Computer Science and Information Systems 18 (3):791–811 https://doi.org/10.2298/CSIS200430037M

Distance based Clustering of Class Association Rules to Build a Compact, Accurate and Descriptive Classifier

Jamolbek Mattiev[1,3] and Branko Kavšek[1,2]

[1] University of Primorska,
Glagoljaška 8, 6000 Koper, Slovenia
jamolbek.mattiev@famnit.upr.si, branko.kavsek@upr.si
[2] Jožef Stefan Institute,
Jamovacesta 39, 1000 Ljubljana, Slovenia
branko.kavsek@ijs.si
[3] Urgench State University,
Khamid Alimdjan 14, 220100 Urgench, Uzbekistan
jamolbek_1992@mail.ru

Abstract. Huge amounts of data are being collected and analyzed nowadays. By using the popular rule-learning algorithms, the number of rule discovered on those "big" datasets can easily exceed thousands. To produce compact, understandable and accurate classifiers, such rules have to be grouped and pruned, so that only a reasonable number of them are presented to the end user for inspection and further analysis.

In this paper, we propose new methods that are able to reduce the number of class association rules produced by "classical" class association rule classifiers, while maintaining an accurate classification model that is comparable to the ones generated by state-of-the-art classification algorithms. More precisely, we propose new associative classifiers, called DC, DDC and CDC, that use distance-based agglomerative hierarchical clustering as a post-processing step to reduce the number of its rules, and in the rule-selection step, we use different strategies (based on database coverage and cluster center) for each algorithm. Experimental results performed on selected datasets from the UCI ML repository show that our classifiers are able to learn classifiers containing significantly fewer rules than state-of-the-art rule learning algorithms on datasets with a larger number of examples. On the other hand, the classification accuracy of the proposed classifiers is not significantly different from state-of-the-art rule-learners on most of the datasets.

Keywords: Frequent Itemset, Class Association Rules (CAR), Associative Classification, Agglomerative Clustering.

1. Introduction

Huge amounts of data are being collected and stored nowadays in many world applications. Mining association rules from those datasets and reducing is becoming a popular and important knowledge discovery technique [1]. A huge number of rules are being discovered in "real-life" datasets that will lead to combinatorial complexity. To overcome this problem, rules have to be pruned and clustered while the compact,

accurate and understandable classifier (model) is being built to reduce the number of rules.

Association rule (AR) mining [2] aims to generate all existing rules in the database that satisfy some user-defined minimum support and confidence thresholds, while classification rule mining tries to extract a small subset of rules to form accurate and efficient models to predict the class label of unknown objects. Associative Classification (AC) is a combination of these two important data mining techniques, namely, classification and association rule mining [3]. Recently, researchers have proposed several associative classification methods [4-11] that aim to build accurate and efficient classifiers based on association rules. Research studies prove that AC methods could achieve higher accuracy than some of the traditional classification methods, although the efficiency of AC methods depends on the user-defined parameters such as minimum support and confidence. Other important approaches are clustering methods (unsupervised learning) studied in [12-14]. These clustering techniques are split into two main parts: partitional and hierarchical clustering. In partitional clustering [15,16], objects are grouped into disjoint clusters such that objects in the same cluster are more similar to each other than objects in another cluster. Hierarchical clustering [17], on the other hand, is a nested sequence of partitions. In the bottom-up method, larger clusters are built by merging smaller clusters, while the top-down method starts with the one cluster containing all objects and divides into smaller clusters.

In this research work, we propose new associative classification methods based on hierarchical agglomerative clustering (complete linkage). We define the new normalized distance metrics based on direct and indirect measures to measure the similarities between CARs, which we later use to cluster CARs in a bottom-up hierarchical agglomerative fashion (firstly, we group the class association rules based on their class label and then rules that are in the same group are clustered together). Once we cluster the rules, the natural number of clusters is identified for each group of CARs by cutting the dendrogram from the point that achieves the maximum difference between two consecutive cluster heights.

Once CARs are clustered, we define a "representative" CAR within each cluster. We propose two methods of extracting the "representative" CAR for each cluster, (1) we choose the CAR based on database coverage and (2) based on cluster center.

We have performed experiments on 14 selected datasets from the UCI Machine Learning Database Repository [18] and compared the performance of our proposed methods with the 8 most popular associative and classical classification algorithms (Decision Table and Naïve Bayes (DTNB) [19], Decision Table (DT) [20], FURIA (FR) [21], PART (PT) [22], C4.5 [23], CBA [3], Ripple Down Rules (RDR) [24], Simple Associative Classifier (SA) [25]).

The rest of the paper is organized as follows. Section 2 highlights the related work to our research work. The problem statement and our goals are provided in section 3. Our proposed method is described in section 4. Section 5 highlights the experimental evaluation. Conclusions and future plans are given in Section 6. The Acknowledgement and References close the paper.

2. Related Work

The novelty in our proposed approach is in the way we select "strong" class association rules, how we cluster them and how we choose the "representative" class association rule for each cluster. Other related research also deals with the notion of clustering CARs, but all of them use different approaches. This section presents these related approaches to clustering CARs and emphasizes the similarities and differences related to our proposed approach.

In [26], researchers have proposed a new method to cluster the association rules by K-means (partitional) clustering algorithm. The main goal of this research is the clustering of discovered association rules to make it easy for users to choose the best rules. The algorithm is divided into 4 steps: (1) ARs generated from the frequent pattern by the "APRIORI" algorithm is extracted; (2) interestingness measures such as Lift, Cosinus, Conviction and Information Gain are computed for all rules generated in step 1; (3) a set of association rules is partitioned into disjoint clusters by using K-means algorithm; they try to cluster the rules which have the smallest similarities degree between them, and Euclidian and Degree of similarity distances are used to apply the K-means algorithm; (4) finally, they classify the group of rules from the best to the worst by using a centroid of each cluster. Our proposed method uses the hierarchical agglomerative approach for clustering CARs instead of k-means and employs a different way of selecting "strong" CARs in the first place. The distance metric used by our approach during clustering is also different.

The FURIA (Fuzzy Unordered Rule Induction Algorithm) [21] is a rule based classification method which is a modified and extended version of RIPPER [32] algorithm. FURIA learns fuzzy rules instead of conventional rules and unordered rule sets (namely a set of rules for each class in a one-vs-rest scheme) instead of rule lists. Moreover, to deal with uncovered examples, it makes use of an efficient rule stretching method. The idea is to generalize the existing rules until they cover the example.

In [33], we produced a relatively simple, descriptive and accurate classifier (J&B) by exhaustively searching the entire example space. More precisely, we select the strong class association rules according to their contribution for improving the overall coverage of the learning set. J&B has a stopping criterion in the rule-selection process based on the training dataset's coverage. In our current research, we just applied J&B method without stopping criterion in the representative CAR-selection process. Since the number of clusters (the size of the classifier) is identified with different strategy, we don't need to apply stopping criterion in this approach.

