MoMAC : multi-objective optimization to combine multiple association rules into an interpretable classification

Noname manuscript No.
(will be inserted by the editor)

MoMAC: Multi-objective Optimization to combine Multiple Association Rules into an Interpretable Classification

Danh Bui-Thi · Pieter Meysman · Kris Laukens

Received: date / Accepted: date

Abstract A crucial characteristic of machine learning models in various domains (such as medical diagnosis, financial analysis, or real-time process monitoring) is the interpretability. The interpretation supports humans in understanding the meaning behind every single prediction made by the machine, and enables the user to assess trustworthiness before acting on the predictions. This article presents our work in building an interpretable classification model based on association rule mining and multi-objective optimization. The classification model itself is a rule list, making a single prediction based on multiple rules. The rule list consists of IF ... THEN statements that are understandable to humans. We choose these rules from a large set of pre-mined rules according to an interestingness measure which is formulated as a function of basic probabilities related to the rules. We learned the interestingness measure through multi-objective optimization, concentrating on two objectives: the classifier's size in terms of number of rules and prediction accuracy. The model is called MoMAC, "Multi-Objective optimization to combine Multiple Association rules into an interpretable Classification". The experimental results on benchmark datasets demonstrate that MoMAC outperforms other existing rule-based classification methods in terms of classification accuracy.

Keywords rule based classification · association rule mining · multi-objective optimization · interestingness measures

1 Introduction

Machine learning models are often branded as black-boxes because they do not clearly explain or identify the logic behind their predictions in terms that humans can understand [32]. Especially in domains with significant social or financial impact such as healthcare, criminal justice or financial analysis, interpretability of the model is often extremely important. An interpretable model offers relevant information underlying the decision process to the domain expert, so they can assess

D. Bui-Thi, P. Meysman, K. Laukens
Adrem Data Lab, Department of Computer Science, University of Antwerp, Belgium
E-mail: {danh.bui-thi, pieter.meysman, kris.laukens}@uantwerpen.be

trustworthiness, further analyse the decision process and take appropriate action. Moreover, machine learning is susceptible to bias from training data which can teach models to discriminate against protected groups without oversight or intervention [25]. In such cases, interpretability can be useful to detect bias in machine learning models. To tackle this black-box issue, different studies have followed two possible strategies: creating models that are inherently interpretable or building separate models that can explain the black-box model [32]. Our study follows the former, creating an inherently interpretable model which gives its own explanation about what the model actually computes.

The most well-known interpretable machine learning models are decision trees and decision rules. Decision trees recursively split the feature space into partitions to differentiate between classes. CART, ID3 and C4.5 are commonly used decision tree methods [27, 29]. Each split in these trees is determined in isolation without considering the possible impact of future splits. As a result, these trees can fail to capture the underlying characteristics of data, and often suffer from over-fitting. However finding optimal decision trees is well-known as a NP-hard problem [3]. Meanwhile, decision rules are lists of *if ... then ...* statements in which the *if* statements define a partition of the feature set and the *then* statements correspond to the classes. A small and accurate decision list can be constructed corresponding to the multivariate splits denoted by the rules. Compared to optimal decision trees, searching for optimal decision rules is more practically feasible, because pre-mined rules can reduce the search space to that of rule permutations as opposed to all possible sets of splits.

Several methods have been proposed to find optimal decision rules. In general, they can be grouped into two categories: rule-list based and rule-set based approach. Rule-list based approach takes the order of rules into account, the rules with high priority are considered first when making decision. CBA [21], CMAR [20] and CPAR [42] are well-known associative classifiers which follow the rule-list based approach. CBA and CMAR select their rules greedily from pre-mined rules according to confidence and support of the rules while CPAR builds its rule list by adding literals one by one using a gain score. To do a better rule ranking, Mattiev et al.[23] and Rajab et al. [30] append the number of items in *if* statement to their ranking factor list, besides confidence and support. In another study, Venturini et al. [36] aims to scale associative classifiers for large datasets. They proposed to split data into partitions, learning an independent model for each partition and finally merging them into a single model. Rule-pruning techniques were also considered in order to retain only statistically significant rules, such as the minimum χ^2 threshold in Venturini et al. [36] and Fisher's exact test in Sood et al [35]. Emerging pattern based classifiers [11, 12, 14, 19] are also rule-list based methods. They choose their rules from emerging patterns according to a score that was derived from a support measure. Emerging patterns (EP) are the patterns of which support changes significantly from one class to another class. Instead of using greedy strategy, other studies [2, 4, 18, 31, 37, 41] concentrated on searching fully optimal rule lists. They developed exhaustive search or heuristic search strategies to populate these rule lists. In the rule-set based approach, the order of rules is not important. Classification models in this case are considered as a linear combination of decision rules; there is no "else" statement connecting these rules. As a result, they are also referred to as rule ensembles. Lakkaraju et al. [17] proposed a complex objective function to search for decision rule sets via a smooth local

search. This objective function consists of misclassification error, rule count, rule length, rule overlap, and fraction of classes. Other methods, including RuleFit [15], MLRules [7], ENDER [8], and recently GLRM [38] and LIBRE [24], joined rules into a linear combination form and learned the form through minimizing a loss function indicating the penalty for wrong predictions. These rule ensemble learning algorithms differ greatly in terms of rule generation method and optimization technique.

In this study, we approach building interpretable models as learning optimal rule lists. These lists consist of several rules sorted according to their interestingness. Any prediction on unseen data is based on decisions from the top K matching rules instead of a single rule. To search for optimal rule lists, we exert a greedy strategy in which we prioritize the pre-mined rules based on a relevance measure and subsequently pick rules one by one following their priority until all training samples are covered. We propose a new prioritizing measure from data to substitute for confidence and support, which are popularly used in preceding studies of associative classifiers. The main contributions of our paper can be summarized as follows:

- We define the prioritizing measure as a parameterized function of the basic probabilities related to the rules. Thus, we can approximate the measure through searching the right parameters for the function.
- We develop a novel method to approximate the above measure and build associative classifier simultaneously. Our method is based on a multi-objective optimization technique, focusing on two objectives: the obtained classifier should restrict the size of the rule list and achieve high accuracy in prediction. The optimization will search for the sufficient parameters so that their corresponding measure can produce an associative classifier satisfying these objectives.
- We evaluate our classification model on 12 benchmark datasets, most of which are obtained from UCI repository [13]. Experimental result shows that the model can generate decision rule lists that give better prediction accuracy than the state-of-the-art associative classifiers do.

We named our method MoMAC, i.e Multi-Objective optimization and Multiple Association rules into interpretable Classification. MoMAC itself can be used for both binary classification and multi-class classification problems. The source code of MoMAC model is available at https://github.com/banhdzui/MoMAC-v1.git.

In the next section, we first present related work. This is followed by the methodology section which describes in detail how we formulate interestingness measure as well as how we search for an optimal rule list. Section 4 contains the experimental results and discussion. The final section provides a conclusion and hints towards future work.

