

Article

On Discovering Discriminative Itemsets Based on Detecting Frequent Itemsets in Succession

Aristotelis Kompothrekas *  and Basilis Boutsinas 

Management Information Systems & Business Intelligence Lab, University of Patras, 26504 Patras, Greece; vutsinas@upatras.gr

* Correspondence: a.kompothrekas@upatras.gr

Abstract

Discovery of Association Rules is one of the most common Data Mining techniques. Contrast data mining is a focused data mining research area for discovering interesting contrast patterns that state the significant differences between datasets, i.e., frequent itemsets in one dataset with much higher frequencies than the same itemsets in other datasets. In this paper, we present a new algorithm for discovering discriminative itemsets based on detecting items in succession. Association Rules in Succession consider proximity of transactions. We present extensive empirical results that exhibit the performance of the proposed algorithm with respect to similar ones.

Keywords: association rules; data mining; discriminative patterns; frequent itemsets; multi-dataset analysis

1. Introduction

Nowadays, the rapidly evolving information technology forces commercial and scientific organizations to acquire and keep huge volume of data. The analysis of this data in order to extract interesting knowledge has been an attractive research topic yet a difficult task.

Discovery of Association Rules is one of the most common Data Mining techniques (see [1–3] for a survey). The field was initiated by Agrawal et al. [4], who introduced the problem of discovering correlations between itemsets in large transactional databases. Their work inspired a broad range of extensions and refinements (e.g., [5–12]). This work informed later studies for the Apriori algorithm [13], which introduced the downward closure property to effectively prune the search space. Subsequent research focused on addressing the performance bottlenecks inherent in Apriori-like approaches, particularly the costly generation of candidate sets. Innovations such as the DHP (Direct Hashing and Pruning) algorithm by Park et al. [14] utilized hash-based techniques to filter ineffective candidates, while the Dynamic Itemset Counting (DIC) algorithm [15] sought to reduce database scans by counting itemsets dynamically. The maintenance of discovered rules also received attention, with algorithms like FUP (Fast Update) [7] enabling efficient updates as new transactions arrived. Efficiency was improved with the FP-growth algorithm [16], which circumvented candidate generation entirely through the use of a compressed FP-tree structure and a divide-and-conquer strategy.

The principles of association mining were naturally extended to sequential data by Agrawal and Srikant [17], and later generalized by the GSP algorithm [8] to accommodate time constraints and taxonomies. Efficiency remained a primary driver, leading



Academic Editor: Yousef Farhaoui

Received: 10 June 2026

Revised: 18 July 2026

Accepted: 20 July 2026

Published: 23 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Pei et al. [18] to develop PrefixSpan, a pattern-growth approach that projects databases based on frequent prefixes to avoid candidate generation. In the realm of event sequences, Mannila et al. [9] established frameworks for discovering “episodes”, partially ordered sets of events occurring in close temporal proximity.

The introduction of the temporal dimension necessitates careful handling of item validity. Ale and Rossi [19] proposed temporal association rules that respect the “lifespan” of items, calculating support only within relevant time intervals. This concept was further sophisticated by Chen et al. [20], who integrated fuzzy set theory to mine fuzzy temporal association rules. A comprehensive categorization of these methods, distinguishing how time is integrated as either an implied or integral component, has been provided by Segura-Delgado et al. [21].

Parallel to the development of sequential and temporal extensions, another major evolution in pattern mining has been the shift toward class-based discrimination. Contrast data mining is a focused data mining research area for discovering interesting contrast patterns that state the significant differences between datasets [22]. For example, discriminative itemsets [23] identify the frequent itemsets in one dataset with much higher frequencies than the same itemsets in other datasets. Discriminative pattern analysis has been applied across various domains. Lee et al. [24] demonstrated its value in classifying trajectories on road networks, highlighting the importance of ordering information. In the domain of recommender systems, Panteli and Boutsinas [25] applied these techniques to the “cold-start” problem, clustering users and extracting discriminative patterns to predict preferences for new users effectively.

While discriminative itemsets mining successfully captures differences across datasets, classical frequency measures often overlook the local concentration of items within transactions. To address this limitation, Association Rules in Succession (ARIS) incorporate transaction proximity into the definition of support [26]. In both the following datasets, as shown in Table 1, AB is a frequent itemset given a minimum support threshold of 0.60. However, in the second dataset, the occurrences of AB are more concentrated within neighboring transactions. Motivated by this observation, in this work we extend Association Rules in Succession (ARIS) from traditional frequent-pattern mining to the discovery of discriminative itemsets across multiple ordered transactional datasets.

Table 1. Dataset 1 and Dataset 2.

TID	Items	TID	Items
TID1	A, B	TID1	A, B
TID2	C, D	TID2	A, B
TID3	A, B	TID3	A, B
TID4	E, F	TID4	C, D
TID5	A, B	TID5	E, F

The proposed work builds upon two well-established research directions: Association Rules in Succession (ARIS), which introduces succession-based support for mining ordered transactional datasets, and discriminative itemset mining, which identifies patterns that distinguish one dataset from others. The main contributions of this work are summarized as follows:

1. We propose a new discriminative itemset mining algorithm by incorporating succession-based support from the ARIS framework.
2. We introduce a succession-based discriminative itemset mining algorithm that exploits not only the frequency of itemsets but also the locality of their occurrences within each dataset.

3. We experimentally evaluate the proposed algorithms on synthetic and real-world datasets, demonstrating their applicability and computational performance.

2. Related Work

Over time, knowledge discovery has evolved from finding frequent itemsets to identifying patterns that differentiate between datasets or classes. This section explores the development of contrast and discriminative pattern mining, together with advances in redundancy control that have helped make modern discriminative methods more effective and interpretable.

2.1. Discriminative and Contrast Pattern Mining

Research gradually shifted from studying simple co-occurrence relationships to identifying patterns that differ significantly across groups. One of the earliest contributions in this direction was the concept of contrast sets, introduced by Bay and Pazzani [27] to uncover meaningful differences between groups. Their STUCCO algorithm combined contrast set mining with chi-square testing to control Type I errors and assess statistical significance. Around the same time, Dong and Li [28] proposed the concept of *Emerging Patterns* (EPs), defined as itemsets whose support changes substantially between datasets. To cope with the potentially large number of such patterns, they introduced compact border-based representations that enabled more efficient storage and analysis.

Due to computational inefficiencies of the traditional “two-step” approach, mining frequent patterns first and then filtering for discriminative patterns, Cheng et al. [29] developed direct mining methodologies. Their DDPMine algorithm integrated feature selection directly into the mining process via a branch-and-bound search on a progressively shrinking FP-tree. This “feature-centered” paradigm proved superior to two-step methods in both speed and accuracy. Further refining this approach, Seyfi et al. [23] introduced DISSparse, which utilizes heuristics to confine the search to potential discriminative subtrees, thereby managing the exponential search space more effectively.

2.2. Discriminative Sequential Mining and Redundancy Control

Recent work has focused on combining sequence mining with discriminative capabilities. Sheng et al. [30] introduced “Minimal Distinguishing Subsequences” (MDS) to identify patterns that are frequent in a positive class but rare in a negative one, utilizing gap constraints for flexibility. A challenge in this domain is redundancy, where a pattern appears discriminative only because it contains a truly discriminative sub-pattern. To mitigate this, Meng et al. [31] proposed “Conditional Discriminative Sequential Patterns,” evaluating discriminative power conditioned on sub-patterns. Furthermore, ensuring statistical validity has become a priority, with Lu et al. [32] integrating multiple hypothesis testing corrections, such as Family-Wise Error Rate (FWER) control, directly into the mining process.

Unlike these approaches, the present work focuses on the discovery process itself and does not incorporate redundancy elimination or statistical hypothesis testing into the mining stage.

3. Problem Definition

We adopt the formalization of discriminative itemset mining proposed by Seyfi et al. [23] and extend it from the two-dataset case to a collection of $k \geq 2$ datasets that share the same alphabet of items. Let Σ be the alphabet of items, and let $\mathcal{D} = \{D_1, D_2, \dots, D_k\}$ be a collection of transactional databases over Σ , where D_i contains n_i transactions. A dataset $D_i \in \mathcal{D}$ is designated as the *target dataset*, while the remaining datasets $\{D_j\}_{j \neq i}$ play the role of *general datasets*.

For an itemset $I \subseteq \Sigma$, let $f_i(I)$ denote the frequency of I in D_i , that is, the number of transactions of D_i that contain I . The corresponding frequency ratio of I in D_i is

$$r_i(I) = \frac{f_i(I)}{n_i}. \quad (1)$$

The discriminative value of itemset I between target dataset D_i and a general dataset D_j is

$$R_{ij}(I) = \frac{r_i(I)}{r_j(I)} = \frac{f_i(I) n_j}{f_j(I) n_i}. \quad (2)$$

The larger $R_{ij}(I)$, the more strongly I discriminates D_i from D_j . By convention, when $f_j(I) = 0$ we set $R_{ij}(I) = \infty$, meaning that I is exclusive to D_i with respect to D_j . Without further restrictions, however, $R_{ij}(I)$ may become arbitrarily large simply because $f_j(I)$ is negligible while $f_i(I)$ itself is also low. Following [23], we therefore couple the discriminative criterion with a minimum-support requirement on the target dataset. Let $\theta > 1$ be a user-defined discriminative level threshold and $\varphi \in (0, 1/\theta)$ a user-defined support threshold.

Definition 1 (Discriminative itemset). Given the collection $\mathcal{D} = \{D_1, \dots, D_k\}$ of datasets with target dataset D_i , a discriminative threshold $\theta > 1$, and a support threshold $\varphi \in (0, 1/\theta)$, an itemset $I \subseteq \Sigma$ is a discriminative itemset for D_i if and only if

$$f_i(I) \geq \varphi \theta n_i \quad \text{and} \quad \min_{j \neq i} R_{ij}(I) \geq \theta. \quad (3)$$

Succession-Based Discriminative Itemset Mining

The definition above formulates discriminative itemset mining over multiple datasets using a class-specific support requirement and a frequency-ratio discrimination criterion. In the conventional setting, the support of an itemset is evaluated solely by counting its occurrences within each dataset, independently of the order in which the transactions appear.

In this work, we extend this formulation by replacing conventional support with succession-based support, as defined in the ARIS framework.

Definition 2 (Succession-Based Discriminative Itemset). Let $D = \{D_1, D_2, \dots, D_k\}$ be a collection of ordered transactional datasets, and let $f_i^{\text{succ}}(I, d)$ denote the frequency in succession of an itemset I in dataset D_i , computed using the ARIS succession-support definition with succession distance d .

An itemset I is said to be succession-based discriminative for dataset D_i if it satisfies

$$f_i^{\text{succ}}(I, d) \geq \varphi n_i$$

and

$$R_{ij}^{\text{succ}}(I, d) = \frac{r_i^{\text{succ}}(I, d)}{r_j^{\text{succ}}(I, d)} \geq \theta, \quad \forall j \neq i,$$

where

$$r_i^{\text{succ}}(I, d) = \frac{f_i^{\text{succ}}(I, d)}{n_i}.$$

In other words, a succession-based discriminative itemset is an itemset whose occurrences are both sufficiently frequent in succession within the target dataset and significantly more concentrated than in every other dataset according to the discrimination threshold θ .

