UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA


# TOWARDS ASSOCIATIVE CLASSIFICATION USING TRICLUSTERING

**Alexandre da Silva Monforte**

**Mestrado em Ciência de Dados**

Versão Provisória

Dissertação orientada por:
Profª. Doutora Helena Isabel Aidos Lopes Tomás
Profª. Doutora Sara Alexandra Cordeiro Madeira

2024

# Acknowledgments

I would like to thank Professor Helena Aidos and Professor Sara Madeira for their valuable guidance and support throughout this research. I also want to express my appreciation to Diogo Soares for his helpful contributions and assistance during the development of this work.

I would like to thank FCT for their financial support, which reflects their confidence in my capabilities and their investment in my potential.

A special thank you to my partner, Margarida for her invaluable insights and support during the writing of this thesis. Your thoughtful suggestions and assistance were truly invaluable throughout this process.

To my friends, Beatriz, Bruna, Camila, Joana, and Maria, I am deeply grateful for their constant support and presence throughout all phases of this study, each in their own unique way.

To my family—my mom Marlene, my father Ângelo, my brother Ademar, my cousin Marlene, my stepfather Carlos, and my stepmother Cláudia—I would like to express my heartfelt thanks for their unshakable confidence in my ability to achieve this work. I know they are proud of me, and that means the world to me.

To Gabriel who knew how to encourage me in the toughest moments, celebrate each achievement, and guide my emotions through the darkest times.

Thank You.

*Dedicatória*

*A minha tese é dedicada aos meus avós, Alice e Aristides. É um prazer enorme para mim realizar um dos maiores sonhos da minha Mami, concluindo o meu mestrado.*

# Resumo

Esta dissertação foca-se no desenvolvimento de um classificador associativo temporal, projetado para lidar com as complexidades inerentes à análise de dados biomédicos, através da utilização de triclusters discriminativos. Dados biomédicos, frequentemente caracterizados por elevada dimensionalidade, desalinhamento temporal e dados incompletos, apresentam desafios significativos para métodos de classificação tradicionais. Neste contexto, o presente estudo propõe um classificador que tira proveito das técnicas de triclustering para identificar e explorar padrões dentro de subestruturas tridimensionais, com um foco particular na dimensão temporal, permitindo uma análise mais profunda e precisa dos dados.

A motivação para este trabalho surge da necessidade crescente de métodos eficazes e precisos para a análise de dados biomédicos multidimensionais. A identificação de triclusters discriminativos, que são subestruturas tridimensionais dentro de um conjunto de dados, tem o potencial de transformar a compreensão e o tratamento de várias condições médicas. Nesses triclusters, os objetos exibem padrões específicos que estão associados a uma determinada classe. Este avanço pode proporcionar diagnósticos mais precisos e tratamentos personalizados, melhorando significativamente a qualidade dos cuidados de saúde e os resultados clínicos. O classificador desenvolvido visa não apenas melhorar a precisão das previsões, mas também garantir que os resultados sejam interpretáveis, um fator crítico em ambientes clínicos, onde as decisões precisam ser baseadas em informações claras e compreensíveis.

A metodologia adotada neste estudo envolveu uma análise rigorosa da eficácia das técnicas de triclustering dentro de um framework de classificação. Foram realizadas comparações entre o desempenho do classificador associativo proposto e abordagens já estabelecidas que são Triclustering-Based que usam classificadores como Random Forest e XGBoost, ambos adaptados para lidar com dados temporais tridimensionais. A avaliação dos resultados foi feita com base em métricas como precisão, especificidade e sensibilidade, escolhidas pela sua relevância no contexto de dados clínicos, onde os erros de classificação podem ter implicações graves, como diagnósticos incorretos que podem levar a tratamentos inadequados ou atrasos críticos em intervenções médicas.

Os resultados obtidos demonstraram que o classificador associativo exibe um desempenho competitivo, especialmente em cenários onde os padrões são predominantemente identificados na dimensão de contexto. No entanto, o estudo também revelou limitações importantes, como a dependência da qualidade e da quantidade de triclusters gerados, fatores que influenciam diretamente a eficácia geral do classificador.

Um dos principais desafios encontrados durante o desenvolvimento do classificador associativo foi a geração de dados para os testes. Para este propósito, foi utilizada a ferramenta G-HTric, reconhecida na simulação de dados temporais tridimensionais. Esta escolha permitiu a criação de cenários controlados, essenciais para uma avaliação rigorosa do classificador durante as fases iniciais do desenvolvimento. Contudo, surgiram desafios adicionais, especialmente no que diz respeito à capacidade do algoritmo TCTriCluster de lidar com dados categóricos. Este algoritmo, que foi usado para calcular os triclusters, mostrou-se limitado na interpretação desses valores, tratando-os exclusivamente como numéricos. Esta limitação teve um impacto direto na relevância dos resultados, limitando o potencial do classificador em cenários clínicos mais variados e complexos.

Apesar dessas dificuldades, o classificador associativo apresentou um desempenho superior nos testes que envolveram dados que apresentavam padrões exclusivamente na dimensão dos contextos. Esses resultados evidenciam o potencial do classificador associativo, especialmente em contextos onde os padrões temporais desempenham um papel crucial na previsão de resultados clínicos. Além disso, o estudo demonstrou que a capacidade do classificador de condensar triclusters complexos num conjunto mais gerenciável e interpretável de regras representa um avanço significativo, com importantes implicações para a aplicação prática em contextos clínicos.

Um ponto de análise fundamental no estudo foi o impacto do algoritmo FP-Growth no desempenho do classificador associativo. Este algoritmo foi incorporado com o objetivo de reduzir o número de regras, agrupando triclusters em regras derivadas pelo FP-Growth. Embora esta simplificação tenha facilitado o processo de classificação ao reduzir o número de regras, nos casos em que foram calculados menos do que 50 triclusters, resultou numa perda de informação que prejudicou o desempenho do classificador. Este fenómeno sugere que a aplicação do FP-Growth deve ser cuidadosamente avaliada, considerando os benefícios e as desvantagens, especialmente em termos do equilíbrio entre interpretabilidade e precisão. A aplicação inadequada pode comprometer a capacidade do classificador de fazer previsões precisas, especialmente em situações onde a perda de informação significativa ocorre.

Além disso, a expansão do âmbito do classificador para incluir padrões nas características (features) foi sugerida como uma necessidade para alcançar uma análise mais holística e precisa dos dados biomédicos multidimensionais. Atualmente, o foco na dimensão dos contextos limita a capacidade do classificador de capturar interações mais complexas que possam ocorrer entre diferentes dimensões dos dados. Ao expandir a abordagem para integrar padrões nas características, espera-se que o classificador se torne mais robusto, permitindo a descoberta de insights mais profundos e potencialmente transformadores no campo da análise de dados biomédicos. Esta expansão pode permitir que o classificador seja aplicado a uma gama mais ampla de cenários clínicos, aumentando assim a sua utilidade e aplicabilidade em diferentes áreas da medicina.

Outro aspeto crítico explorado na dissertação foi a validação do classificador com dados clínicos reais. Embora os conjuntos de dados sintéticos gerados pela ferramenta G-HTric tenham fornecido um ambiente controlado para os testes iniciais, eles não capturam completamente a variabilidade e o ruído típicos dos dados clínicos reais. A aplicação do classificador a dados clínicos

reais, como registos longitudinais de pacientes, não só validaria a utilidade do classificador em contextos práticos, mas também ofereceria insights valiosos sobre o seu desempenho em condições mais desafiadoras e realistas. Esta validação é crucial, pois a eficácia de um classificador em ambientes clínicos depende da sua capacidade de lidar com a complexidade e o ruído dos dados reais, garantindo que as previsões feitas sejam tanto precisas quanto confiáveis.

Outro ponto importante identificado foi a necessidade de investigar mais profundamente o impacto da quantidade e qualidade dos triclusters gerados no desempenho do classificador. A pesquisa revelou que a performance do classificador está altamente dependente desses fatores, o que levanta questões sobre como melhorar a geração de triclusters de forma a otimizar o desempenho do classificador em diferentes cenários. A realização de estudos adicionais que explorem a relação entre o número de triclusters e a precisão das previsões poderia fornecer insights cruciais para o aprimoramento do classificador, tornando-o mais adaptável a diferentes tipos de dados clínicos.

Finalmente, a dissertação conclui que, apesar das limitações identificadas, o classificador associativo temporal proposto oferece uma contribuição significativa para o campo da análise de dados biomédicos. A capacidade do classificador de condensar triclusters complexos num conjunto mais gerenciável e interpretável de regras representa um avanço importante, com aplicações práticas em contextos clínicos, onde tanto a precisão quanto a interpretabilidade são fundamentais. Este trabalho estabelece uma base sólida para futuras investigações, que poderão focar-se na melhoria dos algoritmos de triclustering, no desenvolvimento de técnicas mais avançadas de alinhamento temporal, e na ampliação do âmbito do classificador para integrar padrões nas características.

Além disso, o estudo sugere que a investigação futura deve explorar o desenvolvimento de algoritmos mais eficientes e robustos, capazes de lidar com a alta dimensionalidade e a natureza incompleta dos dados biomédicos, garantindo que os classificadores sejam capazes de fornecer previsões precisas e úteis em cenários clínicos reais. A combinação dessas melhorias pode contribuir significativamente para o avanço da medicina de precisão, permitindo uma análise mais detalhada e personalizada dos dados de saúde, e, em última análise, melhorando os resultados dos pacientes.

Para concluir, este trabalho contribui para o avanço de ciência de dados em áreas da saúde. A aplicação de técnicas de triclustering na classificação associativa pode transformar a forma como os dados clínicos são analisados, oferecendo novas oportunidades para diagnósticos mais precisos, tratamentos personalizados e uma melhoria na qualidade dos cuidados de saúde.

**Palavras-chave:** dados biomédicos, classificação associativa, triclustering, dados temporais tridimensionais

# Abstract

This thesis presents the development of a temporal associative classifier aimed at addressing the inherent complexities of biomedical data analysis through the use of discriminative triclusters. Biomedical datasets are often characterized by high dimensionality, temporal misalignment, and missing data, which pose significant challenges for traditional classification methods.

The developed classifier leverages triclustering techniques to uncover patterns within three-dimensional substructures, particularly focusing on the temporal dimension. These patterns, identified by triclustering algorithms, are utilized to enhance the accuracy and interpretability of predictions within clinical contexts.

The research involves a comprehensive evaluation of the classifier's performance against established Triclustering-Based approaches using Random Forest and XGBoost classifiers, which are also designed to handle three-way temporal data. The evaluation metrics include accuracy, specificity, and sensitivity, carefully chosen to reflect the clinical importance of minimizing both false positives and false negatives. The results indicate that the associative classifier exhibits competitive performance, particularly in scenarios where constant, additive, and multiplicative patterns are predominantly present in the context dimension. However, the study also reveals limitations, notably the classifier's dependency on the quality and quantity of triclusters generated, which significantly influences its overall effectiveness.

Despite these limitations, the classifier demonstrates potential in condensing complex triclusters into a more interpretable set of rules, thereby enhancing its applicability in clinical settings where both accuracy and interpretability are critical. The study contributes to the field of temporal data classification by providing valuable insights into the use of triclustering in biomedical data analysis. It also provide the basis for future research aimed at refining triclustering algorithms, and expanding the developed classifier's capability to capture patterns on the feature dimension.

Ultimately, this research aspires to advance precision medicine by offering a robust tool that can improve diagnostic accuracy and treatment personalization, thereby positively impacting healthcare outcomes.

**Keywords:** biomedical data, associative classification, triclustering, three-way temporal data

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This chapter offers an overview of the motivation for this research, outlining the specific goals to be achieved and the key challenges addressed. It begins by exploring the significance of analyzing multidimensional biomedical data and introduces the concept of using discriminative triclusters in temporal associative classifiers. The chapter also outlines the contributions this research makes to the field of biomedical data analysis, leading to a deeper understanding of the methods and approaches discussed in subsequent chapters.

## 1.1 Motivation

Biomedical research frequently encounters multidimensional data, which encompass dimensions such as objects, features, and temporal contexts, collectively referred to as three-way data [1]. This complexity is exemplified by the dynamics of gene expression over time, the progression of individual health profiles, and the intricate interactions within biological networks. The significance of analyzing these multidimensional data lies in their potential to uncover hidden patterns, identify key biological elements, understand disease progression, and classify entities based on similar behaviors [2].

In this context, the development of a temporal associative classifier using discriminative triclusters is crucial. A discriminative tricluster refers to a three-dimensional substructure within a dataset where the objects exhibit patterns specifically associated with a certain class [1]. Triclustering algorithms facilitate the detection of patterns across multiple dimensions by grouping observations that display similar patterns. This concept is particularly advantageous for uncovering complex patterns in high-dimensional data. By revealing inherent temporal patterns and associations within these data, the classifier offers invaluable tools for biomedical researchers, clinicians, and decision-makers, enabling them to make informed decisions, identify critical biological markers, and enhance the understanding and management of diseases [1].

Analyzing multidimensional data in biomedical research presents significant challenges due to the complexity involved and the necessity to extract meaningful information from large data volumes. Three-way data can reveal essential hidden patterns crucial for understanding biological processes and diseases [2]. However, traditional approaches commonly fail to effectively

addressing this complexity. The proposed development of a temporal associative classifier based on discriminative triclusters emerges as a promising solution to these challenges.

By employing advanced temporal classification techniques, this research aims to advance the field of data science while bringing benefits to medicine and public health. The identification of specific temporal patterns can lead to more accurate diagnoses and personalized treatments, significantly improving patient outcomes.

This research is situated at the intersection of data science and biomedical sciences. In the contemporary biomedical field, there is an increasing demand for sophisticated analytical tools capable of managing and interpreting large amounts of multidimensional data. The integration of data science methodologies into biomedical research is encouraged by the need to handle the complexity and volume of data generated by modern biomedical studies.

## 1.2 Goal

The primary objective of this research is to develop a temporal associative classifier leveraging discriminative triclusters. This goal is driven by the necessity for more accurate and interpretable models capable of handling the complex, multidimensional nature of biomedical and clinical data, particularly datasets characterized by temporal components.

To achieve this goal, a comprehensive analysis will be conducted to evaluate the potential advantages and viability of integrating triclustering techniques into an associative classification framework. Triclustering, unlike traditional biclustering methods, can simultaneously consider three dimensions of data, making it particularly highly appropriate for the intricate patterns present in high-dimensional biomedical datasets. This enhanced capability is expected to provide a more refined analysis, improving the predictive accuracy and robustness of the classifier [3].

The specific objectives of this study are as follows:

- **Develop a Temporal Associative Classifier:** Design and implement a classifier that leverages discriminative triclusters, which capture the temporal dynamics and complex interactions within the data, enabling the classifier to deliver more accurate predictions.
- **Evaluate Triclustering Efficacy:** Conduct a comprehensive evaluation to assess how the use of discriminative triclusters impacts the classifier's performance.

By addressing these objectives, this research aims to contribute to the field of biomedical data analysis, offering methodological advancements that can benefit a wide range of medical and scientific applications. The ultimate goal is to provide clinicians with reliable and interpretable tools that can aid in decision-making processes.

## 1.3 Problem

The primary challenge addressed in this research is the development of an effective associative classifier adapted for biomedical and clinical data, capable of making accurate predictions while

accounting for the temporality of the data, and ensuring that these predictions are interpretable within a clinical context.

This complexity arises due to several inherent factors in medical datasets, which include:

- **Temporal Components and Misalignments:** Medical datasets often contain time-series data with irregular intervals and temporal misalignments, complicating analysis and classification tasks [4]. The evolution of diseases over time provides significant insights by analyzing how biomarkers change throughout disease progression. Accurate temporal alignment and tracking of biomarker evolution are essential for effective classification [3].
- **High Dimensionality:** Biomedical data are typically high-dimensional, encompassing both patterned and non-patterned data. High dimensionality can lead to challenges in model training, such as overfitting, and can mask the underlying patterns within the data. Dimensionality reduction techniques or regularization methods are often necessary to manage these risks [5].

To clarify these challenges, this research utilizes generated datasets using a 3-way data generator tool. The key challenges addressed in this case study include:

- **Prediction Accuracy:** Accurately predicting the objects' class from temporal patterns identified by a triclustering algorithm.
- **Timely Prognosis:** Providing timely predictions is essential for effective clinical data management. However, current methods often fail to obtain accurate and timely prognoses. Accurate and timely predictions allow for better preparation and resource allocation, directly impacting patient quality of life.
- **Interpretability and Usability:** Clinicians require models that are not only accurate but also interpretable and easy to use in clinical settings. The usability of the model in real-world clinical context is essential to ensure its effective integration into decision-making processes [6].

To address these issues, this work explores the potential of triclustering algorithms. By incorporating the temporal component, this research enhances our understanding of biomedical data, crucial given the context of disease progression over time. This approach offers deeper insights into disease evolution and improves the predictive power of the models.

Despite the robustness of existing associative classifiers using biclustering algorithms like FleBiC [5], which is designed to discover non-constant patterns and is resilient to noise, it is not equipped to manage three-way data. Therefore, adapting these algorithms for triclustering represents a significant methodological advancement, creating new opportunities for analyzing multidimensional biomedical data.

Ultimately, this research aims to bridge the gap in predictive modeling of biomedical and clinical data, enhancing our understanding of disease markers and improving clinical decision-making

processes. By addressing these critical challenges, the research contributes both to methodological advancements and practical clinical applications, potentially transforming how we approach disease prediction and management.

## 1.4 Contributions

This research makes significant contributions to the field of biomedical data analysis, particularly in the development and application of a novel classification algorithm for clinical ends.

This study introduces a temporal associative classifier that leverages discriminative triclusters to capture and analyze temporal dynamics within biomedical datasets. By integrating triclustering methods, the proposed classifier enhances the robustness and accuracy of predictions when meaningful patterns exist in the temporal dimension, even in the presence of high-dimensional data.

This research demonstrates that triclustering can be applied and structured to improve both the interpretability of models and generalizability. This contribution provides the basis for future research to further explore and refine triclustering techniques in various data applications.

Finally, this research establishes a framework for future studies by addressing the complexities of high-dimensional data. It sets a precedent for the development of more sophisticated and effective classification algorithms.

## 1.5 Document Structure

This remaining document is organized as follows:

- **Chapter 2 – Literature Review:** Discusses the existing research on associative classifiers, pattern discovery in three-way temporal data, and triclustering algorithms and their applications in biomedical data.
- **Chapter 3 – Associative Classifier using Discriminative Triclusters:** Details the development and implementation of the proposed temporal associative classifier, including experimental results and analysis.
- **Chapter 4 – Conclusion and Future Work:** Summarizes the findings, discusses the implications, and suggests directions for future research.

# Chapter 2

# Backgroung and Literature Review

This chapter presents a comprehensive study of the foundational concepts and recent advancements that support this research. It begins with a background section, where the fundamental principles of associative classifiers, three-way temporal data analysis, and triclustering techniques are introduced. Following this, the related work section reviews the latest studies and methodologies in these areas, providing a critical analysis of their strengths and limitations. This chapter establishes the theoretical and methodological framework necessary for better understanding this work.

## 2.1 Background

This section provides a comprehensive overview of the fundamental concepts underlying associative classifiers, pattern discovery in three-way temporal data, and triclustering techniques. These methodologies are essential in analyzing high-dimensional data, identifying meaningful patterns, and enhancing predictive modeling capabilities. Understanding these concepts is fundamental for addressing the challenges and opportunities presented by three-way data analysis in biomedical and clinical contexts.

### 2.1.1 Associative Classifiers

Associative Classification (AC) is an approach in data mining that effectively combines association rule mining with classification tasks to construct predictive models [7, 8]. The fundamental idea behind AC is to leverage association rules, which represent significant relationships between features in a dataset, to enhance the accuracy and interpretability of classification models.

#### Key Concepts

Association rule mining focuses on discovering correlations, frequent patterns, associations, or causal structures among sets of items in transaction databases, relational databases, and other information repositories [7]. The core concepts of associative classifiers focus on:

- **Frequent Itemsets**: These are sets of items or features that frequently occur together in a dataset. Identifying frequent itemsets is the first step in generating association rules [9].

- **Association Rules**: These are implications of the form $X \rightarrow Y$, where $X$ and $Y$ are itemsets. The rule suggests that if $X$ occurs in a transaction, there is a significant probability that $Y$ will also occur. Association rules are characterized by two primary metrics: support and confidence [10].

- **Support and Confidence**:

  - *Support* quantifies how often an itemset appears in the dataset and is defined as:

$$Sup(X) = \frac{\text{Transactions containing } X}{\text{Total number of transactions}} \tag{2.1}$$

  - *Confidence* measures the reliability of the implication and is defined as:

$$Conf(X \rightarrow Y) = \frac{Sup(X \cap Y)}{Sup(X)} \tag{2.2}$$

**Building and Implementing an Associative Classifier**

The construction and implementation of an associative classifier involve several key stages, which together create a robust model for predictive analysis [5]. This subsection outlines the process, supported by a pseudo-algorithm (Algorithm (1)), to illustrate the practical application of these concepts. Figure (2.1) shows the workflow of a generic associative classifier.

- **Training Phase**: The training phase begins with identifying frequent itemsets from a dataset $D$ containing instances $I = \{i_1, i_2, ..., i_n\}$, where each instance has a set of features and a class label. These itemsets form the basis for generating association rules, which capture significant relationships between different features.

- **Rule Generation and Filtering**: For each frequent itemset, a frequent pattern mining algorithm is used to generate association rules. The confidence of each rule is computed to evaluate its predictive power. Rules that meet or exceed a predefined metric threshold are retained in the rule set $R$, ensuring that only the most relevant rules contribute to the classifier's accuracy and efficiency.

- **Testing Phase**: The rules present in $R$ are applied to new, unseen instances during the testing phase to predict their class labels. A voting mechanism determines the most probable class label based on the application of rules from $R$. The classifier's performance is assessed using evaluation metrics [11].

The associative classifier Algorithm (1) is implemented as follows:

1. **Frequent Itemset Mining:** The algorithm starts by identifying frequent itemsets from the dataset $D$.

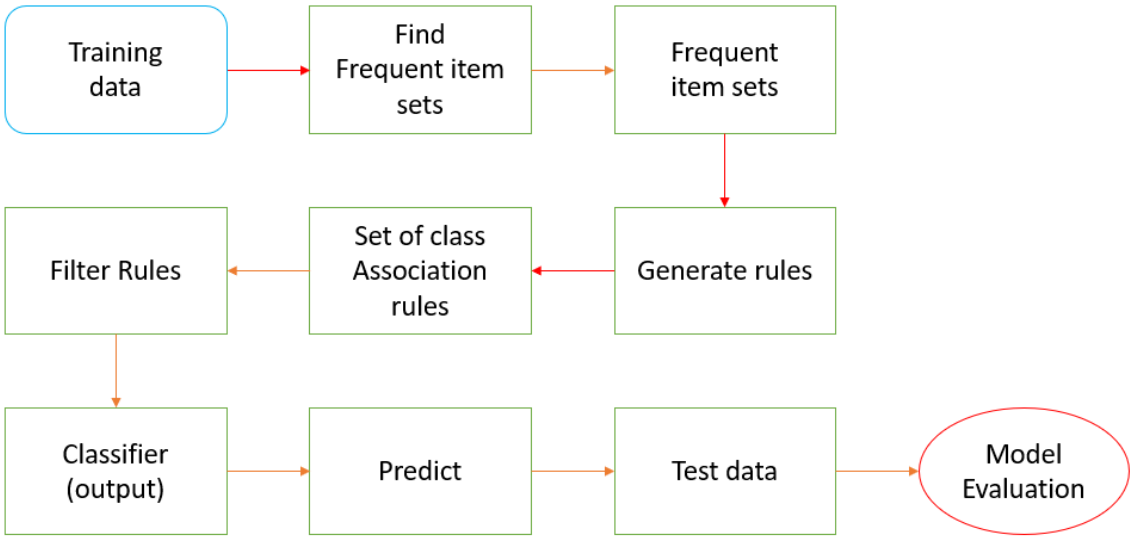


Figure 2.1: Associative Classifier workflow adapted from [5].

2. **Rule Generation:** Association rules are generated from each frequent itemset, with a focus on connecting itemsets to class labels. The predictive power of these rules is evaluated based on their confidence.

3. **Rule Filtering:** Rules that meet a predefined confidence threshold are retained in the rule set $R$.

4. **Classification of New Instances:** The classifier utilizes the rule set $R$ to predict class labels for new instances, employing a voting mechanism to determine the most likely class label.

A classifier's performance can be evaluated using multiple metrics:

- **Accuracy** (Equation (2.3)): Measures the overall correctness of the classifier.
- **Sensitivity (or Recall)** (Equation (2.4)): Assesses the classifier's ability to correctly identify positive instances.
- **Specificity** (Equation (2.5)): Evaluates the classifier's ability to correctly identify negative instances.
- **Precision** (Equation (2.6)): Quantifies the accuracy of positive predictions.
- **F1-score** (Equation (2.7)): Provides a balance between precision and recall.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{2.3}$$

$$Sensitivity\ (Recall) = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{2.4}$$

**Algorithm 1** Associative Classifier Algorithm

**Input:** Dataset $D$ with instances $I = \{i_1, i_2, ..., i_n\}$, each instance having a set of attributes and a class label.
**Output:** Classifier model $C$ and set of rules $R$.
Initialize $R = \emptyset$ {Initialize the rule set}
Find frequent itemsets $F$ using a minimum support threshold.
**for** each frequent itemset $f \in F$ **do**
  **for** each possible class label $c$ **do**
    Generate rule $r : f \rightarrow c$
    Calculate the confidence of $r$.
    **if** confidence of $r \geq$ minimum confidence threshold **then**
      Add $r$ to $R$
    **end if**
  **end for**
**end for**
**return** Rule set $R$
Initialize $votingTable = \emptyset$ {Initialize a voting table}
**for** each rule $r \in R$ **do**
  **if** instance $i$ satisfies the condition of rule $r$ **then**
    Increment the vote for class $c$ in votingTable
  **end if**
**end for**
**return** Class label with the highest votes in votingTable

$$Specificity = \frac{\text{True Negatives}}{\text{True Negatives + False Positives}} \tag{2.5}$$

$$Precision = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2.6}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision + Recall}} \tag{2.7}$$

By employing the associative classifier algorithm, researchers can construct robust models that leverage association rules for accurate and interpretable classification across various domains. This process highlights the integration of association rule mining with classification tasks, offering a powerful tool for data mining applications [7, 8, 11].

### Challenges and Considerations

One of the primary challenges in associative classification is handling class imbalance, which is prevalent in clinical and biomedical data [12]. Class imbalance occurs when certain classes are underrepresented in the dataset, leading to biased rule generation. Techniques such as ordered rule application and rule overlapping are employed to address this issue, ensuring that the classifier remains robust and unbiased.

- **Ordered Rule Application**: Rules in an associative classifier are applied in a specific order based on their relevance or specificity. This approach ensures that the most significant rule that matches an instance is used for classification, enhancing the classifier's precision [13].

- **Rule Overlapping**: In cases where multiple rules may apply to the same instance, a voting or weighting mechanism is used to combine the predictions of overlapping rules. This allows the classifier to capture complex relationships between features and improve its predictive accuracy [13].

