The Journal of Open Source Software

NiaARM.jl: A Julia Framework for Numerical Association Rule Mining Using Nature-Inspired Optimization Algorithms

Žiga Stupan[1], **Tilen Hliš**[1], **and Iztok Fister Jr.**[1¶]

1 University of Maribor, Faculty of Electrical Engineering and Computer Science ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Mehmet Hakan Satman

Reviewers:

- @LaurenzBeck
- @Kiwy3

Submitted: 18 December 2025
Published: unpublished

Summary

NiaARM.jl is an open-source Julia package for Numerical Association Rule Mining (NARM) based on stochastic population-based nature-inspired optimization algorithms (Telikani et al., 2020). It brings the capabilities of the original Python-based NiaARM framework (Stupan & Fister, 2022) to the Julia ecosystem, enabling researchers and data scientists working with datasets with mixed attribute types (consisting of categorical and numerical attributes) to discover numerical association rules. NiaARM.jl supports loading datasets, preprocessing, association rule mining, and extraction of discovered rules with associated interestingness metrics. In line with the rule mining part, this package also implements several well-known stochastic population-based nature-inspired algorithms, such as Differential Evolution (DE) (Storn & Price, 1997), Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), and several other metaphor-based nature-inspired algorithms to act as solvers for the numerical association rule mining task. The entire numerical association rule mining workflow is further supported by visualization methods for numerical association rules, which is achieved through NarmViz.jl, a package well integrated with NiaARM.jl (Fister Jr et al., 2024).

State of the Field

Classical association rule mining (ARM) is well supported in general purpose libraries such as the R package *arules* (Hahsler et al., 2005), the SPMF Java library (Fournier-Viger & others, 2020), and the Julia package RuleMiner.jl (Schwartz, 2024). These tools focus on frequent itemset and rule mining over categorical or discretized data. Frameworks such as uARMSolver (Fister & Fister Jr, 2020) go a step further by formulating ARM as an optimization problem and supporting numerical attributes.

Dedicated support for NARM is currently provided mainly by the original Python based NiaARM framework (Stupan & Fister, 2022) and the more recent *niarules* package for R (Fister Jr et al., 2026). Both adopt population-based nature-inspired algorithms to search for numerical association rules.

NiaARM.jl is designed as a Julia native, modular re-implementation of this line of work rather than a wrapper around existing frameworks. It combines the full NARM pipeline (data handling, problem formulation, optimization, and rule representation) with extensibility for new interestingness measures and optimization algorithms, and integrates tightly with NarmViz.jl for visualization. In this way, NiaARM.jl complements existing Python and R frameworks while filling a gap in the Julia ecosystem for high performance NARM.

Software Design

The core of NiaARM.jl is organized around four main modules: (1) the `Dataset` module, which handles loading and preprocessing of transaction data from CSV files or DataFrames, representing numerical attributes as interval features and categorical attributes as sets of categories; (2) the `Problem` module, which formulates numerical association rule mining as a continuous optimization problem, where candidate solutions are encoded as real-valued vectors and decoded into association rules; (3) the optimization algorithms module, which provides implementations of various nature-inspired algorithms that explore the search space to identify rules; and (4) the `Rule` module, which represents discovered association rules and provides methods for computing interestingness measures.

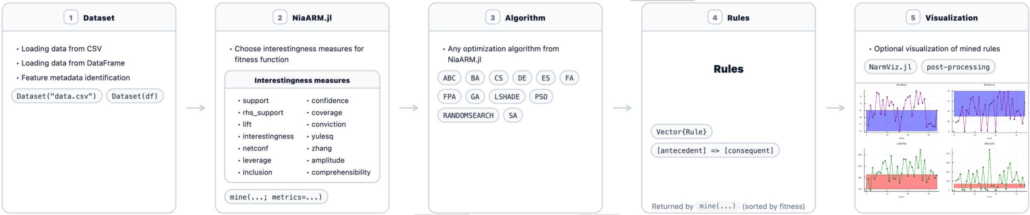

Figure 1: NiaARM.jl flow.

The flow of the NiaARM.jl framework is shown in Figure 1. Users can construct a dataset either from a CSV file (via a file path) or directly from `DataFrame`. The dataset is then used to build the numerical association rule mining optimization problem together with user-selected interestingness measures, which are used in the computation of the fitness function. The optimization problem can be solved using any of the population-based nature-inspired algorithms implemented in NiaARM.jl (e.g., DE, ABC, PSO) to mine numerical association rules from the dataset. The discovered rules are returned as a collection of `Rule` objects and can be further analyzed, exported (e.g., to CSV in downstream post-processing), or visualized using NarmViz.jl (Fister Jr et al., 2024), which provides several visualization methods for numerical association rules.

