.} \quad & a_i + \sum_{j=1}^{d} \bar{x}_{ij} z_j \geq 1, \quad a_i \geq 0, \qquad i \in \mathcal{I}_+ \\ & a_i + z_j \leq 1, \qquad i \in \mathcal{I}_-, \quad j : \bar{x}_{ij} = 1 \\ & z_j \in \{0,1\}, \qquad j = 1, \dots, d, \end{aligned} \tag{9}$$

where the subscript k has been dropped in favor of encoding by $\{z_j\}$. The constraints in (9) ensure that a_i acts as the conjunction of the selected x_{ij}'s. For $i \in \mathcal{I}_+$, $a_i = 1$ only if all selected features have $x_{ij} = 1$ ($\bar{x}_{ij} z_j = 0\ \forall j$), otherwise $a_i = 0$ since $r_i a_i$ is minimized in the objective. For $i \in \mathcal{I}_-$, $r_i < 0$, a_i is maximized, and the corresponding constraint enforces the same behavior for a_i.

Therefore, our column generation algorithm alternates between solving the restricted log-likelihood problem (4) and searching for new columns by solving (9) for both signs. We initialize the restricted set $\mathcal{S}$ to be the set of first-degree rules discussed in Section 3.1, optionally including original numerical features as well. The algorithm terminates with a certificate that problem (4) is solved to optimality if the optimal value of problem (9) is non-negative for both signs. For practical reasons we also have a secondary termination criteria that depends on the total number of column generation iterations and the total CPU time spent.

This algorithm is guaranteed to terminate in *finite* time as there are only a finite number of candidate columns (conjunctions) and the column generation procedure would not generate the same conjunction more than once as the partial derivative of the conjunctions in the restricted problem are guaranteed to be non-negative. We note that finite termination is not guaranteed for models with different regularization parameters, for example for ℓ_2-regularization, as repeating a conjunction with a non-zero weight would improve the optimal value of (7) by simply splitting the weight of an original conjunction into two equal parts. After splitting the first term in (7) would stay the same whereas the regularization penalty would decrease.

Once the column generation algorithm terminates, we solve the log-likelihood problem (4) one last time to de-bias the solution. In this final run we restrict conjunctions to the ones with $\beta_k^\pm > 10^{-5}$ in the last round and we drop the regularization term in the objective.

4. Column Generation Approaches

The column generation subproblem (9) is solved using either integer programming (IP) or a heuristic algorithm. We have implemented our column generation procedure in Java using LibLinear (Fan et al., 2008) to solve the regularized logistic regression problem (4) and using Cplex callable library (version 12.7.1) to solve the integer program (9) for column generation. As LibLinear package only allows simple ℓ_1-regularization (as opposed to using different weights λ_k that depend on the complexity of the conjunction k), we scale the a_{ik} values by $1/\lambda_k$.

The proposed heuristic algorithm performs a limited search of the rule space, proceeding in order of increasing conjunction degree $D = \sum_j z_j$ from 1 up to a maximum $D_{\max}$. To describe the heuristic, we define the children of a conjunction as those conjunctions that involve one additional feature, i.e., have one additional $z_j = 1$ in terms of the representation in (9). At each degree D, only those conjunctions that are children of a chosen parent (of degree $D-1$) are evaluated. These children are evaluated for two purposes: 1) To determine whether any improve upon the incumbent solution, defined as the best solution observed thus far; 2) to select a parent conjunction for the next highest degree. Evaluation for the first purpose is based only on the objective value achieved in (9), while evaluation for the second also considers a lower bound on future objective values, as discussed next.

We illustrate the objective value and lower bound calculations for degree 1 conjunctions, i.e., children of the initial empty conjunction. Higher degrees are analogous. Objective values of children can be computed via their increments and decrements relative to the value of the parent. For $D = 1$, setting $z_j = 1$ forces $a_i = 0$ for i such that $x_{ij} = 0$ ($\bar{x}_{ij} = 1$). The change in value of child j is thus

$$\Delta v(j) = \lambda_1 - \sum_{i \in \mathcal{I}_+} r_i \bar{x}_{ij} - \sum_{i \in \mathcal{I}_-} r_i \bar{x}_{ij}. \tag{10}$$