Associative classifiers offer a powerful approach to data mining by integrating association rule mining with classification. Their ability to uncover hidden patterns and relationships in data makes them particularly valuable in domains such as healthcare, where accurate predictions can significantly impact decision-making processes [14, 15, 16]. Understanding the framework of associative classifiers establishes the basis for further exploration of advanced techniques and their applications in various fields.

### 2.1.2 Pattern Discovery in Three-Way Temporal Data

Pattern discovery in three-way temporal data is an essential task in analyzing complex datasets, particularly in biological studies involving genes and/or patients. This type of data often consists of multiple dimensions, such as individuals, features, and time points, making traditional clustering methods insufficient for capturing the intricate relationships present [2].

In traditional clustering and biclustering methods, the primary focus is on identifying patterns within two-dimensional datasets, such as gene expression across different conditions. However, when monitoring patients over time, the data becomes inherently more complex due to inconsistencies in time measurements and the presence of multiple interacting variables. This complexity necessitates the development of advanced techniques capable of analyzing three-way data, where synchronization of time measurements and the heterogeneity of subjects must be addressed [1].

Three-way data analysis involves identifying patterns across three dimensions simultaneously. This is crucial for capturing the dynamic interactions between different biological components, such as genes, proteins, or patient responses over time. By considering the temporal aspect, researchers can gain insights into the progression of diseases or the impact of treatments at various stages.

#### Challenges in Three-Way Temporal Data Analysis

One of the primary challenges in analyzing three-way temporal data is the inherent heterogeneity and asynchrony present in the datasets. Subjects may exhibit different responses at varying time points, and these responses can be influenced by numerous factors, leading to complex temporal patterns.

- **Heterogeneity**: The variability among subjects or samples in terms of their responses or characteristics can complicate the identification of consistent patterns. It is essential to

account for this heterogeneity to ensure that the discovered patterns are meaningful and applicable across different conditions or populations [2].

- **Asynchrony**: In many biological datasets, time measurements are not synchronized across subjects, which can lead to misinterpretation of patterns if not properly addressed. Techniques that allow for flexible alignment of time points are crucial for accurate pattern discovery [2].
- **Dimensionality**: The high dimensionality of three-way data poses computational challenges and increases the risk of overfitting. Efficient algorithms are needed to handle large datasets while maintaining the ability to discover significant patterns [17].
- **Complex Temporal Relationships**: The intricate temporal relationships in three-way data require sophisticated methods to uncover hidden patterns and associations. This includes dealing with noise, missing data, and the need for robust statistical measures to evaluate the significance of discovered patterns [18].

### Approaches to Pattern Discovery

To address these challenges, various methodologies have been developed, focusing on the flexibility and adaptability needed to analyze three-way temporal data effectively.

- **Triclustering**: This approach extends traditional clustering techniques to consider three dimensions, enabling the identification of patterns that span across subjects, features, and time. It allows researchers to detect coordinated activities that would be missed by two-dimensional analysis [19].
- **Fuzzy Association Analysis**: Fuzzy logic is a form of logic that allows for reasoning with degrees of truth rather than binary true/false values, making it ideal for handling uncertainty and imprecision in complex systems. By employing fuzzy logic, this method accounts for uncertainty and imprecision in the data, offering a way to model complex relationships. Fuzzy association rules help in identifying associations at different levels of detail, which is particularly useful in biological systems where interactions are often not evident [20].
- **Deep Learning Techniques**: Deep learning has emerged as a powerful tool for pattern discovery in temporal data. Its ability to learn hierarchical representations makes it adapted for capturing the complex, non-linear relationships typical of three-way data [21].
- **Local Similarity Analysis**: This technique focuses on identifying local patterns of similarity within the data, allowing researchers to uncover subtle but significant associations that may be indicative of underlying biological processes or disease mechanisms [22].

### Applications in Biomedical Research

The application of three-way temporal pattern discovery in biomedical research has significant implications for understanding complex biological systems. By analyzing multidimensional datasets,

researchers can gain insights into the dynamics of disease progression, identify potential biomarkers, and improve treatment strategies.

- **Multi-Omics Data Analysis**: Three-way pattern discovery is particularly valuable in multi-omics studies, where integrating data from various omics levels (e.g., genomics, proteomics, metabolomics) provides a comprehensive view of biological functions and interactions [23].
- **Clinical Studies**: In clinical settings, three-way data analysis helps in understanding patient responses to treatments over time, identifying patterns that correlate with successful outcomes, and adapted interventions to individual patient needs [24].
- **Systems Biology**: The integration of three-way temporal data is crucial for systems biology, where the goal is to model complex biological networks and understand how different components interact over time [25].

Pattern discovery in three-way temporal data is a crucial aspect of modern data analysis in biomedical research. By addressing the challenges of heterogeneity, asynchrony, and dimensionality, researchers can uncover meaningful patterns that provide valuable insights into biological systems. The ongoing development of advanced methodologies continues to enhance our ability to analyze and interpret complex datasets, paving the way for breakthroughs in personalized medicine and systems biology.

While pattern discovery in three-way temporal data provides valuable insights by capturing complex interactions across multiple dimensions, it often requires more sophisticated techniques to fully exploit the richness of such data. Traditional clustering approaches struggle to uncover meaningful patterns in high-dimensional contexts where correlations exist across time, features, and subjects. This limitation has led to the development of triclustering methods, which allow for the simultaneous analysis of three dimensions. Triclustering not only enhances the discovery of hidden patterns but also provides a more flexible framework for addressing the complexities inherent in multidimensional data analysis.

### 2.1.3 Triclustering

In various fields, such as biomedical and social sciences, three-way data is common. This data might include dimensions like gene-sample-time or individual-feature-time [17]. Traditional clustering methods, which group observations based on similarities, often present limitations when applied to three-way data, where meaningful correlations may only exist within specific subspaces. This limitation arises because conventional clustering techniques are typically designed for two-dimensional data, where they are unable to fully capture the complexity and interrelations present in higher dimensions. In response, triclustering has emerged as a method to address these challenges by grouping the data within the three dimensions [26].

Triclustering focuses on discovering coherent subspaces within three-way datasets. This technique allows for the identification of triclusters, which are groups of data points that exhibit

strong correlations across all three dimensions. Triclusters can have flexible structures, allowing any number of objects and overlapping based on specific criteria, such as coherency values [1]. This flexibility is vital for capturing the intricate relationships inherent in complex datasets.

### Criteria for Triclustering

The triclustering solutions can be evaluated using several criteria, primarily focusing on homogeneity and statistical significance [1]. These criteria include:

- **Structure**: This involves assessing the number, size, shape, and position of triclusters within the data. A well-structured tricluster should align with the inherent characteristics of the dataset.
- **Coherence**: Coherence is determined by the observed correlation of values within a tricluster. It measures how closely the values adhere to a specific pattern or expectation, with allowance for permissible deviations. This criterion is essential for identifying meaningful patterns in noisy or complex datasets.
- **Quality**: The quality of a tricluster is assessed based on the type and amount of noise tolerated. High-quality triclusters maintain their coherence despite the presence of noise, indicating robust pattern discovery.
- **Statistical Significance**: A tricluster is considered statistically significant if its occurrence probability is unexpectedly low compared to a null data model. This criterion ensures that the identified patterns are not due to random chance but represent genuine interactions within the data.

Figure 2.2 demonstrates how various triclustering structures can be identified based on the discussed criteria. For example, plaid triclusters often show cumulative effects in overlapping regions, while constant pattern triclusters maintain uniformity across observations and contexts. This visual aid supports understanding the complex nature of triclustering evaluations.

### Complexity and Challenges in Triclustering

Triclustering is a computationally challenging task, often classified as an NP-hard problem. This complexity arises from the need to find maximal hypercliques (constant triclusters) within graphs that exhibit multiwise interactions [27]. The difficulty is compounded by the need to accommodate non-trivial coherence forms, flexible structures, and noise tolerance, which necessitates the use of sophisticated algorithms. Many triclustering algorithms employ greedy or stochastic searches to find solutions, although these approaches may lead to suboptimal results [28]. Additionally, restrictions are often imposed on the permissible structure, coherence, and quality of triclusters, further complicating the discovery process [29].

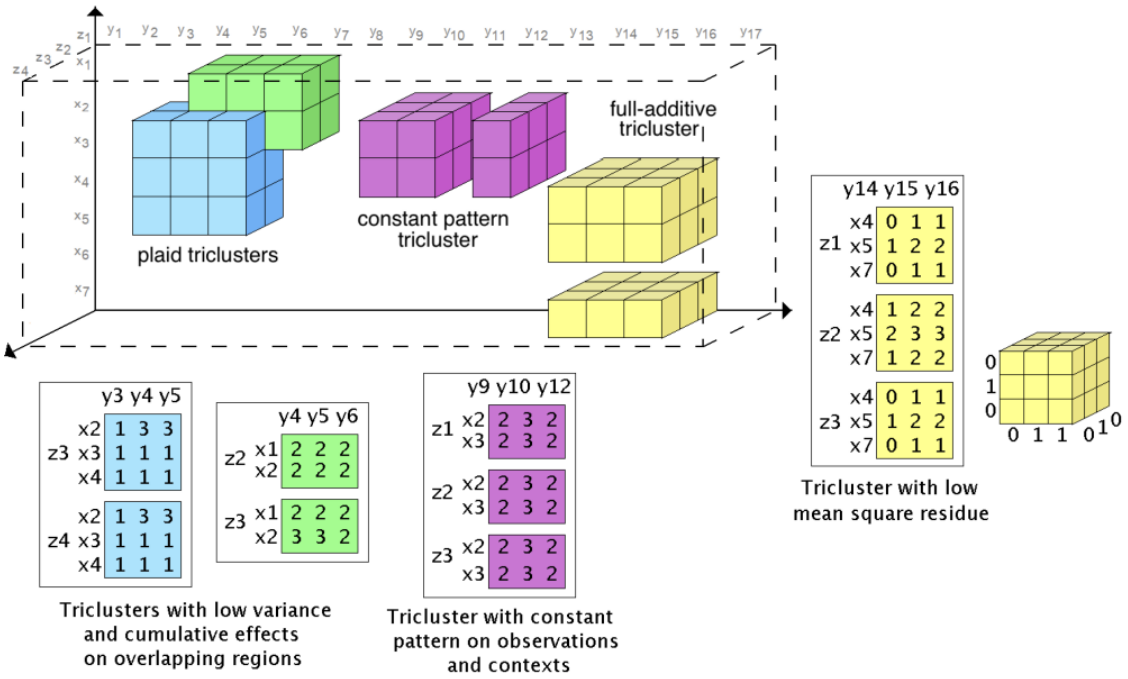


Figure 2.2: Illustration of different triclustering solutions with different coherencies. Image from [1].

### Methods for Evaluating Tricluster Quality

The quality of a tricluster is typically evaluated using merit functions, which consider the values of its elements. For example, variance-based merit functions minimize the variance within a tricluster to identify subspaces with consistent values. Pattern-based merit functions, on the other hand, assess the maximality of triclusters with well-defined patterns, often applied to symbolic or integer 3D data [30]. These methods directly relate to pattern discovery in three-way data [1, 3, 31, 32], where triclusters can reveal temporal lags and correlations among observations.

### Algorithmic Approaches in Triclustering

The diverse nature of triclustering tasks has led to the development of numerous algorithms over the past decade [1]. These algorithms can be classified based on their approach, such as iterative searches (e.g., greedy methods [29, 33]) or identifying distribution parameters (stochastic methods [2, 17]). Additionally, algorithms may offer optimality guarantees (exhaustive approaches [34]) or focus on pattern mining procedures [30]. The choice of algorithm often depends on the specific requirements of the dataset and the desired outcomes of the analysis.

Triclustering algorithms can also be categorized based on their objectives. Some aim to identify a single tricluster at a time, while others focus on subsets or the entire set of triclusters. In cases where multiple triclusters are identified, the algorithm may shift its focus to new subspaces or mask previous discoveries to uncover additional patterns [35].

#### Applications of Triclustering

Triclustering techniques have found practical applications in disease patterns [3], gene expression time series [19, 36, 37], protein interactions, and metabolite concentrations [38, 39]. In the context of biological systems, triclustering is instrumental in analyzing complex interactions between genes, proteins, and other biological entities [3, 19]. Moreover, in labeled datasets, triclustering can incorporate additional criteria to differentiate between classes, thus supporting real-world decision-making [40].

Additionally, triclustering is used to study multiple individuals experiencing extended awareness, enhancing the understanding of functional genomics, improving diagnostics, and identifying new drug targets [41].

Triclustering represents a powerful approach for analyzing complex three-way datasets across diverse fields. By focusing on the identification of coherent subspaces, triclustering facilitates the discovery of meaningful patterns that might otherwise remain hidden in high-dimensional data. As researchers continue to develop more sophisticated algorithms and techniques, the potential applications of triclustering will expand, offering new insights into complex biological, social, and economic systems.

## 2.2 Related Work

This section provides a comprehensive review of the latest advancements and methodologies in associative classifiers, pattern discovery in three-way temporal data, and triclustering algorithms. We examined key studies and their applications, highlighting the strengths and limitations of these techniques. This overview establishes the foundation for understanding the current state of data mining approaches and identifies areas for future research and development.

### 2.2.1 Associative Classifiers

Associative classifiers have acquired significant attention in the data mining community due to their ability to integrate association rule mining with classification tasks. This integration allows for the creation of models that are both highly interpretable and accurate, making them particularly useful in diverse domains that require transparency and precision. Notable applications include healthcare for improving diagnostic accuracy and patient management, and finance for predicting stock prices and enhancing trading strategies.

#### Notable Studies and Applications

Associative classifiers have evolved to address various challenges in different domains. This section reviews recent studies that demonstrate the advancements and applications of associative classifiers.

**Advancements in Algorithmic Techniques**: The development of associative classifiers has been marked by continuous improvements in algorithmic techniques aimed at enhancing both

efficiency and interpretability. Baralis and Garza [42] developed the I-Prune algorithm, which introduced a novel item selection method to improve classification accuracy while reducing complexity. Their approach utilizes a pruning strategy that identifies and eliminates redundant rules during the rule-mining phase rather than as a post-processing step. This method involves calculating a relevant measure for items and pruning those that do not meet a predefined threshold, effectively reducing the number of rules generated and thereby decreasing model generation time. The I-Prune algorithm demonstrated significant improvements in model generation efficiency, with an average time reduction of around 30% using Chi-square threshold compared to Yule's Y and Jaccard thresholds, while either increasing or preserving model accuracy. This method proves particularly beneficial in scenarios where datasets are large and contain numerous irrelevant features, as it ensures that only the most relevant rules are included in the final model.

Tawiah and Sheng [43] extended this work by applying associative classifiers to multi-label classification problems. They developed a framework that effectively manages datasets with multiple labels by utilizing association rules that can predict multiple outcomes simultaneously. Their approach combines algorithm adaptation and problem transformation techniques to handle multi-label data, providing a robust solution for complex classification tasks. This is especially relevant in fields such as genomics, where genes can be associated with multiple traits or conditions. Their empirical studies demonstrated that the adaptive Multi-Label K-Nearest Neighbor (ML-KNN) algorithm, which considers label correlations, performed the best among various multi-label classification algorithms. This highlights the versatility and adaptability of associative classifiers in handling complex datasets and underscores their potential in various real-world applications, from medical diagnosis to document classification.

**Healthcare Applications**: In healthcare, associative classifiers have shown great promise in improving diagnostic accuracy and patient outcomes. Jabbar et al. [8] utilized associative classifiers for heart disease prediction, demonstrating that the model could accurately identify critical risk factors from patient data. By mining association rules, their approach could uncover significant patterns that enhance clinical decision-making. This methodology underscores the potential of associative classifiers to support medical professionals in diagnosing conditions and adapting treatment plans based on identified risk factors.

Farghali et al. [44] further explored healthcare applications by applying associative classifiers to chronic disease diagnosis. Their research focused on developing a hybrid model that combines associative rule mining with machine learning techniques, specifically Support Vector Machines (SVM). This approach aimed to enhance prediction accuracy and reduce the number of generated classification rules. The proposed model integrates association rule mining with SVM to build an efficient and accurate classifier. The methodology includes a Sequential Forward Selection (SFS) based framework for feature selection, optimizing the feature subset, and improving the mining process's efficiency. By using the Gini index for attribute selection, the number of generated rules is minimized, ensuring that only the most informative rules are retained. This integration highlights the potential of associative classifiers to handle complex and high-dimensional data typical in

healthcare environments, providing insights that could lead to early interventions and better patient management. The empirical results demonstrated that this hybrid approach outperformed traditional classification methods in terms of classification accuracy and F-measure, indicating its effectiveness in predicting chronic conditions.

Christopher [45] explored the application of associative classifiers in clinical datasets. Their research focused on improving the efficiency of rule-based classifiers by analyzing the effect of different rule selection approaches and case satisfaction mechanisms. They proposed a hybrid rule combination approach called rule pool, which combines the best rules from decision tree classifiers and associative classifiers. This hybrid approach demonstrated improved classification accuracy in clinical datasets, showing the potential for associative classifiers to be applied in healthcare for knowledge mining and decision support.

**Financial analytics**: Associative classifiers have been explored for various applications in financial analytics. Attanasio et al. [46] employed associative classifiers to develop a stock trading system that utilizes association rules for predicting next-day stock price directions. Their model leverages association rules to identify patterns in stock trading data, offering a transparent approach to generating trading signals. This method allows for the identification and ranking of classification rules that correlate stock descriptors with prediction targets, providing financial analysts with interpretable insights into market behaviors. The backtesting results on 11 years of U.S. market data demonstrated the system's robustness and its potential to enhance decision-making processes in financial markets, highlighting the versatility of associative classifiers in financial analytics.

**Education**: Rajeswari and Deisy [47] applied associative classifiers in the educational domain to predict slow learners. Their study focused on developing a fuzzy logic-based associative classifier that utilizes fuzzy temporal outlier detection (FTOD) methods to identify students who are at risk of poor academic performance. By processing mid-semester examination results, the proposed model effectively distinguishes between actual slow learners and false positives low learners, who might intentionally score low. This capability is crucial in providing targeted interventions to improve academic outcomes, illustrating the significant impact of associative classifiers in the education sector.

**Integration with Big Data and Large-Scale Systems**: Almasi and Saniee Abadeh [48] extended the application of associative classifiers to manage large-scale datasets, illustrating how these classifiers can handle big data efficiently. Their study introduced the CARs-Lands framework, which combines associative rule mining with evolutionary algorithms to create a distributed associative classifier adapted for large datasets. The framework addresses the challenges of memory and time-complexity by generating sub-datasets and employing parallel local association rule mining. This approach enhances the accuracy of predictions by leveraging local patterns within the data, making it a robust solution for various large-scale applications, including smart grids and other big data environments. The findings demonstrate the potential of associative classifiers in processing large amounts of data, thereby supporting sustainable practices and optimizing resource distribution.

**Network Intrusion Detection Systems**: Sivanantham and Ranganathan [49] discussed the use of associative classifiers in intrusion detection systems (IDS). Their study proposed a novel pruning measure named Rule Precision Index (RPI), which enhances the efficiency and accuracy of associative classifiers. The RPI measure helps in effectively pruning association rules, thus improving the classifier's performance in detecting network intrusions. By implementing the RPI classifier on datasets such as NSL-KDD, CICIDS-2017, and KDDCUP99, the study demonstrated significant improvements in accuracy and detection rates compared to traditional methods. This innovative approach highlights the potential of associative classifiers in enhancing network security by efficiently identifying and mitigating intrusion attempts.

### Challenges

Despite their numerous advantages, associative classifiers face several significant challenges that need to be addressed to fully leverage their potential across various domains.

**Handling Imbalanced Datasets**: One of the primary challenges is dealing with imbalanced datasets, which are prevalent in many real-world applications, particularly in healthcare and fraud detection. Imbalanced datasets can lead to biased classifiers that perform well on the majority class but poorly on the minority class. Various techniques such as oversampling, undersampling, and the development of new algorithms specifically designed to handle imbalance are critical areas for future research [12].

**Scalability to Large Datasets**: Another major challenge is scaling associative classifiers to handle large datasets efficiently. As data continues to grow exponentially, traditional associative classification algorithms struggle with memory and computational requirements. Researchers have started addressing this issue by developing distributed and parallel processing algorithms, but there is still a need for more efficient and scalable solutions [48].

**Integration with Deep Learning Techniques**: Integrating associative classifiers with deep learning techniques presents a promising direction for future research. Deep learning models are powerful for automatic feature extraction and handling high-dimensional data, but they often lack interpretability. Associative classifiers can complement deep learning by providing interpretable rules based on the features extracted by deep learning models. This integration could lead to more powerful and transparent models [50].

**Expanding to New Domains**: Expanding the application of associative classifiers to new and emerging domains offers new opportunities. For instance, in the context of smart cities, associative classifiers could be used to analyze traffic patterns and optimize urban planning [51]. In environmental monitoring, they could help in predicting and managing natural disasters by analyzing sensor data [52]. Exploring and adapting associative classifiers to these and other new domains will be crucial for their continued relevance and impact.

In summary, while associative classifiers have demonstrated significant potential across various domains, addressing these challenges through ongoing research and innovation will be key to unlocking their full capabilities. By improving scalability, integrating with advanced techniques,

and expanding their application areas, associative classifiers can continue to provide valuable insights and support decision-making.

Table 2.1: Summary of Associative Classifiers studies.

| **Study** | **Methodology** | **Application Domain** | **Key Contributions** |
|---|---|---|---|
| Baralis and Garza [42] | I-Prune algorithm | Algorithmic Techniques | Improved classification accuracy and efficiency by pruning redundant rules. |
| Tawiah and Sheng [43] | Multi-label classification framework | Genomics | Effective management of multi-label datasets, improved performance of ML-KNN algorithm. |
| Jabbar et al. [8] | Associative classifiers for heart disease prediction | Healthcare | Enhanced diagnostic accuracy, identification of critical risk factors. |
| Farghali et al. [44] | Hybrid model combining associative rule mining and SVM | Healthcare | Improved prediction accuracy for chronic diseases, efficient handling of high-dimensional data. |
| Christopher [45] | Hybrid rule combination approach | Clinical datasets | Improved classification accuracy in clinical datasets, enhanced knowledge mining. |
| Attanasio et al. [46] | Associative classifiers for stock trading system | Financial analytics | Accurate prediction of stock price directions, robust decision-making process. |
| Rajeswari and Deisy [47] | Fuzzy logic-based associative classifier | Education | Effective prediction of slow learners, targeted interventions to improve academic outcomes. |
| Almasi and Saniee Abadeh [48] | CARs-Lands framework | Big Data | Efficient handling of large-scale datasets, improved prediction accuracy. |
| Sivanantham et al. [49] | Rule Precision Index (RPI) for pruning | Network Intrusion Detection Systems | Enhanced accuracy and efficiency in detecting network intrusions. |

### 2.2.2 Pattern Discovery in Three-Way Temporal Data

Pattern discovery in three-way temporal data involves analyzing datasets where three dimensions vary over time. This complex data structure requires advanced techniques to identify meaningful patterns across these dimensions, offering insights into various domains such as healthcare, biological networks, social networks, and traffic data recovery.

#### Applications in Healthcare

The healthcare domain significantly benefits from pattern discovery in three-way temporal data, providing valuable insights into patient care, disease progression, and treatment outcomes. For

instance, Wang et al. [53] utilized a tensor-based temporal multi-task survival analysis to predict patient outcomes by analyzing Electronic Health Records (EHRs). This model simultaneously handles multiple prediction tasks across various time points, capturing inter-task correlations and intra-task temporal smoothness, which enhances survival predictions. The use of tensor decomposition not only identifies patterns indicative of patient outcomes but also provides a robust mechanism for managing the complexities inherent in longitudinal healthcare data, leading to improved personalized medicine strategies and optimized patient care processes.

Nesaragi et al. [54] introduced a novel approach for early sepsis prediction using EHRs through tensor learning of pointwise mutual information (PMI). By constructing PMI matrices for each patient and applying Tucker decomposition, the study effectively extracts latent features representing temporal interactions among clinical variables. These features, used in a gradient boosting machine learning framework, significantly improve prediction performance, achieving an AUROC of 0.8621. This approach emphasizes capturing complex relationships within EHR data, offering promising directions for improving patient monitoring and intervention strategies in critical care settings.

Tiantian Li et al. [55] explored the use of three-way temporal data analysis for early sepsis detection in Intensive Care Unit (ICU) patients. Their novel approach integrates multimodal fusion with a reference-based strategy, specifically targeting the fusion of different Magnetic Resonance Imaging (MRI) modalities to uncover covarying patterns associated with medical diagnoses. While primarily focused on autism spectrum disorder (ASD), the methodology demonstrated potential for ICU settings by examining interactions between vital signs, laboratory results, and time, thereby enhancing patient outcomes through early detection. The results highlighted the potential of this approach to improve survival rates and reduce ICU stays.

Shile et al. [56] investigated patterns in mental health, focusing on schizophrenia using three-way multimodal fusion techniques. The study utilized a novel approach called GICA+pICA, integrating spatial and spatio-temporal data from structural MRI (sMRI), diffusion MRI (dMRI), and functional MRI (fMRI). By incorporating temporal information from fMRI, the method improves the detection of linked components across different imaging modalities, revealing critical insights into the neural mechanisms underlying mental health disorders. This analysis potentially informs more effective therapeutic interventions and advances our understanding of complex psychiatric conditions.

#### Applications in Biological Networks

Dongqi et al. (2022) introduced a framework named Temp-GFSM for learning metrics over temporal graphs, effective in both biological and social network domains [57]. In biological networks, the framework is applied to protein-protein interaction networks, capturing interaction dynamics within metabolic cycles of yeast cells. By modeling both continuous and discrete evolution patterns, Temp-GFSM significantly improves the classification of these dynamic interactions, enhancing the understanding of cellular processes and molecular biology.

### Applications in Social Networks

The Temp-GFSM framework's versatility extends to social networks, where it analyzes human-contact networks to uncover complex interaction patterns and community dynamics [57]. This application demonstrates the framework's ability to handle diverse datasets, making it a powerful tool for exploring temporal graph structures in various fields beyond its original purpose. The use of multi-time evolution modeling and few-shot learning further enhances its adaptability and accuracy in predicting class-specific patterns, setting a new baseline for temporal graph metric learning.

### Application in Traffic Data Recovery

Xinxin et al. [58] introduced a novel approach to traffic data recovery using three-way temporal data analysis. Their method, *spatial-temporal tensor robust principal component analysis* (ST-TRPCA), is designed to handle corrupted and incomplete traffic data observations. The ST-TRPCA approach effectively utilizes spatial-temporal properties to recover missing and erroneous entries, enhancing both accuracy and computational efficiency. This method enables better predictions of traffic flow and congestion patterns, emphasizing the importance of three-way temporal data analysis in urban traffic management systems.

### Challenges

Despite the advantages of three-way temporal data analysis, several challenges remain. One significant limitation is the computational complexity associated with handling large datasets [57]. Efficient algorithms and scalable computing resources are essential for processing such data effectively.