Thus, the proposed formulation evaluates whether an itemset is not only more frequent in one dataset than in the others, but also whether its occurrences are locally concentrated according to the original transaction order. When the succession distance d is sufficiently large, the succession constraint becomes inactive and $f_i^{succ}(I, d)$ coincides with conventional frequency. Therefore, conventional discriminative itemset mining is obtained as a special case of the proposed succession-based formulation.

4. Proposed Algorithm

In this section, we present the proposed algorithm for discriminative pattern mining. The approach builds directly on the Association Rules in Succession (ARIS) framework introduced in [26], which defines support, frequent itemsets and association rules under a proximity constraint over an ordered transactional database. Here we extend that framework to multi-dataset discriminative pattern mining. The approach consists of three steps: data preparation with class markers that encode dataset membership into the transactions, ARIS-Apriori mining in succession on the merged dataset, and discriminative pattern validation using a frequency ratio criterion.

The proposed approach discovers discriminative itemsets through a single-pass strategy on merged data using the ARIS-Apriori algorithm. Rather than mining each dataset independently and comparing results post-hoc, we embed dataset membership directly into the transaction data using class markers. This allows a single execution of ARIS-Apriori to simultaneously discover frequent itemsets in succession and their class associations.

This framework intentionally reuses several components from existing work. Specifically, the definition of succession-based support, the proximity constraint, the frequent-itemset mining procedure, and the ARIS-Apriori algorithm are inherited directly from the original ARIS framework. Likewise, the concept of discriminative itemset mining based on comparing pattern frequencies across multiple datasets has been extensively studied in the literature. The novelty of the present work lies in the integration of these two research directions into a unified framework. In particular, we propose a new discriminative itemset mining algorithm over multiple ordered transactional datasets using succession-based support, introduce succession-based discriminative itemset mining, propose a unified single-pass mining strategy based on merged datasets and class markers, and validate discriminative patterns according to both succession-aware support and frequency-ratio criteria.

4.1. Step 1: Data Preparation with Class Markers

Algorithm 1 presents the preprocessing stage. Each dataset D_i is augmented with a unique class marker Z_i appended to every transaction, encoding dataset membership as an item. Each dataset is sorted independently by its transaction contents, and the sorted datasets are concatenated without global reordering. This preserves dataset boundaries in the merged data: items from the same dataset remain in close proximity, which is significant when ARIS distance constraints are applied.

The class markers serve a dual purpose. First, they enable the mining algorithm to discover associations between items and specific datasets within a single run. Second, since each marker Z_i appears exclusively in the transactions of dataset D_i , any frequent itemset containing Z_i is by construction a pattern associated with that dataset.

For succession-based discriminative mining, the original transaction order is preserved. Lexicographical transaction sorting is applied to impose transactions of the same class to be in succession, thus ARIS could be applied.

Algorithm 1 data_preparation(); Data Preparation with Class Markers**Require:** Datasets $\mathcal{D} = \{D_1, D_2, \dots, D_n\}$ **Ensure:** Merged dataset M , dataset sizes $\{N_1, \dots, N_n\}$

```

1:  $M \leftarrow \emptyset$ 
2: for all  $D_i \in \mathcal{D}$  do
3:    $Z_i \leftarrow "Z" + i$  ▷ Unique class marker
4:   for all transaction  $t \in D_i$  do
5:      $t' \leftarrow \text{Sort}(t \cup \{Z_i\})$  ▷ Append marker and sort items
6:   end for
7:    $D'_i \leftarrow \text{Sort}(\{t' : t \in D_i\})$  ▷ Sort transactions within dataset
8:    $N_i \leftarrow |D'_i|$ 
9:    $M \leftarrow M \parallel D'_i$  ▷ Concatenate (no global sort)
10: end for
11: return  $M, \{N_1, \dots, N_n\}$ 

```

4.2. Step 2: ARIS-Apriori on Merged Data

The merged dataset M is mined using the ARIS-Apriori algorithm, which extends the traditional Apriori candidate generation and pruning by using Algorithm 2 for counting support in succession. The support threshold σ is interpreted as a percentage of each class's own size:

$$s_i = \lfloor \sigma \times N_i \rfloor \quad \text{for each class } D_i \quad (4)$$

The merged dataset is used only to perform a unified ARIS-Apriori mining process with class markers, while the final support and discrimination criteria are evaluated separately for each dataset.

The distance parameter is set to $d = |M|$, which ensures that the successive groupcount mechanism covers the full extent of each dataset's block within the concatenated structure.

Algorithm 2 count_support_in_succession(); Support Counting in Succession

```

1: for all transaction  $t_i \in D$  do
2:    $C_{ti} \leftarrow \text{subset}(C_k, t)$  ▷ Candidates contained in  $t_i$ 
3:   for all candidate  $c = \{c_{i1}, c_{i2}, \dots, c_{ik-1}, c_{ik}\} \in C_{ti}$  do
4:      $p \leftarrow i - c.\text{last}$ 
5:     if  $H(d - p)$  then
6:        $c.\text{groupcount}++$ 
7:        $c.\text{last} \leftarrow i$ 
8:     else
9:       if  $c.\text{groupcount} \geq \text{mingroup}$  then
10:         $c.\text{count} \leftarrow c.\text{count} + c.\text{groupcount}$ 
11:      end if
12:       $c.\text{groupcount} \leftarrow 0$ 
13:    end if
14:  end for
15: end for

```

The ARIS-Apriori algorithm follows the standard Apriori iterative framework, generating candidate itemsets of increasing size from frequent itemsets of the previous level but replaces the simple frequency counting with *count_support_in_succession()*. For each candidate, the algorithm scans the merged data sequentially, tracking the gap p between consecutive occurrences. The Heaviside step function $H(d - p)$ determines whether occurrences are within the distance threshold. Groups of at least *mingroup* proximate occurrences contribute to the support count; isolated occurrences that fail to form a sufficient group are discarded.

A key property is that when $d = |M|$, the distance constraint is satisfied for all inter-occurrence gaps within the merged data, and `count_support_in_succession()` produces support counts identical to standard frequency counting. We verify this equivalence experimentally in Section 5. When $d < |M|$, ARIS adds temporal proximity constraints that favor items appearing in dense clusters of consecutive transactions.

4.3. Step 3: Discriminative Validation

Algorithm 3 identifies discriminative patterns from the frequent itemsets discovered by ARIS-Apriori. First, only itemsets containing exactly one class marker are retained, these represent patterns associated with a specific dataset. Multi-marker itemsets and marker-free itemsets are discarded. The retained itemsets are grouped by their item pattern (the non-marker portion), and the support of each pattern across class markers is compared.

A pattern P is declared *discriminative* for dataset D_i if two conditions hold. Let $\text{sup}(P, Z_j) = |\{t \in D_j : P \subseteq t\}|$ be the count of transactions in D_j that contain P , and let $f_j(P) = \text{sup}(P, Z_j)/N_j$ be the corresponding per-class frequency:

- (a) The support of P in D_i meets that class's own threshold: $\text{sup}(P, Z_i) \geq s_i$
- (b) The frequency in D_i is at least twice the maximum frequency in any other dataset: $f_i(P) \geq 2 \times \max_{j \neq i} f_j(P)$, or P appears in no other dataset (unique pattern)

Algorithm 3 `discr_pattern_validation()`; Discriminative Pattern Validation

Require: Frequent itemsets $\mathcal{F}$ with support counts, per-class thresholds $\{s_1, \dots, s_n\}$, dataset sizes $\{N_1, \dots, N_n\}$

Ensure: Discriminative patterns with class assignments

```

1:                                     ▷ Phase 1: Filter class-associated itemsets
2:  $\mathcal{Z} \leftarrow \{(f, \text{count}) \in \mathcal{F} : f \text{ contains exactly one } Z_i\}$ 
3:
4:                                     ▷ Phase 2: Group by item pattern
5:  $\mathcal{G} \leftarrow \emptyset$                                      ▷ Map: pattern  $\rightarrow \{Z_i: \text{support}\}$ 
6: for all  $(f, \text{count}) \in \mathcal{Z}$  do
7:    $P \leftarrow f \setminus \{Z_i\}$                                      ▷ Item pattern (remove marker)
8:    $Z_i \leftarrow \text{the class marker in } f$ 
9:    $\mathcal{G}[P][Z_i] \leftarrow \text{count}$ 
10: end for
11:
12:                                     ▷ Phase 3: Apply frequency-normalized discriminative criterion
13:  $\mathcal{R} \leftarrow \emptyset$ 
14: for all pattern  $P$ , supports  $\{Z_i : s_i\}$  in  $\mathcal{G}$  do
15:   for all  $Z_i$  with support  $s_i$  do
16:     if  $s_i \geq s$  then
17:        $f_i \leftarrow s_i / N_i$                                      ▷ Target-class frequency
18:        $f_{\max} \leftarrow \max(\{s_j / N_j : j \neq i\})$              ▷ Max frequency in other classes
19:       if  $f_{\max} = 0$  or  $f_i \geq 2 \times f_{\max}$  then
20:          $\mathcal{R} \leftarrow \mathcal{R} \cup \{(P, Z_i, s_i)\}$              ▷ Discriminative pattern for  $D_i$ 
21:       end if
22:     end if
23:   end for
24: end for
25: return  $\mathcal{R}$ 

```

The use of frequencies rather than raw counts prevents large classes from dominating the comparison, which is the correction needed for any imbalanced dataset. On balanced data the two formulations are mathematically equivalent. The discrimination ratio threshold controls the minimum frequency difference required for an itemset to be considered discriminative. In the experiments, a default value of $\theta = 2$ was adopted, requiring that an

itemset be at least twice as frequent in the target dataset as in any other dataset. The influence of this parameter is further investigated in the sensitivity analysis presented in Section 5.8.

Existing discriminative itemset mining algorithms such as DDPMine and DISSparse primarily focus on reducing the search space through specialized pruning strategies or heuristic search techniques. In contrast, the proposed framework does not introduce a new pruning mechanism. Instead, it extends the ARIS framework to discriminative itemset mining by combining succession-based support with a unified merged-dataset mining strategy. Consequently, the contribution of this work lies in extending the mining framework rather than proposing an alternative candidate-generation or pruning algorithm.

4.4. Computational Complexity Analysis

Let $N = \sum_{i=1}^k n_i$ denote the total number of transactions in the merged dataset, where k is the number of datasets and n_i is the number of transactions in dataset D_i . Let m denote the number of distinct items, ℓ the average transaction length, C_r the number of candidate itemsets of size r , and F the total number of frequent itemsets generated by ARIS-Apriori.

The proposed framework consists of three stages: data preparation, ARIS-Apriori mining, and discriminative validation.