2 Related work

The associative classifiers CBA, CMAR, and CPAR collected their rule lists greedily based on ranking measures or interestingness measures. Confidence and support were used in the case of CBA and CMAR because the measures define reliability and generality of rules respectively. They were also commonly used in other studies [23, 30, 35, 36] with other pre-defined measures for the same purpose. According

to the surveys of Geng et al. [16] and Sharma et al. [33], there are more than 30 interestingness measures that can be employed to prioritize rules. However, different measures often provide conflicting rankings, resulting in associative classifiers with different performance. To illustrate this, we ran the CMAR algorithm with a few different existing interestingness measures on different datasets. The resulting micro F1 scores are shown in table 1. We observe a clear change in the prediction accuracy of CMAR as a function of the different measures used. Certain measures bring better performance than confidence in some of the datasets, but their performance fluctuates across datasets as well (e.g. *klosgen* measure). We can conclude that for many studies selecting the proper measure for ranking the rules is a necessity in order to optimize the performance of the associative classifiers. Emerging pattern (EP) based methods use their own measure, named *growth_rate* which is the rate of antecedent's support on a class C and the remaining class. The *growth_rate* was combined with support score to sort mined EPs. In another way, Song et al. [34] proposed a measure, called predictive rate, which is the average of k local predictability values. Each local predictability value is actually the prediction performance of that rule in one local test set which is generated by k-fold cross-validation. In 2008, Yang et al. [40] suggested a personalized association rule ranking method based on Genetic Network Programming (GNP). The method learns a ranking measure which is a linear combination of semantic similarity and other well-known interestingness measures, includes support, confidence, lift and chi-squared. However, the learning step needs support from experts (with domain background) to have ground truth ranking. An extension of this method for a non-linear measure is presented in their later study [39].

	Datasets					
Measures	***anneal***	***breastcancer***	***tcr***	***ionosphere***	***pima***	***tictactoe***
confidence	**0.913**	0.947	0.771	**0.893**	0.763	**0.994**
change of support	0.814	**0.964**	0.775	0.883	0.769	0.942
jaccard	0.805	0.890	0.767	0.729	0.717	0.700
klosgen	0.348	0.889	0.765	0.873	0.759	0.975
lift	0.778	0.814	0.762	0.875	0.697	0.899
zhang	0.878	0.953	**0.783**	0.892	**0.772**	0.987

Table 1: Micro F1 scores of CMAR on different datasets using different interestingness measures

Instead of using interestingness measures, the studies [2, 4, 18, 31, 37, 41] built their rule list based on search algorithms. These methods search for the rules that can optimize some specific objective functions. Rijnbeek et al. [31] developed a branch-and-bound search to find an optimal single rule in Disjunctive Normal Form that optimizes user-defined performance constraints. Angelino et al. [2] devised the CORELS algorithm, which is able to find optimal rule lists that are minimized in both misclassification error and their size in terms of number of rules. They also defined some tight bounds in order to reduce search space efficiently. Other studies [4, 18, 37, 41] chose to examining probabilistic rule lists and implemented Monte Carlo search with or without tight bounds in their approach. One of the common things among these methods is that they were designed for binary classification problems.

In general, the greedy strategy based methods, such as CBA, CMAR, CPAR and EP-based classifiers choose their rule lists according to interestingness measures. These measures are pre-defined based on specific criteria, independently of data. However, identifying which interestingness measure is the best fit for a given dataset is not a trivial task. Other rule-list based models such as CORELS [2] and SoftFRL [4] search for the rules that optimize classifiers' performance on training data. Their search space includes pre-mined rules which can be very large. To reduce the search space, some tight bounds have been exerted. Based on this observation, we propose a novel method that is also based on a greedy strategy but does not use pre-defined interestingness measures. We formulate the interestingness measure as a parameterized function and learn it from training data. The measure is approximated such that it is able to produce a classifier which gives high performance in terms of accuracy on training data and is small in size to avoid overfitting. In addition, most existing interestingness measures are defined from basic probabilities related to the rules, combining them in different forms. Thus, we propose to encode the measure as a small neural network of which the input features are these basic probabilities. Searching for the weights of the network is more practically feasible than working directly on rule permutations.

We modify the Apriori algorithm [1] to generate classification rules, which will be used as pre-mined rules in our framework. Besides Apriori, several algorithms have been proposed throughout the years for mining association [9, 22]. They speed up the rule mining process through different techniques, including parallel computing and heuristic search. We can use these as a tool to generate the pre-mined rules. Basically, we can divide these algorithms into exact and meta-heuristic based approaches [10]. The meta-heuristic based approach is a trade-off between running time and percentage of frequent itemsets extracted. In this work, we focus on selecting interesting rules for associative classifiers. The interestingness of a rule is defined in a measure which can depend on different criteria besides its support. Thus, we opted for an exact approach with a low *minsup* for pre-mined rules, attempting to get as many rules as possible.

3 The MoMAC approach

This section presents our method, Multi-objective Optimization to combine Multiple Association Rules into an Interpretable Classification(MoMAC), in detail. At first, we describe the formulation of a data-driven interestingness measure. Secondly, we demonstrate how the final rule list are selected when the interestingness measure is known. Finally, the optimization framework learning a good interestingness measure from the training data is presented.

3.1 Interestingness measure formulation

As we mentioned in section 2, several interestingness measures have been devised to prioritize association rules and they can be used to build associative classifiers. As none of these measures works best for all datasets, the best measure for each dataset needs to be found or even created. To this end, we define a generic interestingness measure in terms of basic probabilities and learn its form from available

data. This allows us to obtain the best fit of interestingness measure for a specific dataset. As a result, we consider this measure as a "data-driven interestingness measure".

Based on existing interestingness measures, one can recognize that these interestingness measures are created from basic probabilities related to the antecedent and consequent of association rules. Therefore, we propose to represent rules as feature vectors x_r of which features are these basic probabilities and develop a multi-layer neural network to formulate our data-driven measure. Given an association rule $r : A \rightarrow B$, we use 15 basic probabilities in total. They are: $P(A)$, $P(\overline{A})$, $P(B)$, $P(\overline{B})$, $P(A\,B)$, $P(\overline{A}\,B)$, $P(A\,\overline{B})$, $P(\overline{A}\,\overline{B})$, $P(A)P(B)$, $P(A|B)$, $P(A|\overline{B})$, $P(\overline{A}|B)$, $P(\overline{A}|\overline{B})$, $P(B|A)$, $P(\overline{B}|A)$ where $P(\cdot)$ denotes the probability of the corresponding event that is estimated from training data. These values are then encoded by a neural network and mapped into a single value which is corresponding interestingness score $I(r)$.

Figure 1 illustrates the structure of the neural network. It consists of $|x_r|$ input nodes, two hidden layers and only one output node, Sigmoid is used as activation function in every node. The hidden layers are configured so that they half in number for each subsequent layer, as is a common approach to condense information as it passes through the network. The weights of the neural network are trained from the data by optimizing an objective function which is the trade-off between the misclassification error and the size of selected rule list. The details of the optimization framework are presented in section 3.3. The trained interestingness measure plays an important role in prioritizing association rules and predicting classes for upcoming data.

