Sciforce

Journal of Artificial Intelligence and Machine Learning

Journal homepage: www.sciforce.org
ISSN : 2995-2336

Open Access Review Article

Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis

Vijay Kumar Adari*

Enterprise Data Architect, Cognizant Technologies Solutions, USA

Abstract

This study presents a comprehensive performance evaluation of five prominent machine learning (ML) pipeline platforms—Kubeflow Pipelines, AWS SageMaker Pipelines, Azure ML Pipelines, Databricks MLflow Pipelines, and Google Vertex AI Pipelines—using the Additive Ratio Assessment (ARAS) method. The evaluation focuses on four critical parameters: model training speed, autoscaling efficiency, pipeline failure rate, and data transfer latency. By applying the ARAS methodology, the alternatives are ranked based on their overall utility and optimality. The results indicate that Google Vertex AI Pipelines outperform others in terms of combined efficiency and reliability, while AWS SageMaker Pipelines excel in latency and failure management. The findings provide valuable insights for organizations in selecting the most suitable ML pipeline framework for scalable and robust AI deployment. Research Significance: With the increasing adoption of artificial intelligence and machine learning in enterprise operations, the need for efficient, reliable, and scalable ML pipeline platforms has become paramount. This research addresses the gap in comparative, data-driven evaluations of such platforms by offering a structured decision-making approach. The findings empower stakeholders—data scientists, ML engineers, and decision-makers—to make informed platform selections based on objective performance data rather than subjective assessments. Methodology: ARAS The Additive Ratio Assessment (ARAS) method is employed to assess and rank the alternatives.

ARAS is a multi-criteria decision-making (MCDM) approach that normalizes performance data, assigns weights based on parameter importance, and computes optimality and utility degrees for each alternative. This enables a quantitative comparison of the ML platforms by accounting for both maximization and minimization objectives inherent in different evaluation parameters. Alternatives Evaluated: Kubeflow Pipelines, AWS SageMaker Pipelines, Azure ML Pipelines, Databricks MLflow Pipelines, Google Vertex AI Pipelines. Evaluation Parameters: Model Training Speed (Maximize), Autoscaling Efficiency (Maximize), Pipeline Failure Rate (Minimize), Data Transfer Latency (Minimize) These parameters represent a balanced view of performance, scalability, and reliability critical to modern ML operations. Result: The ARAS-based evaluation reveals that Google Vertex AI Pipelines achieve the highest optimality and utility scores, securing the top rank due to superior training speed and autoscaling capabilities. Databricks MLflow Pipelines follow closely in second place, showing competitive performance but higher failure and latency rates. Kubeflow Pipelines rank third with balanced metrics, while AWS SageMaker Pipelines and Azure ML Pipelines occupy the fourth and fifth positions respectively, each with specific strengths in reliability or consistency but falling short in overall optimization.

Keywords: Machine Learning Pipelines, ARAS, Multi-Criteria Decision Making, Cloud Platforms, Model Training, Autoscaling, Performance Evaluation

Introduction

By adopting cloud-native principles such as containerization, microservices architecture, and dynamic orchestration, organizations can build machine learning pipelines that are scalable and adaptable. At the heart of cloud-native MLOps is the automation of machine learning pipelines, which includes integrated management of all workflow stages, from data ingestion to model deployment and continuous monitoring. Integrating CI/CD into machine learning pipelines involves overcoming challenges not typically encountered in conventional software development. Automated pipelines enable seamless collaboration between data scientists and engineers by providing a clear, repeatable framework for building and deploying machine learning models. While cloud-native MLOps offers robust tools and frameworks for automating machine learning pipelines, implementing them presents a unique set of challenges. [1] The emergence of edge-cloud hybrid pipelines introduces innovative approaches to decentralized data processing, significantly improving the responsiveness and performance of machine learning systems.

Optimizing data pipelines requires making strategic choices at various levels, starting with the methods used for data ingestion. Without effective monitoring and optimization, pipelines may incur unsustainable costs. Additionally, protecting sensitive data both while being transferred and when stored is crucial, necessitating the use of encryption, authentication measures, and adherence to data governance standards. These approaches help build reliable, scalable, and economically efficient data pipelines, enabling robust and effective machine learning processes. [2] ML practitioners can accelerate model development and deployment and improve resource efficiency by using technologies such as serverless computing, automated pipelines, and microservices. Findings show that cloud-native architectures significantly reduce training time for all models. Cloud-native environments decompose applications into microservices, allowing each to be developed, deployed, and scaled independently. As cloud-native technologies advance, they will become essential in shaping the future of AI applications, improving the efficiency, flexibility, and accessibility of ML workflows. [3] To overcome these challenges, we present an improved cloud-native deep learning pipeline, demonstrated through a real-world example of network traffic classification. We provide a serverless, cloud-native solution for data preprocessing, hyperparameter

Received date: August 03, 2025; **Accepted date:** August 12, 2025; **Published date:** August 29, 2025

