Evolving Accurate and Comprehensible Classification Rules

Cecilia Sönströd
School of Business and Informatics
University of Borås, Sweden
cecilia.sonstrod@hb.se

Ulf Johansson
School of Business and Informatics
University of Borås, Sweden
ulf.johansson@hb.se

Rikard König
School of Business and Informatics
University of Borås, Sweden
rikard.konig@hb.se

***Abstract*—In this paper, Genetic Programming is used to evolve ordered rule sets (also called decision lists) for a number of benchmark classification problems, with evaluation of both predictive performance and comprehensibility. The main purpose is to compare this approach to the standard decision list algorithm JRip and also to evaluate the use of different length penalties and fitness functions for evolving this type of model. The results, using 25 data sets from the UCI repository, show that genetic decision lists with accuracy-based fitness functions outperform JRip regarding accuracy. Indeed, the best setup was significantly better than JRip. JRip, however, held a slight advantage over these models when evaluating AUC. Furthermore, all genetic decision list setups produced models that were more compact than JRip models, and thus more readily comprehensible. The effect of using different fitness functions was very clear; in essence, models performed best on the evaluation criterion that was used in the fitness function, with a worsening of the performance for other criteria. Brier score fitness provided a middle ground, with acceptable performance on both accuracy and AUC. The main conclusion is that genetic programming solves the task of evolving decision lists very well, but that different length penalties and fitness functions have immediate effects on the results. Thus, these parameters can be used to control the trade-off between different aspects of predictive performance and comprehensibility.**

I. Introduction

In data mining and machine learning, classification is one of the most frequently occurring tasks. A classification problem consists of predicting the value of a *target variable*, taking its value from a fixed set (the *class labels*), based on a number of input variables (or *attributes*). A wealth of techniques and algorithms exist for producing classification models with high predictive performance on unseen data. The models produced can be divided into two main groups, depending on whether they make explicit the relationships they find in the data and use to perform the classification. Models that do not represent these relationships explicitly, and thus make it impossible, or at least, overwhelmingly complex, to understand the reasons behind a classification, are called *opaque* models. The most commonly used techniques for producing opaque classification models are artificial neural networks (ANNs), support vector machines (SVMs) and ensemble schemes.

In contrast, models that enable inspection are called *transparent*, and the two most common representations here are decision trees and ordered rule sets, also called decision lists. A decision tree consists of a number of internal nodes containing tests (typically containing an attribute, a relational operator and a value) that define splits in the tree based on the outcome of the test. Leaf nodes in the tree are used to assign class labels. An instance is classified by starting at the root node and then following the path determined by the conditions in the splits, until reaching a leaf where it receives its classification. A decision list is an ordered rule set, where classification is performed by matching the instance against the rule conditions until a match is found, and that rule will then assign the class label. The last rule is a default rule, assigning a class label without conditions to all instances reaching this point in the decision list. Although conceptually different from a tree, a decision list can be represented as a special case of a decision tree, in which every split leads to at least one leaf node, thus resulting in a maximally unbalanced tree. An example of a decision list and its corresponding tree structure is shown in Figure 1.

```
IF petalwidth <= 0.6 THEN class=Iris-setosa
IF petallength > 5.1 THEN class=Iris-virginica
IF petalwidth > 1.7 THEN class=Iris-virginica
DEFAULT: Iris-versicolor
```

```
if petalwidth <= 0,6
|T: Iris-setosa
|F: if petallength > 5,1
| |T: Iris-virginica
| |F: if petalwidth > 1,7
| | |T: Iris-virginica
| | |F: Iris-versicolor
```

Fig. 1. Decision List in Rule and Tree Representation

It is an accepted fact in the machine learning community that building a transparent model will generally result in worse predictive performance than using a technique producing an opaque model. However, comprehensibility is sometimes a necessity, either because models must explain their predictions or because users want to gain insight from the relationships found by the model. This loss of predictive performance for transparent models has been termed the *accuracy vs. comprehensibility trade-off*. It should noted that not all transparent models are inherently comprehensible; the sheer complexity of a decision tree with a few hundred nodes will make it impossible to understand in its entirety even if the path of a single prediction can be traced. It will also be very difficult to understand the main relationships found by the model in such a tree.

Building a model from training data with good generalization capability is essentially a search problem, regardless of whether an opaque or a transparent model is sought. In this paper, we will focus solely on transparent models that are small or simple enough to be comprehensible, but even with this limitation exhaustive search is impossible. Several state-of-the-art algorithms for producing transparent models exist, most commonly decision trees, but also decision lists. Most of these algorithms use the same basic search strategy, i.e. some sort of local (greedy) search. Decision tree algorithms typically use recursive partitioning to produce leaf nodes with pure class distributions. Decision list algorithms are usually based on the separate-and-conquer algorithm, which in each step will formulate the best possible rule (according to some rule selection criterion) and then remove the instances covered by this rule before formulating the next rule. The more sophisticated algorithms producing transparent models post-process the models by applying pruning and/or optimization procedures using an internal validation set.

