entropy

MDPI

Article

Decision Rules Construction: Algorithm Based on EAV Model

Krzysztof Żabiński † and **Beata Zielosko** *,†

Institute of Computer Science, Faculty of Science and Technology, University of Silesia in Katowice, Będzińska 39, 41-200 Sosnowiec, Poland; krzysztof.kamil.zabinski@gmail.com

* Correspondence: beata.zielosko@us.edu.pl

† These authors contributed equally to this work.

Abstract: In the paper, an approach for decision rules construction is proposed. It is studied from the point of view of the supervised machine learning task, i.e., classification, and from the point of view of knowledge representation. Generated rules provide comparable classification results to the dynamic programming approach for optimization of decision rules relative to length or support. However, the proposed algorithm is based on transformation of decision table into entity–attribute–value (EAV) format. Additionally, standard deviation function for computation of averages' values of attributes in particular decision classes was introduced. It allows to select from the whole set of attributes only these which provide the highest degree of information about the decision. Construction of decision rules is performed based on idea of partitioning of a decision table into corresponding subtables. In opposite to dynamic programming approach, not all attributes need to be taken into account but only these with the highest values of standard deviation per decision classes. Consequently, the proposed solution is more time efficient because of lower computational complexity. In the framework of experimental results, support and length of decision rules were computed and compared with the values of optimal rules. The classification error for data sets from UCI Machine Learning Repository was also obtained and compared with the ones for dynamic programming approach. Performed experiments show that constructed rules are not far from the optimal ones and classification results are comparable to these obtained in the framework of the dynamic programming extension.

Keywords: decision rules; classification; length; support; dynamic programming approach; entity–attribute–value model

Citation: Żabiński, K.; Zielosko, B. Decision Rules Construction: Algorithm Based on EAV Model. *Entropy* **2021**, *23*, 14. https://doi.org/10.3390/e23010014

Received: 3 November 2020
Accepted: 18 December 2020
Published: 24 December 2020

1. Introduction

Currently, data amounts grow constantly and uncontrollably, practically in every domain of life. It results in the demand for efficient and fast methods of their analysis. Generally speaking, raw data occupy more and more disk space, but without converting them to any kind of useful knowledge, they are just redundant. The question is then how to derive knowledge from raw data. Data mining as a scientific discipline tries to answer it. This domain has been developed for many years already; as a result, there are plenty of already existing methods for multiple applications [1–4]. Nevertheless, with the growth of data amounts, the existing methods also need to be further developed and new approaches need to be proposed.

The process of knowledge extraction from data sets is called learning. Learning can be basically divided into two subcategories: supervised and unsupervised learning. The main difference is that with supervised learning, there is a supervision of the learning process (which in practice means that data sets are labeled and labels are assigned to each of the item from the set under consideration). As for unsupervised learning, the labeling does not exist, and the task is basically to find any relations between data items. There are also so-called semisupervised learning methods that combine both labeled and unlabeled records being considered [5].

In this article, we study one of the most popular supervised learning methods, i.e., decision rules construction. Such rules are known and popular as a form of knowledge representation used often in the framework of supervised machine learning.

In the general case, decision rules studied in this paper are expressed as follows [6]:

IF conditions THEN conclusion.

The "IF" part of a rule is known as the rule antecedent (premise part) and it contains one or more conditions (pairs attribute = value) connected by conjunction. The "THEN" part is the rule consequent which present a class label. Based on conditions included in the premise part, the rule assigns class labels to objects. An example would be a rule which predicts when a student enjoy skating:

IF $(Temperature = low)$ AND $(Ice\_Skating\_Rink = yes)$ THEN $(Enjoy\_Sport = yes)$.

Decision rules are popular because of their form which is simple and easy accessible from the point of view of understanding and interpretation of knowledge represented by them. One of the most popular evaluation measure is length, which corresponds to the number of descriptors (pairs $attribute = value$) on the left-hand side of the rule. Another popular measure is support, which represents the number of objects from the learning set that match the rule. There are also many other indicators for evaluation of decision rules [7–9]; however, in this paper, these two are considered.

Construction of short rules with good support is an important task considered in this work. In particular, the choice of short rules is connected with the minimum description length principle [10]: "the best hypothesis for a given set of data is the one that leads to the largest compression of data". Support allows to discover major patterns in the data. These two measures are interesting from the point of view of knowledge representation and classification.

Unfortunately, the problems of minimization of length and maximization of support of decision rules are NP-hard [11–13]. The most part of approaches for decision rules construction, with the exception of brute force, Boolean reasoning, and dynamic programming, cannot guarantee the construction of optimal rules, i.e., rules with minimum length or maximum support. The main drawback of such approaches is limitation of the size of data if the user would like to obtain a solution in some acceptable time. For this reason, authors propose some heuristic, an algorithm for decision rules construction. Generating rules should be enough good from the point of view of knowledge representation and classification.

The paper consists of five main sections. The Introduction, which contains the authors' contribution, is followed by the Related Works section. Then, in the Materials and Methods section, the proposed approach and main notions, including the entity–attribute–value model, standard deviation function, algorithm for construction of rules and computational complexity analysis, are described. In Section 4, the results of experiments connected mainly with evaluation of constructed decision rules using length, support and classification error are presented. Finally, the conclusions are discussed in Section 5.

Contribution

In this paper, we propose an algorithm for decision rules construction that belongs to the group of heuristics as it constitutes approximate rules, but which grows from the root of dynamic programming extensions-based approaches. There exist some similarity between these methods connected with partitioning decision table into subtables; however, the main difference is based on selection of attributes and construction of rules which are close to optimal ones. For this reason, the experimental results obtained for the proposed approach were compared with the results known for dynamic programming extensions from the point of view of knowledge discovery and knowledge representation. It was shown that the proposed approach allows us to reduce the number of attributes under

consideration even to 60% from the whole set of attributes and obtain classification results comparable to the ones obtained by the dynamic programming extension.

The idea of dynamic programming approach for decision rules optimization is based on partitioning of a decision table into subtables which are created for each value of each conditional attribute. In this way, a directed acyclic graph is obtained which nodes correspond to subtables and edges are labeled by values of attributes. Based on the graph, the so-called irredundant decision rules are described, i.e, rules with minimal length or maximal support. However, if the number of attributes and their values is large, the size of the graph (the number of rows and edges) is huge. Therefore, obtaining an exact solution within a reasonable time is not always achievable.

In the proposed approach, the stage of construction of a graph based on which decision rules are described is omitted. Decision table is partitioned into subtables, but only for the values of the selected attributes. Moreover, to accelerate calculations, the decision table is transformed into the so-called entity–attribute–value model [14]. Selection of attributes and their evaluation is based on the analysis of the spread of their values' standard deviation for decision classes. If the value of standard deviation is high, there is a high possibility that the attribute can distinguish objects with different decisions. Based on values of selected attributes, corresponding subtables from the input table are created. The process of partitioning of corresponding subtables is finished when all rows in a given subtable have the same class label or all values of selected attributes were considered. Then, decision rules are created basing on corresponding values of selected attributes.