Another approach is conditional market-basket difference (CMBP) and conditional market-basket log-likelihood (CMBL) methods proposed in [27]. This approach uses a new normalized distance metric to group association rules. Based on the distances, agglomerative clustering is applied to cluster the rules. The rules are further embedded in a vector space with the use of multi-dimensional scaling and clustered using self-organizing maps. This method is very similar to ours, but we propose a new normalized distance metric based on "direct" and "combined" distances between class association rules, whereas "indirect" measures are used based on CARs support and coverage.

Mining clusters with ARs [28] is another related approach. Here the rules are first generated using the "APRIORI" algorithm, but an "indirect" distance metric (based on coverage probabilities) is later used to find the similarities between rules. Rules are then clustered using a top-down hierarchical clustering method for finding clusters in a

population of customers, where the list of products bought by the individual clients is given. Once the rules are clustered, a specific distance metric is introduced to measure the quality of the clustering.

Another interesting clustering-based approach [29] is "Tightness" which quantifies the strength of binding between the items of an association rule. The idea is that certain items in the application domain might get bound together because they are so strongly correlated that they often occur together in transactions. This tightness of binding is not covered by traditional measures like support or confidence. They build their distance function based on indirect measures, that is, the items in AR that obtain the maximum and minimum support. Our proposed methods are all utilized different distance metrics based on "direct" and "combined" measures.

The absolute market-basket difference (AMBD) approach discussed in [30] also aims to cluster the ARs. The CAR-generation part of this method is similar to ours. The procedure is similar except for the clustering part. They focus on clustering the sorted (support and confidence based) association rules with the same consequent. The key differences are in the clustering part: indirect distance metric is used to cluster the rules (the distance gives the percentage of examples in the dataset that are not covered by both rules), whereas our methods use different distance metrics.

3. Problem Definition and Contributions

We assume that a normal relational table is given with N examples (transactions). Each example is described by A distinct attributes and is classified into one of the M known classes. Since our algorithm supports just attributes of a nominal type (like the vast majority of association rule miners), we had to perform discretization on numeric attributes in some cases (the details about discretization are provided in Section 5). The goals and contributions of our proposed approach are the following:

1. Generate "strong" CARs that satisfy some user-defined minimum support and minimum confidence constraints;
2. Propose new normalized similarity measures based on the "direct" and "indirect" distances between two class association rules;
3. Cluster class association rules by using this normalized similarity measure and automatically determine the optimal number of clusters for each class value;
4. Define two methods of extracting a representative CAR for each cluster to produce the final, compact and meaningful classifier;
5. Experimentally, show the usefulness of our methods in reducing the number of CARs, while retaining the classifier's accuracy.

4. Proposed Method

Our approach (Compact, Accurate and Descriptive Associative Classifier) can be divided into 4 steps mentioned in the previous section. Each of these 4 steps is presented in detail in the following 4 subsections.

4.1. Generating the Strong Class Association Rules

We discuss how to discover the strong CARs from frequent itemsets in this subsection. Generation of ARs is usually split up into two main steps: first, minimum support is applied to find all frequent itemsets from the training dataset; second, we use these frequent itemsets and minimum confidence to generate strong association rules. Discovering of CARs is also followed to the same procedure as in AR-generation. The only difference is that in the rule-generation part, the consequence of the rule contains only the class label in CAR-generation while the consequence of rule in AR-generation can include any frequent itemset.

In the first step, the "APRIORI" algorithm is used to find the frequent itemsets. The 'downward-closure' technique is used in the "APRIORI" algorithm to accelerate the searching procedure by reducing the number of candidate itemsets at any level. The "APRIORI" finds the 1-frequent itemset, then, the 1-frequent itemset is used to generate the 2-frequent itemset and so on. If it finds any infrequent itemsets at any level, it is removed in place, because infrequent itemsets cannot generate frequent itemset. The "APRIORI" performs this process before computing their support at any level to reduce the time complexity of the algorithm.

After all frequent itemsets are generated from the training datasets, it is straightforward to generate the strong CARs that satisfy both minimum support and minimum confidence constraints from frequent itemsets found in the first step. Confidence of the rule can be computed by the following formula:

$$confidence(A \rightarrow B) = \frac{support\ \_count\ (A \cup B)}{support\ \_count\ (A)}. \quad (1)$$

The equation (1) is expressed by support count of itemset, where A is a premise (itemset in the left-hand side of the rule), B is a consequence (class label in the right-hand side of the rule), $support\_count(A \cup B)$ is the number of transactions that matches the itemsets $A \cup B$, and $support\_count(A)$ is the number of transactions that matches the itemsets A. Strong class association rules that satisfy the minimum confidence threshold can be generated based on the above equation, as follows:

- All nonempty subsets S are generated for each frequent itemset L and a class label C;
- For every nonempty subset S of L, output the strong rule R in the form of "$S \rightarrow C$" if, $\frac{support\ \_count\ (R)}{support\ \_count\ (S)} \geq min\_conf$, where min_conf is the minimum confidence threshold.

4.2. Distance Metrics

Once we generate strong class association rules in 4.1, our next goal is to cluster them. Since we intend to apply hierarchical agglomerative clustering, we must define a way of measuring the similarity between CARs, that is, how far two rules are apart. Unfortunately, there is not any distance metric for CARs. However, researchers have proposed some indirect distance metrics for association rules, namely, Absolute Market

Basket Difference (AMBD), Conditional Market-basket Probability (CMBP), and Conditional Market-basket Log-likelihood (CMBL) [27] and "Tightness" [29].

Indirect Distance Metrics

We highlight the indirect distance metric for association rules in this section. We call rule distances that are obtained from the data Indirect Distance Metrics. An indirect distance is defined as a function of the market-basket sets that support the two considered rules.

To begin with a simple distance measure (AMBD) introduced to compute the similarity between association rules. Let $rule1: A \Rightarrow C$ and $rule2: B \Rightarrow C$ be two association rules, the distance is defined between rules in terms of the number of market-baskets that they differ in (meaning one rule is supported, but not the other). Based on the number of non-overlapping market-baskets, a distance metric d^{AMBD} between *rule1* and *rule2* can be defined by the following equation:

$$d_{rule\,1,rule\,2}{}^{AMBD} = |m(BS_{rule\,1})| + |m(BS_{rule\,2})| - 2 * |m(BS_{rule\,1}, BS_{rule\,2})| \qquad (2)$$

Where, *BS* is the both side of the rule, that is, the itemset for the association rule. *m(BS)* denotes the set of transactions (baskets) matched by BS and |m(BS)| is the number of such transactions.

Equation (2) illustrates that rules valid for exactly the same baskets have a distance of zero. Rules applying to disjoint sets of baskets have a distance equal to the sum of the numbers of transactions for which each rule is valid. There are several problems with this measure. One such problem is that it grows as the number of market-baskets in the database increases. This can be corrected by normalizing (dividing the measure by the size of the database) and it is appropriate for rules only with the same consequent while this approach is intuitive. However, the measure is still strongly correlated with support. High support rules will on average tend to have higher distances to everybody else. This is an undesired property. For example, two pairs of rules, both pairs consisting of non-overlapping rules, may have different distances. High support pairs have a higher distance than low support pairs.