Data Preparation

Each transaction is augmented with a class marker and its original transaction position. The items of each transaction are sorted to obtain a canonical representation. Since each transaction contains on average ℓ items, this step requires

$$O(N\ell \log \ell)$$

time. If the transactions of each dataset are additionally sorted lexicographically before merging, the sorting cost is

$$O\left(\sum_{i=1}^k n_i \log n_i\right).$$

Thus, the total preprocessing cost is

$$O\left(N\ell \log \ell + \sum_{i=1}^k n_i \log n_i\right).$$

ARIS-Apriori Mining

The ARIS-Apriori stage follows the classical Apriori level-wise candidate-generation procedure. At level r , the algorithm generates C_r candidate itemsets and scans the merged dataset to count their support in succession. For each transaction, candidate containment is checked and the succession variables of each candidate are updated. Therefore, the counting cost at level r can be expressed as

$$O(NC_r),$$

assuming candidate lookup and containment checking are implemented in the standard Apriori manner. Across all levels, the total ARIS-Apriori mining cost is

$$T_{\text{ARIS}} = O\left(N \sum_r C_r\right).$$

As with classical Apriori, the number of generated candidates may be exponential in the number of distinct items m in the worst case, since up to $2^m - 1$ non-empty itemsets

may become candidates. Therefore, the worst-case complexity of the mining stage is exponential in m . In practice, however, the minimum support threshold and the Apriori downward-closure property substantially reduce the candidate search space.

The succession mechanism does not change this asymptotic complexity. Compared with conventional Apriori support counting, ARIS-Apriori maintains additional state variables for each candidate, such as the last occurrence position, the current group count, and the accumulated succession support. These updates require constant time per candidate occurrence. Therefore, succession-based counting changes the constant factor but not the asymptotic order of the Apriori counting phase.

Discriminative Validation

Let F denote the number of frequent itemsets returned by ARIS-Apriori. The discriminative validation stage groups itemsets according to their non-marker items and evaluates their frequencies across the k datasets. Since each frequent itemset is processed once and compared across all datasets, this stage requires

$$O(Fk)$$

time.

Overall Complexity

The total time complexity of the proposed framework is therefore

$$O\left(N\ell \log \ell + \sum_{i=1}^k n_i \log n_i + N \sum_r C_r + Fk\right).$$

Since candidate generation and support counting dominate the computation, the overall complexity is governed by the ARIS-Apriori mining stage. The discriminative extension adds only a linear validation overhead with respect to the number of frequent itemsets and datasets.

The total space complexity is

$$O(N\ell + \sum_r C_r + Fk),$$

where $O(N\ell)$ accounts for storing the merged transactional dataset, $\sum_r C_r$ accounts for candidate itemsets, and Fk accounts for storing class-specific support information for the frequent itemsets.

5. Experimental Results

We tested the ARIS-Apriori pipeline on six experimental configurations: a synthetic dataset with planted ground truth run at two scales, the UCI Mushroom dataset, the UCI Online Retail dataset, a Greek SME innovation survey, and a sample from the Criteo display advertising dataset. The configurations differ in size (818 to 75,000 transactions), class count (two to four), class balance, transaction density, and item-vocabulary cardinality.

5.1. Experimental Setup

The proposed framework requires three user-defined parameters: the minimum support threshold, the succession distance d , and the discrimination ratio threshold θ . The influence of d and θ is investigated through a dedicated sensitivity analysis presented in Section 5.8. The minimum support threshold plays the same role as in classical Apriori-based mining by controlling the size of the candidate search space. Because the experimental datasets differ substantially in both size and density, the support threshold

was selected individually for each dataset to produce a meaningful number of frequent itemsets. The selected values were determined through preliminary exploratory runs and subsequently remained fixed throughout all experiments.

Every experiment follows the three-step pipeline from Section 4: prepare and merge the class-tagged data (Algorithm *data_preparation()*), run ARIS-Apriori on the merged dataset (Algorithm *count_support_in_succession()*), and apply the $2\times$ frequency ratio criterion to identify discriminative patterns (Algorithm *discr_pattern_validation()*). We used Python 3.9 on a single-core Apple M-series machine throughout. The distance parameter is set equal to the full merged dataset size in the main experiments, which makes support counting equivalent to plain frequency counting and lets us study the discriminative criterion in isolation; Section 5.8 then activates the succession constraint on the Online Retail dataset, whose transactions follow invoice-date order.

Two methodological choices apply uniformly across every dataset:

- Per-class support threshold. The Apriori absolute support for class D_i is $s_i = \lfloor \sigma \cdot N_i \rfloor$, computed against that class's own size. A threshold of $\sigma = 30\%$ on a class of 105 transactions means at least 32; on a class of 7481 transactions it means at least 2244. The merged mining phase is used only to identify candidate itemsets.
- Frequency-normalized $2\times$ criterion. A pattern P is discriminative for class z when its per-class support $frequency\ f_z = \sup_z / |Z|$ satisfies $f_z \geq 2 \cdot \max_{z' \neq z} f_{z'}$ or when $\sup_{z'} = 0$ for every $z' \neq z$.

In the synthetic experiments, all classes contain the same number of transactions. Therefore, using relative frequencies or absolute support counts leads to exactly the same discrimination ratios. We nevertheless report support as a percentage to maintain consistency with the real-world experiments.

The table 2 gives an overview of the six experimental configurations. We present the synthetic validation first to establish algorithmic correctness against ground truth, then four real-world experiments ordered by the difficulty of the discrimination problem (Mushroom: balanced and well-separated; Online Retail: multi-class with sparse, high-cardinality items; Greek SME Survey: imbalanced with substantive minority-class signal; Criteo: imbalanced, with minority-class signal that is real but support-sensitive). A cross-dataset synthesis closes the section.

Table 2. Overview of experimental datasets.

Dataset	Classes	Transactions	Items/Trans	Unique Items	Support Range
Synthetic T10	3	30,000	~21	~1034	5–25%
Synthetic T25	3	75,000	~21	~1034	5–25%
UCI Mushroom	2	8124	22	117	58–78%
Online Retail	4	20,728	variable	~3900	2–5%
Greek SME Survey	3	818	17–32	126	20–40%
Criteo CTR	2	20,000	~8	157	1–10%

5.2. Validation with Synthetic Data

To evaluate the correctness of the algorithm, synthetic datasets with planted discriminative patterns and known ground truth were generated. This setup enables the exact computation of precision, recall, and F1-score, which is not possible with real-world datasets where the complete set of discriminative patterns is unknown.

Each synthetic dataset has three equal-size classes. Every transaction draws 16 items uniformly at random from a pool of 1000 background items (per-item frequency $\approx 1.6\%$), and selected transactions receive additional *planted* items. We defined four categories of planted patterns:

1. Unique patterns (U1–U6): Items that appear exclusively in the target class at 30–40% rates. Sizes range from 2 to 4 items. All six should be detected.
2. Ratio-based patterns (R1–R4): Items present in all classes but at $2.3\times$ – $3.5\times$ higher rate in the target. All four should be detected.
3. Borderline pattern (B1): A $1.67\times$ ratio, below the $2\times$ threshold. This should *not* be detected.
4. Negative controls (N1–N3): Near-equal rates across all classes (within 4 percentage points). These should *not* be detected.

We ran two configurations: T10 (10,000 transactions per class, 30,000 total) and T25 (25,000 per class, 75,000 total), both seeded at 42 for reproducibility. Validation results are shown in Table 3.

Scoring the discovered patterns requires a precise definition because the ten planted discriminative families generate considerably more than ten valid discriminative itemsets. By the downward-closure property, every subset of a planted discriminative itemset is itself discriminative for the same class. For example, if the planted itemset $\{1000, 1001\}$ is discriminative for Z_1 , then both $\{1000\}$ and $\{1001\}$ are also discriminative. Likewise, a planted four-item itemset contributes all of its discriminative subsets. Consequently, the ground-truth discriminative set is not limited to the ten planted families but consists of the complete downward closure of these families, yielding a total of 54 discriminative itemsets. This explains the value of 54 reported throughout the experimental evaluation.

A discovered itemset is counted as a *true positive* (TP) if it corresponds to a planted discriminative family, either directly or through one of its subsets or supersets, and satisfies the discrimination criterion based on per-class support counts. A discovered itemset that is unrelated to any planted family is counted as a *false positive* (FP), while a planted family is counted as a *false negative* (FN) if none of the discovered itemsets represents it. Consequently, precision is evaluated at the itemset level, whereas recall is evaluated at the planted-family level, and the reported F1-score combines these two measures.

Table 3. Synthetic validation. All 10 planted families are recovered (zero FN) at every threshold, and F1 reaches 1.000 for every support at or above 15%.

Support	T10 (10K/Class)				T25 (25K/Class)			
	TP	FN	FP	F1	TP	FN	FP	F1
5%	329	0	862	0.433	329	0	862	0.433
8%	272	0	552	0.496	272	0	608	0.472
10%	156	0	162	0.658	166	0	194	0.631
15%	54	0	0	1.000	54	0	0	1.000
20%	54	0	0	1.000	54	0	0	1.000
25%	54	0	0	1.000	54	0	0	1.000

The results show that recall remained 100% across all evaluated support thresholds, with all 10 planted discriminative patterns detected in both T10 and T25 datasets. The algorithm therefore recovered the complete set of known discriminative patterns in the synthetic data.

False positives occur primarily at lower support thresholds and gradually disappear as the minimum support increases. A detailed inspection revealed that these patterns are not incorrect discoveries but valid discriminative itemsets according to the proposed criterion. Most correspond to supersets or combinations of the planted discriminative itemsets that independently satisfy both the minimum support and the discrimination ratio requirements. These patterns do not arise from inaccurate support estimation or from pruning effects during the mining phase. Consequently, the F1-score reaches 1.000 for

all support thresholds of 15% and above. The remaining false positives observed at the 5% and 10% support thresholds are mainly redundant supersets of the planted patterns. Applying a minimal-generator filter would reduce the discovered set from 1191 itemsets at 5% support (or 54 itemsets at 15%) to the same 25 minimal patterns corresponding exactly to the planted discriminative itemsets.

The three negative controls and the borderline pattern ($1.67\times$) were flagged at no threshold, in either configuration. The $2\times$ criterion, applied to per-class frequencies, cleanly separates the planted signal from the near-equal and just-below-threshold decoys.

Execution times for the two dataset scales are reported below in Table 4.

Table 4. Execution time scaling from T10 (30K rows) to T25 (75K rows). Data volume increases $2.5\times$; time scales $1.9\text{--}2.8\times$, showing near-linear behavior at practical thresholds.

Support	T10 Time (s)	T25 Time (s)	Ratio
5%	154.9	293.6	$1.9\times$
8%	60.8	169.4	$2.8\times$
10%	32.5	84.4	$2.6\times$
15%	13.5	31.9	$2.4\times$
20%	10.8	25.5	$2.4\times$
25%	10.2	24.7	$2.4\times$

For support thresholds of 15% and higher, both T10 and T25 produced the same number of discriminative itemsets (54), indicating consistent results across datasets of different sizes.

Comparison with Pysubgroup on the Synthetic Data

For a same-hardware reference point we ran a synthetic configuration we refer to as T5 (3 classes $\times$ 5000 transactions = 15,000 total, same item vocabulary as the larger synthetic experiments) through both ARIS-Apriori and pysubgroup [33]. Each support level requires three Apriori passes from pysubgroup (one per class for `class=Z1, Z2, Z3`), and the same $2\times$ frequency-ratio filter is applied after mining. Counts of discriminative itemsets match ARIS-Apriori exactly at every support tested. Time comparison is shown in Table 5.