In some authors' previous work [15–17], a modification of the dynamic programming approach for decision rules optimization was proposed; however, the idea was connected with decreasing the size of the graph. Subtables of an input decision table were constructed for one attribute with the minimum number of values, and for the rest of the attributes, the most frequent value of each attribute (value of an attribute attached to the maximum number of rows) was selected. In the presented approach, the idea of selection of attributes is different and construction of the graph is omitted.

2. Related Works

There exists a variety of approaches for construction of decision rules. The used approaches depend on the aim for which the rules are constructed. The two main perspectives are knowledge representation and knowledge discovery [18]. Since the aims are different, algorithms for construction of rules with their many modifications are different and quality measures for evaluating of such rules are also different.

Basically, approaches for construction of decision rules can be divided into two categories:

- allowing to obtain exact rules on the data set under consideration,
- generating approximate rules, not perfectly suiting the learning set, but aiming to create rules applicable for general use (these methods will be called heuristics further in this work).

Among the first group, there are algorithms which allow to obtain all decision rules based on exhaustive strategy, there are: the brute-force approach which is applicable to decision tables with a relatively small number of attributes, Boolean reasoning [19], and the dynamic programming approach [20,21] proposed in the framework of rough sets theory [22].

Among the second group of methods, there are popular algorithms based on a sequential covering procedure [23–25]. In this case, decision rules are created and added to the set of rules iteratively until all examples from a training set will be covered. Among them, we can distinguish the general-to-specific search methods, e.g., CN2 [26] and PRISM [27] algorithms. In the framework of directional general-to-specific search approach, the AQ family of algorithms can be indicated [28]. The RIPPER algorithm [29] is an example of search methods with pruning. LEM2 belongs to methods based on reduct from rough sets theory [30].

There are also plenty of heuristics which are useful in situations where it is difficult to find an exact solution in some acceptable time. However, such algorithms often pro-

duce a suboptimal solution since they cannot avoid local optima. Popular approximate approaches include greedy algorithms [31–33] and the whole group of biologically inspired methods, among which the following are worth mentioning: genetic algorithms [34–36], ant colony optimization algorithms [37,38], swarm-based approaches [39] and many others as described in [40,41].

The task of constructing a decision rule is similar to the task of finding the features that define entities of some category. Attributes can be considered as functions which mapping a set of objects into a set of attributes' values. The premise part of a decision rule contains one or more conditions in a form $attribute = value$. The consequent part of a rule represents a category. For both tasks, there may be a problem how to choose the features that the best describe a given category (concept).

Constructed rules can be considered as patterns that cover many situations and match as many examples as possible, so they should be short according to number of conditions to allows some generalization. However, such rules should also take into account unusual situations, ensuring the correct classification of such objects. Often, the same category is described by more than one rule. Therefore, the set of rules learned from data should not contain contradictory or redundant rules. It should be small such that all rules together cover all examples from learning set and describe a given category in a fairly comprehensive way. As it was mentioned above, there exist plenty of algorithms which use different methods and measures during process of rules construction and choosing attributes that constitute premise part of rules.

3. Materials and Methods

In this section, notions corresponding to the proposed approach, the entity–attribute–value model, standard deviation as a distinguishability measure, algorithm for construction of rules and analysis of its computational complexity are presented.

3.1. Decision Rules Construction Approach

The three main stages of the proposed decision rules construction approach are the following:

1. Transformation decision table T into EAVD model,
2. Calculation of standard deviation based on averages' attributes values per decision class,
3. Construction of decision rules taking into account selected attributes.

They are discussed in the next sections.

3.1.1. Main Notions

One of the main structure for data representation is the decision table [22]. It is defined as

$$T = (U, A \cup \{d\}),$$

where U is a nonempty, finite set of objects (rows), $A = \{f_1, \ldots, f_n\}$ is nonempty, finite set of attributes, $f : U \to V_f$ is a function, for any $f \in A$, V_f is the set of values of an attribute f. Elements of the set A are called conditional attributes and $d \notin A$ is a distinguished attribute, called a decision attribute. The decision d determines a partition $\{Class_1, \ldots, Class_{|V_d|}\}$ of the universe U, where $Class_i = \{x \in U : d(x) = d_i\}$ is called the ith decision class of T, for $1 \leq i \leq |V_d|$. A minimum decision value that is attached to the maximum number of rows in T is called the most common decision for T.

The table T is called degenerate if T is empty or all rows of T are labeled with the same decision's value.

A table obtained from T by the removal of some rows is called a subtable of the table T. Let T be nonempty, $f_{i_1}, \ldots, f_{i_m} \in \{f_1, \ldots, f_n\}$ and $a_1, \ldots, a_m$ be values of attributes.

The subtable of the table T that contains only rows that have values $a_1, \ldots, a_m$ at the intersection with columns $f_{i_1}, \ldots, f_{i_m}$ is denoted by

$$T' = T(f_{i_1}, a_1) \ldots (f_{i_m}, a_m).$$

Such nonempty subtable (including the table T) is called separable subtable of T.

In the paper, decision rule is presented in the following form:

$$(f_{i_1} = a_1) \wedge \ldots \wedge (f_{i_m} = a_m) \rightarrow d = v$$

where v is the most common decision for T'.

The length of the decision rule is the number of conditions (pairs $attribute = value$) from the left-hand side of rule.

The support of the decision rule is the number of objects from T matching the conditional part of the rule and its decision.

3.1.2. Entity–Attribute–Value Model

The proposed approach for construction of decision rules is based on representing decision table in the entity–attribute–value form (abbreviated as EAV). The decision table formed in a way that each object contains a set of conditional attributes' values and decision (each object occupies one row in the decision table) is converted in a way that each value of attribute constitutes a separate row in the derived EAV table.

EAV form is very convenient for processing large amounts of data. It is mainly due to the fact, that such a form allows to utilize RDBMS (ang. Relational Database Management System) mechanisms directly. The idea to utilize SQL and RDBMS for data mining tasks has known advantages. As SQL is designed to facilitate dealing with large data sets efficiently, it is a natural choice for machine learning related tasks. Additionally, current RDBMSes are well designed to store and retrieve data efficiently and fast. Combining this technological achievement with efficient algorithms can lead to satisfactory level of rule generating system. The idea of SQL-based approach for association rules generation has been introduced in [42] and extended to decision rules construction in [14].

Exemplary EAV table can be created in RDBMS as shown in the Listing 1 (PostgreSQL example).

Listing 1. EAV table.

```
CREATE TABLE eav
(
id serial primary key,
attribute character varying,
value character varying,
decision character varying,
row bigint
);
```

In order to have better control on the association of the decision to the attribute value, the EAV form can be extended to contain decision too [43]. Such a format of representation can be denoted as EAVD (entity–attribute–value_with_decision form).

Figure 1 present exemplary decision table transformed into EAVD model.

Exemplary decision table

f_1	f_2	f_3	d
1	1	1	1
0	1	0	2
1	1	0	2
0	0	1	3
1	0	0	3

Exemplary decision table in EAVD model

attribute	value	decision	row
f1	1	1	1
f2	1	1	1
f3	1	1	1
f1	0	2	2
f2	1	2	2
f3	0	2	2
f1	1	2	3
f2	1	2	3
f3	0	2	3
f1	0	3	4
f2	0	3	4
f3	1	3	4
f1	1	3	5
f2	0	3	5
f3	0	3	5

Figure 1. Transformation of decision table into EAVD model.