Another challenge is the potential for noise and missing data in three-way temporal datasets, which can affect pattern discovery accuracy. Xinxin et al. [58] emphasize the need for robust data preprocessing techniques to address these issues, ensuring reliable analysis outcomes. They propose the *Spatial-Temporal Tensor Robust Principal Component Analysis* (ST-TRPCA) method to handle incomplete or corrupted traffic data, highlighting the importance of addressing data quality issues.

Interpretability is another significant challenge in three-way temporal data analysis. The complexity of the data can make it difficult for researchers to explore clear conclusions or actionable insights. Models often produce intricate patterns that require domain expertise for accurate interpretation [55]. Developing methods that balance complexity with interpretability is crucial to ensure that insights gained can be practically applied.

Scalability is vital when dealing with large-scale data, particularly for real-time analysis applications. Nesaragi et al. [54] note that many existing methods struggle with scaling to larger datasets or providing timely results. This is especially relevant in healthcare and finance, where timely decisions are critical. Advances in parallel computing and optimized algorithms are needed to address these challenges effectively.

Three-way temporal datasets often include heterogeneous data types, such as numerical, categorical, and textual information, presenting challenges in integrating and analyzing different data

types coherently. Methods that can handle such heterogeneity without losing valuable information or introducing biases are needed [57].

These challenges underscore the need for ongoing research and innovation in three-way temporal data analysis to overcome current limitations and unlock its full potential across various domains.

Table 2.2: Summary of Pattern Discovery in Three-Way Temporal Data studies.

| **Study** | **Methodology** | **Application Domain** | **Key Contributions** |
|---|---|---|---|
| Wang et al. [53] | Tensor-based temporal multi-task survival analysis | Healthcare (Electronic Health Records) | Improved patient outcome prediction, enhanced personalized medicine strategies. |
| Nesaragi et al. [54] | Tensor learning of pointwise mutual information (PMI) | Healthcare (Sepsis Prediction) | Early sepsis prediction, improved ICU patient monitoring. |
| Tiantian Li et al. [55] | Multimodal fusion with reference-guided strategy | Healthcare (Sepsis Detection) | Enhanced sepsis detection, improved patient survival rates. |
| Shile et al. [56] | Three-way multimodal fusion (GICA+pICA) | Mental Health (Schizophrenia) | Insights into schizophrenia neural mechanisms, potential therapeutic interventions. |
| Dongqi et al. [57] | Temp-GFSM for temporal graphs | Biological and Social Networks | Improved classification of dynamic interactions, insights into cellular and social network dynamics. |
| Xinxin et al. [58] | Spatial-temporal tensor robust principal component analysis (ST-TRPCA) | Traffic Data Recovery | Accurate traffic data recovery, improved urban traffic management. |

### 2.2.3 Triclustering applied to Temporal Data

Triclustering has emerged as a powerful technique for analyzing three-way data, where data points are considered across three dimensions simultaneously -[1]. This approach is particularly useful in fields such as gene expression, where complex data structures require sophisticated methods for pattern discovery [19, 36, 37]. The specific triclustering algorithm employed in this work will be discussed in detail in the methodology section.

#### State-of-the-Art Triclustering Algorithms applied to Temporal Data

Several state-of-the-art triclustering algorithms have been developed to effectively handle three-way temporal data. These algorithms are designed to uncover hidden patterns across three dimensions, providing valuable insights into various applications. Here, we discuss some notable triclustering

algorithms. These algorithms were selected to provide a comprehensive view of the various methodological approaches employed in triclustering. By including heuristic-based (triCluster [19]), genetic alogithm-based (TriGen [36]), and optimization-based ($\delta$-TRIMAX [37]) methods, this thesis offers insights into the strengths and weaknesses of different algorithmic strategies.

### triCluster

The triCluster algorithm, developed by Zhao and Zaki, is the pioneering triclustering algorithm designed for detecting coherent clusters in three-way gene expression data [19]. This algorithm employs a heuristic approach that focuses on optimizing the overlap between triclusters to ensure that the identified patterns are statistically significant and biologically relevant. triCluster is particularly effective in handling the inherent complexity and noise present in biological datasets, making it a robust tool for bioinformatics research.

triCluster operates by initially identifying potential triclusters and then refining them through an iterative process that maximizes the overlap among them. This refinement is guided by statistical significance tests to ensure that the detected triclusters are meaningful and biologically relevant. The algorithm's effectiveness is demonstrated through its application to gene expression datasets, where it reveals significant patterns of gene interaction and regulatory dynamics that are critical for understanding complex biological processes.

The primary advantage of triCluster is its ability to handle the inherent complexity and noise in biological data, making it a robust tool for bioinformatics research. By focusing on overlapping triclusters, triCluster provides a more comprehensive view of the data, capturing intricate relationships that simpler models might miss. This capability is particularly valuable in genomics, where understanding the multifaceted interactions between genes across different conditions and time points is essential.

### TriGen

The TriGen algorithm, proposed by Gutiérrez-Avilés et al., employs a genetic algorithm-based approach for triclustering temporal gene expression data [36]. This method leverages evolutionary strategies to explore the search space and identify high-quality triclusters, specifically focusing on gene expression data across time points and experimental conditions. By encoding potential solutions as chromosomes, TriGen applies crossover and mutation operations to generate new triclusters, gradually improving the quality of the solutions through an iterative process.

The TriGen algorithm begins by initializing a population of potential triclusters, each represented by a subset of genes, conditions, and time points. These individuals be subjected to evaluation using two distinct fitness functions. The first fitness function is based on the Mean Squared Residue (MSR) measure, adapted for three dimensions, which evaluates the similarity of gene expression values within a tricluster. The second fitness function uses a least squares approximation to measure the correlation of expression patterns, considering both coherent values and coherent behaviors.

TriGen's evolutionary process includes selection, crossover, and mutation operators designed to optimize the triclusters iteratively. The selection operator employs a random method to choose individuals for reproduction based on their fitness scores. Crossover operations combine the genetic material of parent triclusters to produce offspring, ensuring genetic diversity and the exploration of new areas in the search space. Mutation operations introduce random changes to individuals, further enhancing variability and the potential discovery of optimal solutions.

The algorithm's effectiveness has been demonstrated through applications to both synthetic datasets and real-world gene expression data, such as yeast cell cycle and human inflammation and host response to injury datasets. TriGen successfully identified biologically meaningful triclusters that were validated using Gene Ontology annotations, highlighting its capability to uncover intricate patterns in temporal gene expression data.

Overall, TriGen exemplifies the application of genetic algorithms in triclustering, showcasing its potential in bioinformatics for extracting significant patterns from high-dimensional and complex gene expression datasets [36].

#### $\delta$-TRIMAX

The $\delta$-TRIMAX algorithm, developed by Siswantining et al., is designed to identify triclusters within three-way datasets, particularly gene expression data, by minimizing the mean squared residue (MSR) below a predetermined threshold $\delta$ [37]. This algorithm employs a series of iterative processes, including multiple node deletion, single node deletion, and node addition, to optimize the tricluster quality. By applying the $\delta$-TRIMAX method, the algorithm can efficiently identify homogeneous submatrices within the data that reveal significant biological patterns.

The $\delta$-TRIMAX method uses K-means clustering and the silhouette coefficient to determine an appropriate threshold $\delta$ that maximizes tricluster quality. The algorithm then proceeds to iteratively refine triclusters through a greedy search approach, ensuring that the residual value (difference between actual and estimated values) is minimized. The primary goal is to achieve a tricluster with an MSR lower than the threshold, which indicates high homogeneity.

In the context of gene expression data, the $\delta$-TRIMAX algorithm has demonstrated its effectiveness in identifying critical gene interactions and biological processes. For instance, in a study involving patients with pulmonary arterial hypertension (PAH), the algorithm successfully identified triclusters that exhibited high-quality indices, highlighting its potential in early detection and understanding of complex diseases such as heart failure. The use of the $\delta$-TRIMAX method allows researchers to uncover intricate patterns within high-dimensional data, providing valuable insights into disease mechanisms and therapeutic targets.

#### Applications in Gene Expression Analysis

Triclustering algorithms have been extensively applied to gene expression data [19, 36, 37]. For example, triCluster has been used to identify coherent clusters of genes, conditions, and time points, revealing significant patterns of gene interactions and regulatory mechanisms critical for

understanding complex biological processes [19]. Similarly, TriGen employs genetic algorithms to analyze temporal gene expression data, effectively identifying high-quality triclusters that capture the dynamics of gene expression over time and across different conditions [36]. The $\delta$-TRIMAX algorithm further refines this approach by minimizing the mean squared residue, ensuring the identification of homogeneous triclusters that provide insights into gene interactions and disease mechanisms [37].

### Challenges

Despite the advancements in triclustering algorithms, several challenges remain that need to be addressed to leverage their potential in various domains.

**Scalability and Computational Complexity**: One of the primary challenges is the scalability and computational complexity of triclustering algorithms. Handling large-scale three-way datasets requires significant computational resources and efficient algorithms. For instance, the TriGen algorithm leverages genetic algorithms, which can be computationally intensive due to the iterative nature of the evolutionary process [36]. Similarly, $\delta$-TRIMAX involves multiple iterations of node deletion and addition to refine triclusters, which can be time-consuming for large datasets [37]. Future research should focus on developing more efficient algorithms that can scale to larger datasets while maintaining accuracy and robustness.

**Handling Noise and Missing Data**: Triclustering algorithms often face challenges related to noise and missing data in three-way temporal datasets. Noisy data can obscure meaningful patterns, while missing data can lead to incomplete or biased triclusters. The $\delta$-TRIMAX algorithm addresses this to some extent by minimizing the MSR, but developing robust preprocessing techniques and algorithms that can handle noise and missing data effectively is crucial for improving the reliability of triclustering results [37].

**Application to New Domains**: While triclustering algorithms have been extensively applied to gene expression data, their potential in other domains remains relatively unexplored. Future research should investigate the application of triclustering in areas such as social network analysis, environmental monitoring, and financial analytics. For example, in social network analysis, triclustering could uncover patterns of interaction over time across different user groups and activities. In environmental monitoring, these algorithms could help analyze temporal-spatial data to track pollution levels and climate change indicators. Similarly, in financial analytics, triclustering could be utilized to detect temporal trends and anomalies in stock market data. Exploring new applications will help demonstrate the versatility and utility of triclustering algorithms across different fields, thereby broadening their impact and fostering interdisciplinary innovations.

In summary, addressing these challenges requires a multidisciplinary approach that combines advances in algorithm design, computational efficiency, data preprocessing, and domain-specific knowledge. By overcoming these challenges, future research can unlock the full potential of triclustering algorithms, enabling them to provide valuable insights across a wide range of applications.

The review of related work underscores the significant progress made in associative classifiers,

Table 2.3: Summary of Triclustering Algorithms applied to temporal data.

| **Study** | **Methodology** | **Application Domain** | **Key Contributions** |
|---|---|---|---|
| Zhao and Zaki [19] | Heuristic approach for optimizing tricluster overlap | Gene Expression Data | Identified significant gene interactions and regulatory mechanisms. |
| Gutiérrez-Avilés et al. [36] | Genetic algorithm-based triclustering | Gene Expression Data | Uncovered high-quality triclusters, validated through Gene Ontology annotations. |
| Siswantining et al. [37] | $\delta$-TRIMAX with MSR minimization | Gene Expression Data | Identified homogeneous triclusters, providing insights into gene interactions and disease mechanisms. |

pattern discovery in three-way temporal data, and triclustering algorithms. While these techniques have demonstrated substantial promise across various domains, including healthcare, finance, and bioinformatics, several challenges persist. Issues such as handling imbalanced and noisy datasets, scaling to large volumes of data, and maintaining interpretability without sacrificing accuracy continue to drive research efforts. Future directions suggest integrating advanced machine learning techniques, such as deep learning, and exploring new application areas to enhance the versatility and impact of these methods. Addressing these challenges will be key in unlocking the full potential of data mining technologies and encouraging interdisciplinary innovations.

# Chapter 3

# Associative Classifier using Discriminative Triclusters

This chapter details the development, evaluation, and analysis of the proposed associative classifier using discriminative triclusters for biomedical data analysis. It begins with a comprehensive methodology section that outlines the triclustering approach, the generation of rules, and the training and testing phases of the classifier. The results section then presents the performance evaluation of the classifier across various datasets, highlighting its accuracy, specificity, and sensitivity compared to other Triclustering-Based models. Finally, the discussion section offers a critical analysis of the results, examining the effectiveness of the classifier, the challenges encountered, and the implications for future research in clinical data analysis.

## 3.1 Methodology

In this methodology, we employ discriminative triclustering to enhance temporal associative classification, which can be applied to numerical, categorical, or mixed datasets, specifically for biomedical data analysis. Initially, we define a three-way dataset comprising observations, features, and contexts, from which triclusters are derived. The discriminative strength (DS) of each tricluster is calculated to determine its representativeness for a given class. Triclusters with high DS values are selected for further analysis. Transactions and rules are generated from these triclusters, focusing on their coverage and diversity. The model undergoes training and testing phases, where numeric and categorical rules are evaluated for their fit. The process results in a modular classifier that provides interpretable rules, facilitating both predictive accuracy and feature engineering.

### 3.1.1 Scheme of the Associative Classifier

Figure (3.1) presents a summary of the model's training phase and Figure (3.2) is a summary of the model's testing phase.

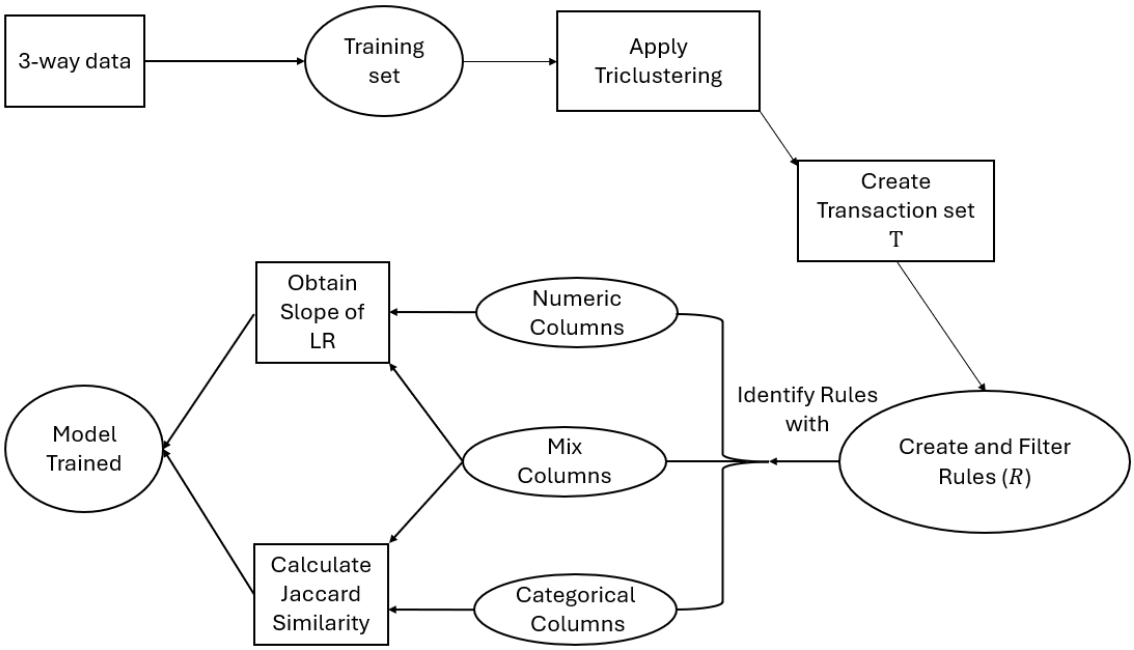


Figure 3.1: Summary of the training phase of our Associative Classifier.

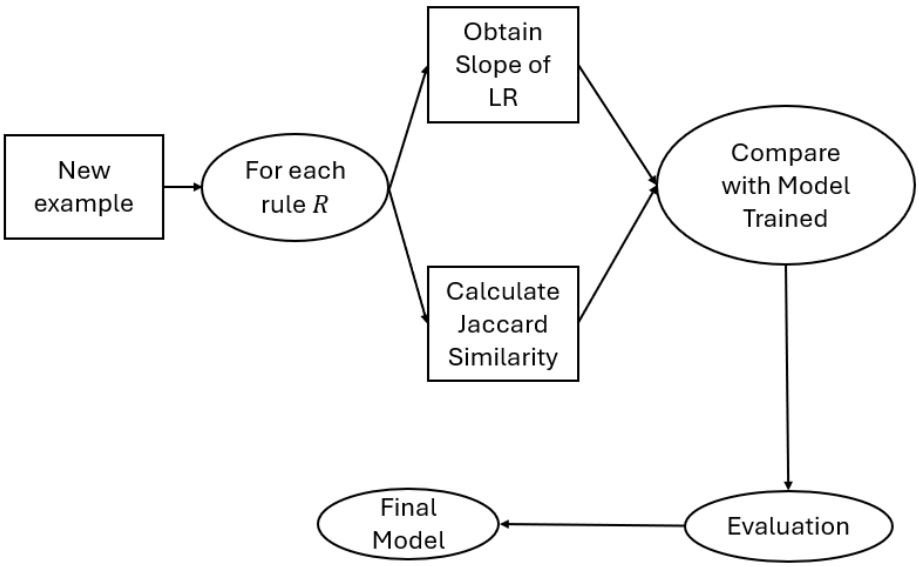


Figure 3.2: Summary of the testing phase of our Associative Classifier.

### 3.1.2 Global Definitions

**Definition 1.** A three-way dataset, $D$, comprises $n$ observations $X = \{x_1, \ldots, x_n\}$, $m$ features $Y = \{y_1, \ldots, y_m\}$, and $p$ contexts $Z = \{z_1, \ldots, z_p\}$, where the elements $d_{ijk}$ relate observation $x_i$, feature $y_j$, and context $z_k$. We denote the set of columns as the concatenation of $y_j$ and $z_k$, represented as $y_j z_k$. Given a set of $r$ classes, $C = \{c_1, \ldots, c_r\}$, a labeled three-way dataset is the set of triples $\{(x_i, Y, Z, c_{ik}) \mid i = 1, \ldots, n;\ c_i \in C\}$, where $x_i$ is an observation, $z_k$ is a context, and $c_{ik}$ is the class of the observation $x_i$ in context $z_k$. Table 3.1 shows an example of a three-way

dataset.

Table 3.1: Ilustrative example of a three-way dataset composed by $n$ observations, $m$ features and $p$ contexts. Each cell $y_i z_j$ represents the measurement of the $i$-th feature under the $j$-th context for the corresponding observation $x$.

| $X$ | $y_0z_0$ | $y_1z_0$ | ... | $y_mz_0$ | $y_0z_1$ | ... | $y_mz_p$ | ... | class |
|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 4 | 10 | ... | 2.5 | 6 | ... | 4.2 | ... | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| $x_n$ | 3 | 5 | ... | 1.2 | 3 | ... | 5.4 | ... | 2 |

**Definition 2.** Given a three-way dataset, a tricluster $B = (I, J, K)$ is a subspace of the original space, where $I \subseteq X$, $J \subseteq Y$, and $K \subseteq Z$ are subsets of observations, features, and contexts, respectively. Table 3.2 illustrates an example of a tricluster with only categorical columns and Table 3.3 represents an example of a tricluster with only numerical columns.

Table 3.2: Example of a tricluster composed by 3 observations (rows), and by 1 attribute and 3 contexts (columns). 2 observations belong to the class 0 and 1 observation belongs to the class 1.

| X | $y_5z_3$ | $y_5z_4$ | $y_5z_5$ | class |
|---|---|---|---|---|
| $x_5$ | 2 | 3 | 4 | 0 |
| $x_7$ | 2 | 4 | 3 | 1 |
| $x_{13}$ | 2 | 3 | 4 | 0 |

Table 3.3: Example of a tricluster composed by 2 observations (rows), and by 2 attributes and 3 contexts (columns). All observations belong to the same class.

| $X$ | $y_4z_2$ | $y_4z_3$ | $y_4z_4$ | $y_9z_3$ | $y_9z_4$ | $y_9z_5$ | class |
|---|---|---|---|---|---|---|---|
| $x_{11}$ | 3 | 4 | 5 | 3 | 3.5 | 4 | 2 |
| $x_{16}$ | 3 | 4 | 5 | 3 | 3.5 | 4 | 2 |

### 3.1.3 Dataset Characteristics and Triclustering Application

We proceed by applying a triclustering algorithm to our training set. The outcome of this application obtains a tricluster solution.

Normalization is performed on the train set to scale the features within each dataset to a similar range, which aids in preventing certain features from dominating the learning process. This ensures that each feature contributes proportionally to the model's training process, regardless of its initial scale.

**Definition 3.** The discriminative strength (DS) defines how discriminative a given tricluster, $t$, is for a given class, $c$. It ranges between 0 and 1 and is computed as follows:

$$\text{DS}(t, c) = \frac{\{x_i \mid C_{x_i} = c \wedge x_i \in t^X\}}{|t^X|} \tag{3.1}$$

$\{x_i \mid C_{x_i} = c \wedge x_i \in t^X\}$ is the number of observations belonging to class $i$ from the tricluster $t$, and $|t^X|$ is the number of observations of the tricluster $t$. The DS is computed for

each class within each tricluster. The higher the value for a given class the more representative is the tricluster. The DS of the majority class also serve as the weight of the tricluster. This weight is subsequently utilized in the classification phase, where it influences the determination of how strongly a tricluster's pattern supports a given class during the prediction process.

From the resulting triclusters, we compute DS for each class within each tricluster. The initial parameter of our model is the DS threshold, which is set to a default value of 0.6, allowing us to filter out triclusters that do not exhibit a discriminative pattern. Subsequently, we filter the triclusters, selecting only the ones where the DS of the majority class equals or exceeds the DS threshold. An example of this process is showed in Figure (3.3) using the tricluster from the Table 3.2.

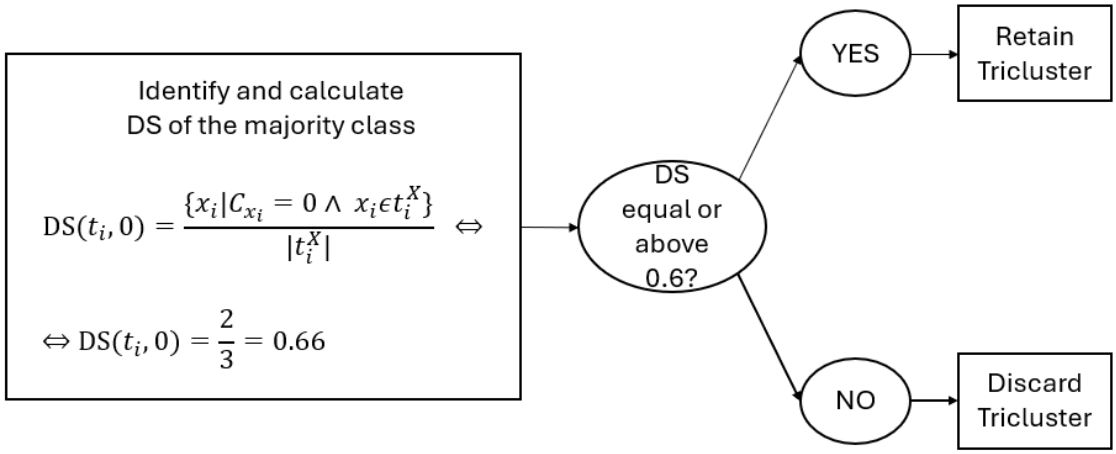


Figure 3.3: Example using the tricluster from the Table 3.2 of determining whether the tricluster is discarded or retained based on the DS value of the majority class in relation to the DS threshold.

### 3.1.4 Transaction Set and Rules

This section outlines the process of converting triclusters into transactions, defining relevant rules, and ensuring their applicability across different triclusters. Each tricluster is represented as a transaction consisting of its columns and the majority class. The relevance of generated rules is determined based on their coverage across multiple triclusters and the diversity of columns they contribute. This step was added by the need for dimensionality reduction, allowing for a more efficient rule extraction process. Instead of using one tricluster as one rule, a reduced number of rules is obtained using the FP-Growth algorithm on the transaction set, ensuring each rule offers unique insights, and aiding in the comprehensive analysis of patterns within the dataset. FP-Growth is a pattern mining algorithm commonly used for mining frequent patterns in large datasets. The algorithm uses this data structure to generate frequent itemsets without explicitly generating all subsets of the data, making it particularly effective for large and sparse datasets.

**Definition 4.1.** Each tricluster has its representation as a transaction. A transaction is represented by the following elements:

- The columns forming the respective tricluster.
- The majority class of that tricluster, denoted as $c_{\text{major}}$. It indicates the class that has the highest frequency among the observations of that tricluster. It will associate the transaction to that class.

To derive rules as combinations of columns, we first create a transaction set, *T*. This transaction set is subsequently divided into ten subsets, and the FP-Growth algorithm is applied to each subset individually. Detailed explanations of these processes are provided below, starting with the definition of what constitutes a relevant rule, thereby justifying the necessity of oversampling *T*.

In pattern mining, a rule is an expression of the form X $\rightarrow$ Y, where X and Y are sets of items The rule indicates that the occurrence of X in a dataset increases the likelihood of Y. In that context of this work, X represents the columns forming the corresponding tricluster, while Y denotes the $c_{\text{major}}$. A generated rule is considered relevant if it meets the following criteria:

- **Coverage Across Multiple Triclusters**: A rule is relevant if the columns in its antecedent appear in multiple triclusters whose majority class matches the rule's consequent. This implies that the columns in the antecedent allow for the extraction of numerous observations from different triclusters. For instance, given the rule "Columns $y_0z_0; y_1z_0; y_1z_0; y_1z_1 \rightarrow$ Class 0", we examine all triclusters where the majority class is 0 and verify if these columns are present. The more triclusters that satisfy these conditions, the more relevant the rule.
- **Diversity of Columns**: The rule must add a set of columns that is significantly different from those in all previously generated rules. For example, if the rule "Columns $y_0z_0; y_1z_0; y_1z_0; y_1z_1 \rightarrow$ Class 0" already exists, then rules such as "Columns $y_0z_0; y_1z_0 \rightarrow$ Class 0" or "Columns $y_1z_0; y_1z_1 \rightarrow$ Class 0" are not deemed relevant, as they represent subsets of an already established rule.

In summary, a rule is considered relevant if it enables the identification of a substantial number of observations from various triclusters, while also providing a distinct set of columns from the other generated rules. These criteria ensure that each rule adds unique and significant value to the analysis, thereby contributing to a more comprehensive understanding of the patterns within the dataset.

In order to obtain more relevant rules, we perform oversampling in *T*. This oversampling targets transactions associated with triclusters exhibiting a high DS of the $c_{\text{major}}$ and encompassing a larger number of observations, as they are deemed more significant. To achieve this, we employ the following methodology:

- **Step 1:** We categorize the DS into $s$ distributions, where $s$ is set by default to 5. Thus, $l \in \{1, 2, 3, 4, 5\}$. The minimum distribution starts at $\frac{1}{\sum_0^r c_i}$, and the size of each distribution is calculated as $\frac{1-\frac{1}{\sum_0^r c_i}}{s}$, where $r$ is the number of different classes. For instance, with two classes, the distributions would be ([0.5,0.6[, [0.6,0.7[, [0.7,0.8], [0.8,0.9[, [0.9,1.0]). We

then associate each transaction with the respective distribution according to the DS of its $c_{\text{major}}$.

