

Towards Fast Finding Optimal Short Classifiers

Egor Dudyrev^{*,†}, Sergei O. Kuznetsov[†]

HSE University, 20 Myasnitskaya St, Moscow, 101000, Russian Federation

Abstract

Studies on Explainable Artificial Intelligence show that a model should be small in order to be human understandable. The restriction on the size of a model drastically reduces the space of possible solutions. Many rule learning models still rely on greedy algorithms for generating ensembles of decision trees. This paper discusses FCA-inspired mathematical and engineering techniques to efficiently find most optimal short binary classifiers, i.e., classifiers that consist of no more than three binary attributes and are optimal w.r.t. F1 score.

Keywords

Supervised Machine Learning, Explainable Artificial Intelligence, Formal Concept Analysis

1. Introduction

Studies on Explainable Artificial Intelligence show that a model should be small in order to be human understandable.

Modern learning models such as Gradient Boosting over Decision Trees [1] or Random Forest [2] are universally recognized as mainstream state-of-the-art solutions for binary classification tasks. However, such models are too complex to be analyzed and to be used in trust requiring scenarios even with the help of XAI [3] [4]. This is a reason for the rising tendency of rejecting complex black box models in favor of small explainable ones [5] [6].

Besides making models highly explainable, restricting the size of model also drastically limits the space of all possible models. Thus, one can search for a globally optimal model instead of approaching locally optimal ones. This paper considers the models operating only up to three binary attributes. In order to find an optimal model meeting these conditions, one should iterate through all combinations of up to three given binary attributes. However, such brute-force algorithm suffers from combinatorial explosion with the increasing number of attributes. This paper dwells on both mathematical and engineering techniques based on Formal Concept Analysis (FCA) [7] that make this brute-force algorithm more efficient.

The challenge of constructing simple logical models has been addressed in many areas of previous research. In the early 1960s, the work on formal logic led to the inception of logical

Published in Sergei O. Kuznetsov, Amedeo Napoli, Sebastian Rudolph (Eds.): The 10th International Workshop "What can FCA do for Artificial Intelligence?", FCA4AI 2022, co-located with IJCAI-ECAI 2022, July 23 2022, Vienna, Austria, Proceedings, pp. 23–34.


*Corresponding author.

†These authors contributed equally.

✉ eo.dudyrev@hse.ru (E. Dudyrev); skuznetsov@hse.ru (S. O. Kuznetsov)

🆔 0000-0002-2144-3308 (E. Dudyrev); 0000-0003-3284-9001 (S. O. Kuznetsov)

© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

programming and rule-learning algorithms [8], [9]. The latter – including algorithms as Skope-Rules, RuleFit [10] and more – often rely on greedy approaches to extract short rules from more complex models (such as large decision trees). Contrary to these greedy approaches resulting in locally optimal models, this paper tackles the problem of finding the most optimal model of given size.

Another possible reason to develop and to use short models is that they make good benchmark for big black box models. Indeed, if a short model outputs the same prediction quality as a big black box then there is no interest in using the latter.

Extensive research shows that a man can operate with premises having no more three plus minus one ideas in his head simultaneously [11] [12]. In this paper, due to complexity constraints, we concentrate on finding the rules with premises having no more than three attributes for two reasons. Since it is our first approach to the problem, we should try to solve its easiest version. In addition, the choice of number three is justified by the cognitive studies. Thus, short rules consisting of more than three attributes represent a specific class of explainable machine learning models.

The structure of the paper is as follows. Section 2 presents the theoretical background used throughout the paper. Section 3 describes the mathematical techniques for optimizing the algorithm by minimizing the number of required operations. Complementing this, Section 4 discusses the engineering approaches to optimize the algorithm by maximizing the computation speed. Section 5 merges all the discussed techniques together in one algorithm. And Section 6 presents the experimental results of this algorithm. Finally, Section 7 concludes the paper.

2. Theoretical Background

This subsection introduces definitions we use throughout the paper. Firstly, we provide the basic terms of Formal Concept Analysis to describe the rule models. Secondly, we describe the space of premises that contains the machine learning models discussed in the paper. Thirdly, we describe the main topics of a binary classification in the language in the FCA notation.