Table 5. ARIS-Apriori vs. pysubgroup [33] on the T5 synthetic dataset (3 classes $\times$ 5K transactions). Both algorithms produce identical counts of discriminative itemsets. Times in seconds on the same hardware.

Support	ARIS (s)	Pysubgroup (s)	Ratio	Discriminants
14%	12.9	32.4	$2.51\times$ slower	372
12%	14.8	33.4	$2.26\times$ slower	462
10%	30.4	34.2	$1.12\times$ slower	904
9%	34.5	39.2	$1.14\times$ slower	1088
8%	38.9	44.3	$1.14\times$ slower	1401

ARIS-Apriori was consistently faster than pysubgroup on the evaluated datasets. At moderate support (14%, 12%) the gap is about $2.2\text{--}2.5\times$; from 10% downward the two converge to a roughly $1.1\times$ ratio as both algorithms start to do significant candidate enumeration. The shape of the two curves reflects a structural difference. pysubgroup's runtime is dominated by fixed per-class overheads that do not depend on the number of patterns actually found: building 1034 Selector wrapper objects, evaluating each Selector as a boolean mask over the pandas DataFrame, and running the quality-function machinery for every candidate, repeated three times (once per class target). On T5 this fixed cost is about 32–34 s and stays essentially flat between 14% and 10% support. ARIS-Apriori

has a much lower fixed cost but its runtime is proportional to the candidate space it must enumerate, which grows sharply as support drops. The two curves in Figure 1 approach each other at the lower end of the range.

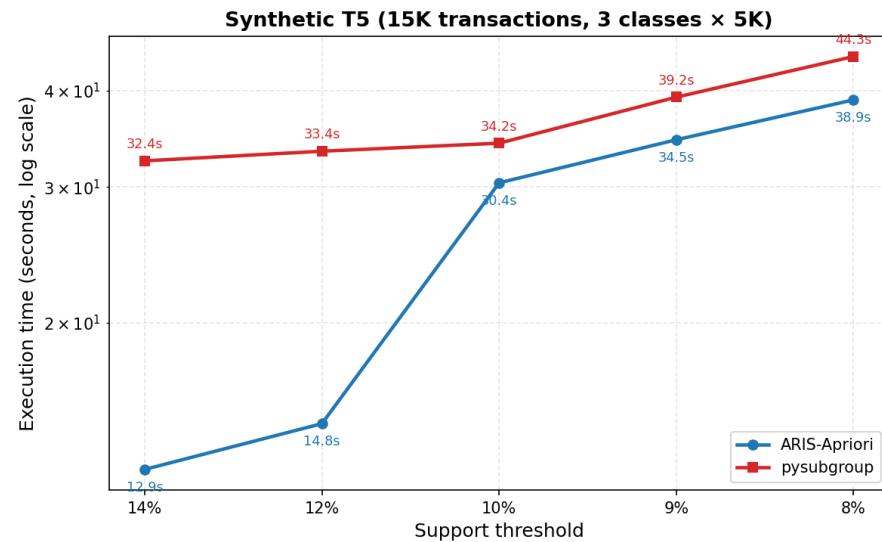


Figure 1. Execution time of ARIS-Apriori vs. pysubgroup on the synthetic T5 dataset (15K transactions, 3 classes $\times$ 5K) across the 14–8% support range (log scale). Both methods produce identical sets of discriminative patterns at every support; ARIS-Apriori is consistently faster, by a factor of 2.2–2.5 $\times$ at moderate support and $\sim$ 1.1 $\times$ at the lower end.

This comparison evaluates whether an existing peer-reviewed implementation and ARIS-Apriori produce the same discriminative patterns under identical conditions and whether their runtimes remain within the same order of magnitude. Across the evaluated support thresholds, ARIS-Apriori maintained a consistent runtime advantage. The comparison is revisited on the Mushroom dataset in the following subsection.

5.3. UCI Mushroom Dataset

The UCI Mushroom dataset [34] is a classical benchmark with 8124 specimens, each described by 22 categorical attributes (cap shape, odor, gill color, stalk surface, and so on) and labeled as edible or poisonous. We treat edibility as the class variable, giving two nearly balanced classes: Edible (Z_1 , 4208 transactions) and Poisonous (Z_2 , 3916). Every transaction has exactly 22 items, one value per attribute, so density is uniform and poses no challenge.

The support threshold is applied per class, following Definition 1: an itemset must reach $s_i = \lfloor \sigma N_i \rfloor$ occurrences in its own class. We sweep this threshold from 58% to 78% of each class's own size, at four-point intervals. At 58% support 706 discriminative itemsets survive; at 78% only 54 remain; below 58% the count climbs into the thousands. The absolute thresholds for each class and the ARIS mining times are listed in Table 6.

Three trends stand out. First, the number of discriminative itemsets falls quickly as the threshold rises, from 706 at 58% to 54 at 78%, while execution time drops by a smaller factor, so much of the work goes into evaluating candidates that never clear the discriminative criterion.

Second, the split between classes moves with the threshold. As shown in Figure 2 at 58% the edible class dominates (610 vs. 96); by 74% the two are close (44 vs. 32); and at 78% the poisonous class edges ahead (32 vs. 22). Many edible-associated patterns carry moderate support, while a few poisonous-associated features, bruises = no above all, stay frequent across the whole band.

Table 6. Counts of discriminative itemsets on the UCI Mushroom dataset. Absolute thresholds are given for the edible and poisonous classes.

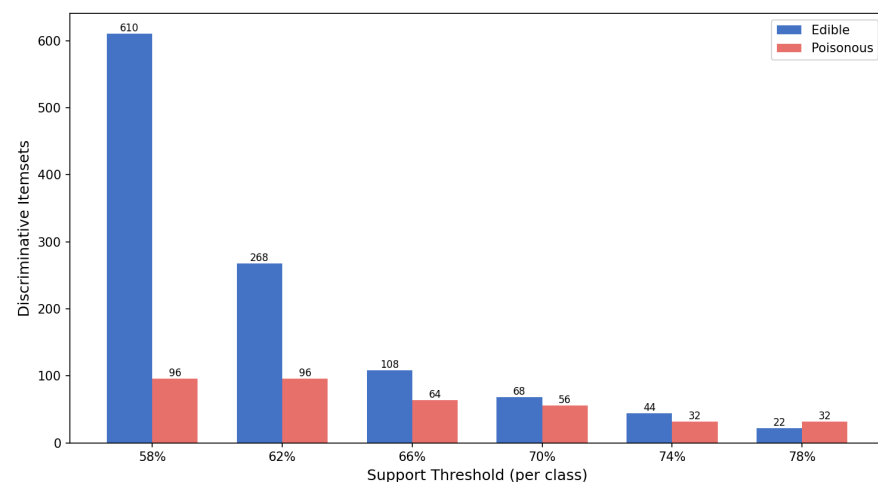
Support (per Class)	Abs (E/P)	Time (s)	Total	Edible	Poisonous
58%	2441/2271	16.8	706	610	96
62%	2609/2428	15.9	364	268	96
66%	2777/2585	10.9	172	108	64
70%	2946/2741	9.3	124	68	56
74%	3114/2898	7.5	76	44	32
78%	3282/3054	6.0	54	22	32

Third, the region around 70% per-class support, as shown in Table 7, gives a reasonably balanced set at moderate cost: 124 discriminative itemsets, comparable representation from both classes, and a mining time under ten seconds.

Table 7. Strongest single-attribute discriminative itemsets at 70% per-class support. Coverage is the own-class frequency.

Class	Pattern	Support	Coverage
Edible	gill-size = broad	3920	93.2%
Edible	stalk-surface-above-ring = smooth	3640	86.5%
Edible	odor = none	3408	81.0%
Edible	stalk-surface-below-ring = smooth	3400	80.8%
Edible	ring-type = pendant	3152	74.9%
Poisonous	bruises = no	3292	84.1%
Poisonous	population = several	2848	72.7%

These line up with the mycological literature. Absence of odor, broad gills, smooth stalk surfaces, and pendant rings are textbook markers for edible specimens; lack of bruising and populations recorded as “several” lean poisonous. odor = none is widely regarded as the single strongest predictor for this dataset, and the algorithm recovers it covering 81% of the edible class at a 26× ratio. gill-spacing = close, despite its high coverage among poisonous specimens (97.1%), is not discriminative: it also appears in 71.5% of edible specimens, a ratio of only 1.36×, well short of the threshold, so it does not appear among the reported markers.

**Figure 2.** Discriminative itemsets by class at each support threshold on the UCI Mushroom dataset.

A practitioner caveat worth flagging: two attributes are nearly constant in this dataset (veil-type = partial appears in 100% of all 8124 transactions, veil-color

= white in nearly all). They show up as “padding” inside compound discriminative itemsets, for example, the four patterns {gill-size = broad}, {veil-type = partial, gill-size = broad}, {veil-color = white, gill-size = broad}, and {veil-type = partial, veil-color = white, gill-size = broad} have nearly identical supports and encode the same single underlying signal. A simple post-processor that collapses patterns sharing both their support and a near-constant item resolves this if compactness matters.

Comparison with Pysubgroup on the Mushroom Data

We now repeat the pysubgroup comparison on Mushroom, following the same protocol used for the synthetic T5 dataset. Because Mushroom is a two-class problem (Edible vs. Poisonous), pysubgroup is executed twice, once with each class serving as the binary target, using MinSupportConstraint set to the corresponding class-specific minimum support threshold. Its built-in support constraint limits the total subgroup size across both classes rather than the support within the target class. Therefore, we apply the target-class support threshold of Definition 1 as an explicit post-filter. The discriminative criterion is then evaluated using support values recomputed directly from the original transactions of each class, rather than those returned by pysubgroup. This reproduces exactly the discriminative evaluation performed by ARIS-Apriori. Both pipelines are executed on the same hardware and the same merged transactions; therefore, the reported execution times in Table 8 reflect differences in the algorithms and their implementations rather than language-level overhead.

Table 8. Runtime comparison on UCI Mushroom against pysubgroup [33], called once per class direction at that class’s own support threshold. “Frequent” is the underlying number of frequent itemsets at each support level.

Support (per Class)	Time (s)		Discriminants	Frequent
	Pysubgroup	ARIS-Apriori		
58%	72.16	16.80	706	864
62%	26.31	15.90	364	500
66%	17.48	10.90	172	264
70%	17.38	9.30	124	196
74%	16.62	7.50	76	136
78%	7.88	6.00	54	102

The discriminative column reports a single value per row because both algorithms recover essentially the same set of patterns on Mushroom, with only a small boundary-case difference at 62% support, the same effect noted for the synthetic comparison. As a result, the more interesting comparison remains execution time rather than pattern recall.

