Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Closed Itemsets and Minimal Generators: A Comparison of Their Effects on Associative Classifiers

Andrea Kulaif

deia.cpk@gmail.com

State University of Campinas

Julia Silva

State University of Campinas

Rosana Veroneze

Université Catholique de Louvain

Fernando J. Von Zuben

State University of Campinas

Research Article

Keywords: Association rules, associative classifiers, closed itemsets, minimal generators, comparative performance

Posted Date: February 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8695223/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Closed Itemsets and Minimal Generators: A Comparison of Their Effects on Associative Classifiers

Andrea Kulaif[1*], Júlia Silva[1], Rosana Veroneze[2], Fernando J. Von Zuben[1]

[1]DCA/FEEC, University of Campinas, 400 Albert Einstein Avenue, Campinas, 13083-852, SP, Brazil.
[2]INGI/ICTEAM, Université catholique de Louvain, Place de l'Université 1, Louvain-la-Neuve, 1348, Wallonia, Belgium.

*Corresponding author(s). E-mail(s): deia.cpk@gmail.com;
Contributing authors: juliaborgescs@gmail.com;
rosana.veroneze@uclouvain.be; vonzuben@unicamp.br;

Abstract

Associative classification (AC) is a well-established family of interpretable classifiers that combines association rule mining with supervised learning. Despite decades of research, a fundamental design choice remains largely unexamined: whether classifiers should be built from general rules, derived from minimal generators (MGs), or from more specific rules, derived from closed itemsets (CIs). While general rules are often assumed to generalize better, CI-based approaches are computationally more efficient and provide a lossless condensed representation of the rule space. However, their practical impact on classification performance has never been systematically isolated and evaluated. In this work, we present a comprehensive empirical study that disentangles rule generality from other algorithmic design choices. Across twenty public datasets with varying characteristics, we construct associative classifiers using either MG-based or CI-based rules, while keeping all other components – rule ranking, selection strategy, and prediction mechanism – strictly identical. Both coverage-based and lazy-based classifier construction strategies are considered, and performance is evaluated using multiple metrics, including accuracy, balanced accuracy, and F1-score. Our results show that classifiers built from MGs and CIs achieve nearly identical predictive performance and often rely on rules from the same equivalence classes, resulting in highly similar classification behavior. In contrast,

CI-based approaches offer substantial computational advantages, significantly reducing rule-mining and classifier-construction costs. These findings challenge the prevailing reliance on general-purpose frequent itemset mining in AC and provide strong empirical support for adopting CI-based mining as the default strategy, without sacrificing effectiveness or interpretability.

Keywords: Association rules, associative classifiers, closed itemsets, minimal generators, comparative performance.

1 Introduction

Classification is a popular research field in machine learning. In many application domains, such as medicine and criminal justice, users are interested not only in the classifier's performance but also in its interpretability, i.e., how decisions are made. Moreover, there is growing concern about interpretability due to new regulations requiring assessment of the model's safety and reliability.

Rule-based classifiers are an important class of interpretable models (Fürnkranz et al., 2012), generally composed of a specific set of IF-THEN rules. In this work, we focus on studying one type of rule-based classifiers: the associative classifiers (Liu et al., 1998).

Associative classification (AC) combines the extraction of association rules (Aggarwal and Han, 2014) and their use to perform classification. It usually relies on a stepwise procedure. First, all rules that meet some constraints are mined by a traditional pattern mining algorithm. The most popular algorithms used in this stage are based on (Geng et al., 2025): Apriori (Agrawal et al., 1994), FP-growth (Han et al., 2000), and Eclat (Zaki et al., 1997), with Apriori being by far the most widely used to date (Geng et al., 2025). In the second stage, the rules are ranked, and a subset of them is selected to form the classifier. Confidence, support, and length of the rules are typical criteria in the ranking stage (Geng et al., 2025).

A fundamental, yet often overlooked, design choice in this process is whether to prioritize shorter, more general rules, or longer, more specific rules. This choice may have profound implications for the resulting classifier's performance, interpretability, and computational cost. Most AC approaches implicitly or explicitly favor general rules. However, proposals that prioritize longer rules (Baralis et al., 2007; Chen et al., 2014) argue that incorporating more contextual attributes may offer greater precision. Moreover, longer rules provide more subsidies for informed decisions. It is particularly interesting in sensitive applications (medicine, law, security), where completeness of explanations may matter more than conciseness.

To formalize this distinction, we consider the concept of equivalence class (Zaki et al., 1997): the set of all itemsets that cover the exact same set of training instances. All rules derived from itemsets within an equivalence class share identical support, confidence, and **any other metric derived from their transaction set** (such as lift, χ^2 test, accuracy-based ones, and principality) (Dam et al., 2022; Makhalova et al., 2020). The difference between them lies in their length. The most general rules

are related to *minimal generators* (MGs), while the most specific ones are related to a *closed itemset* (CI). The debate, therefore, seems to boil down to a selection between MGs and CIs.

The rationale behind favoring one rule type over the other often lacks a deep, evidence-based discussion in the literature. Moreover, a recent comprehensive survey by Geng et al. (2025) reveals that algorithms like Apriori, which mine all frequent itemsets, remain overwhelmingly dominant in AC. This is remarkably inefficient, as many AC methods ultimately discard the vast majority of these mined rules. More surprisingly, even AC algorithms that ultimately select only MGs or CIs for the classifier (e.g., CBA (Liu et al., 1998) and PAM (Chen et al., 2014), respectively) often begin by mining the entire set of frequent itemsets, thereby incurring unnecessary computational overhead (Dam et al., 2022). There are more efficient alternatives. For instance, Charm (Zaki and Hsiao, 2002) and similar algorithms mine directly CIs, producing a more compact and non-redundant rule set. Although early AC systems such as ACCF (Li et al., 2008) successfully used this approach and demonstrated robust performance, specialized miners for CIs (or MGs) remain underutilized.

Why does this inefficiency persist? This is even more puzzling given that we have polynomial-delay algorithms for CI mining (Konecny and Krajča, 2021), and that closed itemsets are a lossless condensed representation of all itemsets (Zaki et al., 1997). We hypothesize that researchers may be hesitant to adopt specific rules, perhaps fearing that longer rules will necessarily lead to poorer generalization, as suggested by the minimum description length (MDL) principle (Li et al., 2006). However, to the best of our knowledge, no prior work has systematically isolated and empirically evaluated the impact of prioritizing general (MG) versus specific (CI) rules on the final classification performance. Existing studies often compare entire AC systems, confounding the effect of rule specificity with other design choices, or remain purely theoretical.

This gap in the literature leaves us without clear answers to several critical research questions (RQs). This work aims to bridge this gap by providing a comprehensive, data-driven analysis. To guide our investigation, we reformulate the core uncertainties into the following RQs:

RQ1 Predictive performance: Considering balanced and imbalanced data, is there a tangible difference in predictive performance between classifiers built from general rules (MGs) and those built from specific rules (CIs) when all other factors are held constant?

RQ2 Computational efficiency: What are the real tradeoffs in terms of computational cost, considering both the initial time spent mining rules and the construction time of the subsequent classifier, between the MG and CI-based approaches?

RQ3 Classifier characteristics: How does the choice of MGs versus CIs affect the properties of the final classifier, such as its size (number of rules) and the distribution of rule lengths?

RQ4 Practical guidance: Based on the empirical evidence, can we establish clear guidelines for practitioners on when to prefer general or specific rules, taking into account predictive performance, computational burden, and model interpretability?

In this work, we conduct an extensive empirical study across twenty diverse public datasets to directly address these RQs. Our methodology is designed to isolate the dispute between rule generality and rule specificity from other sources of variability. We achieve this by:

1. Mining two distinct sets of rules from each dataset: one containing only MGs and the other only CIs, using dedicated and efficient mining algorithms in each case.
2. Constructing classifiers from each rule set using identical ranking, selection, and prediction strategies (including both coverage-based and lazy-based selection).

This experimental design ensures that any observed differences in performance, runtime, or model complexity can be directly attributed to the choice between MGs and CIs. We evaluate the resulting classifiers using multiple performance metrics to ensure a robust assessment, particularly on datasets with class imbalance.

Our findings challenge conventional wisdom, revealing that the significant computational advantages of using CI-based miners do not come at the cost of predictive power. This work provides the empirical foundation for a paradigm shift in the rule-mining stage of AC algorithms, advocating a more efficient approach that can significantly reduce the computational burden of building these models without compromising effectiveness.

The remainder of this paper is organized as follows. In Section 2, we present the concepts of Association Rule Mining for Classification and the main algorithms. Section 3 discusses some relevant literature on the Associative Classification field and gaps in comparing AC methods. Then, in Section 4, we detail our research methodology, the datasets, and the evaluation metrics used in the comparison. After that, Section 5 provides a description of the computational experiments and the results for each proposed scenario. Section 6 presents a discussion of the results. Finally, in Section 7, we present concluding remarks and potential future research steps to further develop AC.

2 Association Rule Mining for Classification

2.1 Preliminaries

Let $\mathbf{A}_{n\times m}$ be a binary matrix containing the set of rows $X = \{1, 2, 3, ..., n\}$ and the set of columns $Y = \{1, 2, 3, ..., m\}$. Each row of $\mathbf{A}$ represents a transaction, and each column represents an item. The element $a_{ij} \in \mathbf{A}$ denotes the relationship between transaction i and item j. (X, Y) represents the matrix $\mathbf{A}$ and (I, J) represents a submatrix of $\mathbf{A}$, being $I \subseteq X$ and $J \subseteq Y$. J is called an *itemset*, and the set of all transactions that contain all items belonging to J is denoted by $J^{\downarrow}$. The support of an itemset J is given by $sup(J) = |J^{\downarrow}|$. J is considered a *frequent* itemset if its support is equal to or greater than a predefined threshold *minsup*.

An itemset J is called a closed itemset (CI) if there is no superset $H \supset J$ for which $H^{\downarrow} = J^{\downarrow}$. If J is a CI, then all subsets $L \subseteq J$ such that $L^{\downarrow} = J^{\downarrow}$ are called generators of J. The set of all generators of J is called the equivalence class of J, denoted by $Equiv(J) = \{L | L \subseteq J \text{ and } L^{\downarrow} = J^{\downarrow}\}$. An itemset $L \in Equiv(J)$ is called a minimal generator (MG) if: $\nexists Z \in Equiv(J)$ s.t. $Z \subset L$. Note that there is only one CI in an equivalence class, which is the maximum element of the class. On the other hand, we

can have several MGs in an equivalence class, which are the minimal elements of the class. If an equivalence class consists of only one element, it is called *trivial*.

Consider the example of a dataset in Table 1. Figure 1 presents the frequent itemsets, CIs, MGs, and equivalence classes for this example (considering minimum support equal to 2). Table 2 presents the support for the equivalence classes. Equivalence class #2, consisting only of one element (itemset {b}), is an example of a trivial equivalence class.

Table 1: Example of a dataset.

	a	**b**	**c**	**d**	**Class**
1	1	1	1	1	1
2	0	1	1	1	0
3	1	1	1	1	1
4	1	1	0	0	0
5	0	1	1	1	0
6	0	1	1	1	0

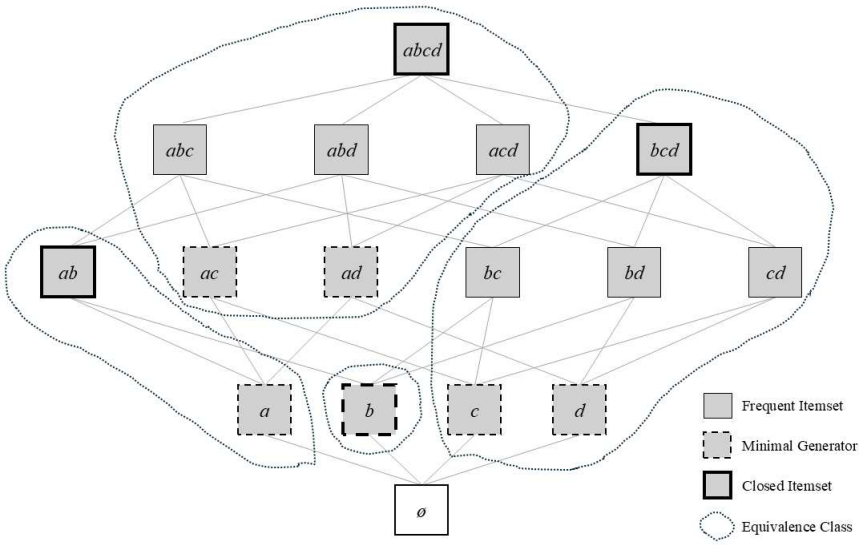


Fig. 1: Frequent itemsets, CIs, MGs, and equivalence classes from the dataset in Table 1 (considering minimum support equal to 2).