Based on the previous approach, another indirect distance measure in the CMBP method is proposed as an improvement of AMBD using the support values of two association rules. Researchers tried to solve two problems that occurred in AMBD: (1) it grows as the database grows, and (2) due to the focus on support values, rules with high support will on average tend to have higher distances to everybody else. To solve the above-mentioned problems, they proposed the new indirect distance metric based on conditional probabilities. Using a probability estimate for distance computation has many advantages. Probabilities are well understood, are intuitive, and a good measure for further processing. The distance d^{CMBP} between two rules *rule1* and *rule2* is the (estimated) probability that one rule does not hold for a basket, given at least one rule holds for the same baskets. This distance is defined as follows:

$$d^{CMBP}_{rule\,1,rule\,2} = 1 - \frac{|m(BS_{rule\,1}, BS_{rule\,2})|}{|m(BS_{rule\,1})| + |m(BS_{rule\,2})| - |m(BS_{rule\,1}, BS_{rule\,2})|} \qquad (3)$$

With this metric, rules having no common market baskets are at a distance of 1, and rules valid for an identical set of baskets are at a distance of 0. The CMBP does not suffer from the support correlation problem of AMBD. Let us call a distance interesting if it is neither 0 nor 1. Rule pairs with an interesting distance are called good neighbors. In most real databases, the majority of all rule pairs are not good neighbors. Manual exploration of a rule's good neighbors showed that intuitive relatedness was captured very well by this metric. For example, rules involving different items but serving equal purposes were found to be close good neighbors. Super-set relationships of the itemsets associated with the rules often lead to very small distances.

The new "Direct" Distance Metric

We compute the distance between two rules by ignoring the class label because we are clustering the rules belonging to the same class label. When we apply indirect distance measures to our proposed method, we get a larger natural number of clusters, that is, the classifier includes a larger number of rules. Therefore, we propose a new normalized Item Based Distance Metric (IBDM) in this research work by considering the differences in rule items.

Let $R = \{r_1, r_2, \ldots., r_n\}$ be a set of class association rules found from relational dataset D that are defined by $A = \{a_1, a_2, \ldots., a_m\}$ distinct items (attribute's value) classified into $C = \{c_1, c_2, \ldots., c_l\}$ known classes. Each rule is denoted as follows:
$r = \{x_1, x_2, \ldots., x_k\} \rightarrow \{c\}$ where, $\{x_1, x_2, \ldots., x_k\} \subseteq A$ and $c \in C$ for $\forall\, r \in R$.
Given two rules $rule1, rule2 \in R$:

$$rule1 = \{y_1, y_2, \ldots., y_s\} \rightarrow \{c\}$$
$$rule2 = \{z_1, z_2, \ldots., z_t\} \rightarrow \{c\}$$

Where$\{y_1, y_2, \ldots., y_s\} \subseteq .A$, $\{z_1, z_2, \ldots., z_t\} \subseteq .A$, and $c \in C$. We compute the similarity between $rule1$ and $rule2$ as follows:

$$distance_q(rule1, rule2) = \begin{cases} if\ y_q = z_q \mid y_q = \emptyset\ \&\ z_q = \emptyset\ , 0; \\ else\ if\ y_q = \emptyset\ \&\ z_q \neq \emptyset \mid y_q \neq \emptyset\ \&\ z_q = \emptyset\ , 1; \\ else\ \ 2\ (y_q \neq z_q). \end{cases} \quad (4)$$

Where q is the index of rule items that cannot exceed from *border* value (defined below).

Equation (4) expresses how close two rules are one from another. If rules have similar items, then the distance function has a low value. An empty rule item is considered closer than a different rule item.

$$border = Max(s, t); \quad (5)$$

border is the length of the longest rule, (5) is used to normalize the distance metric. The distance between two rules is denoted as follows:

$$d^{IBDM}_{rule\,1, rule\,2} = \sum_{i=1}^{border} distance_i / 2 \times border \quad (6)$$

Distance (6) ranges between 0 and 1. CARs having the same items and the same size are at a distance of 0, CARs containing the different items are at a distance of 1.

The new "Combined" Distance Metric

Since conditional market-basket distance is appropriate for the rules having the same consequent, we decided to propose a new Weighted and Combined Distance Metric (WCDM) by combining direct (IBDM) and indirect distance (CMBP) measures. When we apply CMBP distance to our proposed method, we got a larger number of clusters on some datasets. WCDM combines direct measure (rule items) and indirect measure (rule coverage). Both distance metrics (IBDM and WCDM) have their advantage: on some datasets IBDM produces lower number of rules with higher accuracy while WCDM achieves better results on some other datasets. The weighted distance d^{WCDM} between two rules *rule1* and *rule2* is defined as follows:

$$d^{WCDM}_{rule\,1,rule\,2} = \alpha \times d^{IBDM}_{rule\,1,rule\,2} + (1-\alpha) \times d^{CMBP}_{rule\,1,rule\,2} \quad (7)$$

where, α is a weight parameter. We set $\alpha = 0.5$ parameter, the final weighted and combined distance measure is described as follows:

$$d^{WCDM}_{rule\,1,rule\,2} = 0.5 \times d^{IBDM}_{rule\,1,rule\,2} + 0.5 \times d^{CMBP}_{rule\,1,rule\,2}. \quad (8)$$

4.3. Clustering

Clustering algorithms aim to group similar examples; the examples in the same cluster should be similar and dissimilar to the examples in other clusters. There are two types of hierarchical clustering algorithms: top-down (*divisivehierarchical clustering*) and bottom-up (*hierarchical agglomerative clustering*). Bottom-up algorithms initially assume each example as a single cluster and then merge the two closest clusters in every iteration until all clusters have been merged into a unique cluster that contains all examples. The resulting hierarchy of clusters is represented as a tree (or dendrogram). The root of the tree is the unique cluster that gathers all the examples; the leaves are considered as clusters with only one sample. The top-down approach is the opposite of the hierarchical agglomerative clustering method. It considers all examples in a single cluster, and then it splits the clusters into smaller parts until each example forms a cluster or until it satisfies the stopping condition.

We apply the complete linkage method ofhierarchical agglomerative clustering. In the complete linkage (farthest neighbor) method, the similarity of two clusters is the similarity of their most dissimilar examples, therefore, the distance between the farthest groups are taken as an intra-cluster distance. We assume that we have given $N \times N$ distance matrix *D*, where *N* is the total number of rules (that is, total number of clusters). The clusters are numbered 0,1,..,(*N*-1) and *m* is the sequence number of clusters. *L[k]* is the level of the *k*-th clustering and the distance between two clusters *cl1* and *cl2* is defined as *D[cl1,cl2]*. Complete linkage of the hierarchical agglomerative clustering algorithm is outlined in algorithm 1.