A lower bound on future objective values resulting from setting $z_j = 1$, i.e., values of descendants of child j, can be obtained by optimistically assuming that with the addition of one more feature (at a further cost of λ_1), all positive r_i can be eliminated from the objective function in (9) while no negative r_i are eliminated beyond those due to setting $z_j = 1$ itself. Expressed in the same relative terms as (10), this lower bound is

$$\mathrm{LB}(j) = 2\lambda_1 - \sum_{i \in \mathcal{I}_+} r_i - \sum_{i \in \mathcal{I}_-} r_i \bar{x}_{ij}. \tag{11}$$

To determine the parent for the next degree, we first eliminate all children of the current parent whose lower bounds $LB(j)$ are not less than the value of the incumbent solution, since these cannot lead to improvement. Any remaining children are evaluated using the average of $\Delta v(j)$ and $\mathrm{LB}(j)$ and the child with the lowest such average is selected. The motivation is to consider not only the children's current values but also a crude but easily computed estimate of potential future values. Other convex combinations of $\Delta v(j)$ and $\mathrm{LB}(j)$ have not been explored.

Once a new parent conjunction is chosen, corresponding to setting a $z_j = 1$, two operations are performed to reduce the dimensions of the problem and to render it in the same form as for $D = 1$. First, indices (rows) i where $\bar{x}_{ij} = 1$ are removed since a_i is forced to zero as noted above. Second, setting $z_j = 1$ may make other features j' redundant and these can be removed (by setting $z_{j'} = 0$).

We have explored additional variations of the heuristic algorithm as discussed in the supplementary material (SM).

5. Numerical Evaluation

We report on numerical experiments involving both logistic regression (5) (i.e., classification) and linear regression (6). We tested the 4 methods discussed in Section 3: logistic/linear regression on singleton rules (without CG, abbreviated LR1), logistic/linear regression on general rules (with CG, abbreviated LRR), and the same two with the addition of any numerical features originally present in the data (LR1N, LRRN). Column generation is done using the heuristic in Section 4 but we also report a preliminary result using an IP version of LRR (LRRI). We compared these methods to RuleFit (Friedman & Popescu, 2008), which is the closest existing method, and specifically a Python implementation[1] because the original R code is no longer supported. We also tried to test HKL (Jawanpuria et al., 2011) but encountered a no longer supported toolbox as well as numerical problems. Beyond rule ensembles, we also compared to gradient boosted classification/regression trees (GBT) and support vector machines (SVM) with radial basis function (RBF) kernels. These are less interpretable models intended to provide a benchmark for prediction performance. 10-fold cross-validation (CV) is used to estimate all test performance metrics.

Categorical and numerical features were binarized as described at the beginning of Section 2, using sample deciles as thresholds for numerical features. To control for the effect of discretization, numerical features were discretized using the same quantile thresholds for all rule- and tree-based methods in the comparison. Excluded from this treatment are SVMs and the variant of RuleFit that uses numerical features in addition to rules (abbreviated RuleFitN).

5.1. Classification

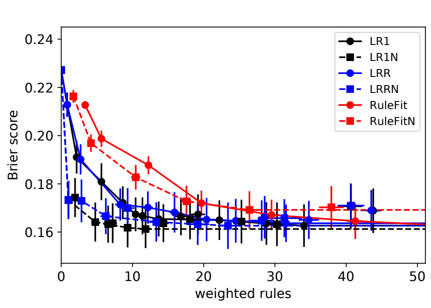
For classification, we used the same 16 datasets considered in (Dash et al., 2018), which also appeared in other recent works on rule-based models (Su et al., 2016; Wang et al., 2017). One of these datasets comes from the recent FICO Explainable Machine Learning Challenge (FICO, 2018).

In the first experiment, we evaluated the performance-complexity trade-offs of the four proposed methods (LR1, LRR, LR1N, LRRN) as well as RuleFit. For performance metrics, we report both accuracy and Brier score (i.e., mean squared error (MSE)), the latter a well-known metric for probabilistic outputs (Hernández-Orallo et al., 2012) as produced by logistic regression-based models. To measure rule ensemble complexity, we consider not only the number of rules (with nonzero coefficients) but also their lengths in terms of number of conditions. Specifically we define the *weight* of a rule similarly to the regularization parameter λ_k as $1 + w \sum_j z_j$, where we take $w = 0.2$ as the weight on the degree. Coefficients corresponding to numerical features receive a weight of 1. For consistency with this definition, the parameters λ_k used by all methods in this comparison are set proportional to the weights, i.e., with $\lambda_1 / \lambda_0 = 0.2$. By varying the remaining free parameter λ_0, we sweep out trade-offs between performance and complexity. RuleFit has an additional parameter, the mean tree size, that is recommended for tuning in (Friedman & Popescu, 2008). We have done so based on test set results, which gives RuleFit a slight advantage.