An Association Rule (AR) comprises an antecedent and a consequent parts. It is an IF-THEN statement in the form $J \rightarrow K$, where J and K are itemsets and $J \cap K = \emptyset$.

Table 2: Equivalence Classes from the example in Table 1, considering minimum support equal to 2.

#	Frequent Itemsets	Minimal Generators	Closed Itemset	Support	Transactions
1	{a, ab}	{a}	{ab}	3/6	{1, 3, 4}
2	{b}	{b}	{b}	6/6	{1, 2, 3, 4, 5, 6}
3	{c, d, bc, bd, cd, bcd}	{c, d}	{bcd}	5/6	{1, 2, 3, 5, 6}
4	{ac, ad, abc, abd, acd, abcd}	{ac, ad}	{abcd}	2/6	{1, 3}

Some metrics can be used to measure the quality of an AR, such as support and confidence. The support of the rule $J \rightarrow K$ is given by the number of transactions that contain the itemset $J \cup K : sup(J \rightarrow K) = sup(J \cup K)$. It frequently appears related to the relative support ($rsup$) when the support is divided by the total number of transactions in the dataset $rsup(J \rightarrow K) = sup(J \rightarrow K)/|X|$. The higher the support, the more common the rule within the dataset, indicating that the combination of items in the rule occurs frequently together. The confidence of the rule $J \rightarrow K$ is given by the ratio between the support of the rule and the support of the itemset J: $conf(J \rightarrow K) = sup(J \rightarrow K)/sup(J)$. The confidence of a rule is the conditional probability that a transaction contains K given that it contains J, i.e., $P(K|J)$. A rule $J \rightarrow K$ is considered *strong* if its confidence is equal to or greater than a predefined threshold $minconf$. Conversely, the completeness of an association rule $J \rightarrow K$ is given by $comp(J \rightarrow K) = sup(J \cup K)/sup(K)$.

A special case of association rule is the Class Association Rule (CAR), which takes the form $J \rightarrow c$, where the consequent c is a class label. In this context, a CAR is constructed from an itemset J in the antecedent and a class label in the consequent, making it suitable for classification tasks. The total number of CARs that can be generated from an itemset J is equal to the number of distinct class labels in the dataset. Each rule will have a unique confidence value, which depends on the support of the itemset $\{J \cup \{c\}\}$ relative to the support of J.

As we can see from the example in Table 2, in a given equivalence class, the number of items in an MG is always less than or equal to the number of items in the corresponding CI. This characteristic makes CARs based on MGs more generalist than rules based on CIs. The more items (attributes) in the antecedent, the more specific the rule. Table 3 shows the rules generated from all MGs and CIs from the example in Table 2, with the consequent part defined considering the majority class in the instances covered by the rule.

2.2 Associative Classification

Associative classification (AC) combines the mining of association rules with classification tasks (Liu et al., 1998). Its algorithms typically follow a stepwise procedure, consisting of rule generation and classifier construction.

The rule generation step includes mining itemsets and assigning a label to each of them. It is usually the most expensive step to create an associative classifier. Given an

Table 3: Rules generated based on MGs and CIs from example in Table 2, considering minimum support equal to 2.

	MGs Rules	CI Rules	Support	Confidence
1	$a \rightarrow 1$	$ab \rightarrow 1$	2/6	2/3
2	$b \rightarrow 0$	$b \rightarrow 0$	4/6	4/6
3	$c \rightarrow 0$ $d \rightarrow 0$	$bcd \rightarrow 0$	3/6	3/5
4	$ac \rightarrow 1$ $ad \rightarrow 1$	$abcd \rightarrow 1$	2/6	2/2

itemset J, two strategies can be used to generate CARs. The first strategy produces multiple CARs with the same antecedent (condition) derived from J, each pointing to a different class label and varying in support and confidence values, as seen in the PAM algorithm (Chen et al., 2014). The second strategy generates only a single CAR – the one associated with the class label that yields the highest confidence among all possible rules with the same antecedent – as used in the CBA algorithm (Liu et al., 1998). In both strategies, minimum support ($minsup$) and confidence ($minconf$) thresholds can be applied, ensuring that the CARs are frequent and strong.

In the classifier construction phase, the rules are ranked, selected, and combined to build the classifier. The ranking process typically considers factors such as confidence, support, and generality (or specificity) of the rules. Rule selection is commonly guided by heuristics. The most widely used heuristic is based on dataset coverage (Alwidian et al., 2018; Liu et al., 1998). This greedy heuristic selects rules iteratively until all training instances are covered by at least one rule, if feasible – that is, if the rules generated in the first step can cover all instances. Another popular heuristic, named *lazy* (Baralis et al., 2007; Baralis and Garza, 2002), discards only *harmful* rules – that is, rules that only misclassify training instances.

For labeling new instances, two main strategies are commonly used (Aoga et al., 2018): *rule lists* (also called decision lists or ordered rule sets) and *rule sets*. In *rule lists*, the rules are ordered, and the last rule is usually a *default rule* (i.e., the *ELSE* rule). Predictions are made using the first rule in the list that matches the unseen instance. In *rule sets*, the rules are unordered, and all matching rules are used to make a prediction, typically through a voting mechanism.

In the next section, we present some of the most widely used AC algorithms in the literature and describe their main aspects.

2.3 Associative Classification Algorithms

The Apriori algorithm (Agrawal et al., 1994) is widely used during the rule generation phase of AC algorithms. In this algorithm, the itemsets are mined by levels, adding a new item to the already mined frequent itemset whenever this keeps its support at or above the user-specified threshold. If the addition of an item makes the itemset infrequent, that new itemset is discarded, as the addition of new items will not make

the itemset frequent. Note that Apriori enumerates all frequent itemsets, which include the MGs and the CIs.

CBA (Liu et al., 1998) uses the Apriori algorithm to extract all candidate rules whose confidence and support are greater than the thresholds determined by the user. During the classifier construction step, the rules obtained are ranked according to the following criteria: (1) higher confidence value, (2) higher support value, and (3) rule that was mined earlier. This last criterion causes the CBA to prioritize more general rules, since the Apriori algorithm generates shorter rules before longer ones. After ranking the rules, CBA applies a coverage-based heuristic to select the rules that will form the classifier, structured as a *rule list*. CBA then discards the rules that do not improve the final accuracy of the resulting classifier. The class label of the default rule is assigned to the class with the highest number of training instances that remain uncovered by a rule.

In many datasets, the number of instances of each class can be quite uneven, leading to mining too many rules for a given class that appears more in the training set and too few or even no rules for classes with few instances. To overcome this problem, CBA2 (Liu et al., 2000) implements multiple supports, setting a different threshold for each class according to its number of occurrences in the training set.

Enhanced CBA (ECBA) (Alwidian et al., 2016) utilizes a modified version of the Apriori algorithm as follows. Initially, it mines all frequent CARs with only one item in the antecedent part. The generated rules are then clustered based on their class labels. For example, if the training dataset contains two class labels, c_1 and c_2, the rules are divided into two separate clusters, each corresponding to one of the class labels. Subsequently, Apriori continues its operation independently within each cluster, ensuring that all rules within a cluster point to the same class label. After completing the rule generation phase, ECBA retains only the rules whose harmonic mean (HM) of confidence and support meets or exceeds a user-defined threshold. The HM of a CAR $R : J \rightarrow c$ is calculated as follows:

$$HM(R) = \frac{2 * conf(R) * rsup(R)}{conf(R) + rsup(R)}. \tag{1}$$

The rules are then ranked according to the following criteria: (1) higher HM, (2) higher confidence, (3) higher support, and (4) rule that was mined earlier (i.e., shorter rules). Like CBA, ECBA also applies a coverage-based heuristic to select the rules that will form the classifier, structured as a *rule list*.

The Principal Association Mining (PAM) algorithm (Chen et al., 2014) also does rule mining using an Apriori-based algorithm. Unlike CBA, PAM retains all conflicting rules, that is, rules with the same condition but pointing to different class labels are kept as candidate rules for the next stage. PAM ranks the rules according to the following criteria: (1) higher confidence, (2) higher support, (3) higher number of items in the rule (i.e., PAM prioritizes specific rules), and (4) smaller lexicographical order of the rule regarding its items. The preference for longer rules stems from their ability to cover the same training instances as their shorter counterparts (derived from MGs) while introducing additional constraints. The authors argue that longer rules help reduce over-generalization from limited sample variety. PAM also adopts a new

metric, named *Principality*, to evaluate the rules, which is a convex combination of confidence and completeness:

$$Principality(R, \lambda) = \lambda * conf(R) + (1 - \lambda) * comp(R), \tag{2}$$

where $R : J \rightarrow c$ is a CAR and $0 \leq \lambda \leq 1.0$ is a weight coefficient to be specified by the user preference. PAM employs a coverage-based heuristic to select rules for constructing the classifier, which is also organized as a rule list. However, unlike CBA, PAM recalculates the quality of the remaining rules at each iteration (based on the Principality metric). This dynamic recalculation significantly increases the computational cost of PAM's heuristic compared to the simpler approach used by CBA.

CMAR (Li et al., 2001) uses a method based on FP-growth (Han et al., 2000) for rule mining. FP-Growth employs a different strategy than Apriori, yet, like Apriori, it is designed to enumerate all frequent itemsets. CMAR ranks the rules according to the following criteria: (1) higher confidence value, (2) higher support value, and (3) fewer items in the rule. CMAR employs two methods for rule pruning. First, given two rules R_i and R_j, such that R_i is a more general rule than R_j, CMAR prunes R_j if R_i has a higher rank and $conf(R_i) > conf(R_j)$. Second, for each rule, $R : J \rightarrow c$, CMAR tests whether the itemset J is positively correlated with the class label c using a χ^2 test. Rules that do not meet a user-defined significance threshold are pruned. The CR-Tree, a prefix tree structure used by CMAR, accelerates both pruning methods. Additionally, CMAR applies a coverage-based heuristic to select rules for the classifier, similar to the approach used in CBA. However, a key difference lies in how CMAR handles covered training instances. Instead of removing an instance immediately after it is covered by a rule, CMAR allows it to remain in the dataset until it is covered by at least δ rules, where δ is a user-defined parameter. Another key difference is that the CMAR's classifier is organized as a rule set.

The MCAR algorithm (Thabtah et al., 2005) adopts an algorithm based on Eclat (Zaki et al., 1997) to mine the rules, which is also designed to enumerate all frequent itemsets. MCAR ranks the rules according to the following criteria: (1) higher confidence value, (2) higher support value, (3) higher support value of the rule's condition, (4) smaller number of items in the rule, and (5) rule that was mined before. It also adopts a coverage-based heuristic to select rules to construct the classifier, which is organized as a rule list.

The ACCF algorithm (Li et al., 2008) uses the Charm algorithm (Zaki and Hsiao, 2002) to generate frequent and strong rules. Charm is designed to enumerate all frequent CIs, which typically results in a significantly smaller set of candidate rules compared to algorithms that enumerate all frequent itemsets. This allows ACCF to discover fewer rules, which speeds up the rule discovery process. Using CIs, ACCF generates more specific rules to construct the classifier compared to rules derived from other representatives in the same equivalence class. ACCF ranks the rules using the following criteria: (1) higher confidence, (2) higher support, (3) smaller number of items in the rule, and (4) smaller lexicographical order of the rule. To construct the classifier, ACCF employs a coverage-based heuristic and organizes the selected rules into a rule list. However, ACCF's rule list does not include a default (ELSE) rule.

Instead, if no rule fully matches a test instance, the first rule that partially matches the instance, i.e., contains at least one matching item, is applied.