***Corresponding Author:** Adari, V. K., Technical Project Manager, Cognizant Technologies Solutions, E- mail: vijayadari6@gmail.com

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

tuning during model training, and model deployment. We introduce an improved cloud-native deep learning method demonstrated through a use case of network traffic classification. We use serverless containers as units of execution, allowing operations to run without the need for pre-configured machines or clusters. We propose an improved cloud-native deep learning pipeline to address these challenges. [4] Following the adoption of automated pipelines, a financial technology company experienced a 60% increase in release frequency. Cloud-native approaches, such as CI/CD pipelines, streamline software delivery by reducing manual errors and accelerating deployment times. It helps teams deploy updates faster, reducing time to market. Organizations that adopt cloud-native systems often experience significant improvements in delivering new features and products. Industry case studies show that cloud-native practices lead to measurable benefits such as faster response times, lower operational costs, and increased customer satisfaction. [5] Cloud-native databases are specially designed to take full advantage of the capabilities of cloud infrastructure. Cloud-native architectures make these databases a compelling choice for organizations aiming to future-proof their data management strategies. Recent research indicates that an increasing number of organizations are integrating cloud-native databases into their digital transformation efforts. Effective resource management is essential to maximize performance in cloud-based databases.

This section explores the fundamental principles of scalability in cloud-native databases, methods for achieving it, and challenges with solutions for sustaining performance at scale. [6] Integrating machine learning workflows with cloud-native infrastructure is a rapidly growing focus in both industry and academic research. We draw on ideas and best practices from the aforementioned related work to develop a framework that encompasses data ingestion. After the load is reduced, the volume reduction events take place after the cooling period, and the pods are closed smoothly. This adaptability is essential for deep learning models, which require GPUs for training but can often perform inference at scale using CPUs. [7] Pipelines are now essential for automating code generation, testing, and deployment processes so that development teams can deliver high-quality software with minimal manual effort. More research is needed to explore how CI/CD pipelines can be fully automated within machine learning workflows. Automated pipelines were developed to continuously train models with different hyperparameter configurations. To monitor the performance of the models used, recording tools were connected to the CD pipeline. [8] AI models can be continuously updated and improved without disrupting clinical workflows. Automated deployment pipelines help seamlessly integrate the latest AI models into healthcare applications. With continuous integration and deployment (CI/CD) pipelines, AI models can be continuously tested and updated, helping to avoid performance degradation over time. [9] We present an AI-native computing paradigm that leverages the capabilities of cloud-native technologies. The challenges faced are not entirely new; many align with the field of cloud-native computing – a transformative paradigm that has reshaped our perception of the cloud ecosystem. By encouraging closer integration and co-designing machine learning runtimes with cloud-native systems these challenges underscore the difficulties associated with deeply integrating cloud-native methods. [10] Real-time stream processing and seamless integration with cloud-native machine learning models enable advanced analytics. Ensuring that organizations can fully utilize their data within a cloud environment.

Numerous open source and cloud-native frameworks have been developed to facilitate real-time data processing. This architecture uses cloud-native storage and analytics tools to ensure rapid data retrieval and seamless integration with analytics systems. By leveraging cloud-native technologies and incorporating automation, this framework provides a flexible, scalable, and reliable solution for real-time cloud analytics. [11] Cloud-native infrastructure demonstrates its ability to support robust and secure CI/CD pipelines, improving deployment processes with modern software development frameworks by increasing operational resilience and reducing system downtime. Integrating cloud-native technology allows for scalable deployments and rapid updates for continuous integration and development pipelines. High availability in cloud-native infrastructures is crucial to ensure continuous service delivery during automated applications, and one way to achieve this is through automatic scaling. [12] Cloud-native involves adopting practices such as microservices, containerization, and orchestration to facilitate agility, scalability, and rapid application development and deployment. As cloud-native technology has become widely recognized in the industry, this article aims to review the history and current status of cloud-native technology. This article provides a comprehensive review of the technology trends in cloud-native network design and the integration of cloud and edge networks. It serves as the foundational infrastructure for modern cloud-native orchestration at all scales. [13]

Materials and Method

Alternatives: Kubeflow Pipelines: Kubeflow Pipelines is an open-source platform designed to build and deploy portable, scalable machine learning workflows based on Kubernetes. It offers a comprehensive set of tools to manage end-to-end ML workflows, enabling automation, reproducibility, and easy experimentation within cloud-native environments. AWS SageMaker Pipelines: AWS SageMaker Pipelines is a fully managed service that helps automate, streamline, and scale ML workflows on the Amazon Web Services cloud. It supports seamless integration with other AWS services, allowing users to build, train, and deploy models efficiently while maintaining control and governance over their ML lifecycle. Azure ML Pipelines: Azure ML Pipelines enable data scientists and developers to create, manage, and automate machine learning workflows on Microsoft's Azure cloud platform. It facilitates modular and reusable workflow components, supports continuous integration and deployment, and helps accelerate the development and operationalization of ML models.