- **Step 2:** If the number of observations in the tricluster associated with the transaction exceeds 5, we use the following expression:

$$\begin{cases} ad = 2l - 1 & \text{if } l \neq 1 \\ ad = 1 & \text{if } l = 1 \end{cases} \tag{3.2}$$

  This expression determines the number of times the transaction will be replicated. For the respective five distributions, the number of repetitions would be: 1-3-5-7-9, which is the repetition list (rp). If the number of observations is less than 5, we proceed to the next step.

- **Step 3:** If the number of observations in the tricluster associated with the transaction is 5 or fewer, the transaction is replicated according to the number of observations as follows:

  – For 3 observations or fewer:

  - If the DS is within the first three distributions, the transaction is replicated according to the 1st element of the repetition list.

  - If the DS is above the third distribution, the transaction is replicated according to the 2nd element of the repetition list.

  - If there are fewer than three distributions available, only the 1st element of the repetition list is used.

  – For 4 or 5 observations:

  - If the transaction is in the 1st distribution, it is replicated according to the 1st element of the repetition list.

  - If the transaction is in the 2nd or 3rd distribution, it is replicated according to the 2nd element of the repetition list.

  - If the transaction is above the 3rd distribution, it is replicated according to the 3rd element of the repetition list.

  - If there are not enough distributions, the values from the immediately preceding observations are used until available.

A summary of the steps above is showed in Figure (3.4).

Taking as an example the tricluster from Table (3.3), according to the steps showed in Figure (3.4), the transaction from that tricluster would be replicated 3 times, because it has 3 elements and its DS is 1.0 (all the observations belong to the same class).

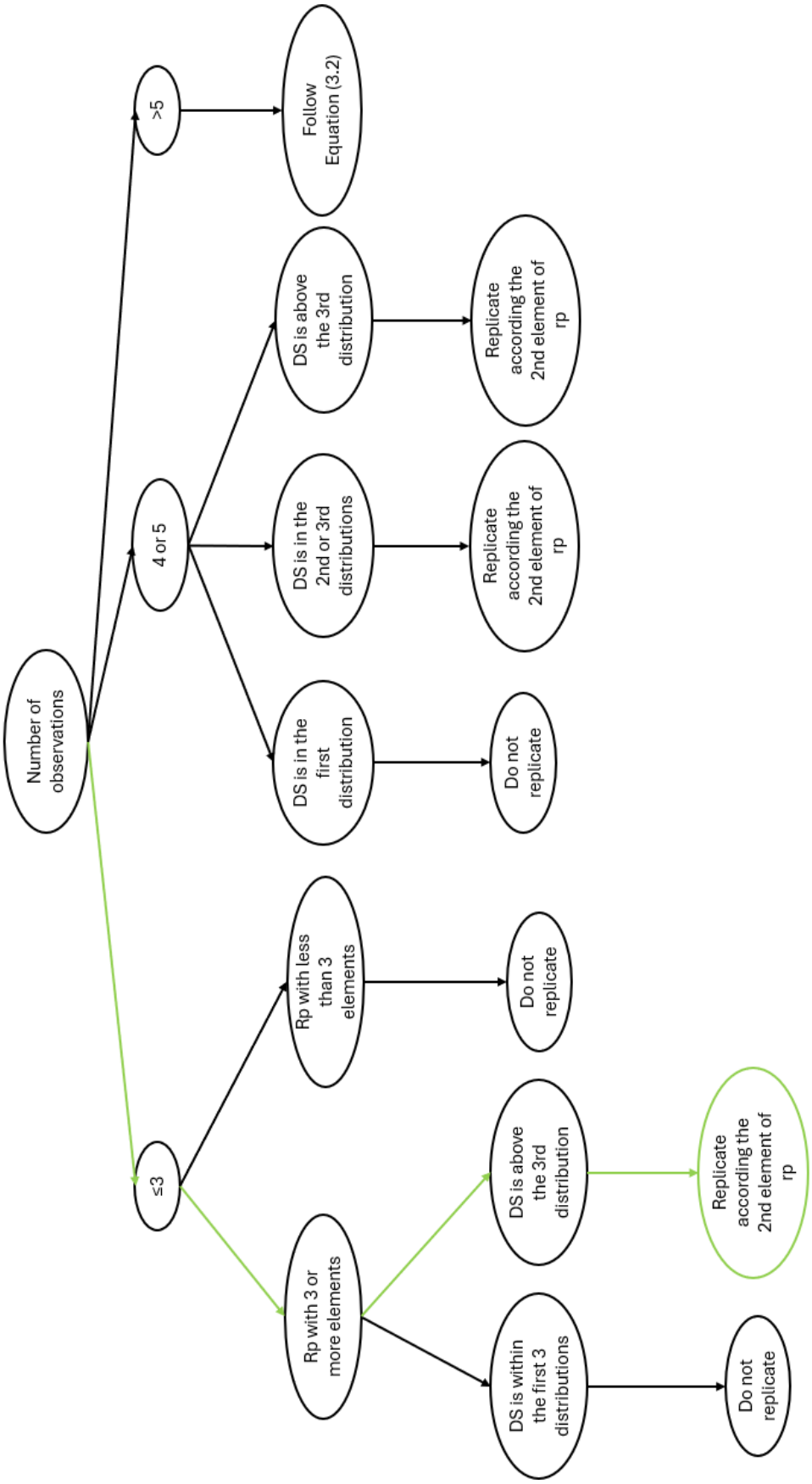


Figure 3.4: Summary of the steps to follow to perform the oversampling of the transactions. The green path highlights the process leading to the decision that the transaction from the tricluster in Table (3.3) will be replicated 3 times.

For the transaction set $T$, we segregate the classes. For each transaction corresponding to a specific class, we generate a maximum of $v$ rules, by default $v$ is set to 75, for a specified support value, defined by the user (by default the support value is set to 0.2), using the FP-Growth algorithm. Setting $v$ to 75 is a secure value because it does not limit the model and offers a good trade-off between time spent and resource consumption.

These rules are then segregated across classes.

The generated rules must adhere to the following criteria: the consequent must exclusively consist of the class label, the antecedent must include at least three columns, and the Jaccard similarity (refer to Equation (3.3)) among all antecedent groups must be below 30% to satisfy the **Diversity of Columns** criterion for a relevant rule. A set of rules, denoted by $R$, is derived. A synthesized schema of this process can be seen on Figure (3.5).

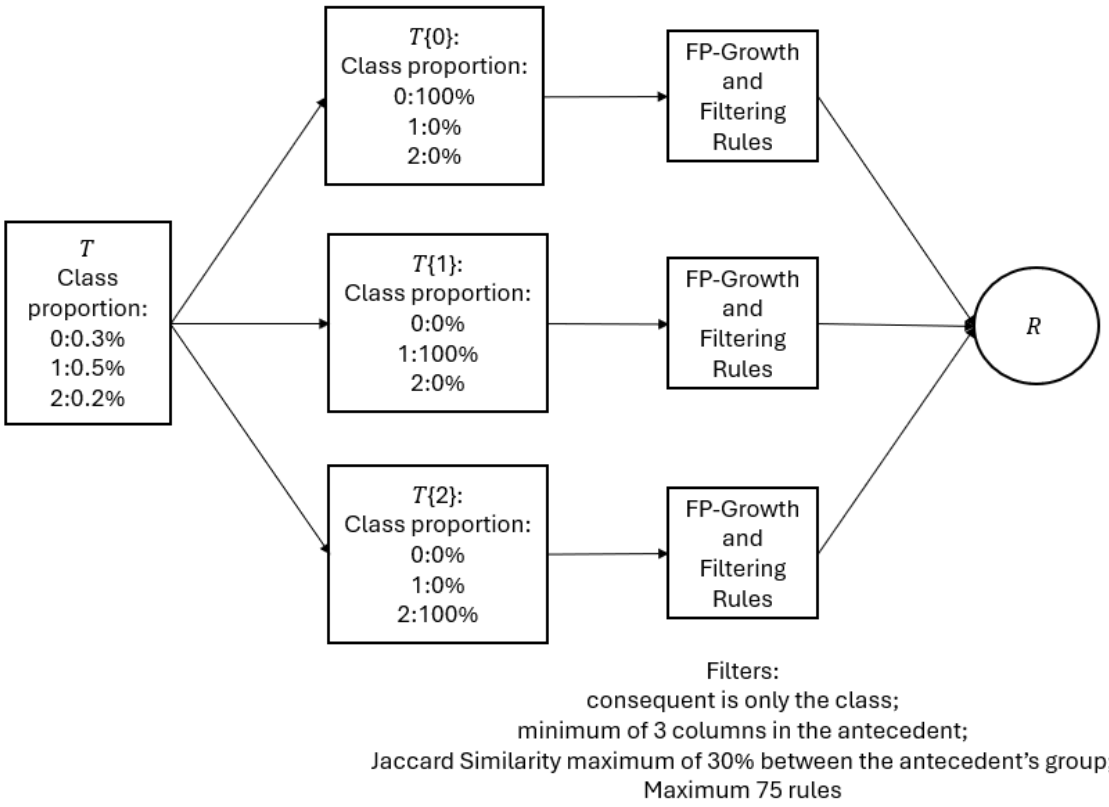


Figure 3.5: Summary of the steps to follow to obtain the set of rules, $R$, from a transaction set $T$.

Within $R$, there exists $k$ subsets, each represented as $\text{rules}_k$, where $k$ indicates the class characterized by the set of rules. Each rule is labeled as $\text{rule}_{ik}$, with $i$ representing the $i$-th rule within the set corresponding to the class combination, $k$.

### 3.1.5 Training

We begin the training phase by categorizing each rule in the set $\text{rules}_{ik}$ as Numeric (N), Categorical (C), or Mixed (M). Numeric rules exclusively contain numeric columns in the antecedent, Categorical rules contain only categorical columns, and Mixed rules incorporate both types of columns in

the antecedent.

For each Mixed rule, we separate its numeric columns from its categorical columns. We further classify Mixed rules into two types: Mixed Numeric (MN) rules and Mixed Categorical (MC) rules. This separation is necessary because we will use one method to handle Numeric and Mixed Numeric rules, and a different method for Categorical and Mixed Categorical rules. Ultimately, the information obtained from the methods applied to the MC and MN rules will be consolidated, and the respective groups of MC and MN will be recombined to form the original Mixed rule.

In the training phase, each rule is associated with all the triclusters having the $c_{\text{major}}$ equal to the rule's class and containing all columns present in the antecedent. Then, we remove all the columns that are not present in the antecedent and remove all the observations that does not belong to the same class as the $c_{\text{major}}$ of each selected tricluster. Figure (3.6) shows this procedure.

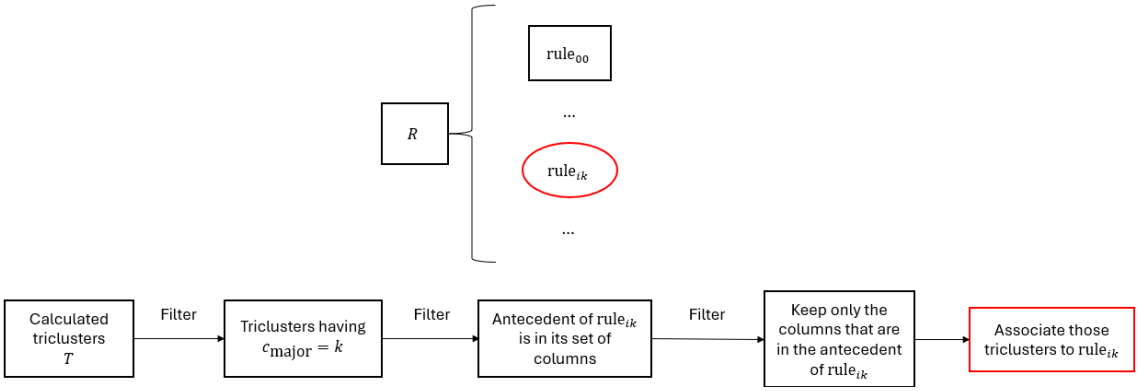


Figure 3.6: Filtering steps in order to associate a set of triclusters to a $\text{rule}_{ik}$.

### Categorical and Mixed Categorical Rules Management

We begin by handling rules from the categorical (C) and Mixed Categorical (MC) categories. Considering only those types of rules, for each $\text{rule}_i$ it is verified the triclusters' coherence across observations. It is done by computing the average Jaccard Similarity (JS) (Equation (3.3)) between each observation $x_i$ and the other observations, for each tricluster.

$$JS(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.3}$$

Here, $A$ and $B$ represent the values of the respective columns between two observations.

Upon completing this process for each tricluster, we select the average with the lowest value as the minimum tolerance JS for that tricluster. This process is explained in Figure (3.7). Another parameter of the model is the minimum tolerance JS threshold. If the minimum tolerance JS falls below this threshold, we discard the tricluster. The default parameter for this threshold is 0.1, to avoid keeping triclusters with a minimum tolerance JS of 0.

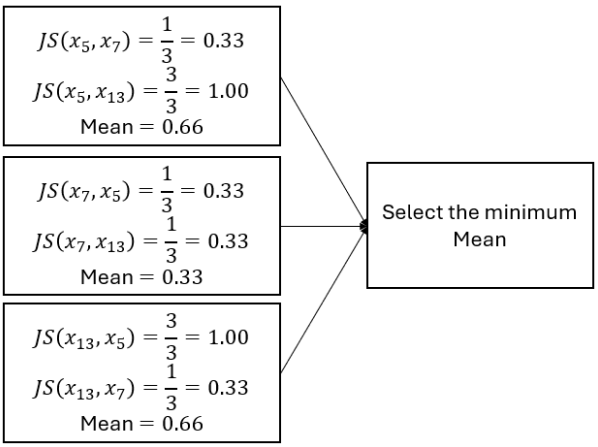


Figure 3.7: Process of calculating the Jaccard Similarity (Equation (3.3)) using as an example the tricluster from Table (3.2) that is associated with a categorical rule$_{ik}$.

### Numeric and Mixed Numeric Rules Management

The objective when using triclusters from the Numeric (N) and Mixed Numeric (MN) rules is to determine the slope from linear regression (LR). Each context provides the time coordinate ($x$), while the feature's value provides the $y$ coordinate. The feature's value is obtained by calculating the mean value of each feature using all the objects presented on the filtered tricluster. Subsequently, we calculate the linear regression to obtain the slope for each feature. Each linear regression yields an $R^2$ (Equation (3.4)) value.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}i)^2}{\sum i = 1^n (y_i - \bar{y})^2} \tag{3.4}$$

In this formula, $y_i$ represents the observed values, $\hat{y}_i$ represents the predicted values, $\bar{y}$ represents the mean of the observed values, and $n$ represents the number of observations.

The tolerance LR is given by the expression:

$$\text{Tolerance LR} = 1 - R^2 + tol \tag{3.5}$$

$tol$ is a parameter of the model defined by the user, set by default to 0.2. To conclude, the tolerance LR is a window that follows this expression:

$$[\text{slope} \pm \text{slope} \times \text{tolerance LR}] \tag{3.6}$$

For instance, if the tolerance LR obtained is 0.1 and the slope obtained is 1.00, then the acceptable range of slopes is between 0.90 and 1.10.

Figure (3.8), shows a toy example explaining the previous steps using the tricluster from Table (3.3).

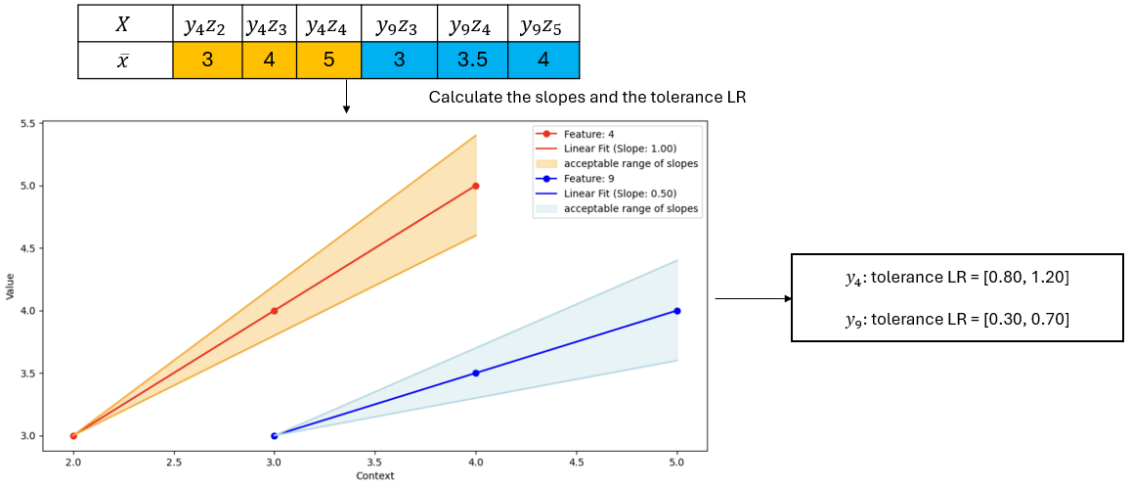


Figure 3.8: Procedure for calculating the slope and tolerance LR using as an example the tricluster from Table (3.3) that is associated with a numerical rule$_{ik}$. $\bar{x}$ represents the mean along the columns. In the graph, the boundaries of the orange and light blue regions are delineated by slopes corresponding to the lower and upper limits of the tolerance LR, respectively, and an intercept value equal to that obtained from the linear fit. This visualization aids in clearly depicting the acceptable range of slopes.

### 3.1.6 Testing

In the testing phase, the trained associative model is used to classify a new observation slice. The test phase begins with the normalization of the test set.

#### Numeric Rules Fit

Given a new observation, we begin by selecting the columns from each tricluster for each rule of categories N and MN (see Figure (3.6)) to perform the linear regressions. Similar to the process described in **Numeric and Mixed Numeric Rules Management**, we calculate the slope for each new observation. If all slopes of the same object obtained fall within the respective windows of tolerance LR, then the rule is respected, and we record the weight of the tricluster used (see Definition 3). However, if at least one slope does not belong within the respective windows of tolerance LR, that tricluster contributes with a weight of 0 (see Figure (3.9) for an example).

It is important to note that each rule may have several different triclusters, resulting in multiple weights for a single rule. The weight assigned to that rule will be the highest weight among the triclusters that the new observation respected.

#### Categorical Rules Fit

Similar to the previous process, we start by selecting the columns from each tricluster for each rule of categories C and MC. For each new observation, we calculate the Jaccard Similarity between the new observation and all other observations within the tricluster. If the average Jaccard Similarity obtained is equal to or greater than the minimum tolerance JS of that tricluster, then we characterize

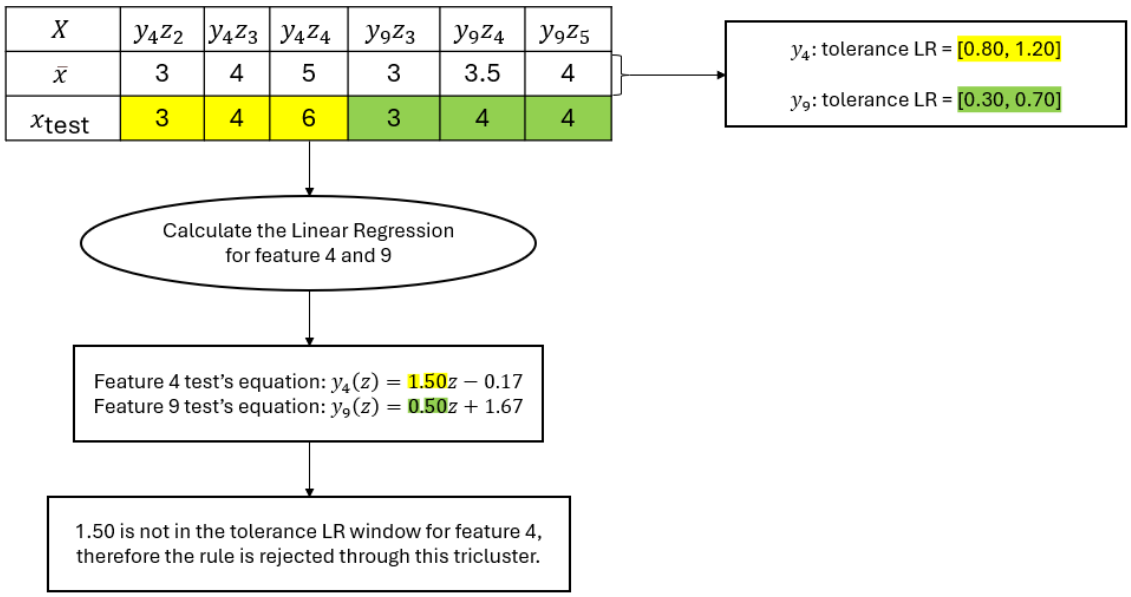


Figure 3.9: Steps followed to characterize a rule as rejected through the tricluster from Table (3.3) associated to rule$_{ik}$ as in Figure (3.8) for a $x_{\text{test}}$.

the rule as respected and record the weight of the tricluster used, otherwise that tricluster contributes with a weight of 0 (see Figure (3.10) for an example). Finally, we associate the rule with the highest weight obtained within that group.

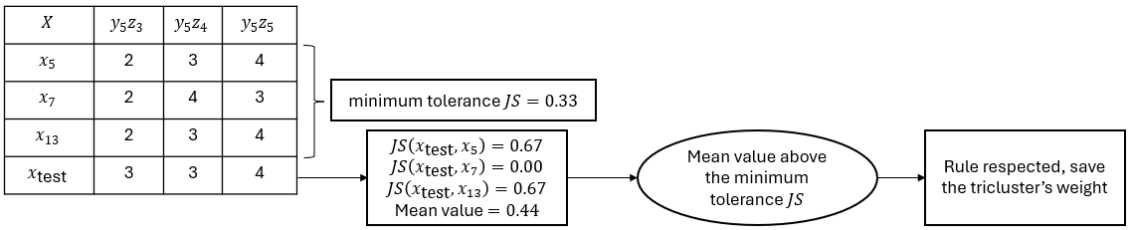


Figure 3.10: Steps followed to characterize a rule as accepted through the tricluster from Table (3.2) associated to rule$_{ik}$ as in Figure (3.7) for a $x_{\text{test}}$.

### Mixed Rules Fit

Combining the procedures outlined in **Numeric Rules Fit** and **Categorical Rules Fit** for Mixed Numeric rules and Mixed Categorical rules, respectively, we characterize the rules as respected only if both MN and MC rules from the same tricluster are respected. If accepted, we then save the weight of the tricluster used, otherwise that tricluster contributes with a weight of 0. Finally, we record the highest weight obtained from that group.

### Classification

Each new observation is assigned with a value ranging from 0 to 1 for every rule. The final parameter, automatically determined, is referred to as the weight threshold, and each class has

its own weight threshold. This threshold sets all weights below its value to zero. Following the application of the weight threshold, the values associated with each rule are aggregated for every class and observation. Since the number of rules varies among classes, classes with a greater number of rules are more likely to accumulate higher sums. To mitigate this discrepancy, the aggregated values are normalized for each class. Classification involves assigning the new observation to the class that obtained the highest normalized sum (see Table 3.4).

We then select a metric to evaluate the trained model. Each weight threshold is tested across all combinations from 0 to 1 in increments of 0.05. This process is repeated for each combination of weight thresholds, and the one yielding the highest value for the chosen metric is selected.

Table 3.4: Example of the pre-classification process. $\Sigma\text{class}_i$ represents the sum from all weights of $\text{class}_i$. The sum of the weights for each class already were affected by the weight threshold. The columns Norm $\text{class}_i$ are the normalization values from the respective column $\Sigma\text{class}_i$. For this example, it was considered having 5 rules associated with class 0, 10 rules with class 1, and 7 rules with class 2.

| $X_{\text{test}}$ | $\Sigma$ class 0 | $\Sigma$ class 1 | $\Sigma$ class 2 | Norm class 0 | Norm class 1 | Norm class 2 | Prediction |
|---|---|---|---|---|---|---|---|
| $x_{\text{test}_1}$ | 3.2 | 4.0 | 0.0 | 0.40 | 0.26 | 0.00 | 0 |
| $x_{\text{test}_2}$ | 2.1 | 4.1 | 2.5 | 0.26 | 0.27 | 0.28 | 2 |
| $x_{\text{test}_3}$ | 2.8 | 6.8 | 6.3 | 0.34 | 0.46 | 0.72 | 2 |

### 3.1.7 Advantages and Limitations

#### Advantages

This novel associative classifier offers several significant advantages. It employs a triclustering algorithm to identify subgroups of observations, features, and contexts that exhibit similar patterns, which is essential for predicting disease progression. Utilizing these patterns, the classifier emphasizes important features during the classification process, guiding researchers towards pertinent features for further investigation.

The classifier incorporates the FP-Growth technique for rule generation, providing straightforward and interpretable rules. These rules not only aid in explaining the model's predictions but also serve as valuable insights for feature engineering.

Unlike many other classifiers, such as Support Vector Machines and Multilayer Perceptron, this approach does not require *OneHotEncoding* to process categorical features. It efficiently handles categorical data through the calculation of Jaccard similarity, thereby facilitating the processing of mixed datasets and significantly reducing the time required for data preparation.

The classifier's modular architecture permits potential scalability, as each component can be independently optimized. Furthermore, although extensive hyperparameter tuning can be challenging, it offers a high degree of customizability, allowing researchers to adapt the model to specific dataset characteristics.

#### Limitations

Despite its innovative approach, the associative classifier presents some limitations. Firstly, the algorithm is inherently complex, integrating multiple techniques such as triclustering, rule generation, similarity calculations, and linear regressions. Managing and combining these various components to produce accurate predictions is a challenging task.

The model's flexibility is both an advantage and a drawback. It includes numerous hyperparameters, making it difficult to explore and optimize the full range of possible values. This extensive tuning process can be time-consuming and computationally intensive.

Additionally, the classifier may exhibit longer training and testing times depending on the dataset being used. This extended processing duration could present a consideration in contexts where timely results are essential.

The performance of the classifier heavily depends on the output of the triclustering algorithm. Identifying patterns that effectively distinguish between objects and classes is critical for the model's success. If the triclusters do not capture meaningful patterns, the classifier's performance can degrade.

The algorithm is computationally intensive, with effective performance dependent on generating a diverse and substantial number of rules. A lack of variety in the rule set may adversely affect the model's predictive capability.

Finally, this classifier is constrained by its reliance on linear patterns in the numerical columns, as the use of linear regression inherently limits its ability to accurately capture non-linear patterns.

### 3.1.8 Code Availability

Available at https://github.com/AlexSilvaM/Associative_Classifier, the proposed associative classifier was coded in Python.

## 3.2 Results

This section presents the evaluation outcomes of the proposed associative classifier, focusing on its performance across various datasets with distinct patterns on the context dimension. We assessed the classifier's accuracy, specificity, and sensitivity, comparing it against Triclustering-Based models using Random Forest (RF) and XGBoost [3]. The findings provide a comprehensive overview of how the classifier handles different temporal data patterns.