3.1.3. Standard Deviation as a Distinguishability Measure

Standard deviation (abbreviated as STD) has been used in this paper as a distinguishability measure. It is based on Bayesian data analysis where attributes and their values are evaluated subject to possible decision classes [44]. We calculated standard deviation of the average values of each decision table attributes grouped per decision values. As attributes in real-life applications often are non-numerical, authors take into consideration numerical equivalents of such attributes. They are simply ordinal numbers according to attributes' values appearance in the data set under consideration. The higher the standard deviation is, the greater the diversity among averages values of attributes in particular classes. Such an observation can lead to the conclusion that the higher the STD is, the better the distinguishability between decision classes becomes. As a result, the attributes of high standard deviation should be prioritized when forming decision rules. This forms a ranking of attributes and allows to perform a feature selection step to the rule generation algorithm.

The exemplary SQL code to calculate the mentioned standard deviation is as shown in the Listing 2 (PostgreSQL example).

Listing 2. SQL command to calculate standard deviation.

```
SELECT attribute, STDDEV(average_value) AS quality FROM (
SELECT e.attribute, e.decision, AVG(v.id) AS average_value
FROM eav e JOIN values v ON e.value = v.value
GROUP BY attribute, decision
) attribute_average_values
GROUP BY attribute
ORDER BY quality DESC
```

Due to the fact that the proposed algorithm needs to work on any alphanumerical values, there was a need to introduce a dictionary table. Each attribute value needs to be converted into its numerical representative, whilst real values need to be transferred to the proposed dictionary table. The mentioned numerical representatives are subsequent RDBMS tables identifiers (in our approach). The exemplary dictionary table is as shown in the Listing 3.

Listing 3. Table containing real values of the given attributes.

```
CREATE TABLE values
(
id serial primary key,
value character varying
);
```

In order to better illustrate the distinguishability of attributes based on STD, an exemplary graph has been suggested (Figure 2), for attributes from decision table presented in Figure 1.

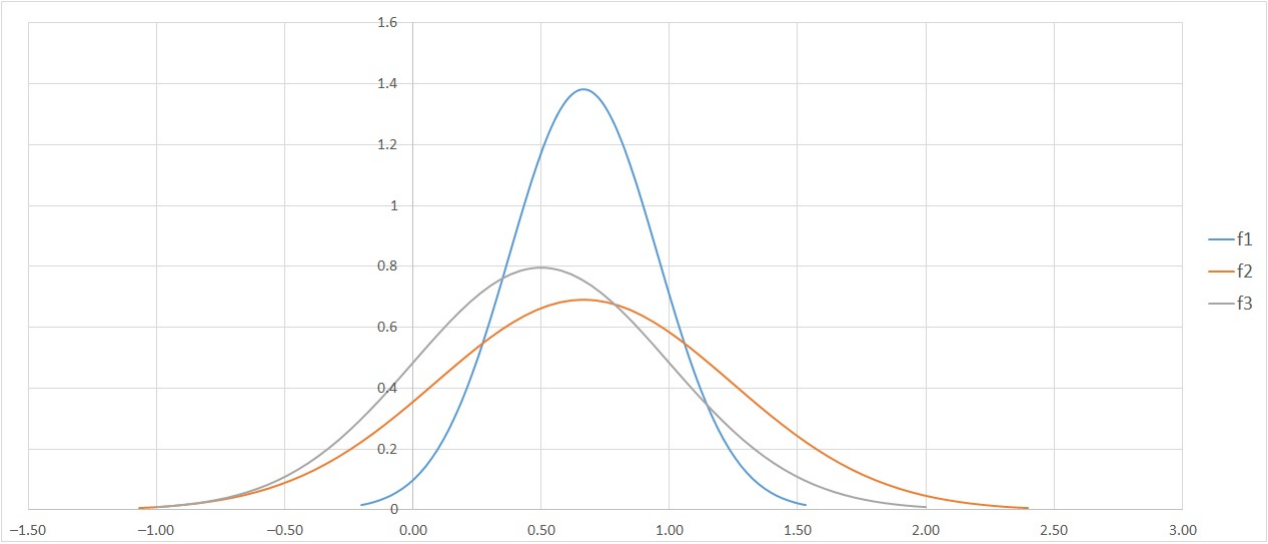

Figure 2. Normal distributions of attributes f1, f2 and f3.

It presents normal distributions of attributes among which every has a different STD value. The amplitude of the plots is not important whilst the width of each curve is a direct indicator of the distinguishability level. Basing on such a distribution graph, the conclusion can be make that the ranking of attributes is as follows: f2, f3, f1.

3.1.4. Construction of Decision Rules

The proposed algorithm can be expressed by means of the following pseudo-code (see Algorithm 1). The Algorithm 1 has been designed to deal with discrete attributes. Nevertheless, the is no assumption on numerous nature of attributes. As a result, symbolic attributes need to be converted to numerical representatives. An additional dictionary table has been introduced. It gathers symbolical attributes with their numerical representatives. Having numerical values of attributes obtained, the standard deviation of their average values per decision class can be calculated. These are standard deviations of the average values of each attribute numerical representation for each decision value. The higher the calculated standard deviation, the higher the distinguishability of decision classes basing on the considered attribute. It allows to generate rules basing on only the best attributes from the point of view of distinguishability. The percentage of best attributes that are used for generation need to be chosen empirically and strongly depends on the structure of the data set under consideration. Due to the fact that the attributes are ordered taking into account the distinguishability they offer with respect to the decision values, the generated rules will consist of small number of attributes (making them close to optimal taking into account their length). Creation a ranking of attributes' basing on STD values introduces a feature selection step. From the point of view of dynamic programming's algorithm graph construction, the proposed algorithm perform graph pre-pruning, as it omits creation of unnecessary paths that will never be utilized for rules generation.

Having chosen the attributes to be considered, the Algorithm 1 generates rules as described below. Firstly, the set of unique combinations of attributes with values per row in the input decision table needs to be determined. The combinations contain only of the best attributes chosen in the previous step. It is where the row information is also needed. The attributes need to appear in the subsequent combinations ordered descending by the values of previously calculated standard deviations. Nevertheless, the information on decision is not taken into consideration at this stage.

Then, for each attribute combination, the subsequent separable subtables of the input decision table get determined. It makes the algorithm similar to the dynamic programming approach which utilizes partitioning of the decision table into separable subtables. For each of the attribute combinations, the procedure starts by taking the first attribute with its value (it is visible now why the attributes in combinations need to be ordered decreasingly by the previously calculated standard deviations). For this attribute and its value, the separable subtable of input decision table gets determined. After that, it is verified if the subtable is degenerate. If it is in fact degenerate, a decision rule is constructed. It consists of the attribute with its value and the decision in the degenerate table. If the separable subtable is not degenerate, the subsequent attribute from the attribute combination gets chosen and then the subsequent separable subtable gets generated. If it is degenerate, the procedure stops and a decision rule gets generated. The procedure continues until either the subtable is degenerate or all the attributes of a given combination are processed. If all the attributes have been used and the subtable is still not degenerate, a decision rule gets generated basing on all these attributes with their values and the most common decision from the corresponding separable subtable.