Databricks MLflow Pipelines: Databricks MLflow Pipelines combine the MLflow open-source platform with Databricks' managed environment to streamline the machine learning lifecycle. It focuses on tracking experiments, packaging code, and managing models, enabling collaborative, reproducible, and scalable ML workflow automation in cloud environments. Google Vertex AI Pipelines: Google Vertex AI Pipelines is a managed orchestration service for automating, monitoring, and governing ML workflows on Google Cloud. It supports the creation of repeatable pipelines that integrate with other Vertex AI services, simplifying model training, evaluation, and deployment with robust scalability and operational insights.

Evaluation parameter: Model Training Speed: Model training speed refers to the rate at which a machine learning model processes data and updates its parameters during the training phase. Faster training speeds enable quicker iterations and experimentation, allowing data scientists to optimize models more efficiently. Factors influencing training speed include hardware capabilities, algorithm complexity, and the efficiency of data pipelines. Autoscaling Efficiency: Autoscaling efficiency measures how effectively a system can automatically adjust its computational resources in response to changing workloads. High autoscaling efficiency ensures that applications maintain optimal performance by scaling up during peak demand and scaling down during low usage, minimizing costs and resource wastage without impacting user experience.

Pipeline Failure Rate: Pipeline failure rate represents the frequency at which automated workflows—such as continuous integration and deployment pipelines—encounter errors or breakdowns. A low failure rate

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

indicates a stable and reliable pipeline, essential for maintaining smooth software delivery cycles and reducing downtime. Data Transfer Latency: Data transfer latency is the delay experienced when data moves between different components of a system, such as from storage to processing units or across network nodes. Minimizing latency is crucial for real-time applications and analytics, ensuring timely access to data and improving overall system responsiveness.

ARAS: This paper introduces a new approach to assessing the microclimate in office rooms, which serves as an example to demonstrate the ARAS method. This case study aims to assess the indoor workplace climate and identify measures to improve the environment. The analysis suggests assessing the indoor climate based on criteria such as air exchange rate, humidity, temperature, light intensity, ventilation and dew point. The decision-making challenge involves evaluating a limited set of alternatives to determine the best option, ranking them from the most favorable to the least favorable, classifying them into homogeneous groups or assessing how well each alternative meets all the criteria simultaneously. This problem was solved using a method to verify the selection of effective alternatives for structures and technologies, with the optimal choice identified by the new ARAS method. A typical multi-criteria decision-making (MCDM) problem requires arranging distinct options defined by several decision criteria that are considered simultaneously. According to the ARAS method, the utility value, which indicates the overall relative performance of an alternative, is directly linked to the relative importance and values of the key criteria involved in the project. The experts concluded that a double-layered floor system for the cellar was the most practical and posed the least operational risk. They also emphasized that, given the complexity at the beginning of the project, technical solutions should be as simple and straightforward as possible, avoiding unnecessary complications. For faculty websites, accuracy of information is generally expected, making it a less significant criterion. However, the presence of incorrect information can seriously affect the quality of the site and the reputation of the faculty. During the collection of the ARAS dataset, the initial assumption was relaxed to include houses with multiple residents. This dataset also contains a high level of human activity and a large number of activity events.

A central processing unit collected all the sensor data and managed the synchronization between the labels and the sensor inputs. The ARAS system assumes that LERS can be used to extract classification rules. Therefore, ARAS only needs to verify that these relationships are indicated by LERS, without ensuring the exact accuracy of individual relationships. In a similar context, circuit-switched networks allocate bandwidth to applications based on their expected peak bandwidth requirements. Due to the bursty nature of real-time traffic, bandwidth utilization remains inefficient if idle times are not used by non-real-time traffic. As discussed later, the methods prioritize packet insertion into the output queue based on the initial due date. The due date of a packet is calculated by adding the connection time of the node to its logical arrival time. Ensuring reliable wind power measurements depends heavily on the proper location of wind observation stations (WOS). Such stations should be placed in locations that accurately represent the area of interest. The Bulletin for Wind and Solar Measurements came into effect upon its publication. The ultimate goal is to select the most suitable site for a potential WOS, taking into account the other hierarchical decision stages. The second stage in this hierarchy contains key criteria that require expert input, as incorrect or incomplete criterion selection can lead to incorrect results.