We need to apply hierarchical clustering algorithm twice: first, we apply *AHCCLH* algorithm to find the cluster heights that we will use later to identify the optimal number of clusters. In this case, number of cluster *S=1* and distance matrix are defined as input parameters. Because if *S=1*, then, *AHCCLH* iterates *N-1* times to find the heights of all the clusters. Second, *AHCCLC* is utilized to identify the cluster of class association

rules. In *AHCCLC*, we provide the number of cluster *S* (found by using the cluster heights) and distance matrixto identify the clusters of CARs.

Algorithm 1: Agglomerative Hierarchical Clustering with Complete Linkage (AHCCLH: Heights || AHCCLC: Clusters)

Input: a distance matrix *D* and number of clusters *S*
Output: Cluster heights (AHCCLH), Cluster of CARs (AHCCLC)

1. **Initialization:** Each rule is a unique cluster *C* at level 0 *(L[0]=0*), sequence number *m*=0 and the optimal number of cluster *S* is identified, so, to get the intended number of clusters (*S*), the algorithm should iterate *K* times *K=N–S*;
2. **Compute:** Find the most similar pair of clusters, *cl1* and *cl2*and merge them into a single cluster *C* to form the next clustering sequence *m*.Increase the sequence number by one: *m=m*+1 and set the new level *L[m]=D[cl1,cl2]*;
3. **Update:** Update the distance matrix *D*, by removing the rows and columns corresponding to *cl1* and *cl2* and adding a new row and column corresponding to the new cluster.The distance between the new cluster (*cl1*,*cl2*) and old cluster *k* is calculated as *D[(cl1, cl2),k]=max{D[k,cl1], D[k,cl2]}*;
4. **Stopping condition:** if *m=K* then **return***L* (*AHCCLH*) || *C* (*AHCCLC*)and stop, otherwise go to step 2.

When we cluster the rules, we need to find the number of clusters. We get the "natural" number of clusters by "cutting" the dendrogram at the point that represents the maximum distance between two consecutive cluster merges. The algorithm that identifies the "natural" number of clusters is presented in Algorithm 2.

Algorithm 2: Computing the optimal number of clusters

Input: an array of cluster heights
Output: Optimal number of cluster

```
1:   Max_height_difference=cluster_height[1]-cluster_height[0];
2:   Opt_number_of_cluster= 1;
3:   N=cluster_height.length;
4:   for (k=2; k≤ N; k++) do begin
5:      if (cluster_height[k]-cluster_height[k-1])>Max_height_difference then
6:         Max_height_difference= cluster_height[k]-cluster_height[k-1];
7:          Opt_number_of_cluster=N-k;
8:      end if
9:   end for
10:  return Opt_number_of_cluster
```

The input to Algorithm 2 is a set of cluster distances that are calculated during the building of the dendrogram (so-called cluster "heights"). The output is the "natural" number of clusters. In lines 1-3 the total number of clusters generated by hierarchical clustering is stored. Lines 4-7 outline the main part of the algorithm, *Opt_number_of_clusters*gets to the point where the difference between two consecutive cluster heights will be maximum. Since we start from *0*, *Opt_number_of_clusters*is equal to *(N-k)*. The last line returns the obtained result.

4.4. Extracting the Representative CAR

Once we found all clusters, our final goal is to extract the representative CAR for each cluster to form our meaningful, compact and descriptive associative classifier. In this research work, we propose two methods of extracting the representative CAR for each cluster.

Representative CAR based on Cluster Center

In this method, we choose the CAR which is closer to the center of the cluster as a representative, that is, the representative CAR must have the minimum average distance to all other rules. Algorithm 3 defines the procedure.

Algorithm 3: A Representative CAR based on Cluster Center (RCC)

```
   Input: a set of class association rules in CARs array
  Output: Arepresentative class association rule
1:    N=CARs.length;
2:   Fill(Dist, 0);
3:   min_avg_distance=Integer.Max.value;
4:   for (i=0; i≤ N;i++) do begin
5:      for (j=0; j≤ N;j++) do begin
6:        Dist[i]=Dist[i]+IBDM(CARs[i], CARs[j]) | WCDM(CARs[i], CARs[j]);
7:      end for
8:      avg_distance=Dist[i]/N;
9:      if (avg_distance<min_avg_distance then
10:        min_avg_distance= avg_distance;
11:        representative_CAR_index=i:
12:     end if
13:   end for
14:    return CARs[representative_CAR_index];
```

The first line gets the number of CARs. We use the distance array "*Dist*" (line 2) to compute the distance from the selected CAR to all other CARs (we use one of the distance measures described in section 4). Initial value of*min_avg_distance* in line 3 is the maximum value of the integer and it is used to store the minimum average distance in line 10. Lines 4-9 find the index of the representative CAR that has the minimum average distance to all other rules and the last line returns the representative CAR.

Representative CAR based on Database Coverage

We decided to propose this method to improve the overall coverage and classification accuracy, while the first method (RCC) suffers to achieve reasonable coverage on some certain datasets. Since we are clustering similar rules having the same class value, it is unnecessary to think about the outer-class overlapping problem (that means some samples from different classes have very similar characteristics), but we should avoid the inter-class overlapping problem (several rules that belong to same class may cover the same samples). We bypass this problem by selecting the representative CAR based

on database coverage. First, we find a rule that has maximum database coverage, then we check if the first CAR classifies at least one new example, then we get it as a representative CAR, otherwise we continue. Once we find the representative, we remove all the examples covered by a representative CAR. The procedure is outlined in algorithm 4.

Algorithm 4: A Representative CAR based on Database Coverage(RDC)

Input: a set of class association rules in *CARs* array, a training dataset *D* and*classified_traindata*array
Output: Arepresentative class association rule

```
1:    N=CARs.length;
2:  CARs= sort(CARs, coverage);
3:  Representative_CAR = CARs[1]:
4:  for i:=1 toN do begin
5:      for j:=1 toD.length do begin
6:          if classified_traindata[j]=false then
7:              if CARs[i] classifies D[j] (e.g. CARs[i].premise⊆D[j].premise) then
8:                      classified_traindata[j]=true;
9:                      contribution=contribution+1:
10:             end if
11:         end if
12:     end for
13:     ifcontribution>0 then
14:         Representative_CAR = CARs[i]:
15:             break;
16:     end if
17: end for
18: return Representative_CAR;
```

In this approach (RDC), we first sort (line 2) the class association rules in coverage descending order, and we start checking rules from first to last (line 4). If a rule classifies at least one new example (line 13), that is, if CAR premise (left-hand side of the rule) matches the premise of the training dataset (line 7), we return that rule as a representative (line 13-18), otherwise we continue. If any rule cannot be a representative, then, the algorithm returns the first rule (line 3) which has the highest coverage as a representative.

Finally, our proposed approach is represented in algorithm 5.

