algorithms

MDPI

Article

Efficient Rule Generation for Associative Classification

Chartwut Thanajiranthorn * and Panida Songram

Department of Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham 44150, Thailand; panida.s@msu.ac.th

* Correspondence: chartwut@bru.ac.th; Tel.: +66-619-395-455

Received: 2 September 2020; Accepted: 14 November 2020; Published: 17 November 2020

Abstract: Associative classification (AC) is a mining technique that integrates classification and association rule mining to perform classification on unseen data instances. AC is one of the effective classification techniques that applies the generated rules to perform classification. In particular, the number of frequent ruleitems generated by AC is inherently designated by the degree of certain minimum supports. A low minimum support can potentially generate a large set of ruleitems. This can be one of the major drawbacks of AC when some of the ruleitems are not used in the classification stage, and thus (to reduce the rule-mapping time), they are required to be removed from the set. This pruning process can be a computational burden and massively consumes memory resources. In this paper, a new AC algorithm is proposed to directly discover a compact number of efficient rules for classification without the pruning process. A vertical data representation technique is implemented to avoid redundant rule generation and to reduce time used in the mining process. The experimental results show that the proposed algorithm archives in terms of accuracy a number of generated ruleitems, classifier building time, and memory consumption, especially when compared to the well-known algorithms, Classification-based Association (CBA), Classification based on Multiple Association Rules (CMAR), and Fast Associative Classification Algorithm (FACA).

Keywords: associative classification; class association rule; vertical data representation; classification

1. Introduction

Nowadays, there a number of classification techniques that have been applied to various real-world applications, i.e., graph convolutional networks for text classification [1], automated classification of epileptic electroencephalogram (EEG) signals [2], iris cognition [3], and anomaly detection [4]. Associative classification (AC) is a well-known classification technique that was first introduced by Lui et al. [5]. It is a combination of two data-mining techniques, association rule mining, and classification. Association rule mining discovers the relationship between items in a dataset. Meanwhile, classification aims to predict the class label of any given instance from learning-labeled dataset. AC focuses on finding Class Association Rules (CARs) that satisfy certain minimum support and confidence thresholds in the form $x \rightarrow c$, where x is a set of attribute values and c is a class label. AC has been reported in the literature to outperform other traditional classifiers [6–13], In addition, a CAR is an if–then rule that can be easily understood by general users. Therefore, AC is applied in many fields, i.e., phishing website detection [6,7,11], heart disease prediction [8,9], groundwater detection [12], and detection of low-quality information in social networks [10].

In traditional AC algorithms, minimum support threshold is a significant key parameter that is used to select frequent ruleitems and to then eliminate frequent ruleitems in which confidence values do not satisfy minimum confidence. This manner leads to a large number of frequent ruleitems. Nguyen and Nguyen [14] demonstrated that the number of 4 million frequent ruleitems can be generated when the minimum support threshold is set to 1%. Moreover, a number of AC-based

techniques, i.e., Classification-based Association (CBA) [5], Fast Associative Classification Algorithm (FACA) [11], CAR-Miner-diff [14], Predictability-Based Class Collative Class Association Rules (PCAR) [15], Weighted Classification Based on Association Rules (WCBA) [16], and Fast Classification Based on Association Rules (FCBA) [17], create all possible CARs in order to determine a set of valid CARs that can be used in the classification process. Recently, Active Pruning Rules (APR) [13] has been proposed as a novel evaluation method. APR can be used to avoid generating all CARs. However, the exhaustive search for finding rules in classifiers may cause an issue in large datasets or low minimum support. Creating candidate CARs consumes intensive computational times and memory. The minimal process of candidate generation is still challenging because it is quite affected in terms of training time, input/output (I/O) overheads, and memory usage [18].

In this paper, a new algorithm is proposed to directly generate a small number of efficient CARs for classification. A vertical data format [19] is used to represent ruleitems associated with their transaction IDs. The intersection technique is used to easily calculate support and confidence values from the format. The ruleitems with 100% of confidence will be added to the classifier as a CAR. Whenever a CAR with 100% confidence is found, the transaction associated with the CAR will be removed by using a set difference to avoid generating redundant CARs. Finally, a compact classifier is built for classification. In conclusion, the contribution of this paper is as follows.

1. To avoid pruning and sorting processes, the proposed algorithm directly generates CARs with 100% confidence to build compact classifiers. The CARs with 100% confidence are anticipated to result in high prediction rates.
2. The proposed algorithm eliminates unnecessary transactions to avoid generating redundant CARs in each stage.
3. Simple set theories, intersection, and set difference are exploited to reduce computational time used in mining process and to reduce memory consumption.

This paper is structured as follows. In Section 2, related works of AC are described. The basic definitions are delineated in Section 3. The proposed algorithm is introduced in Section 4. The discussion on the experimental results is in Section 5. Lastly, the conclusion of the study is stated in Section 6.

2. Related Work

In the past, AC-based algorithms have been proposed and studied. The study's objective is to understand some drawbacks and to increase the effectiveness of the algorithms. Lui et al. [5] introduced the CBA algorithm which integrated association rule mining and classification. The process of the CBA algorithm is divided into two steps. First, CARs are generated based on the famous search method the Apriori algorithm [20]. Second, CARs are sorted and then pruned to select efficient CARs in a classifier. The CBA algorithm was proven to produce a lower error rate than C4.5 [21]. Unfortunately, the CBA algorithm encounters a large number of candidate generation problems due to Apriori inheritance which finds all possible frequent rules at each level.

Li et al. [22] presented the Classification based on Multiple Association Rules (CMAR) algorithm. Unlike CBA, CMAR adopts a Frequent pattern tree (FP-tree) and a Cosine R-tree (CR-tree) for rule generation and classification phases. It divides the subset in FP-tree to search frequent ruleitems and then adds the frequent ruleitems to CR-tree according to their frequencies. Hence, CMAR only needs to scan the database once. The CMAR algorithm uses multiple rules to predict unseen instances based on chi-square method. In the experiment, CMAR was compared with CBA and C4.5 in terms of accuracy. The experimental result shows that CMAR performs better than the others.