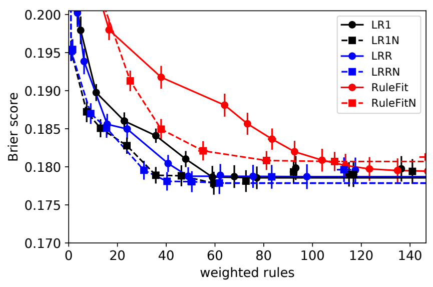
Figure 1 shows the resulting trade-offs with Brier score for 4 of the 16 datasets. The other 12 plots as well as those for accuracy are in the SM. Pareto-efficient points, i.e., those not dominated by points with both lower Brier score and lower complexity, have been connected with line segments for the sole purpose of visualization. RuleFit obtains inferior trade-offs on most of the tested datasets. Even in cases such as Figure 1c where RuleFit eventually attains a lower Brier score, the initial part of the trade-off is worse. Note that the variants employing numerical features generally fare better, as expected. Figure 1c indicates that IP CG (LRRI) can improve upon the heuristic in some cases.

Next we discuss the differences between LR1(N) and

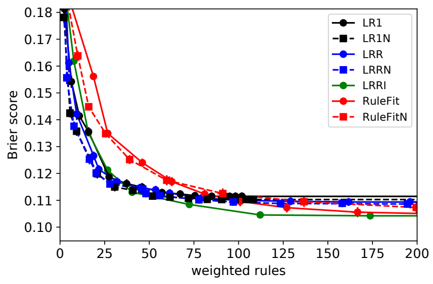
[1]https://github.com/christophM/rulefit



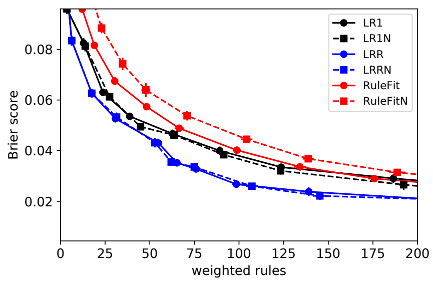
(a) Pima Indians diabetes



(b) FICO Explainable Machine Learning Challenge



(c) MAGIC gamma telescope



(d) Musk molecules

Figure 1. Trade-offs between Brier score and weighted number of rules. Pareto efficient points are connected by line segments. Horizontal and vertical bars represent standard errors in the means.

LRR(N). A general observation is that LRR, which uses CG to produce higher-degree rules, tends to achieve better trade-offs on larger datasets, as exemplified by musk in Figure 1d. This can be explained by the fact that LRR effectively considers a much larger feature space than LR1. If the training sample size is sufficient to support this, then the greater power of the higher-degree rules found by LRR generalizes to test data. The presence of strong interactions in the data also favors LRR. On the other hand, on smaller datasets such as in Figure 1a, LRR may overfit and achieve a worse trade-off relative to LR1. In Figure 1b, LRR outperforms LR1 but the advantage disappears for LRRN compared to LR1N, a pattern that also occurs on other datasets. In Figure 1c, the advantage of LRR(N) lies in attaining a slightly lower minimal Brier score.

In a second experiment, we aim to maximize performance (minimize Brier score or maximize accuracy) by performing nested CV on the training set to select λ_0 and applying the resulting model to the test set. Since performance is now the primary criterion, we broaden the comparison to include GBT and SVM. For GBT, the maximum tree depth was tuned and the number of trees was also determined via a stopping criterion on a validation set, up to a maximum of 500 trees. For SVM, the regularization parameter C and kernel width γ were tuned and Platt scaling (Platt, 1999) was used to calibrate the output scores as probabilities.

Table 1 shows the resulting mean Brier scores while Table 2 shows the weighted number of rules at which the Brier scores were achieved. To help summarize these results, we report the mean rank of each method as well as Friedman tests on these mean ranks, as suggested by a reviewer and following (Demšar, 2006). The overall conclusion is that when tuned for maximum performance, the proposed methods, especially LRRN, compete well with benchmark models and do so using significantly fewer rules than RuleFit.