2.1. Formal Concept Analysis

A formal context K describes the dataset to build the model on. It is presented as a triple $K = (G, M, I)$ where G is a set of objects (rows in a dataset), M is a set of attributes (columns in a dataset), and $I \subseteq G \times M$ represents relations between objects and attributes.

Prime ($'$) operators match a subset of objects $A \subseteq G$ and a subset of attributes B such that the objects from A are “described” by the attributes from B and vice versa:

$$A' = \{m \in M \mid \forall g \in A : gIm\} \quad B' = \{g \in G \mid \forall m \in B : gIm\} \quad (1)$$

Given a subset of attributes $B \subseteq M$, the subset of objects $A = B'$ is called **extent** of B . Dually, the subset of attributes $B = A'$ is called **intent** of A .

2.2. Premises

In this subsection we define a premise as a combination of attributes of a formal context, joined by conjunction, disjunction, and negation operations. The notion of a premise is necessary for the following subsections.

Definition 1. The **premise space** \mathbb{P} is a set of all combinations of attributes M constructed with conjunction \wedge , disjunction \vee , and negation $\bar{}$ operations:

$$\begin{aligned} \mathbb{P} \text{ is a set s.t.} \\ 1) M \subset \mathbb{P}, \\ 2) \forall p, q \in \mathbb{P} : p \wedge q, p \vee q, \bar{p} \in \mathbb{P} \end{aligned} \quad (2)$$

Each premise $p \in \mathbb{P}$ corresponds to a subset of objects $p' \subseteq G$ called an extent of a premise. Extents of conjunction, disjunction, and negation operations are defined as follows:

$$(p \wedge q)' = p' \cap q', \quad (p \vee q)' = p' \cup q', \quad \bar{p}' = G \setminus p' \quad (3)$$

This paper is specifically interested in premises consisting of no more than three attributes. Generally, a set of premises P_n constructed from n attributes can be described in the following way:

$$\begin{aligned} P_1 &= M \cup \{\bar{m} \mid m \in M\} \\ P_{n \in \mathbb{N}} &= \bigcup_{i=1}^{\lfloor n/2 \rfloor} \{p \wedge q, p \vee q, \overline{p \wedge q}, \overline{p \vee q} \mid p \in P_i, q \in P_{n-i}\} \end{aligned} \quad (4)$$

Uniting sets of premises P_n for each natural number n we obtain the premise space \mathbb{P} : $\mathbb{P} = \bigcup_{n \in \mathbb{N}} P_n$. In what follows, we say that premise $p \in \mathbb{P}$ has size n if it belongs to P_n .

2.3. Binary classification

Binary classification is a task in machine learning when a model is asked to predict whether an object g belongs to a “positive” or a “negative” class given the object’s description (given by a subset of attributes). The model is obtained based on the provided training context (dataset) $K = (G, M, I)$ with predefined positive $G_+ \subset G$ and negative $G_- = G \setminus G_+$ objects. The first step to constructing a good binary classifier is to find a model operating the set of attributes M that efficiently separates positive objects from G_+ and negative objects from G_- . After that, the model can be applied to a test context $K_{test} = (G_{test}, M, I_{test})$ to predict its unknown positive and negative objects.

This paper studies binary classifiers of the form “if premise $p \in \mathbb{P}$ is true then object g is predicted positive, otherwise object g is predicted negative”.

The prediction quality of a premise $p \in \mathbb{P}$ on a training dataset is measured by comparing the *given* sets of positive G_+ and negative objects G_- with the sets of positive G_{p+} and negative objects G_{p-} predicted by a premise p . Note that the set G_{p+} is exactly the extent of the premise p : $G_{p+} = p'$.

$$\begin{aligned}
TP_p &= G_+ \cap G_{p+} = G_+ \cap p' & FP_p &= G_- \cap G_{p+} = G_- \cap p' \\
FN_p &= G_+ \cap G_{p-} = G_+ \setminus p' & TN_p &= G_- \cap G_{p-} = G_- \setminus p'
\end{aligned} \tag{5}$$