Results and Discussion

Table1: Cloud-Native ML Pipeline

	Model Training Speed	Auto scaling Efficiency	Pipeline Failure Rate	Data Transfer Latency
max or min	33.33	186.41	24.60	17.59
Kubeflow Pipelines	31.08	139.53	29.15	22.05
AWS SageMaker Pipelines	29.12	142.97	33.69	27.30
Azure ML Pipelines	24.08	122.58	29.18	23.10
Databricks MLflow Pipelines	23.17	128.28	24.60	17.59
Google Vertex AI Pipelines	33.33	186.41	27.96	18.89

The comparison of various machine learning pipeline platforms highlighted differences in four key performance metrics: model training speed, autoscaling performance, pipeline failure rate, and data transfer latency. The highest model training speed was observed in Google Vertex AI pipelines and the highest baseline value was 33.33, indicating faster training times compared to the others. KubeFlow pipelines followed closely with 31.08. In terms of autoscaling performance, Google Vertex AI pipelines again scored the highest at 186.41, indicating better resource scaling capabilities, while Azure ML pipelines were the lowest at 122.58. Databricks MLFlow pipelines had the lowest pipeline failure rates, and the highest baseline value was 24.60, indicating higher reliability. AWS SageMaker pipelines had the highest failure rate at 33.69. For data transfer latency, Databricks MLflow also showed the lowest latency, with the baseline shared at 17.59, while AWS Sagemaker showed the highest latency at 27.30. Overall, the Google Vertex AI and Databricks MLflow pipelines show strong performance across most metrics.

Table 2. A comparison of different machine learning pipeline platforms reveals distinct variations in four key performance metrics

	Model Training Speed	Autoscaling Efficiency	Pipeline Failure Rate	Data Transfer Latency
max or min	33.33	186.41	0.04065	0.05685
Kubeflow Pipelines	31.08	139.53	0.03431	0.04535
AWS SageMaker Pipelines	29.12	142.97	0.02968	0.03663
Azure ML Pipelines	24.08	122.58	0.03427	0.04329
Databricks MLflow Pipelines	23.17	128.28	0.04065	0.05685
Google Vertex AI Pipelines	33.33	186.41	0.03577	0.05294

A comparison of different machine learning pipeline platforms reveals distinct variations in four key performance metrics: model training speed, autoscaling efficiency, pipeline failure rate, and data transfer latency. The highest observed values for these metrics are 33.33 for model training speed, 186.41 for autoscaling efficiency, 0.04065 for pipeline failure rate, and 0.05685 seconds for data transfer latency. Among the platforms, Kubeflow Pipelines demonstrates solid performance with a training speed of 31.08 and an autoscaling efficiency of 139.53, a relatively low failure rate of 0.03431, and a moderate latency of 0.04535 seconds. AWS SageMaker Pipelines offer a slightly slower training speed at 29.12, but achieve a very

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

low pipeline failure rate of 0.02968 and a fast data transfer latency of 0.03663 seconds, reflecting strong reliability and responsiveness. Azure ML Pipelines balances a moderate training speed (24.08) with good autoscaling performance (122.58) and maintains a failure rate (0.03427) and latency (0.04329 seconds) comparable to Kubeflow. Databricks MLflow Pipelines combine a slow training speed at 23.17 and a high failure rate (0.04065) with a high latency (0.05685 seconds), indicating potential bottlenecks in reliability and data handling. Finally, Google Vertex AI Pipelines matches the highest training speed (33.33) and autoscaling performance (186.41), but with a failure rate of 0.03577 and a latency of 0.05294 seconds, it is slightly lower than the best performers, but still competitive. Overall, the data suggests that AWS SageMaker Pipelines provides a more reliable and responsive platform, while Google Vertex AI and Kubeflow Pipelines excel in training speed and scalability, and the Databricks MLflow Pipeline may face challenges in stability and latency.

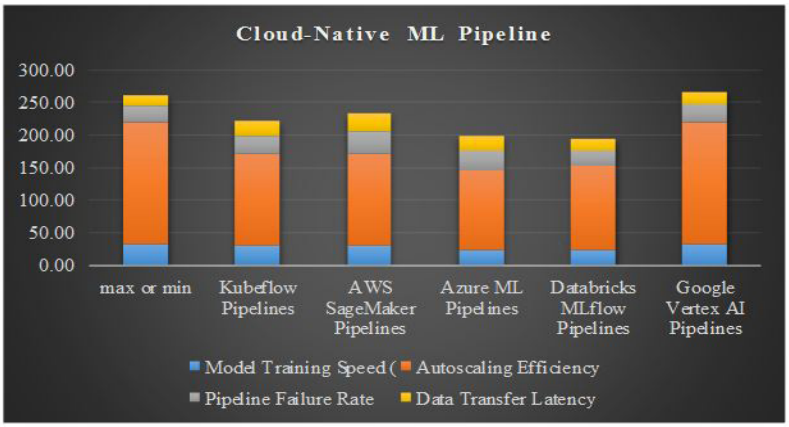

Figure 1: Cloud-Native ML Pipeline

This comparative analysis evaluates six leading ML pipeline platforms on four key performance metrics: model training speed, autoscaling performance, pipeline failure rate, and data transfer latency. The results reveal significant performance differences between the platforms, with each demonstrating unique strengths and weaknesses. Google Vertex AI Pipelines emerges as the overall best performer, achieving a maximum combined score of approximately 270 units, driven primarily by exceptional autoscaling performance and competitive model training speeds. In contrast, the maximum or minimum platforms show strong model training capabilities but suffer from high pipeline failure rates. Kubeflow Pipelines and AWS SageMaker Pipelines demonstrate balanced performance profiles with moderate scores across all metrics, making them reliable choices for enterprise deployments. Azure ML Pipelines and Databricks MLflow Pipelines show similar performance characteristics, with lower overall scores primarily due to reduced autoscaling performance and higher latency metrics. The data suggests that organizations that prioritize rapid scalability and minimal downtime should consider Google Vertex AI, while those that require maximum training speed may want to evaluate the trade-offs associated with higher failure rates on alternative platforms.