Abdelhamid [6] proposed an Enhanced Multi-label Classifier-based Associative Classification (eMCAC) for phishing website detection. It generates rules with multiple class labels from a single dataset without recursive learning. The eMCAC algorithm applies a vertical data format to represent datasets. The support and confidence values for a multi-label rule are calculated based on the average

support and confidence values of all classes. The class is assigned to the test instance if attribute values are fully matched to the rule's antecedent. The experimental results show that the eMCAC algorithm outperforms CBA, PART, C4.5, jRiP, and MCAR [23] on the real-world phishing data in terms of accuracy.

Hadi et al. [11] proposed the FACA algorithm for phishing website detection. It applies a Diffset [24] in the rule-generation process to increase the speed of classifier building time. First, the FACA algorithm discovers k-ruleitems by extending frequent (k − 1)-ruleitems. Then, ruleitems are ranked according to the number of attribute values, confidence, support, and occurrence. To predict unseen data, the FACA algorithm utilizes the All Exact Match Prediction Method. The method matches unseen data with all CARs in the classifiers. Next, unseen data are assigned to the class label with the highest count. From the experimental result, the FACA algorithm outperforms CBA, CMAR, MCAR, and ECAR [25] in terms of accuracy.

Song and Lee [15] introduced Predictability-Based Collective Class Association Rule algorithm (PCAR) to enhance rule evaluation. The PCAR algorithm uses inner cross-validation between the test dataset and train dataset to calculate a predictability value of CARs. Then, CARs are ranked according to rule predictive values, rule confidence, rule support, rule antecedent length, and rule occurrences. Finally, the full-matching method is applied to assign a class label for unseen data. To evaluate the performance of PCAR, PCAR was compared with C4.5, RIPPER, CBA, and MCAR on the accuracy, and PCAR was shown to outperform the others.

Alwidian et al. [16] proposed the WCBA algorithm to enhance the accuracy of a classifier based on the weighting technique. WCBA assumes that the importance of attributes is not equal. For example, in medicine, some attributes are more important than other attributes for prediction. Consequently, weights of all attributes are assigned by experts in the domain. Then, the weighted method is used to select useful CARs and a statistical measure is used for the pruning process. In addition, CARs are priors sorted by using the harmonic mean, which is an average value between support and confidence. The WCBA algorithm is more significantly accurate than CBA, CMAR, MCAR, FACA, and ECBA. However, the WCBA algorithm generates CARs based on the Apriori technique that scans the database many times.

Rajab proposed [13] the Active Pruning Rule (APR) algorithm. The new pruning process was introduced in APR. CARs are ranked by confidence, support, and rule length. Each training instance is matched over a set of CARs. The first rule that matches an instance is added to the classifier. Then, instances containing the first rule are removed. The support and confidence of remaining rules are recalculated, and all CARs are re-ranked. The APR algorithm was proven to reduce the size of the classifier and to maintain predictive accuracy performance. However, the APR algorithm still has to face a massive number of candidates from a rule-generation process. From previous works, the advantages and disadvantages are shown in Table 1.

The previous algorithms on AC generally result in high predictability of rules. However, most of them produce k-ruleitems from (k − 1)-ruleitems. They have to calculate supports when a new ruleitems is recovered. To calculate support and confidence values, they have to search all transactions in databases multiple times. Moreover, a huge number of candidate CARs are generated and pruned later to reduce unnecessary CARs. To reduce the problems, the proposed algorithm will directly generate efficient CARs for classification so that the pruning and sorting processes are not necessary. The efficient CARs in our works are rules with 100% confidence which are generated based on the idea that some attribute values can immediately indicate the class label if all attribute values belong to a class label. To easily check attribute values belonging to any class label, vertical data representation is used in the proposed algorithm. Furthermore, simple set theories, intersection, and set difference are adapted to easily calculate support and confidence values without scanning a database multiple times.

Table 1. Advantages and disadvantages of Associative classification (AC) algorithms.

Algorithms	Advantage	Disadvantage
CBA	It adopted the association rule technique to classify data that is proven to be more accurate than the traditional classification technique.	It has to face a sensitivity of the minimum support threshold. A massive number of rules are generated when a low minimum support threshold is given.
CMAR	It uses an efficient FP-tree, which consumes less memory and space compared to CBA.	The FP-tree will not always fit in the main memory, especially when the number of attributes is large.
eMCAC	It adopts vertical data representation to reduce space usage to find a multi-label class.	It is based on an Apriori-like technique that can result in a large number of frequent itemsets.
FACA	Set difference is adopted to consume low memory and to reduce the mining time.	It is based on an Apriori-like technique; therefore, the algorithm is required to search for all frequent itemsets from all possible candidate itemsets at each level.
PCAR	It uses predictability value to prune unnecessary rules.	The execution time is slow since it includes the inner cross-validation phase for calculation predictability value.
WCBA	It uses a weighted method to select useful rules and to improve the performance of the classifier.	Weighted factors are subject to change due to the decisions of experts which can cause a different experimental result.
APR	A new evaluation method with a small classifier and high accuracy rate.	Generation of a large number of rules when a low minimum support threshold is given.

3. Basic Definitions

Let $A = \{a_1, a_2, \ldots, a_m\}$ be a finite set of all attributes in dataset. $C = \{c_1, c_2, \ldots, c_n\}$ is a set of classes, $g(x)$ is a set of transactions containing itemset x, and $|g(x)|$ is the number of transactions containing x.

Definition 1. *An item can be described as an attribute a_i containing a value v_j, denoted as (a_i, v_j).*

Definition 2. *An itemset is the set of items, denoted as $(a_{i1}, v_{i1}), (a_{i2}, v_{i2}), ..., (a_{ik}, v_{ik})$.*

Definition 3. *A ruleitem is of the form $\langle itemset, c_j \rangle$, which represents an association between itemsets and class in a dataset; basically, it is represented in the form $itemset \rightarrow c_j$.*

Definition 4. *The length of a ruleitem is the number of items, denoted as $k - ruleitem$.*

Definition 5. *The absolute support of ruleitem r is the number of transactions containing r, denoted as $sup(r)$. The support of r can be found from (1).*

$$sup(r) = |g(r)| \tag{1}$$

Definition 6. *The confidence of ruleitem $\langle itemset, c_j \rangle$ is the ratio of the number of transactions that contains the itemset in class in c_j and the number of transactions containing the itemset, as in (2).*

$$conf(\langle itemset, c_j \rangle) = \frac{|g(\langle itemset, c_j \rangle)|}{|g(itemset)|} \times 100 \tag{2}$$

Definition 7. *Frequent ruleitem is a ruleitem in which support is not less than the minimum support threshold ($minsup$).*

Definition 8. *Class Association Rule (CAR) is a frequent ruleitem in which confidence is not less than the minimum confidence threshold ($minconf$).*

4. The Proposed Algorithm

In this section, a new algorithm, called the Efficient Class Association Rule Generation (ECARG) algorithm, is presented. The pseudo code of the proposed algorithm is shown in Algorithm 1.