Algorithm 1 Pseudo-code of algorithm generating decision rules for a decision table T.

Input: Input decision table T, number p of best attributes to be taken into consideration.
Output: Set of decision rules R

represent T as entity–attribute–value (EAV) form with separate decision;
represent each attribute's value in a discrete numerical form;
obtain attributes' standard deviation per decision class.
take p number of attributes of largest STD—in a descending order;
from T in EAV form select sets v of unique values (including decision) of attributes grouped per decision table's rows;
while there exist sets v_i in v not marked as processed **do**
  generate one-item v_i' set with initial value from v_i which corresponds to creation of separable subtable $T' = T(f_i, a_i)$;
  set v_i is not processed;
  while iterations number < $sizeof(v_i)$ OR separable subtable is not degenerate **do**
    extend v_i' by supplying it with the subsequent element from v_i which corresponds to next partition of T';
  end while
  generate decision rule basing on the values of attributes from v_i' (consequent is the most common decision for T' corresponding to v_i');
  supply the set R with the newly created rule;
  set v_i being processed.
end while

Where: T is a decision table; p is the ceiling of then number of percentage of the selected best attributes in the formed ranking; R is the set of generated rules; v is the unique set of values from the T in EAV form grouped per rows of input table T; v_i and v_i' are temporary subsets of v for the sake of rule generating iteration, based on the values included in the set v and its subsets separable subtables are created.

Example 1. *This example presents work of the Algorithm 1 for decision table shown in Figure 1.*

Percentage of best attributes which will be taken into consideration is 40%*, so ceiling of the number of attributes needs to be taken is 2. The attributes standard deviations per decision class are calculated using averages' values of attributes, for each attribute and decision value from EAVD representation, and they are the following: for* $f_1, STD = 0.29$*, for* $f_2, STD = 0.58$*, for* $f_3, STD = 0.5$*. It allows to create a ranking of attributes:*

- $f_2 \rightarrow 0.58$,
- $f_3 \rightarrow 0.5$,
- $f_1 \rightarrow 0.29$.

So the chosen best attributes are: f_2 *and* f_3*. Standard deviations are the highest and normal distributions are the widest (see Figure 2). Value sets* v *of the chosen* p *attributes, for each row, from the decision table are the following:*

- $f_2, f_3 = \{1, 1\}$.
  Separable subtable T(f2,1)

f_1	f_2	f_3	d
1	1	1	1
0	1	0	2
1	1	0	2

  is not degenerate (rows have different decisions), so subtable $T(f2, 1)$ *needs to be partitioned and subtable* $T(f2, 1)(f3, 1)$ *is obtained.*

f_1	f_2	f_3	d
1	1	1	1

  It is degenerate table, so the rule $f_2 = 1 \wedge f_3 = 1 \rightarrow d = 1$*, associated with the first row of exemplary decision table is derived.*
- $f_2, f_3 = \{1, 0\}$
  Separable subtable $T(f2, 1)$ *is not degenerate, so subtable* $T(f2, 1)(f3, 0)$ *is obtained.*

f_1	f_2	f_3	d
0	1	0	2
1	1	0	2

  It is degenerate table, so decision rule $f_2 = 1 \wedge f_3 = 0 \rightarrow d = 2$ *is derived. This rule is associated with the second and third row from exemplary decision table.*
- $f_2, f_3 = \{0, 1\}$
  Subtable $T(f_2, 0)$ *is degenerate.*

f_1	f_2	f_3	d
0	0	1	3
1	0	0	3

  Derived decision rule $f_2 = 0 \rightarrow d = 3$ *is associated with the fourth and fifth row of exemplary decision table.*
- $f_2, f_3 = \{0, 0\}$
  Subtable $T(f_2, 0)$ *is degenerate and decision rule* $f_2 = 0 \rightarrow d = 3$ *is associated with the fourth and fifth row of exemplary decision table.*

  The resulting set R of decision rules derived for rows from exemplary decision table is as follows:

- $1, f_2 = 1 \wedge f_3 = 1 \rightarrow d = 1$
- $2, f_2 = 1 \wedge f_3 = 0 \rightarrow d = 2$
- $3, f_2 = 1 \wedge f_3 = 0 \rightarrow d = 2$
- $4, f_2 = 0 \rightarrow d = 3$
- $5, f_2 = 0 \rightarrow d = 3$

  When duplicated rules are removed we obtain:

- $f_2 = 1 \wedge f_3 = 1 \rightarrow d = 1$
- $f_2 = 1 \wedge f_3 = 0 \rightarrow d = 2$
- $f_2 = 0 \rightarrow d = 3$

The state transition diagram (Figure 3) visually presents the Algorithm 1 for decision rule construction.

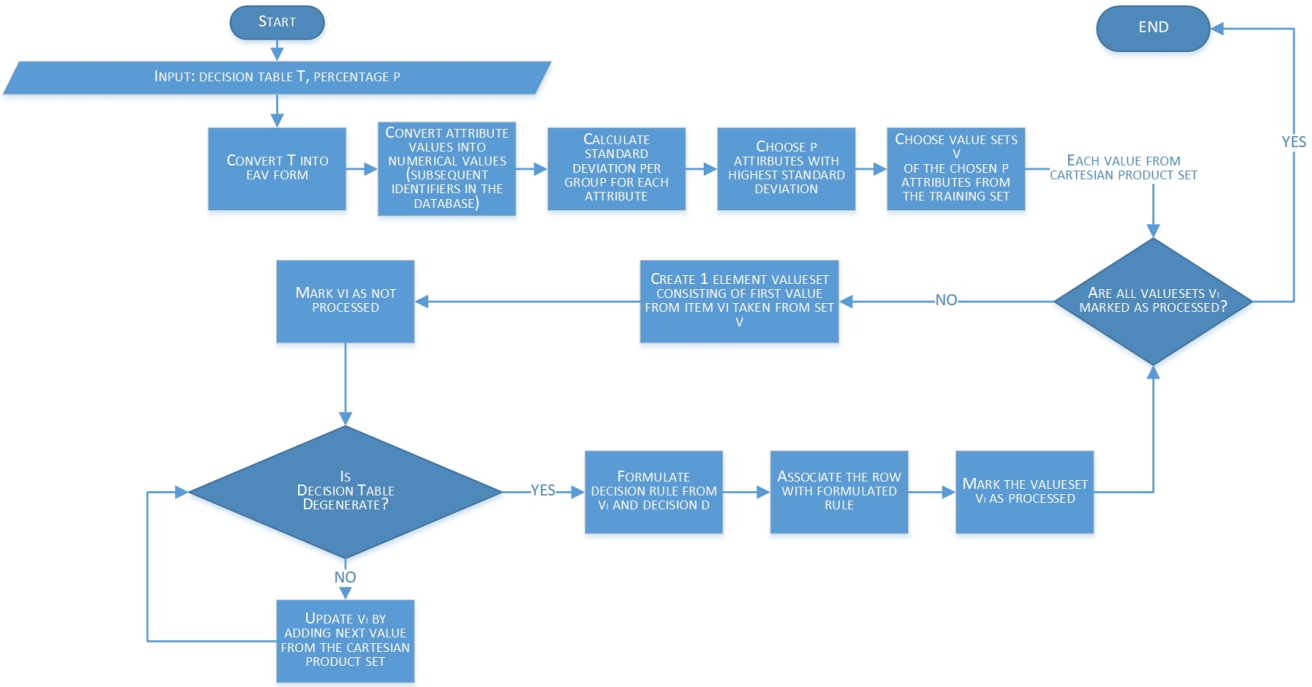

Figure 3. State transition diagram (STD) of the Algorithm 1.