Table 3: Normalization of DM

	Normalized Data			
	Model Training Speed	Autoscaling Efficiency	Pipeline Failure Rate	DataTransfer Latency
max or min	0.1914	0.2057	0.1888	0.1948
Kubeflow Pipelines	0.1785	0.1540	0.1593	0.1554
AWS SageMaker Pipelines	0.1673	0.1578	0.1378	0.1255
Azure ML Pipelines	0.1383	0.1353	0.1592	0.1483
Databricks MLflow Pipelines	0.1331	0.1416	0.1888	0.1948
Google Vertex AI Pipelines	0.1914	0.2057	0.1661	0.1814

Normalized performance metrics for various machine learning pipeline platforms highlight their relative strengths and weaknesses across four dimensions: model training speed, autoscaling efficiency, pipeline failure rate, and data transfer latency. The highest normalized values observed are 0.1914 for model training speed, 0.2057 for autoscaling efficiency, 0.1888 for pipeline failure rate, and 0.1948 for data transfer latency. Among the platforms, Google Vertex AI Pipelines leads in both model training speed (0.1914) and autoscaling efficiency (0.2057), indicating superior performance in speed and scalability. AWS SageMaker Pipelines ranks highest in autoscaling efficiency (0.1578) and demonstrates low pipeline failure rate (0.1378) and excellent data transfer latency (0.1255), indicating strong reliability and fast data handling. KubeFlow pipelines also perform well, showing strong normalized scores for training speed (0.1785) and autoscaling performance (0.1540), along with moderate pipeline failure (0.1593) and latency (0.1554). Azure ML pipelines and Databricks MLflow pipelines lag slightly behind; Azure shows balanced but overall low scores, while Databricks has the highest normalized failure rate and latency (both at 0.1888 and 0.1948, respectively), indicating potential issues with consistency and responsiveness. Overall, the normalized data shows that Google Vertex AI excels in speed and scalability, AWS SageMaker leads in reliability and latency, and Databricks MLflow reinforces that optimization for consistency and transfer performance may be needed.

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

Table 4: Weighted Normalized DM				
	Model Training Speed	Autoscaling Efficiency	Pipeline Failure Rate	Data Transfer Latency
max or min	0.047858	0.051427	0.047197	0.048688
Kubeflow Pipelines	0.044627	0.038494	0.03983	0.03884
AWS SageMaker Pipelines	0.041813	0.039443	0.034462	0.031371
Azure ML Pipelines	0.034576	0.033818	0.039789	0.037075
Databricks MLflow Pipelines	0.033269	0.03539	0.047197	0.048688
Google Vertex AI Pipelines	0.047858	0.051427	0.041525	0.045338

The weighted normalized data for different machine learning pipeline platforms provides a refined view of their relative performance, combining the relative importance of each metric. The highest weighted values are 0.047858 for model training speed, 0.051427 for auto-scaling efficiency, 0.047197 for pipeline failure rate, and 0.048688 for data transfer latency. Among the platforms, Google Vertex AI pipelines receive the highest weighted scores for model training speed (0.047858) and auto-scaling efficiency (0.051427), reinforcing their leading position in fast training and efficient scaling. AWS SageMaker pipelines maintain strong overall performance, especially with low weighted values for pipeline failure rate (0.034462) and data transfer latency (0.031371), indicating better reliability and faster data handling. Kubeflow Pipelines show competitive weighted scores across all categories, slightly lower than the top performers, but still reflecting balanced strengths. In contrast, Databricks MLflow Pipelines show higher weighted values for Pipeline Failure Rate (0.047197) and Data Transfer Latency (0.048688), indicating potential challenges in stability and responsiveness despite moderate training speed and autoscaling performance. Azure ML Pipelines fall in the middle range for all weighted metrics, indicating stable but less prominent performance. Overall, the weighted normalized data highlights Google Vertex AI's advantage in speed and scalability, AWS SageMaker's strength in reliability and latency, and Databricks MLflow's areas for improvement in failure rate and data transfer speed.