L3 (Baralis et al., 2007; Baralis and Garza, 2002) uses an extension of the Essence algorithm (Baralis and Chiusano, 2004) to generate rules. Essence leverages the concepts of CIs and MGs to produce a compact representation of the complete rule set, ensuring efficient use of space. This compression is lossless, meaning that the original rule set can be fully reconstructed from the compact form. In addition to the traditional confidence threshold, L3 also prunes the rules using a χ^2 threshold. Rules are ranked according to the following criteria: (1) higher confidence, (2) higher support, (3) higher number of items in the rule, and (4) higher lexicographical order of the rule with respect to its items. Consequently, L3 prioritizes specific rules. L3 employs a lazy-based heuristic to select the rules that form the classifier, which also incorporates a coverage-based criterion within its selection process. Only harmful rules – those that exclusively misclassify training instances – are definitively pruned. Rules that are neither pruned nor selected by the coverage-based heuristic are placed in a spare rule set. The selected and spare rule sets are exploited to construct a two-level classifier: rules in the selected rule set are assigned to Level I, whereas rules in the spare rule set are assigned to Level II. The classifier is organized as a rule list. However, to classify an unlabeled instance, Level I is consulted first. If no rule in Level I matches the instance, then Level II is considered. If no rule matches the instance in either level, the instance remains unlabeled.

The Fast Associative Classification Algorithm (FACA) (Hadi et al., 2016) uses the dEclat algorithm (Zaki and Gouda, 2003) to generate frequent and strong rules. dEclat is a variant of Eclat that uses *diffsets* (Zaki and Gouda, 2003) to optimize support calculation. FACA ranks the rules according to the following criteria: (1) fewer items in the rule, (2) higher confidence, (3) higher support, and (4) the rule that was mined earlier. FACA adopts a coverage-based heuristic to select rules to construct the classifier, which is organized as a rule set.

The CPAR algorithm proposed in (Yin and Han, 2003) combines the strengths of AC and traditional rule-based classification by integrating elements from both approaches. Unlike traditional associative classifiers that rely on itemset mining algorithms for rule generation, CPAR employs a greedy algorithm inspired by FOIL (Quinlan, 1990). In their extension of FOIL, instead of removing an example once it is correctly covered by a rule, its weight is decreased by multiplying a factor. This weighted approach allows the generation of more rules compared to the original FOIL. After generating the rules, the prediction power of each rule is estimated based on the *Laplace expected error estimate* (Clark and Boswell, 1991) as follows:

$$LaplaceAccuracy = \frac{(nc+1)}{(ntot+k)}, \tag{3}$$

where k represents the number of classes, $ntot$ represents the number of training set instances covered by the rule, among which nc belongs to the class associated with the rule consequent. For prediction, CPAR adopts a rule set scheme: from all the rules that match the test instance, the m rules of each class (where m is a user-defined parameter) that have the highest Laplace accuracy are selected. For each class, the

average expected accuracy of its rules is calculated, and the class with the highest average is assigned to the instance.

It is worth noting that the CBA, CBA2, ECBA, CMAR, MCAR, and FACA algorithms prioritize generalist rules, whereas the PAM, ACCF, and L3 algorithms favor specific rules. In the case of L3, the choice between prioritizing specific or general rules is configurable, with specific rules set as the default option.

It is also worth highlighting that many approaches in the literature mine all frequent itemsets and only decide between shorter or longer rules during the classifier construction. However, these works often lack an in-depth discussion about the rationale behind favoring one type of rule over the other. In many cases, since the classifier ultimately prioritizes either more specific or more generalist rules, the rule mining process could be optimized by focusing on these characteristics from the beginning, such as by applying CI or MG mining algorithms. Indeed, Dam et al. (2022) demonstrated, for instance, that: (1) the condition of a CAR selected for CBA's classifier is an MG of minimum length, and (2) the condition of a CAR selected for PAM's classifier is a CI. This reveals the inefficiencies in mining many unnecessary rules during the rule generation stage, which negatively impacts runtime, memory usage, and overall computational efficiency.

3 Related Works

The literature on associative classification (AC) contains few dedicated reviews and comparison studies. Most comparisons arise when a new algorithm is proposed and evaluated against existing ones. However, these comparisons are typically conducted by the authors of the new algorithm and focus on overall performance rather than on analyzing specific strategies used by different AC algorithms. In what follows, we refer to those research positioning works.

Associative classification techniques were reviewed by Thabtah (2007). The algorithms CPAR, CMAR, MCAR, MMAC, and L3 were theoretically compared in terms of their approaches to rule mining, ranking, and pruning, as well as their prediction strategies. The authors pointed out the advantages and disadvantages of each algorithm without presenting experiments. The authors noted that most AC algorithms apply the ranking of rules based on support, confidence, and rules' length, giving preference to more generalist rules, i.e., with fewer items in the antecedent.

Wedyan (2014) presented a comprehensive review of various aspects of AC algorithms, including rule generation, rule ranking, rule selection, and class prediction for unseen data points. They categorized and discussed various algorithms based on these aspects, highlighting their underlying methodologies. However, the paper remained a conceptual analysis, as it did not include any experimental comparisons of the reviewed AC algorithms. Similar to (Wedyan, 2014), Abdelhamid and Thabtah (2014) also explored the theoretical aspects of AC algorithms, providing examples of algorithms for each aspect discussed.

Padillo et al. (2019) conducted an experimental comparison of CBA, CBA2, CMAR, CPAR, and other rule-based classifiers, such as RIPPER (Cohen, 1995), focusing on predictive performance, interpretability, and computational efficiency. Their

study, which analyzed 30 classical public datasets, compared these algorithms as whole systems rather than examining their specific strategies. Notably, all three pure AC algorithms (CBA, CBA2, and CMAR) prioritize short-length rules in their ranking process. Additionally, the authors proposed adaptations of CBA and CPAR for Big Data environments using Apache Spark and Flink. They then evaluated these adaptations on 40 datasets (including 10 Big Data datasets) and compared them to other rule-based algorithms designed for Big Data. To further assess scalability, they also conducted experiments with synthetic data under different runtime conditions. While the Big Data scenario is relevant, our focus remains on the comparative evaluation of AC algorithms in traditional settings.

Recently, Geng et al. (2025) presented a literature review on AC, following the *Systematic Literature Review Guidelines* from software engineering (Kitchenham et al., 2007). Their comprehensive analysis examines AC research from both methodological and application perspectives. At the methodological level, they review a broad spectrum of AC approaches, including traditional, fuzzy, and evidential methods. These methods are categorized according to their strategies for rule mining, rule ranking, rule pruning (selection), and class prediction. On the application side, the authors identified ten distinct domains, drawing insights from 200 application-oriented papers. Additionally, the review highlighted current research challenges and outlined potential future directions for AC.

Although some publications compare methodologies for different stages of building an associative classifier, these studies remain purely theoretical. The impact of prioritizing general or specific rules on classification performance has not been thoroughly explored. The works discussed above provide little insight into how favoring general or specific rules affects classifier performance when applied to datasets of different natures. The following experimental section aims to bridge this gap by offering a more comprehensive analysis to guide informed decisions in AC.

4 Experiments

4.1 Datasets adopted in the experiments

The experiments were carried out in 20 different datasets extracted from the UCI Machine Learning Repository, each with a different configuration of shape, density, and number of items. Table 4 presents the datasets used in our experiments. The column #Frequent Items represents the average number of items that have support greater than or equal to the smallest *min_sup* (among the multiple ones) provided by the user.

The original datasets were pre-processed. All numerical attributes in the datasets were partitioned by the Minimum Description Length-based discretizer (MDLP) (Fayyad and Irani, 1993). For a categorical feature, all possible values give rise to an item. The final data matrix has one column for each bin or category, with binary values. The dataset was then divided into 10 folders for cross-validation. For missing data, zeros were imputed.

Table 4: The main attributes of the selected datasets

Dataset	#Rows	#Cols	#Items	#Freq[1]	#Classes	Class Distribution	Missing Values
Adult	48842	14	161	101	2	{76%, 24%}	Yes
Bank	45211	15	96	96	2	{88%, 12%}	No
Breast	699	9	28	28	2	{66%, 34%}	Yes
Car	1728	6	21	21	4	{22%, 4%, 70%, 4%}	No
Chess	3196	36	38	36	2	{48%, 52%}	No
Contraceptive	1473	9	25	25	3	{43%, 22%, 35%}	No
CovType	581012	54	331	330	7	{36%, 49%, 6%, 0.5%, 2%, 3%, 4%}	No
Diabetic	1151	19	32	32	2	{47%, 53%}	No
German	1000	20	68	68	2	{30%, 70%}	No
Internet	3279	1558	1583	1578	2	{14%, 86%}	Yes
Iris	150	4	12	12	3	{33.3% each}	No
Kidney	400	24	89	82	2	{62%,38%}	Yes
Magic	19020	10	78	76	2	{65%, 35%}	No
Mushroom	8124	22	111	96	2	{52%, 48%}	Yes
Nursery	12960	8	26	26	5	{33%, 33%, 0.015%, 31%, 2.5%}	No
PageBlocks	5473	10	71	71	5	{89.8%, 6%, 0.5%, 1.6%, 2.1%}	No
PenDigits	10992	16	165	165	10	{10% each}	No
Shoppers	12330	17	52	52	2	{85%,15%}	No
TicTacToe	958	9	27	27	2	{35%, 65%}	No
Wine	178	13	32	32	3	{33%, 40%, 27%}	No

[1] #Frequent Items

4.2 Strategies for the comparative analysis

In this work, we conduct an extensive empirical study to investigate the impact of prioritizing specific rules (CIs) versus generalist rules (MGs) in associative classifiers. Using dedicated algorithms for mining CIs and MGs, we ensure that the rule mining stage reflects these two approaches. Classifiers are built and evaluated using the same methods, isolating the effect of rule specificity on predictive performance and computational cost.

We could have adopted any algorithm for enumerating CIs and MGs. However, for reasons of familiarity, we chose the In-Close5 algorithm (Andrews, 2018) to mine CIs and the Talky-G algorithm (Szathmary et al., 2009) to mine MGs. As in many AC proposals, such as CBA, only the rule with the highest confidence is considered among all rules with the same antecedent (condition). For example: let $R_1 : \{x_1, x_2\} \rightarrow c_1$, $R_2 : \{x_1, x_2\} \rightarrow c_2$, $conf(R_1) = 55\%$ and $conf(R_2) = 45\%$, only R_1 is considered.

After mining the rules in the training dataset, the next step is to rank the rules. In this work, we ranked the rules according to the following criteria: (1) higher confidence value, (2) higher support value, and (3) rule that was mined earlier.

We employed two heuristics – coverage-based and lazy-based – for selecting rules for the classifier, aiming to mitigate any potential bias that could arise from this selection process.

Our coverage-based heuristic is described in Algorithm 1. It takes as input the ranked rules and the training dataset. The heuristic iteratively selects all rules that classify correctly at least one instance not covered by previously selected rules, continuing until all instances in the training set are covered or no more rules remain to be selected.

Algorithm 1 CoverageSelection(*candidateRulesRanked*, *trainDataset*)

```
1:  selectedRules ← []
2:  for each rule r in candidateRulesRanked do
3:      if rule r classifies correctly at least one instance in trainDataset then
4:          add the rule r in selectedRules
5:          remove from trainDataset all instances classified correctly by the rule r
6:      end if
7:      if trainDataset is empty then
8:          return selectedRules
9:      end if
10: end for
11: return selectedRules
```

Our lazy-based heuristic is described in Algorithm 2. It also takes as input the ranked rules and the training dataset. Going through all the candidate rules, it checks which ones classify at least one instance of the training set correctly and includes them at Level 1 of selected rules. The instances covered by them are removed from the training set. The rules that do not classify any instance correctly and classify at least one instance incorrectly are discarded. This process is repeated until all instances in the training set are covered or no more rules remain to be selected. All rules that were not discarded (non-harmful rules) and that are not in Level 1 are included in Level 2 of the rule selection.

Both classifiers (produced by the coverage-based and the lazy-based heuristics) are structured as a rule list. For the lazy-based ones, rules belonging to Level 1 appear first in the list, followed by rules from Level 2. The class label of the default rule (ELSE) is determined by the majority class in the training dataset.

4.3 Assessing the performance of the classifiers

To evaluate the performance of our classifiers, we employed 10-fold cross-validation. Thus, the reported results represent the average across all folds.