Taking into account the state transition diagram, it is seen that the algorithm consists of a set of preprocessing steps before two main rule generating loops start to work. Two nested loops are a potential bottle neck from the computational complexity analysis point of view, but as it is explained in the next section, in an average case it is negligible. Moreover, thanks to the attribute ranking, in the most likely case scenario, algorithm can finish operation fast and generate good quality rules (which is detailed described in the Section 4).

3.1.5. Algorithm Computational Complexity Analysis

In order to calculate the computational complexity of the proposed algorithm, let us analyze each of the algorithm's steps. The steps have been rewritten from Figure 3 and gathered with their execution times in Table 1. The information is based on the assumption that N is the number of objects in the decision table T, whilst P is the number of input parameters, V is the number of valuesets v_i and V_i is the number of attributes' values for a given attribute i.

Summing up all execution times, we get the following time complexity of the algorithm (Equation (1)):

$$\begin{aligned} T(N) = {} & t_1 + t_2 + t_3 + t_4 + t_5 + V \cdot t_6 + V \cdot t_7 + V \cdot t_8 + V \cdot V_i \cdot t_9 + \\ & + V \cdot V_i \cdot t_{10} + V \cdot t_{11} + V \cdot t_{12} + V \cdot t_{13} + t_{14} \end{aligned} \tag{1}$$

which can be expressed in the form (Equation (2)):

$$\begin{aligned} T(N) = {} & V \cdot V_i \cdot (t_9 + t_{10}) + \\ & + V \cdot (t_6 + t_7 + t_8 + t_{10} + t_{11} + t_{12} + t_{13}) + \\ & + t_1 + t_2 + t_3 + t_4 + t_5 + t_{14}. \end{aligned} \tag{2}$$

Approximating the equation by introducing time constants, i.e.: $t_{9:10} = t_9 + t_{10}$, $t_{6:8-10:13} = t_6 + t_7 + t_8 + t_{10} + t_{11} + t_{12} + t_{13}$ and $t_{1:5-14} = t_1 + t_2 + t_3 + t_4 + t_5 + t_{14}$, Equation (2) can be simplified as follows (Equation (3)):

$$T(N) = V \cdot V_i \cdot t_{9:10} + V \cdot t_{6:8-10:13} + t_{1:5-14}. \tag{3}$$

Taking into account the properties mentioned earlier: $1 \leq V \leq N$ and $1 \leq V_i \leq N$, computational complexity of the algorithm can be determined:

- optimistic complexity:
$$T_O(N) = 1 \tag{4}$$
- average complexity—due to the fact that V_i is typically a small constant, sometimes even equal to 2 (for binary attributes) whilst V is typically close to N, the complexity can be expressed as follows:
$$T_A(N) = N \tag{5}$$
- pessimistic (worst) complexity:
$$T_W(N) = N^2. \tag{6}$$

Taking into account the computational complexity of the dynamic programming approach introduced in [21], it is seen that the proposed solution is much more time efficient. Whilst for the described algorithm, the computational complexity is linear in the average scenario, for the dynamic programming approach, it could be exponential in many cases.

Table 1. Algorithm 1 steps with their execution times.

Step	Operation	Execution Time
1	Convert T into eav form	t_1
2	Convert attribute values into numerical values	t_2
3	Calculate std per decision class for each attribute	t_3
4	Choose p attributes with highest std value	t_4
5	Choose sets v of the chosen p attributes from the training set	t_5
6	Are all valuesets v_i marked as processed	$V \cdot t_6$, where $1 \leq V \leq N$
7	Create 1 element valueset consisting of first value from item v_i taken from set v	$V \cdot t_7$, where $1 \leq V \leq N$
8	Mark v_i as not processed	$V \cdot t_8$, where $1 \leq V \leq N$
9	Does the corresponding subtable is degeneare?	$V \cdot V_i \cdot t_9$, where $1 \leq V \leq N$ and $1 \leq V_i \leq N$
10	Update v_i by adding next value from the cartesian product set	$V \cdot V_i \cdot t_{10}$, where $1 \leq V \leq N$ and $1 \leq V_i \leq N$
11	Formulate decision rule from v_i and the most common decision d	$V \cdot t_{11}$, where $1 \leq V \leq N$
12	Associate the row with the formulated rule	$V \cdot t_{12}$, where $1 \leq V \leq N$
13	Mark the valueset v_i as processed	$V \cdot t_{13}$, where $1 \leq V \leq N$
14	END	t_{14}

4. Results

Experiments were performed on decision tables from UCI Machine Learning Repository [45]. When for some of the decision tables, there were attributes taking unique value

for each row, such attributes were removed. When some of the decision tables contained missing values, each of these values was replaced with the most common value of the corresponding attribute. When, in some of the decision tables, there were equal values of conditional attributes but different decisions, then each group of identical rows was replaced with a single row from the group with the most common decision for this group.

The aim of performed experiments was to measure quality of decision rules constructed by proposed algorithm. Decision rules were evaluated from the point of view of:

- knowledge representation, i.e., length and support of obtained rules were calculated and compared with optimal decision rules obtained by dynamic programming approach,
- knowledge discovery, i.e., classification error was calculated and compared with classifiers obtained by dynamic programming approach.

Tables 2–4 gather information on minimum, average and maximum rule length and minimum and average and maximum support of the decision rules generated by the proposed algorithm, respectively. The results were obtained for the 100%, 80% and 60% of best attributes chosen during the rule generation.

Taking into account the rule length it can be seen that for some data sets, the maximum and average values are much smaller than the number of conditional attributes in decision table, for example, lymphography, zoo-data. In case of support, maximum value should be noticed comparing to the number of rows in decision table, for cars, hayes-roth-data, house-votes and zoo-data.

Table 2. Rule-quality measures for 100% attributes.

Decision Table	Number of		Rule Length			Rule Support		
	Rows	Attributes	Min	Avg	Max	Min	Avg	Max
balance-scale	625	4	3	3.64	4	1	2.44	5
breast-cancer	266	9	1	5.40	9	1	3.72	22
cars	1728	6	1	3.61	6	1	207.01	576
hayes-roth-data	69	5	1	2.64	4	1	3.81	12
house-votes	279	16	3	6.61	16	1	30.73	82
lymphography	148	18	1	3.93	12	1	2.70	9
nursery	12,960	8	7	7.14	8	1	2.72	3
shuttle-landing-control	15	6	1	3.07	6	1	1.93	3
soybean-small	47	35	2	4.40	9	1	2.79	5
zoo-data	59	16	1	4.86	10	1	5.61	12

Table 3. Rule-quality measures for 80% of best attributes.