For the sake of brevity, let us use lowercase letters to denote the cardinalities of so-called true positives TP_p , false positives FP_p , false negatives FN_p , and true negatives TN_p . Likewise, we use y_+ , y_- to denote the cardinalities of the set of positive objects G_+ and the set of negative objects G_- respectively:

$$\begin{aligned}
tp_p &= |TP_p|, & fp_p &= |FP_p|, & fn_p &= |FN_p|, & tn_p &= |TN_p|, \\
y_+ &= |G_+|, & y_- &= |G_-|
\end{aligned} \tag{6}$$

One of the most widely used quality scores for binary classifications are precision ($prec$), recall (rec), and their harmonic mean called F1 score ($F1$). Their definitions are as follows:

$$prec(p) = \frac{tp_p}{|p'|}, \quad rec(p) = \frac{tp_p}{y_+}, \quad F1(p) = 2 \frac{prec(p) * rec(p)}{prec(p) + rec(p)} \tag{7}$$

This paper focuses on finding the premise p^* of size not bigger than 3 having the maximal F1 score:

$$p^* = \arg \max_{p \in \bigcup_{i=1}^3 P_i} F1(p) \tag{8}$$

The presented techniques can be easily adjusted for other quality measures.

3. Minimizing the number of comparisons

3.1. F1 score optimization

The definition of F1 score as a harmonic mean of precision and recall makes the possible optimization strategies obscure. This subsection simplifies the task of maximizing the F1 score through maximizing the number of true positives and true negatives.

Proposition 1. *F1 score $F1(p)$ is comonotonic to Jaccard score $J(p)$ (denoted by \sim), where the latter represents the Jaccard similarity coefficient between the set of positive objects G_+ and the set of objects predicted positive $G_{p+} = p'$:*

$$F1(p) \sim J(p) = \frac{|G_+ \cap p'|}{|G_+ \cup p'|} \tag{9}$$

Proof. Let us describe the Jaccard score in terms of true positives tp_p and true negatives tn_p :

$$J(p) = \frac{|G_+ \cap p'|}{|G_+ \cup p'|} = \frac{tp_p}{|G| - tn_p} \tag{10}$$

Now we should also express F1 score in terms of true positives tp_p and true negatives tn_p :

$$F1(p) = 2 \frac{prec(p) * rec(p)}{prec(p) + rec(p)} = \frac{2tp_p}{y_+ + |p'|} = \frac{2tp_p}{(|G| - fp_p - tn_p) + (tp_p + fp_p)}$$

$$= \frac{2tp_p}{|G| + tp_p - tn_p} = \frac{2}{1 + \frac{|G| - tn_p}{tp_p}} = \frac{2}{1 + \frac{1}{J(p)}} \quad (11)$$

Therefore we obtain the relation:

$$F1(p) \sim J(p)$$

□

Since F1 score $F1(p)$ is monotonic with respect to the Jaccard score $J(p)$ then the F1 score optimization problem can be viewed as the problem of optimizing the fraction $tp_p/(|G| - tn_p)$, i.e. maximizing the number of true positives and true negatives:

$$\arg \max_{p \in \mathbb{P}} F1(p) = \arg \max_{p \in \mathbb{P}} J(p) = \arg \max_{p \in \mathbb{P}} \frac{tp_p}{|G| - tn_p} \quad (12)$$

3.2. Logical operations effect on Jaccard score

This subsection discusses how conjunction and disjunction operations affect the Jaccard score. That is, given the Jaccard score of two premises $p, q \in \mathbb{P}$ can we expect that premises $p \wedge q, p \vee q$ would have higher or lower Jaccard score values?

Firstly, let us express the true positives and true negatives of premises constructed by conjunction and disjunction with the true positives and true negatives of the original premises:

$$\begin{aligned} TP_{p \wedge q} &= G_+ \cap (p' \cap q') = TP_p \cap TP_q & TP_{p \vee q} &= G_+ \cap (p' \cup q') = TP_p \cup TP_q \\ TN_{p \wedge q} &= G_- \setminus (p' \cap q') = TN_p \cup TN_q & TN_{p \vee q} &= G_- \setminus (p' \cup q') = TN_p \cap TN_q \end{aligned} \quad (13)$$

Therefore, conjunction operation shrinks the set of true positives while expanding the set of true negatives. Opposite to it, disjunction operation expands the set of true positives while shrinking the set of true negatives.

