An Adaptive Data Preprocessing Framework for Improving Data Quality in Machine Learning Pipelines

Rusma Anieza Ruslan[1], Nureize Arbaiy[2], Pei-Chun Lin[3]

[1]Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia

[2]Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

[3]Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan

DOI: https://doi.org/10.47772/IJRISS.2026.101000592

Received: 30 October 2026; Accepted: 04 November 2026; Published: 11 November 2026

ABSTRACT

Data quality is an important factor influencing the reliability and predictive performance of machine learning models. Conventional preprocessing pipelines commonly apply fixed transformation procedures, such as mean imputation, standardisation, and outlier removal, regardless of changes in incoming data characteristics. Although these procedures are effective under relatively stable conditions, their performance may deteriorate when data distributions evolve, missing values increase, or feature characteristics become inconsistent over time. This study proposes an Adaptive Data Preprocessing Framework (ADPF) to improve data quality in machine learning pipelines through continuous monitoring and policy-driven preprocessing adaptation. The framework consists of three main components: Data Quality Monitor, Dynamic Policy Logic Engine, and Adaptive Execution Engine. The monitoring component evaluates feature-level missingness, Wasserstein distance, and feature skewness to identify changes in incoming data. The policy engine determines suitable preprocessing actions based on predefined data quality conditions, while the execution engine applies the selected operations, including missing-value imputation, outlier mitigation, scaling, and Yeo-Johnson transformation. A controlled simulation-based evaluation was designed using a synthetic dataset containing 10,000 observations, 20 input features, and a binary target variable. The evaluation considered different missingness levels and gradual distribution changes across sequential data batches. The results reported in the working experimental draft indicate that the adaptive approach maintained higher classification performance under increasing missingness, with accuracy decreasing from 88.4% to 78.9%, compared with a decrease from 88.4% to 51.0% for the static pipeline. Under the simulated drift scenario, the adaptive framework also maintained approximately 86% accuracy towards the end of the evaluation period, whereas the static pipeline declined to approximately 54%. These findings suggest that monitoring-driven preprocessing can improve the resilience of machine learning pipelines under changing data quality conditions. Further validation using reproducible experimental runs is required to establish the statistical reliability of the reported outcomes.

Keywords: Machine Learning Pipelines, Adaptive Data Preprocessing, Data Drift, Data Quality Monitoring, Production MLOps, Automated Feature Engineering

INTRODUCTION

Machine learning has become an important component of modern data-driven applications, supporting tasks such as classification, prediction, anomaly detection, and decision support. The performance of these applications depends not only on the selected learning algorithm but also on the quality and consistency of the data supplied to the model. In practical environments, data may contain missing values, inconsistent feature distributions, extreme observations, and changes in statistical characteristics. These issues can affect the

reliability of the learning process and reduce the generalisation capability of trained models. Consequently, data preprocessing is an essential stage in the development and deployment of machine learning systems.

Conventional preprocessing pipelines generally apply predefined operations to the training and testing datasets. Common procedures include missing-value imputation, categorical encoding, feature scaling, and outlier treatment. Although these methods provide a structured approach to data preparation, they are commonly configured during the initial development stage and remain unchanged during subsequent execution. Such static configurations may become unsuitable when incoming data characteristics differ from those observed during model development. Nevin et al. (2025) highlighted the importance of maintaining consistency between training and testing data-cleaning pipelines, demonstrating that preprocessing decisions can influence model behaviour. Similarly, research on data pipeline quality has identified data-related problems and processing decisions as important factors affecting the reliability of data-intensive applications (Foidl et al., 2024).

The challenge becomes more significant in continuous data pipelines, where data are received through recurring batches or streaming processes. Changes in data distributions may occur gradually or abruptly due to changes in user behaviour, operational conditions, data collection procedures, or external environments. These changes can affect the statistical properties of input features and may eventually reduce model performance. Concept drift research has therefore emphasised the importance of monitoring evolving data environments and identifying changes that require an appropriate response (Hinder et al., 2024; Arora et al., 2024). However, detecting distribution changes alone does not necessarily resolve data quality problems. A preprocessing mechanism is also required to determine whether existing transformations remain appropriate or whether alternative operations should be applied.