In [1], some observations regarding the suitability of Genetic Programming (GP) for classification tasks are made. Most importantly, GP is flexible enough to be applicable to classification problems in different ways, such as building models with different representations or performing pre- or post-processing tasks. The authors also point out the obvious fact that the fitness function can be designed to encourage smaller models, which will increase both generalization capability and comprehensibility. A large number of approaches for evolving transparent classifiers are surveyed. Many of these incorporate a length penalty term in the fitness function to force the search into looking for small models and some also evaluate the use of different fitness functions. However, most of the approaches that use GP to construct classification rules use unordered rule sets, or evolve separate rules for each class. Each such rule is, of course, comprehensible, given that a suitable rule representation is chosen, but the whole model will contain rules with overlapping, or worse, conflicting conditions, making comprehension of the entire model cumbersome. In contrast, decision lists are often cited as being easy to interpret, see e.g. [2]. However, few approaches to evolving classifiers with the decision list representation exist, and those that do contain only limited evaluation, e.g. [3], [4] and [5]. In [6], we proposed a technique, named *GEDEL* (*GEnetic DEcision Lists*), for evolving decision lists. However, the investigation only consisted of using one setup, with a fitness function based on Brier score [7] and evaluation was limited to 17 binary classification problems.

The main purpose of this paper is to apply GP to the problem of producing decision lists (ordered rule sets) and to evaluate the resulting models based on predictive performance and comprehensibility over a larger number of benchmark data sets, by comparing them to models produced by the state-of-the-art decision list algorithm RIPPER [8]. Furthermore, it is investigated how using tailor-made fitness functions optimizing different aspects of predictive performance and varying the length penalty affect the resulting models' performance.

II. Background

G-REX (*GEnetic Rule Extraction*), first presented in [9], is a GP framework originally intended for rule extraction, i.e., the process of extracting comprehensible representations from opaque models. For a thorough presentation and evaluation of G-REX on a large number of rule extraction tasks, see [10]. Recently, G-REX has evolved into a general GP data mining framework, see [11] and has also been integrated into the WEKA [12] data mining workbench. G-REX is thus able to evolve classifiers directly from data, using a variety of representation languages, including decision trees and decision lists, and use all the evaluation mechanisms provided by WEKA.

GEDEL is based on G-REX, where the basic algorithm is standard GP. The built-in ability to vary the representation language makes it possible to produce decision lists, simply by supplying an appropriate grammar. The sets *F* and *T* describe the available functions and terminals, respectively. Here, the functions used are an *if*-statement and three relational operators. The terminals are attributes from the data set and random real numbers. The exact grammar used is presented, using Backus-Naur form, in Figure 2 below. It should be noted that no conjunctions are used in the rules.

$$\mathtt{F} = \{\ \mathtt{if},\ \mathtt{==},\ \leq,\ >\ \}$$
$$\mathtt{T} = \{\ i_1,\ i_2,\ \ldots,\ i_n,\ c_1,\ c_2,\ \ldots,\ c_m,\ \Re\ \}$$

DTree	:-	(if RExp Class DList) | Class
RExp	:-	(ROp ConI ConC) | (== CatI CatC)
ROp	:-	$\leq$ | $>$
CatI	:-	Categorical input variable
ConI	:-	Continuous input variable
CatC	:-	Categorical attribute value
ConC	:-	$\Re$
Class	:-	c_1 | c_2 | ... | c_m

Fig. 2. Representation Language for Genetic Decision Lists

The most successful decision list algorithm is RIPPER (*Repeated Incremental Pruning to Produce Error Reduction*), proposed by Cohen in [8]. Like most decision list algorithms, RIPPER is based on the sequential covering algorithm, which in each step constructs a rule covering a number of instances and then removes those instances from the dataset before the next rule is found. This procedure is repeated until some stop criterion is met, when a default rule, classifying all remaining instances, is formulated. RIPPER is reported to have good performance regarding both accuracy [12], running time and ability to handle noise and unbalanced data sets [13]. In WEKA, RIPPER is implemented as JRip, which is the version of the algorithm that is described here. The main idea in RIPPER is to construct rules only for the minority class(es) and using the default rule for the majority class. For multi-class problems, rules will be constructed for all classes, except the majority class, in order of class distribution. In short, RIPPER constructs its rule sets in three phases, called *growing*, *pruning* and *optimization*. In the growing phase, conditions are added to a rule as long as no negative examples are covered. In the pruning phase, conditions are removed based on performance on a validation set. Pruning may thus result in the rule covering negative examples. In [14], most of RIPPER's power is attributed to its optimization (post-pruning) procedure and the authors argue that post-pruning is probably the best way to obtain good predictive performance for decision list algorithms. It should be noted that even though JRip usually performs very well, especially on the benchmark datasets from the UCI machine learning repository [15], the algorithm is designed to optimize accuracy and will sometimes do so at the expense of comprehensibility, without any clear possibility for the user to control this trade-off via parameter settings.