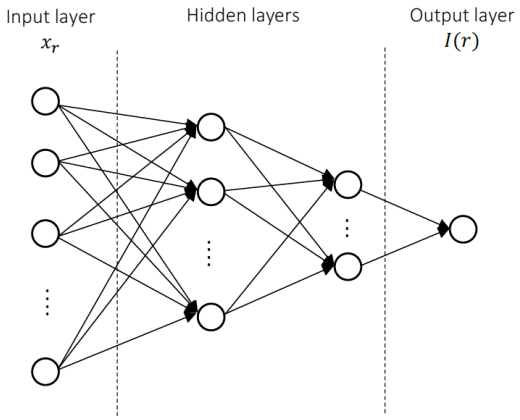


Fig. 1: An illustration of a multi-layer neural network formulating the interestingness measure. The input nodes correspond to basic probabilities of the rule, while the output node represents the interestingness score. The network consists of 2 hidden layers which contain 8 and 4 hidden nodes, respectively.

Algorithm 1 Rule selection based on database coverage

Input:
Training data D
Pre-mined association rules, *Rules*
A database coverage threshold $K \geq 1$. Default is $K = 3$.
An interestingness measure $I(r)$
Output: A subset of rules for classification C.

```
Put Rules in a priority queue Q whose priority is (I(r), support_r).
cover_count ← 0 for each training sample.
while Q not empty and ∃cover_count < K do
    r ← dequeue(Q)
    for each sample D[i] with cover_count < K do
        if D[i] match the rule r then
            Increase cover_count of the i^th sample to 1
        end if
    end for
    if the rule r covers at least one sample then
        Add the rule r to C
    end if
end while
Select a default class for remaining data (if yes).
```

3.2 Rule selection

Algorithm 1 illustrates how we select a subset of rules from pre-mined classification rules for classifiers. Given a training data D, an interestingness measure $I(r)$ and a database coverage threshold K, the algorithm pushes the association rules into a priority queue of which the priority is defined as the tuple *(interestingness score, support)*. Rules with a higher interestingness score are in higher priority, which are dequeued first. In the case that interestingness scores of two rules are identical, their supports are compared. Thanks to the priority queue, the rules are examined one by one until the training data is fully covered or there is no more rule to check. A data sample is considered as "covered" if it matches with at least K selected rules. A data sample matches with a rule if the left hand side of the rule is a subset of that sample. The algorithm is thus similar to the CMAR method [20] in concept, but there are two major differences. First, we eliminate the pruning step based on χ^2 testing which CMAR uses to filter low positive correlated rules. Secondly, we still add rules that do not make a correct decision into the final list instead of ignoring them. These modifications enable the optimization framework to learn a sufficient measure, without pruning step bias.

3.3 Optimization Framework

Assume that we build an associative classifier for a multi-class classification problem on a training dataset $D = \{(x_i, y_i)\}_{i=1}^{m}$ where x_i is sample features and $y_i \in \{0, .., L-1\}$ indicates the class label of the i^{th} data object; L is the number of unique classes. The classifier consists of an ordered rule list $R = (r_1, r_2, .., r_n)$ and a default rule r_0. Each association rule r_i in R has the form of $A_i \rightarrow B_i$ where A_i is an item-set and B_i is a class label while the default rule has the form of $true \rightarrow B_0$. As described in previous section, each rule r_i is represented as a

15-dimensional feature vector x_{r_i}. Given an interestingness measure $I(r, W)$, an associative classifier can be built using Algorithm 1. Here W denotes the weights of the neural network which need to be learned.

To classify a data object x_i, the classifier consults the first K rules in R of which antecedents match with x_i. Let R_K denote the indices of these matching rules, then $P(y_i = l|x_i)$ is computed as follows:

$$P(y_i = l|x_i) = \frac{z_l}{\sum_{j=1}^{L} z_j} \tag{1}$$

where

$$z_l = \sum_{j \in R_K} 1\{B_j = l\} I(r_j, W) + 1\{B_j \neq l\} \frac{1 - I(r_j, W)}{L - 1} \tag{2}$$

$1\{B_j = l\}$ is the indicator function: it is equal to 1 if the consequent of r_j is the class label l; otherwise it is equal to 0. If none of the rules in R match with the object, then only the default rule is used. In this case, $P(y_i = B_0|x_i) = 1.0$ and $P(y_i \neq B_0|x_i) = 0.0$. Below is an example of computing $P(y_i = l|x_i)$ for a 3-class associative classifier. Suppose that there is a data object x matches with $K = 3$ following rules in the classifier:

- $r_1 : a_1, a_3 \rightarrow l = 0$, $I(r_1, W) = 1.0$
- $r_2 : a_5 \rightarrow l = 0$, $I(r_2, W) = 0.9$
- $r_3 : a_2 \rightarrow l = 2$, $I(r_3, W) = 0.8$

then $z_0 = 2.0, z_1 = 0.15$ and $z_2 = 0.85$. According to equation 1, we have $P(y = 0|x) = 0.67, P(y = 1|x) = 0.05$ and $P(y = 2|x) = 0.28$. This means the object x is classified to the class 0.

The optimization is in fact an iterative process between finding the interestingness measure and building the associative classifier. Firstly, we choose some random interestingness measures, constructing respective classifiers. We then run the classifiers to predict training data, their performance is used as fitness to update the current interestingness measures to their better instances. In this context, we are modifying the parameters W to update the interestingness measure. The optimal W would produce an associative classifier that is low in classification error and small in size. Therefore, we formulate our problem in terms of a multi-objective optimization as follows:

$$\begin{aligned} &\underset{W}{\text{argmin}}\left(\frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(x_i, y_i, R_W), |R_W|\right) \\ &s.t. \frac{1}{|X_l|}\sum_{j \in X_l} \mathcal{L}(x_j, y_j, R_W) \leq c, X_l = \{k|y_k = l\} \\ &|R_W| \leq N \end{aligned} \tag{3}$$

where, R_W denotes the rule list corresponding to the parameters W, $|R_W|$ is the size of the rule list R_W, $\mathcal{L}(x_i, y_i, R_W)$ is the classification error for a sample (x_i, y_i) using K best matching rules. c denotes the maximum average classification error for each class and N denotes the maximum number of rules that can be included in classifiers. Both c and N are user-defined thresholds. As we can observe, the optimal W need to deal with the trade-off between two conflicting objectives:

the average classification error and the size of selected rule list. It also takes into account constraints related to classification error for each class and maximum size of the classifier. We compute $\mathcal{L}(x_i, y_i, R_W)$ according to:

$$\mathcal{L}(x_i, y_i, R_W) = -\sum_{l=1}^{L} 1\{y_i = l\} log P(y_i = l|x_i) \tag{4}$$

where $1\{y_i = l\}$ is the indicator function; it is equal to 1 if the ground truth label y_i is l; otherwise it is equal to 0. The above cost function brings us a more detailed look into classification error than just using binary values which represent for correct and incorrect prediction. This can help the model to find a soft split between classes.