Existing automated machine learning approaches provide mechanisms for selecting algorithms, configuring models, and optimising machine learning workflows. Nevertheless, preprocessing decisions remain challenging when data quality conditions change continuously. Mumuni and Mumuni (2025) discussed the role of automated data processing and feature engineering in machine learning and big data applications, while Baratchi et al. (2024) reviewed the development of automated machine learning methods. These studies indicate the growing importance of automation in reducing manual configuration. However, a monitoring-driven mechanism that directly connects changing data quality indicators with preprocessing decisions remains an important research direction.

To address this issue, this study proposes an Adaptive Data Preprocessing Framework (ADPF) that integrates data quality monitoring, dynamic policy selection, and adaptive preprocessing execution. The framework continuously evaluates incoming data using feature-level missingness, Wasserstein distance, and feature skewness. Based on the detected conditions, the policy engine selects suitable preprocessing actions, which are subsequently executed by the adaptive processing component.

LITERATURE REVIEW

Automated Machine Learning and Pipeline Automation

Automated machine learning (AutoML) aims to reduce the manual effort involved in developing machine learning systems through automated model selection, hyperparameter optimisation, and pipeline configuration. The development of AutoML has expanded from conventional model optimisation towards more comprehensive workflows involving data preparation and feature engineering. Barbudo et al. (2023) reviewed the development and trends of AutoML, highlighting its growing role in automating different stages of machine learning development. Baratchi et al. (2024) also discussed the evolution of AutoML and its potential applications across different learning environments.

ecent studies have further explored automated processing in large-scale and evolving data environments. Mumuni and Mumuni (2025) examined automated data processing and feature engineering for deep learning and big data applications. Celik et al. (2023) investigated online AutoML for adaptive learning, while Verma et al. (2024) presented an approach for scalable and efficient AutoML in data streams. These developments demonstrate the importance of automation when data and learning conditions change over time.

Despite these developments, automated model selection does not automatically guarantee that preprocessing operations remain appropriate under changing data quality conditions. The proposed ADPF addresses this issue by focusing on the relationship between monitored data characteristics and preprocessing decisions.

Table I Comparison Related Studies Related to Automated Machine Learning and Data Preprocessing

Study	Research Focus	Contribution	Consideration
Barbudo et al. (2023)	AutoML development	Reviews AutoML categories and trends	Broad focus on automation
Baratchi et al. (2024)	AutoML methods	Reviews past, present, and future directions	Does not specifically address continuous data quality adaptation
Celik et al. (2023)	Online AutoML	Supports adaptative online learning	Emphasis on learning adaptation
Mumuni & Mumuni (2025)	Automated data preprocessing	Reviews preprocessing and feature engineering	Broad coverage of processing techniques
Verma et al. (2024)	AutoML for data streams	Supports scalable stream-based AutoML	Focuses on AutoML execution

Table 1 shows that previous studies have investigated automation across different machine learning stages, including model optimisation, feature engineering, and online learning. However, the relationship between continuous data quality monitoring and the selection of preprocessing operations remains an important area for further investigation.

Data Quality in Machine Learning Pipelines

Data quality refers to the degree to which data are suitable for their intended analytical or operational purpose. In machine learning pipelines, poor data quality may arise from missing values, invalid records, inconsistent representations, duplicated observations, and unexpected statistical distributions. These problems may affect model training, prediction reliability, and downstream analytical decisions. Priestley et al. (2023) discussed data quality requirements that are particularly relevant to machine learning development pipelines, emphasising the importance of considering data quality throughout the development process.

Data quality problems can also originate from pipeline design and execution. Foidl et al. (2024) investigated factors influencing data pipeline quality, including data-related issues and processing problem areas encountered by developers. Patel et al. (2023) explored a data-centric AI framework for automating exploratory data analysis and data quality tasks. Tu et al. (2023) investigated the automatic generation of data quality constraints for recurring data pipelines. These studies demonstrate the importance of systematic data validation and quality assessment.

For machine learning applications, data quality monitoring should consider not only whether individual records satisfy predefined rules but also whether the statistical characteristics of incoming data remain consistent with the reference data. A dataset may contain valid records while exhibiting substantial changes in feature distributions. Therefore, the proposed framework combines missingness monitoring with distribution and skewness indicators.