III. Method

A. GP Settings and Fitness Functions

Using the grammar described in Figure 2, decision lists were evolved using the GP parameter settings shown in Table I. These are quite similar to the ones used in [6], where GEDEL was introduced, but with some tweaking of mutation rate and creation depth, and also using larger populations and more generations.

TABLE I
GP Parameters

Parameter	Value
Crossover rate	0.8
Mutation rate	0.01
Population size	1000
Generations	100
Creation depth	7
Creation method	Ramped-half-and-half
Selection	Roulette wheel
Elitism	Yes

The fitness functions used are based on three different measures of predictive performance; *accuracy*, *area-under-ROC* (*AUC*) and *Brier score*. Each fitness function also incorporates a length penalty term which will decrease the fitness of individual proportionally to its length, calculated as the number of elements (functions and terminals). For a detailed discussion about the different fitness functions, see [16] and [17].

1) Accuracy Fitness: Accuracy is the arguably the simplest metric available to measure predictive performance and is defined as the proportion of correctly classified instances:

$$Accuracy = \frac{\#correct}{\#instances} \tag{1}$$

The fitness value for an individual i based on accuracy is calculated using:

$$ACC\_Fitness_i = 100 \times Accuracy_i - length_i \times p \tag{2}$$

where p is the penalty term applied.

2) AUC Fitness: Using accuracy for evaluation of predictive performance has several drawbacks, as argued in [18]. In recent years, it has become standard procedure to also use a metric based on the Receiver Operator Curve (ROC), called AUC, for area-under-ROC. While accuracy is based only on the final classification, AUC measures the ability of the model to rank instances according to how likely they are to belong to a certain class; see e.g. [19]. AUC can be interpreted as the probability of ranking a true positive instance ahead of a false positive; for details, see [20]. To calculate AUC Fitness, G-REX employs Laplace estimates for ranking instances during the ROC analysis. The main reason for using a Laplace estimate is that the basic (maximum-likelihood) estimate does not consider the number of training instances supporting a specific decision, just the proportions. Intuitively, a decision based on many training instances is a better estimator of class membership probabilities. With this in mind, the general Laplace estimator calculates the estimated probability as:

$$P_{classA} = \frac{n+1}{N+C} \tag{3}$$

where n is the number of training instances belonging to class A, N is the total number of training instances involved in the decision, and C is the number of classes. Specifically, the Laplace estimate does not assign a zero probability to a class even if it is not supported at all. When applying the Laplace estimate to a decision tree or a decision list, the calculation is performed in the specific leaf reached by the test instance; i.e. n is the number of training instances supporting class A and N is the total number of training instances reaching that leaf.

The AUC fitness of an individual i is calculated in a similar manner to accuracy, but using the AUC value instead, thus the fitness value for an individual i based on AUC is calculated using:

$$AUC\_Fitness_i = 100 \times AUC_i - length_i \times p \tag{4}$$

3) Brier Score: A Brier score measures the accuracy of a set of probability assessments. Proposed by Brier in 1950 [21], Brier score is the average deviation between predicted probabilities for a set of events and their outcomes; i.e. a lower score represents higher accuracy. Using Brier score for classification thus also requires a probability estimation for each class, in this case provided by the LaPlace estimate. For binary problems, the Brier score is calculated using:

$$BS = \sum_{i=1}^{N}(y_i - p_i)^2 \tag{5}$$

where y_i is the class value of instance i, i.e. $y_i = 0$ or 1 and p_i denotes the probability estimate that instance i belongs to class 1. The Brier score is thus the sum of squares of differences between the true class and the predicted probability over all instances. For multi-class problem, a generalized variant is used:

$$GBS = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{C} (y_{ij} - p_{ij})^2 \quad (6)$$

where p_{ij} denotes the probability estimate that instance i belongs to class j and y_{ij} is 1 if y_i belongs to class j and 0 otherwise.

The Brier score for an individual i over N instances thus becomes:

$$Brier\_Score_i = \frac{\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{C} (y_{ij} - p_{ij})^2}{N} \quad (7)$$

and the corresponding fitness function is:

$$Brier\_Fitness_i = 100 \times (1 - Brier\_Score_i) - length_i \times p \quad (8)$$

B. Data sets

For the experiments, 25 publicly available data sets from the UCI repository [15] were used. Table II below summarizes the characteristics for these data sets, showing number of instances (*Inst.*), number of classes (*Classes*), number of continuous (*Cont.*) and categorical (*Cat.*) attributes. As is seen in the table, both binary and multi-class problems are present.