To assess the overall complexity of classifiers built using CIs and MGs, we compared the number of: (*i*) rules mined, (*ii*) rules selected for the classifier, and (*iii*) rules in

Algorithm 2 LazySelection($candidateRulesRanked, trainDataset$)

```
Level1 ← []
harmful ← []
for each rule r in candidateRulesRanked do
    if rule r classifies correctly at least one instance in trainDataset then
        add the rule r in Level1
        remove from trainDataset all instances covered by the rule r
    else if rule r classifies incorrectly at least one instance in trainDataset then
        add the rule r in harmful
    end if
    if trainDataset is empty then
        break
    end if
end for
Level2 = candidateRulesRanked - Level1 - harmful
return Level1, Level2
```

the classifier that evaluated *True* at least once when predicting labels for the test set. Computational resource demand was measured using the runtime of both rule mining and classifier construction phases. Predictive performance was evaluated using Accuracy, Balanced Accuracy, and F1-score.

Accuracy is a classification metric that measures the proportion of correctly classified instances out of the total instances in a dataset. While it is a simple and widely used measure, it may not be suitable for imbalanced datasets, where certain class labels are significantly more frequent than others. To address this limitation, we also considered Balanced Accuracy, which provides a more reliable assessment in such cases. Additionally, we considered the F1-score, which is the harmonic mean of precision and recall. We applied the weighted F1-score, which calculates metrics for each label, and finds their average weighted by the number of true instances for each label.

To statistically compare the predictive performance of the classifiers built on MGs and CIs, we employed the Wilcoxon signed-rank test. This non-parametric test assesses whether the distribution of differences between paired observations – in our case, the performance scores from each fold of the 10-fold cross-validation – is symmetric around zero. By applying this test to the results of the two approaches across the 10 folds, we aim to determine whether any observed performance differences are statistically significant.

5 Results

The experiments were carried out on an Intel Core i9-9900K 3.60GHz machine with GeForce GTX 660 Ti running on the Ubuntu GNU / Linux operating system with 32 GB of RAM.

We executed the **10-fold cross-validation** sequentially for each dataset to prevent resource competition and to avoid a negative effect on runtime. Notice that the **reported results represent the average across all folds**.

We used our own implementations of the In-Close5 and Talky-G algorithms, which are available at https://github.com/rveroneze/inclose5 and https://github.com/rveroneze/talkyg, respectively. Other codes and data are available at https://github.com/andreacpk/CIxMG.

For the rule generation step, we set multiple threshold values for $minsup$, assigning a distinct threshold for each class label. Specifically, the threshold for each class was defined as 1% of the number of instances belonging to that class. Additionally, we set $minconf$ to 0%.

We used the SciPy library's implementation of the Wilcoxon signed-rank test[1] to perform our statistical comparisons. Specifically, we set the *zero_method* parameter to *zsplit*, which handles zero differences by including them in the ranking process and evenly splitting their ranks between positive and negative differences. All other parameters were kept at their default values.

5.1 Rule Comparison

Table 5 shows the number of itemsets mined and the runtime in seconds for each approach, including the difference between MG and CI methods. Considering the total amount, we can notice in most cases a slight difference in the number of mined itemsets. In the case of the Kidney dataset, more than twice as many itemsets were obtained using the MG method than using the CI method, with only 8% of the itemsets matching each other. However, for the Shoppers and Diabetics datasets, both algorithms gathered a similar number of rules, but distinct sets of rules. The same happened for the German dataset, despite the large amount of CI and MG mined. Almost 6 times more MGs than CIs were found for the Internet dataset, with only a 5% coincidence between them. Note that the matches are related to rules originated from trivial equivalence classes.

However, the difference in runtime between the two algorithms is very large in all cases because Talky-G is slower than the In-Close5 algorithm, even in cases where the same number of itemsets was mined. For example, for the Car and Nursery datasets, both methods mined exactly the same rules, but Talky-G took twice or three times as long as In-Close5 for the same task. In the case of Covtype, although the number of itemsets mined by both methods is very close, with a match of 84%, the time to mine MGs was more than 8 times greater. It is important to note that this would occur with other algorithm choices as well, since the process of mining CIs is inherently more straightforward compared to that of mining MGs.

Table 6 shows the runtime of the classifier construction step. As expected, in cases where there is a large discrepancy in the number of MGs and CIs mined, there is also a large divergence in the computational cost of this step, regardless of the selection approach used. The coverage-based heuristic generally has a shorter runtime than the lazy-based one.

Table 7 shows the number of rules in the classifiers constructed using the coverage-based heuristic for rule selection. Although the number of rules selected is practically the same, regardless of the rule mining approach, we have found that these rules do not always coincide. Even with the same number of rules selected, the Bank, Internet,

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html

Table 5: Results of the two rule mining processes for each dataset considering the number of itemsets mined and the runtime in seconds.

	Rule Generation					
	CI		MG			
Dataset	Mined	Runtime(s)	Mined	Runtime(s)	Matching	% Match
Adult	55099	2.5698	76353	28.7602	1	0%
Bank	37649	1.6914	42288	12.6589	24698	58%
Breast	4577	0.0123	4963	0.0255	3385	68%
Car	3187	0.0079	3187	0.0153	3184	100%
Chess	16410	0.0774	18023	0.2135	6089	34%
Contraceptive	5486	0.0180	5509	0.0394	4256	77%
CovType	11902	5.3780	11922	44.1840	10009	84%
Diabetic	3788	0.0184	4156	0.0464	0	0%
German	509860	1.1589	556388	2.0118	1	0%
Internet	5271	0.0426	31086	0.1674	1472	5%
Iris	62	0.0001	74	0.0003	32	44%
Kidney	890832	2.1956	1821837	7.0625	143147	8%
Magic	17956	0.3858	17957	1.8046	14911	83%
Mushroom	54061	0.6234	85774	2.7228	1	0%
Nursery	10516	0.0846	10516	0.2647	10515	100%
PageBlocks	32277	0.1582	34121	0.4287	4616	15%
PenDigits	696738	2.1932	738878	4.1729	328807	45%
Shoppers	49695	0.5166	51061	1.7766	0	0%
TicTacToe	16283	0.0338	16303	0.0576	15561	95%
Wine	5436	0.0110	8009	0.0252	1568	20%

and Iris cases have a low coincidence rate between the rules, 62%, 53%, and 43%, respectively. However, in general, the selection by coverage culminated in maintaining or increasing compatibility between the set of rules.

Table 8 shows the number of rules in the classifiers constructed using the lazy-based heuristic for rule selection. The number of rules selected by the lazy method presents, in most cases, the same variation from the rule mining process when comparing the use of the CI and MG.

In the Internet and Kidney cases, which had very low compatibility in mining, the selection by coverage obtained a greater coincidence between the rules, 53% and 75%, respectively. However, through Lazy selection, the coincidence rate was maintained when compared to mining, 5% and 8%, respectively. This is because the number of rules selected by the latter is much greater than that of the former. In fact, the Lazy selection did not prune any mined rules for the Kidney dataset.

An exception is the Chess dataset, which, for both selection methods, obtained a large increase in compatibility between the rules at this stage, with 100% for coverage and 91% for lazy, considering that only 34% of the MGs and CIs mined coincided.

Up to this point, we have focused on analyzing the similarity between the rules mined based on CIs and MGs and the preferences of these two selection methods to check whether there is an impact on the final version of the constructed classifier.

Table 6: Runtime of the classifier construction for each rule mining method.

	Runtime (s)			
	CI		MG	
Dataset	Coverage	Lazy	Coverage	Lazy
Adult	517.0077	546.9375	519.2165	559.5828
Bank	478.0655	505.3146	470.8807	500.3843
Breast	1.3645	3.2070	1.4139	3.3717
Car	1.9285	3.6925	1.9370	3.1197
Chess	13.0420	19.8083	13.5962	20.9443
Contraceptive	6.4710	8.1907	6.3846	7.8612
CovType	8210.2705	7765.6369	8193.6495	7744.8708
Diabetic	2.5548	4.1071	2.6145	4.1843
German	133.7792	359.0173	144.9595	384.1970
Internet	6.2839	8.5417	12.7483	25.7477
Iris	0.0348	0.0533	0.0376	0.0616
Kidney	220.5759	617.8796	449.8501	1224.7868
Magic	191.5213	196.7741	191.1788	196.5410
Mushroom	16.2081	44.9101	24.5723	65.6850
Nursery	17.6753	22.5088	18.1767	22.5678
PageBlocks	12.9879	27.6985	12.1201	25.4739
PenDigits	250.0854	907.5415	261.4411	594.3123
Shoppers	52.5320	76.1915	52.4828	75.6063
TicTacToe	4.2303	14.2892	4.2545	11.0709
Wine	1.3861	3.7409	2.0637	5.3955

Next, we will proceed with the comparison of the rules in the classifier that evaluated True at least once when predicting labels for the test set, not considering when the ELSE rule was applied. These results are in Tables 9 and 10. The %CI and %MG columns represent the percentage of rules that evaluated True at least once for the test dataset out of the total classifier rules. When interpreting these results, it is important to keep in mind that we used a 10-fold cross-validation, which means that each test set contains 10% of the data, while the classifier was trained on the remaining 90%. As a result, the test data in each fold may have limited variability, which can lead to some rules not being evaluated as True at least once. Thus, these rules may represent information that is not present in the test set.

For most datasets, in experiments with coverage selection, the number of rules in the classifier that evaluated True at least once when predicting labels for the test set is similar when comparing the two rule mining methods. It also keeps the coincidence rate close to that observed when considering all rules of the classifier, with some exceptions. For the Chess dataset, the coincidence between the rules, which had increased in the classifier, returns to the level presented in the mining stage, around 37%. For the Iris dataset, until the previous phases, the two methods shared around 43% of the rules. This rate fell to 17% in the prediction phase, a significant reduction. These last two behaviors were also observed in experiments with lazy selection.

Table 7: Number of rules in the classifiers constructed using the coverage-based heuristic for rule selection, and number of rules common to CI and MG experiments.

	Coverage-based selection			
Dataset	CI	MG	Matching	%Match
Adult	1905	1905	0	0%
Bank	2649	2649	1638	62%
Breast	65	64	58	91%
Car	181	181	175	97%
Chess	459	460	459	100%
Contraceptive	461	461	321	70%
CovType	3518	3517	3012	86%
Diabetic	117	117	0	0%
German	348	348	0	0%
Internet	207	207	109	53%
Iris	14	14	6	43%
Kidney	12	12	9	75%
Magic	1514	1514	1009	67%
Mushroom	46	46	0	0%
Nursery	447	446	436	98%
PageBlocks	340	275	181	66%
PenDigits	1413	1413	1074	76%
Shoppers	789	790	0	0%
TicTacToe	28	29	25	86%
Wine	11	12	8	67%

Table 8: Number of rules in the classifiers constructed using the lazy-based heuristic for rule selection, and number of rules common to CI and MG experiments.

	Lazy-based selection			
Dataset	CI	MG	Matching	%Match
Adult	53027	73662	1	0%
Bank	32694	36946	20857	56%
Breast	4322	4707	3138	67%
Car	3019	3019	3014	100%
Chess	15507	17039	15507	91%
Contraceptive	4727	4750	3618	76%
CovType	10576	10594	8911	84%
Diabetic	3607	3962	0	0%
German	502509	548912	1	0%
Internet	5167	30744	1429	5%
Iris	62	74	33	45%
Kidney	890832	1821837	143148	8%
Magic	14693	14694	12070	82%
Mushroom	54061	85774	1	0%
Nursery	9989	9991	9988	100%
PageBlocks	32241	34121	4626	15%
PenDigits	684388	726148	322021	44%
Shoppers	49026	50386	0	0%
TicTacToe	16283	16303	15562	95%
Wine	5436	8009	1569	20%

Table 9: Number of rules in the classifier that evaluated True at least once when predicting labels for the test set in the experiments involving the coverage selection and number of rules common to CI and MG.

	Applied Rules - Coverage					
Dataset	CI	%CI	MG	%MG	Matching	%Match
Adult	689	36%	689	36%	0	0%
Bank	1079	41%	1079	41%	653	61%
Breast	16	25%	16	25%	16	100%
Car	46	25%	46	25%	44	96%
Chess	156	34%	156	34%	59	38%
Contraceptive	98	21%	98	21%	65	66%
CovType	1529	43%	1529	43%	1251	82%
Diabetic	39	33%	39	33%	0	0%
German	66	19%	67	19%	1	1%
Internet	89	43%	89	43%	52	58%
Iris	6	43%	6	43%	1	17%
Kidney	6	50%	6	50%	4	67%
Magic	622	41%	621	41%	379	61%
Mushroom	40	87%	40	87%	0	0%
Nursery	222	50%	222	50%	218	98%
PageBlocks	98	29%	87	32%	66	76%
PenDigits	379	27%	378	27%	301	80%
Shoppers	281	36%	281	36%	0	0%
TicTacToe	17	61%	17	59%	17	100%
Wine	6	55%	6	50%	3	50%

Despite the discrepancy in the number of rules selected by the lazy method, the number of rules in the classifier that evaluated True at least once when predicting labels for the test set is exactly the same when we compare the experiments with the two mining methods. It is also possible to observe a higher coincidence rate between the rules applied, especially for the Breast, Internet, Kidney, Pageblocks, Pendigits, and Wine datasets.