### 3.2.1 TCTriCluster Algorithm

In this study, the TCTriCluster algorithm [3] was selected as the triclustering method to be integrated within our associative classifier. The TCTriCluster algorithm is an evolution of the foundational triCluster algorithm introduced by Zhao and Zaki [19]. While the original triCluster algorithm effectively identified coherent clusters within three-dimensional gene expression data, encompassing dimensions such as gene, sample, and time, it did not impose temporal continuity, which is a critical

feature for analyzing time-series data, particularly in clinical studies. To address this limitation, the TCTriCluster algorithm incorporates a temporal constraint, ensuring that the identified triclusters maintain contiguity over time. This temporal coherence enhances the interpretability of the discovered patterns, aligning them with the natural progression of clinical events [3].

The TCTriCluster algorithm is distinguished by its quasi-exhaustive approach, which facilitates the discovery of overlapping triclusters that exhibit constant, scaling, and shifting patterns, which is essential for capturing the complex nature of temporal data [3]. Although initially designed for gene expression data, the TCTriCluster algorithm has been adapted in this study to address the complexities inherent in clinical datasets. This adaptation involves the preprocessing of clinical data to mirror the structure of gene-sample-time data, thereby enabling the effective application of TCTriCluster to object-feature-time data [3].

The TCTriCluster algorithm operates through a systematic three-step process [3]:

1. **Construction of Multigraphs:** The process begins with the construction of a multigraph based on the similarity of data points across different objects. This multigraph captures the relationships between data points, accounting for both temporal and spatial coherence.

2. **Mining of Maximal Biclusters:** At each time point, maximal biclusters are extracted from the multigraph. These biclusters represent significant patterns within a two-dimensional slice of the three-way data, encapsulating the relationships between features and objects at specific time intervals.

3. **Extraction and Refinement of Triclusters:** Finally, triclusters are derived by merging similar biclusters across different time points. The algorithm includes a refinement step that allows for the removal or merging of overlapping triclusters based on predefined criteria, ensuring that the identified patterns are statistically significant and relevant [3].

The flexibility of the TCTriCluster algorithm is enhanced by several tunable parameters, which were carefully adjusted in this study to optimize performance [3]:

- *sT MINT*: Defines the minimum size for the **Context** (z) dimension in each generated tricluster.

- *sS MINS*: Specifies the minimum size for the **Features** (y) dimension.

- *sG MING*: Establishes the minimum size for the **Object** (x) dimension.

- *w WINSZ*: Represents the size of the subspaces considered for tricluster formation. Larger values increase the subspace size, leading to the identification of more extensive triclusters.

These parameters are critical as they directly influence the characteristics and number of triclusters calculated, which, in turn, impacts the overall performance of the classifier [3].

Upon executing the TCTriCluster algorithm, a set of triclusters is calculated, which serves as the triclustering solution. These triclusters, representing coherent patterns across the three dimensions,

are integral to the subsequent classification process and form the basis for the model's predictive capabilities [3].

The successful application of the TCTriCluster algorithm in this study is demonstrated by its ability to identify clinically relevant patterns in the progression of Amyotrophic Lateral Sclerosis (ALS) patients [3]. By tailoring the algorithm's parameters to the specific challenges posed by longitudinal clinical data, TCTriCluster has proven effective in uncovering temporal patterns that are not only statistically significant but also clinically interpretable. These insights have provided valuable contributions to understanding disease progression, thereby supporting the development of predictive models that can facilitate timely interventions and personalized treatment strategies for ALS patients [3].

### 3.2.2 Synthetic Data Generation

In the initial phase of this research, synthetic datasets were generated using the G-HTric generator [59], which played a pivotal role in the early development and validation of our associative classifier. G-HTric represents a significant advancement over its predecessor, G-Tric [60], offering enhanced capabilities in synthetic data generation specifically tailored for triclustering algorithms. This generator addresses key limitations associated with handling heterogeneous datasets by supporting mixed-type triclustering solutions and providing robust data annotations [59].

The G-HTric generator, developed by Soares *et al.*[59], excels in creating complex synthetic datasets that simulate the challenges encountered in real-world triclustering applications. It not only extends the functionality of the original G-Tric by accommodating heterogeneous datasets but also allows for the inclusion of ground truth data in the annotation of triclustering solutions, which is essential for validating the performance of triclustering algorithms in practical scenarios[59].

G-HTric offers extensive customization options, enabling users to tailor the generated datasets to their specific research needs. Users can define the dataset size by specifying the number of observations ($|X|$), variables ($|Y|$), and contexts ($|Z|$). For datasets requiring heterogeneity, users can specify the proportion of symbolic ($|Ysym|$) and numeric variables ($|Ynum|$), thereby accommodating the complexities inherent in real-world clinical data [59].

A notable feature of G-HTric is its ability to simulate a variety of patterns within the generated datasets, including constant, additive, multiplicative, order-preserving, and none. These patterns can be introduced across multiple dimensions ($|X| \times |Y| \times |Z|$), providing a comprehensive framework for evaluating the robustness of triclustering methods [59].

Additionally, the generator allows for precise control over the class distribution within the dataset, including the number of target classes and their prevalence. This feature is particularly valuable for generating datasets that reflect the imbalanced nature of many real-world problems [59].

The G-HTric generator provides a range of background distribution options, such as Uniform, Normal, Discrete, and Missing, which can be applied to different variables within the dataset. This flexibility ensures that the generated data can mimic a wide array of real-world conditions, from well-structured environments to those with significant noise and missing values [59].

Furthermore, G-HTric's overlapping capabilities allow for the creation of intricate data scenarios where triclusters can share elements, thereby testing the algorithm's capacity to discern complex patterns. Parameters such as Plaid Coherence and the extent of overlap among triclusters provide granular control over these scenarios, making G-HTric an invaluable tool for synthetic data generation [59].

As the research progresses, more complex datasets will be generated using various configurations of the G-HTric generator. These will include datasets with heterogeneous variables and multiple classes to further challenge the classifier. Additionally, the introduction of missing values, noise, and errors will allow for a comprehensive evaluation of the classifier's robustness under more realistic conditions.

### 3.2.3 Experiments

In this section, we present a series of experiments conducted to evaluate the performance of the proposed associative classifier. The performance was assessed using three key metrics: accuracy (Equation (2.3)), specificity (Equation (2.5)), and sensitivity (Equation (2.4)). Across all experiments, the associative classifier's performance was compared with two Triclustering-Based models: one using Random Forest (RF) and the other using XGBoost, as developed by Soares et al. [3]. All results were obtained through 5x5 fold cross-validation.

For each experiment, we generated four datasets using identical parameters, focusing on various types of patterns present in the context dimension. The G-HTric generator was utilized to create datasets containing constant patterns, additive patterns, multiplicative patterns, and order-preserving patterns. Additionally, experiments were conducted to analyze the classifier's performance on datasets with a reduced number of triclusters, avoiding the application of FP-Growth to assess its impact on the classifier's performance in scenarios with fewer triclusters. A final set of experiments was conducted using datasets specifically created for this study, incorporating constant, additive, multiplicative, and mixed patterns.

#### Datasets Generated Using G-HTric

The datasets used in these experiments were generated using the G-HTric generator [59], with the following parameters consistent across all datasets:

- *Number of Objects*: 1000;
- *Number of Features*: 10;
- *Number of Contexts*: 5;
- *Number of Classes*: 2, with 50% of elements in each class;
- *Dataset type*: Real numeric values between 0 and 10 with 2 decimal places;
- *Number of triclusters planted*: 5;

- *Triclusters' object structure*: Minimum 100 objects, maximum 200 objects;
- *Triclusters' feature structure*: Minimum 2 features, maximum 10 features;
- *Triclusters' context structure*: Minimum 2 contexts, maximum 5 contexts;
- *Triclusters' contiguity*: Applied to contexts;
- *Plaid Coherency*: No overlapping;
- *Missing elements*: 0%;
- *Noisy elements*: 0%;
- *Errors*: 0%.

Table 3.5 presents the tricluster patterns selected from the generator to produce the datasets used in this study. It is important to note that due to a limitation of the generator, it was not possible to select the object and feature patterns as *None* to generate only additive and multiplicative patterns in the context dimension.

Table 3.5: Tricluster patterns selected for generating datasets with each type of pattern.

| **Pattern type** | **Object Pattern** | **Feature Pattern** | **Context Pattern** |
|---|---|---|---|
| Constant | None | None | Constant |
| Additive | Constant | Constant | Additive |
| Multiplicative | Constant | Constant | Multiplicative |
| Order preserving | None | None | Order preserving |

For detailed information on the number of objects, features, and contexts for each tricluster in the datasets, please refer to Table 1 for constant patterns, Table 2 for additive patterns, Table 3 for multiplicative patterns, and Table 4 for order-preserving patterns, all in the context dimension, in the Appendix.

The results were obtained using the default parameters for both the Triclustering-Based classifiers and our associative classifier. However, an exception was made for the hyperparameter *maximum of v rules* derived from the FP-Growth algorithm from the associative classifier. This particular hyperparameter was selected through hyperparameter tuning, as it has a significant impact on the classifier's performance. This adjustment was crucial, as the optimal value for *maximum of v rules* can vary significantly depending on the specific dataset, and therefore, it was carefully tuned to enhance the classifier's effectiveness.

Initially, four datasets with constant patterns planted in the context dimension were generated. The performance results for the associative classifier, the Triclustering-Based RF classifier, and the Triclustering-Based XGBoost classifier are summarized in Table 3.6. The mean values across all datasets are depicted in Figure 3.11, with exact values and their respective standard deviations available in Table 5 in the Appendix.

Table 3.6: Results with corresponding standard deviations for each classifier using datasets generated with constant patterns in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| Constant 0 | Accuracy | 0.583 ± 0.022 | 0.639 ± 0.019 | 0.601 ± 0.061 |
| | Specificity | 0.600 ± 0.028 | 0.622 ± 0.050 | 0.582 ± 0.055 |
| | Sensitivity | 0.551 ± 0.021 | 0.656 ± 0.048 | 0.610 ± 0.063 |
| Constant 1 | Accuracy | 0.561 ± 0.024 | 0.653 ± 0.033 | 0.633 ± 0.008 |
| | Specificity | 0.580 ± 0.021 | 0.706 ± 0.029 | 0.642 ± 0.047 |
| | Sensitivity | 0.540 ± 0.020 | 0.600 ± 0.053 | 0.624 ± 0.036 |
| Constant 2 | Accuracy | 0.602 ± 0.068 | 0.597 ± 0.024 | 0.617 ± 0.025 |
| | Specificity | 0.622 ± 0.052 | 0.618 ± 0.063 | 0.602 ± 0.037 |
| | Sensitivity | 0.582 ± 0.058 | 0.578 ± 0.043 | 0.632 ± 0.018 |
| Constant 3 | Accuracy | 0.542 ± 0.012 | 0.557 ± 0.019 | 0.588 ± 0.019 |
| | Specificity | 0.550 ± 0.032 | 0.572 ± 0.027 | 0.594 ± 0.015 |
| | Sensitivity | 0.538 ± 0.024 | 0.542 ± 0.029 | 0.542 ± 0.037 |

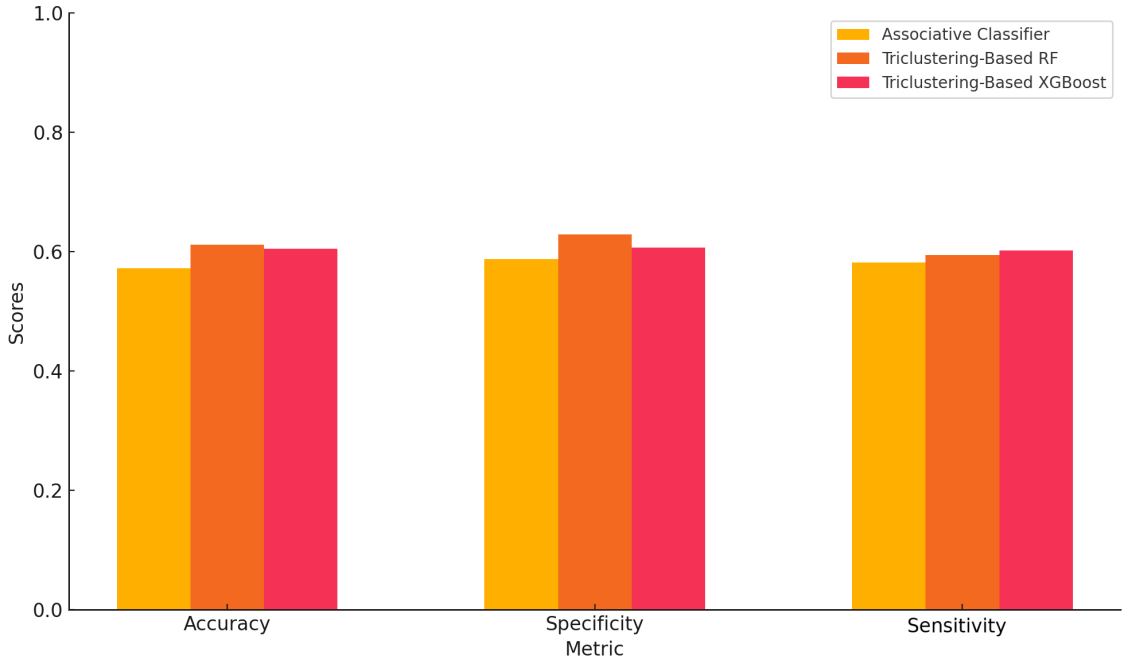


Figure 3.11: Mean values of results obtained for all datasets with constant patterns in the context dimension.

Subsequently, four additional datasets incorporating additive patterns in the context dimension were generated. The performance metrics for the three classifiers are detailed in Table 3.7, with corresponding mean values shown in Figure 3.12. Detailed results, including standard deviations, are provided in Table 6 in the Appendix.

Next, datasets featuring multiplicative patterns in the context dimension were produced and evaluated. The outcomes are presented in Table 3.8, highlighting the comparative performance of the classifiers. Figure 3.13 offers a visual representation of the mean values, with comprehensive data and standard deviations available in Table 7 in the Appendix.

Table 3.7: Results with corresponding standard deviations for each classifier using datasets generated with additive patterns in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| Additive 0 | Accuracy | 0.504 ± 0.009 | 0.674 ± 0.016 | 0.680 ± 0.028 |
| | Specificity | 0.508 ± 0.009 | 0.672 ± 0.040 | 0.680 ± 0.039 |
| | Sensitivity | 0.500 ± 0.009 | 0.676 ± 0.060 | 0.680 ± 0.054 |
| Additive 1 | Accuracy | 0.611 ± 0.018 | 0.540 ± 0.040 | 0.551 ± 0.020 |
| | Specificity | 0.600 ± 0.028 | 0.554 ± 0.024 | 0.558 ± 0.015 |
| | Sensitivity | 0.551 ± 0.021 | 0.526 ± 0.059 | 0.544 ± 0.039 |
| Additive 2 | Accuracy | 0.522 ± 0.009 | 0.540 ± 0.050 | 0.551 ± 0.030 |
| | Specificity | 0.524 ± 0.014 | 0.546 ± 0.063 | 0.554 ± 0.031 |
| | Sensitivity | 0.520 ± 0.012 | 0.534 ± 0.039 | 0.548 ± 0.039 |
| Additive 3 | Accuracy | 0.560 ± 0.017 | 0.770 ± 0.035 | 0.750 ± 0.039 |
| | Specificity | 0.572 ± 0.023 | 0.806 ± 0.039 | 0.780 ± 0.054 |
| | Sensitivity | 0.548 ± 0.029 | 0.734 ± 0.047 | 0.720 ± 0.059 |

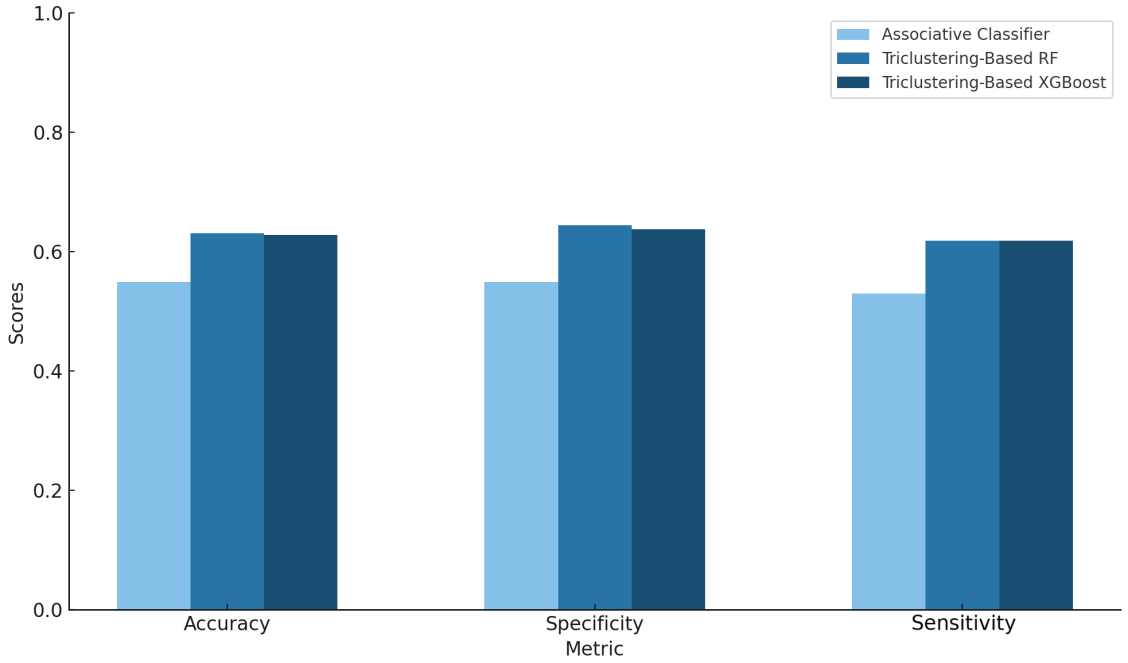


Figure 3.12: Mean values of results obtained for all datasets with additive patterns in the context dimension.

Finally, four datasets were created with order-preserving patterns in the context dimension. The evaluation of these datasets is summarized in Table 3.9, and the mean performance values are illustrated in Figure 3.14. A detailed breakdown of the results and associated standard deviations is provided in Table 8 in the Appendix.

Table 3.10 presents the number of triclusters identified by the TCTriCluster algorithm and the number of rules generated by the associative classifier following the application of the FP-Growth algorithm for each dataset. This table also highlights the datasets selected for comparing the associative classifier with and without the FP-Growth algorithm, where each tricluster is considered

Table 3.8: Results with corresponding standard deviations for each classifier using datasets generated with multiplicative patterns in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| Multiplicative 0 | Accuracy | 0.576 ± 0.013 | 0.635 ± 0.024 | 0.598 ± 0.037 |
| | Specificity | 0.600 ± 0.018 | 0.680 ± 0.039 | 0.604 ± 0.045 |
| | Sensitivity | 0.551 ± 0.015 | 0.590 ± 0.046 | 0.592 ± 0.075 |
| Multiplicative 1 | Accuracy | 0.573 ± 0.012 | 0.541 ± 0.043 | 0.556 ± 0.049 |
| | Specificity | 0.582 ± 0.023 | 0.558 ± 0.041 | 0.540 ± 0.053 |
| | Sensitivity | 0.554 ± 0.021 | 0.524 ± 0.086 | 0.572 ± 0.059 |
| Multiplicative 2 | Accuracy | 0.530 ± 0.010 | 0.529 ± 0.025 | 0.540 ± 0.035 |
| | Specificity | 0.540 ± 0.015 | 0.540 ± 0.029 | 0.530 ± 0.035 |
| | Sensitivity | 0.519 ± 0.012 | 0.518 ± 0.075 | 0.550 ± 0.082 |
| Multiplicative 3 | Accuracy | 0.590 ± 0.040 | 0.733 ± 0.029 | 0.708 ± 0.027 |
| | Specificity | 0.594 ± 0.046 | 0.790 ± 0.055 | 0.728 ± 0.075 |
| | Sensitivity | 0.588 ± 0.038 | 0.686 ± 0.039 | 0.680 ± 0.026 |

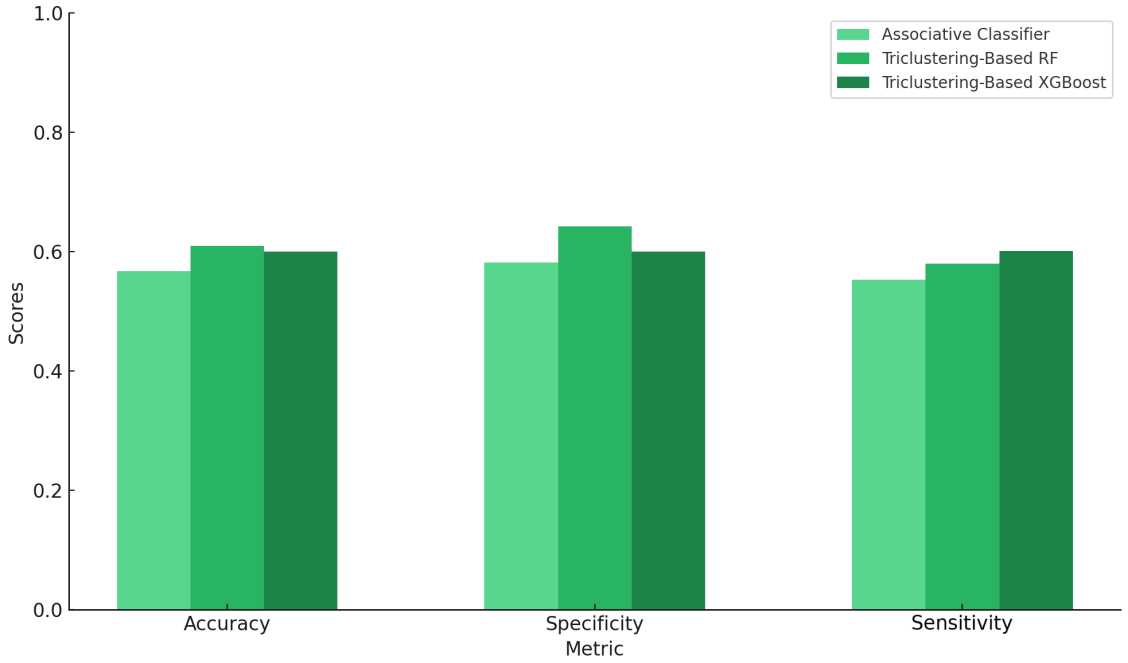


Figure 3.13: Mean values of results obtained for all datasets with multiplicative patterns in the context dimension.

a rule in the absence of FP-Growth. This setup establishes the foundation for the subsequent series of experiments.

### Comparison of the Results Without Applying the FP-Growth Algorithm

The datasets highlighted in Table 3.10 were selected based on the number of triclusters calculated by the TCTriCluster algorithm. In cases where the number of triclusters was minimal, the results obtained by the classifier without applying the FP-Growth algorithm were identical to those produced by the classifier in its standard operation. However, in datasets such as Order Preserving 0 and Order Preserving 2, where the number of triclusters was high, resulting in 22500 and 149 rules,

Table 3.9: Results with corresponding standard deviations for each classifier using datasets generated with order preserving patterns in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| Order Preserving 0 | Accuracy | 0.594 ± 0.023 | 0.688 ± 0.028 | 0.646 ± 0.033 |
| | Specificity | 0.621 ± 0.025 | 0.698 ± 0.031 | 0.612 ± 0.040 |
| | Sensitivity | 0.561 ± 0.020 | 0.678 ± 0.053 | 0.670 ± 0.088 |
| Order Preserving 1 | Accuracy | 0.628 ± 0.037 | 0.563 ± 0.019 | 0.558 ± 0.028 |
| | Specificity | 0.640 ± 0.033 | 0.578 ± 0.033 | 0.566 ± 0.023 |
| | Sensitivity | 0.616 ± 0.040 | 0.548 ± 0.033 | 0.560 ± 0.043 |
| Order Preserving 2 | Accuracy | 0.672 ± 0.019 | 0.568 ± 0.036 | 0.578 ± 0.028 |
| | Specificity | 0.682 ± 0.027 | 0.584 ± 0.056 | 0.572 ± 0.031 |
| | Sensitivity | 0.661 ± 0.021 | 0.552 ± 0.032 | 0.584 ± 0.042 |
| Order Preserving 3 | Accuracy | 0.550 ± 0.021 | 0.726 ± 0.033 | 0.690 ± 0.024 |
| | Specificity | 0.560 ± 0.022 | 0.750 ± 0.074 | 0.678 ± 0.062 |
| | Sensitivity | 0.540 ± 0.020 | 0.702 ± 0.032 | 0.702 ± 0.032 |

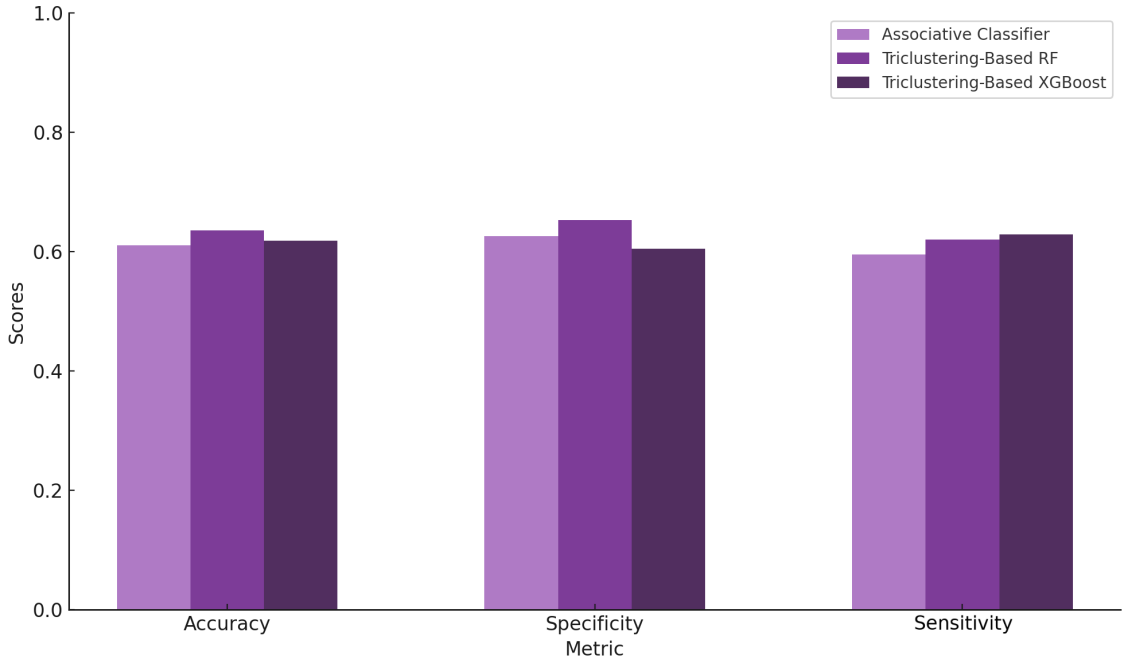


Figure 3.14: Mean values of results obtained for all datasets with order preserving patterns in the context dimension.

respectively, the computational demands exceeded the system's memory capacity. Through the analysis of these datasets, this study aims to evaluate the impact of the FP-Growth algorithm on the performance of the associative classifier, particularly in scenarios where the number of calculated triclusters is low but still exceeds ten.