TABLE II
DATA SET CHARACTERISTICS

Data set	Inst.	Classes	Cont.	Cat.
Breast cancer	286	2	0	9
CMC	1473	3	2	7
Colic	368	2	7	15
Credit-A	690	2	6	9
Credit G	1000	2	7	13
Cylinder	540	2	18	21
Diabetes	768	2	8	0
E-coli	336	8	7	0
Glass	214	6	9	0
Haberman	306	2	3	0
Heart-C	303	2	6	7
Heart-S	270	2	6	7
Hepatitis	155	2	6	13
Ionosphere	351	2	34	0
Iris	150	3	4	0
Labor	57	2	8	8
Liver disorders	345	2	6	0
Lymphography	148	4	3	15
Sonar	208	2	60	0
TAE	151	3	1	4
Vehicle	846	4	18	0
Vote	435	2	0	16
Wine	178	3	13	0
WBC	699	2	9	0
Zoo	100	7	0	16

C. Experiments

The experiments were conducted using the data mining workbench WEKA [12], using WEKA's implementation of RIPPER, called JRip, and a version of G-REX integrated into the WEKA system. JRip was run with its default parameter settings. All experiments used 10-fold cross-validation with identical folding for all techniques. Reported accuracy and AUC values are averaged over all ten folds, whereas size measures are given for the model that WEKA outputs, constructed using the entire data set.

In Experiment 1, the effect of using different values for the length penalty part of the fitness function was investigated. Specifically, the three penalty values of 0.01, 0.005 and 0.001 were used. These choices were based on earlier experiences of using G-REX to evolve decision trees. To ensure comparable results, the same fitness function was used, resulting in the following three setups:

- **G1-Acc**: Fitness function *accuracy* and length penalty 0.01, geared towards small models.
- **G2-Acc**: Fitness function *accuracy* and length penalty 0.005, geared towards slightly larger models.
- **G3-Acc**: Fitness function *accuracy* and length penalty 0.001, geared towards larger models.

In Experiment 2, the fitness function was varied, while the length penalty was kept fixed at the value producing the best results in Experiment 1. The three fitness functions used, and the resulting setups, were:

- **G2-Acc**: Fitness function *accuracy* and length penalty 0.005.
- **G2-AUC**: Fitness function *area-under-ROC* and length penalty 0.005.
- **G2-Brier**: Fitness function *Brier score* and length penalty 0.005.

IV. Results

The accuracy results from Experiment 1, where the use of three different length penalty values is investigated, are shown in Table III below.

The main observation regarding accuracy is that all GEDEL setups achieve slightly higher average accuracy than JRip. For most of the data sets, accuracy levels are quite similar. However, there are some notable exceptions, such as Heart-C where JRip clearly outperforms all GEDEL setups, and TAE, where all GEDEL variants performs much better than JRip. The differences between the techniques become much more obvious when considering the mean ranks. Here, G2-Acc, i.e. using the middle value for length penalty, emerges as clear winner, while the two other GEDEL setups obtain quite similar average ranks and JRip achieving the worst average rank.

To determine if these differences in ranks are statistically significant, the procedure recommended by Demšar [22] for comparing several classifiers over a number of data sets is used. This procedure consists of a Friedman test [23], followed by the Nemenyi post-hoc test [24]. With four classifiers and 25 data sets, the critical distance (for $\alpha = 0.05$) is 0.94, which means that the only significant difference is that G2-Acc performs significantly better than JRip on accuracy on these data sets. For a direct comparison between G2-Acc and JRip, the sign test can be used, where 18 wins out of the 24

TABLE III
ACCURACY RESULTS - EXPERIMENT 1

Data set	**JRip**	**G1-Acc**	**G2-Acc**	**G3-Acc**
Breast cancer	71.0	75.5	74.8	74.5
CMC	52.4	52.9	54.5	54.8
Colic	84.2	86.4	85.3	84.8
Credit-A	85.8	85.2	85.2	84.9
Credit-G	71.7	69.2	69.6	71.0
Cylinder	65.2	65.6	67.4	65.0
Diabetes	76.0	75.3	75.3	75.8
E-coli	81.3	78.0	81.6	77.7
Glass	68.7	59.8	64.0	62.2
Haberman	72.9	73.5	74.5	75.8
Heart-C	81.5	73.9	73.6	76.2
Heart-S	78.9	79.3	81.1	79.3
Hepatitis	78.1	76.8	80.0	81.9
Ionosphere	89.7	86.9	79.5	83.2
Iris	94.0	92.7	95.3	95.3
Labor	77.2	84.2	86.0	80.7
Liver	64.6	68.1	68.7	66.4
Lymph	77.7	79.7	81.8	79.1
Sonar	73.1	71.2	73.6	74.0
TAE	42.4	58.3	57.6	57.0
Vehicle	68.6	62.3	62.4	61.4
Vote	95.4	96.1	95.9	95.9
Wine	91.6	93.8	94.4	92.7
WBC	95.4	95.7	95.4	94.3
Zoo	86.1	88.1	88.1	88.1
Mean	**76.9**	**77.1**	**77.8**	**77.3**
Mean rank	**2.84**	**2.40**	**1.88**	**2.52**