To solve above problem, we use Non-dominated Sorting Genetic Algorithm II, NSGAII [6], a Pareto-based multi-objective evolutionary algorithm. NSGAII is derivative-free and it can be used for numerical optimization of non-linear or non-convex continuous problems. NSGAII is an evolutionary algorithm, thus it works on the repeated interplay of variation and selection. In the variation part, offsprings are generated from parents that are selected using binary tournament selection. The crossover operator and mutation operator are simulated binary crossover and poly-nominal mutation respectively. In the selection part, the best individuals with respect to a ranking are selected as the new population. This ranking procedure consists of two levels: non-dominated sorting and crowding distance sorting. In non-dominated sorting, individuals p are sorted into their domination count, which is the number of individuals which dominate p. Those individuals in the first non-dominated front have their domination count as zero. To the individuals who share same non-dominated rank, they would be ranked at second level according to their crowding distance. The crowding distance values are computed as the sum of individual distance values corresponding to each objective. When the optimization process is done, non-dominated or Pareto optimal solutions can be used as potential candidates for the last solution. A solution is non-dominated, or Pareto optimal if none of the objective functions can be improved in value without deteriorating some of the other objective values. For our problem, each assignment of the weights W is an individual in NSGAII. We flatten them into a single vector and run the algorithm to find the Pareto optimal solutions for the problem defined in equation 3. Figure 2 shows the flowchart of the algorithm. $W_i^{(k)}$ denotes the i^{th} individual at the k^{th} generation. At the start, a set of Ws are initialized randomly. The individuals are then evolved through the variation and selection operators, depending on their objective function values. The individuals remaining in the last generation are the candidates for the optimal W. The green panel in the right side illustrates how the objective functions are evaluated. Each individual $W_i^{(k)}$ defines an interestingness measure, a Sigmoid function or a neural network. The measure is then used in the rule selection procedure to generate an associative classifier. When the procedure accomplishes, we also obtain the size of the classifier and its classification error on training data.

Of the Pareto optimal solutions, some could bring classifiers that perform in high accuracy on training data but are large in their size. This often happens when rules having high confidence but low support are considered first. Besides these solutions, there are some others produce classifiers that contain much less rules but compensate this for their performance. Rules with high support but low confidence

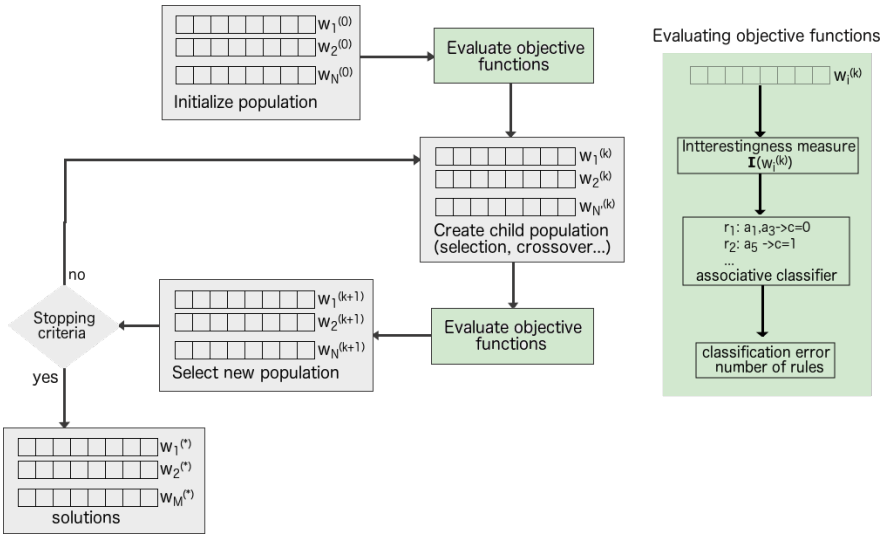


Fig. 2: The flowchart of the optimization process, searching Pareto optimal solutions for W.

may be ranked on the top of rule lists in this case. The solutions placed between the two extremes could be promising ones, which represent to different interestingness measures. To provide an interactive view that allows users to select a desirable solution, we display all found Pareto optimal solutions on a single 2D scatter plot where the x-axis illustrates the classification error and the y-axis corresponds to the size of classifiers. Alongside this information, classification error for each class are shown as well. Figure 3 is an example of all Pareto solutions obtained on Anneal dataset [13]. The common guideline that we followed was to use the model at the hinge point between small classifiers and classification accuracy.

4 Experiments

In this section, we present a detailed experimental evaluation of the proposed MoMAC algorithm. To demonstrate the added benefit of using a complex neural network approach, we define a simple linear combination as a baseline comparison. For this baseline, we apply a Sigmoid function to scale the interestingness value to range $[0, 1]$ as follows:

$$I(r, w, b) = \frac{1}{1 + e^{-(w^T x_r + b)}} \tag{5}$$

where,

- x_r: a feature vector of the rule r. It consists of the basic probabilities related to the rule
- w: a weight vector associated to rule's features
- b: a bias