For Table 1, the Friedman statistic is 12.41, corresponding to a p-value of 0.082 using the F-distribution approximation. Hence the null hypothesis of no significant differences is rejected at the 0.10 level but not at 0.05. A post-hoc test, comparing all other algorithms to LRRN and correcting for multiple comparisons using Holm's procedure, shows that the only significant difference at the 0.10 level is with LR1, i.e., when excluding both higher-degree rules and numerical features. The mean ranks also show that LRR(N) outperforms LR1(N) although most of the differences are not statistically significant.

In Table 2, it is clear that RuleFit produces much more (3-4 times) complex classifiers than all four proposed methods. The Friedman statistic of 34.01 (p-value $\sim 10^{-6}$) is significant as expected. Post-hoc comparisons to LRRN confirm that RuleFit and RuleFitN are significantly more complex, this time at the 0.05 level and again with Holm's correction. Note that including original numerical features does not significant increase classifier complexity (compar-

Table 1. Mean test Brier scores (standard error in parentheses). Best values in **bold**.

dataset	LR1	LRR	RuleFit	LR1N	LRRN	RuleFitN	GBT	SVM
banknote	2.58 (0.75) E-3	2.58 (0.98) E-3	3.46 (0.99) E-3	**7.43** (1.94) **E-5**	1.54 (0.95) E-3	1.53 (0.83) E-3	4.50 (1.04) E-3	3.58 (2.47) E-4
heart	1.33 (0.09) E-1	1.33 (0.10) E-1	**1.22** (0.09) **E-1**	1.34 (0.10) E-1	1.32 (0.09) E-1	1.25 (0.10) E-1	1.33 (0.08) E-1	1.26 (0.12) E-1
ILPD	1.86 (0.06) E-1	1.82 (0.05) E-1	2.50 (0.00) E-1	1.86 (0.06) E-1	1.82 (0.05) E-1	2.50 (0.00) E-1	**1.78** (0.03) **E-1**	1.94 (0.02) E-1
ionosphere	7.26 (0.97) E-2	6.63 (0.95) E-2	6.31 (1.39) E-2	7.07 (1.11) E-2	6.99 (1.09) E-2	4.95 (1.30) E-2	6.98 (1.08) E-2	**3.96** (1.18) **E-2**
liver	2.50 (0.12) E-1	2.47 (0.09) E-1	2.55 (0.11) E-1	2.57 (0.13) E-1	2.50 (0.11) E-1	2.48 (0.16) E-1	2.47 (0.06) E-1	**2.35** (0.08) **E-1**
pima	1.63 (0.09) E-1	1.66 (0.09) E-1	1.62 (0.09) E-1	1.61 (0.08) E-1	1.65 (0.08) E-1	1.66 (0.11) E-1	1.66 (0.07) E-1	**1.57** (0.10) **E-1**
tic-tac-toe	1.62 (0.30) E-2	1.71 (0.41) E-2	**3.78** (2.58) **E-7**	1.62 (0.30) E-2	1.71 (0.41) E-2	**3.78** (2.58) **E-7**	8.22 (1.60) E-3	1.53 (0.35) E-2
transfusion	1.66 (0.05) E-1	1.55 (0.04) E-1	1.62 (0.09) E-1	**1.53** (0.03) **E-1**	1.55 (0.03) E-1	1.66 (0.11) E-1	1.60 (0.03) E-1	1.68 (0.03) E-1
WDBC	2.52 (0.61) E-2	1.76 (0.38) E-2	2.01 (0.35) E-2	2.08 (0.61) E-2	**1.41** (0.34) **E-2**	2.62 (0.44) E-2	3.09 (0.37) E-2	1.63 (0.26) E-2
adult	1.05 (0.01) E-1	1.05 (0.01) E-1	1.07 (0.01) E-1	9.78 (0.09) E-2	9.78 (0.09) E-2	**9.01** (0.09) **E-2**	1.04 (0.01) E-1	1.11 (0.01) E-1
bank-mkt	1.00 (0.00) E-1	7.79 (0.07) E-2	1.73 (0.00) E-1	1.00 (0.00) E-1	**7.77** (0.07) **E-2**	9.99 (0.00) E-2	7.98 (0.07) E-2	8.94 (0.06) E-2
gas	3.93 (0.41) E-3	**3.40** (0.35) **E-3**	4.19 (0.49) E-3	3.99 (0.33) E-3	4.40 (0.34) E-3	3.70 (0.30) E-3	5.72 (0.50) E-3	4.11 (0.46) E-3
magic	1.11 (0.01) E-1	1.09 (0.01) E-1	1.06 (0.02) E-1	1.10 (0.01) E-1	1.09 (0.02) E-1	**9.33** (0.17) **E-2**	9.56 (0.17) E-2	9.34 (0.15) E-2
mushroom	5.05 (1.75) E-7	2.02 (1.10) E-7	**0.00** (0.00) **E-7**	5.05 (1.75) E-7	2.02 (1.10) E-7	**0.00** (0.00) **E-7**	4.69 (2.51) E-4	3.10 (1.73) E-4
musk	2.53 (0.31) E-2	1.37 (0.09) E-2	2.02 (0.11) E-2	3.00 (0.49) E-2	**1.20** (0.10) **E-2**	1.76 (0.12) E-2	4.27 (0.33) E-2	1.30 (0.38) E-2
FICO	1.79 (0.01) E-1	1.79 (0.02) E-1	1.80 (0.02) E-1	1.78 (0.02) E-1	**1.78** (0.01) **E-1**	1.79 (0.01) E-1	1.80 (0.01) E-1	1.88 (0.01) E-1
mean rank	5.81	4.12	4.84	4.75	**3.56**	3.59	5.31	4.00