Algorithm 1: Efficient Class Association Rule Generation (ECARG) algorithm main process

```
   Input: dataset, minsup
   Output: classifier
1  ruleItems = 1-ruleitem generation from dataset
2  while at least one rule's support meet minsup do
3      R = maximum confidence rule from ruleItems // ruleitem's support ≥ minsup
4      if R's confidence < 100 and R is not null then
5          R = extend R with the other ruleitems
6      if R is not null then
7          insert R to classifier
8          redundant rule removal
9          update support and confidence for each ruleItems
10     else
11         exit while loop
12 finding the default class
13 return classifier
```

First, 1-frequent ruleitems are generated (line 1). To quickly find 1-frequent ruleitems, the proposed algorithm takes the advantage of a vertical data format to calculate the support of

the ruleitems. The support of the ruleitems can be obtained from $|g(itemset) \cap g(c_k)|$. If any 1-ruleitem does not meet the minimum support threshold, it will not be extended with the other ruleitems. Moreover, the confidence of the frequent ruleitems can be calculated from Equation (2) by using the vertical data format. If the confidence of the ruleitem is 100%, the ruleitems will be added to the classifier directly (line 7); otherwise, it will be considered extended with the others (line 5).

After discovering the most effective CAR with 100% confidence, the transaction IDs associated with the CAR will be removed to avoid redundant CARs (line 8). To remove the transaction IDs, a set difference plays an important role in our algorithm. Let r_i be a CAR with 100% confidence and T be a set of ruleitems in the same class of r_i. For all $r_j \in T$, the new transaction IDs of r_j is $g(r_j) = g(r_j) - g(r_i)$. Then, the new transaction IDs, support, and confidence values of all rules are updated (line 9).

In each iteration, if there is no CAR with 100% confidence, the ruleitem r with the highest confidence will be first to be considered extended in a breadth-first search manner. It will be combined with other ruleitems in the same class until the new CAR has 100% confidence (line 5). If r_i is extended with r_j to be r_{new} and $g(r_j) \subseteq g(r_i)$, then $conf(r_{new}) = 100\%$. After the extended CAR is added to the classifier, the transaction IDs associated with the CAR will be removed. Finally, if no ruleitem satisfies the minimum support threshold, the CAR generation will be stopped.

The proposed algorithm continues to find a default class in order to insert it to the classifier. The class with the most remaining transaction IDs is selected as the default class (line 12).

To demonstrate the examples, the dataset in Table 2 is used as example data. The minimum support and confidence thresholds are set to 2 and 50%, respectively.

Table 2. A sample dataset.

TID	atr1	atr2	atr3	Class Label
1	a_1	b_1	c_1	A
2	a_1	b_1	c_2	A
3	a_1	b_2	c_1	A
4	a_1	b_3	c_1	A
5	a_2	b_1	c_2	B
6	a_2	b_2	c_2	B
7	a_2	b_3	c_1	B
8	a_3	b_2	c_2	A
9	a_2	b_3	c_1	A

The vertical data format represents associated transaction IDs of 1-ruleitem, as shown in Table 3. The last 2 columns of Table 3 show the support and confidence of ruleitems that are calculated. From Table 2, the a_2 value in $atr1$ occurs in transaction IDs 5, 6, 7, and 9, denoted as $g(\langle atr1, a_2\rangle) = \{5,6,7,9\}$. Class A is in transaction IDs 1, 2, 3, 4, 8, and 9, denoted as $g(A) = \{1,2,3,4,8,9\}$, while class B is in transaction IDs 5, 6, and 7, denoted as $g(B) = \{5,6,7\}$. The transaction IDs containing $\langle atr1, a_2\rangle \rightarrow A$ are $g(\langle atr1, a_2\rangle) \cap g(A) = \{5,6,7,9\} \cap \{1,2,3,4,8,9\} = \{9\}$, so the supports of $\langle atr1, a_2\rangle \rightarrow A$ are 1. The rule $\langle atr1, a_2\rangle \rightarrow A$ will not be extended because its support is less than the minimum support threshold. Transaction IDs containing $\langle atr1, a_2\rangle \rightarrow B$ are $g(\langle atr1, a_2\rangle) \cap g(B) = \{5,6,7,9\} \cap \{5,6,7\} = \{5,6,7\}$, so the supports of $\langle atr1, a_2\rangle \rightarrow B$ are 3. Hence, this rule is a frequent ruleitem.

The confidence of $\langle atr1, a_2\rangle \rightarrow B$ can be obtained from $\frac{|g(5,6,7)|}{|g(5,6,7,9)|} \times 100 = \frac{3}{4} \times 100 = 75\%$. The confidence of $\langle atr1, a_2\rangle \rightarrow B$ is not 100% so it will be extended, whereas the confidence of $\langle atr1, a_1\rangle \rightarrow A$ is $\frac{|g(1,2,3,4)|}{|g(1,2,3,4)|} \times 100 = \frac{4}{4} \times 100 = 100\%$, so it is the first CAR added to the classifier.

Table 3. The rules that meet minimum support threshold (white background cell).