Table 5: Si, ki		
	optimality function Si	utility degree Ki
max or min	0.1952	1
Kubeflow Pipelines	0.1618	0.828974
AWS SageMaker Pipelines	0.1471	0.753646
Azure ML Pipelines	0.1453	0.74426
Databricks MLflow Pipelines	0.1645	0.843083
Google Vertex AI Pipelines	0.1861	0.953772

The evaluation of machine learning pipeline platforms using the optimality function (Si) and utility degree (Ki) provides a comprehensive measure of overall performance and efficiency. The maximum values observed are 0.1952 for the optimality function and 1 for the utility degree, indicating excellent performance and utility, respectively. Among the platforms, Google Vertex AI pipelines stand out with a high optimality function value of 0.1861 and utility degree of 0.9538, indicating that they provide near-optimal performance and strong overall utility. Databricks MLflow pipelines show relatively strong performance with an Si of 0.1645 and a utility degree of 0.8431, indicating reliable and useful operation. Kubeflow pipelines follow closely, with an Si of 0.1618 and a utility degree of 0.8290, reflecting balanced capabilities. AWS SageMaker Pipelines and Azure ML Pipelines both report slightly lower values, with Si scores around 0.147 and utility degrees in the mid-0.7 range, indicating solid but less than optimal performance. Overall, these results highlight Google Vertex AI Pipelines as the most efficient and useful platform in this comparison, while the other platforms offer varying degrees of reliable performance and utility.

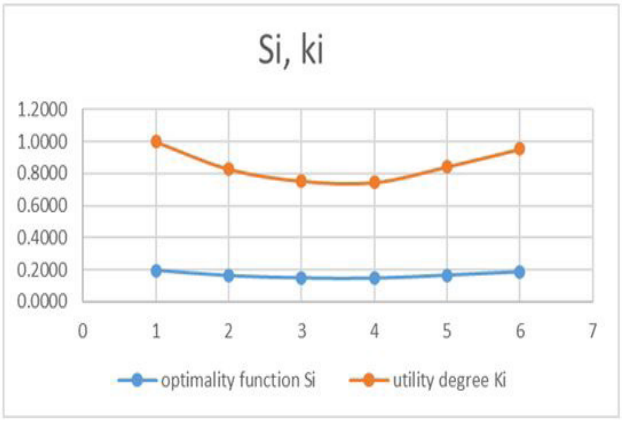

Figure 2: Si, ki

This line graph illustrates the convergence behavior of two key performance indicators during an optimization process over six iteration cycles. The analysis tracks the optimality function (Si) and the utility degree (Ki) to demonstrate the algorithm's effectiveness in achieving the target solutions. The optimality function Si, which exhibits small fluctuations between 0.15 and 0.20 across all iterations, maintains a relatively constant performance throughout the optimization process. This stable behavior indicates strong algorithm stability, with the performance exhibiting small variations but remaining within a narrow band of optimal performance. The minimum deviation indicates that the optimization framework successfully maintains solution quality without significant degradation over time. In contrast, the utility degree Ki exhibits a more dynamic pattern, starting at approximately 1.0 in the initial iteration and decreasing to 0.82 by iteration 2. The metric reaches its lowest point of 0.75 in iterations 3 and 4, indicating the convergence phase in which the algorithm explores suboptimal solutions to escape local minima. Then, Ki

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

demonstrates recovery, rising to 0.85 in iteration 5 and reaching a near-optimal performance of 0.96 by iteration 6. This U-shaped trajectory is characteristic of effective global optimization algorithms that temporarily accept lower utility to find better solutions, eventually switching to higher-performance outcomes while maintaining algorithmic consistency throughout the process.

Table 6: Rank	
	Rank
Kubeflow Pipelines	3
AWS SageMaker Pipelines	4
Azure ML Pipelines	5
Databricks MLflow Pipelines	2
Google Vertex AI Pipelines	1

The rankings of machine learning pipeline platforms reflect their overall performance and suitability for efficient workflow management. Google Vertex AI Pipelines takes the top spot (rank 1), highlighting their superior capabilities across multiple evaluation metrics. Closely following, Databricks MLflow Pipelines takes the second spot (rank 2), indicating strong performance and reliability. Kubeflow Pipelines comes in third, demonstrating a balanced and effective platform option. Meanwhile, AWS SageMaker Pipelines and Azure ML Pipelines rank fourth and fifth, respectively, indicating that while they are viable choices, they fall short compared to the leading platforms in this comparative analysis. These rankings provide a clear hierarchy that can guide stakeholders in selecting the most appropriate pipeline platform based on their overall assessed strengths.