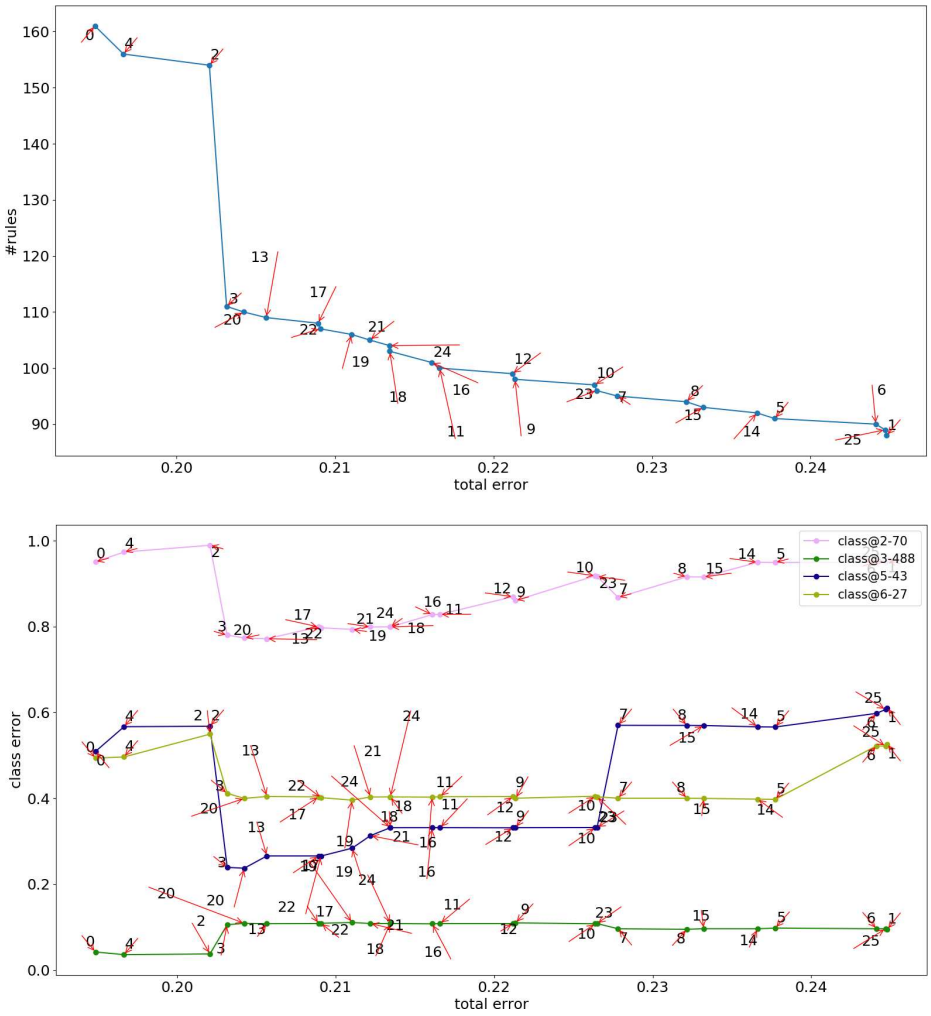


Fig. 3: An illustration of the non-dominated solutions on the Anneal dataset. The top plot shows the classification error and the number of rules while the bottom plot shows the classification error on every class. Each annotated point indicates a non-dominated solution. From the plots, we can observe that the first solution obtains the lowest training error, less than 0.2, with 160 rules. The number of rules then decreases significantly at the solutions 3 and 20, while corresponding training error increases slightly, around 0.005. These solutions provide the lowest classification error for individual classes, except *class@3*. Therefore, solution 3 or 20 are good candidates for the optimal solution on the Anneal dataset.

The weights are learned in a similar manner as that of the neural network presented above. This baseline is referred to as MoMACSig in the results, while the more advanced method that uses the neural network is referred to as MoMACNet. We compare our linear MoMACSig and non-linear MoMACNet with other existing

classification approaches in terms of classification accuracy, model size, and execution time. The comparisons include CBA [21], CMAR [20], CORELS [2], FRL [4], SoftFRL [4], RuleFit [15], GLRM [38], LIBRE [24] and Random Forest (RF) from scikit-learn library [28] with *max depth* of 10 and 100 trees. We run NSGAII algorithm with population size of 100 and used $10,000$ function evaluations.

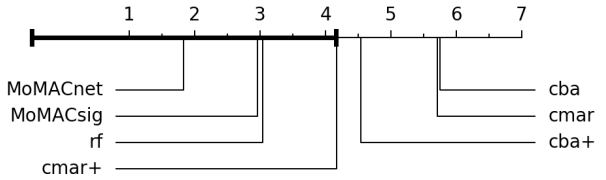
All methods are compared on 12 benchmark datasets collected from the UCI repository [13], the T-cell receptor (TCR) dataset [5], and the NYCLU 2014 stop-and-frisk dataset[26]. The goal of the TCR dataset is to predict the target T-cell antigen based on the T-cell receptor beta-chain CDR3 amino acid sequence. The NYCLU 2014 is explored to predict whether the stop can lead to an arrest. Datasets containing continuous attributes are discretized first to ensure that they only have categorical attributes. Pairs of *attribute-value* are then considered as items in transaction databases.

To evaluate the performance of these classifiers, we use 5-fold cross validation and compute the average micro F1 score. Along with the original CBA and CMAR using confidence as a ranking score, multiple alternative versions of CBA and CMAR were evaluated with the other interestingness measures from Geng et al. [16]. For simplicity, we only report the CBA and CMAR versions with the best performance on each dataset as CBA+ and CMAR+ respectively.

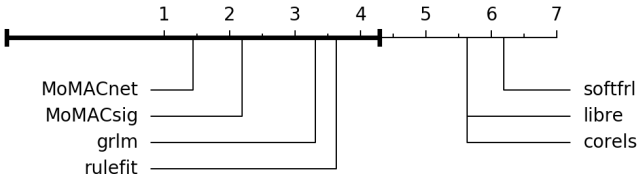
4.1 Classification performance

Table 2 reports the average micro F1 score of the classification models on the benchmark datasets: (a) for the multi-class classification models and (b) for the binary classification models. Using the Friedman test, F_F for the Table 2(a) is 9.533 and for the Table 2(b) is 19.243. Both are larger than associate critical value of the F distribution for $\alpha = 0.05$, so we reject the null hypothesis. Figure 4 visualizes the result of the post-hoc test, where we used Bonferroni-Dunn test to compare all algorithms with MoMACNet. The diagrams show that the performance of CBA, CBA+, CMAR, CMAR+, CORELS, SoftFRL and LIBRE are significantly worse than MoMACNet, but the difference with Random Forest, RuleFit, GLRM and MoMACSig was not found to be significant. However, MoMACNet has a lower average rank compared to these methods on the experimental datasets, and thus often more often scores as the best or one of the best methods.

Figure 5 illustrates the classification error of MoMACNet classifiers on training and testing data of four datasets: breast cancer, credit, ionosphere, and pima. These classifiers are created using Pareto optimal solutions found by NSGAII algorithm. For simplicity, we sort the Pareto optimal solutions of each dataset in ascending order of corresponding classifier's size. They are then represented as indices on x-axis from left to right. From the plots, we can observe that different interestingness measures had been found. We derived classifiers from the obtained measures, ranging the classifiers from small to large one. We also observe the decrease of the error on training data when the models are based on more rules. The solutions at the most left part of the plots tend to produce under-fitting models while the ones at the most right part tend to generate over-fitting models. The error on the testing data fluctuates among the solutions. However, in general the error goes down when the model contains more rules and it then goes up when the model grows over a limit.



(a) Multi-class classification models



(b) Binary classification models

Fig. 4: Comparison of MoMACNet against the others with the Bonferroni-Dunn test. The top line in the diagram is the axis representing the average ranks of methods. All classifiers with ranks outside the marked interval are significantly different ($p < 0.05$) from MoMACNet.

4.2 Classifier size

Table 3 shows the average size of the rule-based classifiers on the benchmark datasets with Table 3(a) for the multi-class classifiers and 3(b) for the binary classifiers. From the tables we can see that CORELS, CBA(+), SoftFRL and GLRM are small classifiers in terms of rule quantity compared to RuleFit, CMAR(+), MoMACSig and MoMACNet. CMAR and MoMAC tend to produce larger rule lists compared to other methods like CBA, SoftFRL, CORELS due to their principle in selecting rules: every training sample needs to be covered by $K = 3$ rules instead of a single rule. If we narrow down the comparison to the methods that requires high coverage, which includes MoMAC, RuleFit and CMAR, then the MoMAC algorithms on average gives the smallest models. Moreover, the neural-network-based interesting measure generates better models than the linear baseline in both classification accuracy and model size.