5.2 Predictive Performance Comparison

Tables 11 and 12 show the predictive performance of classifiers built using coverage-based and lazy-based selection, respectively. For all experiments, varying the mining and rule selection methods, equivalent results were observed when analyzing the accuracy, balanced accuracy, and F1-score criteria. The biggest difference between the methods was due to the mining runtime of rules, which presents CI mining as significantly less expensive.

Tables 13 and 14 present the number of instances of the test dataset incorrectly classified by each classifier and the number of failures in common instances between them. Note that both methods almost always make mistakes for the same instances.

The reason why the methods consistently fail on the same instances is explained in Table 15. This table presents the number of instances where both the CI-based and MG-based methods apply rules from the same equivalence class during the prediction stage. In most cases, the rule used by the CI-based method to classify a test instance

Table 10: Number of rules in the classifier that evaluated True at least once when predicting labels for the test set in the experiments involving lazy selection and number of rules common to CI and MG.

	Applied Rules - Lazy					
Dataset	CI	%CI	MG	%MG	Matching	%Match
Adult	686	1%	685	1%	0	0%
Bank	1079	3%	1079	3%	652	60%
Breast	16	0%	16	0%	16	100%
Car	48	2%	47	2%	43	91%
Chess	155	1%	155	1%	58	37%
Contraceptive	97	2%	97	2%	65	67%
CovType	1486	14%	1486	14%	1214	82%
Diabetic	39	1%	39	1%	0	0%
German	66	0%	67	0%	1	1%
Internet	89	2%	89	0%	52	58%
Iris	6	10%	6	8%	1	17%
Kidney	6	0%	6	0%	4	67%
Magic	617	4%	617	4%	375	61%
Mushroom	40	0%	40	0%	0	0%
Nursery	222	2%	222	2%	217	98%
PageBlocks	97	0%	88	0%	65	74%
PenDigits	396	0%	384	0%	300	78%
Shoppers	280	1%	280	1%	0	0%
TicTacToe	17	0%	18	0%	16	89%
Wine	6	0%	6	0%	3	50%

belongs to the same equivalence class as the rule applied by the MG-based method. Notably, this pattern holds for over 90% of the instances across all datasets.

Table 16 presents the p-values from the Wilcoxon signed-rank test used to compare the predictive performance of classifiers built on MGs and CIs. As anticipated from the prior analyses, we observe no statistically significant difference in performance between the two types of classifiers in nearly all cases. Interestingly, in the few cases where a statistically significant difference was detected (considering a significance level of $\alpha = 0.05$), the actual numerical difference in accuracy, balanced accuracy, or F1-score remained marginal. Moreover, contrary to the common expectation that MG-based classifiers would perform better due to their generality, classifiers built on CIs slightly outperformed those based on MGs in these specific cases.

5.3 Rule Length Comparison

An analysis was also carried out to examine the distribution of rule lengths (i.e., the number of items in each rule) for: the mined rules, the rules selected for the classifier, and the applied rules (those that evaluated *True* at least once when predicting labels for the test set). Figures 2 to 61 illustrate this distribution, with the x-axis representing the number of items in the rule antecedent and the y-axis representing the number of rules with that length.

From Figures 2 to 21, we can observe the length distribution of the rules mined by each rule mining algorithm. It can be seen that the peaks for both are close for

Table 11: Performance of the classifiers when using the coverage rule selection method in conjunction with the CI and MG mining approaches, considering the metrics of accuracy, balanced accuracy, and F1-score.

Dataset	CI + Coverage						MG + Coverage					
	Accuracy		Balanced Accuracy		F1-Score		Accuracy		Balanced Accuracy		F1-Score	
	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
Adult	84.44%	0.0045	68.95%	0.0097	0.8201	0.0065	84.44%	0.0045	68.95%	0.0096	0.8201	0.0065
Bank	88.46%	0.0008	50.79%	0.0035	0.8325	0.0020	88.46%	0.0007	50.79%	0.0031	0.8325	0.0018
Breast	95.56%	0.0207	95.07%	0.0224	0.9556	0.0205	95.56%	0.0207	95.07%	0.0224	0.9556	0.0205
Car	92.01%	0.0197	74.35%	0.0797	0.9171	0.0214	92.07%	0.0233	75.41%	0.0766	0.9183	0.0254
Chess	92.99%	0.0146	93.04%	0.0144	0.9299	0.0146	92.96%	0.0162	93.00%	0.0159	0.9296	0.0162
Contraceptive	51.79%	0.0307	47.41%	0.0347	0.5023	0.0325	51.73%	0.0348	47.49%	0.0407	0.5030	0.0376
CovType	71.14%	0.0009	55.06%	0.0038	0.6967	0.0010	71.14%	0.0009	55.06%	0.0036	0.6967	0.0010
Diabetic	66.90%	0.0427	66.23%	0.0432	0.6633	0.0452	66.81%	0.0412	66.14%	0.0417	0.6625	0.0437
German	70.20%	0.0481	60.14%	0.0596	0.6834	0.0525	70.60%	0.0335	59.86%	0.0446	0.6834	0.0428
Internet	97.68%	0.0046	93.36%	0.0169	0.9763	0.0048	97.68%	0.0043	93.54%	0.0161	0.9764	0.0045
Iris	92.67%	0.0696	92.67%	0.0696	0.9247	0.0714	92.67%	0.0696	92.67%	0.0696	0.9247	0.0714
Kidney	99.75%	0.0075	99.67%	0.0100	0.9975	0.0076	99.75%	0.0000	99.67%	0.0000	0.9975	0.0000
Magic	83.89%	0.0054	78.96%	0.0060	0.8309	0.0056	83.89%	0.0056	78.96%	0.0067	0.8309	0.0060
Mushroom	100.00%	0.0000	100.00%	0.0000	1.0000	0.0000	100.00%	0.0000	100.00%	0.0000	1.0000	0.0000
Nursery	95.15%	0.0043	79.05%	0.0711	0.9486	0.0047	95.13%	0.0044	79.02%	0.0691	0.9484	0.0046
PageBlocks	95.94%	0.0050	65.64%	0.0701	0.9564	0.0062	94.57%	0.0052	67.44%	0.0673	0.9537	0.0059
PenDigits	92.93%	0.0061	92.89%	0.0063	0.9297	0.0061	92.70%	0.0053	92.66%	0.0054	0.9274	0.0053
Shoppers	88.04%	0.0068	63.45%	0.0196	0.8534	0.0101	88.05%	0.0069	63.54%	0.0190	0.8538	0.0100
Tic	100.00%	0.0000	100.00%	0.0000	1.0000	0.0000	100.00%	0.0106	100.00%	0.0149	1.0000	0.0108
Wine	97.22%	0.0373	97.35%	0.0344	0.9716	0.0384	98.30%	0.0272	98.57%	0.0273	0.9830	0.0274

Table 12: Performance of the classifiers when using the lazy rule selection method in conjunction with the CI and MG mining approaches, considering the metrics of accuracy, balanced accuracy, and F1-score.

Dataset	CI + Lazy						MG + Lazy					
	Accuracy		Balanced Accuracy		F1-Score		Accuracy		Balanced Accuracy		F1-Score	
	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev	Mean	Dev
Adult	84.43%	0.0045	68.91%	0.0097	0.8198	0.0065	84.43%	0.0045	68.91%	0.0096	0.8198	0.0065
Bank	88.46%	0.0008	50.80%	0.0035	0.8326	0.0020	88.45%	0.0006	50.77%	0.0029	0.8324	0.0016
Breast	95.56%	0.0207	95.07%	0.0224	0.9556	0.0205	95.71%	0.0192	95.28%	0.0209	0.9570	0.0190
Car	91.14%	0.0187	73.66%	0.0712	0.9070	0.0204	91.09%	0.0239	74.31%	0.0737	0.9068	0.0264
Chess	93.27%	0.0147	93.31%	0.0145	0.9327	0.0147	93.21%	0.0158	93.24%	0.0156	0.9321	0.0158
Contraceptive	51.52%	0.0230	47.31%	0.0288	0.5006	0.0263	51.66%	0.0295	47.55%	0.0368	0.5030	0.0336
CovType	71.23%	0.0009	55.04%	0.0038	0.6977	0.0011	71.23%	0.0009	55.04%	0.0036	0.6977	0.0010
Diabetic	67.07%	0.0422	66.38%	0.0426	0.6647	0.0446	66.98%	0.0408	66.30%	0.0411	0.6640	0.0431
German	71.60%	0.0480	57.90%	0.0604	0.6729	0.0535	71.50%	0.0436	58.40%	0.0558	0.6767	0.0522
Internet	97.65%	0.0060	93.43%	0.0192	0.9761	0.0062	97.62%	0.0056	93.32%	0.0189	0.9757	0.0058
Iris	92.67%	0.0696	92.67%	0.0696	0.9247	0.0714	92.67%	0.0696	92.67%	0.0696	0.9247	0.0714
Kidney	100.00%	0.0075	100.00%	0.0100	1.0000	0.0076	99.75%	0.0075	99.67%	0.0100	0.9975	0.0076
Magic	84.00%	0.0056	79.24%	0.0062	0.8326	0.0058	84.01%	0.0054	79.22%	0.0065	0.8326	0.0057
Mushroom	100.00%	0.0000	100.00%	0.0000	1.0000	0.0000	100.00%	0.0000	100.00%	0.0000	1.0000	0.0000
Nursery	95.13%	0.0039	78.75%	0.0694	0.9483	0.0043	95.13%	0.0041	78.68%	0.0686	0.9483	0.0044
PageBlocks	96.02%	0.0071	65.48%	0.0748	0.9563	0.0072	95.49%	0.0076	67.64%	0.0745	0.9532	0.0066
PenDigits	93.38%	0.0064	93.35%	0.0065	0.9337	0.0064	92.95%	0.0068	92.90%	0.0069	0.9296	0.0068
Shoppers	88.00%	0.0069	63.36%	0.0194	0.8530	0.0101	88.00%	0.0070	63.43%	0.0190	0.8532	0.0100
Tic	98.54%	0.0000	97.91%	0.0000	0.9853	0.0000	98.85%	0.0098	98.36%	0.0139	0.9885	0.0099
Wine	96.67%	0.0260	96.71%	0.0218	0.9664	0.0259	98.30%	0.0260	98.57%	0.0218	0.9830	0.0259

Table 13: Number of misclassified instances of the test dataset when using the coverage rule selection method in conjunction with the CI and MG mining approaches.

	Errors with Coverage Selection			
Dataset	# Test Instances	# CI Errors	# MG Errors	# Matching Errors
Adult	4884	760	760	760
Bank	4521	522	522	522
Breast	69	3	3	3
Car	172	14	14	13
Chess	319	22	23	22
Contraceptive	147	71	71	70
CovType	58101	16771	16771	16771
Diabetic	115	38	38	38
German	100	30	29	28
Internet	327	8	8	7
Iris	15	1	1	1
Kidney	40	0	0	0
Magic	1902	307	307	306
Mushroom	812	0	0	0
Nursery	1296	63	63	63
PageBlocks	547	22	30	21
PenDigits	1099	78	80	76
Shoppers	1233	148	147	147
TicTacToe	95	1	1	1
Wine	17	1	0	0

Table 14: Number of misclassified instances of the test dataset when using the lazy rule selection method in conjunction with the CI and MG mining approaches.