Table 3.11 presents the performance of the associative classifier operating in its standard mode and without the FP-Growth algorithm on the datasets highlighted in light gray in Table 3.10.

Table 3.10: Number of triclusters calculated by the TCTriCluster algorithm for each dataset and the number of rules used by the associative classifier after applying the FP-Growth algorithm. Datasets highlighted in light gray indicate those where a comparison was made between the associative classifier with and without the FP-Growth algorithm.

| Dataset | Triclusters | Rules |
|---|---|---|
| Constant 0 | 8 | 8 |
| Constant 1 | 4 | 4 |
| Constant 2 | 25 | 20 |
| Constant 3 | 6 | 6 |
| Additive 0 | 2 | 2 |
| Additive 1 | 9 | 9 |
| Additive 2 | 6 | 6 |
| Additive 3 | 47 | 16 |
| Multiplicative 0 | 4 | 4 |
| Multiplicative 1 | 15 | 9 |
| Multiplicative 2 | 6 | 6 |
| Multiplicative 3 | 44 | 16 |
| Order Preserving 0 | 22500 | 60 |
| Order Preserving 1 | 43 | 20 |
| Order Preserving 2 | 149 | 47 |
| Order Preserving 3 | 31 | 26 |

Table 3.11: Comparison of performance metrics, with corresponding standard deviations, for the Associative Classifier using datasets generated with different patterns, with and without the FP-Growth algorithm. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Associative Classifier without FP-Growth |
|---|---|---|---|
| Constant 2 | Accuracy | 0.602 ± 0.068 | 0.720 ± 0.088 |
| | Specificity | 0.622 ± 0.052 | 0.742 ± 0.034 |
| | Sensitivity | 0.582 ± 0.058 | 0.689 ± 0.029 |
| Additive 3 | Accuracy | 0.560 ± 0.017 | 0.593 ± 0.024 |
| | Specificity | 0.572 ± 0.023 | 0.611 ± 0.038 |
| | Sensitivity | 0.548 ± 0.029 | 0.570 ± 0.032 |
| Multiplicative 1 | Accuracy | 0.573 ± 0.012 | 0.650 ± 0.016 |
| | Specificity | 0.592 ± 0.023 | 0.655 ± 0.020 |
| | Sensitivity | 0.554 ± 0.021 | 0.600 ± 0.017 |
| Multiplicative 3 | Accuracy | 0.590 ± 0.040 | 0.655 ± 0.028 |
| | Specificity | 0.594 ± 0.046 | 0.681 ± 0.036 |
| | Sensitivity | 0.588 ± 0.038 | 0.616 ± 0.030 |
| Order Preserving 1 | Accuracy | 0.628 ± 0.037 | 0.588 ± 0.024 |
| | Specificity | 0.640 ± 0.038 | 0.612 ± 0.020 |
| | Sensitivity | 0.616 ± 0.040 | 0.547 ± 0.021 |
| Order Preserving 3 | Accuracy | 0.550 ± 0.021 | 0.584 ± 0.017 |
| | Specificity | 0.560 ± 0.022 | 0.592 ± 0.022 |
| | Sensitivity | 0.540 ± 0.020 | 0.576 ± 0.024 |

### Manually Created Datasets

In this final set of experiments, we address the limitations of the G-HTric generator by manually creating datasets with patterns exclusively present in the context dimension without using the

G-HTric generator. An example of a tricluster that could not be generated by the G-HTric generator is presented in Table (3.12). This tricluster could have been created by selecting an additive pattern type, with the object and feature patterns set to *None*, and the context pattern set to *Additive*.

Table 3.12: Example of a tricluster composed by 2 observations (rows), and by 2 attributes and 3 contexts (columns). All observations belong to the same class. This tricluster could have been generated by G-HTric, with the object and feature patterns set to *None* and the context pattern set to *Additive*, if not for the limitations of the generator.

| $X$ | $y_5z_2$ | $y_5z_3$ | $y_5z_4$ | $y_6z_3$ | $y_6z_4$ | $y_6z_5$ | class |
|---|---|---|---|---|---|---|---|
| $x_{23}$ | 3 | 4 | 5 | 3 | 3.5 | 4 | 2 |
| $x_{31}$ | 4 | 5 | 6 | 4 | 4.5 | 5 | 2 |

This approach rigorously evaluates the associative classifier's performance under these specific conditions. Four datasets were generated for each pattern type: constant, additive, multiplicative, and a combination of all patterns. All patterns were confined to the context dimension.

The manually created datasets share the following parameters:

- *Number of Objects*: 100;
- *Number of Features*: 5;
- *Number of Contexts*: 4;
- *Number of Classes*: 2, with 50% of elements in each class;
- *Dataset type*: Real numeric values between 0 and 10 with 1 decimal place;
- *Number of triclusters planted*: 10;
- *Triclusters' object structure*: 10 objects;
- *Triclusters' feature structure*: Minimum 2 features, maximum 4 features;
- *Triclusters' context structure*: 4 contexts;
- *Triclusters' contiguity*: Applied to contexts;
- *Plaid Coherency*: No overlapping;
- *Missing elements*: 0%;
- *Noisy elements*: 0%;
- *Errors*: 0%.

Each dataset contains 10 triclusters, with each tricluster representing a distinct pattern and incorporating 10 objects across the same features in all contexts. Each object is restricted to follow only one pattern. For all datasets, the cells not part of any tricluster were populated with random

real numbers between 0 and 10. A code was developed to generate datasets for each pattern type. First, we describe the process of generating the data for each pattern type, followed by the results obtained by each classifier in the following order: constant pattern, additive pattern, multiplicative pattern, and a combination of all pattern types.

The initial set of datasets was generated with constant patterns confined to the context dimension. For each feature influenced by a given pattern, a random value was assigned and uniformly maintained across all contexts. The performance metrics for the associative classifier, the Triclustering-Based RF classifier, and the Triclustering-Based XGBoost classifier are presented in Table 3.13. The mean performance values across these datasets are visualized in Figure 3.15, while the specific values along with their corresponding standard deviations are detailed in Table 9 in the Appendix. Additionally, Table 10 in the Appendix provides a comprehensive summary of the features impacted by each pattern within the datasets defined by constant patterns.

Table 3.13: Results with corresponding standard deviations for each classifier using the manually created datasets characterized by the constant pattern in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| M-Constant 0 | Accuracy | 0.680 ± 0.120 | 0.430 ± 0.051 | 0.480 ± 0.117 |
| | Specificity | 0.602 ± 0.088 | 0.420 ± 0.248 | 0.460 ± 0.258 |
| | Sensitivity | 0.718 ± 0.112 | 0.440 ± 0.206 | 0.500 ± 0.063 |
| M-Constant 1 | Accuracy | 0.560 ± 0.110 | 0.520 ± 0.121 | 0.420 ± 0.068 |
| | Specificity | 0.690 ± 0.106 | 0.660 ± 0.150 | 0.540 ± 0.174 |
| | Sensitivity | 0.430 ± 0.118 | 0.380 ± 0.172 | 0.300 ± 0.126 |
| M-Constant 2 | Accuracy | 0.650 ± 0.084 | 0.540 ± 0.159 | 0.590 ± 0.146 |
| | Specificity | 0.550 ± 0.100 | 0.520 ± 0.214 | 0.540 ± 0.258 |
| | Sensitivity | 0.750 ± 0.085 | 0.560 ± 0.150 | 0.640 ± 0.080 |
| M-Constant 3 | Accuracy | 0.810 ± 0.089 | 0.530 ± 0.051 | 0.560 ± 0.037 |
| | Specificity | 0.795 ± 0.078 | 0.520 ± 0.117 | 0.500 ± 0.190 |
| | Sensitivity | 0.825 ± 0.092 | 0.540 ± 0.150 | 0.620 ± 0.160 |

Subsequently, we developed four datasets featuring additive patterns within the context dimension. For each affected feature under a given pattern, an initial random value was assigned to the context set to 0. Then, for each following context, this baseline value was consistently incremented by a constant value across all contexts for the affected features. The performance metrics for the associative classifier, the Triclustering-Based RF classifier, and the Triclustering-Based XGBoost classifier are detailed in Table 3.14. The average performance values across the datasets are illustrated in Figure 3.16, with precise values and their standard deviations documented in Table 11 in the Appendix. Furthermore, Table 12 in the Appendix offers a detailed account of the features impacted by each pattern within the datasets characterized by additive patterns.

Next, four datasets incorporating multiplicative patterns in the context dimension were produced. For each feature influenced by a pattern, an initial random value was assigned in the context set to 0. This value was then multiplied by a predefined constant for each subsequent context, establishing the new value for that context. The performance results for the associative classifier, the Triclustering-Based RF classifier, and the Triclustering-Based XGBoost classifier are summarized in Table 3.15.

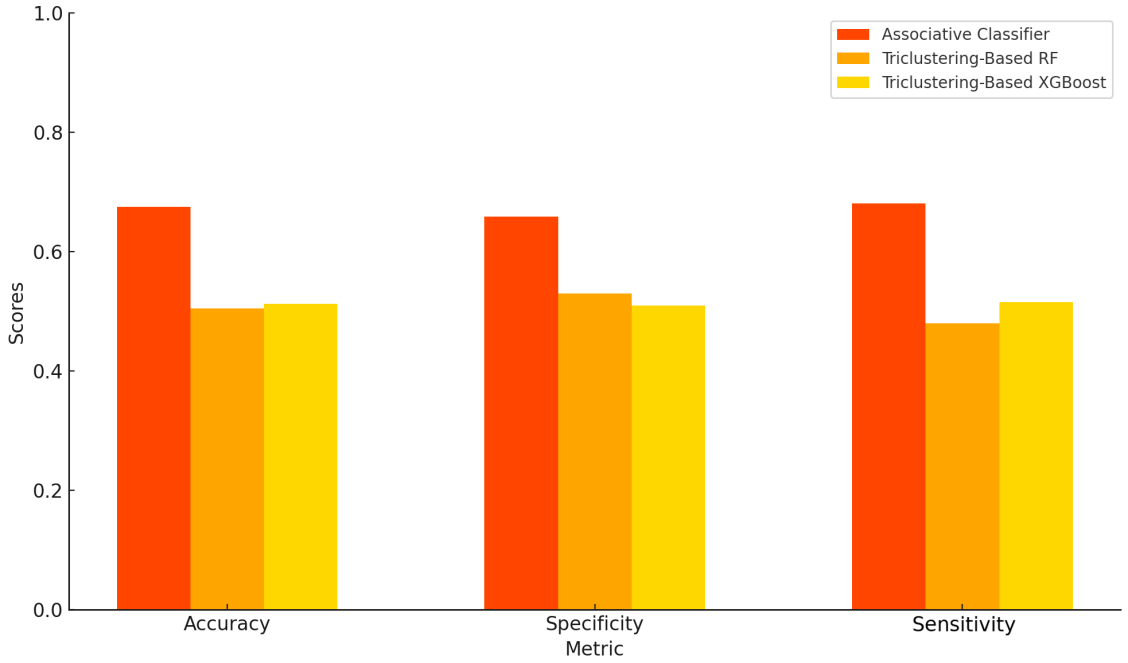


Figure 3.15: Mean values of results obtained for all manually created datasets with constant patterns in the context dimension.

Table 3.14: Results with corresponding standard deviations for each classifier using the manually created datasets characterized by the additive pattern in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| M-Additive 0 | Accuracy | 0.700 ± 0.061 | 0.600 ± 0.063 | 0.560 ± 0.049 |
| | Specificity | 0.720 ± 0.058 | 0.600 ± 0.063 | 0.520 ± 0.117 |
| | Sensitivity | 0.680 ± 0.068 | 0.600 ± 0.141 | 0.600 ± 0.063 |
| M-Additive 1 | Accuracy | 0.720 ± 0.055 | 0.430 ± 0.051 | 0.480 ± 0.117 |
| | Specificity | 0.710 ± 0.046 | 0.420 ± 0.248 | 0.460 ± 0.258 |
| | Sensitivity | 0.730 ± 0.052 | 0.440 ± 0.206 | 0.500 ± 0.063 |
| M-Additive 2 | Accuracy | 0.730 ± 0.071 | 0.630 ± 0.103 | 0.610 ± 0.116 |
| | Specificity | 0.700 ± 0.075 | 0.580 ± 0.194 | 0.600 ± 0.179 |
| | Sensitivity | 0.760 ± 0.074 | 0.680 ± 0.075 | 0.620 ± 0.133 |
| M-Additive 3 | Accuracy | 0.710 ± 0.097 | 0.660 ± 0.058 | 0.680 ± 0.060 |
| | Specificity | 0.740 ± 0.102 | 0.680 ± 0.147 | 0.700 ± 0.179 |
| | Sensitivity | 0.680 ± 0.111 | 0.640 ± 0.185 | 0.660 ± 0.150 |

The average values across all datasets are displayed in Figure 3.17, with specific values and their standard deviations provided in Table 13 in the Appendix. Additionally, Table 14 in the Appendix includes a thorough overview of the features affected by each pattern within the datasets characterized by multiplicative patterns.

Finally, four datasets were created with a combination of all patterns in the context dimension. The performance results for the associative classifier, the Triclustering-Based RF classifier, and the Triclustering-Based XGBoost classifier are summarized in Table 3.16. The mean values across all datasets are depicted in Figure 3.18, with exact values and their respective standard deviations

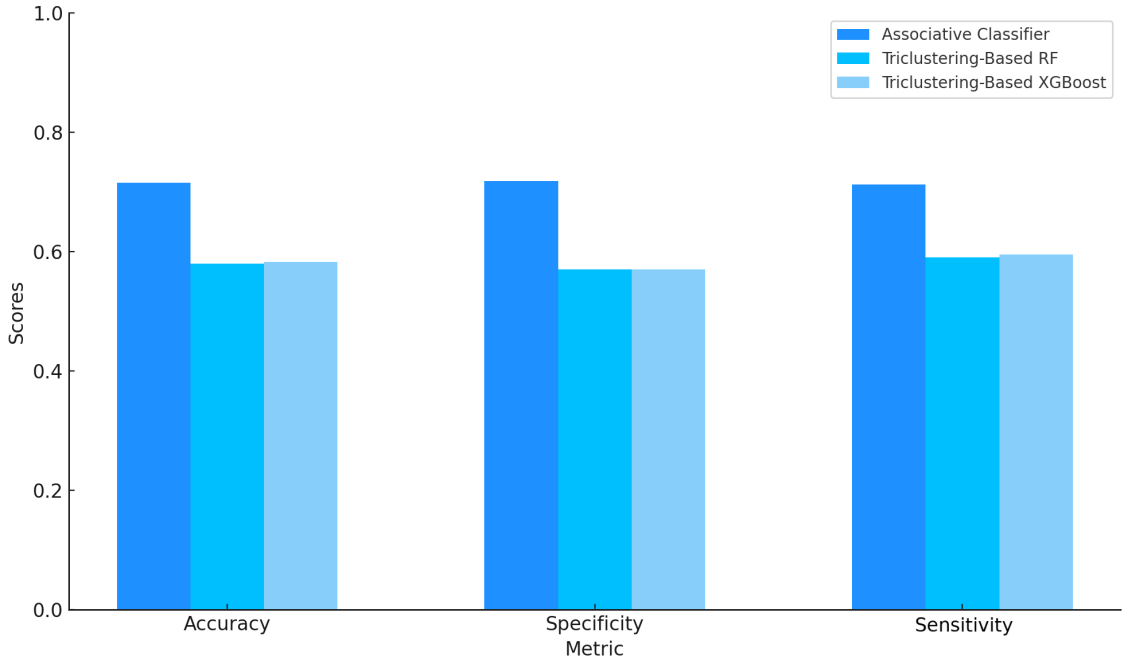


Figure 3.16: Mean values of results obtained for all manually created datasets with additive patterns in the context dimension.

Table 3.15: Results with corresponding standard deviations for each classifier using the manually created datasets characterized by the multiplicative pattern in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| M-Multiplicative 0 | Accuracy | 0.670 ± 0.065 | 0.560 ± 0.086 | 0.570 ± 0.040 |
| | Specificity | 0.600 ± 0.058 | 0.520 ± 0.117 | 0.500 ± 0.126 |
| | Sensitivity | 0.740 ± 0.071 | 0.600 ± 0.200 | 0.640 ± 0.102 |
| M-Multiplicative 1 | Accuracy | 0.710 ± 0.041 | 0.660 ± 0.058 | 0.640 ± 0.102 |
| | Specificity | 0.700 ± 0.032 | 0.640 ± 0.080 | 0.600 ± 0.126 |
| | Sensitivity | 0.720 ± 0.030 | 0.680 ± 0.147 | 0.680 ± 0.204 |
| M-Multiplicative 2 | Accuracy | 0.800 ± 0.112 | 0.680 ± 0.108 | 0.640 ± 0.086 |
| | Specificity | 0.820 ± 0.098 | 0.720 ± 0.183 | 0.680 ± 0.172 |
| | Sensitivity | 0.780 ± 0.102 | 0.640 ± 0.224 | 0.600 ± 0.200 |
| M-Multiplicative 3 | Accuracy | 0.770 ± 0.112 | 0.530 ± 0.040 | 0.540 ± 0.116 |
| | Specificity | 0.760 ± 0.099 | 0.560 ± 0.174 | 0.520 ± 0.075 |
| | Sensitivity | 0.780 ± 0.091 | 0.500 ± 0.141 | 0.560 ± 0.206 |

available in Table 15 in the Appendix. Additionally, Table 16 in the Appendix provides a detailed overview of which features are affected by each pattern within the datasets characterized by mixed patterns.

Table 3.17 highlights the reduction of triclusters into a fewer number of rules for each manually created dataset.

The results obtained across various datasets highlight the associative classifier's capabilities and areas for further improvement. The subsequent section will delve into a detailed discussion of these findings, examining their implications and potential impacts on the field of biomedical data

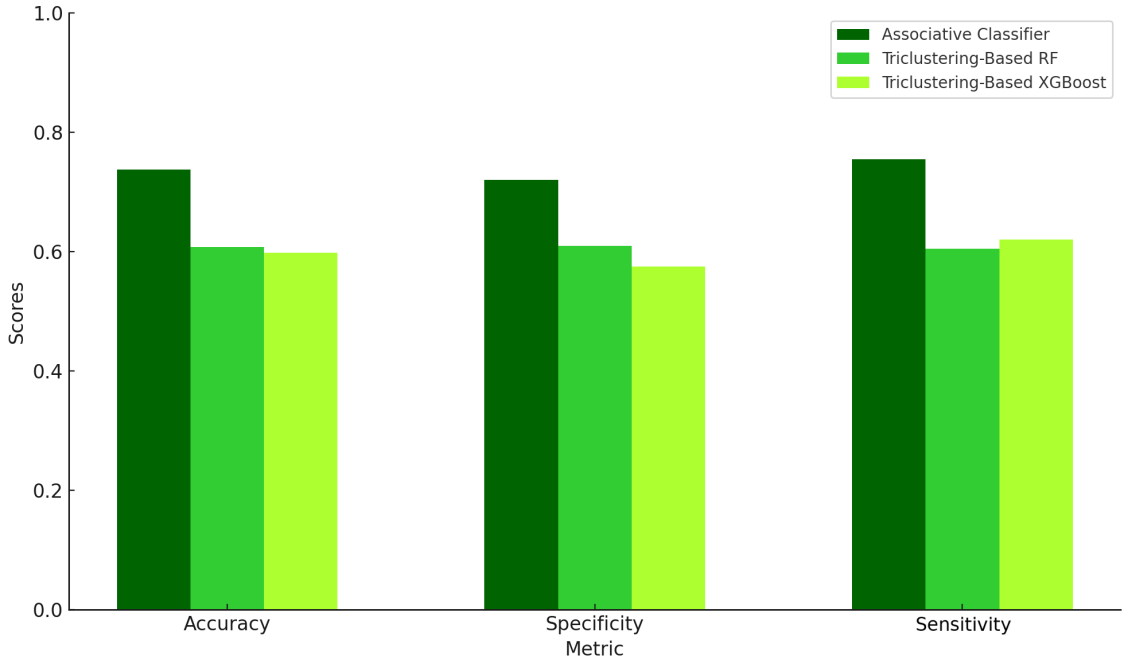


Figure 3.17: Mean values of results obtained for all manually created datasets with multiplicative patterns in the context dimension.

Table 3.16: Results with corresponding standard deviations for each classifier using the manually created datasets characterized by mixed patterns in the context dimension. All tests were conducted using 5x5 fold cross-validation. Cells highlighted in light green indicate the highest results for each classifier.

| Dataset | Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|---|
| M-All 0 | Accuracy | 0.690 ± 0.040 | 0.560 ± 0.080 | 0.480 ± 0.068 |
| | Specificity | 0.720 ± 0.038 | 0.520 ± 0.194 | 0.440 ± 0.136 |
| | Sensitivity | 0.660 ± 0.035 | 0.600 ± 0.063 | 0.520 ± 0.117 |
| M-All 1 | Accuracy | 0.700 ± 0.061 | 0.690 ± 0.107 | 0.660 ± 0.066 |
| | Specificity | 0.770 ± 0.077 | 0.740 ± 0.080 | 0.660 ± 0.080 |
| | Sensitivity | 0.630 ± 0.070 | 0.640 ± 0.162 | 0.660 ± 0.150 |
| M-All 2 | Accuracy | 0.880 ± 0.027 | 0.840 ± 0.049 | 0.810 ± 0.058 |
| | Specificity | 0.890 ± 0.030 | 0.880 ± 0.075 | 0.800 ± 0.110 |
| | Sensitivity | 0.870 ± 0.023 | 0.800 ± 0.063 | 0.820 ± 0.098 |
| M-All 3 | Accuracy | 0.670 ± 0.027 | 0.560 ± 0.066 | 0.520 ± 0.121 |
| | Specificity | 0.660 ± 0.027 | 0.500 ± 0.190 | 0.500 ± 0.228 |
| | Sensitivity | 0.680 ± 0.030 | 0.620 ± 0.075 | 0.540 ± 0.206 |

analysis.

## 3.3 Discussion

In this section, we analyze the results of our study, focusing on the performance of the developed associative classifier in comparison to established Triclustering-Based approaches. We delve into the rationale behind our choice of metrics, explore the challenges encountered during data generation and triclustering, and examine the varying impact of these factors on classifier performance.

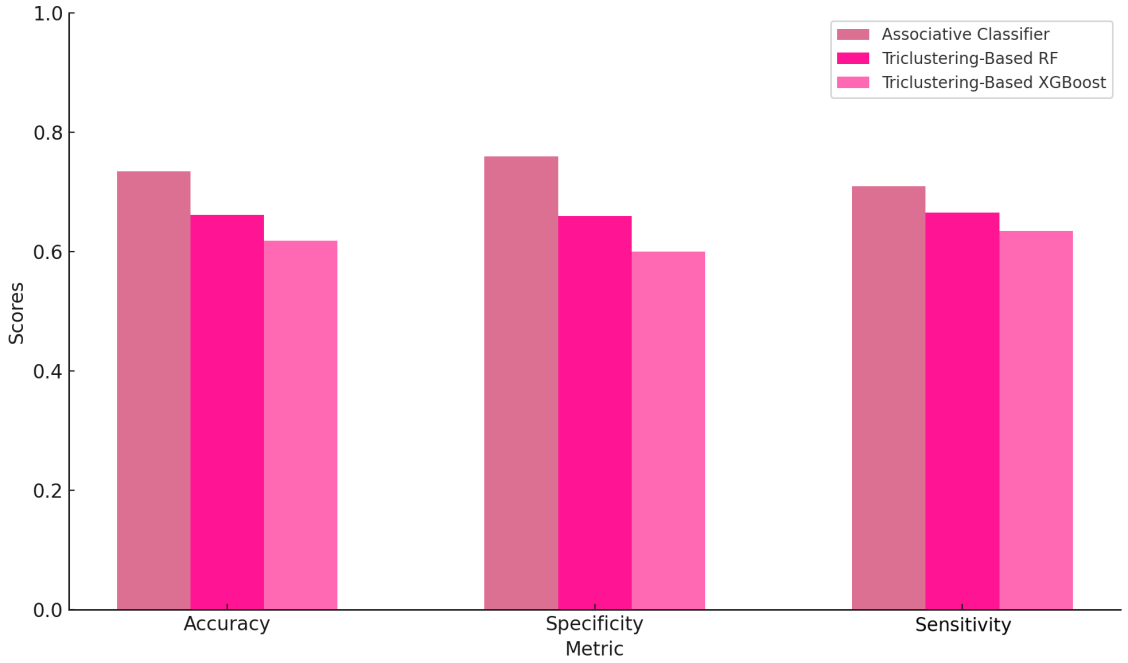


Figure 3.18: Mean values of results obtained for all manually created datasets with mixed patterns in the context dimension.

Table 3.17: Number of triclusters calculated by the TCTriCluster algorithm for each manually created dataset and the number of rules used by the associative classifier after applying the FP-Growth algorithm.

| **Dataset** | **Triclusters** | **Rules** |
|---|---|---|
| M-Constant 0 | 230 | 18 |
| M-Constant 1 | 551 | 27 |
| M-Constant 2 | 555 | 24 |
| M-Constant 3 | 420 | 26 |
| M-Additive 0 | 351 | 37 |
| M-Additive 1 | 429 | 28 |
| M-Additive 2 | 545 | 24 |
| M-Additive 3 | 373 | 57 |
| M-Multiplicative 0 | 375 | 60 |
| M-Multiplicative 1 | 501 | 42 |
| M-Multiplicative 2 | 623 | 38 |
| M-Multiplicative 3 | 466 | 56 |
| M-All 0 | 203 | 56 |
| M-All 1 | 544 | 18 |
| M-All 2 | 521 | 50 |
| M-All 3 | 410 | 40 |

Additionally, we evaluate the implications of our findings in the context of clinical data analysis, offering insights for future research directions.

### 3.3.1 Comparison with Triclustering-Based Classifiers

In the results section, we compared the performance of our developed associative classifier against two Triclustering-Based approaches using Random Forest and XGBoost. The rationale behind selecting these classifiers for comparison stems from the fact that all the classifiers involved utilize a triclustering algorithm and are designed to handle 3-way temporal data. In this study, the TCTriCluster algorithm, as used in the Triclustering-Based classifiers, was employed. Unlike conventional classifiers such as Random Forest, Support Vector Machines, K-Nearest Neighbors, and XGBoost, which do not explicitly account for the temporal dimension and treat it as regular features, the Triclustering-Based classifier developed by Soares et al. [3] has already demonstrated its effectiveness on 3-way temporal data from patients with ALS.