4.3 Computation time

Figure 6 reports training time of four representative methods RuleFit, SoftFRL, MoMACSig and MoMACNet on five datasets. We can observe a big difference in the training time between the MoMAC methods and RuleFit. However, the MoMAC models are trained much faster than SoftFRL in most datasets. The computation time of MoMAC methods depends on the number of pre-mined rules and the size of the training data. As we mined all possible rules from the data, the number of the pre-mined rules can be very large for some datasets. In addition, repeatedly computing classification error during the optimization step takes

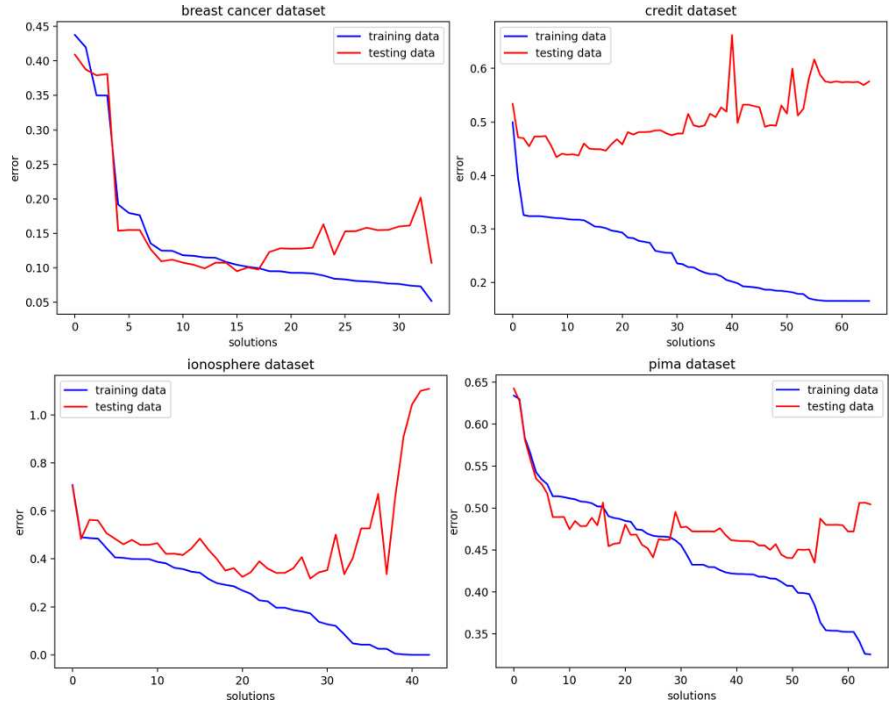


Fig. 5: Classification error of MoMACNet classifiers on training and testing data of different datasets. The x-axis represents Pareto optimal solutions, each which is associated to a classifier. The solutions are sorted in ascending order of classifier's size and they are represented as indices from left to right of the axis.

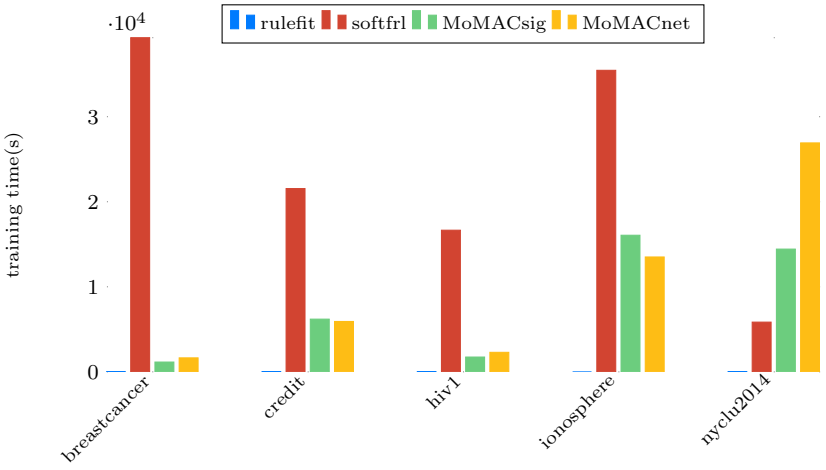


Fig. 6: Training time of four methods RuleFit, SoftFRL, MoMACSig and MoMACNet on five datasets.

Datasets	Methods						
	CBA	***CMAR***	***CBA+***	***CMAR+***	***RF***	***MoMACSig***	***MoMACNet***
anneal	0.923(0.021)	0.913(0.032)	0.923(0.021)	0.913(0.032)	**0.945(0.024)**	0.938(0.028)	0.934(0.029)
breast cancer	0.953(0.020)	0.947(0.019)	0.953(0.020)	0.964(0.008)	0.964(0.006)	0.963(0.014)	**0.967(0.017)**
credit	0.851(0.024)	0.856(0.026)	0.856(0.023)	0.858(0.025)	0.874(0.02)	0.876(0.026)	**0.877(0.027)**
hiv-1 cleavage	0.863(0.044)	0.863(0.027)	0.877(0.031)	0.879(0.050)	0.907(0.022)	0.903(0.029)	**0.912(0.012)**
ionosphere	0.902(0.039)	0.893(0.027)	0.920(0.037)	0.893(0.027)	0.920(0.041)	0.920(0.019)	**0.936(0.022)**
nyclu2014	0.811(0.007)	0.773(0.021)	0.812(0.007)	0.787(0.008)	0.812(0.011)	**0.818(0.007)**	**0.818(0.008)**
pima	0.754(0.044)	0.763(0.044)	0.771(0.032)	0.772(0.044)	0.756(0.037)	0.770(0.046)	**0.777(0.030)**
sonar	0.794(0.076)	0.769(0.068)	0.794(0.076)	0.769(0.068)	**0.807(0.056)**	0.788(0.044)	**0.807(0.070)**
tcr	0.738(0.053)	0.771(0.020)	0.792(0.022)	0.798(0.016)	0.776(0.017)	0.782(0.020)	**0.797(0.021)**
tic tac toe	0.993(0.005)	0.994(0.006)	0.993(0.005)	0.994(0.006)	0.982(0.005)	**0.995(0.007)**	0.993(0.005)
vehicles	0.676(0.035)	0.684(0.033)	0.676(0.035)	0.684(0.033)	**0.708(0.048)**	0.697(0.040)	0.700(0.029)
waveform	0.790(0.010)	0.800(0.006)	0.790(0.010)	0.804(0.014)	**0.821(0.010)**	0.808(0.020)	0.812(0.012)
average rank	5.75	5.708	4.542	4.167	3.042	2.958	**1.833**

(a)

Datasets	Methods						
	CORELS	***SoftFRL***	***RuleFit***	***GLRM***	**LIBRE**	***MoMACSig***	***MoMACNet***
breast cancer	0.934(0.014)	0.901(0.046)	0.957(0.012)	0.958(0.012)	0.961(0.011)	0.963(0.014)	**0.967(0.017)**
credit	0.851(0.024)	0.847(0.023)	0.863(0.025)	0.859(0.026)	0.818(0.038)	0.876(0.026)	**0.877(0.027)**
hiv-1 cleavage	0.741(0.055)	0.721(0.051)	0.923(0.03)	**0.934(0.016)**	0.857(0.032)	0.903(0.029)	0.912(0.012)
ionosphere	0.834(0.041)	0.851(0.031)	0.908(0.036)	0.915(0.036)	0.899(0.026)	0.92(0.019)	**0.936(0.022)**
nyclu2014	0.808(0.009)	0.804(0.009)	0.790(0.021)	0.814(0.006)	0.77(0.015)	**0.818(0.007)**	**0.818(0.008)**
pima	0.737(0.038)	0.744(0.049)	0.752(0.038)	0.744(0.036)	0.708(0.045)	0.770(0.046)	**0.777(0.03)**
sonar	0.729(0.066)	0.716(0.036)	0.804(0.06)	0.782(0.078)	0.773(0.044)	0.788(0.044)	**0.807(0.070)**
tic tac toe	0.739(0.01)	0.674(0.012)	0.978(0.009)	0.981(0.01)	0.678(0.013)	**0.995(0.007)**	0.993(0.005)
average rank	5.625	6.1875	3.625	3.3125	5.625	2.1875	**1.4375**

(b)

Table 2: Average micro F1 scores of the classification models on benchmark datasets, where (a) for multi-class classification models and (b) binary classification models.