Ruleitem	TIDs	Sup	Conf (%)
$\langle atr1, a_1\rangle \rightarrow A$	1, 2, 3, 4	4	100
$\langle atr1, a_2\rangle \rightarrow A$	9	1	-
$\langle atr1, a_2\rangle \rightarrow B$	5, 6, 7	3	75
$\langle atr1, a_3\rangle \rightarrow A$	8	1	-
$\langle atr2, b_1\rangle \rightarrow A$	1, 2	2	66.67
$\langle atr2, b_1\rangle \rightarrow B$	5	1	-
$\langle atr2, b_2\rangle \rightarrow A$	3	1	-
$\langle atr2, b_2\rangle \rightarrow B$	6	1	-
$\langle atr2, b_3\rangle \rightarrow A$	4, 8, 9	3	75
$\langle atr2, b_3\rangle \rightarrow B$	7	1	-
$\langle atr3, c_1\rangle \rightarrow A$	1, 3, 4, 9	3	80
$\langle atr3, c_1\rangle \rightarrow B$	7	1	-
$\langle atr3, c_2\rangle \rightarrow A$	2, 8	2	50
$\langle atr3, c_2\rangle \rightarrow B$	5, 6	2	50

After discovering the first CAR, the transaction IDs associated with the CAR will be removed. From Table 3, if $\langle atr1, a_1\rangle$ is found, the class will absolutely be A. Hence, $\langle atr1, a_1\rangle \rightarrow A$ does not need to be extended with the other attribute values and transaction IDs 1, 2, 3, and 4 should be removed. The ECARG algorithm adopts a set difference, which can help to remove transaction IDs more conveniently.

For example, $g(\langle atr1, a_1\rangle \rightarrow A) = \{1,2,3,4\}$ and $g(\langle atr3, c_1\rangle \rightarrow A) = \{1,3,4,9\}$. The new transaction IDs of $g(\langle atr3, c_1\rangle \rightarrow A) = g(\langle atr3, c_1\rangle \rightarrow A) - g(\langle atr1, a_1\rangle \rightarrow A) = \{1,3,4,9\} - \{1,2,3,4\} = \{9\}$. Then, the new transaction IDs, support, and confidence values of all rules are updated as shown in Table 4.

From Table 4, there is no CAR with 100% confidence. $\langle atr1, a_2\rangle \rightarrow B$ has the maximum confidence, and $\langle atr3, c_2\rangle \rightarrow B = \{5,6\}$ is a subset of $g(\langle atr1, a_2\rangle \rightarrow B) = \{5,6,7\}$. Hence, the new rule $\langle (atr1, a_2), (atr3, c_2)\rangle \rightarrow B$ is found with 100% confidence. Then the extension of $\langle (atr1, a_2), (atr3, c_2)\rangle \rightarrow B$ is stopped. For 2-ruleitem extended from $\langle atr1, a_2\rangle \rightarrow B$, there is only one rule with 100% confidence and it is added to the classifier as the second CAR.

Table 4. The remained transaction IDs after generating the first Class Association Rule (CAR).

Ruleitem	TIDs	Sup	Conf (%)
$\langle atr1, a_2\rangle \rightarrow A$	9	1	-
$\langle atr1, a_2\rangle \rightarrow B$	5, 6, 7	3	75
$\langle atr1, a_3\rangle \rightarrow A$	8	1	-
$\langle atr2, b_1\rangle \rightarrow B$	5	1	-
$\langle atr2, b_2\rangle \rightarrow B$	6	1	-
$\langle atr2, b_3\rangle \rightarrow A$	8, 9	2	66.67
$\langle atr2, b_3\rangle \rightarrow B$	7	1	-
$\langle atr3, c_1\rangle \rightarrow A$	9	1	-
$\langle atr3, c_1\rangle \rightarrow B$	7	1	-
$\langle atr3, c_2\rangle \rightarrow A$	8	1	-
$\langle atr3, c_2\rangle \rightarrow B$	5, 6	2	66.67

After the second CAR is added to classifiers, the transaction IDs associated with CAR are removed. The remaining transaction IDs are shown in Table 5. There is only one ruleitem that satisfies the minimum support threshold: the ruleitem $\langle atr2, b_3\rangle \rightarrow A$ which does not meet 100% of confidence. No ruleitem passes the minimum support threshold to be extended with the ruleitem $\langle atr2, b_3\rangle \rightarrow A$ so CAR generation is stopped.

Table 5. Transaction IDs after generating the second CAR.

Ruleitem	TIDs	Sup	Conf (%)
$\langle atr1, a_2\rangle \rightarrow A$	9	1	-
$\langle atr1, a_2\rangle \rightarrow B$	7	1	-
$\langle atr1, a_3\rangle \rightarrow A$	8	1	-
$\langle atr2, b_3\rangle \rightarrow A$	8, 9	2	66.67
$\langle atr2, b_3\rangle \rightarrow B$	7	1	-
$\langle atr3, c_1\rangle \rightarrow A$	9	1	-
$\langle atr3, c_1\rangle \rightarrow B$	7	1	-
$\langle atr3, c_2\rangle \rightarrow A$	8	1	-

With the remaining transaction IDs in Table 5, the ECARG algorithm continues to find a default class and to add it to the classifier. In this step, the class with the most relevant transaction IDs is selected as the default class. In Table 5, class A remains in transaction IDs 8 and 9 while class B remains in transaction ID 7. The remaining transaction IDs are relevant to class A the most, so the default class is A. In case the number of associated remaining transaction IDs with each class is not changed, the majority class in the classifier is the default class. Finally, all CARs in the classifier are shown in Table 6.

Table 6. All CARs from ECARG.

CAR ID	CAR
R1	$\langle atr1, a_1\rangle \rightarrow A$
R2	$\langle (atr1, a_2), (attr3, c_2)\rangle \rightarrow B$
Default Class	A

To observe the effect of 100% confidence ruleitems, we tested another version of ECARG, ECARG2. The difference in ECARG2 is ruleitem extension. If a ruleitem with 100% confidence cannot be found from the extension, the ruleitem with the highest confidence will be selected as a CAR and added to classifiers. For example, in Table 5, ruleitem $\langle atr2, b_3\rangle \rightarrow A$ is the only ruleitem that satisfies the minimum support and minimum confidence. Hence, ECARG2 selects the ruleitem as the third CAR. The associated transaction IDs are removed, and the remaining transaction ID is shown in Table 7. There is only one transaction ID with class B. Consequently, the default class is B. Finally, all CARs from ECARG2 are shown in Table 8.

Table 7. Transaction IDs after ECARG2 generated the third CAR.

Rule Item	TIDs	Sup	Conf (%)
$\langle atr1, a_2\rangle \rightarrow B$	7	1	-
$\langle atr2, b_3\rangle \rightarrow B$	7	1	-
$\langle atr3, c_1\rangle \rightarrow B$	7	1	-

Table 8. All CARs from ECARG2.