Table II Comparison of Studies Related to Data Quality Monitoring

Study	Research Focus	Contribution	Relevance to ADPF
Priestley et al. (2023)	Data quality requirements for machine learning	Identifies quality requirements in machine learning pipelines	Supports quality-oriented preprocessing
Patel et al. (2023)	Data-centric artificial intelligence	Automates exploratory analysis and quality tasks	Supports automated quality assessment
Tu et al. (2023)	Data quality constraints	Generates constraints for recurring pipelines	Supports repeated validation
Foidl et al. (2024)	Data pipeline quality	Examines factors and causes of pipeline problems	Supports pipeline-level quality management
Mirzaie et al. (2023)	Data stream quality control	Reviews quality control approaches for streams	Supports continuous monitoring

Table 2 indicates that existing research provides important mechanisms for assessing data quality and validating recurring pipelines. Nevertheless, quality assessment and preprocessing execution are often treated as separate activities. ADPF connects these activities through a policy engine that translates monitoring outcomes into preprocessing actions.

Data Drift and Distribution Monitoring

Data drift occurs when the statistical characteristics of incoming data change over time. In machine learning applications, changes in input feature distributions may affect the relationship between the input variables and the target variable, potentially reducing predictive performance. Drift may develop gradually or occur suddenly, depending on changes in the underlying data-generating environment.

Hinder et al. (2024) reviewed methods for detecting concept drift in evolving environments and discussed the importance of monitoring changing data distributions. Arora et al. (2024) presented a systematic review of concept drift detection and adaptation using machine learning techniques. Lukats et al. (2025) investigated unsupervised concept drift detectors for real-world data streams, highlighting the importance of evaluating drift detection methods in environments where immediate access to labels may be limited.

Distribution monitoring can be conducted using statistical indicators that quantify differences between reference and incoming data. The Wasserstein distance is one measure that can be used to quantify differences between probability distributions. Its application to drift monitoring allows a system to evaluate whether the distribution of an incoming feature has changed relative to a reference distribution. However, the distance must be interpreted carefully when features have different scales. Therefore, this study uses normalised feature-wise Wasserstein values for policy decisions.

Feature skewness is also monitored because changes in asymmetry may indicate that the distribution of a feature has become more concentrated in one direction. When considered together with missingness and distribution distance, skewness provides additional information for determining whether a preprocessing operation remains appropriate.

Table III Comparison of Studies Related to Data Drift Detection

Study	Research Focus	Contribution	Consideration
Hinder et al. (2024)	Concept drift detection	Surveys monitoring approaches in evolving environments	Focuses on detection

Arora et al. (2024)	Drift detection and adaptation	Reviews machine learning techniques for drift	Broad coverage of adaptation
Lukats et al. (2025)	Unsupervised drift detection	Benchmarks detectors on real-world streams	Focuses on detector performance
Mirzaie et al. (2023)	Data stream quality	Reviews quality control for data streams	Focuses on stream quality control

Table 3 shows that drift detection has received substantial attention, particularly in streaming and evolving environments. However, identifying a distribution change does not automatically determine which preprocessing operation should be applied. ADPF extends monitoring by connecting drift indicators to a policy-based preprocessing mechanism.

Static and Adaptive Data Preprocessing

Static preprocessing applies a fixed set of operations to the dataset, typically determined during model development. Common operations include mean or median imputation, outlier treatment, feature scaling, and power transformations. These operations can be effective when the characteristics of the data remain relatively stable. However, fixed parameters may become unsuitable when the incoming data distribution changes.

The consistency of preprocessing operations between training and testing is important because different transformations can alter the representation supplied to the model. Nevin et al. (2025) examined the effects of mismatched data-cleaning pipelines on regression models, highlighting the importance of maintaining compatible preprocessing procedures. In addition, Yeo and Johnson (2000) introduced a power transformation that can be applied to data containing both positive and negative values. Raymaekers and Rousseeuw (2022) further examined robust power transformations, including extensions related to Box-Cox and Yeo-Johnson approaches.

Adaptive preprocessing attempts to respond to changes in data characteristics by modifying processing decisions according to monitored conditions. Nevertheless, adaptation must be controlled to prevent inconsistencies between the representation used during model training and the representation supplied during inference. For example, changing a scaler or transformation parameter while retaining a frozen model may cause the model to receive values that no longer correspond to its learned decision boundaries. Therefore, ADPF applies adaptation under a model-compatibility constraint. Transformations that affect the model's learned representation must remain fixed unless the corresponding model is retrained or appropriately updated.