TABLE IV
AUC RESULTS - EXPERIMENT 1

Data set	**JRip**	**G1-Acc**	**G2-Acc**	**G3-Acc**
Breast cancer	0.598	0.621	0.577	0.584
CMC	0.632	0.644	0.667	0.669
Colic	0.823	0.838	0.822	0.821
Credit-A	0.874	0.847	0.880	0.887
Credit-G	0.593	0.613	0.640	0.687
Cylinder	0.666	0.613	0.648	0.645
Diabetes	0.739	0.731	0.733	0.747
E-coli	0.920	0.901	0.911	0.893
Glass	0.803	0.756	0.786	0.770
Haberman	0.613	0.561	0.611	0.592
Heart-C	0.831	0.774	0.754	0.782
Heart-S	0.781	0.794	0.783	0.793
Hepatitis	0.664	0.743	0.789	0.756
Ionosphere	0.900	0.843	0.743	0.790
Iris	0.959	0.941	0.975	0.975
Labor	0.779	0.830	0.822	0.813
Liver	0.653	0.636	0.632	0.645
Lymph	0.795	0.792	0.781	0.769
Sonar	0.759	0.721	0.737	0.734
TAE	0.598	0.662	0.659	0.681
Vehicle	0.855	0.829	0.820	0.827
Vote	0.942	0.938	0.940	0.946
Wine	0.955	0.964	0.965	0.974
WBC	0.973	0.962	0.956	0.954
Zoo	0.925	0.983	0.985	0.972
Mean	**0.785**	**0.781**	**0.785**	**0.788**
Mean rank	**2.28**	**2.76**	**2.56**	**2.36**

non-tied data sets are needed for statistical significance. The count shows that G2-Acc is very close to this, being just one win short.

Table IV shows the corresponding AUC results from Experiment 1. As can be seen in the table, the differences in performance are smaller for AUC than for accuracy. JRip performs on par with the best GEDEL setup for average AUC, and actually gets a better average rank. Although differences are far too small to be statistically significant, it is interesting to note the ordering between the three GEDEL setups, both on average AUC and on ranks, with G3-Acc (with the lowest length penalty) performing best, followed by G2-Acc (middle length penalty) and G3-Acc (with the highest length penalty) performing worst. This seems to indicate that the smaller models enforced by a high length penalty will result in models with worse ranking ability, even if they maintain high accuracy.

In Table V, rule set sizes from Experiment 1 are shown. To enable a fair comparison, the total number of tests in the rule set is used as the size measure. A JRip rule will thus have the total number of conjuncts as its rule set size, while a GEDEL rule will have, as its size, the number of rules since no conjunctions are used in GEDEL rules.

Since model size is evaluated mainly to study to what extent it is possible to obtain accurate models that are comprehensible, it is very encouraging to note that most models produced are small enough to enable human understanding.

The main observation when looking at average model size and ranks for the different techniques is that the pattern is very clear. All GEDEL setups obtain considerable smaller models than JRip and there is a clear ordering between the GEDEL models, corresponding directly to the length penalty used. JRip's poor average is, of course, affected adversely by the very large rule sets for the Vehicle data set, and to a lesser degree by the relatively large models for CMC, E-coli and Glass. It is notable that no GEDEL setup ever produces a model with 10 or more conditions in it. Using the statistical tests described above, where the critical distance in ranks is again 0.94 (for $\alpha = 0.05$), there are a few statistically significant differences. G1-Acc and G2-Acc are both significantly better than JRip, and G1-Acc is also significantly better than G3-Acc.

However, when putting these results together with the accuracy and AUC results, it is seen that the smaller models do come at the price of slightly worse predictive performance. This can be seen at individual data set level, with the most striking example being JRip and the TAE data set. The JRip model consisting of only one condition is outperformed by more than 10 percentage points in accuracy by all GEDEL models. The pattern does not hold in general, however, even though G1-Acc obtained the worst averages for both accuracy and AUC, G2-Acc clearly outperformed G3-Acc on accuracy.

Turning to Experiment 2, where three GEDEL setups with different fitness functions, using the same length penalty, are evaluated, the accuracy results are shown in Table VI below. To enable comparison, the G2-Acc results from Experiment 1 are repeated in all Experiment 2 tables.