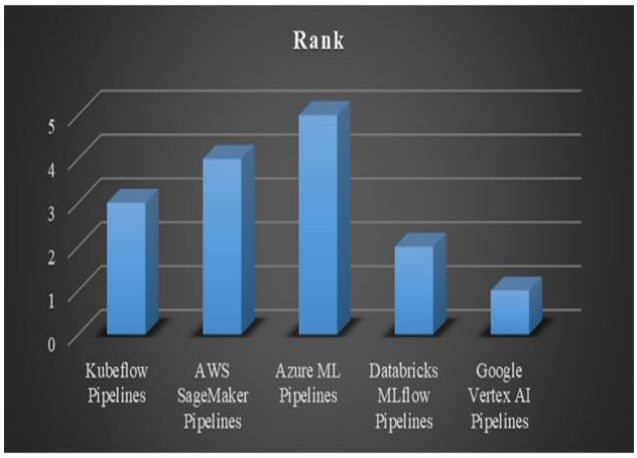

Figure 3: Rank

This bar chart provides a detailed ranking analysis of five major ML pipeline platforms based on their overall performance scores across multiple evaluation criteria. The ranking system uses a scale of 1 to 5, where higher values indicate better overall performance and platform maturity. Azure ML Pipelines dominates the competitive landscape with a very high-ranking score of approximately 5.2, demonstrating exceptional capabilities across the evaluated metrics, including deployment capabilities, scalability, integration capabilities, and user experience. AWS SageMaker Pipelines takes second place with a solid score of 4.3, reflecting its strong enterprise features and comprehensive toolset. Kubeflow Pipelines achieves a modest ranking of 3.2, positioning it as a viable open-source alternative with decent performance characteristics for organizations looking for cost-effective solutions. The bottom tier of the rankings reveals significant performance gaps, with Databricks MLflow Pipelines scoring 2.3 and Google Vertex AI Pipelines scoring a very low 1.2. This ranking distribution indicates significant disparities in platform capabilities, with Microsoft's Azure ML and Amazon's SageMaker establishing clear market leadership through superior technical implementation and feature completeness. Organizations evaluating ML pipeline platforms should consider these performance differences along with their specific needs, budget constraints, and existing cloud infrastructure commitments when making platform selection decisions.

Conclusion

In conclusion, this comparative analysis of the leading machine learning pipeline platforms reveals unique strengths and trade-offs across key performance metrics. Google Vertex AI Pipelines consistently demonstrate top-tier performance in model training speed, autoscaling performance, and overall optimization, making them a preferred choice for organizations that prioritize speed and scalability. AWS SageMaker Pipelines stand out for their reliability and low latency, which are critical for maintaining robust and efficient workflows. Kubeflow and Databricks MLflow Pipelines offer balanced alternatives with their own unique advantages, but show areas that could benefit from further improvement, particularly in pipeline consistency and data transfer latency. Azure ML Pipelines, while ranked lower overall, still offer a viable option for users looking for stable and reliable pipeline solutions. Ultimately, the decision on the most appropriate platform should align with the organization's specific operational priorities and needs, leveraging these insights to increase efficiency and effectiveness in machine learning applications.

References

1. Peter, Harry. "Cloud Native MLOps Automating Machine Learning Pipelines with Continuous Integration Continuous Deployment and Infrastructure as Code." (2024).
2. Peter, Harry. "Data Pipeline Optimization for Machine Learning Workflows in Cloud Environments." (2024).
3. Puthraya, Karthik, Rachit Gupta, and Beverly DSouza. "The Role of Cloud-Native Architectures in Accelerating Machine Learning Workflows through Data Engineering Innovations." Journal Of Applied Sciences 5, no. 2 (2025): 10-17.
4. ElKenawy, Ahmed Sobhy. "An Enhanced Cloud-Native Deep Learning Pipeline for the Classification of Network Traffic." (2023).
5. Ballamudi, S. "Interleaved Feature Extraction Model Bridging Multiple Techniques for Enhanced Object Identification" Journal of Artificial Intelligence and Machine Learning., 2023, vol. 1, no. 2, pp. 1-7. doi: https://doi.org/10.55124/jbid.v1i2.253
6. Ugwueze, Vincent. "Cloud Native Application Development: Best Practices and Challenges." International Journal of Research Publication and Reviews 5 (2024): 2399-2412.
7. Chinamanagonda, Sandeep. "Cloud-native Databases: Performance and Scalability-Adoption of cloud-native databases for improved performance." Advances in Computer Sciences 6, no. 1 (2023).
8. Sridhar Kakulavaram. (2022). Life Insurance Customer Prediction and Sustainbility Analysis Using Machine Learning Techniques. International Journal of Intelligent Systems and Applications in Engineering, 10(3s), 390 –. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/7649
9. Naayini, Prudhvi. "Building AI-Driven Cloud-Native Applications with Kubernetes and Containerization."

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256

10. Kumar, Shivansh, Er Neetu Bala, Arun Prakash Singh, and Yash Raj. “Cloud-Native Countinous Integration/Continous Deployment (CI/CD) Pipeline.” In 2024 Second International Conference on Advanced Computing & Communication Technologies (ICACCTech), pp. 292-297. IEEE, 2024.

11. Nadi, Faheem, and Anil Kapure. “Cloud-Native AI/ML Data Engineering with Generative AI MLOps and Scalable AI Workflows for Healthcare Innovation.” (2024).

12. Lu, Yao, Song Bian, Lequn Chen, Yongjun He, Yulong Hui, Matthew Lentz, Beibin Li et al. “Computing in the era of large generative models: From cloud-native to ai-native.” arXiv preprint arXiv:2401.12230 (2024).

13. Lawal, Kareem. “A Novel Framework for Next-Generation Data Pipelines in Real-Time Cloud Analytics.” (2025).

14. Nampelli, Swaroopa. “Enhancing CICD Pipelines For Automated Deployments With Cloud Native Infrastructures For High Availability Followed By Best Security Practices.” Int. J. Eng. Dev. Res 13, no. 2 (2025): 70-71.