Table IV Comparison of Static and Adaptive Preprocessing Approaches

Aspect	Static Preprocessing	Adaptive Preprocessing
Configuration	Fixed before deployment	Selected based on monitored conditions
Data quality monitoring	Usually performed periodically or during initial preparation	Continuously or repeatedly monitored
Missing-value treatment	Uses predefined imputation rules	Selects treatment based on missingness conditions
Distribution changes	May not trigger preprocessing updates	Can trigger policy evaluation

Model compatibility	Uses a consistent fitted transformation	Requires explicit compatibility management
Operational complexity	Relatively low	Higher due to monitoring and policy execution

Table 4 highlights the main difference between the two approaches. Static preprocessing is simpler to operate, whereas adaptive preprocessing provides a mechanism for responding to changing data conditions. However, adaptive operations must preserve consistency with the model's training representation.

METHODOLOGY

Research Design

This study adopts a quantitative, simulation-based experimental design to evaluate the proposed Adaptive Data Preprocessing Framework. The evaluation focuses on three data quality conditions: increasing missingness, gradual feature distribution drift, and computational processing overhead. A static preprocessing pipeline is used as the baseline, while the proposed adaptive framework is evaluated under the same general data conditions. Both approaches use the same classification task and evaluation partition to support a consistent comparison.

The framework is organised into three main components. The Data Quality Monitor evaluates incoming data characteristics, the Dynamic Policy Logic Engine determines suitable preprocessing actions, and the Adaptive Execution Engine applies the selected operations. The experimental evaluation considers classification performance, resilience under changing data conditions, and computational requirements.

Dataset Description and Experimental Setup

The experimental design uses a synthetic dataset containing 10,000 observations and 20 input features, comprising 15 numerical features and five categorical features. The target variable contains two classes, with 7,000 observations belonging to Class 0 and 3,000 observations belonging to Class 1. This produces a class distribution of 70:30.

The dataset is divided into training, validation, and testing subsets using a 70:15:15 partition. The training subset contains 7,000 observations, while the validation and testing subsets each contain 1,500 observations. For the drift evaluation, the testing observations are arranged into 30 sequential batches, with 50 observations in each batch.

The missingness evaluation considers six conditions: 0%, 5%, 15%, 30%, and 50% missing values, together with the complete-data reference condition. Missing values are introduced using a missing-completely-at-random (MCAR) mechanism in the reported experimental design. The drift evaluation simulates gradual changes in feature distributions across sequential batches.

Table VI Policy Matrix for Adaptive Preprocessing

Indicator	Low Condition	Moderate Condition	High Condition
Feature missingness	Below 0.05: Retain existing treatment	0.05 – 0.15: Apply suitable imputation	Above 0.15: Apply stronger missingness handling and flag the feature
Normalised Wassertein distance	Below 0.10: Retain reference policy	0.10 – 0.20: Evaluate distribution change	Above 0.20: Trigger drift assessment and policy review
Absolute skewness	Below 1.0: Retain current transformation	1.0 – 2.0: Evaluate transformation suitability	Above 2.0: Evaluate power transformation

Table 6 presents the policy thresholds used in the experimental design. The thresholds provide operational decision boundaries rather than universal statistical standards. Their suitability should therefore be evaluated against the characteristics of the application dataset. In this study, the policy engine uses the monitored values to determine whether the current preprocessing configuration should be retained or whether an alternative action should be considered.

The policy engine also applies a compatibility check before allowing a transformation to change. This is important because changing fitted scaling or power-transformation parameters may alter the feature representation expected by a previously trained model. When a change requires a new feature representation, the corresponding model must be retrained or versioned using an appropriate training procedure before

Adaptive Execution Engine

The Adaptive Execution Engine applies the preprocessing actions selected by the policy engine. The main operations include missing-value imputation, outlier mitigation, scaling, and Yeo-Johnson transformation.

Missing-value imputation is selected according to the missingness condition and the feature type. Numerical features may use statistical imputation, while categorical features may use an appropriate category-based strategy. Outlier mitigation is considered when extreme observations are identified as potentially affecting feature distributions. Winsorisation may be used to limit the influence of extreme values while retaining the corresponding records.

Feature scaling is applied to maintain comparable numerical ranges where required by the learning algorithm. The Yeo-Johnson transformation is considered when feature distributions exhibit substantial skewness. This transformation is suitable for a broader range of numerical values, including zero and negative values (Yeo & Johnson, 2000). Robust extensions of power transformations have also been discussed by Raymaekers and Rousseeuw (2022).