	Errors with Lazy Selection			
Dataset	# Test Instances	# CI Errors	# MG Errors	# Matching Errors
Adult	4884	761	761	761
Bank	4521	522	522	522
Breast	69	3	3	3
Car	172	15	15	15
Chess	319	22	22	22
Contraceptive	147	71	71	71
CovType	58101	16715	16715	16714
Diabetic	115	38	38	38
German	100	28	29	27
Internet	327	8	8	7
Iris	15	1	1	1
Kidney	40	0	0	0
Magic	1902	304	304	303
Mushroom	812	0	0	0
Nursery	1296	63	63	63
PageBlocks	547	22	25	20
PenDigits	1099	73	78	68
Shoppers	1233	148	148	148
TicTacToe	95	1	1	1
Wine	17	1	0	0

Table 15: Number of test instances where the CI-based and MG-based methods applied rules from the same equivalence class during prediction.

Dataset	#Test Instances	Coverage		Lazy	
		#Instances	%Instances	#Instances	%Instances
Adult	4884	4880	99.9%	4880	99.9%
Bank	4521	4507	99.7%	4506	99.7%
Breast	69	69	100.0%	69	100.0%
Car	172	166	96.5%	166	96.5%
Chess	319	315	98.7%	315	98.7%
Contraceptive	147	141	95.9%	140	95.2%
CovType	58101	58098	100.0%	58096	100.0%
Diabetic	115	113	98.3%	113	98.3%
German	100	83	83.0%	88	88.0%
Internet	327	325	99.4%	325	99.4%
Iris	15	15	100.0%	15	100.0%
Kidney	40	37	92.5%	37	92.5%
Magic	1902	1895	99.6%	1894	99.6%
Mushroom	812	809	99.6%	809	99.6%
Nursery	1296	1282	98.9%	1282	98.9%
PageBlocks	547	522	95.4%	522	95.4%
PenDigits	1099	1049	95.5%	1049	95.5%
Shoppers	1233	1225	99.4%	1225	99.4%
TicTacToe	95	94	98.9%	94	98.9%
Wine	17	17	100.0%	17	100.0%

most cases, with a difference of only 2 or 3 items. The greatest divergence seems to be for the Mushroom case, which mined mostly MGs with around 5 items and CIs with around 10 items.

We observed that the length of the itemsets extracted by either method is quite small compared to the total number of items in each dataset. For example, for the Internet dataset, which has 1587 items, itemsets with a maximum of 45 items were mined. The Pendigits dataset contains 165 items, but the longest mined itemset contains only 13 items. Even for the datasets that, among all of them, mined itemsets with more items, such as Internet and Kidney, with a maximum of 45 and 24 items, respectively, we see a preference for shorter rules in the prediction phase, where rules with a maximum of 9 and 4 items in the antecedent, respectively, were applied. This happens even when confidence and support are used as a ranking metric, i.e., without purposely favoring the shortest rules. This occurs because highly specific rules are penalized in the ranking, which prioritizes higher confidence and support. For instance, consider a CAR $X \rightarrow c$ with 100% confidence, where X is a CI. Any more specific rule derived from X maintains the same confidence but has lower support, making it less favorable in our ranking.

Table 16: The p-values from the Wilcoxon signed-rank test used to compare the predictive performance of classifiers built on MGs and CIs.

p-values from the Wilcoxon signed-rank test

Dataset	Coverage			Lazy		
	Accuracy	Balanced Accuracy	F1-Score	Accuracy	Balanced Accuracy	F1-Score
Adult	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
Bank	1.00000	1.00000	1.00000	0.15272	0.07593	0.07613
Breast	1.00000	1.00000	1.00000	0.57910	0.57910	0.57910
Car	0.91769	0.64409	0.83836	0.71756	0.95925	0.91876
Chess	0.57910	0.57910	0.57910	0.30458	0.30494	0.30494
Contraceptive	0.83784	1.00000	1.00000	0.60558	0.47235	0.44161
CovType	0.63288	0.71103	0.46521	**0.03887**	**0.03887**	**0.02097**
Diabetic	0.57910	0.95694	0.95694	0.57910	0.95694	0.95694
German	0.41023	0.47525	1.00000	0.91817	0.62500	0.69531
Internet	1.00000	0.79757	0.87769	0.46399	0.40393	0.46521
Iris	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
Kidney	1.00000	1.00000	1.00000	0.57910	0.57910	0.57910
Magic	0.95898	0.95909	0.95909	0.68250	0.75977	0.91881
Mushroom	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
Nursery	0.30494	0.63384	0.56044	0.91668	0.71756	0.87769
PageBlocks	**0.00195**	0.43164	**0.04883**	**0.00589**	0.37500	**0.01953**
PenDigits	**0.04614**	**0.04883**	**0.02734**	**0.00391**	**0.00391**	**0.00391**
Shoppers	0.60093	0.33130	0.33162	1.00000	0.64343	0.64430
Tic	1.00000	1.00000	1.00000	0.15272	0.15272	0.15301
Wine	0.46399	0.32130	0.40393	0.29846	0.14712	0.30138

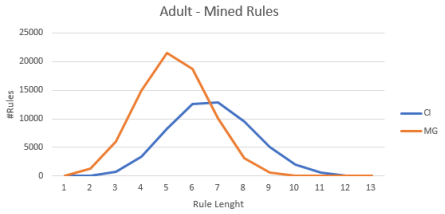


Fig. 2: Number of items in mined rules - Adult dataset

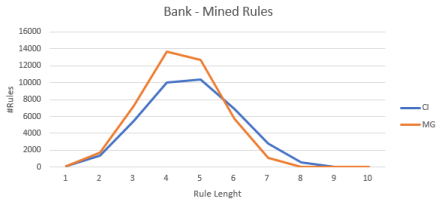


Fig. 3: Number of items in mined rules - Bank dataset



Fig. 4: Number of items in mined rules - Breast dataset



Fig. 5: Number of items in mined rules - Car dataset



Fig. 6: Number of items in mined rules - Chess dataset



Fig. 7: Number of items in mined rules - Contraceptive dataset

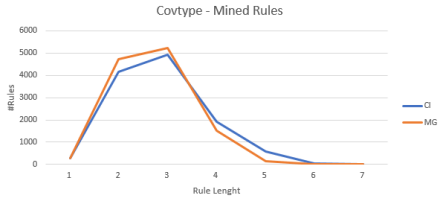


Fig. 8: Number of items in mined rules - Covtype dataset

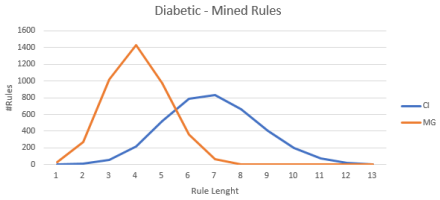


Fig. 9: Number of items in mined rules - Diabetic dataset

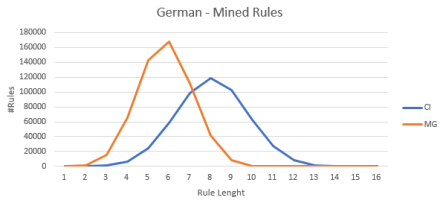


Fig. 10: Number of items in mined rules - German dataset

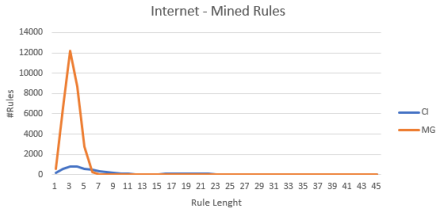


Fig. 11: Number of items in mined rules - Internet dataset

6 Discussion

In this study, we investigated the impact of rule generality versus specificity on associative classifiers by comparing models built from MGs with those built from CIs. Our analysis focused on four core RQs. Below, we discuss our findings, drawing out practical implications for each RQ.

RQ1 – Predictive performance

A central question in AC is whether rule specificity affects predictive performance. It is often assumed that general rules, such as those based on MGs, generalize better and thus lead to more accurate predictions. However, our results do not support this hypothesis.

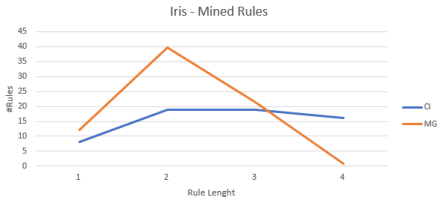


Fig. 12: Number of items in mined rules - Iris dataset

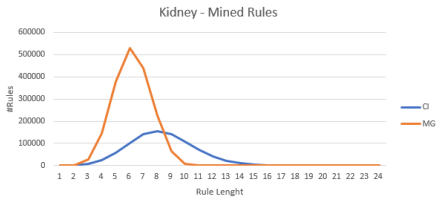


Fig. 13: Number of items in mined rules - Kidney dataset



Fig. 14: Number of items in mined rules - Magic dataset

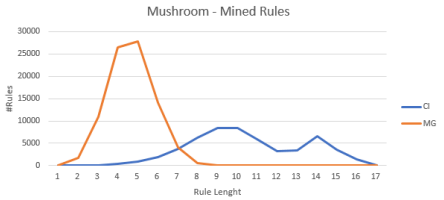


Fig. 15: Number of items in mined rules - Mushroom dataset



Fig. 16: Number of items in mined rules - Nursery dataset



Fig. 17: Number of items in mined rules - Pageblocks dataset

When all other variables were held constant, models using MGs and models using CIs achieved nearly identical results. Across both selection strategies –coverage-based and lazy-based – we observed no statistically significant difference in predictive performance (measured by accuracy, balanced accuracy, and F1-score) between classifiers built on MGs and those built on CIs. In the few cases where statistically significant differences were found, the actual performance gains were marginal. In fact, classifiers based on CIs slightly outperformed those based on MGs in these specific cases, contrary to the expectation that general rules would be more effective.

A deeper look into classification errors further reinforces this finding. When we examined which test instances were misclassified by each classifier, we found that MG-based and CI-based models made nearly identical errors, frequently misclassifying the same instances.