### 3.3.2 Justification for Metric Selection

The selection of accuracy (Equation (2.3)), specificity (Equation (2.5)), and sensitivity (or recall) (Equation (2.4)) as metrics to evaluate the performance of the associative classifier was made with careful consideration of the context of biomedical and clinical data, where the consequences of classification errors can be significant.

Accuracy is a straightforward and easily interpretable metric, providing a general measure of the classifier's overall performance by calculating the proportion of correct predictions over the total number of cases. However, accuracy alone can be misleading, especially in cases of imbalanced datasets, which are common in clinical data where one class (e.g., non-disease cases) might significantly outnumber another (e.g., disease cases) [61].

Specificity (true negative rate) measures the proportion of actual negatives (e.g., healthy patients) that are correctly identified by the classifier. In clinical and biomedical research, high specificity is crucial when the goal is to minimize false positives instances where healthy individuals are incorrectly classified as having a condition. Reducing false positives is essential to prevent unnecessary anxiety, avoid unwarranted further testing, and reduce healthcare costs [62].

Sensitivity (true positive rate or recall) measures the proportion of actual positives (e.g., patients with a disease) that are correctly identified by the classifier. In a clinical setting, high sensitivity is vital when the primary concern is to ensure that patients with a condition are correctly identified, thereby avoiding the risk of missing cases (false negatives). This is particularly important in early disease detection, where timely diagnosis can significantly impact treatment outcomes [62].

In biomedical and clinical applications, the balance between sensitivity and specificity is critical. A classifier with high sensitivity but low specificity might lead to over-diagnosis, causing unnecessary treatments, while a classifier with high specificity but low sensitivity might miss actual cases, leading to under-diagnosis or delayed treatment [62]. Therefore, using both sensitivity and specificity alongside accuracy allows for a more nuanced evaluation of the classifier's performance, ensuring that the model is not only accurate but also clinically reliable.

### 3.3.3 Challenges in Data Generation and Triclustering

The use of G-HTric to generate our data experiments was chosen because it has been extensively used and tested to simulate clinical 3-way temporal data [59]. Initially, we generated simple and easily interpretable data characterized by a smaller number of objects (120). Subsequently, the classifier was tested with mixed data, consisting of columns characterized by numeric or categorical instances. Although our associative classifier functioned under these new conditions, the TCTriCluster algorithm's inability to manage categorical data and its interpretation of those values exclusively as numeric limited the pertinence of the results. Additionally, we conducted tests to evaluate our classifier against imbalanced datasets and 3-class datasets. However, due to time constraints, we were unable to present conclusive and reliable results under these conditions.

### 3.3.4 Analysis of Results

Summarizing the results obtained from the G-HTric generated datasets, the performance was not satisfactory. Indeed, the associative classifier did not outperform the Triclustering-Based classifiers for any metric across all tested patterns: Figure (3.11) for the constant pattern, Figure (3.12) for the additive pattern, Figure (3.13) for the multiplicative pattern, and Figure (3.14) for the order-preserving pattern. However, the bar graphs show that the differences in results among the classifiers were not substantial. Excluding the approximately 9% difference observed in the mean results for the additive pattern, for the other patterns, the performance differences ranged between 2% and 5%, suggesting a degree of competitiveness among the classifiers.

When examining the results for each dataset individually (Table (3.6) for the constant pattern datasets, Table (3.7) for the additive pattern, Table (3.8) for the multiplicative pattern, and Table (3.9) for the order-preserving pattern), there were instances where our classifier outperformed the Triclustering-Based classifiers. Specifically, the results from the Additive 1 dataset (Table (3.7)) outperformed the Triclustering-Based XGBoost by 6%, the Multiplicative 1 dataset (Table (3.8)) outperformed the Triclustering-Based XGBoost by 2%, and the Order Preserving 1 and 2 datasets (Table (3.9)) outperformed the Triclustering-Based RF by 6% and the Triclustering-Based XGBoost by 9%, respectively.

### 3.3.5 Impact of Triclustering on Classifier's Performance

The varying results can be attributed to the number of triclusters calculated by TCTriCluster for these datasets (Table (3.10)). Several points need to be clarified regarding the application of TCTriCluster to the datasets generated by G-HTric. For each dataset, the time required to calculate the triclusters ranged from 2 to 6 days to obtain a minimum useful number of triclusters (at least ten). Running the TCTriCluster algorithm with a *w WINSZ* value (see the *w WINSZ* value in the **TCTriCluster Algorithm** section) that would result in a runtime of less than 2 days on these datasets yielded 0 or fewer than 10 triclusters. For the datasets used to present the results, we selected a *w WINSZ* value that would maximize the number of triclusters obtained within the

available time. Even then, TCTriCluster calculated fewer than ten triclusters in 50% of the datasets within a 6-day runtime.

An interesting observation is the significant number of triclusters obtained in the Order Preserving 0 dataset, which amounted to 22500 triclusters. Initially, we selected a *w WINSZ* value of 0.05, which took 2 days to yield 0 triclusters. However, running the algorithm with a *w WINSZ* value of 0.06 took 6 days to generate 22500 triclusters. This result strongly suggests that TCTriCluster has a critical point for each dataset where the number of calculated triclusters increases exponentially.

### 3.3.6 Evaluation of FP-Growth Algorithm's Impact

Initially, the incorporation of the FP-Growth algorithm into this associative classifier aimed to reduce the number of rules by grouping triclusters into rules obtained by the FP-Growth algorithm. However, when observing the number of triclusters obtained for each generated dataset (Table (3.10)), we noticed that it might be possible to test our classifier without using the FP-Growth algorithm on certain datasets. This would allow us to assess the potential loss of information caused by the application of FP-Growth in cases where the memory capacity of the computer allows for each tricluster to be used as a rule. The results presented in Table (3.11) suggest that, generally, the loss of information due to grouping triclusters into rules by the FP-Growth algorithm can result in worse classifier's performance. In some cases, the loss of information led to approximately 5% worse results, such as in the Additive 3 and Order Preserving 3 datasets. In other cases, the information loss was more pronounced, resulting in approximately 10% worse results, as observed in the Constant 2, Multiplicative 1, and Multiplicative 3 datasets. However, there was one instance where grouping 43 triclusters into 20 rules resulted in a 4% improvement, as seen in the Order Preserving 1 dataset. Comparing the results obtained by the classifier without using the FP-Growth algorithm with the results obtained by the Triclustering-Based classifiers, we can see that in the Constant 2 and Multiplicative 1 datasets, the associative classifier without FP-Growth outperformed the Triclustering-Based classifiers. These results suggest that a more in-depth and intensive set of tests on cases where each tricluster can be treated as a rule may be important to evaluate and examine how grouping the calculated triclusters into rules can negatively impact the classifier's performance.

### 3.3.7 Investigating the Limited Performance and Triclustering Challenges

Given the previous results, a more thorough investigation was conducted to understand the limited number of triclusters calculated and the suboptimal performance observed. Combining the fact that the G-HTric generator is unable to generate datasets with patterns exclusively in the context dimension (Table (3.5)), the TCTriCluster algorithm's step 2, **Mining of Maximal Biclusters** (**TCTriCluster Algorithm** section), which indicates that the algorithm starts by finding patterns within two-dimensional slices (objects and features), and our associative classifier is specialized in identifying and working with temporal patterns through linear regression across the time dimension, we arrived at a better understanding of the issues.

TCTriCluster initially identifies patterns across two dimensions (the object and feature dimensions). The third dimension (context dimension) is subsequently added to the previously identified biclusters. As a result, this prioritizes patterns appearing in the object and feature dimensions, making patterns in the third dimension harder to detect when more distinguishable two-dimensional patterns are present. In Figure (3.19), we see a representative tricluster obtained by TCTriCluster on the Additive 1 dataset generated by G-HTric. This tricluster successfully grouped all the objects that exhibited the exact same value between the objects and followed the same additive pattern across contexts. All the calculated triclusters in this dataset (9 triclusters) were of the same type, grouping objects with the same value and following the same pattern in the context dimension. Because the dataset also incorporated constant patterns in the object and feature dimensions, the triclustering algorithm focused on these types of patterns, as it begins by grouping patterns into biclusters based on the object and feature dimensions, making it difficult to obtain other types of triclusters and resulting in only 9 triclusters after several days of runtime. This output from TCTriCluster significantly limited our associative classifier, which had to classify all objects based on only 9 rules. The output from the other additive and multiplicative pattern generated datasets was similar, highlighting the classifier's heavy dependence on the effectiveness of the triclusters calculated by the triclustering algorithm.

This observation also reveals a limitation of the associative classifier in that it only considers patterns from the context dimension. Future work should aim to enhance the classifier's capabilities to also capture patterns observed in the feature dimension and to identify whether a present pattern should be considered a feature pattern, a context pattern, or both.

|T|x|S|x|G|: 3x9x128
Time: 0

| | S-0 | S-1 | S-2 | S-3 | S-5 | S-6 | S-7 | S-8 | S-9 |
|---|---|---|---|---|---|---|---|---|---|
| G-71 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 |
| G-96 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 |
| G-101 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 |
| G-103 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 |
| G-106 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 | 7.11 |

Figure 3.19: Example of a single tricluster calculated by TCTriCluster composed of 3 contexts, 9 features, and 128 objects on the dataset generated by G-HTric Additive 1.

### 3.3.8 Creation of Customized Datasets and Their Impact on Classifier's Performance

Based on the observations and deductions above, we decided to create our own simple datasets by only incorporating patterns in the context dimension. To avoid the TCTriCluster algorithm focusing too much on biclusters that exhibit the type of pattern illustrated previously, all objects were assigned a random number to start the same type of pattern (all incorporated patterns are listed in Table (10), Table (12), Table (14), and Table (16) in the Appendix). This choice proved to be very effective in allowing the TCTriCluster algorithm to calculate several triclusters (Table (3.17)) in a short amount of time (the average runtime was about 20 minutes).

Indeed, this limitation in G-HTric could be unfavorable when trying to generate datasets similar to those obtained from clinical records. It is common for two different patients to follow a similar

pattern on some features across different contiguous contexts. However, this does not mean that the values in the considered features will be the same or even in the same scale or units, making the created datasets with constant patterns across objects and features and an additive pattern on contexts inaccurate for such situations.

In the clinically tested ALS dataset by Soares et al. [3], where the TCTriCluster algorithm was applied, none of the considered features had the same units, making it inaccurate to generate a dataset similar to this type by including a constant pattern in the objects and features. This is because the features may have values on different scales or even on the same scale but with slightly different values. However, this limitation does not encompass all types of clinical data.

For future work, it would be valuable to explore the performance of our associative classifier by varying the number of calculated triclusters. Investigating how the number of triclusters influences classifier's performance could provide insights into optimizing the classifier for different datasets.

### 3.3.9 Validation of the Classifier on the Manually Generated Datasets

Finally, in the last phase of testing, we focused on creating datasets that only incorporated patterns across the context dimension. Datasets with a reduced number of objects (100) were generated, in contrast to the 1000 objects used in previous tests, maintaining datasets with only 2 balanced classes. However, we incorporated 10 triclusters, each with 10 objects, to evaluate whether the classifier could capture these patterns.

According to the bar graphs in Figure (3.15), Figure (3.16), Figure (3.17), and Figure (3.18), the mean results obtained by our associative classifier compared with the Triclustering-Based classifiers were approximately 15% higher across the evaluated metrics. When the Triclustering-Based classifiers achieved performances around 50% to 70%, the associative classifier achieved 60% to 80%. According to the workflow of the Triclustering-Based classifier, all the calculated triclusters have a weight different from 0 [3], meaning that an unrepresentative tricluster could negatively impact the classifier's performance. A significant advantage of our associative classifier is that if a tricluster does not fit a specific pattern, its weight is set to 0, discarding the impact of that tricluster and making our classifier more robust to unrepresentative triclusters. In summary, our associative classifier benefits from a high number of calculated triclusters. According to Table (3.13), Table (3.14), Table (3.15), and Table (3.16), except for the sensitivity metric in the M-All 1 dataset, where the Triclustering-Based XGBoost obtained the highest result, our associative classifier outperformed the other classifiers in all the remaining datasets and metrics.

### 3.3.10 Classifier's Interpretability

From the analysis, our developed associative classifier has demonstrated its effectiveness when applied to 3-way temporal datasets, particularly in capturing patterns across the context dimension. A significant strength of this classifier is its ability to condense several triclusters into a smaller set of rules (see Table (3.17)), which simplifies the classification process. The reduction in the number of rules, up to 5% of the original number of triclusters, enhances the classifier's efficiency

without compromising accuracy. This approach not only streamlines the classification process but also improves interpretability, making it easier to identify the most relevant triclusters for each test object. This targeted identification process is crucial for advancing personalized analysis in clinical settings, potentially leading to more accurate and individualized patient care.

### 3.3.11 Implications for Clinical Data Analysis

The implications of our findings extend beyond the technical performance of the classifier. By enabling researchers to focus on a smaller, more relevant subset of triclusters, this approach significantly enhances the efficiency of clinical data analysis. This is particularly important in a clinical context, where the ability to quickly and accurately identify key patterns can have a profound impact on patient outcomes. By prioritizing the most critical triclusters, our method supports more targeted and personalized patient care, aligning with the broader goals of precision medicine. Future work should explore the application of this classifier across different types of clinical datasets, investigate the potential integration of feature patterns, and further refine the classifier's capabilities to enhance its applicability in diverse medical contexts.

### 3.3.12 Final Remarks

In summary, our associative classifier has proven effective in managing 3-way temporal datasets, particularly by condensing triclusters into a more interpretable set of rules. This capability not only enhances the efficiency of the classification process but also supports more personalized clinical data analysis. These findings underscore the potential of our approach to contribute significantly to precision medicine. The insights gained from this study will inform the concluding remarks and outline future research directions in the subsequent section.

# Chapter 4

# Conclusion and Future Work

This section provides a comprehensive summary of the research conducted, highlighting the key findings and their implications for the field of biomedical data analysis. It also outlines potential avenues for future research, emphasizing the need for further refinement and testing to enhance the applicability and robustness of the developed associative classifier. By addressing both the conclusions drawn from the study and the opportunities for future work, this section aims to offer a clear direction for advancing this research area.

## 4.1 Conclusion

This study explored the development and evaluation of a temporal associative classifier that leverages discriminative triclusters to address the complexities inherent in biomedical data analysis. The research aimed to create a classifier capable of handling high-dimensional which is a common challenge in clinical datasets.

The results of the study revealed both the potential and the limitations of the proposed associative classifier. The classifier exhibited robust performance in scenarios where patterns were predominantly observed within the context dimension. In these cases, the classifier effectively identified and utilized triclusters, which led to more accurate predictions. However, when patterns were less prominent or were distributed across other dimensions, the classifier's performance was more limited, highlighting its dependency on the nature of the data and the quality of triclusters generated.

Despite these challenges, the study contributes valuable insights into the use of triclustering for temporal data classification in biomedical contexts. The associative classifier's ability to condense complex triclusters into a manageable and interpretable set of rules is a significant advancement, offering practical applications in clinical settings where interpretability and accuracy are crucial. The classifier's performance, particularly in personalized and precision medicine, underscores its potential to aid in the analysis of complex biomedical data, providing a foundation for future enhancements.

Overall, this research lays the groundwork for further exploration and refinement of Triclustering-Based associative classifiers, with the goal of improving their applicability and robustness in

real-world biomedical applications. The insights gained suggest that with targeted improvements, such classifiers could play a crucial role in advancing healthcare solutions.

## 4.2 Future Work

The findings from this research open several avenues for future investigation:

- **Enhancing Triclustering Algorithms:** The performance of the associative classifier heavily relies on the quality and quantity of triclusters generated by the TCTriCluster algorithm. Future research should focus on refining triclustering algorithms to better handle heterogeneous data types, including categorical data, and to improve the algorithm's scalability for large and complex datasets.

- **Improving Temporal Alignment Techniques:** One of the challenges identified in this study is the effective alignment of temporal data. Future work could develop more sophisticated temporal alignment techniques that account for irregular intervals and misaligned data points, which are common in biomedical datasets. This improvement would enhance the classifier's ability to detect and utilize subtle temporal patterns that may be crucial for disease progression analysis.

- **Expanding the Scope to Feature Patterns:** The current study focused on patterns within the context dimension, which limited the classifier's ability to capture more complex relationships within the data. Future research could expand the scope to include feature patterns, enabling the classifier to identify and leverage interactions across all three dimensions (object, feature, and context). This would likely result in a more holistic and accurate model, capable of uncovering deeper insights from multidimensional biomedical data.

- **Addressing the Limitation of Linear Pattern Identification:** The classifier's current approach, which relies on linear regression for numerical features, is restricted to identifying linear patterns across the context dimension. A significant direction for future work would be to adapt this approach to handle both linear and non-linear patterns. Incorporating methods such as genetic algorithms, which are capable of identifying a broader range of patterns, could substantially improve the classifier's ability to capture complex, non-linear relationships within the data.

- **Validation with Real Clinical Data:** While synthetic datasets generated by the G-HTric tool provided a controlled environment for initial testing, real clinical datasets would present additional challenges and opportunities. Future research should focus on applying the associative classifier to real-world clinical data, such as longitudinal patient records. This would not only validate the classifier's utility in practical settings but also provide insights into its performance under more variable and noisy conditions typical of real-world data.

- **Optimizing Computational Efficiency:** The TCTriCluster algorithm's computational demands, particularly the time required to generate a sufficient number of triclusters, pose a significant challenge. Future research could explore optimizations in algorithm design, including parallel processing and the use of more efficient data structures, to reduce computational overhead. Additionally, developing heuristics or approximation methods to balance accuracy with computational feasibility could make the classifier more practical for use in time-sensitive clinical environments.

- **Exploring the Impact of Tricluster Quantity and Quality:** The study highlighted that the performance of the associative classifier is highly dependent on the number and quality of triclusters generated. Future research could investigate the relationship between the number of triclusters and classifier performance in greater detail. This could include experimenting with different thresholds for tricluster significance and exploring the trade-offs between tricluster quantity and classification accuracy.

- **In-depth Analysis of FP-Growth Algorithm's Impact:** The FP-Growth algorithm was used to reduce the number of rules by grouping triclusters. However, this grouping sometimes led to a loss of information, negatively impacting classifier performance. Future research could focus on a more detailed analysis of the FP-Growth algorithm's impact, investigating scenarios where its application may or may not be beneficial. Developing criteria for when to apply FP-Growth could optimize the balance between interpretability and accuracy.

- **Custom Dataset Generation for Clinical Relevance:** The limitations of the G-HTric generator in producing datasets that mimic real-world clinical scenarios were identified as a constraint. Future research could develop more advanced data generators that better simulate the heterogeneity and complexity of clinical data. This would provide a more rigorous test for the associative classifier and potentially reveal new insights into its performance in realistic settings.

In conclusion, while the proposed temporal associative classifier has shown promise, particularly in its ability to handle temporal patterns, further research is needed to address its limitations and enhance its applicability in clinical settings. The proposed advancements could significantly contribute to ongoing efforts to leverage data science for improved healthcare outcomes.

# Bibliography


[1] Henriques, R., Madeira, S.C. (2019). Triclustering Algorithms for Three-Dimensional Data Analysis: A Comprehensive Survey. *ACM Computing Surveys*, 51(5), Article 95. https://doi.org/10.1145/3195833.

[2] Amar, D., Yekutieli, D., Maron-Katz, A., Hendler, T., Shamir, R. (2015). A hierarchical Bayesian model for flexible module discovery in three-way time-series data. *Bioinformatics*, 31(12), i17–i26. https://doi.org/10.1093/bioinformatics/btv228.

[3] Soares, D.F., Henriques, R., Gromicho, M., de Carvalho, M., Madeira, S.C. (2023). Triclustering-based classification of longitudinal data for prognostic prediction: targeting relevant clinical endpoints in amyotrophic lateral sclerosis. *Scientific Reports*, 13(1). https://doi.org/10.1038/s41598-023-33223-x.

[4] Peng, R.D., Bell, M.L. (2010). Spatial misalignment in time series studies of air pollution and health data. *Biostatistics*, 11(4), pp. 720–740. https://doi.org/10.1093/biostatistics/kxq017.

[5] Henriques, R., Madeira, S.C. (2021). FleBiC: Learning classifiers from high-dimensional biomedical data using discriminative biclusters with non-constant patterns. *Pattern Recognition*, 115, 107900. https://doi.org/10.1016/j.patcog.2021.107900.

[6] Soares, D.F., Henriques, R., Gromicho, M., de Carvalho, M., Madeira, S.C. (2022). Learning prognostic models using a mixture of biclustering and triclustering: Predicting the need for non-invasive ventilation in Amyotrophic Lateral Sclerosis. *Journal of Biomedical Informatics*, 134, 104172. https://doi.org/10.1016/j.jbi.2022.104172.

[7] Abdelhamid, N., Thabtah, F. (2014). Associative Classification Approaches: Review and Comparison. *Journal of Information & Knowledge Management*, 13(03), 1450027. https://doi.org/10.1142/s0219649214500270.

[8] Jabbar, M.A., Deekshatulu, B.L., Chandra, P. (2013). Knowledge Discovery Using Associative Classification for Heart Disease Prediction. In: Abraham, A., Thampi, S. (eds) *Intelligent Informatics*. Advances in Intelligent Systems and Computing, vol 182. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32063-7_4.

[9] Borgelt, C. (2012). Frequent item set mining. *WIREs Data Mining and Knowledge Discovery*, 2(6), pp. 437–456. https://doi.org/10.1002/widm.1074.

[10] Kotsiantis, S., Kanellopoulos, D. (2006). Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering*, 32(1), pp. 71-82.

[11] Raju, V. G., Lakshmi, K. P., Jain, V. M., Kalidindi, A., Padma, V. (2020, August). Study the influence of normalization/transformation process on the accuracy of supervised classification. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 729-735. IEEE.

[12] Jeni, L. A., Cohn, J. F., De La Torre, F. (2013). Facing Imbalanced Data—Recommendations for the Use of Performance Metrics. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pp. 245-251. IEEE. https://doi.org/10.1109/ACII.2013.47.

[13] Li, W., Han, J., Pei, J. (2001). CMAR: Accurate and efficient classification based on multiple class-association rules. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01)*, pp. 369-376. IEEE. https://doi.org/10.1109/ICDM.2001.989541.

[14] Noh, K., Lee, H.G., Shon, H.S., Lee, B.J., Ryu, K.H. (2006). Associative Classification Approach for Diagnosing Cardiovascular Disease. In: Huang, D.S., Li, K., Irwin, G.W. (eds) *Intelligent Computing in Signal Processing and Pattern Recognition*. Lecture Notes in Control and Information Sciences, vol 345. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-37258-5_82.

[15] Alwidian, J., Hammo, B. H., Obeid, N. (2018). WCBA: Weighted classification based on association rules algorithm for breast cancer disease. *Applied Soft Computing*, 62, pp. 536–549. https://doi.org/10.1016/j.asoc.2017.11.013.

[16] Wang, Z., Chung, J. W., Jiang, X., Cui, Y., Wang, M., Zheng, A. (2018). Machine Learning-Based Prediction System For Chronic Kidney Disease Using Associative Classification Technique. *International Journal of Engineering & Technology*, 7(3.36), pp. 177-182. https://doi.org/10.14419/ijet.v7i4.36.25377.

[17] Mankad, S., Michailidis, G. (2014). Biclustering three-dimensional data arrays with plaid models. *Journal of Computational and Graphical Statistics*, 23(4), pp. 943–965. https://doi.org/10.1080/10618600.2013.851608.

[18] Jiang, C., Duan, Y. (2023). A Novel Three-Way Deep Learning Approach for Multigranularity Fuzzy Association Analysis of Time Series Data. *IEEE Transactions on Fuzzy Systems*. https://doi.org/10.1109/TFUZZ.2023.3332921.

[19] Zhao, L., Zaki, M. J. (2005). Tricluster: An effective algorithm for mining coherent clusters in 3D microarray data. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05*, pp. 694–705. ACM. https://doi.org/10.1145/1066157.1066232.

[20] Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8(3), pp. 338-353. https://doi.org/10.1016/S0019-9958(65)90241-X.

[21] LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. *Nature*, 521, pp. 436–444. https://doi.org/10.1038/nature14539.

[22] Cheng, L., Hu, Y., Huang, T. (2018). Assessing local similarity in time series data by an alignment-free approach. *Bioinformatics*, 34(1), pp. 83-89. https://doi.org/10.1093/bioinformatics/btx532.

[23] Hasin, Y., Seldin, M., Lusis, A. (2017). Multi-omics approaches to disease. *Genome Biology*, 18, 83. https://doi.org/10.1186/s13059-017-1215-1.

[24] Bersanelli, M., Mosca, E., Remondini, D., Castellani, G., Milanesi, L. (2016). Network diffusion-based analysis of high-throughput data for the detection of differentially enriched modules. *Scientific Reports*, 6, 34841. https://doi.org/10.1038/srep34841.

[25] Bruggeman, F. J., Westerhoff, H. V. (2007). The nature of systems biology. *Trends in Microbiology*, 15(1), pp. 45-50. https://doi.org/10.1016/j.tim.2006.11.003.

[26] Madeira, S. C., Oliveira, A. L. (2004). Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1), pp. 24–45. https://doi.org/10.1109/TCBB.2004.2.

[27] Lin, Y. R., Sun, J., Castro, P., Konuru, R., Sundaram, H., Kelliher, A. (2009). Metafac: Community discovery via relational hypergraph factorization. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 527–536. ACM. https://doi.org/10.1145/1557019.1557079.

[28] Mahanta, P., Ahmed, H. A., Bhattacharyya, D. K., Kalita, J. K. (2011). Triclustering in gene expression data analysis: A selected survey. In *Proceedings of the 2nd National Conference on Emerging Trends and Applications in Computer Science (NCETACS'11)*, pp. 1-6. IEEE. https://doi.org/10.1109/NCETACS.2011.5751302.

[29] Sim, K., Aung, Z., Gopalkrishnan, V. (2010). Discovering correlated subspace clusters in 3D continuous-valued data. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, pp. 471–480. IEEE. https://doi.org/10.1109/ICDM.2010.53.

[30] Liu, Y. C., Lee, C. H., Chen, W. C., Shin, J. W., Hsu, H. H., Tseng, V. S. (2010). A novel method for mining temporally dependent association rules in three-dimensional microarray datasets. In *Proceedings of the 2010 International Computer Symposium (ICS'10)*, pp. 759–764. IEEE. https://doi.org/10.1109/ICS.2010.5690808.

[31] Kakati, T., Ahmed, H. A., Bhattacharyya, D. K., Kalita, J. K. (2018). THD-Tricluster: A robust triclustering technique and its application in condition-specific change analysis in HIV-1 progression data. *Computational Biology and Chemistry*, 75, pp. 154-167. https://doi.org/10.1016/j.compbiolchem.2018.05.007.

[32] Wu, X., Zheng, D. (2020). Tri-Clustering Based Exploration of Temporal Resolution Impacts on Spatio-Temporal Clusters in Geo-Referenced Time Series. *ISPRS International Journal of Geo-Information*, 9(4), 210. https://doi.org/10.3390/ijgi9040210.