A critical requirement of the execution engine is to maintain model compatibility. The fitted preprocessing parameters used by the deployed model must remain consistent with the model's learned representation. If the framework determines that a new fitted transformation is necessary, a corresponding model update or retraining procedure is required. The framework must not arbitrarily change transformation parameters while continuing to use an incompatible frozen model.

Evaluation Metrics

Classification performance is evaluated using accuracy, macro-precision, macro-recall, and macro-F1-score. Accuracy provides an overall measure of correct predictions, while macro-averaged metrics give equal importance to each target class. This is useful for examining whether preprocessing changes affect the recognition of both classes.

The robustness evaluation examines performance degradation under increasing missingness and gradual distribution drift. Computational performance is evaluated using mean batch latency, P95 batch latency, throughput, peak memory usage, and CPU utilisation. These measurements provide information about the operational cost of applying adaptive preprocessing.

RESULTS AND ANALYSIS

Performance under Different Missingness Level

The evaluation compares static preprocessing and ADPF under increasing missingness conditions. Both approaches begin with an accuracy of 88.4% when no missing values are introduced. As missingness increases, the static pipeline exhibits a larger reduction in classification performance. In comparison, the adaptive framework maintains higher accuracy across the evaluated conditions.

Table VII Classification Accuracy under Different Missingness Levels

Missingness Level	Static Preprocessing	ADPF	Improvement
0%	88.4	88.4	0.0
5%	86.1	88.2	2.1
15%	79.3	87.5	8.2
30%	68.2	84.1	15.9
50%	51.0	78.9	27.9

Table 7 shows that the performance difference becomes larger as missingness increases. At 5% missingness, ADPF achieves an accuracy of 88.2%, compared with 86.1% for static preprocessing. At 30% missingness, the difference increases to 15.9 percentage points. Under the highest missingness condition of 50%, static preprocessing records 51.0% accuracy, whereas ADPF records 78.9%.

These results suggest that a fixed preprocessing configuration may become less effective when missing values increase substantially. By monitoring missingness at the feature level, ADPF can identify variables that require more appropriate treatment. The results also indicate that the benefit of adaptation is more apparent under severe missingness.

Performance under Gradual Data Drift

The evaluation examines the behaviour of both approaches across 30 sequential batches. The static pipeline retains its original preprocessing configuration throughout the evaluation. ADPF, on the other hand, monitors incoming data distributions and evaluates whether the existing preprocessing policy remains suitable.

The reported results indicate that static preprocessing experiences a progressive decline in accuracy, reaching approximately 54% by the end of the evaluation period. In comparison, ADPF maintains approximately 86% accuracy towards the end of the 30-batch evaluation.

Table VIII Accuracy Trend under Gradual Data Drift

Evaluation Period	Static Preprocessing	ADPF
Initial period	Higher initial performance	Higher initial performance
Intermediate period	Progressive decline	Relatively stable
Final period (Batch 30)	Approximately 54%	Approximately 86%

Table 8 summarises the reported trend rather than providing unverified batch-level measurements. The final-period results indicate a difference of approximately 32 percentage points between the two approaches. This suggests that continuous data quality monitoring and policy-based preprocessing may help maintain performance when the statistical characteristics of incoming data change over time.

The results also demonstrate the importance of distinguishing between drift detection and drift response. A monitoring mechanism can identify distribution changes, but a pipeline requires an appropriate processing strategy to respond to those changes. ADPF combines both activities by connecting monitoring outcomes to preprocessing policy decisions.

Computational Performance

The evaluation examines the computational requirements of static preprocessing and ADPF. The adaptive framework introduces additional processing overhead because it evaluates data quality indicators and executes policy decisions before completing the preprocessing stage.

Table IX Computational Performance Comparison

Measurement	Static Preprocessing	ADPF
Mean batch latency (ms)	120	168
P95 batch latency (ms)	150	225
Throughput (records/s)	833.3	595.2
Peak memory usage (MB)	420	510
CPU utilisation	32	45
Policy evaluation time (ms)	-	18
Mean latency overhead	-	40.0

Table 9 indicates that ADPF requires more computational resources than the static pipeline. Mean batch latency increases from 120 ms to 168 ms, representing a 40% increase. P95 latency also increases from 150 ms to 225 ms. Throughput decreases from 833.3 to 595.2 records per second, while peak memory usage increases from 420 MB to 510 MB.