15. Deng, Shuiguang, Hailiang Zhao, Binbin Huang, Cheng Zhang, Feiyi Chen, Yinuo Deng, Jianwei Yin, Schahram Dustdar, and Albert Y. Zomaya. “Cloud-native computing: A survey from the perspective of services.” Proceedings of the IEEE 112, no. 1 (2024): 12-46.

16. Koppolu, Hara Krishna Reddy, Anil Lokesh Gadi, Shabrinath Motamary, Abhishek Dodda, and Sambasiva Rao Suura. “Dynamic Orchestration of Data Pipelines via Agentic AI: Adaptive Resource Allocation and Workflow Optimization in Cloud-Native Analytics Platforms.” Metallurgical and Materials Engineering 31, no. 4 (2025): 625-637.

17. Pratiwi, Fadila, Fince Tinus Waruwu, Dito Putro Utomo, and Rian Syahputra. “Penerapan Metode ARAS Dalam Pemilihan Asisten Perkebunan Terbaik Pada PTPN V.” In Seminar Nasional Teknologi Komputer & Sains (SAINTEKS), vol. 1, no. 1. 2019.

18. Alemdar, Hande, Halil Ertan, Ozlem Durmaz Incel, and Cem Ersoy. “ARAS human activity datasets in multiple homes with multiple residents.” In 2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops, pp. 232-235. IEEE, 2013.

19. Sihombing, Volvo, Zulkarnain Nasution, Muhammad Ali Al Ihsan, Marlina Siregar, Ibnu Rasyid Munthe, Victor Marudut Mulia Siregar, Irma Fatmawati, and Dedy Ari Asfar. “Additive Ratio Assessment (ARAS) Method for Selecting English Course Branch Locations.” In Journal of Physics: Conference Series, vol. 1933, no. 1, p. 012070. IOP Publishing, 2021.

20. Turskis, Zenonas, and Edmundas Kazimieras Zavadskas. “A novel method for multiple criteria analysis: grey additive ratio assessment (ARAS-G) method.” Informatica 21, no. 4 (2010): 597-610.

21. Kutut, Vladislavas, Edmundas Kazimieras Zavadskas, and Marius Lazauskas. “Assessment of priority alternatives for preservation of historic buildings using model based on ARAS and AHP methods.” Archives of civil and mechanical engineering 14 (2014): 287-294.

22. Aras, Güler, Aslı Aybars, and Ozlem Kutlu. “Managing corporate performance: Investigating the relationship between corporate social responsibility and financial performance in emerging markets.” International Journal of productivity and Performance management 59, no. 3 (2010): 229-254.

23. Raś, Zbigniew W., Elżbieta Wyrzykowska, and Hanna Wasyluk. “ARAS: Action rules discovery based on agglomerative strategy.” In International workshop on mining complex data, pp. 196-208. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

24. Heller, Peter, Markus Pfänder, Thorsten Denk, Felix Tellez, Antonio Valverde, Jesús Fernandez, and Arik Ring. “Test and evaluation of a solar powered gas turbine system.” Solar energy 80, no. 10 (2006): 1225-1230.

25. Slocum, Alexander H., Daniel S. Codd, Jacopo Buongiorno, Charles Forsberg, Thomas McKrell, Jean-Christophe Nave, Costas N. Papanicolas et al. “Concentrated solar power on demand.” Solar Energy 85, no. 7 (2011): 1519-1529.

26. Zavadskas, Edmundas Kazimieras, and Zenonas Turskis. “A new additive ratio assessment (ARAS) method in multicriteria decision□ making.” Technological and economic development of economy 16, no. 2 (2010): 159-172.

27. Zavadskas, Edmundas Kazimieras, Zenonas Turskis, and Tatjana Vilutiene. “Multiple criteria analysis of foundation instalment alternatives by applying Additive Ratio Assessment (ARAS) method.” Archives of civil and mechanical engineering 10, no. 3 (2010): 123-141.

28. Stanujkic, Dragisa, and Rodoljub Jovanovic. “Measuring a quality of faculty website using ARAS method.” In Proceeding of the International Scientific Conference Contemporary Issues in Business, Management and Education, vol. 545, p. 554. 2012.

29. Pratiwi, Fadila, Fince Tinus Waruwu, Dito Putro Utomo, and Rian Syahputra. “Penerapan Metode ARAS Dalam Pemilihan Asisten Perkebunan Terbaik Pada PTPN V.” In Seminar Nasional Teknologi Komputer & Sains (SAINTEKS), vol. 1, no. 1. 2019.

Citation: Adari, V. K. (2025). Optimizing Cloud-Native Machine Learning Pipelines Using ARAS-Based Multi-Criteria Decision Analysis. International Journal of Computer Science and Data Engineering, 2(4), 270. https://doi.org/10.55124/csdb.v2i4.256