Now we derive the equations for the infimum and the supremum for the cardinalities of true positives and true negatives. To do so we incorporate two notions from Equation 13. Firstly, we use set cardinality restrictions for intersection and union operations. Secondly, since the number of true positives tp is limited by the number of positive objects y_+ , then for two premises $p, q \in \mathbb{P}$ if the sum of tp_p, tp_q exceeds y_+ , then the corresponding true positives should have at least $tp_p + tp_q - y_+$ objects in common (analogous conclusions can be provided for the number of true negatives tn and the number of negative objects y_-).

$$\begin{aligned} \max(tp_p + tp_q - y_+, 0) &\leq tp_{p \wedge q} \leq \min(tp_p, tp_q) \\ \max(tn_p, tn_q) &\leq tn_{p \wedge q} \leq \min(tn_p + tn_q, y_-) \\ \max(tp_p, tp_q) &\leq tp_{p \vee q} \leq \min(tp_p + tp_q, y_+) \\ \max(tn_p + tn_q - y_-, 0) &\leq tn_{p \vee q} \leq \min(tn_p, tn_q) \end{aligned} \quad (14)$$

Thus, the Jaccard score bounds are:

$$\frac{\max(tp_p + tp_q - y_+, 0)}{|G| - \max(tn_p, tn_q)} \leq J(p \wedge q) \leq \frac{\min(tp_p, tp_q)}{|G| - \min(tn_p + tn_q, y_-)} \quad (15)$$

$$\frac{\max(tp_p, tp_q)}{|G| - \max(tn_p + tp_q - y_-, 0)} \leq J(p \vee q) \leq \frac{\min(tp_p + tp_q, y_+)}{|G| - \min(tn_p, tn_q)}$$

Figure 1 visualizes the bounds on TruePositive-TrueNegative plane for an abstract formal context with 40% of objects being positive. Maximizing the Jaccard score, shown by increasing grey colour gradient, requires us to reach the upper-right corner of the plane. However, conjunction and disjunction operations over premises $p, q \in \mathbb{P}$ can only “move” the premises to the upper-left ($p \vee q$) and lower-right ($p \wedge q$) corners.