Lines 1-2 generate the strong CARs that satisfy the user-specified minimum support and minimum confidence constraints from training dataset *D* by using the "APRIORI" algorithm. The third line sorts the CARs in confidence and supports descending order according to the following criteria:

$R_1$and R_2 are two CARs, R_1 is said to have a higher rank than R_2, denoted as $R_1 > R_2$,

- If and only if, $conf(R_1) > conf(R_2)$; or
- If $conf(R_1) = conf(R_2)$ but, $supp(R_1) > supp(R_2)$; or
- If $conf(R_1) = conf(R_2)$ and $supp(R_1) = supp(R_2)$, R_1has fewer attribute values in its left-hand side than R_2does;
- If all the parameters of the rules are equal, we can choose any of them.

Algorithm 5: Learning the proposed Associative Classifier

Input: A training dataset *D*, minimum support and minimum confidence
Output: Associative classifier

```
1:    F=frequent_itemsets(D, minsup);
2:   R=genCARs(F, minconf);
3:   R=sort(R, minsup, minconf);
4:   G=Group(R);
5:    for (i=0; i≤ number_of_class;i++) do begin
6:      Distance=IBDM(G[i]) | WCDM(G[i]);
7:      Cluster_heights=AHCCLH(Distance, 1);
8:      N=optimal_number_of cluster(Cluster_heights);
9:      Cluster=AHCCLC(Distance, N);
10:     Fill(classified_traindata, false);
11:     for (j=0; j≤ N;j++) do begin
12:          Y= RDC(Cluster[i], D, classified_traindata) | RCC(Cluster[i]);
13:          Associative_Classifier.add(Y);
14:      end for
15:   end for
16:    return Associative_Classifier;
```

CARs are grouped according to their class label in line 4. For each group of CARs (lines 5-15), the distance matrix is constructed by using one of the distance measures defined in subsection 4.2 (line 6), the hierarchical clustering algorithm complete linkage method (*AHCCLH*) computes the cluster heights (distances between clusters) by using the distance matrix in line 7 and these heights (distances) are used to find the optimal number of clusters (line 8). Then, we apply the hierarchical clustering algorithm (*AHCCLC*) again to identify the clusters of CARs (a *Cluster* array stores the list of clustered CARs). Since we are clustering the class association rules class by class, we need a "*classified_traindata*" array to store the information about classified examples, that is, we update this array for the same class only. When we start the clustering of CARs for a new class, we need to initialize the "*classified_traindata*" array. In lines 11-14, the representative CAR is extracted by using one of the methods described in section 4.4 for each cluster and added to our final classifier. The last line returns the descriptive, compact and meaningful classifier. The classification process of proposed methods is shown in algorithm 6.

Algorithm 6 predicts the class label of the test example by using the classifier. The first line files the *class_count* array with 0 (the size of *class_count*array equals to the number of classes). For each rule in the classifier (line 2), if the rule can classify the example correctly, then we increase the corresponding class count by one and store it (lines 3-5).In lines 7-10, if none of the rules can classify the new example correctly, then the algorithm returns the majority class value. Otherwise, it returns the class value that is the most common among the rules that classify the test example.

We built the following different classifiers: "DC" method is built based on direct distance measure (IBDM) and the method for extracting a representative CAR is based on cluster center (RCC), "DDC" method is formed based on direct distance measure (IBDM) and the method for extracting a representative CAR is based on database coverage (RDC), and "CDC" method is formed based on combined distance measure

(WCDM) and the method for extracting a representative CAR is based on database coverage (RDC).

Algorithm 6: Classification process of our proposed approaches

Input: A Classifier and a *test_example*
Output: Predicted class

```
1:   Fill(class_count, 0);
2:   for each rule y ∈ Classifier do begin
3:     if y classify test_example then
4:       class_count[y.class]++;
5:     end if
6:   end for
7:   if max(class_count)==0 then
8:     predicted_class=majority_class;
9:   else predicted_class= max_index(class_count);
10:  end if
11:  return predicted_class
```

5. Experimental Results

We evaluated our classifiers by comparing them with 8 well-known rule-based classification algorithms on classification accuracy and the number of rules. All differences were tested for statistical significance by performing a paired t-test (with a 95% significance threshold).

Associative classifiers were run with default parameters *minimum support = 1%* and *minimum confidence = 60%* (on some datasets, however, *minimum support* was lowered to *0.5%* or even *0.1%* and confidence was lowered to *50%* to ensure "enough" CARs ("enough" means at least 5-10 rules for each class value- this situation mainly happens with imbalanced datasets) were generated for each class value). For all other 8 rule learners we used their WEKA workbench [31] implementation with default parameters. Since AR learning does not support numeric attributes, all numeric attributes (in all datasets) were pre-discretized with WEKA's "class-dependent" discretization method. The description of the datasets and input parameters are shown in Table 1.

Furthermore, all experimental results were produced by using a 10-fold cross-validation evaluation protocol.

Experimental results on classification accuracies (average values over the 10-fold cross-validation with standard deviations) are shown in Table 2.

We can observe from Table 2 that our proposed associative classifiers achieved comparable average accuracies (DC: 80.7%, DDC:81.5% and CDC:82.0% respectively) to other classification models on selected datasets. Interestingly, CDC significantly outperforms all rule-learners on the "Breast Cancer" (except DDC), "Hayes-root" and "Lymp" datasets, while on the "Car.Evn", "Nursery" and "Monks" datasets, our proposed methods obtained worse accuracy than all other algorithms (except for SA). Standard deviations of accuracy results decrease with an increasing number of examples in a dataset, which is expected behavior.

Table 1. Description ofdatasets and AC algorithm parameters

Dataset	# of attributes	# of classes	# of records	*Min support*	*Min confidence*	# of analyzed rules
Breast Can	10	2	286	1%	60%	1000
Balance	5	3	625	1%	50%	218
Car.Evn	7	4	1728	1%	50%	1000
Vote	17	2	435	1%	60%	500
Tic-Tac-Toe	10	2	958	1%	60%	3000
Nursery	9	5	12960	0.5%	50%	3000
Hayes-root	6	3	160	0.1%	50%	1000
Lymp	19	4	148	1%	60%	1500
Spect.H	23	2	267	0.5%	50%	3000
Abalone	9	3	4177	1%	60%	1000
Adult	15	2	45221	0.5%	60%	5000
Insurance	7	3	1338	1%	50%	722
Monks	7	2	432	1%	50%	800
Laptop	11	3	1303	1%	50%	1480

Table 2. Overallaccuracies with standard deviations:

Dataset	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA	DC	DDC	CDC
Breast.Can	70.4±4.1	69.2±6.7	75.0±6.9	74.0±4.0	75.1±5.3	71.8±5.7	71.9±9.8	79.3±4.4	81.2±4.0	81.9±4.1	82.6±4.5
Balance	81.4±8.1	66.7±5.0	64.4±4.3	76.2±5.6	77.5±7.6	68.5±4.3	73.2±3.8	74.0±4.1	72.8±2.4	73.2±2.9	73.2±3.0
Car.Evn	95.4±0.8	91.3±1.7	92.1±1.7	94.3±1.0	91.8±1.1	91.0±1.8	91.2±3.9	86.2±2.1	85.8±1.4	88.5±1.2	87.1±1.6
Vote	94.7±3.4	94.9±3.7	94.7±4.4	94.8±4.2	94.4±2.8	95.6±4.1	94.4±2.6	94.7±2.3	92.9±2.5	93.2±2.8	90.6±2.3
Tic-Tac-Toe	69.9±2.7	74.4±4.4	85.2±2.7	94.3±3.3	94.1±3.1	94.3±2.9	100.0±0.0	91.7±1.5	87.3±1.3	91.8±1.0	92.4±1.1
Nursery	94.0±1.5	93.6±1.2	95.4±1.4	96.7±1.7	91.0±1.4	92.5±1.5	92.1±2.4	91.6±1.2	88.5±1.1	89.3±1.1	92.3±0.9
Hayes-root	75.0±7.2	53.4±8.3	78.7±8.4	73.1±9.7	77.7±8.7	74.3±7.1	75.6±10.9	73.1±6.0	79.9±5.7	77.8±5.2	82.7±6.1
Lymp	72.9±9.0	72.2±8.3	76.2±8.7	81.7±9.0	80.0±8.2	78.3±7.3	79.0±9.7	73.7±5.1	78.4±6.7	80.0±6.1	84.0±6.4
Spect.H	79.3±2.7	79.3±1.6	80.0±9.0	80.4±5.6	80.4±2.2	80.4±2.2	79.0±1.6	79.1±2.1	81.5±0.7	81.3±1.1	82.8±1.3
Abalone	62.1±1.3	61.8±1.5	62.3±1.2	62.3±1.1	61.7±1.6	60.8±0.8	61.1±1.0	61.0±0.9	61.0±1.1	60.7±1.0	60.7±1.2
Adult	73.0±4.1	82.0±2.3	82.4±4.7	82.1±4.7	75.2±3.2	80.8±2.7	81.8±3.4	80.8±2.6	81.9±2.4	82.0±2.6	82.8±3.0
Insurance	74.2±1.1	75.7±1.6	75.8±1.4	75.0±1.8	75.8±1.4	73.4±1.7	75.5±2.0	74.5±1.6	74.0±1.1	74.2±1.1	73.2±1.5
Monks	98.9±0.9	98.9±0.9	98.9±0.9	98.9±0.9	98.9±0.9	97.1±0.7	97.8±1.4	92.1±1.3	92.5±0.8	93.6±0.9	91.1±0.8
Laptop	75.7±2.6	72.9±2.9	75.3±2.3	74.5±2.9	75.4±2.1	73.2±1.8	75.4±2.0	72.0±1.4	71.6±2.1	73.8±2.3	72.6±1.7
Average(%):	80.0±3.5	77.6±3.6	81.2±4.1	82.7±4.0	82.1±3.5	80.8±3.2	82.0±3.9	80.3±2.6	80.7±2.4	81.5±2.4	82.0±2.5

Statistically significant testing (wins/losses counts) on accuracy between DC and other classification models is shown in Table 3. **W**: our approach was significantly better than compared algorithm; **L**: selected rule-learning algorithm significantly outperformed our algorithm; **N**: no significant difference has been detected in the comparison.

Table 3. Statistically significant wins/losses counts of DC method on accuracy:

	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA	DDC	CDC
W	6	6	4	2	3	3	3	3	1	1
L	5	4	5	7	5	5	5	2	3	4
N	3	4	5	5	6	6	6	9	10	9

Table 3 illustrates that the performance of DC method on accuracy was better than DTNB, DT and SA methods. Although DC obtained similar result with C4.5 and DDC (there is no statistical difference on 10 datasets out of 14), it is eaten by all other methods according to win/losses counts. However, on average, the classification accuracies of DC are not much different from those of the other 8 rule-learners.

The same experiment on DDC is shown in Table 4. Since DC is compared with DDC in Table 3, it is not included in Table 4.

Table 4. Statistically significant wins/losses counts of DDC method on accuracy:

	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA	CDC
W	5	5	4	2	2	3	3	4	2
L	4	3	3	5	4	5	4	1	3
N	5	6	7	7	8	6	7	9	9

It can be seen from the table that DDC's performance on accuracy is better than DC. It outperformed the DTNB, DT, C4.5, SA and DC methods (by win/losses counts).

Table 5. Statistically significant wins/losses counts of CDC method on accuracy:

	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA
W	6	6	6	4	5	5	4	4
L	5	4	6	5	6	3	6	1
N	3	4	2	5	3	6	4	9

CDC achieved the statistically comparable results in terms of classification accuracy with "classical" and "associative" classification approaches. CDC statistically lost to C4.5, FR and CBA methods on 6 datasets out of 14, while it outperformed the rest of the algorithms except PT.

The comparison between our methods and other classification methods on the number of classification rules is shown in Table 6. Since DC and DDC differ in the representative CAR selection process, the number of classification rules generated by both methods stays the same. Thus, DC and DDC methods are merged in Table 6.

Experimental evaluations on the number of classification rules show that DC and DDC significantly outperforms all other rule-learners on 8 datasets out of 14 (except CDC) and it produces classifiers that have on average far fewer rules than those produced by the other 8 rule-learning methods included in the comparison.

Our proposed methods generated a reasonable smaller number of rules on bigger datasets compared to other classification methods. Even though our approaches could not achieve the best classification accuracies on "Car.Evn", "Nursery" and "Laptop" datasets, it produced the statistically smallest classifier on those datasets.

Experimental evaluations on the number of classification rules show that DC and DDC significantly outperforms all other rule-learners on 8 datasets out of 14 (except CDC) and it produces classifiers that have on average far fewer rules than those produced by the other 8 rule-learning methods included in the comparison.

Our proposed methods generated a reasonable smaller number of rules on bigger datasets compared to other classification methods. Even though our approaches could not achieve the best classification accuracies on "Car.Evn", "Nursery" and "Laptop" datasets, it produced the statistically smallest classifier on those datasets.

Table 6. Number of CARs:

Dataset	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA	DC&DDC	CDC
Breast.Can	122	22	10	20	13	13	63	20	8	9
Balance	31	35	35	27	44	22	77	45	34	79
Car.Evn	144	432	123	62	100	119	72	160	32	32
Vote	270	24	11	8	17	7	22	30	6	6
Tic-Tac-Toe	258	121	88	37	21	13	23	60	24	17
Nursery	1240	804	301	172	288	141	141	175	79	80
Hayes-root	5	8	22	14	11	10	34	45	19	80
Lymp	129	19	20	10	17	11	23	60	5	7
Spect.H	145	2	9	13	17	12	4	50	8	5
Abalone	165	60	49	71	20	57	131	155	14	14
Adult	737	1571	279	571	150	175	126	130	13	88
Insurance	23	48	21	49	22	22	84	62	18	20
Monks	12	36	14	8	12	10	40	26	14	14
Laptop	101	101	72	60	28	32	41	75	19	18
Average(%):	241	235	76	81	55	46	63	78	21	34

CDC got an unexpected larger number of rules (this is mainly imbalanced and discretized datasets) on “Hayes-root” and “Balance” datasets.