Table 2. Mean weighted number of rules (standard error in parentheses) corresponding to Table 1. Best values in **bold**.

dataset	LR1	LRR	RuleFit	LR1N	LRRN	RuleFitN
banknote	32.3 (0.8)	54.9 (2.4)	57.8 (0.7)	**16.4** (0.4)	50.6 (3.3)	1124.9 (67.7)
heart	13.4 (2.7)	7.6 (0.7)	34.3 (0.9)	14.3 (2.1)	**6.0** (0.8)	59.4 (2.4)
ILPD	14.6 (3.7)	11.6 (1.4)	**0.0** (0.0)	14.7 (4.9)	11.3 (1.3)	**0.0** (0.0)
ionosphere	**114.4** (28.5)	122.8 (35.4)	1007.3 (12.0)	130.3 (23.7)	122.6 (44.1)	983.4 (145.2)
liver	28.9 (5.8)	16.9 (2.6)	66.7 (11.7)	25.7 (5.7)	**14.0** (2.0)	89.7 (36.8)
pima	22.1 (2.3)	25.9 (1.4)	64.8 (1.1)	**12.1** (1.0)	18.7 (2.9)	183.8 (34.9)
tic-tac-toe	**21.6** (0.0)	78.2 (6.3)	1640.7 (99.2)	**21.6** (0.0)	78.2 (6.3)	1640.7 (99.2)
transfusion	15.2 (4.3)	17.6 (1.1)	64.8 (1.1)	24.3 (2.5)	**12.4** (1.9)	183.8 (34.9)
WDBC	145.7 (18.0)	271.4 (13.6)	809.4 (89.6)	**86.1** (12.6)	248.4 (28.0)	562.3 (83.3)
adult	87.1 (1.6)	**83.2** (5.3)	102.4 (4.6)	85.8 (2.4)	104.7 (4.3)	719.9 (58.6)
bank-mkt	**0.0** (0.0)	68.9 (3.6)	**0.0** (0.0)	**0.0** (0.0)	61.8 (3.6)	0.2 (0.0)
gas	**483.7** (8.0)	694.8 (2.1)	2663.1 (235.9)	950.2 (12.9)	1145.2 (25.6)	2920.8 (125.7)
magic	**93.1** (2.2)	202.1 (13.5)	496.7 (5.9)	97.9 (2.9)	219.9 (21.5)	947.4 (7.0)
mushroom	**24.7** (0.6)	30.1 (1.6)	1308.9 (207.5)	**24.7** (0.6)	30.1 (1.6)	1308.9 (207.5)
musk	**263.0** (39.3)	1255.7 (25.4)	1152.1 (298.9)	313.9 (101.6)	1348.3 (50.0)	2000.3 (314.8)
FICO	92.6 (5.9)	72.9 (5.1)	239.8 (2.5)	81.0 (5.2)	**51.8** (2.7)	183.4 (3.0)
mean rank	**2.31**	3.19	4.66	2.56	2.94	5.34

ing LR1 with LR1N and LRR with LRRN).