CAR ID	CAR
R1	$\langle atr1, a_1 \rangle \rightarrow A$
R2	$\langle (atr1, a_2), (attr3, c_2) \rangle \rightarrow B$
R3	$\langle atr2, b_3 \rangle \rightarrow A$
Default Class	B

5. Experimental Setting and Result

The experiments were implemented and tested on a system with the following environment: Intel Core i3-6100u 2.3 GHz processor with 8 GB DDR4 main memory, running Microsoft Windows 10 64-bit version. Our algorithm is compared with the well-known algorithms CBA, CMAR, and FACA. All algorithms were implemented in java. The implementing java version of the CBA algorithm using CR-tree is from WEKA [26]. The implementation of CMAR in JAVA is from [27]. Four algorithms are tested on 14 datasets from the UCI Machine Learning Repository. The characteristics of the datasets are shown in Table 9. Ten-fold cross-validation is used to divide testing instances and training instances based on previous works [12,17,23,26,27]. Accuracy rates, the number of CARs, classifier building times, and memory consumption are used to measure the performance of the four algorithms.

Table 9. Characteristics of the experiment datasets.

Data Sets	# of Attributes	# of Classes	Instances
Anneal	38	6	798
Breast	11	2	699
Cars	6	4	1,728
Contact-lenses	4	3	24
Diabetes	7	2	768
Iris	4	3	150
Labor	17	2	57
Lymph	18	4	148
Mushroom	22	2	8214
Post-operative	9	4	90
Tic-tac-toe	9	2	958
Vote	16	2	435
Wined	13	3	178
Zoo	17	7	101

To study the sensitivity of thresholds on the ECARG algorithm, we set different minimum support thresholds and different minimum confidence thresholds in the experiment. First, we set the minimum support thresholds from 1% to 4% and analyze different minimum confidence thresholds between 60%, 70%, 80%, and 90%. Figure 1 shows the accuracy rates of all datasets. The results show that, when the minimum support thresholds are increased, the accuracy rates are decreased. If the minimum confidence thresholds are increased, the accuracy rates are slightly down.

The highest accuracy rates are given in most datasets, Diabetes, Iris, Labor, Lymph, Mushroom, Post-operative, Tic-tac-toe, Vote, Wine, and Zoo, when minimum support and minimum confidence are set to 2% and 60%, respectively. Therefore, the minimum support is set to 2%, and minimum confidence is set to 60% in the next experiments.

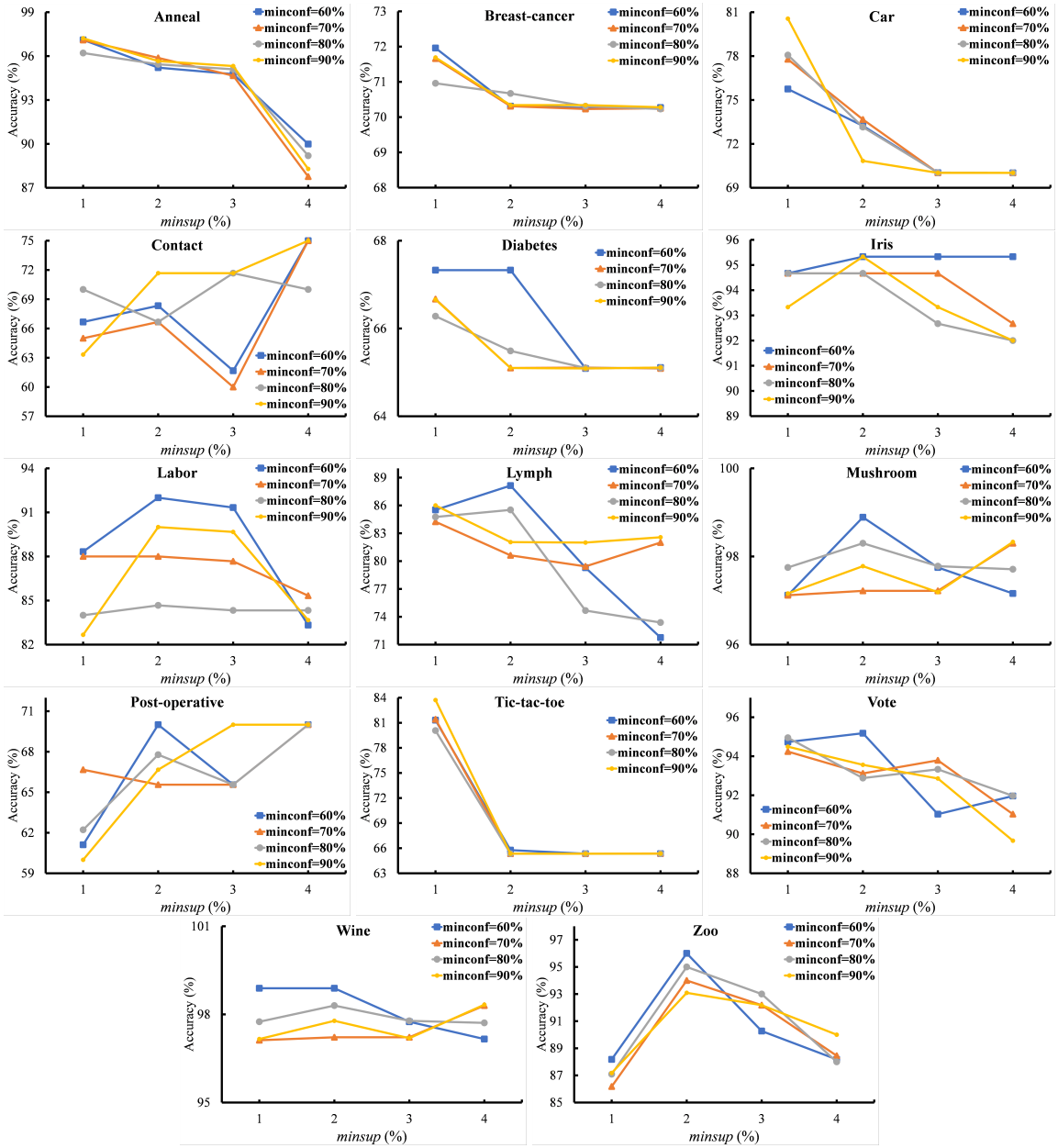

Figure 1. Accuracy rates in various *minsup* and *minconf* on all datasets.