As shown in Figure 3, ARIS-Apriori is faster than pysubgroup at every support level we tested, with the gap ranging from $1.3\times$ at 78% to $4.3\times$ at 58%. The same structural and implementation factors discussed for T5 apply here: the per-class Apriori multiplier (now $2\times$ instead of $3\times$ because Mushroom has only two classes), the pandas DataFrame indexing overhead, the quality-function machinery, and the Selector wrapper costs. With a smaller item vocabulary (117 unique items vs. 1034 on T5) the fixed-overhead component is also smaller, so the flat-runtime regime that produced a crossover on T5 does not appear within the Mushroom range tested. At lower support, pysubgroup hits a memory wall well before ARIS-Apriori does, consistent with its heavier per-candidate state (boolean masks, quality scores, Selector objects); we leave a full characterization of that regime to future work.

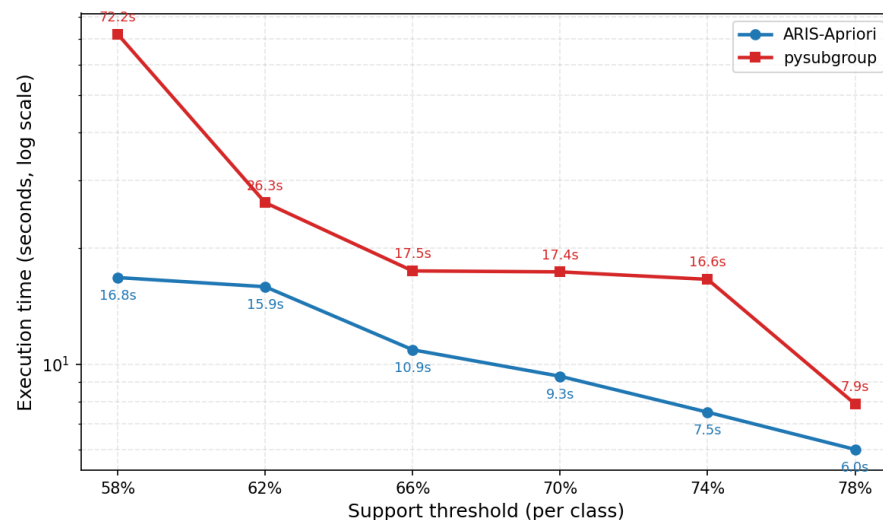


Figure 3. Execution time of ARIS-Apriori vs. pysubgroup on UCI Mushroom across the support range (log scale). Both methods produce essentially the same discriminative patterns across the sweep; ARIS-Apriori is consistently faster.

5.4. Online Retail Dataset

The UCI Online Retail dataset records customer invoices from a UK-based retailer between December 2010 and December 2011. We retained the 20,728 invoices dated in 2011 and grouped them by calendar quarter, yielding four classes: Q1 (3777 transactions), Q2 (4625), Q3 (4845), and Q4 (7481). The objective is to identify discriminative itemsets whose occurrence is associated with a specific quarter.

Compared with the previous datasets, the Online Retail dataset contains a substantially larger item vocabulary (~3900 distinct StockCodes) and exhibits high sparsity, with most products appearing in fewer than 1% of transactions. We apply the per-class support threshold of Definition 1. Mining below 2% per class combinatorially explodes the candidate space on this sparse, high-vocabulary catalog, so we vary the threshold from 2% to 5% of each quarter's own size.

In Table 9 Q4 dominates at every threshold, from 55% of patterns at 2% support to 74% at 5%. This is a real seasonal effect: the holiday quarter introduces a wide catalog of items (Christmas decorations, hot-water bottles, hand warmers) whose annual sales are concentrated in October through December. The algorithm keeps finding patterns well past 4% support: genuinely popular, quarter-specific products keep clearing the bar even as the threshold rises. The practical sweet spot remains around 2–2.5% support, where every quarter is represented and mining completes in about a minute or less. A graphical comparison of discriminative itemsets by quarter at each support threshold is displayed in Figure 4.

Table 9. Counts of discriminative itemsets on the Online Retail dataset by quarter.

Support (per Class)	Time (s)	Total	Q1	Q2	Q3	Q4
2.0%	67.7	283	48	10	68	157
2.5%	33.5	153	23	4	31	95
3.0%	21.5	95	11	3	22	59
3.5%	14.0	63	9	2	9	43
4.0%	10.1	41	6	1	5	29
5.0%	8.0	23	3	1	2	17

The decoded product names exhibit distinct patterns across quarters. Many of the strongest Q4 itemsets are associated with holiday-themed and gift-related products,

whereas Q3 contains several shopping-bag-related itemsets. Q2 is characterized by itemsets containing outdoor and household products, while Q1 groups around Easter and kitchenware. These observations are specific to the Online Retail dataset and are intended as descriptive interpretations of the discovered itemsets. Table 10 lists the five strongest discriminative itemsets per quarter at 2.5% support, together with their exact cross-class frequency and ratio.

Table 10. Strongest discriminative itemsets per quarter on the Online Retail dataset at 2.5% support. “Max other” is the highest frequency the pattern reaches in any other quarter.

Quarter	Pattern (Decoded)	Own Freq	Max Other	Ratio
Q1	SET OF 6 SPICE TINS PANTRY DESIGN	9.3%	3.8%	2.4×
Q1	SET OF 3 HEART COOKIE CUTTERS	6.7%	3.0%	2.3×
Q1	{SET OF 3 CAKE TINS PANTRY DESIGN, SET OF 6 SPICE TINS PANTRY DESIGN}	5.8%	1.9%	3.1×
Q1	GARLAND WOODEN HAPPY EASTER	4.7%	1.1%	4.4×
Q1	SET 3 PAPER VINTAGE CHICK PAPER EGG	4.4%	1.0%	4.5×
Q2	RED RETROSPOT PICNIC BAG	5.5%	2.6%	2.1×
Q2	STRAWBERRY PICNIC BAG	3.7%	1.8%	2.1×
Q2	WATERING CAN BLUE ELEPHANT	3.4%	1.6%	2.1×
Q2	WATERING CAN GREEN DINOSAUR	2.5%	1.1%	2.2×
Q3	SET OF 3 REGENCY CAKE TINS	8.8%	4.2%	2.1×
Q3	{JUMBO BAG VINTAGE DOILY, JUMBO BAG RED RETROSPOT}	5.4%	2.6%	2.0×
Q3	SET OF TEA COFFEE SUGAR TINS PANTRY	4.9%	2.0%	2.4×
Q3	SET OF 6 TEA TIME BAKING CASES	4.5%	2.2%	2.1×
Q3	{LUNCH BAG RED RETROSPOT, LUNCH BAG VINTAGE DOILY}	4.4%	2.2%	2.0×
Q4	PAPER CHAIN KIT 50’S CHRISTMAS	12.0%	4.7%	2.6×
Q4	RABBIT NIGHT LIGHT	9.2%	4.1%	2.2×
Q4	PAPER CHAIN KIT VINTAGE CHRISTMAS	8.7%	3.1%	2.8×
Q4	CHOCOLATE HOT WATER BOTTLE	7.4%	3.6%	2.0×
Q4	HAND WARMER OWL DESIGN	6.9%	2.3%	3.0×

Several of the size-2 itemsets pair products from the same line: {SET OF 3 CAKE TINS, SET OF 6 SPICE TINS} in Q1, {JUMBO BAG VINTAGE DOILY, JUMBO BAG RED RETROSPOT} and {LUNCH BAG RED RETROSPOT, LUNCH BAG VINTAGE DOILY} in Q3. These indicate recurring co-purchase behavior within specific quarters and may be useful for downstream retail analyses, including product grouping, assortment planning, and promotion design.

Every pattern in Table 10 is ratio-based rather than exclusive, and this is the norm on this dataset. Of the 153 discriminative itemsets at 2.5% support, only 11 are truly exclusive (zero occurrences in every other quarter), all of them in Q4: niche items such as TRADITIONAL PICK UP STICKS GAME and WALL ART STOP FOR TEA that happen not to appear at all outside the holiday quarter. True exclusivity becomes rarer as the threshold rises, from 15 of 283 patterns at 2% down to zero of 23 at 5%: the itemsets that survive a higher threshold are, almost by construction, popular enough to have some presence everywhere. The retail dataset’s discriminative signal is overwhelmingly about seasonal concentration, not hard exclusivity: the Christmas paper chain kit sells in every quarter but

still moves at $2.6\times$ the rate of its next-best quarter in Q4, which is exactly the pattern the $2\times$ ratio criterion was designed to catch.

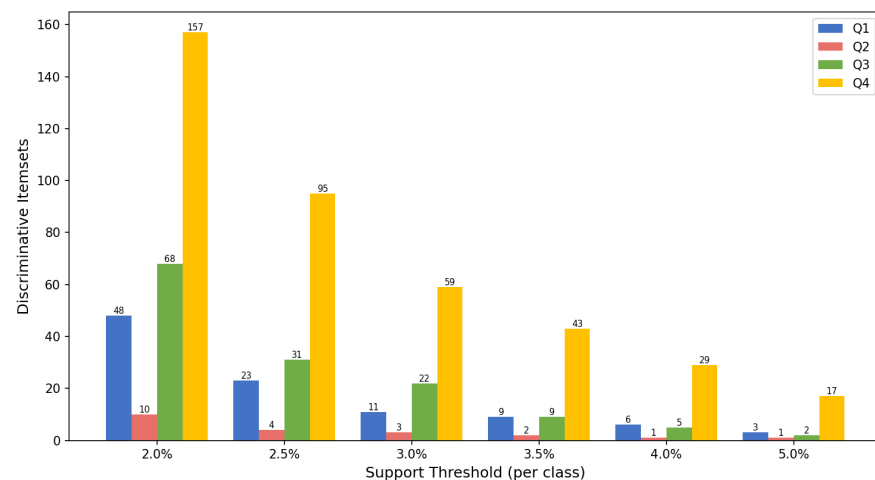


Figure 4. Discriminative itemsets by quarter at each support threshold on the Online Retail dataset.

5.5. Greek SME Innovation Survey

We applied ARIS-Apriori to a survey of 818 Greek SMEs covering roughly 85 questions on company profile, partnerships, innovation activities, certifications, promotion strategies, and perceived barriers. The class variable is the number of employees, grouped into three categories: Z_1 (solo owner, 287 responses), Z_2 (1–9 employees, 426), and Z_3 (10 or more, 105). The main methodological issue for this dataset is class imbalance: Z_3 is four times smaller than Z_2 .

Each response becomes a transaction of $\text{ColumnName} = \text{Value}$ items, such as $\text{Certificates} = \text{Yes}$ or $\text{EstYear} = \text{Pre2010}$. We adopted a *positive-only* encoding, dropping all = No items. Including them would double the vocabulary and push the average transaction length from ~ 23 to ~ 57 items; we tried this, and Apriori candidate generation became intractable below 50% support without producing additional interpretable discriminative patterns.

This dataset is where the class-size imbalance is most severe in our experiments: Z_3 has only 105 responses, a quarter of Z_2 's 426. We apply the support threshold of Definition 1 and compare per-class frequencies $f_z = \sup_z / |Z|$; We vary the threshold from 20% to 40% of each class's own size: high enough to keep mining fast and every reported itemset well supported. The counts of discriminative itemsets under this methodology are shown in Table 11.

Table 11. Counts of discriminative itemsets on the Greek SME Survey. Every discriminative itemset in this band belongs to Z_3 ; Z_1 and Z_2 contribute none.