Here, the average accuracies between the three GEDEL

TABLE V
RULE SET SIZES - EXPERIMENT 1

Data set	JRip	G1-Acc	G2-Acc	G3-Acc
Breast cancer	4	2	2	5
CMC	14	2	2	5
Colic	6	2	5	5
Credit-A	7	1	5	4
Credit-G	5	2	3	3
Cylinder	6	2	2	7
Diabetes	9	2	3	6
E-coli	19	4	4	4
Glass	18	5	4	4
Haberman	3	2	3	6
Heart-C	6	3	4	8
Heart-S	8	3	3	6
Hepatitis	5	3	9	5
Ionosphere	2	2	6	5
Iris	3	3	4	5
Labor	4	4	5	4
Liver	4	2	2	3
Lymph	8	3	4	6
Sonar	9	2	3	4
TAE	1	3	3	4
Vehicle	43	3	3	4
Vote	6	4	5	5
Wine	4	3	3	4
WBC	9	4	4	8
Zoo	6	7	6	7
Mean	**8.36**	**2.92**	**3.88**	**5.08**
Mean rank	**3.08**	**1.20**	**1.88**	**2.80**

TABLE VI
ACCURACY RESULTS - EXPERIMENT 2

Data set	G2-Acc	G2-AUC	G2-Brier
Breast cancer	74.8	71.7	70.3
CMC	54.5	51.1	51.1
Colic	85.3	84.2	84.8
Credit-A	85.2	85.9	84.6
Credit-G	69.6	70.7	70.8
Cylinder	67.4	67.6	66.9
Diabetes	75.3	73.6	73.6
E-coli	81.6	77.7	79.5
Glass	64.0	62.6	64.5
Haberman	74.5	73.5	71.2
Heart-C	73.6	77.2	77.2
Heart-S	81.1	77.0	80.7
Hepatitis	80.0	74.1	78.1
Ionosphere	79.5	81.5	70.7
Iris	95.3	94.7	96.0
Labor	86.0	80.7	89.5
Liver	68.7	58.6	65.8
Lymph	81.8	76.4	78.4
Sonar	73.6	70.2	75.0
TAE	57.6	49.7	53.0
Vehicle	62.4	58.5	61.8
Vote	95.9	95.4	95.2
Wine	94.4	92.7	90.5
WBC	95.4	93.6	95.9
Zoo	88.1	89.1	89.1
Mean	**77.8**	**75.5**	**76.6**
Mean rank	**1.56**	**2.32**	**2.00**

setups differ much more than in Experiment 1, with both the AUC and Brier Score fitness function setups achieving markedly lower averages than the variants evaluated in Experiment 1. These differences are further accentuated in the ranks. With 25 data sets and three classifiers, the critical distance becomes 0.66, which means that the only significant difference is that G2-Acc is better than G2-AUC. Hence, the GEDEL setup that uses accuracy as its fitness function manages to successfully optimize this aspect of predictive performance, whereas the AUC and Brier score fitness functions turned out to be less profitable for achieving high accuracy.

Table VII shows the corresponding AUC results from Experiment 2. These AUC results are strikingly different from the accuracy results presented above. Here, G2-AUC clearly performs best, obtaining both the highest average AUC and also the best average rank. The differences between the techniques are not significant; although it is quite close to being so with the difference in rank between G2-AUC and G2-Acc being 0.60, with 0.66 needed for significance at $\alpha = 0.05$. From these results it is seen that the fitness function again manages to optimize the chosen property very well. Since JRip performed best regarding AUC in Experiment 1, it is interesting to compare G2-AUC and JRip head-to-head using the standard sign test. The difference is very close to being statistically significant, with G2-AUC obtaining 17 wins out of 25 data sets, with 18 wins needed for significance at $\alpha = 0.05$.

Finally, Table VIII shows the model size results from Experiment 2.

Again, the overall impression is that all models are quite small, and the averages support this. With model size never exceeding 9 conditions and the average model consisting of between 3 and 4 conditions, all models produced by these GEDEL setups must again be deemed to be comprehensible. Regarding average size, G2-AUC has, on average, one test more than the two other setups, which does not really affect comprehensibility all that much. However, the ranks does show that these differences in model size are statistically significant. Using the Friedman and Nemenyi post-hoc tests in the same manner as before, the critical distance (for $\alpha = 0.05$) is again 0.60 and thus both G2-Acc and G2-Brier are significantly better than G2-AUC regarding model size.

To summarize Experiment 2, it is clearly seen that the choice of fitness function plays a major part in what properties the models produced will have. In this experiment, using accuracy as the fitness function produced superior performance for this measure, and using AUC as fitness function had the same effect on model AUC. Also, there seems to be a trade-off between performing well on accuracy and on AUC; since G2-Acc performed worst on AUC and G2-AUC was significantly worse than G2-Acc on accuracy. The Brier score fitness function seems to represent a middle ground, obtaining good, but not excellent, performance for both accuracy and AUC. Regarding size, all models produced were small enough to be comprehensible, but optimizing AUC gave rise to models that were slightly larger, with this difference being statistically significant.