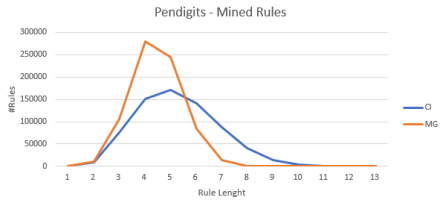


Fig. 18: Number of items in mined rules - Pendigits dataset



Fig. 19: Number of items in mined rules - Shoppers dataset

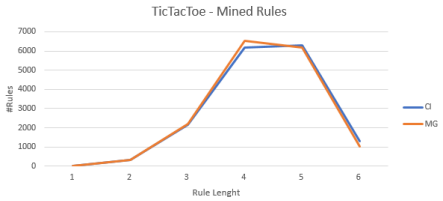


Fig. 20: Number of items in mined rules - TicTacToe dataset

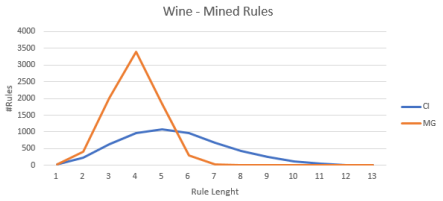


Fig. 21: Number of items in mined rules - Wine dataset

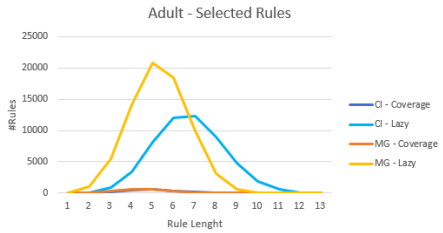


Fig. 22: Number of items in selected rules - Adult dataset



Fig. 23: Number of items in selected rules - Bank dataset



Fig. 24: Number of items in selected rules - Breast dataset



Fig. 25: Number of items in selected rules - Car dataset



Fig. 26: Number of items in selected rules - Chess dataset

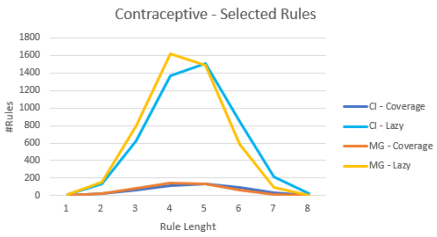


Fig. 27: Number of items in selected rules - Contraceptive dataset

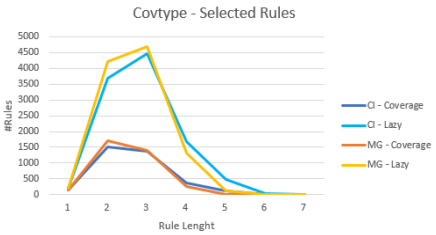


Fig. 28: Number of items in selected rules - Covtype dataset

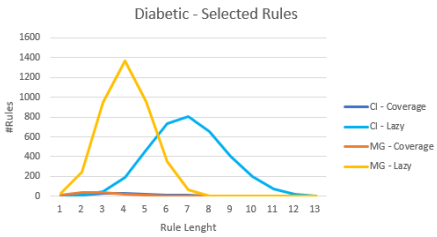


Fig. 29: Number of items in selected rules - Diabetic dataset

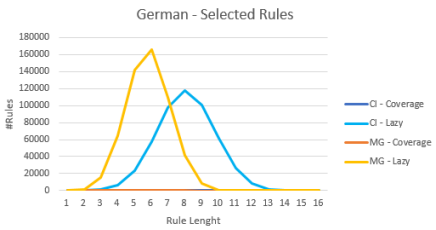


Fig. 30: Number of items in selected rules - German dataset

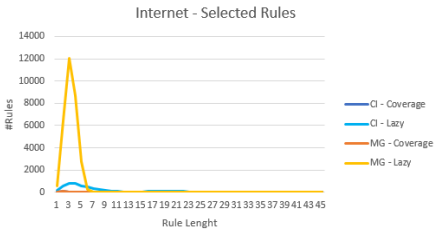


Fig. 31: Number of items in selected rules - Internet dataset

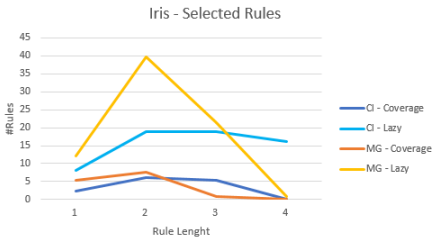


Fig. 32: Number of items in selected rules - Iris dataset

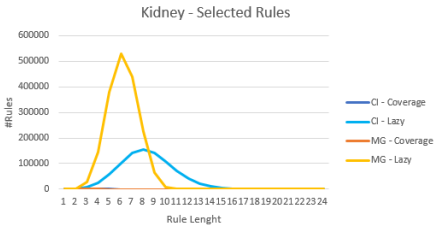


Fig. 33: Number of items in selected rules - Kidney dataset

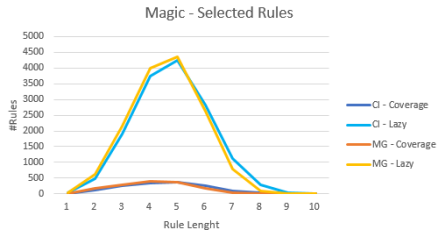


Fig. 34: Number of items in selected rules - Magic dataset

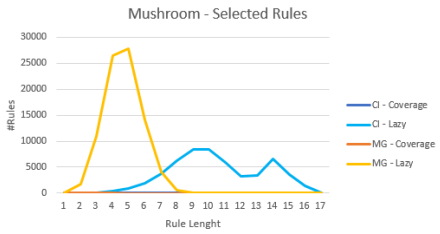


Fig. 35: Number of items in selected rules - Mushroom dataset



Fig. 36: Number of items in selected rules - Nursery dataset



Fig. 37: Number of items in selected rules - Pageblocks dataset

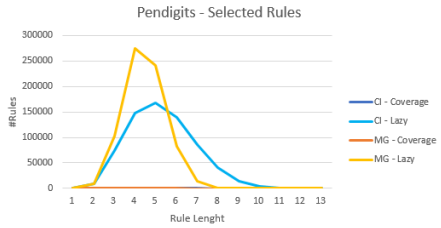


Fig. 38: Number of items in selected rules - Pendigits dataset



Fig. 39: Number of items in selected rules - Shoppers dataset

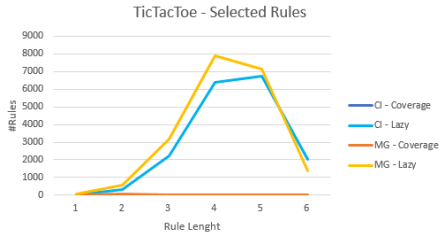


Fig. 40: Number of items in selected rules - TicTacToe dataset

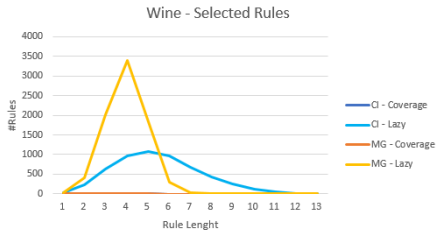


Fig. 41: Number of items in selected rules - Wine dataset



Fig. 42: Number of items in applied rules - Adult dataset



Fig. 43: Number of items in applied rules - Bank dataset



Fig. 44: Number of items in applied rules - Breast dataset

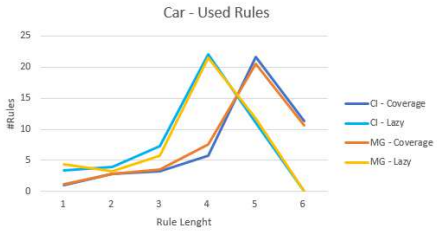


Fig. 45: Number of items in applied rules - Car dataset



Fig. 46: Number of items in applied rules - Chess dataset

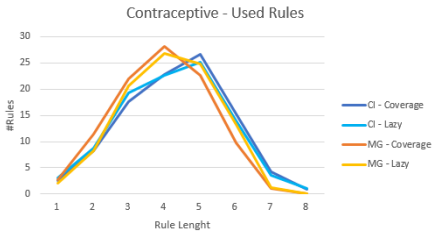


Fig. 47: Number of items in applied rules - Contraceptive dataset

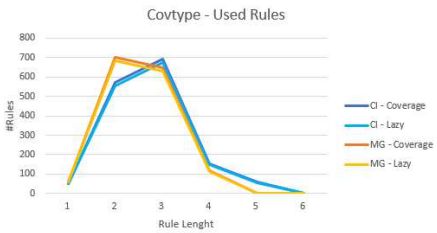


Fig. 48: Number of items in applied rules - Covtype dataset

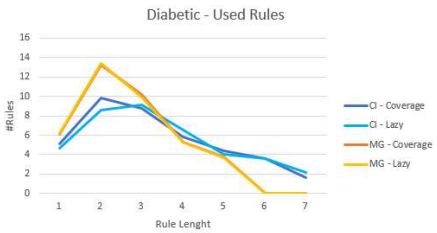


Fig. 49: Number of items in applied rules - Diabetic dataset

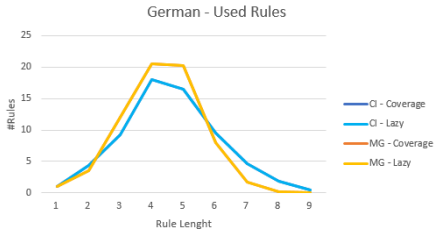


Fig. 50: Number of items in applied rules - German dataset

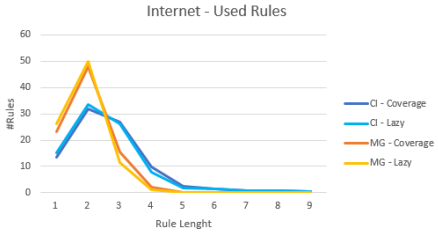


Fig. 51: Number of items in applied rules - Internet dataset

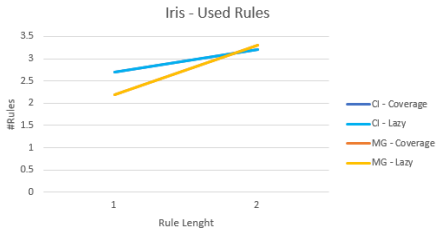


Fig. 52: Number of items in applied rules - Iris dataset

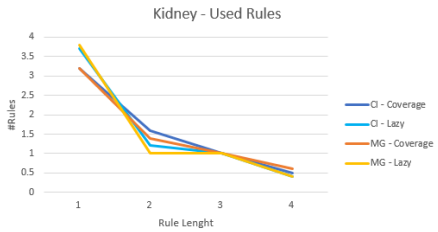


Fig. 53: Number of items in applied rules - Kidney dataset



Fig. 54: Number of items in applied rules - Magic dataset

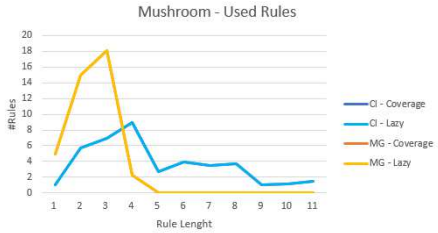


Fig. 55: Number of items in applied rules - Mushroom dataset



Fig. 56: Number of items in applied rules - Nursery dataset



Fig. 57: Number of items in applied rules - Pageblocks dataset



Fig. 58: Number of items in applied rules - Pendigits dataset



Fig. 59: Number of items in applied rules - Shoppers dataset

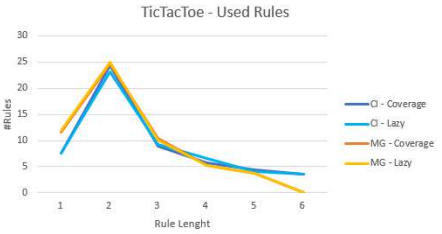


Fig. 60: Number of items in applied rules - TicTacToe dataset

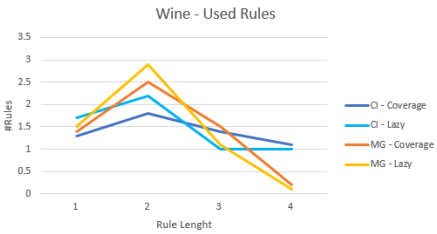


Fig. 61: Number of items in applied rules - Wine dataset

The key to understanding this result lies in the concept of equivalence class and the nature of our rule ranking. Most AC systems, including our experimental setup, prioritize rules based on metrics computed on the transaction set. Since all rules within the same equivalence class cover the exact same transactions, they inherently share the same values for these metrics (confidence and support in our case). Consequently, they are tied in the initial, most critical stages of the ranking process. The choice between a general MG and a specific CI only comes into play as a secondary, tie-breaking criterion.

Our analysis confirms this theoretical underpinning. We observed that for over 90% of the test instances, the CI-based and MG-based classifiers ultimately used rules originating from the same equivalence class to make their predictions. This demonstrates that, in practice, both approaches converge to the same underlying patterns. The debate over whether a shorter or longer rule is "better" becomes meaningless if the classifier-construction strategy guides to the same family of rules. This finding strongly suggests that the feeling that specific rules (CIs) lead to overfitting or poorer generalization may be largely unfounded when standard rule ranking schemes are employed.

RQ2 – Computational efficiency

While predictive performance was largely unchanged, we observed substantial differences in computational efficiency between MG- and CI-based approaches. Mining

MGs is significantly more resource-intensive, both in terms of time and storage. This advantage stems from two fundamental sources.

First, the number of MGs in a dataset can be much larger than the number of CIs. By focusing only on CIs, the mining algorithm starts with a potentially much smaller target, immediately reducing computational and memory overhead.

Second, algorithms designed for CI mining are inherently faster by design. Closed itemsets form a mathematical structure known as a closure system, which creates a well-defined and highly structured search space (a lattice). Efficient algorithms, like the In-Close5 algorithm used in our study, cleverly exploit this structure. They use powerful pruning techniques and closure operators to cleverly navigate the search space, avoiding redundant calculations and eliminating vast sections of the search tree at once. This makes them fundamentally more efficient than algorithms that must enumerate all frequent itemsets or MGs.

Furthermore, this efficiency gain is not confined to the mining stage alone. The disparity in the number of initial rules directly propagates to the classifier construction phase. A larger initial set of MGs means that both coverage-based and lazy-based selection algorithms have a much larger search space to navigate when building the final model. Therefore, the greater the difference between the number of MGs and CIs for a given dataset, the more pronounced the downstream computational savings become during the classifier-building process.

RQ3 – Classifier characteristics

The choice between MGs and CIs did, as expected, influence the final characteristics of the classifiers, particularly their size (number of rules) and average rule length.

Regarding classifier size, when using a coverage-based selection method, the final classifiers had a very similar number of rules. This is another consequence of the rule ranking logic: since both approaches identify the same high-confidence, high-support equivalence classes, they select a comparable number of rules to cover the training data. For lazy-based selection, which retains all non-harmful rules, the MG-based classifiers were naturally larger, as they started from a much larger initial pool of candidate rules.