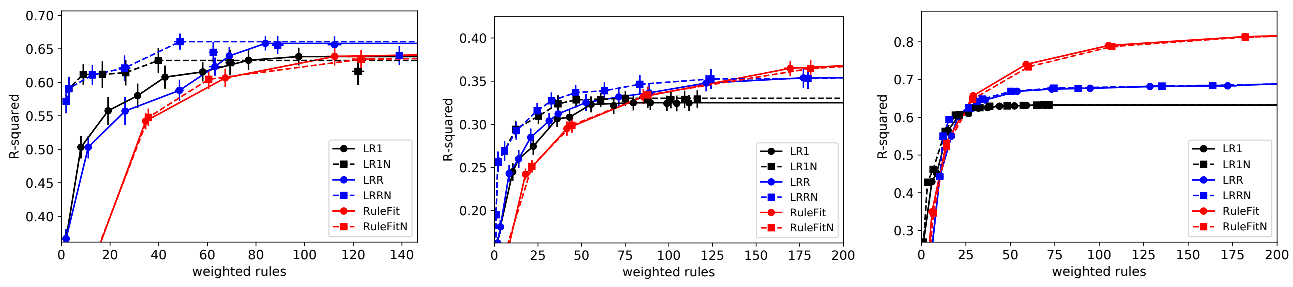
5.2. Regression

For regression, we experimented with an additional 8 datasets, 7 of which are drawn from previous works on rule ensembles (Friedman & Popescu, 2008; Dembczyński et al., 2010) and the UCI repository (Dua & Karra Taniskidou, 2017). The last dataset comes from the Medical Expenditure Panel Survey (MEPS) (Agency for Healthcare Research and Quality, 2018) of the US Department of Health and Human Services, specifically panel 19 from the year 2015. The task is to predict the annual healthcare expenditure of individuals based on demographics and self-reported medical conditions.

The same two experiments are conducted for regression, using the linear regression variants (6) of both the proposed approaches as well as RuleFit. The coefficient of determination R^2 is chosen as the performance metric and the model complexity metric is the same as in Section 5.1.

In Figure 2, we show trade-offs between R^2 and weighted number of rules for 3 of the datasets; the other 5 can be found in the SM. RuleFit is a stronger competitor in regression due to sometimes achieving higher R^2 at higher complexities. On 4 of 8 datasets, the curves for RuleFit(N) remain below their LRR(N) counterparts as in Figure 2a. On another 2 datasets, the curves cross at moderate to high complexities as in Figure 2b, while in Figure 2c, RuleFit obtains much higher R^2. Among the proposed methods, the advantage of LRR(N) vs. LR1(N) is generally larger in regression than in classification (cf. Figure 1), indicating the benefit of generating higher-degree rules. A possible explanation may be that interaction terms matter more in regression where the output range is wider.

Tables 3 and 4 show the results of selecting parameter λ_0 through nested CV to maximize R^2. The same overall conclusion as in Section 5.1 holds for LRRN in particular, namely that it yields highly competitive R^2 values while using fewer rules than RuleFit. The Friedman statistic for Table 3 is 13.63 (p-value 0.046), and in post-hoc comparisons to LRRN, the only significant difference (0.05 level) is again with LR1. Among the proposed methods, advantages due to CG and/or numerical features in Figure 2 carry over into Table 3. In Table 4, RuleFit(N) again yields more



(a) Communities and Crime (b) Wine Quality (c) Bike Sharing

Figure 2. Trade-offs between coefficient of determination R^2 and weighted number of rules.

Table 3. Mean test R^2 (%, standard error in parentheses). Best values in **bold**.