TABLE VII
AUC RESULTS - EXPERIMENT 2

Data set	**G2-Acc**	**G2-AUC**	**G2-Brier**
Breast cancer	0.577	0.620	0.582
CMC	0.667	0.672	0.672
Colic	0.822	0.841	0.827
Credit-A	0.880	0.894	0.876
Credit-G	0.640	0.697	0.696
Cylinder	0.648	0.727	0.670
Diabetes	0.733	0.762	0.715
E-coli	0.911	0.911	0.900
Glass	0.786	0.803	0.799
Haberman	0.611	0.650	0.595
Heart-C	0.754	0.803	0.796
Heart-S	0.783	0.790	0.809
Hepatitis	0.789	0.706	0.743
Ionosphere	0.743	0.870	0.756
Iris	0.975	0.971	0.979
Labor	0.822	0.789	0.884
Liver	0.632	0.570	0.641
Lymph	0.781	0.816	0.781
Sonar	0.737	0.756	0.774
TAE	0.659	0.639	0.675
Vehicle	0.820	0.846	0.844
Vote	0.940	0.963	0.925
Wine	0.965	0.963	0.935
WBC	0.956	0.950	0.962
Zoo	0.985	0.975	0.983
Mean	**0.785**	**0.799**	**0.793**
Mean rank	**2.28**	**1.68**	**1.92**

TABLE VIII
RULE SET SIZES - EXPERIMENT 2

Data set	**G2-Acc**	**G2-AUC**	**G2-Brier**
Breast cancer	2	2	2
CMC	2	3	3
Colic	5	5	5
Credit-A	5	6	4
Credit-G	3	4	1
Cylinder	2	3	3
Diabetes	3	5	1
E-coli	4	6	5
Glass	4	5	5
Haberman	3	5	2
Heart-C	4	4	3
Heart-S	3	5	3
Hepatitis	9	7	4
Ionosphere	6	3	6
Iris	4	6	5
Labor	5	5	6
Liver	2	3	3
Lymph	4	4	6
Sonar	3	6	2
TAE	3	7	3
Vehicle	3	5	4
Vote	5	5	4
Wine	3	6	5
WBC	4	7	5
Zoo	6	6	5
Mean	**3.88**	**4.92**	**3.80**
Mean rank	**1.44**	**2.28**	**1.56**

V. CONCLUSION

In this paper, the use of genetic programming for evolving ordered rule sets (decision lists) has been evaluated. Experiments have studied the effects of using both different length penalties and fitness functions optimizing different aspects of predictive performance. All in all, five different parameter settings have been used. The decision lists evolved have been compared on predictive performance, measured as both accuracy and AUC, and on model size to the state-of-the-art decision list algorithm RIPPER, implemented in WEKA as JRip.

Regarding length penalty, the experiments clearly show that there is a direct correspondence between the length penalty applied during evolution and resulting model size. However, all decision list models evolved are small enough to be comprehensible, with the average number of test varying between 3 and 5 conditions for the different setups. When looking at the predictive performance measured as accuracy, all genetic decision lists achieved higher average accuracy and better average ranks than JRip. Furthermore, the best setup, using the middle value for length penalty, was significantly better then JRip, when comparing ranks using the standard statistical tests. For AUC, the results for all techniques were similar, but with a slight advantage for JRip. Among the genetic decision list setups, there were indications that a more severe length penalty produced rule sets with worse AUC values. All in all, the result of using different length penalties was that there appears to be a trade-off in that smaller models evolved using a high length penalty will have slightly worse predictive performance.

The main conclusion regarding the effect of different fitness functions is that the genetic programming successfully optimized the property that is encoded in the fitness function. Thus, decision lists evolved using a fitness function based on accuracy performed clearly better regarding accuracy than those based on Brier score and area-under-ROC, with the difference between the accuracy fitness and the area-under-ROC fitness being statistically significant. In a similar way, the area-under-ROC fitness function yielded models that achieved significantly better AUC than the accuracy fitness function. The Brier score fitness function produced models with adequate performance on both measures. Finally, the relationship between fitness function and model size was that using an area-under-ROC fitness function gave rise to slightly larger models, with the difference being statistically significant compared to using accuracy or Brier score fitness.

The main conclusion that can be drawn from these experiments is that genetic programming is very capable of evolving decision list models that outperform the state-of-the-art algorithm, both regarding predictive performance measured as either accuracy or AUC and also regarding model size. However, despite the robustly good performance, no genetic decision list achieved superior results on all criteria, which means that one must employ the use of different fitness function and length penalties in order to optimize the desired property in evolved models.