[33] Bhar, A., Haubrock, M., Mukhopadhyay, A., Maulik, U., Bandyopadhyay, S., Wingender, E. (2012). $\delta$-TRIMAX: Extracting triclusters and analysing coregulation in time series gene expression data. In *International Workshop on Algorithms in Bioinformatics*, pp. 165–177. Springer. https://doi.org/10.1007/978-3-642-33122-0_13.

[34] Jiang, D., Pei, J., Ramanathan, M., Tang, C., Zhang, A. (2004). Mining coherent gene clusters from gene-sample-time microarray data. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'04)*, pp. 430–439. ACM. https://doi.org/10.1145/1014052.1014101.

[35] Ahmed, H. A., Mahanta, P., Bhattacharyya, D. K., Kalita, J. K., Ghosh, A. (2011). Intersected coexpressed subcube miner: An effective triclustering algorithm. In *2011 World Congress on Information and Communication Technologies (WICT'11)*, pp. 846–851. IEEE. https://doi.org/10.1109/WICT.2011.6141358.

[36] Gutiérrez-Avilés, D., Rubio-Escudero, C., Martínez-Álvarez, F., Riquelme, J. C. (2014). TriGen: A genetic algorithm to mine triclusters in temporal gene expression data. *Neurocomputing*, 132, pp. 42–53. https://doi.org/10.1016/j.neucom.2013.03.061.

[37] Siswantining, T., Fithriasari, K., Titisari, P. (2022). $\delta$-TRIMAX: A robust triclustering technique and its application in condition-specific change analysis in HIV-1 progression data. *Computational Biology and Chemistry*, 75, pp. 154-167. https://doi.org/10.1016/j.compbiolchem.2018.05.007.

[38] Gutiérrez-Avilés, D., Rubio-Escudero, C. (2014). Mining 3D patterns from gene expression temporal data: A new tricluster evaluation measure. *Scientific World Journal*, 2014, pp. 1–16. https://doi.org/10.1155/2014/624371.

[39] Bhar, A., Haubrock, M., Mukhopadhyay, A., Wingender, E. (2015). Multiobjective triclustering of timeseries transcriptome data reveals key genes of biological processes. *BMC Bioinformatics*, 16(1), 1. https://doi.org/10.1186/s12859-015-0635-8.

[40] Li, Y., Ngom, A. (2010). Classification of clinical gene-sample-time microarray expression data via tensor decomposition methods. In *Proceedings of the Computational Intelligence Methods for Bioinformatics and Biostatistics (CIBB)*, pp. 275-286. Springer. https://doi.org/10.1007/978-3-642-21946-7_22.

[41] Tchagang, A. B., Phan, S., Famili, F., Shearer, H., Fobert, P., Huang, Y., Zou, J., Huang, D., Cutler, A., Liu, Z., Pan, Y. (2012). Mining biological information from 3D short time-series gene expression data: The OPTricluster algorithm. *BMC Bioinformatics*, 13(1), 1. https://doi.org/10.1186/1471-2105-13-54.

[42] Baralis, E., Garza, P. (2012). I-prune: Item selection for associative classification. *International Journal of Intelligent Systems*, 27(3), pp. 279-299. https://doi.org/10.1002/int.21524.

[43] Tawiah, C. A., Sheng, V. (2013). A study on multi-label classification. In *Advances in Data Mining. Applications and Theoretical Aspects*. Lecture Notes in Computer Science, 7987, pp. 137–150. https://api.semanticscholar.org/CorpusID:10583980.

[44] Farghaly, H., Ali, A., Abd El-Hafeez, T. (2020). Building an Effective and Accurate Associative Classifier Based on Support Vector Machine. *Sylwan*. https://www.researchgate.net/publication/339750509.

[45] Christopher, J. (2019). The Science of Rule-based Classifiers. In *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 299-303). IEEE. https://doi.org/10.1109/CONFLUENCE.2019.8776954.

[46] Attanasio, G., Cagliero, L., Baralis, E. (2020). Leveraging the explainability of associative classifiers to support quantitative stock trading. In *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling (DSMM '20)*. Association for Computing Machinery, New York, NY, USA, Article 2, pp. 1–6. https://doi.org/10.1145/3401832.3402679.

[47] Rajeswari, A.M., Deisy, C. (2020). Fuzzy logic based associative classifier for slow learners prediction. *Journal of Intelligent and Fuzzy Systems*, 36, pp. 2691-2704. https://doi.org/10.3233/JIFS-18748.

[48] Almasi, M., Saniee Abadeh, M. (2020). CARs-Lands: An associative classifier for large-scale datasets. *Pattern Recognition*, 100, 107128. https://doi.org/10.1016/j.patcog.2019.107128.

[49] Sivanantham, S., Mohanraj, V., Suresh, Y., et al. (2023). Rule precision index classifier: an associative classifier with a novel pruning measure for intrusion detection. *Personal and Ubiquitous Computing*, 27, pp. 1395–1403. https://doi.org/10.1007/s00779-021-01599-0.

[50] Kabir, M.R., Vaid, S., Sood, N., Zaïane, O. (2022). Deep Associative Classifier. In *Proceedings of the 2022 IEEE International Conference on Knowledge Graph (ICKG)*, pp. 113-122. IEEE. https://doi.org/10.1109/ICKG55886.2022.00022.

[51] Muhammad, A. N., Aseere, A. M., Chiroma, H., et al. (2021). Deep learning application in smart cities: recent development, taxonomy, challenges and research prospects. *Neural Computing & Applications*, 33, pp. 2973–3009. https://doi.org/10.1007/s00521-020-05151-8.

[52] Jankulak, M. L. (2012). Prediction of rapid intensity changes in tropical cyclones using associative classification. Doctoral dissertation, University of Miami.

[53] Wang, P., Shi, T., Reddy, C. K. (2021). A Novel Tensor-Based Temporal Multi-Task Survival Analysis Model. *IEEE Transactions on Knowledge and Data Engineering*, 33(9), pp. 3311–3322. https://doi.org/10.1109/TKDE.2020.2967700.

[54] Nesaragi, N., Patidar, S., Aggarwal, V. (2021). Tensor learning of pointwise mutual information from EHR data for early prediction of sepsis. *Computers in Biology and Medicine*, 134, 104430. https://doi.org/10.1016/j.compbiomed.2021.104430.

[55] Li, T., Fu, Z., Liu, X., Qi, S., Calhoun, V. D., Sui, J. (2019). Multimodal neuroimaging patterns associated with social responsiveness impairment in autism: A replication study. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pp. 409-413. IEEE. https://doi.org/10.1109/ISBI.2019.8759515.

[56] Qi, S., Plis, S. M., Miller, R., Silva, R. F., Vergara, V. M., Jiang, R., Zhi, D., Sui, J., Calhoun, V. D. (2021). 3-way parallel fusion of spatial (sMRI/dMRI) and spatio-temporal (fMRI) data with application to schizophrenia. In *Proceedings - International Symposium on Biomedical Imaging*, April 2021, pp. 1577–1581. IEEE. https://doi.org/10.1109/ISBI48211.2021.9433935.

[57] Fu, D., Fang, L., MacIejewski, R., Torvik, V. I., He, J. (2022). Meta-Learned Metrics over Multi-Evolution Temporal Graphs. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 367–377. ACM. https://doi.org/10.1145/3534678.3539313.

[58] Feng, X., Zhang, H., Wang, C., Zheng, H. (2022). Traffic Data Recovery From Corrupted and Incomplete Observations via Spatial-Temporal TRPCA. *IEEE Transactions on Intelligent Transportation Systems*, 23(10), pp. 17835–17848. https://doi.org/10.1109/TITS.2022.3151925.

[59] Soares, D. F., Henriques, R., Madeira, S. C. (n.d.). G-HTric: Synthetic Generation of Heterogeneous Three-way Datasets with Annotated Triclustering Solutions. (submitted).

[60] Lobo, J., Henriques, R., Madeira, S. C. (2021). G-tric: generating three-way synthetic datasets with triclustering solutions. *BMC Bioinformatics*, 22(1), pp. 1–28. https://doi.org/10.1186/s12859-021-04045-9.

[61] Roy, D., Roy, A., Roy, U. (2024). Learning from Imbalanced Data in Healthcare: State-of-the-Art and Research Challenges. In: Acharjya, D. P., Ma, K. (eds) *Computational Intelligence in Healthcare Informatics*. Studies in Computational Intelligence, vol 1132. Springer, Singapore. https://doi.org/10.1007/978-981-99-8853-2_2.

[62] Patadia, V. K., Schuemie, M. J., Coloma, P., et al. (2015). Evaluating performance of electronic healthcare records and spontaneous reporting data in drug safety signal detection. *International Journal of Clinical Pharmacy*, 37, pp. 94–104. https://doi.org/10.1007/s11096-014-0044-5.

# Appendix

This appendix contains supplementary tables referenced throughout the thesis. These materials provide additional detail and support for the findings discussed in the main chapters. Each table is labeled and referenced.

Table 1: Number of objects, features, and contexts for each tricluster generated in the datasets incorporating the constant pattern in the context dimension.

| Dataset | Objects | Features | Contexts |
|---|---|---|---|
| Constant 0 | 143 | 3 | 2 |
| | 173 | 9 | 3 |
| | 148 | 7 | 2 |
| | 175 | 5 | 4 |
| | 123 | 8 | 4 |
| Constant 1 | 175 | 8 | 3 |
| | 103 | 8 | 3 |
| | 105 | 5 | 2 |
| | 189 | 3 | 4 |
| | 122 | 2 | 3 |
| Constant 2 | 171 | 8 | 3 |
| | 172 | 5 | 4 |
| | 163 | 8 | 4 |
| | 129 | 2 | 3 |
| | 186 | 2 | 4 |
| Constant 3 | 129 | 6 | 2 |
| | 180 | 7 | 2 |
| | 153 | 4 | 2 |
| | 150 | 7 | 3 |
| | 136 | 5 | 3 |

Table 2: Number of objects, features, and contexts for each tricluster generated in the datasets incorporating the additive pattern in the context dimension.

| Dataset | Objects | Features | Contexts |
|---|---|---|---|
| Additive 0 | 115 | 3 | 4 |
| | 187 | 3 | 2 |
| | 159 | 5 | 3 |
| | 171 | 9 | 3 |
| | 141 | 8 | 4 |
| Additive 1 | 149 | 2 | 3 |
| | 113 | 9 | 3 |
| | 127 | 3 | 2 |
| | 112 | 7 | 3 |
| | 128 | 9 | 3 |
| Additive 2 | 150 | 4 | 3 |
| | 153 | 6 | 2 |
| | 199 | 9 | 2 |
| | 147 | 3 | 3 |
| | 164 | 2 | 2 |
| Additive 3 | 188 | 7 | 3 |
| | 117 | 9 | 2 |
| | 154 | 2 | 3 |
| | 165 | 9 | 4 |
| | 105 | 6 | 4 |

Table 3: Number of objects, features, and contexts for each tricluster generated in the datasets incorporating the multiplicative pattern in the context dimension.

| Dataset | Objects | Features | Contexts |
|---|---|---|---|
| Multiplicative 0 | 114 | 6 | 2 |
| | 111 | 4 | 2 |
| | 156 | 6 | 3 |
| | 107 | 4 | 4 |
| | 198 | 3 | 2 |
| Multiplicative 1 | 181 | 4 | 2 |
| | 104 | 5 | 4 |
| | 194 | 3 | 4 |
| | 154 | 5 | 3 |
| | 133 | 7 | 3 |
| Multiplicative 2 | 126 | 4 | 2 |
| | 116 | 8 | 2 |
| | 183 | 4 | 2 |
| | 137 | 4 | 4 |
| | 116 | 2 | 3 |
| Multiplicative 3 | 130 | 9 | 3 |
| | 190 | 2 | 3 |
| | 151 | 8 | 4 |
| | 103 | 5 | 3 |
| | 130 | 3 | 3 |

Table 4: Number of objects, features, and contexts for each tricluster generated in the datasets incorporating the order preserving pattern in the context dimension.

| Dataset | Objects | Features | Contexts |
|---|---|---|---|
| Order Preserving 0 | 198 | 7 | 2 |
| | 146 | 3 | 4 |
| | 123 | 3 | 3 |
| | 178 | 4 | 4 |
| | 160 | 4 | 3 |
| Order Preserving 1 | 151 | 6 | 2 |
| | 129 | 9 | 4 |
| | 181 | 5 | 4 |
| | 155 | 7 | 4 |
| | 175 | 4 | 4 |
| Order Preserving 2 | 126 | 4 | 2 |
| | 116 | 8 | 2 |
| | 183 | 4 | 2 |
| | 137 | 4 | 4 |
| | 158 | 7 | 4 |
| Order Preserving 3 | 109 | 5 | 4 |
| | 171 | 8 | 3 |
| | 145 | 7 | 2 |
| | 180 | 9 | 4 |
| | 146 | 3 | 3 |

Table 5: Mean values of the results obtained for all the datasets with the constant pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.572 ± 0.038 | 0.612 ± 0.024 | 0.605 ± 0.035 |
| Specificity | 0.588 ± 0.032 | 0.629 ± 0.064 | 0.607 ± 0.041 |
| Sensitivity | 0.582 ± 0.029 | 0.594 ± 0.045 | 0.602 ± 0.046 |

Table 6: Mean values of the results obtained for all the datasets with the additive pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.549 ± 0.014 | 0.631 ± 0.037 | 0.628 ± 0.032 |
| Specificity | 0.549 ± 0.023 | 0.644 ± 0.042 | 0.638 ± 0.034 |
| Sensitivity | 0.530 ± 0.028 | 0.618 ± 0.047 | 0.618 ± 0.046 |

Table 7: Mean values of the results obtained for all the datasets with the multiplicative pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.567 ± 0.019 | 0.610 ± 0.030 | 0.600 ± 0.037 |
| Specificity | 0.582 ± 0.023 | 0.642 ± 0.044 | 0.600 ± 0.070 |
| Sensitivity | 0.553 ± 0.020 | 0.580 ± 0.061 | 0.601 ± 0.078 |

Table 8: Mean values of the results obtained for all the datasets with the order preserving pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.611 ± 0.026 | 0.636 ± 0.028 | 0.618 ± 0.029 |
| Specificity | 0.626 ± 0.028 | 0.653 ± 0.052 | 0.605 ± 0.037 |
| Sensitivity | 0.595 ± 0.025 | 0.620 ± 0.038 | 0.629 ± 0.045 |

Table 9: Mean values of the results obtained for all the manually created datasets with the constant pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.675 ± 0.104 | 0.505 ± 0.106 | 0.513 ± 0.104 |
| Specificity | 0.659 ± 0.094 | 0.530 ± 0.188 | 0.510 ± 0.223 |
| Sensitivity | 0.681 ± 0.103 | 0.480 ± 0.171 | 0.515 ± 0.114 |

Table 10: Number of objects, features, and contexts for each tricluster generated in the manually created datasets incorporating only the constant pattern in the context dimension.

| Dataset | Pattern | Feature |
|---|---|---|
| M-Constant 0 | Pattern 1 | y0 |
| | Pattern 2 | y1 |
| | Pattern 3 | y2 |
| | Pattern 4 | y3 |
| | Pattern 5 | y4 |
| | Pattern 6 | y0, y1 |
| | Pattern 7 | y2, y3 |
| | Pattern 8 | y0, y3, y4 |
| | Pattern 9 | y0, y1, y2 |
| | Pattern 10 | y2, y4 |
| M-Constant 1 | Pattern 1 | y0, y2 |
| | Pattern 2 | y1, y3 |
| | Pattern 3 | y0, y4 |
| | Pattern 4 | y1, y2 |
| | Pattern 5 | y3, y4 |
| | Pattern 6 | y0, y1, y4 |
| | Pattern 7 | y2, y3, y0 |
| | Pattern 8 | y1, y4, y2 |
| | Pattern 9 | y3, y0, y1 |
| | Pattern 10 | y4, y2, y3 |
| M-Constant 2 | Pattern 1 | y1 |
| | Pattern 2 | y3 |
| | Pattern 3 | y4 |
| | Pattern 4 | y0, y2 |
| | Pattern 5 | y1, y4 |
| | Pattern 6 | y2, y3 |
| | Pattern 7 | y0, y1 |
| | Pattern 8 | y3, y4 |
| | Pattern 9 | y0, y2, y4 |
| | Pattern 10 | y1, y2, y3 |
| M-Constant 3 | Pattern 1 | y1, y4 |
| | Pattern 2 | y0, y2, y3 |
| | Pattern 3 | y1, y2, y4 |
| | Pattern 4 | y0, y1, y3 |
| | Pattern 5 | y3, y4 |
| | Pattern 6 | y0, y2 |
| | Pattern 7 | y1, y3, y4 |
| | Pattern 8 | y0, y1, y2, y4 |
| | Pattern 9 | y2, y3, y4 |
| | Pattern 10 | y0, y2, y4 |

Table 11: Mean values of the results obtained for all the manually created datasets with the additive pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| **Metric** | **Associative Classifier** | **Triclustering-Based RF** | **Triclustering-Based XGBoost** |
|---|---|---|---|
| Accuracy | 0.715 ± 0.073 | 0.580 ± 0.072 | 0.583 ± 0.091 |
| Specificity | 0.718 ± 0.073 | 0.570 ± 0.178 | 0.570 ± 0.190 |
| Sensitivity | 0.713 ± 0.079 | 0.590 ± 0.158 | 0.595 ± 0.112 |

Table 12: Number of objects, features, and contexts for each tricluster generated in the manually created datasets incorporating only the additive pattern in the context dimension.

| Dataset | Pattern | Feature |
|---|---|---|
| M-Additive 0 | Pattern 1 | y0 (+0.5) |
| | Pattern 2 | y1 (+1) |
| | Pattern 3 | y2 (-0.5) |
| | Pattern 4 | y3 (-1) |
| | Pattern 5 | y4 (+1.3) |
| | Pattern 6 | y0 (+1.1), y1 (+0.7) |
| | Pattern 7 | y2 (-0.3), y3 (-0.5) |
| | Pattern 8 | y0 (+0.2), y3 (+0.3), y4 (+0.4) |
| | Pattern 9 | y0 (+0.9), y1 (+1.4), y2 (-0.7) |
| | Pattern 10 | y2 (+1.2), y4 (+1.6) |
| M-Additive 1 | Pattern 1 | y0 (+1.2) |
| | Pattern 2 | y1 (-0.8) |
| | Pattern 3 | y2 (+0.7) |
| | Pattern 4 | y3 (-1.5) |
| | Pattern 5 | y4 (+0.5) |
| | Pattern 6 | y0 (+0.6), y3 (-0.4) |
| | Pattern 7 | y3 (+1.1), y4 (-0.7) |
| | Pattern 8 | y0 (-0.3), y3 (+0.9), y4 (-0.2) |
| | Pattern 9 | y1 (+1.3), y2 (-0.5), y4 (+0.6) |
| | Pattern 10 | y2 (-1.0), y4 (+1.8) |
| M-Additive 2 | Pattern 1 | y0 (-1.0) |
| | Pattern 2 | y1 (+1.5) |
| | Pattern 3 | y2 (+0.8) |
| | Pattern 4 | y3 (-0.8) |
| | Pattern 5 | y4 (-1.2) |
| | Pattern 6 | y0 (+0.6), y2 (-0.4) |
| | Pattern 7 | y1 (+0.5), y3 (+1.2) |
| | Pattern 8 | y0 (+0.4), y3 (+0.5), y4 (-0.6) |
| | Pattern 9 | y1 (+0.8), y2 (-1.0), y4 (+0.9) |
| | Pattern 10 | y2 (-1.3), y3 (+0.7) |
| M-Additive 3 | Pattern 1 | y0 (+1.0) |
| | Pattern 2 | y1 (-0.6) |
| | Pattern 3 | y2 (+0.9) |
| | Pattern 4 | y3 (-1.2) |
| | Pattern 5 | y4 (+0.7) |
| | Pattern 6 | y0 (-0.5), y2 (+0.4) |
| | Pattern 7 | y1 (+1.4), y3 (-0.8) |
| | Pattern 8 | y0 (+0.3), y3 (-0.9), y4 (+0.1) |
| | Pattern 9 | y1 (-1.1), y2 (+0.6), y4 (-0.4) |
| | Pattern 10 | y2 (+1.5), y3 (-1.0) |

Table 13: Mean values of the results obtained for all the manually created datasets with the multiplicative pattern on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| **Metric** | **Associative Classifier** | **Triclustering-Based RF** | **Triclustering-Based XGBoost** |
|---|---|---|---|
| Accuracy | 0.738 ± 0.088 | 0.608 ± 0.077 | 0.598 ± 0.091 |
| Specificity | 0.720 ± 0.077 | 0.610 ± 0.145 | 0.575 ± 0.129 |
| Sensitivity | 0.755 ± 0.078 | 0.605 ± 0.180 | 0.620 ± 0.182 |

Table 14: Number of objects, features, and contexts for each tricluster generated in the manually created datasets incorporating only the multiplicative pattern in the context dimension.

| Dataset | Pattern | Feature |
|---|---|---|
| M-Multiplicative 0 | Pattern 1 | y0 (x1.2) |
| | Pattern 2 | y1 (x1.6) |
| | Pattern 3 | y2 (x1.1) |
| | Pattern 4 | y3 (x-1.1) |
| | Pattern 5 | y4 (x1.4) |
| | Pattern 6 | y0 (x1.2), y2 (x1.4) |
| | Pattern 7 | y1 (x-1.3), y3 (x-1.4) |
| | Pattern 8 | y0 (x1.1), y3 (x1.1), y4 (x1.2) |
| | Pattern 9 | y1 (x-1.1), y2 (x1.3), y4 (x1.2) |
| | Pattern 10 | y2 (x1.5), y3 (x-1.5) |
| M-Multiplicative 1 | Pattern 1 | y0 (x1.1) |
| | Pattern 2 | y1 (x1.3) |
| | Pattern 3 | y2 (x1.2) |
| | Pattern 4 | y3 (x-1.3) |
| | Pattern 5 | y4 (x1.2) |
| | Pattern 6 | y0 (x1.1), y2 (x1.1) |
| | Pattern 7 | y1 (x-1.2), y3 (x-1.4) |
| | Pattern 8 | y0 (x1.3), y3 (x1.2), y4 (x1.2) |
| | Pattern 9 | y1 (x-1.1), y2 (x1.4), y4 (x1.3) |
| | Pattern 10 | y2 (x1.2), y3 (x-1.4) |
| M-Multiplicative 2 | Pattern 1 | y0 (x1.2) |
| | Pattern 2 | y1 (x1.4) |
| | Pattern 3 | y2 (x1.1) |
| | Pattern 4 | y3 (x-1.3) |
| | Pattern 5 | y4 (x1.2) |
| | Pattern 6 | y1 (x1.2), y2 (x1.4) |
| | Pattern 7 | y4 (x-1.2), y3 (x-1.2) |
| | Pattern 8 | y1 (x1.2), y2 (x1.1), y4 (x1.4) |
| | Pattern 9 | y3 (x-1.1), y2 (x1.3), y4 (x1.2) |
| | Pattern 10 | y1 (x1.2), y2 (x-1.1) |
| M-Multiplicative 3 | Pattern 1 | y0 (x1.2) |
| | Pattern 2 | y1 (x1.3) |
| | Pattern 3 | y2 (x1.1) |
| | Pattern 4 | y3 (x-1.3) |
| | Pattern 5 | y4 (x1.2) |
| | Pattern 6 | y0 (x1.2), y3 (x1.3) |
| | Pattern 7 | y2 (x-1.2), y3 (x-1.1) |
| | Pattern 8 | y0 (x1.4), y1 (x1.2), y4 (x1.2) |
| | Pattern 9 | y1 (x-1.1), y2 (x1.3), y4 (x1.1) |
| | Pattern 10 | y1 (x1.2), y3 (x-1.1) |

Table 15: Mean values of the results obtained for all the manually created datasets with a mix of all patterns on the context dimensions. The cells highlihted in light green represents the higher results between each classifier.

| Metric | Associative Classifier | Triclustering-Based RF | Triclustering-Based XGBoost |
|---|---|---|---|
| Accuracy | 0.735 ± 0.041 | 0.662 ± 0.078 | 0.618 ± 0.082 |
| Specificity | 0.760 ± 0.052 | 0.660 ± 0.146 | 0.600 ± 0.149 |
| Sensitivity | 0.710 ± 0.043 | 0.665 ± 0.097 | 0.635 ± 0.148 |

Table 16: Number of objects, features, and contexts for each tricluster generated in the manually created datasets incorporating only mixed patterns in the context dimension.

| Dataset | Pattern | Feature |
|---|---|---|
| M-All 0 | Pattern 1 | y0 (+0) |
| | Pattern 2 | y1 (+0), y2 (+0) |
| | Pattern 3 | y3 (+1) |
| | Pattern 4 | y0 (+1), y4 (-1) |
| | Pattern 5 | y2 (x1.6) |
| | Pattern 6 | y1 (x1.3), y3 (x-1.3) |
| | Pattern 7 | y4 (+0), y0 (x1.5) |
| | Pattern 8 | y0 (+1.3), y1 (+1.8) |
| | Pattern 9 | y2 (+0), y3 (-0.7) |
| | Pattern 10 | y0 (+0), y4 (+0), y2 (+0.2), y3 (x1.2) |
| M-All 1 | Pattern 1 | y1 (+0) |
| | Pattern 2 | y0 (+0), y3 (+0) |
| | Pattern 3 | y4 (-1.5) |
| | Pattern 4 | y2 (+2), y3 (-1) |
| | Pattern 5 | y1 (x1.2) |
| | Pattern 6 | y0 (x1.3), y2 (x1.1) |
| | Pattern 7 | y4 (+0), y3 (x1.4) |
| | Pattern 8 | y0 (+1.1), y4 (+1.5) |
| | Pattern 9 | y2 (+0), y1 (-0.8) |
| | Pattern 10 | y3 (+0), y4 (+0), y1 (+0.5), y0 (x1.2) |
| M-All 2 | Pattern 1 | y0 (+0) |
| | Pattern 2 | y1 (+0), y3 (+0) |
| | Pattern 3 | y4 (+1.8) |
| | Pattern 4 | y1 (-1.2), y2 (+1.5) |
| | Pattern 5 | y3 (x1.3) |
| | Pattern 6 | y4 (x-1.2), y0 (x1.1) |
| | Pattern 7 | y2 (+0), y1 (x1.4) |
| | Pattern 8 | y3 (-1.5), y0 (+2) |
| | Pattern 9 | y0 (+0), y4 (+0.7) |
| | Pattern 10 | y2 (+0), y3 (+0), y1 (-0.5), y4 (x-1.3) |
| M-All 3 | Pattern 1 | y3 (+0) |
| | Pattern 2 | y2 (+0), y4 (+0) |
| | Pattern 3 | y1 (+1.2) |
| | Pattern 4 | y0 (-0.8), y3 (+1.5) |
| | Pattern 5 | y4 (x1.1) |
| | Pattern 6 | y2 (x-1.3), y1 (x1.4) |
| | Pattern 7 | y0 (+0), y3 (x1.2) |
| | Pattern 8 | y2 (+2), y4 (+0.5) |
| | Pattern 9 | y1 (+0), y3 (-1.1) |
| | Pattern 10 | y1 (+0), y4 (+0), y0 (+0.3), y2 (x-1.1) |