dataset	LR1	LRR	RuleFit	LR1N	LRRN	RuleFitN	GBT	SVM
abalone	51.2 (1.2)	51.2 (1.3)	51.1 (1.1)	55.4 (1.3)	55.5 (1.3)	54.7 (1.3)	51.8 (1.0)	**56.7** (1.0)
boston	77.9 (4.1)	76.4 (4.2)	84.3 (2.3)	78.8 (3.5)	78.2 (3.7)	**84.3** (2.2)	79.3 (3.5)	82.0 (2.7)
bike	63.2 (0.5)	69.1 (0.5)	83.0 (0.4)	63.3 (0.5)	68.9 (0.5)	83.0 (0.4)	**83.9** (0.3)	54.3 (0.5)
california	70.2 (0.4)	73.3 (0.4)	75.2 (0.3)	72.7 (0.3)	75.6 (0.3)	**76.6** (0.3)	75.6 (0.4)	76.4 (0.4)
crime	60.6 (1.7)	63.9 (1.3)	64.1 (1.5)	61.3 (2.2)	**65.7** (1.2)	63.5 (1.6)	63.9 (1.2)	54.5 (1.0)
parkinsons	17.8 (0.6)	45.2 (1.3)	25.3 (3.4)	17.8 (0.6)	46.1 (1.2)	**49.2** (7.9)	29.3 (0.8)	7.0 (0.3)
wine	32.3 (1.0)	35.5 (1.2)	33.6 (0.9)	32.9 (0.9)	35.6 (1.4)	32.6 (1.1)	**38.0** (1.1)	36.5 (0.8)
MEPS	16.4 (1.5)	16.4 (1.4)	15.3 (1.6)	**16.6** (1.5)	16.5 (1.4)	14.5 (1.4)	15.7 (1.5)	5.5 (0.6)
mean rank	6.75	4.88	4.50	5.00	**3.00**	3.50	3.38	5.00

Table 4. Mean weighted number of rules (standard error in parentheses) corresponding to Table 3. Best values in **bold**.

dataset	LR1	LRR	RuleFit	LR1N	LRRN	RuleFitN
abalone	**69.4** (1.0)	141.7 (13.0)	90.1 (3.5)	74.9 (0.9)	144.1 (13.8)	83.4 (2.1)
boston	56.3 (3.0)	96.1 (21.3)	432.3 (41.1)	**50.2** (3.1)	97.1 (24.9)	285.8 (18.2)
bike	**69.7** (1.0)	287.1 (14.7)	558.3 (29.6)	70.7 (0.9)	285.0 (16.0)	568.9 (28.0)
california	82.1 (0.6)	219.8 (2.9)	315.3 (11.3)	**78.7** (0.7)	204.4 (3.8)	447.2 (15.9)
crime	237.4 (2.8)	76.8 (5.8)	161.7 (3.7)	130.9 (16.5)	**55.1** (5.9)	168.5 (5.0)
parkinsons	**2.4** (0.0)	136.5 (5.9)	14.0 (2.0)	**2.4** (0.0)	134.1 (4.5)	389.8 (120.2)
wine	75.7 (2.2)	188.0 (10.9)	112.2 (7.9)	**61.8** (1.7)	189.8 (9.5)	108.2 (11.6)
MEPS	99.4 (2.2)	80.6 (7.4)	115.1 (3.3)	104.9 (2.7)	**78.9** (7.6)	107.2 (3.1)
mean rank	2.31	3.75	4.62	**1.94**	3.50	4.88

complex solutions on average, although the difference is not as large as in Table 2. The Friedman statistic is 16.16 (p-value 0.002); however, no post-hoc comparisons with LRRN (which occupies a middle position) as the reference show statistically significant differences.

6. Discussion

The numerical results in Section 5 may raise the question of the interpretability of rule ensembles with hundreds of rules. First we note that because of the shape of many of the trade-off curves, with steep improvement at low complexity followed by a flatter region, it may be possible to obtain substantially simpler models, with say a few tens of rules, that are not too far from maximum performance. Apart from model simplification, the fact that a rule ensemble is also a linear model facilitates model inspection, for example by focusing on the rules corresponding to the largest coefficients β_k. Friedman & Popescu (2008) discuss a slightly more refined measure of rule importance, which is also used to assess the importance of (original) input features. Based on the discussion in Section 3, we also suggest a division between singleton rules and linear terms on the one hand, and higher-degree rules on the other. Since the former constitute a GAM, their effect can be summarized visually by univariate plots. The higher-degree rules can be ranked and the interactions studied further as described in (Friedman & Popescu, 2008).

The following extensions are suggested for future work: 1) We have used a fixed binarization of the features to facilitate formulation of the column generation subproblem (9) as an IP. However, if CG is done using a heuristic, it may be possible to refine the binarization with each CG iteration. Doing so may improve results, particularly for regression. 2) In the experiments in Section 5, we have fixed the ratio λ_1/λ_0 to match the definition of rule weight. One may also vary this ratio to encourage or discourage longer rules, or tune it to the level of interaction present in a dataset.

Acknowledgements

We thank Karthikeyan Natesan Ramamurthy for help with the MEPS dataset.

References

Agency for Healthcare Research and Quality. Medical Expenditure Panel Survey (MEPS). http://www.ahrq.gov/data/meps.html, 2018. Last accessed 2019-01.