Decision Table	Number of		Rule Length			Rule Support		
	Rows	Attributes	Min	Avg	Max	Min	Avg	Max
balance-scale	625	4	3	3.64	4	1	2.44	5
breast-cancer	266	9	1	5.32	8	1	3.72	22
cars	1728	6	1	3.20	5	2	207.55	576
hayes-roth-data	69	5	1	2.64	4	1	3.81	12
house-votes	279	16	3	6.69	13	1	30.73	82
lymphography	148	18	1	3.93	12	1	2.70	9
nursery	12,960	8	7	7.00	7	2	2.86	3
shuttle-landing-control	15	6	1	2.58	5	1	1.93	3
soybean-small	47	35	2	4.40	9	1	2.79	5
zoo-data	59	16	1	4.41	10	1	5.61	12

Table 4. Rule-quality measures for 60% of best attributes.

Decision Table	Number of		Rule Length			Rule Support		
	Rows	Attributes	Min	Avg	Max	Min	Avg	Max
balance-scale	625	4	3	3.00	3	2	3.97	5
breast-cancer	266	9	1	4.83	6	1	4.12	22
cars	1728	6	1	2.78	4	6	210.46	576
hayes-roth-data	69	5	1	2.33	3	1	3.94	12
house-votes	279	16	3	6.21	10	1	30.99	82
lymphography	148	18	1	3.87	11	1	2.70	9
nursery	12,960	8	5	5.00	5	8	10.17	12
shuttle-landing-control	15	6	1	2.36	4	1	1.93	3
soybean-small	47	35	2	4.40	9	1	2.79	5
zoo-data	59	16	1	4.46	10	1	5.61	12

The mentioned quality measures needed to be compared with the ones obtained for the optimal rules (with respect to length and support respectively) generated by the DP (dynamic programming) approach shown in [46]. In order to be able to make the comparison more informative, the relative difference of the respective results have been calculated. It is defined as follows:

$$Relative\_difference = \frac{value\_for\_algorithm - value\_for\_DP}{value\_for\_DP} \quad (7)$$

The results are presented in Tables 5–7. Following the formula for the relative difference, it can be seen that positive values mean that results for the proposed algorithm are larger than the ones obtained for the DP approach, whilst negative values mean that results for the DP approach are larger than the ones obtained for the proposed algorithm. Zero means that both values are equal.

Table 5. Relative difference of length and support of decision rules for 100% attributes.

Decision Table	Average Rule Length	Average Rule Support
balance-scale	**0.14**	**−0.42**
breast-cancer	1.03	−0.61
cars	**0.48**	**−0.38**
hayes-roth-data	**0.23**	**−0.42**
house-votes	1.60	−0.58
lymphography	0.97	−0.87
nursery	1.29	−1.00
shuttle-landing-control	1.19	**−0.09**
soybean-small	3.40	−0.78
zoo-data	2.12	−0.49

Table 6. Relative difference of length and support of decision rules for 80% attributes.

Decision Table	Average Rule Length	Average Rule Support
balance-scale	**0.14**	**−0.42**
breast-cancer	1.00	−0.61
cars	**0.32**	**−0.38**
hayes-roth-data	**0.23**	**−0.42**
house-votes	1.60	−0.58
lymphography	0.97	−0.87
nursery	1.25	−1.00
shuttle-landing-control	0.85	**−0.09**
soybean-small	3.40	−0.78
zoo-data	1.83	−0.49

Table 7. Relative difference of length and support of decision rules for 60% attributes.

Decision Table	Average Rule Length	Average Rule Support
balance-scale	**−0.06**	**−0.07**
breast-cancer	0.81	−0.57
cars	**0.14**	**−0.37**
hayes-roth-data	**0.09**	−0.40
house-votes	1.45	−0.58
lymphography	0.94	−0.87
nursery	0.60	−0.99
shuttle-landing-control	0.69	**−0.09**
soybean-small	3.40	−0.78
zoo-data	1.86	−0.49

The presented results show that the rules generated by the proposed algorithm are not far from the optimal ones. Moreover, reducing the number of attributes resulted in an improvement of the rule quality (for 60% of attributes quality is the closest to the optimal one). Values in Tables 5–7 marked in bold denote results close to the ones obtained for the DP approach for the rules optimized with respect to their length and support respectively. It should be noticed that in Table 7 for the balance-scale decision table, the negative relative difference value is due to the fact that the number of attributes in data set was reduced to 60% of the whole number of attributes, so the average value of the constructed rules was shorter than the optimal value.

Experiments connected with classification have also been performed. To make comparison with results obtained for DP approach, the classification procedure was the same as the one described in [20].

Table 8 presents average classification error, for decision tables with 100%, 80% and 60% of best attributes, using two-fold cross validation method. For each decision table experiments were repeated 50 times. Each dataset was randomly divided randomly into three parts: train—30%, validation—20%, and test—50%. Rule-based classifier was constructed on the train part, then pruned by minimum error on validation set and used on test part of decision table. Presented classification error it is the number of objects from the test part of decision table which are incorrectly classified divided by the number of all objects in the test part of decision table. The last row of Table 8 presents the average classification error for all considered decision tables. Column Std denotes standard deviation for obtained results. The smallest mean errors have been marked in bold-the ones for which mean error is smaller than twenty percent.

Table 8. Average classification error for the proposed algorithm.

Decision Table	Number of		100% of Attributes		80% of Attributes		60% of Attributes	
	Rows	Attributes	Mean Error	Std	Mean Error	Std	Mean Error	Std
balance-scale	625	4	0.45	0.08	0.45	0.08	0.48	0.08
breast-cancer	266	9	**0.00**	0.00	**0.00**	0.00	**0.00**	0.02
cars	1728	6	**0.07**	0.11	**0.08**	0.13	**0.18**	0.21
hayes-roth-data	69	5	0.52	0.10	0.52	0.10	0.51	0.12
house-votes	279	16	0.25	0.12	0.25	0.12	0.26	0.14
lymphography	148	18	**0.16**	0.11	**0.16**	0.10	**0.14**	0.11
nursery	12,960	8	**0.00**	0.01	**0.00**	0.01	**0.00**	0.0
shuttle-landing-control	15	6	0.22	0.19	**0.18**	0.17	0.22	0.18
soybean-small	47	35	0.21	0.14	0.21	0.14	0.21	0.14
zoo-data	59	16	0.40	0.26	0.38	0.25	0.36	0.24
average	-	-	0.21	0.10	0.20	0.10	0.21	0.11

The classification results obtained for the DP approach (introduced in [20]) are presented in Table 9.

Table 9. Average classification error for the DP approach.

Decision Table	DP Optimized w/r Length	DP Optimized w/r Support
	Mean Error	Mean Error
balance-scale	0.29	0.28
breast-cancer	0.31	0.30
cars	0.22	0.21
hayes-roth-data	0.37	0.35
house-votes	0.08	0.05
lymphography	0.35	0.28
nursery	0.05	0.05
shuttle-landing-control	0.40	0.39
soybean-small	0.17	0.17
zoo-data	0.24	0.18
average	0.25	0.23

Statistical analysis of classification results using the Wilcoxon two-tailed test has also been performed as per [47]. The classification results have been compared with the ones for DP approach for decision rules optimized relative to length and support respectively. For Table 8 it turns out that $min(W_+, W_-) > W_{crit}$ (for 100%, 80% and 60% of best attributes), so the conclusion can be made that there is no significant difference between classification results obtained by the proposed algorithm and the DP approach. The null hypothesis has been confirmed. The goal of the proposed algorithm was to construct short rules, of enough good support, but keeping high level of decision values' distinguishability.