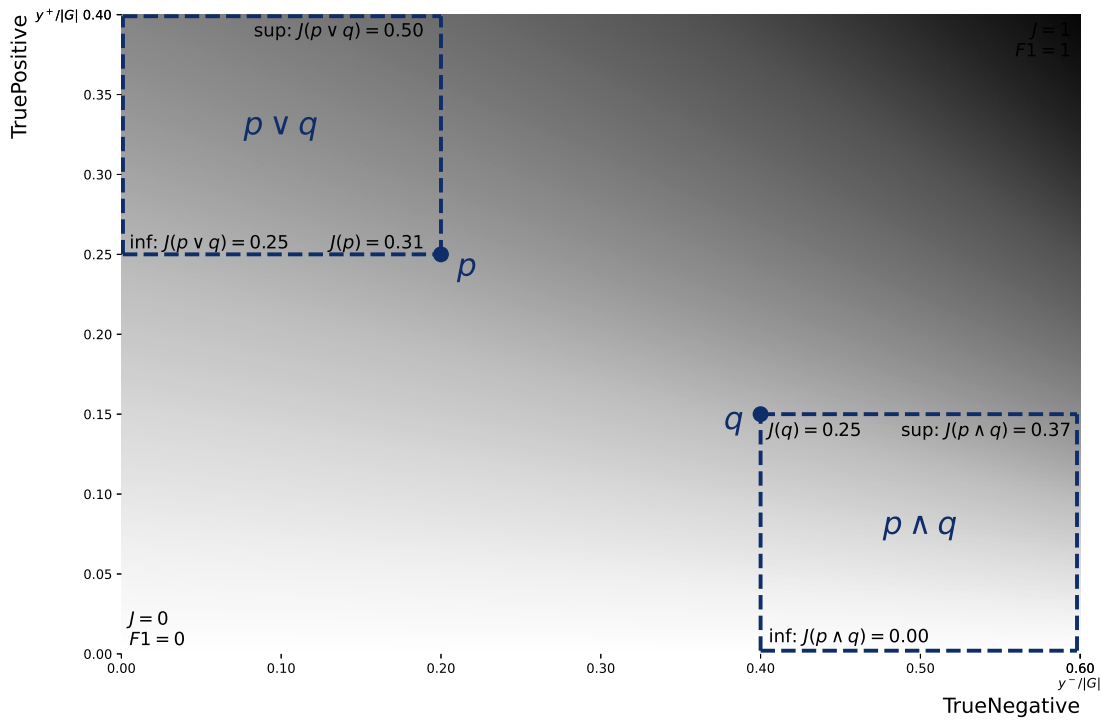


Figure 1: The Jaccard score bounds for conjunction and disjunction operations on abstract premises p, q of abstract formal context. The gray colour intensity represents the value of the Jaccard score for a specific point on the plane.

The bounds derived for conjunction and disjunction operations are vague: they do not say whether the newly formed premise would have the higher or lower Jaccard score. The more precise estimation is only possible via intersecting the extents of premises p, q . In this case, however, one can immediately compute the resulting Jaccard score itself, rather than operating the estimations.

Proposition 2. Given a threshold $\theta \in \mathbb{R}$ and a premise $p \in \mathbb{P}$, one can identify whether the Jaccard score of any premise $p \wedge q, p \vee q \in \mathbb{P}$ will not exceed the threshold θ using the following inequations:

$$\begin{aligned} tp_p \leq y_+ \theta &\implies J(p \wedge q) \leq \theta, \quad \forall q \in \mathbb{P} \\ tn_p \leq |G| - \frac{y_+}{\theta} &\implies J(p \vee q) \leq \theta, \quad \forall q \in \mathbb{P} \end{aligned} \quad (16)$$

Proof. Let us describe how the bounds in formulae 15 depend on true positives and true negatives of a premise $p \in \mathbb{P}$:

$$J(p \wedge q) \leq \frac{\min(tp_p, tp_q)}{|G| - \min(tn_p + tn_q, y_-)} \leq \frac{tp_p}{|G| - y_-} = \frac{tp_p}{y_+} \quad (17)$$

$$J(p \vee q) \leq \frac{\min(tp_p + tp_q, y_+)}{|G| - \min(tn_p, tn_q)} \leq \frac{y_+}{|G| - tn_p} \quad (18)$$

Now we compare the obtained fractions with a threshold θ :

$$\frac{tp_p}{y_+} \leq \theta \Leftrightarrow tp_p \leq y_+ \theta, \quad \frac{y_+}{|G| - tn_p} \leq \theta \Leftrightarrow tn_p \leq |G| - \frac{y_+}{\theta} \quad (19)$$

Thus we result in the initial implications:

$$tp_p \leq y_+ \theta \implies J(p \wedge q) \leq \theta, \quad \forall q \in \mathbb{P} \quad (20)$$

$$tn_p \leq |G| - \frac{y_+}{\theta} \implies J(p \vee q) \leq \theta, \quad \forall q \in \mathbb{P} \quad (21)$$

□

4. Engineering to maximize the speed of comparisons

Mathematical tricks help to minimize the number of comparing rules. However, the number of such rules is still high. This section concentrates on engineering tricks to make each comparison faster.

4.1. Extents and bitarrays

In this subsection we use two important facts about concept extents: (i) different premises may correspond to the same extent $\exists p, q \in \mathbb{P} : p \neq q, p' = q'$, (ii) any prediction quality measure of a premise p relies on the premise extent p' (see eq. 5).