Support (per Class)	Abs ($Z_1/Z_2/Z_3$)	Time (s)	Total
20%	57/85/21	0.78	208
25%	72/106/26	0.20	79
30%	86/128/32	0.08	33
35%	100/149/37	0.04	12
40%	115/170/42	0.03	8

As it is displayed in Figure 5, across the entire support range, no discriminative itemsets are identified for either Z_1 (solo owners) or Z_2 (micro enterprises). Although candidate itemsets satisfy the corresponding support thresholds, none achieves the required $2\times$ frequency ratio relative to the remaining classes. In contrast, class Z_3 (10 or more employees)

exhibits a substantial number of discriminative itemsets, with 33 discovered at 30% support. The characteristics of the discriminative itemsets for each class are discussed below.

5.5.1. Z_1 (Solo Owners, 287 Transactions)

Z_1 has no discriminative itemset at 20% support or above: no candidate recovers a $2\times$ margin against both other classes at this evidence level. The strongest candidate anywhere in the dataset is Sector = Energy, at a $2.7\times$ ratio on 22 of 287 solo-owner transactions (7.7% relative support), well below the 20% threshold used here. The pattern {EstYear = 2020s, Region = WestGreece}, occurring in 44.6% of Z_1 transactions, is not discriminative: its Z_2 frequency is 27.9%, only a $1.6\times$ ratio, short of the threshold. We find no robust discriminative signal for Z_1 at the support levels considered here.

5.5.2. Z_2 (Micro Enterprises, 426 Transactions)

No discriminative itemsets are identified for Z_2 at support thresholds of 20% or higher. Even its strongest single-attribute candidate, RevenueFromInnovation = Over60pct, reaches a discrimination ratio of only $1.9\times$, remaining below the required $2\times$ threshold. Similarly, although Region = WestGreece is frequent within Z_2 (69.5%), it is even more frequent in both Z_1 (76.7%) and Z_3 (76.2%), and therefore provides no discriminative value. These results indicate that the response profile of Z_2 substantially overlaps with those of the other two classes. As the largest and most representative group in the survey, Z_2 exhibits no individual attribute or attribute combination that satisfies the proposed discrimination criterion.

5.5.3. Z_3 (10 or More Employees, 105 Transactions)

Class Z_3 is the only class for which discriminative itemsets are discovered within the examined support range. At the 30% support threshold, the proposed method identifies 33 discriminative itemsets. These patterns are organized around three dominant characteristics of larger enterprises: formal certification (Certificates = Yes, particularly ISO 9001 certification), long-term operation (EstYear = Pre2010), and active business development through participation in trade exhibitions and investment in R&D. In addition to identifying individually discriminative attributes, the proposed method also reveals interactions between attributes that are not discriminative in isolation. For example, Region = WestGreece does not satisfy the discrimination criterion on its own, yet its combination with EstYear = Pre2010 achieves a discrimination ratio of $2.4\times$, demonstrating that the discriminative power arises from the joint occurrence of the two attributes rather than from either attribute individually. The strongest discriminative itemsets are presented in Table 12.

Table 12. Selected Z_3 discriminative itemsets at 30% support on the Greek SME Survey.

Pattern	Z_3 Freq	Ratio
Certificates = Yes	61.0%	$2.6\times$
CertificatesDetail_ISO9001 = Yes	36.2%	$2.9\times$
PromotionMethods_Exhibits = Yes	56.2%	$2.1\times$
EstYear = Pre2010, PromotionMethods_Digital = Yes	52.4%	$2.4\times$
EstYear = Pre2010, Partnerships_CHAMB = Yes	31.4%	$4.8\times$
EstYear = Pre2010, InnovationGoals_QUAL = Yes	32.4%	$4.2\times$
Investments_RD = Yes	40.0%	$2.0\times$

The discriminative patterns identified for Z_3 consistently characterize larger and more established enterprises. They combine attributes related to formal certification (particularly ISO 9001), longer operational history (EstYear = Pre2010), and stronger innovation and

market-development activities, including investment in R&D and participation in trade exhibitions. Together, these patterns suggest that firms with 10 or more employees are distinguished by a combination of organizational maturity and innovation-oriented practices rather than by any single characteristic. All reported discrimination ratios are computed relative to non-zero frequencies in the comparison classes, indicating that these patterns are significantly more frequent in Z_3 rather than being exclusive to it.

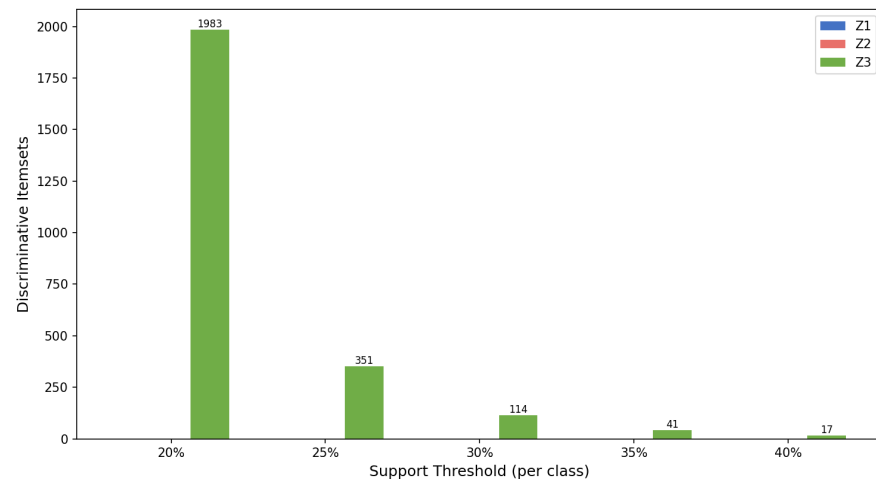


Figure 5. Discriminative itemsets by class at each support threshold on the Greek SME Survey.

5.6. Criteo Display Advertising Dataset

As a final stress test we applied ARIS-Apriori to a sample from the Criteo Display Advertising Challenge dataset, a large-scale click-through rate (CTR) prediction benchmark containing 45.8 million ad impressions. Each record has 13 integer count features (I1–I13) and 26 categorical features (C1–C26), all anonymized via 32-bit hashing. The class variable is whether the ad was clicked (Label = 1) or not (Label = 0).

We sampled 20,000 rows stratified by label from the first 100,000 records, yielding 4532 clicked and 15,468 not-clicked impressions, a 3.4:1 class imbalance typical of CTR data. To keep the item vocabulary manageable, we retained only the 10 categorical features with low cardinality (3–46 unique values each: C5, C6, C8, C9, C14, C17, C20, C22, C23, C25), capping C5 and C8 at their 20 most frequent values. This produced transactions averaging 8.2 items from a vocabulary of 157 unique items. Integer features were excluded to avoid arbitrary binning choices on anonymized counts. We apply the per-class support threshold of Definition 1. We vary the support from 1% to 10% of each class’s own size (Table 13).

Table 13. Counts of discriminative itemsets on the Criteo CTR dataset by class.

Support (per Class)	Time (s)	Total	Clicked	Not Clicked
1%	44.8	594	239	355
2%	17.2	156	47	109
3%	9.3	60	14	46
5%	4.9	18	4	14
8%	2.0	4	0	4
10%	1.4	1	0	1

Clicked-class discriminants are plentiful at low-to-moderate support: 239 at 1%, falling to 4 at 5%, and disappearing only once the threshold climbs past by 8%, where not-clicked signal thins out as well. The clicked class carries substantial discriminative structure once support is measured against its own size, despite having less than a third as many rows as not-clicked.

At 3% support, the two classes are comparably represented: 14 clicked discriminants and 46 not-clicked. Both sets are dominated by compound itemsets rather than single feature values. The 14 clicked patterns comprise one singleton, five pairs, five triples, two size-4 conjunctions, and one of size 5; the strongest is C14 = 64c94865 alone, at 5.7% own-class frequency against a 2.6% maximum elsewhere ($2.2\times$), and the same value anchors nine of the thirteen compounds. The 46 not-clicked patterns include five singletons, with the rest compounds up to size 4; C9 = 7cc72ec2 and C5 = 25c83c98 each anchor 16 of them. Since the Criteo features are anonymized, these hashed values cannot be interpreted directly, but their repeated appearance shows that both classes carry real, if hashed, discriminative structure.

Feature-column participation differs by class: clicked discriminants at 3% draw from C14, C9, C17, C23, C5, C8, and C20, while not-clicked discriminants draw from C17, C9, C5, C8, C6, C23, C14, and C20. Neither class produces a single discriminative pattern from C22 or C25 at any threshold we tested: these two features' value distributions are too even across classes for any single value to clear the $2\times$ ratio, itself a structurally informative result.

From a computational standpoint, mining time falls off sharply as the threshold rises, from 44.8 seconds at 1% to 1.4 seconds at 10%, reflecting the much larger frequent-itemset universe at the lower end of the sweep. Even the slowest level completes in under a minute on 20,000 transactions with 157 items.

5.7. Cross-Dataset Comparison

Table 14 pulls together the main findings from all six experimental configurations.

Table 14. Cross-dataset comparison of ARIS-Apriori discriminative mining.

Dataset	Classes	Trans.	Best Support	Disc.	Key Finding
Synthetic T10	3 (planted)	30,000	15%	54	F1 = 1.000; 100% recall at all thresholds
Synthetic T25	3 (planted)	75,000	15%	54	Same detections; near-linear time scaling
UCI Mushroom	2 (balanced)	8124	70%	124	Edible and poisonous counts converge at 74–78%; matches mycological markers
Online Retail	4 (quarterly)	20,728	2–2.5%	153–283	Q4 holiday dominance; ratio-based signal is the norm, not the exception
Greek SME Survey	3 (imbalanced)	818	30%	114/33 min	Only Z ₃ has signal; Z ₁ , Z ₂ have none
Criteo CTR	2 (imbalanced)	20,000	3%	60	Clicked class carries real discriminative signal despite 3.4:1 imbalance

Taken together, the experimental results lead to the following conclusions:

- **Correctness on ground-truth data.** In the synthetic experiments, the algorithm recovered all ten planted families, achieving 100% recall across every support threshold and a perfect F1 of 1.000 for all thresholds at or above 15% on both scales. Equally important, exact recounting left no censoring artifacts, so above that threshold the discovered set equals the planted closure exactly. The same accuracy held when the dataset grew by a factor of 2.5 (T25), while execution time scaled approximately linearly with the data.
- **Domain validity.** The discriminative itemsets discovered by the method align well with meaningful domain knowledge. In the Mushroom dataset, the identified patterns correspond to well-known mycological characteristics, including odor = none, gill-size = broad, and bruises = no. Similarly, in the Retail dataset, the decoded

product names reflect clear seasonal purchasing trends, with Christmas decorations and hot-water bottles dominating Q4, lunch and shopping bags appearing in Q3, picnic and garden items in Q2, and Easter and kitchenware products in Q1. These results suggest that the method is able to capture relevant and interpretable signals across different domains without requiring domain-specific tuning.