Datasets	Methods					
	CBA	***CMAR***	***CBA+***	***CMAR+***	***MoMACSig***	***MoMACNet***
anneal	**75(8.276)**	181(5.916)	**75(8.276)**	181(5.916)	182(42.226)	127.8(56.469)
breast cancer	**64(2.345)**	162.8(5.975)	**64(2.345)**	152.4(5.413)	75.6(20.44)	132(77.321)
credit	150.4(11.082)	404(19.609)	**3.2(0.447)**	435.4(23.776)	92(88.043)	70.8(41.632)
hiv-1 cleavage	**171.2(25.233)**	260.4(21.732)	196.4(20.852)	360(47.569)	235.2(92.654)	202.2(81.266)
ionosphere	**65.4(3.507)**	181.2(6.261)	**65.2(3.834)**	189.8(20.03)	87.8(18.089)	81.6(17.672)
nyclu2014	380.6(19.932)	162.8(19.93)	362.2(17.992)	**45.2(6.496)**	416.8(427.204)	463(510.944)
pima	182.2(11.798)	183.4(14.553)	**25.2(2.049)**	206.8(11.883)	220(108.187)	156.4(89.187)
sonar	**61.2(6.723)**	169.4(11.480)	**61.2(6.723)**	169.4(11.480)	133.4(35.381)	79.8(38.127)
tcr	**44.6(3.647)**	78.8(5.07)	108.8(15.434)	137.6(40.82)	143(43.658)	135.8(50.241)
tic tac toe	**28.2(1.304)**	383(8.515)	**28.2(1.304)**	383(8.515)	137.6(4.393)	143.2(7.19)
vehicles	**238(11.726)**	661.8(25.917)	**238(11.726)**	661.8(25.917)	524.4(89.771)	427.2(149.401)
waveform	**1161.4(16.965)**	2539(52.655)	**1161.4(16.965)**	2588.8(56.327)	2138.75(194.977)	2511.75(405.381)
average rank	2	4.583	**1.667**	5.083	4.25	3.417

(a)

Datasets	Methods						
	CORELS	***SoftFRL***	***RuleFit***	***GLRM***	***LIBRE***	***MoMACSig***	***MoMACNet***
breast cancer	**2(0.0)**	71.8(4.087)	197.4(9.263)	13.8(1.304)	10.2(1.304)	75.6(20.44)	132(77.321)
credit	**2(0.0)**	7.8(2.49)	185(4.062)	12.4(1.14)	85.2(10.686)	92(88.043)	70.8(41.632)
hiv-1 cleavage	**2(0.0)**	18(3.937)	212.4(6.387)	70.2(5.541)	26.2(4.147)	235.2(81.266)	202.2(81.266)
ionosphere	**2(0.0)**	25.6(3.975)	184.4(7.893)	32.2(2.387)	24.2(4.817)	87.8(18.089)	81.6(17.672)
nyclu2014	**2(0.0)**	14(3.082)	433.4(20.367)	25.6(1.14)	184.2(2.588)	416.8(427.204)	463(510.944)
pima	**2(0.0)**	8.4(2.302)	124.2(8.075)	8.4(0.548)	81.8(9.706)	220(108.187)	156.4(89.187)
sonar	**2(0.0)**	17.2(2.049)	205.4(16.994)	38.2(2.28)	35.4(3.912)	133.4(35.381)	79.8(38.127)
tic tac toe	**2(0.0)**	10(4.243)	65(4.637)	18(0.0)	234(13.472)	137.6(4.393)	143.2(7.190)
average rank	**1**	2.4375	6.125	3.3125	3.75	5.875	5.5

(b)

Table 3: Average number of rules of the rule-based classification models for the benchmark datasets: (a) multi-class classification models and (b) binary classification models

significant time when training data is large. Therefore, MoMAC methods are in general slower than CBA, CMAR, CORELS, RuleFit, GLRM and Random Forest but they are still competitive to SoftFRL.

4.4 Interpretability of the MoMAC model

Figure 7 is the snapshot of the optimal rule list generated by MoMACNet on Winscosin breast cancer dataset. This optimal rule list consists of 91 rules whose antecedents are cytological characteristics of breast fine-needle aspirates(FNAs) which were valued on a scale of 1 to 10 and consequent can be benign or malignant. The numbers on the right side of the rules are their interestingness scores. As the

bare nuclei=10 → class=malignant (0.9974)
uniformity of cell size=10 → class=malignant (0.9960)
clump thickness=10 → class=malignant (0.9957)
bare nuclei=10, mitoses=1 → class=malignant (0.9956)
. . .
uniformity of cell size=1 → class=benign (0.9925)
uniformity of cell size=8 → class=malignant (0.9924)
uniformity of cell size=8 → class=malignant (0.9924)
bare nuclei=1, mitoses=1 → class=benign (0.9924)
uniformity of cell size=1, normal nucleoli=1 → class=benign (0.9924)
. . .
clump thickness=5, marginal adhesion=1, bland chromatin=3 → class=benign (0.9918)
single epithelial cell size=4 → class=malignant (0.9917)
clump thickness=5, marginal adhesion=3, mitoses=1 → class=benign (0.9916)
Default: benign

Fig. 7: The snapshot of the optimal rule list generated by MoMACNet on Winscosin breast cancer dataset.

rule lists in the form of if ... then ... statements are interpretable to humans themselves, we can understand what the model learned from the data. However, understanding learned interestingness measure is difficult because we formulated the measure as a fully-connected neural network, which is a black-box model.

5 Conclusion

In this study, we propose a new rule based classification model, MoMAC, for the multi-class classification problem. The MoMAC model consists of an optimal rule list that is greedily selected based on a data-driven interestingness measure. This interestingness measure is learned from data through a multi-objective optimization process which attempts to balance between classifier's size and its performance. Our experiments on benchmark datasets demonstrate significant gains of MoMAC over the existing rule-based classification methods in terms of classification accuracy. However, MoMAC method tends to generate larger rule lists than others including CBA, CORELS, SoftFRL, GLRM and LIBRE. Longer training time is also a drawback of the current implementation of MoMAC.