Table 7. Statistically significant wins/losses counts of DC and DDC method on rules:

	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA	CDC
W	11	11	10	10	11	9	12	14	3
L	2	2	0	3	2	4	1	0	2
N	1	1	4	1	1	1	1	0	9

Table 7 shows that C4.5 and SA methods could not produce statistically smaller classifier than DC and DDC methods on any datasets. The most importantly, DC and CDC generated statistically smaller classifiers than all other models on bigger datasets (over 1000 examples), which was our main goal in this research.

CDC statistically got the worse result than all methods on “Balance” (except CBA) and “Hayes.R” datasets in terms of classification rules.

Our main goal in proposing the DDC and CDC methods is to improve the overall coverage (shown in Table 9) and accuracy achieved by the DC method. Experimental results show that we could achieve our goal: DDC and CDC gained better average classification accuracy with 81.5% and 82% (this is still not the best result in terms of average accuracy, but 0.8% and 1.3% higher than the DC method). Average coverage of DDC (90.4%) and CDC (90.5%) increased to 6% compared to DC (84.4%). More precisely, the overall coverage of DDC and CDC was improved on 9 datasets and they achieved better classification accuracies on 8 datasets out of 14 compared to DC. However, DC produced a comparable associative classifier with all other “classical” and “associative” classifiers.

Table 8- Statistically significant wins/losses counts of CDC method on rules:

	DTNB	DT	C4.5	PT	FR	RDR	CBA	SA
W	11	11	9	10	10	8	11	12
L	2	3	2	3	2	4	1	2
N	1	0	3	1	2	2	2	0

Table 9- Overall Coverage:

Dataset	DC	DDC	CDC
Breast Cancer	65.2	72.0	72.7
Balance	74.5	82.8	86.3
Car.Evn	88.7	100.0	100.0
Vote	88.4	86.9	85.1
Tic-Tac-Toe	89.0	92.0	86.0
Nursery	90.4	98.1	100.0
Hayes-root	100.0	100.0	100.0
Lymp	81.0	90.0	88.4
Spect.H	80.9	80.7	79.4
Abalone	74.1	87.6	78.9
Adult	100.0	100.0	100.0
Insurance	81.5	89.5	100.0
Monks	82.4	86.7	90.6
Laptop	86.1	99.0	100.0
Average(%):	84.4	90.4	90.5

On the other hand, accuracy of DC, DDC and CDC was higher than its coverage on "Breast cancer", "Vote" and "Monks" datasets. This fact is not surprising, since uncovered examples get classified by the majority classifier. When the overall coverage is above 85%, proposed methods tend to get a reasonably high accuracy on all datasets.
All of our proposed classifiers achieved the best accuracy on "Breast Cancer" and "Spect.H" datasets among all rule-learner approaches while CDC generated slightly higher number of classification rules comparing to DC and DDC, but on average all of our proposed method achieved the best result in terms of classification rules. Evaluation of our proposed classifiers is shown in Fig 1.

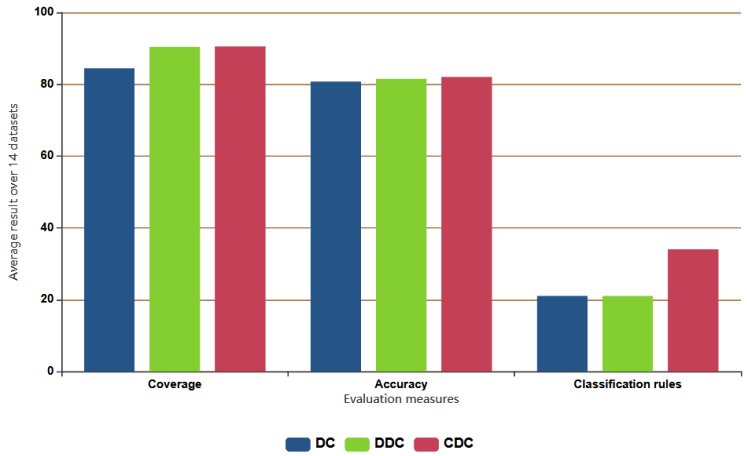

Fig 1. Comparisonof performance of our proposed associative classification models

Fig 1 illustrates that all three methods obtained similar average accuracy. Although CDC gained better coverage than DC, it got worse result in terms of the number of classification rules than that method.

The most important advantage of our proposed methods was to generate a smaller classifier on bigger datasets.

6. Conclusion and Future Work

Experimental evaluations show that we could somehow achieve our intended goal in this research to produce a compact and meaningful classifier by exhaustively searching the entire example space using constraints and clustering. Our DC, DDC and CDC classifiers were able to reduce the number of classification rules while maintaining a classification accuracy that was comparable to state-of-the-art rule-learning classification algorithms. Moreover, we showed in the experiments that our classifiers were able to reduce the number of rules in the classifier by 2-4 times on average compared to the other rule-learners, while this ratio is even bigger on datasets with a higher number of examples.

All three proposed associative classifiers have their advantage on some certain datasets; they are even comparable to each other on the number of generated rules and classification accuracy.

The main drawback of our proposed methods is their time efficiency. In future work, we plan to parallelize DC, DDC, and CDC to bring their time complexity at least a bit closer to state-of-the-art “divide-and-conquer” rule-learning algorithms.

Acknowledgment. The authors gratefully acknowledge the European Commission for funding the InnoRenewCoE project (Grant Agreement #739574) under the Horizon2020 Widespread-Teaming program and the Republic of Slovenia (Investment funding of the Republic of Slovenia and the European Union of the European Regional Development Fund). Jamolbek Mattiev is also

funded for his Ph.D. by the “El-Yurt-Umidi” foundation under the Cabinet of Ministers of the Republic of Uzbekistan.