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., and Rudin, C. Learning certifiably optimal rule lists. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 35–44, 2017.

Bonami, P., Gleixner, A. M., Linderoth, J., and Misener, R. Designing and implementing algorithms for mixed-integer nonlinear optimization. In *Report from Dagstuhl Seminar 18081*, 2018.

Borndörfer, R., Löbel, A., Reuther, M., Schlechte, T., and Weider, S. Rapid branching. *Public Transport*, 5:3–23, 01 2013. doi: 10.1007/s12469-013-0066-8.

Clark, P. and Boswell, R. Rule induction with CN2: Some recent improvements. In *Proceedings of the European Working Session on Machine Learning (EWSL)*, pp. 151–163, 1991.

Cohen, W. W. Fast effective rule induction. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 115–123, 1995.

Cohen, W. W. and Singer, Y. A simple, fast, and effective rule learner. In *Proc. Conf. Artif. Intell. (AAAI)*, pp. 335–342, 1999.

Conforti, M., Cornuejols, G., and Zambelli, G. *Integer programming*. Springer, 2014.

Dash, S., Günlük, O., and Wei, D. Boolean decision rules via column generation. In *Proc. Thirty-second Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

Dembczyński, K., Kotłowski, W., and Słowiński, R. ENDER: a statistical framework for boosting decision rules. *Data Mining and Knowledge Discovery*, 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, January 2002.

Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, January 2006.

Dua, D. and Karra Taniskidou, E. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9: 1871–1874, 2008.

FICO. FICO Explainable Machine Learning Challenge. https://community.fico.com/community/xml, 2018. Last accessed 2018-05-16.

Fortz, B., Labbe, M., and Poss, M. A branch-and-cut-and-price framework for convex minlp applied to a stochastic network design problem. In *Proc. European Workshop on MINLP*, pp. 131–139, 2010.

Friedman, J. H. and Popescu, B. E. Predictive learning via rule ensembles. *Annals of Applied Statistics*, 2(3):916–954, Jul 2008.

Fürnkranz, J., Gamberger, D., and Lavrač, N. *Foundations of Rule Learning*. Springer-Verlag, Berlin, 2014.

Garcia, R., Marin, A., and Patriksson, M. Column generation algorithms for nonlinear optimization, i: Convergence analysis. *Optimization*, 52(2):171–200, 2003.

Gilmore, P. C. and Gomory, R. E. A linear programming approach to the cutting-stock problem. *Operations Research*, 9:849–859, 1961.

Hastie, T. J. and Tibshirani, R. J. *Generalized Additive Models*. Chapman and Hall/CRC, 1990. ISBN 9780412343902.

Hernández-Orallo, J., Flach, P., and Ferri, C. A unified view of performance metrics: Translating threshold choice into expected classification loss. *J. Mach. Learn. Res.*, 13:2813–2869, October 2012.

Jawanpuria, P., Nath, J. S., and Ramakrishnan, G. Efficient rule ensemble learning using hierarchical kernels. In *Proc. Int. Conf. Mach. Learn. (ICML)*, 2011.

Lakkaraju, H., Bach, S. H., and Leskovec, J. Interpretable decision sets: A joint framework for description and prediction. In *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining (KDD)*, pp. 1675–1684, 2016.

McCullagh, P. and Nelder, J. *Generalized Linear Models, Second Edition*. Chapman & Hall, 1989.

Platt, J. C. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pp. 61–74, 1999.

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

Rückert, U. and Kramer, S. A statistical approach to rule learning. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 785–792, 2006.

Rudin, C. and Ertekin, Ş. Learning customized and optimized lists of rules with mathematical programming. *Mathematical Programming Computation*, 10(4):659–702, Dec 2018.

Su, G., Wei, D., Varshney, K. R., and Malioutov, D. M. Learning sparse two-level Boolean rules. In *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, pp. 1–6, September 2016.

Wang, T., Rudin, C., Doshi-Velez, F., Liu, Y., Klampfl, E., and MacNeille, P. A Bayesian framework for learning rule sets for interpretable classification. *Journal of Machine Learning Research*, 18(70):1–37, 2017.

Weiss, S. M. and Indurkhya, N. Lightweight rule induction. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 1135–1142, 2000.

Yang, H., Rudin, C., and Seltzer, M. Scalable Bayesian rule lists. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 1013–1022, 2017.