- Per-class support make minority-class detection possible. The frequency-normalized $2\times$ criterion identifies minority-class discriminative itemsets in the Survey dataset (Z_3 , a class four times smaller than Z_2) and in Criteo (the clicked class, outnumbered 3.4:1). In Criteo the clicked side yields 239 discriminants at 1% support and 14 at 3%, comparable in number to the not-clicked side despite having less than a third of its rows.
- Ratio-based discriminative itemsets are the norm, not the exception. The large majority of Retail's discriminative itemsets are ratio-based rather than exclusive: at 2.5% support, only 11 of 153 patterns are genuinely absent from every other quarter, and that count falls to zero by 5%. Products such as the Christmas paper chain kit sell in every quarter but spike sharply in their dominant one. These are precisely the patterns a pure exclusivity test would miss, and the finding validates retaining the $2\times$ ratio criterion alongside the unique-pattern check.
- Threshold sensitivity is real and dataset-specific. The best support level varies widely: 15% on synthetic, 70% per class on Mushroom, 2–2.5% per class on Retail, 30% per class on Survey, 3% per class on Criteo, and depends on transaction density, item count, vocabulary size, and how strongly the classes separate. There is no single universal value, and we recommend that practitioners sweep a band rather than fix a threshold up front.

All four real-world experiments (Sections 5.3–5.6) use the per-class threshold of Definition 1. In all experiments reported above, the succession distance parameter was set to $d = |M|$, causing the succession support to coincide with conventional support counting. This configuration was intentionally adopted to enable a direct comparison with existing discriminative itemset mining algorithms, which are based on conventional support, and to isolate the contribution of the proposed discriminative mining framework from the additional effect introduced by the succession parameter. While this setting enables a fair baseline comparison, it does not demonstrate the impact of succession-based support itself. Therefore, an additional set of experiments evaluating different values of d is presented in the following subsection.

5.8. Parameter Sensitivity Analysis

Beyond the support threshold, the method has two parameters: the distance d , which decides which occurrences count as support, and the ratio threshold θ , which decides how large a frequency differential must be before a pattern is called discriminative. We vary both parameters jointly on two datasets. The synthetic T10 configuration, at its 15% operating point, gives every cell a ground-truth interpretation. Online Retail, at its 2.5% operating point, is the dataset whose transaction order carries real meaning: invoices are stored in date order at roughly 55 invoices per trading day, so $d = 50$ spans about one day. Figures 6 and 7 report the discriminative-itemset count at every combination of $d \in \{10, 25, 50, 100, 250, |M|\}$ and $\theta \in \{1.5, 1.75, 2, 2.25, 2.5, 3, 3.5, 4\}$.

Figure 6 illustrates the sensitivity of the proposed method to the discrimination ratio threshold θ and the succession distance d , using the synthetic dataset for which the ground truth is known. For all values of d , every threshold between $\theta = 1.75$ and $\theta = 2.25$ recovers exactly the 54 itemsets belonging to the planted downward closure. This interval lies between the strongest non-planted pattern ($1.67\times$) and the weakest planted pattern ($2.3\times$),

demonstrating that the commonly adopted value $\theta = 2$ is robust rather than the result of parameter fine-tuning.

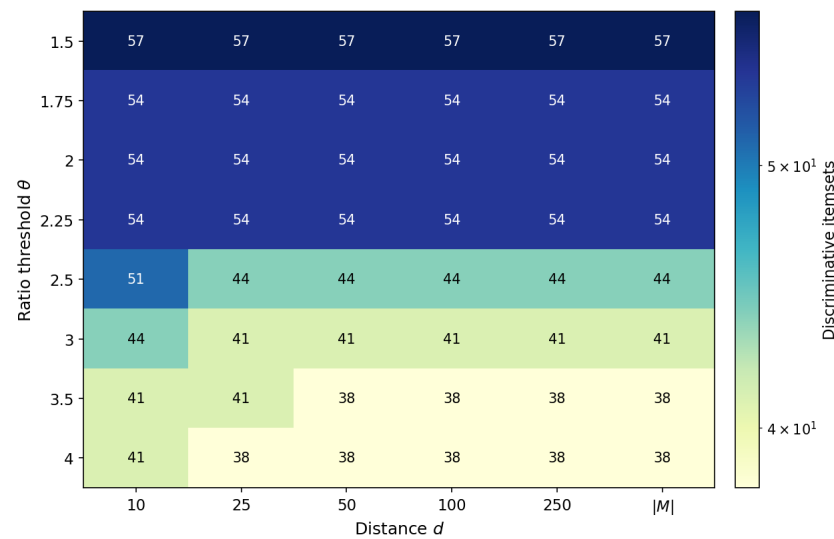


Figure 6. Discriminative itemsets on the synthetic T10 dataset at every combination of distance d and ratio threshold θ , at 15% support. Every cell with θ between 1.75 and 2.25 contains exactly the 54 itemsets of the planted closure.

Reducing the threshold to $\theta = 1.5$ admits the strongest non-planted pattern together with its two discriminative subsets, increasing the total number of discovered itemsets from 54 to 57. Conversely, increasing θ beyond 2.25 progressively removes the weaker planted families, reducing the number of discovered itemsets from 54 to 44, then to 41, and finally to 38. The remaining 38 itemsets correspond to the downward closure of the six uniquely discriminative planted families, which remain discriminative for any higher threshold.

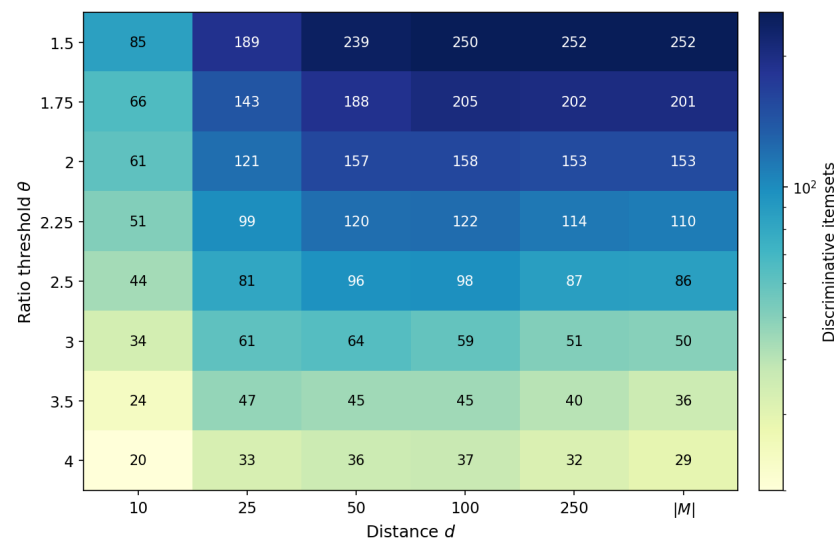


Figure 7. Discriminative itemsets on the Online Retail dataset at every combination of distance d and ratio threshold θ , at 2.5% support. The moderate-distance band ($d = 25$ – 100) yields more discriminative itemsets than frequency counting ($d = |M|$) at every θ .

The influence of the succession distance is considerably smaller. For $d \geq 25$, the number of discovered itemsets remains unchanged for every value of θ , as expected, since the synthetic transactions have no meaningful ordering and the planted itemsets occur densely within their target classes (30–40%), making succession breaks unlikely. The only

noticeable deviation occurs at $d = 10$, where several itemsets reappear (51 instead of 44 at $\theta = 2.5$). This behavior is caused by the succession constraint reducing the measured support of scattered occurrences in the comparison classes, thereby increasing the observed discrimination ratios. This result confirms that succession-based support is beneficial only when the transaction order carries meaningful information, whereas applying a small succession distance to unordered data may artificially inflate discrimination.

Figure 7 demonstrates that both the discrimination ratio threshold θ and the succession distance d influence the discovered patterns. As θ increases, the number of discriminative itemsets decreases steadily for every value of d , from 252 itemsets at $\theta = 1.5$ to 29 at $\theta = 4$ under conventional frequency counting ($d = |M|$). This gradual decline indicates that larger values of θ simply retain the most strongly quarter-specific products, without exhibiting any critical threshold around the default value $\theta = 2$.

The succession distance has a different effect. For moderate values ($d = 25$ – 100), the proposed succession-based support consistently discovers more discriminative itemsets than conventional frequency counting, and the advantage becomes more pronounced as θ increases. For example, at $\theta = 3$, using $d = 50$ yields 64 discriminative itemsets compared with only 50 when $d = |M|$.

The composition of the discovered patterns also changes. At $\theta = 2$ and $d = 50$, 24 of the 153 frequency-based discriminative itemsets disappear because their occurrences are too sparsely distributed to satisfy the succession criterion. At the same time, 28 new itemsets emerge. These patterns are not sufficiently discriminative under conventional support but become discriminative when succession-based support is used because their occurrences are concentrated into temporal bursts within the target quarter while remaining scattered across the comparison quarters. A representative example is `WOODEN ROUNDERS GARDEN SET`. Under conventional support it reaches a frequency of 2.98% in Q2 compared with a maximum of 1.72% in the remaining quarters, corresponding to a discrimination ratio of only $1.73\times$. Under succession-based support with $d = 50$, however, the ratio increases to $2.2\times$, allowing the itemset to satisfy the discrimination criterion. Since the discrimination ratio itself is independent of the support threshold, such patterns cannot be recovered by adjusting the minimum support alone; they become visible only when the temporal locality of transactions is taken into account.

Overall, the two parameters play complementary roles. Increasing θ increases the required discriminative strength, whereas decreasing d increases the required temporal concentration of itemset occurrences. The strictest configuration ($d = 10$, $\theta = 4$) therefore identifies only 20 itemsets that are both highly concentrated in time and strongly associated with a particular quarter. These results demonstrate that the succession distance is meaningful only for datasets whose transaction order carries information. For datasets such as Mushroom and the Greek SME Survey, where transaction order is arbitrary, setting $d = |M|$ remains the appropriate choice, as also confirmed by the synthetic experiments.

6. Conclusions

We have presented a procedure for discriminative pattern mining that extends the Association Rules in Succession (ARIS) framework [26] to multi-dataset cases. The proposed approach consists of three steps. First, dataset membership is encoded through class markers within a single merged transactional database. Next, ARIS-Apriori is executed with a support threshold calculated with respect to each class's own size. Finally, a frequency-normalized $2\times$ ratio criterion, evaluated by exact cross-class counting, is applied to identify itemsets whose frequency dominates the maximum frequency in any other class.

Across six experimental configurations (two synthetic scales with planted ground truth, the UCI Mushroom and Online Retail benchmarks, a Greek SME innovation survey,

and a Criteo CTR sample), the method recovers the intended structure. On the synthetic data it achieves 100% recall at every threshold and a perfect F1 of 1.000 for every support at or above 15%, confirming algorithmic correctness against ground truth. On Mushroom it surfaces well-known mycological markers (odor = none, gill-size = broad, bruises = no) at 70% per-class support. The results across the four real-world datasets highlight both the strengths and the reach of the approach. In Online Retail, the method captures clear seasonal trends across all four quarters; most of its discriminative itemsets are ratio-based rather than exclusive, demonstrating the value of the ratio criterion beyond pure exclusivity checks. Activating the succession constraint on this dataset, whose invoices follow time order, surfaces 28 further seasonal discriminants at $d = 50$ that frequency counting alone does not find. In the Greek SME survey, only the minority class (Z_3) carries a robust discriminative signal; the solo-owner and micro-enterprise classes have none. In Criteo, the clicked class carries substantial discriminative signal despite being outnumbered 3.4:1 by not-clicked impressions, once support is measured against its own size.