Different premises $p, q \in \mathbb{P}, p \neq q$ may correspond to the same extent $p' = q' \subseteq G$ for many reasons. First, the premises can be logically equivalent: e.g. $(\bar{p} \wedge \bar{q})' = (\bar{p} \vee \bar{q})'$ by De Morgan laws. Second, if one premise is less general than another $p' \subset q'$ then their conjunction would correspond to the extent p' and their disjunction would correspond to the extent q' . Lastly, different premises may correspond to the same extents due to the specific characteristics of the formal context K : e.g. it may occur that an extent of the conjunction of two premises $p, q \in \mathbb{P}$ would be equal to the extent of the third premise $r \in \mathbb{P} \setminus \{p, q\} : (p \wedge q)' = r'$. Since the

computation of prediction quality measures relies on extents of premises, then many various premises corresponding to the same extent would have the same prediction quality (on the train context K).

Thus, we propose to search for the most optimal extent instead of the most optimal premise. In order to formalize this idea, let us define the sets extents E_n :

$$E_1 = \{p' \mid \forall p \in P_1\} = \bigcup_{m \in M} \{m', (\bar{m})'\} \quad (22)$$

$$E_n \in \mathbb{N} = \bigcup_{i=1}^{\lfloor n/2 \rfloor} \{a \wedge b, a \vee b \mid a \in E_i, b \in E_{n-i}\} \setminus \bigcup_{i=1}^{n-1} E_i$$

The proposed definition of the sets of extents ensures that any extent $e \in E_n$ is generated by a premise of size at least n :

$$\forall n, i \in \mathbb{N}, e \in E_n, p \in P_i : e = p' \implies n \leq i \quad (23)$$

Thus the optimization problem becomes as the following:

$$e^* = \arg \max_{e \in \bigcup_{n=1}^3 E_n} F1(e) \quad (24)$$

where F1 score function is slightly modified to take an extent as its parameter and not the premise.

The last but not the least, an extent e , being a subset of objects G , can be represented and stored in a computer as a bit mask (a tuple of bits) of length $|G|$ where each bit represents whether the corresponding object is in the extent or not. Conjunction and disjunction operations become operations on bit masks, that are the most efficient operations performed of modern binary coded computers. So the use of extents instead of premises not only reduces the number of comparisons, it also highly accelerates each of the comparisons.

4.2. Array operations

This subsection describes the trick, that is well-known among data science practitioners, however we should cover it for the full disclosure.

Inequalities, presented in Proposition 2, allow us to skip the processing of many conjunctions $p \wedge q$ and disjunctions $p \vee q$ based on the characteristics of the initial premises $p, q \in \mathbb{P}$ and a prediction quality threshold θ . However, these characteristics are still to be computed. And to compute these numerical characteristics the most efficiently we use Numpy [13] package for Python. The package is specifically designed to work with large volume of numerical data through the use of C++ code.

5. The proposed algorithm

Here is a pseudo-code of the algorithm for finding the best k premises of size no bigger than 3:

- Step 1. Find all extents of size 1 that will not lose quality after conjunction, disjunction:

1. Compute all extents E_1 ;
 2. Find the quality threshold θ as the minimal quality of the k best extents from E_1 ;
 3. Filter out extents from E_1 that satisfy both inequalities presented in Prop. 2;
- Step 2. Find all extents of size 2 that will not lose quality after conjunction, disjunction:
 1. Compute all extents E_2 based on filtered set E_1 while keeping the information about a pair of extents $a, b \in E_1$ and an operation (\wedge, \vee) used for constructing each extent $e \in E_2$;
 2. Find the quality threshold θ as the minimal quality of the k best extents from $E_1 \cup E_2$
 3. Filter out extents from E_1, E_2 that satisfy both inequalities presented in Prop. 2
 - Step 3. Find only the best extents of size 3:
For each pair of extent e_1, e_2 from filtered E_1, E_2
 1. If any of the extents e_1, e_2 satisfy both inequalities in Prop. 2 then proceed to the next pair; otherwise:
 2. Compute and measure the prediction quality of the conjunction $e_1 \cap e_2$;
 3. Compute and measure the prediction quality of the disjunction $e_1 \cup e_2$;
 4. Update θ if needed;
 - Step 4. Reconstruct the premises corresponding to the best k extents using the kept information about extents and operations.