VI. Discussion and Future Work

The two experiments in this paper each focus on either length penalty or fitness function variation. A natural extension is to study how different length penalties work together with different fitness functions, possibly using more evaluation criteria on predictive performance.

Even though GEDEL's predictive performance clearly is satisfactory overall, there are a few data sets where all GEDEL models are soundly outperformed by JRip. For these data sets, JRip produces very large models. Given the low creation depth and relatively severe length penalties applied, the GP will never produce models even approaching this complexity. Thus, it would be interesting to design and evaluate a GEDEL setup intended at larger models on this type of data set.

Using model size to measure comprehensibility is, of course, a simplification. We have previously, see e.g. [25], proposed a number of comprehensibility criteria on decision list models. One of these, called *brevity*, takes into account the fact the rules are ordered. This means that a decision list classifying many instances with its top rules must be considered more easy to interpret than another decision list of the same length that uses more test for the majority of its classifications. The corresponding numerical measure is called classification complexity and measures the average number of test needed to classify an instance in the data set. It would, of course, be very interesting to formulate a fitness function based on brevity and compare this to the setups used in this paper on predictive performance, model size and classification complexity.

References

[1] P. G. Espejo, S. Ventura, and F. Herrera, "A survey on the application of genetic programming to classification," *Trans. Sys. Man Cyber Part C*, vol. 40, pp. 121–144, March 2010.

[2] J. Han and M. Kamber, *Data mining: concepts and techniques*, 2nd ed. Morgan Kaufmann, 2006.

[3] K. Tan, A. Tay, T. Lee, and C. Heng, "Mining multiple comprehensible classification rules using genetic programming," *Computational Intelligence, Proceedings of the World on Congress on*, vol. 2, pp. 1302–1307, 2002.

[4] E. Carreno, G. Leguizamón, and N. Wagner, "Evolution of classification rules for comprehensible knowledge discovery," in *IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 1261–1268.

[5] P. Gonzlez, C. Romero, S. Ventura, and C. Hervs, "Induction of classification rules with grammar-based genetic programming," in *Second International Conference on Machine Intelligence (ACIDCA-ICMI), Tozeur (Tunis)*, 2005.

[6] C. Sönströd, U. Johansson, R. König, and L. Niklasson, "Genetic decision lists for concept description," in *DMIN*, R. Stahlbock, S. F. Crone, and S. Lessmann, Eds. CSREA Press, 2008, pp. 450–456.

[7] G. Brier, "Verification of forecasts expressed in terms of probability," *Monthly Weather Review*, vol. 78, no. 1, pp. 1–3, 1950.

[8] W. W. Cohen, "Fast effective rule induction," in *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 115–123.

[9] U. Johansson, R. König, and L. Niklasson, "Rule extraction from trained neural networks using genetic programming," in *ICANN, supplementary proceedings*, 2003, pp. 13–16.

[10] U. Johansson, *Obtaining Accurate and Comprehensible Data Mining Models: An Evolutionary Approach. PhD-thesis*. Institute of Technology, Linköping University, 2007.

[11] R. König, U. Johansson, and L. Niklasson, "G-REX: A versatile framework for evolutionary data mining." in *ICDM Workshops*. IEEE Computer Society, 2008, pp. 971–974.

[12] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, June 2005.

[13] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 1st ed. Addison Wesley, May 2005.

[14] J. Fürnkranz and P. Flach, "An analysis of stopping and filtering criteria for rule learning," in *In Boulicaut, J.-F.* SpringerVerlag, 2004, pp. 123–133.

[15] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[16] R. König, U. Johansson, and L. Niklasson, "Using genetic programming to increase rule quality," in *FLAIRS Conference*, D. Wilson and H. C. Lane, Eds. AAAI Press, 2008, pp. 288–293.

[17] U. Johansson, R. König, and L. Niklasson, "Genetic rule extraction optimizing brier score," in *GECCO'10*, 2010, pp. 1007–1014.

[18] F. J. Provost, T. Fawcett, and R. Kohavi, "The case against accuracy estimation for comparing induction algorithms," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 445–453.

[19] T. Fawcett, "Using rule sets to maximize roc performance," in *Proceedings of the 2001 IEEE International Conference on Data Mining, ICDM'01*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 131–138.

[20] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[21] G. Brier, "Verification of forecasts expressed in terms of probability," *Monthly Weather Review*, vol. 78, pp. 1–3, 1950.

[22] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.

[23] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of American Statistical Association*, vol. 32, pp. 675–701, 1937.

[24] P. B. Nemenyi, *Distribution-free multiple comparisons. PhD-thesis*. Princeton University, 1963.

[25] C. Sönströd, U. Johansson, and R. König, "Towards a unified view on concept description," in *DMIN*, R. Stahlbock, S. F. Crone, and S. Lessmann, Eds. CSREA Press, 2007, pp. 59–65.