Table 10 reports the accuracy rates of the CBA, CMAR, FACA, ECARG, and ECARG2 algorithms on the UCI datasets. The results show that both of our algorithms outperform the others on average. This gain resulting from the methodology found the most efficient rule in each iteration and eliminated redundant rules simultaneously. To be more precise, we further analyzed the win-lost-tie records. Based on Table 10, the win-lost-tie records of the ECARG2 algorithm against CBA, CMAR, FACA, and ECARG in terms of accuracy are 11-3-0, 11-3-0, and 9-4-1, 8-6-0, respectively. We can observe that ECARG gives an accuracy slightly less than ECARG2. However, the ECARG algorithm results in the highest accuracy in 6 of 14 datasets.

Table 10. Accuracies of CBA, CMAR, FACA, ECARG, and ECARG2.

Datasets	CBA	CMAR	FACA	ECARG	ECARG2
Anneal	83.19	73.27	87.31	95.21	**96.77**
Breast	67.16	**74.83**	72.44	70.33	73.02
Cars	78.29	73.73	70.02	73.43	**87.79**
Contact	66.67	37.5	63.33	**70.83**	65.00
Diabetes	**74.47**	57.03	73.56	67.32	73.7
Iris	92.67	**97.33**	96.00	95.33	96.00
Labor	75.67	26.32	87.67	**92.67**	84.00
Lymph	77.76	43.24	82.43	**88.51**	81.81
Mushroom	93.40	86.25	96.52	98.15	**98.9**
Post-oper.	56.67	**70.00**	67.78	**70.00**	60.00
Tic-tac-toe	**99.16**	53.03	90.23	65.34	88.94
Vote	94.02	92.64	91.92	**95.31**	95.17
Wine	89.97	62.92	92.16	**98.87**	97.16
Zoo	60.27	79.21	86.00	95.00	**96.00**
Average	79.24	66.24	82.67	84.02	**84.42**

Table 11 shows the average number of CARs generated from CBA, CMAR, FACA, ECARG, and ECARG2 algorithms. The result shows that the CMAR algorithm generates the highest number of rules, while the ECARG algorithm generates the lowest. In particular, the ECARG algorithm generates 8 CARs on average against 14 datasets whereas the CBA, CMAR, FACA, and ECARG2 algorithms derive 19, 240, 13, and 18 CARs on average, respectively. The accomplishment of the proposed algorithm is the discovery of the most efficient CAR in each iteration and the elimination of unnecessary transaction IDs that leads to redundant CARs.

Table 11. The average number of generated rules on the UCI datasets.

Data Sets	CBA	CMAR	FACA	ECARG	ECARG2
Anneal	**3**	165	15	14	17
Breast	16	127	23	**3**	35
Cars	25	272	9	**5**	18
Contact	9	25	**5**	7	8
Diabetes	56	115	24	**4**	38
Iris	11	38	7	**4**	9
Labor	**8**	297	15	9	9
Lymph	26	465	**15**	19	20
Mushroom	**8**	28	16	12	13
Post-oper.	35	51	12	**11**	27
Tic-tac-toe	28	713	12	**6**	33
Vote	30	658	12	11	**10**
Wined	**5**	237	11	9	7
Zoo	**10**	97	11	**10**	**10**
Average	19	240	13	**8**	18

Table 12 shows the average classifier building time of the proposed algorithm against CBA, CMAR, and FACA. The experimental result clearly shows that our algorithm is the fastest among all algorithms in the 14 datasets. ECARG takes fewer seconds to construct the classifier than CBA, CMAR, FACA, and ECARG2 by 2.134, 0.307, 2.883, and 0.0162, respectively. This can be explained by the fact that CBA and FACA uses an Apriori-style approach to generate candidates. When the value for minimum support is low on large datasets, it is costly to handle a large number of candidate ruleitems. The CMAR algorithm based on FP-growth is better than CBA and FACA in some cases, but it takes more classifier-generating time than ECARG and ECARG2.

Table 12. The classifier building time in seconds.

Data Sets	CBA	CMAR	FACA	ECARG	ECARG2
Anneal	1.050	**0.098**	0.877	0.123	0.164
Breast	0.670	0.169	0.185	**0.007**	0.027
Cars	0.220	0.249	0.640	**0.057**	0.062
Contact	0.010	0.075	0.004	**0.001**	0.002
Diabetes	1.160	0.107	0.558	**0.032**	0.085
Iris	0.030	0.008	0.010	**0.004**	**0.004**
Labor	1.170	0.924	0.027	**0.005**	**0.005**
Lymph	1.320	3.782	3.700	**0.016**	**0.016**
Mushroom	25.830	**0.104**	21.500	4.049	4.128
Post-oper.	0.090	0.041	0.063	**0.008**	0.012
Tic-tac-toe	0.230	0.235	0.800	**0.101**	0.135
Vote	1.540	2.601	5.300	**0.034**	0.034
Wined	0.120	0.273	0.190	**0.007**	**0.007**
Zoo	0.900	0.005	0.047	0.020	**0.013**
Average	2.453	0.623	2.422	**0.319**	0.335

Table 13 reveals the memory consumption in the classifier building process of all 5 algorithms. The results show that ECARG consumes less memory than CBA, CMAR, FACA, and ECARG2 by 22.62 MB, 73.15 MB, 36.57 MB, and 0.98 MB, respectively. The memory consumption of ECARG is the best since it eliminates unnecessary data in each iteration. From the result in Table 14, our proposed algorithm gives a higher F-measure on average than the other algorithms. In particular, the ECARG2 outperformed CBA, CMAR, FACA, and ECARG by 3.82%, 25.38%, 25.38%, 12.74%, and 1.84%, respectively.

Table 13. The classifier building memory consumption in megabytes.