All experiments were performed on a portable computer with the following technical specifications:

- Intel i5-8365U CPU,
- 16 GB of RAM memory,
- Windows 10 Enterprise x64 operating system.

The algorithm has been implemented in Java 8 accompanied by Spring Boot framework. All data related calculations have been performed on PostgreSQL 13.0 RDBMS. Software communicates with the database through JDBC connector.

5. Conclusions

A new algorithm for decision rules generation has been introduced in this paper. It has been shown that in the average case its computational complexity is linear. It makes the algorithm applicable to a vast majority of different data sets. Moreover, it has been experimentally shown that the rules generated by the mentioned algorithm are, of comparable classification quality to the ones generated by the dynamic programming approach which is not enough good from the time and memory complexity point of view, in opposite to the proposed one. Additionally, presented approach due to its specificity, allows to reduce number of attributes choosing the most significant ones and thus allowing to minimize computational effort even to a larger extent. Having compared the results by means of Wilcoxon test, it is possible to state that for 100%, 80% and 60% of attributes, classification results for the rules generated by the proposed algorithm are comparable to the ones obtained by the dynamic programming approach. It means that the number of attributes taken into consideration can be reduced by forty percent and still the classification results are satisfactory. Furthermore, reduction of the number of attributes often increases the support of generated decision rules and helps to avoid their over-learning.

In our future works, we would like to compare the proposed solution with other approaches for decision rules construction, e.g., heuristic-based approach. We also plan to

look for further improvements and algorithm tuning. Additionally, we plan to look for its potential real-world applications.

Author Contributions: Conceptualization K.Ż. and B.Z.; methodology, K.Ż. and B.Z.; investigation, B.Z. and K.Ż.; software, K.Ż.; validation, K.Ż. and B.Z.; writing—original draft preparation, K.Ż. and B.Z.; writing—review and editing, K.Ż. and B.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [45].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

EAV	Entity–Attribute–Value
EAVD	Entity–Attribute–Value-Decision
DP	Dynamic Programming
STD	Standard Deviation

References

1. Tsybunov, E.; Shubenkova, K.; Buyvol, P.; Mukhametdinov, E. Interactive (Intelligent) Integrated System for the Road Vehicles' Diagnostics. In *Intelligent Transport Systems—From Research and Development to the Market Uptake*; Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering; Springer International Publishing: Cham, Switzerland, 2018; Volume 222, pp. 195–204.
2. Stańczyk, U.; Zielosko, B.; Żabiński, K. Application of Greedy Heuristics for Feature Characterisation and Selection: A Case Study in Stylometric Domain. In Proceedings of the Rough Sets—International Joint Conference, IJCRS 2018, Quy Nhon, Vietnam, 20–24 August 2018; Nguyen, H., Ha, Q., Li, T., Przybyla-Kasperek, M., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018; Volume 11103, pp. 350–362.
3. Raza, M.S.; Qamar, U. *Understanding and Using Rough Set Based Feature Selection: Concepts, Techniques and Applications*, 2nd ed.; Springer: Singapore, 2019.
4. Baron, G.; Stańczyk, U. Performance evaluation for ranking-based discretisation. In Proceedings of the 24th International Conference Knowledge-Based and Intelligent Information & Engineering Systems: KES-2020, Virtual Event, Verona, Italy, 16–18 September 2020; Cristani, M., Toro, C., Zanni-Merk, C., Howlett, R.J., Jain, L.C., Eds.; Procedia Computer Science; Elsevier: Amsterdam, The Netherlands, 2020; Volume 176, pp. 3335–3344.
5. Mitrokhin, M.; Zaharov, S.; Mitrokhina, N. Semisupervised learning in pattern recognition with concept drift. In Proceedings of the 2020 Moscow Workshop on Electronic and Networking Technologies (MWENT), Moscow, Russia, 11–13 March 2020; pp. 1–4.
6. Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*; Morgan Kaufmann: Burlington, MA, USA, 2011.
7. Liu, H.; Cocea, M. Induction of classification rules by Gini-index based rule generation. *Inf. Sci.* **2018**, *436-437*, 227–246. [CrossRef]
8. Wróbel, L.; Sikora, M.; Michalak, M. Rule Quality Measures Settings in Classification, Regression and Survival Rule Induction—An Empirical Approach. *Fundam. Inform.* **2016**, *149*, 419–449. [CrossRef]
9. An, A.; Cercone, N. Rule Quality Measures Improve the Accuracy of Rule Induction: An Experimental Approach. In Proceedings of the International Symposium on Methodologies for Intelligent Systems (ISMIS), Charlotte, NC, USA, 11–14 October 2000; Raś, Z.W., Ohsuga, S., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1932, pp. 119–129.
10. Rissanen, J. Modeling by shortest data description. *Automatica* **1978**, *14*, 465–471. [CrossRef]
11. Nguyen, H.S.; Ślęzak, D. Approximate Reducts and Association Rules—Correspondence and Complexity Results. In Proceedings of the 7th International Workshops on Rough Sets, Fuzzy Sets and Granular Soft Computing (RSFDGrC'96), Yamaguchi, Japan, 9–11 November 2020; Zhong, N.; Skowron, A., Ohsuga, S., Eds.; LNCS; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1711, pp. 137–145.
12. Bonates, T.; Hammer, P.L.; Kogan, A. Maximum Patterns in Datasets. *Discret. Appl. Math.* **2008**, *156*, 846–861. [CrossRef]
13. Moshkov, M.J.; Piliszczuk, M.; Zielosko, B. *Partial Covers, Reducts and Decision Rules in Rough Sets—Theory and Applications*; Studies in Computational Intelligence; Springer: Berlin/Heidelberg, Germany, 2008; Volume 145.