Statement of need

Despite the growing importance of NARM in domains such as finance, sport, and medicine, the Julia ecosystem has lacked a dedicated and efficient framework for this task. NiaARM.jl fills this gap by providing a comprehensive, optimization-driven approach to numerical association rule mining based on stochastic population-based nature-inspired algorithms. The package enables researchers and data scientists to mine numerical association rules from mixed-type datasets while leveraging Julia's strengths in performance, composability, and scientific computing. Altogether, the main benefits of NiaARM.jl can be summarized as follows:

- The framework enables researchers to easily apply the full NARM pipeline, i.e. from dataset loading to visualization of the identified rules.
- The package contains a vast collection of stochastic population-based nature-inspired algorithms.
- Julia's performance allows for significantly faster discovery of numerical association rules compared to the Python version.
- The framework is designed in a modular way, allowing components to be flexibly combined and extended.

Research Impact Statement

NiaARM.jl framework lowers the barrier to high-performance NARM by making NARM accessible, faster than previous implementations especially on complex datasets, and extensible within the Julia ecosystem. By providing a dedicated Julia framework for dealing with NARM tasks, the package extends an established methodology that was previously available primarily in the Python programming language. It allows researchers and practitioners who rely on Julia for high-performance scientific computing to solve NARM tasks. NiaARM.jl is also well integrated with other packages, for example NarmViz, which extends the rule mining pipeline to include visualization.

In addition to its applied impact, NiaARM.jl contributes to methodological research by providing reference implementations of multiple stochastic population-based nature-inspired algorithms within a unified framework. This supports reproducibility, benchmarking, and comparative studies in metaheuristic optimization beyond solving NARM tasks alone. The modular design of the framework encourages community-driven extensions, enabling rapid prototyping of new algorithms, fitness functions, and interestingness measures.

AI Usage Disclosure

During the preparation of this work the authors used language tools such as Lumo (the AI assistant from Proton), Grammarly in order to improve the article's readability. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article. During codebase development, AI-assisted tools were used for documentation refinement and code readability. The majority of the codebase had been designed and implemented prior to the widespread adoption of AI-assisted development tools, ensuring that the conceptual design, architectural decisions, and core algorithmic implementations are entirely the result of the authors original work.

References

Fister, I., & Fister Jr, I. (2020). Uarmsolver: A framework for association rule mining. *arXiv Preprint arXiv:2010.10884*.

Fister Jr, I., Emsenhuber, G., Plümer, J. H., Fister, I., & Holzinger, A. (2026). Niarules: Advancing interpretable machine learning through numerical association rule mining and 3D coral plot visualization. *SoftwareX*, *33*, 102470.

Fister Jr, I., Fister, I., Podgorelec, V., Salcedo-Sanz, S., & Holzinger, A. (2024). NarmViz: A novel method for visualization of time series numerical association rules for smart agriculture. *Expert Systems*, *41*(3), e13503. https://doi.org/10.1111/exsy.13503

Fournier-Viger, P., & others. (2020). Spmf: A java open-source data mining library. *URL: Www. Philippe-Fournier-Viger. Com/Spmf/(visited on 08/15/2018)*.

Hahsler, M., Grün, B., & Hornik, K. (2005). Arules - a computational environment for mining association rules and frequent item sets. *Journal of Statistical Software*, *14*(15), 1–25. https://doi.org/10.18637/jss.v014.i15

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*(3), 459–471. https://doi.org/10.1007/s10898-007-9149-x

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95-International Conference on Neural Networks*, *4*, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Schwartz, J. (2024). *RuleMiner.jl: Efficient frequent itemset and association rule mining in* *julia*. https://github.com/JaredSchwartz/RuleMiner.jl.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359. https://doi.org/10.1023/A:1008202821328

Stupan, Ž., & Fister, I. (2022). Niaarm: A minimalistic framework for numerical association rule mining. *Journal of Open Source Software*, *7*(77), 4448. https://doi.org/10.21105/joss.04448

Telikani, A., Gandomi, A. H., & Shahbahrami, A. (2020). A survey of evolutionary computation for association rule mining. *Information Sciences*, *524*, 318–352. https://doi.org/10.1016/j.ins.2020.02.073