The time complexity of this algorithm is $O(|M|^3)$ where M is a set of attributes in a formal context K . From the asymptotic point of view, this is the same time complexity as that of the brute force algorithm to test all premises of size not bigger than three. However, the use of extents, as well as reducing the number of combinations, allows us to minimize the practical processing time of the algorithm.

6. Experiments

This section applies the proposed algorithm in practice. First, we study the statistics of the number of comparisons the algorithm has to make. Second, we roughly compare the prediction quality of short models with that of the black box model to show that there are cases where the former performs as efficient as the latter.

The algorithm is run on a real-world Myocard dataset [14] from UCI repository. The dataset contains 1700 objects and 124 attributes. The task behind Myocard dataset is to predict whether a hospital patient will have or have not a chronic heart failure based on its data. We were not successful to run the algorithm in fast time (i.e. hours and days) on bigger datasets due to the combinatorial explosion. However, we consider Myocard dataset being big enough to test the algorithm.

Table 1
Statistics for algorithm iterations

premise size	1	2	3
# premises	1752	3.07e+06	1.08e+10
# ext. combinations	1644	2.53e+06	1.49e+09
# ext. combs. to test	1644	2.53e+06	1.17e+09
# new extents	1644	9.44e+05	9.55e+08
# extents to keep	1644	9.44e+05	259
computation time	7.08 ms	6.55 s	1.06 h

6.1. Number of comparisons

Table 1 shows that the use of bounds from Prop. 2 does not filter out many extent combinations. For example, for the premise size of 3, the bounds filter only 21.5% of extents combinations (from 1.49e+09 to 1.17e+09). Although such percentage is better than nothing, it still requires to test 1.17 billions extent combinations. However, the use of bit arrays allows us to test 1.17 billions of extent combinations in only 1 hour on a laptop with 8 GB of RAM.

Legend for Table 1:

- # premises: number of premises of a given size; It is the combinatorically computed maximal amount of iterations of the algorithm;
- # ext. combinations: number of extent combinations resulting in a premise of a given size;
- # ext. combs. to test: number of extent combinations that can result in a good prediction quality (filtered by Prop. 2);
- # new extents: the number of newly generated extents resulted from testing extent combinations;
- # extents to keep: the number of extents to keep in memory. For size 1 and 2 we keep all the extents, for size 3 we keep only the extents with high prediction quality;
- computation time: the time it took to process all combinations for a given premise size.

6.2. Prediction quality of short rules

The simplicity of short rule models allows us to fully describe some of the obtained models in the paper. The following list provides the best short models (w.r.t. F1 score defined in a standard way for a binary classification method) obtained on Myocard dataset:

- Premise size 1: F1 score = 0.401426
Premise: There is data about the use of painkillers in intensive care unit in the third day of the hospital period
- Premise size 2: F1 score = 0.448819
Premise: (a person had a chronic heart failure) OR (has diabetes mellitus in the anamnesis)
- Premise size 3: F1 score = 0.473786
Premise: (has data on use of opioid drugs in the intensive care unit in the third day of the hospital period) AND ((Had Chronic heart failure) OR (Age \geq 66))

- XGBoost model: F1 score = 0.464000
The model contains 100 decision trees of max depth 6
- CatBoost model: F1 score = 0.434783
The model contains 1000 decision trees of depth 6

We can also show all the obtained short rule models on TruePositive-TrueNegative space.