References

1. Lent, B., Swami, A., Widom, J.: Clustering association rules. In: Proceedings of the Thirteenth International Conference on Data Engineering, Gray, A., Larson, P. England, 220–231. (1997)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: VLDB '94 Proceedings of the 20th International Conference on Very Large Data Bases. Morgan Kaufmann, Santiago, Chile, 487–499. (1994)
3. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings of the 4th InternationalConference on Knowledge Discovery and Data Mining. Agrawal, R., Stolorz, P. New York, USA, 80–86. (1998).doi: 10.5555/3000292.3000305
4. Hu, L. Y., Hu, Y. H., Tsai, C. F., Wang, J. S., Huang, M. W.: Building an associative classifier with multiple minimum supports. SpringerPlus Vol. 5, No. 528. (2016)
5. Deng, H., Runger, G., Tuv, E., Bannister, W.: CBC: an associative classifier with a small number of rules. Decision Support Systems, Vol. 50, No. 1, 163–170, (2014)
6. Khairan, D. R.: New Associative Classification Method Based on Rule Pruning for Classification of Datasets. IEEE Access, Vol. 7, 157783-157795. (2019)
7. Ramesh, R., Saravanan, V., Manikandan, R.,: An Optimized Associative Classifier for Incremental Data Based On Non-Trivial Data Insertion. International Journal of Innovative Technology and Exploring Engineering (IJITEE), Vol. 8, No. 12, 4721-4726. (2019)
8. Thabtah, F. A., 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, 127–133. (2005)
9. Thabtah, F. A., Cowling, P., Peng, Y.: MMAC: a new multi-class, multi-label associative classification approach. In: Proceedings of the fourth IEEE international conference on data mining. Brighton, UK, 217–224. (2004)
10. Abdellatif, S., Ben Hassine, M. A., Ben Yahia, S., Bouzeghoub, A.: ARCID: A New Approach to Deal with Imbalanced Datasets Classification. 44th International Conference on Current Trends in Theory and Practice of Computer Science, Lecture Notes in Computer Science, Austria, Vol. 10706, 569-580. (2008)
11. Chen, G., Liu, H., Yu, L., Wei, Q., Zhang, X.: A new approach to classification based on association rule mining. Decision Support Systems, Vol. 42, No. 2, 674–689. (2008)
12. Kaufman, L., Rousseeuw, P., J.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, USA (1990)
13. Zait, M., Messatfa, H.: A Comparative Study of Clustering Methods. Future Generation Computer Systems, Vol. 13, No. (2-3), 149-159. (1997)
14. Arabie, P., Hubert, L. J.: An Overview of Combinatorial Data Analysis. In: Clustering and Classification. Arabie, P., Hubert, L. J, Soete, G. D. New Jersey, USA, 5–63. (1996)
15. Ng, T. R., Han, J.: Efficient and Effective Clustering Methods for Spatial Data Mining. In: Proceedings of the 20th Conference on Very Large Data Bases (VLDB). Morgan Kaufmann. Santiago, Chile, 144-155. (1994)
16. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An Efficient Data Clustering Method for Very Large Databases. In: Proceedings of the ACM-SIGMOD International conference on Management of Data. Montreal, Canada,103-114. (1996)
17. Theodoridis, S., Koutroumbas, K.: Hierarchical Algorithms. Pattern Recognition, Vol. 4, No. 13, 653-700. (2009)
18. Dua, D., Graff, C.: UCI Machine Learning Repository, Irvine, CA: University of California (2019)

19. Hall, M., Frank, E.: Combining Naive Bayes and Decision Tables. In proceedings of Twenty-First Artificial Intelligence Research Society Conference, Florida, USA, 318-319. (2008)
20. Kohavi, R.: The Power of Decision Tables. In: 8th European Conference on Machine Learning. Lavrač, N., Wrobel, S. Crete, Greece, 174–189. (1995)
21. Hühn, J., Hüllermeier, E.: FURIA: an algorithm for unordered fuzzy rule induction. Data Mining and Knowledge Discovery, Vol. 19, 293–319, DOI: doi.org/10.1007/s10618-009-0131-8, (2019)
22. Frank, E., Witten, I.: Generating Accurate Rule Sets Without Global Optimization. In: Fifteenth International Conference on Machine Learning, Shavlik, J.W. USA, 144–151. (1998)
23. Quinlan, J.: C4.5: Programs for Machine Learning. Machine Learning, Vol. 16, No. 3, 235-240. (1993)
24. Richards, D.: Ripple down rules: a technique for acquiring knowledge. Decision making Support Systems: achievements, trends and challenges for the new decade, Mora, M., Forgionne, G. A., Gupta, J. N. D. USA, 207–226. (2002)
25. Mattiev, J., Kavšek, B.: Simple and Accurate Classification Method Based on Class Association Rules Performs Well on Well-Known Datasets. In: Machine Learning, Optimization, and Data Science, LOD 2019. Nicosia G., Pardalos P., Umeton R., Giuffrida G., Sciacca V, Siena, Italy, Vol. 11943, 192–204. (2019)
26. Dahbi, A., Mouhir, M., Balouki, Y., Gadi, T.: Classification of association rules based on K-means algorithm.In: 4th IEEE International Colloquium on Information Science and Technology. Mohajir, M.E., Chahhou, M., Achhab, M.A., Mohajir, B.E. Tangier, Morocco, 300–305. (2016)
27. Gupta, K. G., Strehl, A., Ghosh, J., Distance based clustering of association rules. Proceedings of artificial neural networks in engineering conference, USA, 759-764. (1999)
28. Kosters, W., Marchiori, E., Oerlemans, A., Mining Clusters with Association Rules. Lecture Notes in Computer Science, DOI:10.1007/3-540-48412-4_4, (1999)
29. Natarajan, R., Shekar, B..: Tightness: A novel heuristic and a clustering mechanism to improve the interpretation of association rules. In Proceedings of the IEEE International Conference on Information Reuse and Integration, USA, 308-313. (2008)
30. Toivonen, H., Klemettinen, M., Ronkainen, P., Hatonen, K., and Mannila, H.: Pruning and grouping discovered association rules. In ECML-95 Workshop on Statistics, Machine Learning, and Knowledge Discovery in Databases, Greece, 47-52. (1995)
31. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I, H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations, Vol. 11, No. 1, (2009)
32. Cohen, W., W.: Fast Effective Rule Induction.: In: ICML'95 Proceedings of the Twelfth International Conference on Machine Learning, California, USA, 115-123. (1995)
33. Mattiev, J., Kavšek, B.: A compact and understandable associative classifier based on overall coverage. Procedia computer science, Vol. 170, Warsaw, Poland, 1161-1167 (2020)

Jamolbek Mattiev has a Master degree in Computer Science from National University of Uzbekistan. He was awarded with first degree diploma at “the best Master Dissertation Work of Uzbekistan” competition in his master studies. He obtained his PhD degree at the Department of Information Sciences and Technologies of the University of Primorska, Slovenia. His research fields include Artificial Intelligence, Data Mining, Machine Learning. In particular, the subfields of Supervised and Unsupervised Learning, Frequent Pattern Discovery and Association Rule Learning, Classification, Clustering.

Branko Kavšek has a PhD in Computer Science from the University of Ljubljana. He is an assistant professor at the University of Primorska, Faculty of Mathematics, Natural Sciences and Information Technologies, a researcher and Vide Head at the Department of Information Sciences and Technologies and member of the Artificial Intelligence Laboratory at the Jozef Stefan Institute in Ljubljana. His research fields include Artificial Intelligence, Data Mining, Machine Learning. In particular, the subfields of Supervised and Unsupervised Learning, Frequent Pattern Discovery and Association Rule Learning, Learning Probabilistic Models and Bayesian Networks Learning, Clustering, and Data Mining applied to Big Data. He is the co-author of several scientific publications and currently involved in national and international projects.

Received: April 30, 2020; Accepted: November 08, 2020.