Regarding rule length, MGs are, by definition, shorter than or equal to their corresponding CIs. While our results reflected this, the observed difference in length was less dramatic than one might expect. Even in large datasets with many items, such as the Internet dataset (1587 items), mined itemsets had moderate sizes, with a maximum of 45 items, and applied rules typically containing 9 or fewer items. Several factors might uphold this result, such as the support threshold and the average transaction length.

Trasierras et al. (2024) showed how the choice of different minimum support thresholds may produce very dissimilar data subsets, i.e., data heterogeneity is not an immutable data characteristic. Thus, had we selected different thresholds or different datasets, such as those in (Dam et al., 2022; Veroneze et al., 2024), more pronounced differences may have emerged. Our experiments highlight the context-sensitive nature of these design decisions and emphasize that classifier characteristics are shaped not only by the choice of MGs versus CIs, but also by interactions with data properties, constraint thresholds, and ranking heuristics.

We emphasize that by prioritizing high support, the ranking naturally disfavors extremely long, overly specific rules, as they tend to cover fewer transactions. This creates a soft cap on specificity, pulling both MG and CI-based classifiers toward moderately sized rules.

RQ4 – Practical guidance

Synthesizing, these findings lead to a clear and actionable conclusion. The long-held practice of using general-purpose miners like Apriori, which mines all frequent itemsets only to discard most of them, is indubitably inefficient. Our work provides strong empirical evidence for a paradigm shift in the rule-mining stage of AC methods.

Our primary recommendation is for practitioners to resort, by default, to efficient, dedicated CI-mining algorithms. The significant computational advantages –faster runtimes and lower memory usage – come at virtually no cost to predictive performance. This makes the CI-based approach the superior choice for most standard classification tasks.

We recommend prioritizing general rules and using algorithms that mine MGs only when the application has a specific requirement for minimal rule length. Such scenarios include, for instance:

- High interpretability: when classifiers must be easily read, understood, or even memorized by human experts.
- Embedded feature selection: when the primary goal is not just prediction but also identifying the smallest possible set of attributes that are collectively predictive, as this provides deeper insight into feature relevance (Veroneze et al., 2024).

For all other cases, the evidence suggests that the AC community can embrace the efficiency of CI-mining without compromising the effectiveness of the resulting classifiers.

7 Conclusion

This paper addressed a fundamental yet underexplored design choice in associative classification (AC): whether classifiers should be built from general rules derived from minimal generators (MGs) or from specific rules derived from closed itemsets (CIs). While this distinction is often treated as a matter of intuition or tradition, we showed that it can and should be examined empirically, under controlled conditions that isolate rule generality from other algorithmic factors. Note that our analysis is independent of the algorithm used to extract MGs or CIs, provided the algorithm is correct and efficient, since all sound MG miners produce the same set of minimal generators, and all sound CI miners produce the same set of closed itemsets.

Through an extensive experimental study on twenty diverse datasets, we demonstrated that the choice between MG-based and CI-based rules has little to no impact on predictive performance when standard rule-ranking and selection strategies are employed. Across multiple evaluation metrics, class distributions, and classifier construction strategies, both approaches achieved nearly identical results. This finding is explained by the equivalence-class structure of itemsets: rules derived from MGs and

CIs originating from the same transaction sets are indistinguishable to ranking criteria such as support and confidence. As a result, classifiers built exclusively from MGs and exclusively from CIs tend to rely on the same underlying patterns and even commit errors on the same instances.

In contrast, the computational implications of this choice are substantial. Our results show that CI-based approaches offer significant advantages in both rule-mining time and classifier-construction cost. By exploiting the mathematical properties of closure systems and by operating on a far more compact search space, dedicated CI-mining algorithms dramatically reduce the computational burden without sacrificing classification effectiveness. These gains propagate downstream, simplifying classifier construction and improving scalability to larger or denser datasets.

Beyond performance and efficiency, we analyzed how rule generality affects classifier characteristics – number of rules and average rule length. With coverage-based selection, both approaches produced classifiers of similar size. In contrast, lazy-based selection led to larger MG-based classifiers, as it usually starts from a larger initial pool of candidate rules and retains all non-harmful rules. While MG-based classifiers naturally contain shorter rules, the difference was smaller than anticipated. Even in large datasets, the CI-based rules remained relatively short.

Taken together, our results provide strong empirical support for a shift toward CI-based mining as the default strategy in AC. CIs constitute a lossless and compact representation of the rule space, and leveraging them directly enables more efficient learning pipelines without compromising predictive power.

We do not argue that MGs are without value. In applications where extreme conciseness or embedded feature selection is paramount, MG-based approaches remain appropriate. However, in the absence of such explicit constraints, our evidence suggests that the widespread reliance on general-purpose frequent itemset miners, or the focus on the most general rules, entails unnecessary computational expense.

Future work will systematically analyze which dataset characteristics – such as item cardinality, transaction length distribution, density, and support thresholds – lead to greater divergences between CI-based and MG-based associative classifiers, with the aim of better understanding when and why CIs and MGs could yield substantially different models.

Acknowledgements

This work was supported by the National Council for Scientific and Technological Development (CNPq), Brazil, with reference numbers 141274/2018-2 and 308793/2022-6. This work was also supported by the São Paulo Research Foundation (FAPESP) – grants 2017/21174-8 and 2020/00123-9. This work was also partially supported by the AIDE project funded by the Belgian SPF BOSA under the program "Financing of projects for the development of artificial intelligence in Belgium" with reference number 06.40.32.33.00.10.

Declarations

Funding

This work was supported by the National Council for Scientific and Technological Development (CNPq), Brazil, with reference numbers 141274/2018-2 and 308793/2022-6. The Federal Institute of Education, Science and Technology of São Paulo also supported this research. This work was also supported by the São Paulo Research Foundation (FAPESP) – grants 2017/21174-8 and 2020/00123-9. This work was also partially supported by the AIDE project funded by the Belgian SPF BOSA under the program "Financing of projects for the development of artificial intelligence in Belgium" with reference number 06.40.32.33.00.10.

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

References

Abdelhamid N, Thabtah F (2014) Associative classification approaches: review and comparison. Journal of Information & Knowledge Management 13(03):1450027

Aggarwal CC, Han J (eds) (2014) Frequent Pattern Mining. Springer Cham, Berlin, Germany

Agrawal R, Srikant R, et al (1994) Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, VLDB, vol 1215. Morgan Kaufmann, Santiago, Chile, pp 487–499

Alwidian J, Hammo B, Obeid N (2016) Enhanced cba algorithm based on apriori optimization and statistical ranking measure. In: Proceeding of 28th international business information management association (IBIMA) conference on vision, vol 2020. IBIMA, New York, USA, pp 4291–4306

Alwidian J, Hammo BH, Obeid N (2018) Wcba: Weighted classification based on association rules algorithm for breast cancer disease. Applied Soft Computing 62:536–549

Andrews S (2018) A new method for inheriting canonicity test failures in Close-by-One type algorithms. In: The 14th International Conference on Concept Lattices and Their Applications. Springer, Kraków, Poland, pp 255–266

Aoga JO, Guns T, Nijssen S, et al (2018) Finding probabilistic rule lists using the minimum description length principle. In: International Conference on Discovery Science. Springer, Groningen, Netherlands, pp 66–82

Baralis E, Chiusano S (2004) Essential classification rule sets. ACM Transactions on Database Systems (TODS) 29(4):635–674

Baralis E, Garza P (2002) A lazy approach to pruning classification rules. In: 2002 IEEE International Conference on Data Mining, 2002. Proceedings. IEEE, Maebashi City, Japan, pp 35–42

Baralis E, Chiusano S, Garza P (2007) A lazy approach to associative classification. IEEE Trans Knowl Data Eng 20(2):156–171

Chen F, Wang Y, Li M, et al (2014) Principal association mining: An efficient classification approach. Knowl-Based Syst 67:16–25
Clark P, Boswell R (1991) Rule induction with cn2: Some recent improvements. In: Machine Learning—EWSL-91: European Working Session on Learning Porto, Portugal, March 6–8, 1991 Proceedings 5. Springer, Porto, Portugal, pp 151–163
Cohen WW (1995) Fast effective rule induction. In: Machine learning proceedings 1995. Elsevier, California, USA, p 115–123
Dam KHT, Given-Wilson T, Legay A, et al (2022) Packer classification based on association rule mining. Appl Soft Comput 127:109373
Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, Citeseer, vol 93. Morgan Kaufmann, Chambéry, France, pp 1022–1029
Fürnkranz J, Gamberger D, Lavrač N (2012) Foundations of Rule Learning. Springer Science & Business Media
Geng X, Yang Z, Jiao L, et al (2025) Association rule-based classification: A comprehensive review of methodologies and applications. Expert Systems with Applications p 127454
Hadi W, Aburub F, Alhawari S (2016) A new fast associative classification algorithm for detecting phishing websites. Applied Soft Computing 48:729–734
Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. ACM SIGMOD Record 29(2):1–12
Kitchenham B, Charters S, et al (2007) Guidelines for performing systematic literature reviews in software engineering version 2.3. Engineering 45(4ve):1051
Konecny J, Krajča P (2021) Systematic categorization and evaluation of cbo-based algorithms in fca. Information Sciences 575:265–288
Li J, Li H, Wong L, et al (2006) Minimum description length principle: Generators are preferable to closed patterns. In: Aaai. AAAI Press, Massachusetts, USA, pp 409–414
Li W, Han J, Pei J (2001) Cmar: Accurate and efficient classification based on multiple class-association rules. In: Proceedings 2001 IEEE international conference on data mining, IEEE, pp 369–376
Li X, Qin D, Yu C (2008) ACCF: Associative classification based on closed frequent itemsets. In: 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery, vol 2. IEEE, Jinan, China, pp 380–384
Liu B, Hsu W, Ma Y (1998) Integrating classification and association rule mining. In: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining. AAAI Press, New York, USA, pp 80–86
Liu B, Ma Y, Wong CK (2000) Improving an association rule based classifier. In: European Conference on Principles of Data Mining and Knowledge Discovery. Springer, Lisbon, Portugal, pp 504–509
Makhalova T, Kuznetsov SO, Napoli A (2020) Closure structure: A deeper insight. In: FCA4AI@ ECAI. Springer, Santiago, Chile, pp 45–56
Padillo F, Luna JM, Ventura S (2019) Evaluating associative classification algorithms for big data. Big Data Analytics 4:1–27

Quinlan JR (1990) Learning logical definitions from relations. Machine learning 5:239–266

Szathmary L, Valtchev P, Napoli A, et al (2009) Efficient vertical mining of frequent closures and generators. In: International Symposium on Intelligent Data Analysis. Springer, London, United Kingdom, pp 393–404

Thabtah F (2007) A review of associative classification mining. The Knowledge Engineering Review 22(1):37–65

Thabtah F, Cowling P, Peng Y (2005) MCAR: multi-class classification based on association rule. In: The 3rd ACS/IEEE International Conference onComputer Systems and Applications, 2005. IEEE, Amman, Jordan, p 33

Trasierras AM, Luna JM, Fournier-Viger P, et al (2024) Data heterogeneity's impact on the performance of frequent itemset mining algorithms. Information Sciences 678:120981

Veroneze R, Bertrand Van Ouytsel CH, Dam KHT, et al (2024) Feature selection for packer classification based on association rule mining. Engineering Applications of Artificial Intelligence 137:109083. https://doi.org/https://doi.org/10.1016/j.engappai.2024.109083, URL https://www.sciencedirect.com/science/article/pii/S0952197624012417

Wedyan S (2014) Review and comparison of associative classification data mining approaches. International Journal of Industrial and Manufacturing Engineering 8(1):34–45

Yin X, Han J (2003) CPAR: Classification based on predictive association rules. In: Proceedings of the 2003 SIAM international conference on data mining. SIAM, California, USA, pp 331–335

Zaki MJ, Gouda K (2003) Fast vertical mining using diffsets. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, Washington, D.C., USA, pp 326–335

Zaki MJ, Hsiao CJ (2002) CHARM: An efficient algorithm for closed itemset mining. In: Proceedings of the 2002 SIAM International Conference on Data Mining, vol 2. SIAM, Pennsylvania, USA, pp 457–473

Zaki MJ, Parthasarathy S, Ogihara M, et al (1997) New algorithms for fast discovery of association rules. In: Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD), vol 97. AAAI Press, Florida, USA, pp 283–286