Data Sets	CBA	CMAR	FACA	ECARG	ECARG2
Anneal	73.47	29.16	10.78	**10.68**	13.38
Breast	25.44	23.96	24.4	**1.92**	3.54
Cars	60.08	21.17	8.98	**3.05**	3.76
Contact	2.65	**0.99**	1.87	1.78	1.84
Diabetes	28.08	26.74	24.61	**3.01**	7.30
Iris	4.16	2.40	1.88	**1.17**	**1.17**
Labor	18.34	420.88	124.01	**1.95**	**1.95**
Lymph	27.31	250.93	231.75	**2.86**	**2.86**
Mushroom	28.89	29.12	24.52	**24.27**	24.31
Post-oper.	15.17	8.78	16.38	**2.03**	2.61
Tic-tac-toe	31.76	62.23	12.76	**4.73**	8.44
Vote	23.57	2.65	3.13	**3.09**	3.15
Wine	20.87	175.36	59.52	**1.82**	**1.82**
Zoo	21.41	34.33	31.93	2.20	**2.13**
Average	27.23	77.76	41.18	**4.61**	5.59

Table 14. F-measure of Classification-based Association (CBA), Classification based on Multiple Association Rules (CMAR), Fast Associative Classification Algorithm (FACA), ECARG, and ECARG2.

Data Sets	CBA	CMAR	FACA	ECARG	ECARG2
Anneal	75.93	43.73	43.64	61.24	**89.28**
Breast	66.15	66.32	66.39	58.42	**68.43**
Cars	**73.85**	33.31	31.04	45.81	71.57
Contact	53.31	43.08	49.94	**71.67**	61.67
Diabetes	**74.4**	49.01	74.3	56.56	71.21
Iris	92.70	**97.98**	93.56	90.41	96.16
Labor	70.85	28.29	75.46	**88.89**	85.87
Lymph	78.83	48.14	53.63	**81.94**	72.08
Mushroom	93.75	87.61	96.52	96.58	**98.90**
Post-oper.	52.51	20.59	**56.00**	54.45	39.57
Tic-tac-toe	**98.90**	43.88	64.40	95.44	87.64
Vote	**94.95**	72.82	91.82	93.82	94.44
Wine	87.03	69.14	92.47	**98.65**	94.37
Zoo	54.61	62.00	53.73	**91.76**	89.37
Average	76.27	54.71	67.35	78.25	**80.09**

Table 15 shows standard deviations of accuracy rate, the number of generated rules, building times, memory consumption, and F-measure of ECARG. The standard deviation values of building time and memory consumption are low and show that the building time and memory consumption in each fold is approximately marginal. The standard deviation values of the number of generated rules are relevant.

The standard deviation values of accuracy rates and F-measure show that the values of accuracy rates and F-measure in each fold are marginally different on almost all datasets. However, when evaluating the small datasets, Contact-lenses, Labor, Lymph, and Post-operative, the standard deviation values are high because 10-fold cross-validation splits a very small testing set that can potentially affect the efficiency of the classifier. For example, the Contact-lenses dataset composes only 2 or 3 transactions in each testing set. Consequently, only one false classification occurs in the testing set and then reduces the accuracy rate dramatically.

Table 15. Standard deviations of ECARG.

Data Sets	Accuracy		# of Rules		Building Time		Memory		F-1	
	AVG	S.D.	AVG	S.D.	AVG	S.D.	AVG	S.D.	AVG	S.D.
Anneal	95.21	2.10	14	0.94	0.123	0.05	10.68	0.03	61.24	6.98
Breast	70.33	5.10	3	1.40	0.007	0.01	1.92	0.02	58.42	1.78
Car	73.43	6.11	5	0.82	0.057	0.05	3.05	0.00	45.81	7.16
Contact	70.83	28.81	7	0.92	0.001	0.00	1.78	0.02	71.67	30.54
Diabetes	67.32	6.75	4	1.34	0.032	0.01	3.01	0.03	56.56	3.41
Iris	95.33	5.44	4	0.52	0.004	0.00	1.17	0.00	90.41	5.48
Labor	92.67	14.05	9	1.26	0.005	0.00	1.95	0.02	88.89	13.72
Lymph	88.51	10.00	19	3.37	0.016	0.00	2.86	0.00	81.94	15.5
Mushroom	98.15	0.32	12	0.00	4.049	0.64	24.27	0.03	96.58	0.31
Post-oper	70.00	17.41	11	3.34	0.008	0.00	2.03	0.02	54.45	14.21
Tic-tac-toe	65.34	6.16	6	2.13	0.101	0.06	4.73	0.06	95.44	3.85
Vote	95.31	2.72	11	2.26	0.034	0.01	3.09	0.03	93.82	2.84
Wined	98.87	2.34	9	0.53	0.007	0.00	1.82	0.03	98.65	2.31
Zoo	95.00	6.99	10	0.70	0.020	0.01	2.20	0.03	91.76	13.65
Average	84.02	8.16	8	1.395	0.320	0.06	4.61	0.02	78.25	8.70

From the experimental results, the ECARG algorithm outperforms CBA, CMAR, and FACA in terms of accuracy rate and the number of generated rules. A key achievement of the ECARG algorithm is that the technique generates valid rules with 100% confidence to build classifiers. The high confidence demonstrates the high possibility of class occurrences occurring in an itemset. Therefore, the ECARG algorithm produces a small classifier but gives high accuracy. While the CBA, CMAR, and FACA algorithms build classifiers from CARs that meet the minimum confidence threshold, some of the CARs have low confidences so they may predict incorrect classese and then the accuracies of CBA, CMAR, and FACA are lower than the proposed algorithm in the most dataset.

Moreover, ECARG outperforms the others in terms of building time and memory consumption. This key achievement applies simple set theories, i.e., intersection and set difference, processing on vertical data, which can potentially reduce time and memory consumption. Furthermore, the search space can be reduced as unnecessary transactions are eliminated in each stage and, therefore, the classifier building time is minimized.

6. Conclusions

This paper proposes algorithms to enhanced associative classification. Unlike the traditional algorithms, the proposed algorithms do not need a sorting and pruning process. Candidate generation is carried out by attempting to select a first general rule with the highest accuracy. Moreover, a search space is reduced early by cutting down items with low statistical significance. Furthermore, a vertical

data format, intersection, and set difference methods are applied to calculate support and confidence and to remove unnecessary transaction IDs, decreasing computation time and memory consumption.