We also identify some interesting directions for future work. Designing a sufficient data structure to index rule lists can speed up searching satisfied rules. This allows to reduce the computational cost of both training and predicting process, especially for large datasets. We opted for Apriori algorithm, an exact approach, in the experiment to generate pre-mined rules. However, the algorithm suffers from a very high computational cost as well as a high memory cost on large datasets. Using faster, more efficient algorithms for mining rules is also important to speed up our framework. MoMAC model now works merely on positive association rules. Extending the method so that it can include both positive and negative association rules is a promising avenue to improve the resulting classifiers. In addition, one can consider an extension to allow interpretation of the learned interestingness measure beyond the current black box neural network approach. A related question is whether there are general interestingness measures that are shared among multiple datasets or data type groups.

Declarations

Funding: The work was supported by Universiteit Antwerpen under BOF docpro grant to the first author.

Conflicts of interest: Not applicable

Code availability: https://github.com/banhdzui/MoMAC-v1.git

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. Proceedings of the Interntational Conference on Very Large Databases pp. 487–499 (1994)
2. Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., Rudin, C.: Learning certifiably optimal rule lists for categorical data. The Journal of Machine Learning Research **18**, 8753–8830 (2017)
3. Bertsimas, D., Dunn, J.: Optimal classification trees. Machine Learning **106**, 1039–1082 (2017)
4. Chen, C., Rudin, C.: An optimization approach to learning falling rule lists. AISTATS pp. 604–612 (2018)
5. Dash, P., Fiore-Gartland, A.J., Hertz, T., Wang, G.C., Sharma, S., Souquette, A., Crawford, J.C., Clemens, E.B., Nguyen, T.H., Kedzierska, K., et al.: Quantifiable predictive features define epitope-specific t cell receptor repertoires. Nature **547**, 89–93 (2017)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**, 182–197 (2002)
7. Dembczyński, K., Kotłowski, W., Słowiński, R.: Maximum likelihood rule ensembles. Proceedings of the International Conference on Machine Learning pp. 224–231 (2008)
8. Dembczyński, K., Kotłowski, W., Słowiński, R.: Ender: a statistical framework for boosting decision rules. Data Mining and Knowledge Discovery **21**, 52–90 (2010)
9. Djenouri, Y., Belhadi, A., Fournier-Viger, P., Fujita, H.: Mining diversified association rules in big datasets: A cluster/GPU/genetic approach. Information Sciences **459**, 117–134 (2018)
10. Djenouri, Y., Lin, J.C.W., Nørvåg, K., Ramampiaro, H.: Highly efficient pattern mining based on transaction decomposition. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 1646–1649. IEEE (2019)
11. Dong, G., Li, J.: Emerging pattern based classification. Encyclopedia of Database Systems p. 985 (2009)

12. Dong, G., Zhang, X., Wong, L., Li, J.: CAEP: Classification by aggregating emerging patterns. International Conference on Discovery Science pp. 30–42 (1999)
13. Dua, D., Graff, C.: UCI machine learning repository (2019). URL `http://archive.ics.uci.edu/ml`
14. Fan, H., Ramamohanarao, K.: Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers. IEEE Trans. Knowl. Data Eng **18**, 721–737 (2006)
15. Friedman, J.H., Popescu, B.E.: Predictive learning via rule ensembles. The Annals of Applied Statistics **2**, 916–954 (2008)
16. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Computing Surveys (CSUR) **38**, 9–es (2006)
17. Lakkaraju, H., Bach, S.H., Leskovec, J.: Interpretable decision sets: A joint framework for description and prediction. Proceedings of the International Conference on Knowledge Discovery and Data Mining pp. 1675–1684 (2016)
18. Letham, B., Rudin, C., McCormick, T.H., Madigan, D.: Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. The Annals of Applied Statistics **9**, 1350–1371 (2015)
19. Li, J., Dong, G., Ramamohanarao, K.: Instance-based classification by emerging patterns. European Conference on Principles of Data Mining and Knowledge Discovery pp. 191–200 (2000)
20. Li, W., Han, J., Pei, J.: CMAR: accurate and efficient classification based on multiple class-association rules. IEEE International Conference on Data Mining pp. 369–376 (2001)
21. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. KDD pp. 80–86 (1998)
22. Luna, J.M., Fournier-Viger, P., Ventura, S.: Frequent itemset mining: A 25 years review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **9**(6) (2019)
23. Mattiev, J., Kavšek, B.: A compact and understandable associative classifier based on overall coverage. Procedia Computer Science **170**, 1161–1167 (2020)
24. Mita, G., Papotti, P., Filippone, M., Michiardi, P.: LIBRE: Learning interpretable boolean rule ensembles. In: International Conference on Artificial Intelligence and Statistics, pp. 245–255. PMLR (2020)
25. Molnar, C.: Interpretable Machine Learning: A Guide for Making Black Box Models Explainable. Leanpub (2018)
26. New York Civil Liberties Union: Stop and frisk data. `http://www.nyclu.org/content/stop-and-frisk-data` (2014)
27. Paul, Y., Kumar, N.: A comparative study of famous classification techniques and data mining tools. In: Proceedings of ICRIC, pp. 627–644. Springer (2020)
28. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
29. Quinlan, J.R.: C4.5: Programs for machine learning. Elsevier (2014)
30. Rajab, K.D.: New associative classification method based on rule pruning for classification of datasets. IEEE Access **7**, 157783–157795 (2019)
31. Rijnbeek, P.R., Kors, J.A.: Finding a short and accurate decision rule in disjunctive normal form by exhaustive search. Machine Learning **80**, 33–62 (2010)
32. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**, 206–215 (2019)
33. Sharma, R., Kaushik, M., Peious, S.A., Yahia, S.B., Draheim, D.: Expected vs. unexpected: selecting right measures of interestingness. In: International Conference on Big Data Analytics and Knowledge Discovery, pp. 38–47. Springer (2020)
34. Song, K., Lee, K.: Predictability-based collective class association rule mining. Expert Systems with Applications pp. 1–7 (2017)
35. Sood, N., Zaiane, O.: Building a competitive associative classifier. In: International Conference on Big Data Analytics and Knowledge Discovery, pp. 223–234. Springer (2020)
36. Venturini, L., Baralis, E., Garza, P.: Scaling associative classification for very large datasets. Journal of Big Data **4**(1), 1–24 (2017)
37. Wang, F., Rudin, C.: Falling rule lists. Artificial Intelligence and Statistics pp. 1013–1022 (2015)
38. Wei, D., Dash, S., Gao, T., Günlük, O.: Generalized linear rule models. Proceedings of the International Conference on Machine Learning pp. 6687–6696 (2019)

39. Yang, G., Mabu, S.M., Shimada, K., Gong, Y., Hirasawa, K.: Ranking association rules for classification based on genetic network programming. Proceedings of the Annual Conference on Genetic and Evolutionary Computation pp. 1917–1918 (2009)
40. Yang, G., Shimada, K., Mabu, S., Hirasawa, K.: A nonlinear model to rank association rules based on semantic similarity and genetic network programming. IEEE Trans. on Electrical and Electronic Engineering pp. 1–9 (2008)
41. Yang, H., Rudin, C., Seltzer, M.: Scalable bayesian rule lists. Proceedings of the International Conference on Machine Learning **70**, 3921–3930 (2017)
42. Yin, X., Han, J.: CPAR: Classification based on predictive association rules. Proceedings of the SIAM International Conference on Data Mining pp. 331–335 (2003)