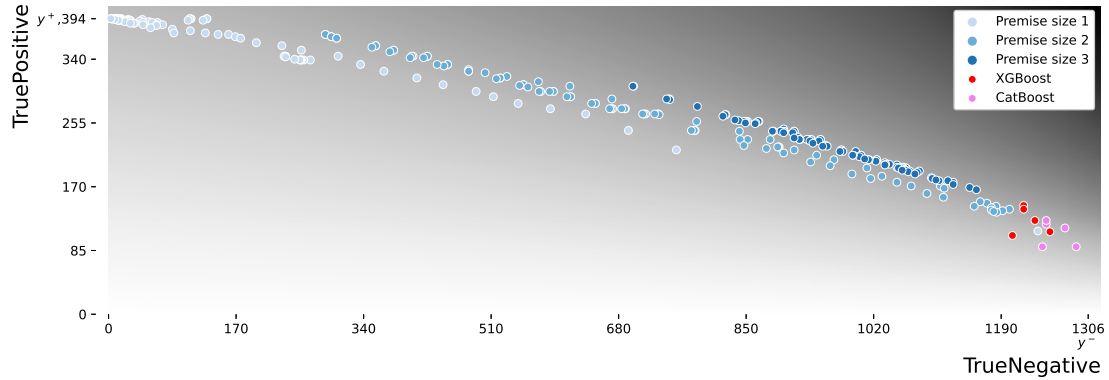


Figure 2: Short models prediction quality on TruePositive-TrueNegative space. The prediction quality of the default XGBoost and CatBoost models are presented as a reference.

Figure 2 shows the prediction quality of the obtained short models on TruePositive-TrueNegative scale. It should be noted that the points corresponding to complex black box gradient boosting models are not far away from the ones of short models. Thus, it is not reasonable to use complex black boxes on Myocard dataset, since simple short models offer the same prediction quality.

7. Conclusion

In this paper we have presented some preliminary results on finding the most optimal rule with antecedent consisting of no more than three binary attributes. We described the F1 score optimization task in terms of true positive and true negative predictions. We computed upper and lower bounds on Jaccard coefficients for premises obtained with conjunction and disjunction operations. We also covered FCA-inspired technique of iterating over extents of premises in order to minimize the computation runtime.

In the following studies we plan to develop sharper lower and upper bounds on Jaccard score for premises constructed with conjunction and disjunction operations. We also plan to discuss other logical operations that will increase the prediction quality of rules keeping the number of used attributes the same.

Acknowledgments

The article was prepared within the framework of the Basic Research Program at HSE University, RF.

References

- [1] J. Friedman, Greedy function approximation: A gradient boosting machine, *Annals of Statistics* 29 (2001) 1189–1232. doi:10.2307/2699986.
- [2] L. Breiman, Random forests, *Machine Learning* 45 (2001) 5–32. doi:10.1023/A:1010933404324.
- [3] C. Molnar, *Interpretable Machine Learning*, 2019. <https://christophm.github.io/interpretable-ml-book/>.
- [4] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al., Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai, *Information fusion* 58 (2020) 82–115.
- [5] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, *Nature Machine Intelligence* 1 (2019) 206–215. doi:10.1038/s42256-019-0048-x.
- [6] O. Pianykh, E. Dudyrev, A. Sharp, G. Gusev, S. O. Kuznetsov, I. Semenov, *Human knowledge models: Learning applied knowledge from the data*, 2022. Unpublished.
- [7] B. Ganter, R. Wille, *Formal Concept Analysis*, Springer, Berlin, 1999.
- [8] M. M. Bongard, *The recognition problem*, Technical Report, FOREIGN TECHNOLOGY DIV WRIGHT-PATTERSON AFB OHIO, 1968.
- [9] R. S. Michalski, *Discovering classification rules using variable-valued logic system vl1* (1973).
- [10] J. H. Friedman, B. E. Popescu, Predictive learning via rule ensembles, *The annals of applied statistics* 2 (2008) 916–954.
- [11] N. Cowan, The magical number 4 in short-term memory: A reconsideration of mental storage capacity, *Behavioral and brain sciences* 24 (2001) 87–114.
- [12] G. S. Halford, R. Baker, J. E. McCredden, J. D. Bain, How many variables can humans process?, *Psychological science* 16 (2005) 70–76.
- [13] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, et al., Array programming with NumPy, *Nature* 585 (2020) 357–362. doi:10.1038/s41586-020-2649-2.
- [14] S. E. Golovenkin, J. Bac, A. Chervov, E. M. Mirkes, Y. V. Orlova, E. Barillot, A. N. Gorbun, A. Zinovyev, Trajectories, bifurcations, and pseudo-time in large clinical datasets: Applications to myocardial infarction and diabetes data, *GigaScience* 9 (2020) g128.