We benchmarked ARIS-Apriori against pysubgroup [33], a peer-reviewed Python library for subgroup discovery, on the same hardware and same merged datasets. On synthetic T5 (15K transactions, three classes) ARIS-Apriori is $2.2\text{--}2.5\times$ faster than pysubgroup at moderate support and the gap narrows to about $1.1\times$ as both algorithms enter the candidate-enumeration regime. On Mushroom ARIS-Apriori is faster at every support level tested, by up to $4.3\times$ at the low end of the band, and sustains lower support levels before memory becomes a constraint. The two libraries produce essentially identical sets of discriminative itemsets where both complete, so the runtime comparison reflects implementation and architecture choices rather than algorithmic differences. Overall, the results show that a specialized algorithm for discriminative itemset mining can produce the same answers as a general-purpose library while maintaining a consistent speed advantage at the support levels typically used by researchers.

The comparison with pysubgroup was performed under $d = |M|$, where succession-based support is equivalent to conventional support. This setting was intentionally adopted to validate the proposed discriminative itemset mining framework against an existing implementation under identical support semantics. Under these conditions, both methods discover the same class of patterns, namely conventional discriminative itemsets. When smaller values of d are employed, the proposed succession-based discriminative itemset mining framework incorporates the locality of itemset occurrences in the original transaction order and discovers succession-based discriminative itemsets. Since existing discriminative mining algorithms are based exclusively on conventional support and do not consider transaction proximity, the patterns discovered under $d < |M|$ belong to a different pattern class and therefore cannot be directly compared with their outputs. Consequently, the purpose of the additional sensitivity analysis is not to demonstrate equivalence with existing methods, but rather to illustrate the effect of the succession parameter and to highlight the additional information provided by succession-based discriminative itemset mining.

Several directions remain open. First, although the proposed framework now demonstrates the effect of succession-based support through a sensitivity analysis over different values of the succession distance parameter, a more extensive evaluation on large-scale temporal and sequential transactional datasets (e.g., event logs, clickstreams, server traces, or behavioral sessions) would provide further insight into the practical benefits of succession-based discriminative itemset mining. Second, the discriminative criterion is currently based on a user-defined ratio threshold. While the sensitivity analysis illustrates the influence of this parameter, adaptive or statistically calibrated discrimination thresholds could further improve robustness across datasets with different class distribu-

tions. Third, the current framework focuses on the complete discovery of succession-based discriminative itemsets and does not incorporate redundancy reduction or statistical significance testing. Integrating closed or maximal discriminative pattern mining together with multiple-testing correction techniques (e.g., FWER or FDR control) would constitute a valuable extension of the framework. Finally, scaling ARIS-Apriori to very large transactional datasets through distributed implementations based on Spark or MapReduce would broaden the applicability of the proposed method beyond the moderate-scale datasets considered in this work.

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

Funding: This research received no external funding.

Institutional Review Board Statement: The study involved an anonymous, voluntary survey and did not collect personally identifiable or sensitive personal data. No experimental intervention was performed.

Informed Consent Statement: All participants were informed about the purpose of the study, participated voluntarily, and provided informed consent before completing the anonymous survey.

Data Availability Statement: The UCI Mushroom, UCI Online Retail, and Criteo Display Advertising datasets are publicly available. The synthetic datasets can be regenerated with the seeds reported in Section 5.2. The Greek SME survey data are available from the authors on reasonable request.

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

References

1. Zhao, Q.; Bhowmick, S.S. *Association Rule Mining: A Survey*; Technical Report 2003116; CAIS, Nanyang Technological University: Singapore, 2003.
2. Kotsiantis, S.; Kanellopoulos, D. Association Rules Mining: A Recent Overview. *GESTS Int. Trans. Comput. Sci. Eng.* **2006**, *32*, 71–82.
3. Ghafari, S.M.; Tjortjis, C. A survey on association rules mining using heuristics. *WIREs Data Min. Knowl. Discov.* **2019**, *9*, e1307.
4. Agrawal, R.; Imielinski, T.; Swami, A. Mining Association Rules Between Sets of Items in Large Databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Washington, DC, USA, 26–28 May 1993; pp. 207–216.
5. Han, J. Mining knowledge at multiple concept levels. In *Proceedings of the Fourth International Conference on Information and Knowledge Management (CIKM)*, Baltimore, MD, USA, 28 November–2 December 1995; pp. 19–24.
6. Koperski, K.; Han, J. Discovery of spatial association rules in geographic information databases. In *Proceedings of the Advances in Spatial Databases (SSD 1995)*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1995; Volume 951, pp. 47–66.
7. Cheung, D.W.; Han, J.; Ng, V.T.; Wong, C.Y. Maintenance of discovered association rules in large databases: An incremental updating technique. In *Proceedings of the Twelfth International Conference on Data Engineering*, New Orleans, LA, USA, 26 February–1 March 1996; pp. 106–114.
8. Srikant, R.; Agrawal, R. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the International Conference on Extending Database Technology*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 1–17.
9. Mannila, H.; Toivonen, H.; Verkamo, A.I. Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.* **1997**, *1*, 259–289.
10. Liu, B.; Hsu, W.; Ma, Y. Mining association rules with multiple minimum supports. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; ACM: New York, NY, USA, 1999; pp. 337–341.
11. Boutsinas, B.; Meletiou, G.C.; Vrahatis, M.N. Mining Encrypted Data. In *Supply Chain and Finance*; Pardalos, P.M., Migdalas, A., Baourakis, G., Eds.; Series on Computers and Operations Research; World Scientific: Singapore, 2004; Volume 2, pp. 273–281.
12. Jiang, N.; Gruenwald, L. Research issues in data stream association rule mining. *ACM SIGMOD Rec.* **2006**, *35*, 14–19.
13. Agrawal, R.; Srikant, R. Fast Algorithms for Mining Association Rules. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, Santiago de Chile, Chile, 12–15 September 1994; pp. 478–499.

14. Park, J.S.; Chen, M.S.; Yu, P.S. An Effective Hash Based Algorithm for Mining Association Rules. In Proceedings of the ACM SIGMOD International Conference on Management of Data, San Jose, CA, USA, 22–25 May 1995; pp. 175–186.
15. Brin, S.; Motwani, R.; Ullman, J.; Tsur, S. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Tucson, AZ, USA, 13–15 May 1997; pp. 255–264.
16. Han, J.; Pei, J. Mining frequent patterns by pattern-growth: Methodology and implications. *ACM SIGKDD Explor. Newsl.* **2000**, *2*, 14–20.
17. Agrawal, R.; Srikant, R. Mining Sequential Patterns. In Proceedings of the Eleventh International Conference on Data Engineering (ICDE), Taipei, Taiwan, 6–10 March 1995; pp. 3–14.
18. Pei, J.; Han, J.; Mortazavi-Asl, B.; Pinto, H.; Chen, Q.; Dayal, U.; Hsu, M.C. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In Proceedings of the 17th International Conference on Data Engineering (ICDE), Heidelberg, Germany, 2–6 April 2001; pp. 215–224.
19. Ale, J.M.; Rossi, G.H. An approach to discovering temporal association rules. In *Proceedings of the 2000 ACM Symposium on Applied Computing—Volume 1 (SAC '00)*; ACM: New York, NY, USA, 2000; pp. 294–300.
20. Chen, C.H.; Lan, G.C.; Hong, T.P.; Lin, S.B. Mining fuzzy temporal association rules by item lifespans. *Appl. Soft Comput.* **2016**, *41*, 265–274.
21. Segura-Delgado, A.; Gacto, M.J.; Alcalá, R.; Alcalá-Fdez, J. Temporal association rule mining: An overview considering the time variable as an integral or implied component. *WIREs Data Min. Knowl. Discov.* **2020**, *10*, e1367.
22. Dong, G.; Bailey, J. *Contrast Data Mining: Concepts, Algorithms, and Applications*; CRC Press: Boca Raton, FL, USA, 2012.
23. Seyfi, M.; Nayak, R.; Xu, Y.; Geva, S. Efficient mining of discriminative itemsets. In Proceedings of the International Conference on Web Intelligence, Leipzig, Germany, 23–26 August 2017; pp. 451–459.
24. Lee, J.G.; Han, J.; Li, X.; Cheng, H. Mining Discriminative Patterns for Classifying Trajectories on Road Networks. *IEEE Trans. Knowl. Data Eng.* **2011**, *23*, 713–726. <https://doi.org/10.1109/TKDE.2010.198>.
25. Panteli, A.; Boutsinas, B. Addressing the Cold-Start Problem in Recommender Systems Based on Frequent Patterns. *Algorithms* **2023**, *16*, 182. <https://doi.org/10.3390/a16040182>.
26. Kompothrekas, A.; Boutsinas, B. On Defining Association Rule Mining in Succession. In Proceedings of the 9th International Conference on Mathematics and Statistics (ICOMS 2026), Paris, France, 11–13 September 2026.
27. Bay, S.D.; Pazzani, M.J. Detecting Group Differences: Mining Contrast Sets. In Proceedings of the Data Mining and Knowledge Discovery, San Francisco, CA, USA, 26–29 August 2001; Volume 5, pp. 213–246.
28. Dong, G.; Li, J. Efficient mining of emerging patterns: Discovering trends and differences. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99)*; ACM: New York, NY, USA, 1999; pp. 43–52. <https://doi.org/10.1145/312129.312191>.
29. Cheng, H.; Yan, X.; Han, J.; Yu, P.S. Direct discriminative pattern mining for effective classification. In Proceedings of the 24th International Conference on Data Engineering (ICDE), Cancun, Mexico, 7–12 April 2008; pp. 169–178. <https://doi.org/10.1109/ICDE.2008.4497425>.
30. Sheng, Y.; Li, J.; Zhang, K.; Yi, K.; Xiao, X.; Chang, W.C. Mining Minimal Distinguishing Subsequences with Gap Constraints. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '19)*; ACM: New York, NY, USA, 2019; pp. 1140–1148.
31. Meng, Z.; He, J.; Wong, R.K. Mining conditional discriminative sequential patterns. *Inf. Sci.* **2019**, *478*, 524–539. <https://doi.org/10.1016/j.ins.2018.11.033>.
32. Lu, Y.; Zhao, C.; Zheng, L.; Wang, H. Significance-based discriminative sequential pattern mining. *Expert Syst. Appl.* **2019**, *122*, 54–64. <https://doi.org/10.1016/j.eswa.2018.12.046>.
33. Lemmerich, F.; Becker, M. pysubgroup: Easy-to-Use Subgroup Discovery in Python. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2018), Part III*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 11053, pp. 658–662. https://doi.org/10.1007/978-3-030-10997-4_46.
34. Mushroom [Dataset]. UCI Machine Learning Repository; University of California, Irvine: Irvine, CA, USA, 1981. <https://doi.org/10.24432/C5959T>.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.