14. Świeboda, W.; Nguyen, H.S. Rough Set Methods for Large and Spare Data in EAV Format. In Proceedings of the IEEE RIVF International Conference on Computing Communication Technologies, Research, Innovation, and Vision for the Future, Ho Chi Minh City, Vietnam, 27 February–1 March 2012; pp. 1–6.
15. Zielosko, B. Optimization of Approximate Decision Rules Relative to Coverage. In *Beyond Databases, Architectures, and Structures, Proceedings of the 10th International Conference, BDAS 2014, Ustron, Poland, 27–30 May 2014*; Kozielski, S., Mrozek, D., Kasprowski, P., Malysiak-Mrozek, B., Kostrzewa, D., Eds.; Communications in Computer and Information Science; Springer: Cham, Switzerland, 2014; Volume 424, pp. 170–179.
16. Zielosko, B.; Żabinski, K. Optimization of Approximate Decision Rules Relative to Length. In *Beyond Databases, Architectures and Structures. Facing the Challenges of Data Proliferation and Growing Variety, Proceedings of the 14th International Conference, BDAS 2018, 24th IFIP World Computer Congress, WCC 2018, Poznan, Poland, 18–20 September 2018*; Kozielski, S., Mrozek, D., Kasprowski, P., Malysiak-Mrozek, B., Kostrzewa, D., Eds.; Communications in Computer and Information Science; Springer: Cham, Switzerland, 2018; Volume 928, pp. 171–181.
17. Zielosko, B.; Żabinski, K. Optimization of Decision Rules Relative to Length Based on Modified Dynamic Programming Approach. In *Advances in Feature Selection for Data and Pattern Recognition*; Stańczyk, U., Zielosko, B., Jain, L.C., Eds.; Intelligent Systems Reference Library; Springer: Cham, Switzerland, 2018; Volume 138, pp. 73–93.
18. Stefanowski, J.; Vanderpooten, D. Induction of decision rules in classification and discovery-oriented perspectives. *Int. J. Intell. Syst.* **2001**, *16*, 13–27. [CrossRef]
19. Pawlak, Z.; Skowron, A. Rough sets and Boolean reasoning. *Inf. Sci.* **2007**, *177*, 41–73. [CrossRef]
20. Amin, T.; Chikalov, I.; Moshkov, M.; Zielosko, B. Dynamic programming approach to optimization of approximate decision rules. *Inf. Sci.* **2013**, *119*, 403–418. [CrossRef]
21. Moshkov, M.; Zielosko, B. *Combinatorial Machine Learning—A Rough Set Approach*; Studies in Computational Intelligence; Springer: Berlin/Heidelberg, Germany, 2011; Volume 360, pp. 1–170.
22. Pawlak, Z.; Skowron, A. Rudiments of rough sets. *Inf. Sci.* **2007**, *177*, 3–27. [CrossRef]
23. Sikora, M.; Wróbel, L.; Gudyś, A. GuideR: A guided separate-and-conquer rule learning in classification, regression, and survival settings. *Knowl.-Based Syst.* **2019**, *173*, 1–14. [CrossRef]
24. Błaszczyński, J.; Słowiński, R.; Szeląg, M. Sequential covering rule induction algorithm for variable consistency rough set approaches. *Inf. Sci.* **2011**, *181*, 987–1002. [CrossRef]
25. Fürnkranz, J.; Flach, P.A. ROC 'n' rule learning–towards a better understanding of covering algorithms. *Mach. Learn.* **2005**, *58*, 39–77. [CrossRef]
26. Clark, P.; Niblett, T. The CN2 induction algorithm. *Mach. Learn.* **1989**, *3*, 261–283. [CrossRef]
27. Cendrowska, J. PRISM: An algorithm for inducing modular rules. *Int. J. Man-Mach. Stud.* **1987**, *27*, 349–370. [CrossRef]
28. Wojtusiak, J.; Michalski, R.S.; Kaufman, K.A.; Pietrzykowski, J. The AQ21 Natural Induction Program for Pattern Discovery: Initial Version and its Novel Features. In Proceedings of the 2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06), Arlington, VA, USA, 13–15 November 2006; pp. 523–526.
29. Cohen, W.W. Fast Effective Rule Induction. In *Machine Learning Proceedings 1995*; Prieditis, A., Russell, S., Eds.; Morgan Kaufmann: Burlington, MA, USA, 1995; pp. 115–123.
30. Grzymała-Busse, J.W. A New Version of the Rule Induction System LERS. *Fundam. Inform.* **1997**, *31*, 27–39. [CrossRef]
31. Nguyen, H.S. Approximate Boolean reasoning: Foundations and applications in data mining. In *Transactions on Rough Sets V*; Peters, J.F., Skowron, A., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4100, pp. 334–506.
32. Moshkov, M.J.; Piliszczuk, M.; Zielosko, B. On Construction of Partial Reducts and Irreducible Partial Decision Rules. *Fundam. Inform.* **2007**, *75*, 357–374.
33. Valmarska, A.; Lavrač, N.; Fürnkranz, J.; Robnik-Šikonja, M. Refinement and selection heuristics in subgroup discovery and classification rule learning. *Expert Syst. Appl.* **2017**, *81*, 147–162. [CrossRef]
34. Freitas, A.A. Genetic Algorithms for Rule Discovery. In *Data Mining and Knowledge Discovery with Evolutionary Algorithms*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 107–137.
35. Ślęzak, D.; Wróblewski, J. Order Based Genetic Algorithms for the Search of Approximate Entropy Reducts. In *RSFDGrC 2003*; Wang, G., Liu, Q., Yao, Y., Skowron, A., Eds.; LNCS; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2639, pp. 308–311.
36. Rothlauf, F.; Schunk, D.; Pfeiffer, J. Classification of human decision behavior: Finding modular decision rules with genetic algorithms. In Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation (GECCO), Washington, DC, USA, 25–29 June 2005; pp. 2021–2028.
37. Forsati, R.; Moayedikia, A.; Jensen, R.; Shamsfard, M.; Meybodi, M.R. Enriched ant colony optimization and its application in feature selection. *Neurocomputing* **2014**, *142*, 354–371. [CrossRef]
38. Liu, B.; Abbass, H.A.; McKay, B. Classification Rule Discovery with Ant Colony Optimization. In Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003), Halifax, NS, Canada, 13–17 October 2003; pp. 83–88.
39. Zomorodi-moghadam, M.; Abdar, M.; Davarzani, Z.; Zhou, X.; Pławiak, P.; Acharya, U. Hybrid particle swarm optimization for rule discovery in the diagnosis of coronary artery disease. *Expert Syst.* **2019**. [CrossRef]
40. Grzegorowski, M.; Ślęzak, D. On resilient feature selection: Computational foundations of r-C-reducts. *Inf. Sci.* **2019**, *499*, 25–44. [CrossRef]

41. Jensen, R.; Shen, Q. Semantics-preserving dimensionality reduction: Rough and fuzzy-rough-based approaches. *IEEE Trans. Knowl. Data Eng.* **2004**, *16*, 1457–1471. [CrossRef]
42. Sarawagi, S.; Thomas, S.; Agrawal, R. Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications. *Data Min. Knowl. Discov.* **2004**, *4*, 89–125. [CrossRef]
43. Kowalski, M.; Stawicki, S. SQL-Based Heuristics for Selected KDD Tasks over Large Data Sets. In Proceedings of the 2012 Federated Conference on Computer Science and Information Systems (FedCSIS), Wroclaw, Poland, 9–12 September 2012; pp. 303–310.
44. Ślęzak, D. Rough Sets and Bayes Factor. *Transactions on Rough Sets III*; Peters, J.F., Skowron, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 202–229.
45. Asuncion, A.; Newman, D.J. UCI Machine Learning Repository, University of Massachusetts Amherst, USA. 2007. Available online: https://archive.ics.uci.edu (accessed on 30 September 2020).
46. Amin, T.; Chikalov, I.; Moshkov, M.; Zielosko, B. Dynamic Programming Approach for Exact Decision Rule Optimization. In *Rough Sets and Intelligent Systems—Professor Zdzisław Pawlak in Memoriam—Volume 1*; Skowron, A., Suraj, Z., Eds.; Intelligent Systems Reference Library; Springer: Berlin/Heidelberg, Germany, 2013; Volume 42, pp. 211–228.
47. Hole, G. *Graham Hole Research Skills*; University of Sussex: Brighton, UK, 2011. Available online: http://users.sussex.ac.uk/~grahamh/RM1web/WilcoxonHandoout2011.pdf (accessed on 31 October 2020).