The experiments were conducted on 14 UCI datasets. The experimental results show that the ECARG algorithm outperforms the CBA, CMAR, and FACA algorithms in terms of accuracy by 4.78%, 17.79%, and 1.35%, respectively. Furthermore, ECARG generates smaller rules than the other algorithms in almost all datasets. In addition, ECARG results in the most optimal classifier-generating time and memory usage on average. We can conclude that the proposed algorithm gives a compact classifier with a high accuracy rate, improves computation time, and reduces memory usage.

However, the ECARG algorithm does not well perform on imbalanced datasets, such as Breast, Car, Diabetes, and Post-operative. This is because the ECARG algorithm tends to find 100% confidence CARs and to eliminate unnecessary transactions. Therefore, ruleitems belonging to minority classes will not meet the minimum support threshold or 100% confidence and they are eliminated accordingly. Consequently, the classifier cannot classify the minority class correctly.

Author Contributions: Methodology, C.T.; supervision, P.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was financially supported by Mahasarakham University (Grant year 2021).

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

References

1. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7370–7377.
2. Jukic, S.; Saracevic, M.; Subasi, A.; Kevric, J. Comparison of Ensemble Machine Learning Methods for Automated Classification of Focal and Non-Focal Epileptic EEG Signals. *Mathematics* **2020**, *8*, 1481. [CrossRef]
3. Adamović, S.; Miškovic, V.; Maček, N.; Milosavljević, M.; Šarac, M.; Saračević, M.; Gnjatović, M. An efficient novel approach for iris recognition based on stylometric features and machine learning techniques. *Future Gener. Comput. Syst.* **2020**, *107*, 144–157. [CrossRef]
4. Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S.A.; Binder, A.; Müller, E.; Kloft, M. Deep one-class classification. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 4393–4402.
5. Liu, B.; Yiming, M.; Hsu, W. Integrating Classification and Association Rule Mining. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 27–31 August 1998.
6. Abdelhamid, N. Multi-label rules for phishing classification. *Appl. Comput. Inform.* **2015**, *11*, 29–46. [CrossRef]
7. Abdelhamid, N.; Ayesh, A.; Thabtah, F. Phishing detection based associative classification data mining. *Expert Syst. Appl.* **2014**, *41*, 5948–5959. [CrossRef]
8. Jabbar, M.; Deekshatulu, B.; Chandra, P. Heart Disease Prediction System using Associative Classification and Genetic Algorithm. *arXiv* **2013**, arXiv: 1303.5919.
9. Singh, J.; Kamra, A.; Singh, H. Prediction of heart diseases using associative classification. In Proceedings of the 5th International Conference on Wireless Networks and Embedded Systems (WECON), Rajpura, India, 14–16 October 2016; pp. 1–7. [CrossRef]
10. Wang, D. Analysis and detection of low quality information in social networks. In Proceedings of the 2014 IEEE 30th International Conference on Data Engineering Workshops, Chicago, IL, USA, 31 March–4 April 2014; pp. 350–354. [CrossRef]
11. Hadi, W.; Aburub, F.; Alhawari, S. A new fast associative classification algorithm for detecting phishing websites. *Appl. Soft Comput.* **2016**, *48*, 729–734. [CrossRef]
12. Hadi, W.; Issa, G.; Ishtaiwi, A. ACPRISM: Associative classification based on PRISM algorithm. *Inf. Sci.* **2017**, *417*, 287–300. [CrossRef]

13. Rajab, K.D. New Associative Classification Method Based on Rule Pruning for Classification of Datasets. *IEEE Access* **2019**, *7*, 157783–157795. [CrossRef]
14. Nguyen, L.; Nguyen, N.T. An improved algorithm for mining class association rules using the difference of Obidsets. *Expert Syst. Appl.* **2015**, *42*, 4361–4369. [CrossRef]
15. Song, K.; Lee, K. Predictability-based collective class association rule mining. *Expert Syst. Appl.* **2017**, *79*, 1–7. [CrossRef]
16. Alwidian, J.; Hammo, B.H.; Obeid, N. WCBA: Weighted classification based on association rules algorithm for breast cancer disease. *Appl. Soft Comput.* **2018**, *62*, 536–549. [CrossRef]
17. Alwidian, J.; Hammo, B.; Obeid, N. FCBA: Fast Classification Based on Association Rules Algorithm. *Int. J. Comput. Sci. Netw. Secur.* **2016**, *16*, 117.
18. Abdelhamid, N.; Jabbar, A.A.; Thabtah, F. Associative classification common research challenges. In Proceedings of the 2016 45th International Conference on Parallel Processing Workshops (ICPPW), Philadelphia, PA, USA, 16–19 August 2016; pp. 432–437.
19. Ogihara, Z.P.; Zaki, M.; Parthasarathy, S.; Ogihara, M.; Li, W. New algorithms for fast discovery of association rules. In Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Newport Beach, CA, USA, 14–17 August 1997.
20. Agrawal, R.; Srikant, R. Fast algorithms for mining association rules. In Proceedings of the 20th International Conference Very Large Data Bases, VLDB, Santiago, Chile, 12–15 September 1994; Volume 1215, pp. 487–499.
21. Quinlan, J. *C4.5: Programs for Machine Learning*; Morgan Kaufmann Publisher, Inc.: Los Altos, CA, USA, 1993.
22. Li, W.; Han, J.; Pei, J. CMAR: Accurate and efficient classification based on multiple class-association rules. In Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, CA, USA, 29 November–2 December 2001; pp. 369–376.
23. Thabtah, F.; Cowling, P.; Peng, Y. MCAR: Multi-class classification based on association rule. In Proceedings of the 3rd ACS/IEEE International Conference on Computer Systems and Applications, Cairo, Egypt, 6 January 2005. [CrossRef]
24. Zaki, M.; Gouda, K. Fast Vertical Mining Using Diffsets. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 24–27 August 2003; ACM: New York, NY, USA, 2003; pp. 326–335. [CrossRef]
25. Hadi, W. ECAR: A new enhanced class association rule. *Adv. Comput. Sci. Technol.* **2015**, *8*, 43–52.
26. Mutter, S. Class JCBA. 2013. Available online: https://github.com/bnjmn/weka (accessed on 30 September 2018).
27. Padillo, F.; Luna, J.M.; Ventura, S. LAC: Library for associative classification. *Knowl.-Based Syst.* **2019**, *193*, 105432. [CrossRef]