The additional overhead is associated with monitoring feature-level quality indicators and evaluating preprocessing policies. Although this overhead reduces processing throughput, it provides the pipeline with an additional mechanism for responding to changing data conditions. Therefore, the operational suitability of ADPF depends on whether the improvement in data quality and model performance justifies the additional processing cost.

CONCLUSIONS

This study proposed an Adaptive Data Preprocessing Framework (ADPF) to improve data quality management and enhance the reliability of machine learning pipelines under changing data conditions. The framework integrates three main components, namely the Data Quality Monitor, Dynamic Policy Logic Engine, and Adaptive Execution Engine, to monitor incoming data characteristics and select appropriate preprocessing operations based on missingness, Wasserstein distance, and feature skewness. The reported simulation results indicate that ADPF maintains better classification performance than static preprocessing under increasing missingness and gradual data drift, although it introduces additional computational overhead in terms of processing latency and memory consumption. These findings highlight the potential of adaptive preprocessing in addressing data quality challenges in continuous machine learning pipelines. Future work will focus on conducting repeated experiments to establish its effectiveness, robustness, and computational efficiency.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, for its support, assistance, and contributions towards the successful completion of this research.

REFERENCES

1. Arora, S., et al. (2024). A systematic review on detection and adaptation of concept drift in streaming data using machine learning techniques. WIREs Data Mining and Knowledge Discovery, 14(4), e1536. https://doi.org/10.1002/widm.1536
2. Baratchi, M., Wang, C., Limmer, S., van Rijn, J. N., Hoos, H. H., Bäck, T., & Olhofer, M. (2024). Automated machine learning: Past, present and future. Artificial Intelligence Review, 57, Article 122. https://doi.org/10.1007/s10462-024-10726-1
3. Barbudo, R., Ventura, S., & Romero, J. R. (2023). Eight years of AutoML: Categorisation, review and trends. Knowledge and Information Systems, 65, 5097–5149. https://doi.org/10.1007/s10115-023-01935-1
4. Celik, B., Singh, P., & Vanschoren, J. (2023). Online AutoML: An adaptive AutoML framework for online learning. Machine Learning, 112, 1897–1921. https://doi.org/10.1007/s10994-022-06262-0
5. Ehrlinger, L., Wöß, W., & colleagues. (2022). A survey of data quality measurement and monitoring tools. [Please verify the complete author list and publication details before final submission.] https://pmc.ncbi.nlm.nih.gov/articles/PMC9009315/
6. Foidl, H., Golendukhina, V., Ramler, R., & Felderer, M. (2024). Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers. Journal of Systems and Software, 207, 111855. https://doi.org/10.1016/j.jss.2023.111855
7. Hinder, F., Vaquet, V., & Hammer, B. (2024). One or two things we know about concept drift—a survey on monitoring in evolving environments. Part A: Detecting concept drift. Frontiers in Artificial Intelligence, 7. https://doi.org/10.3389/frai.2024.1330257
8. Lukats, D., Zielinski, O., Hahn, A., & Stahl, F. (2025). A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams. International Journal of Data Science and Analytics, 19, 1–31. https://doi.org/10.1007/s41060-024-00620-y
9. Mirzaie, M., Behkamal, B., Allahbakhsh, M., Paydar, S., & Bertino, E. (2023). State of the art on quality control for data streams: A systematic literature review. Computer Science Review, 48, 100554. https://doi.org/10.1016/j.cosrev.2023.100554
10. Mumuni, A., & Mumuni, F. (2025). Automated data processing and feature engineering for deep learning and big data applications: A survey. Journal of Information and Intelligence, 3(2), 113–153. https://doi.org/10.1016/j.jiixd.2024.01.002
11. Neutatz, F., Lindauer, M., & Abedjan, Z. (2024). AutoML in heavily constrained applications. The VLDB Journal, 33, 957–979. https://doi.org/10.1007/s00778-023-00820-1
12. Nevin, J., Lees, M., & Groth, P. (2025). The effects of mismatched train and test data cleaning pipelines on regression models: Lessons for practice. PeerJ Computer Science, 11, e2793. https://doi.org/10.7717/peerj-cs.2793
13. Patel, H., Guttula, S., Gupta, N., Hans, S., Mittal, R. S., & Lokesh, N. (2023). A data-centric AI framework for automating exploratory data analysis and data quality tasks. ACM Journal of Data and Information Quality, 15(4), Article 44. https://doi.org/10.1145/3603709
14. Priestley, M., O'Donnell, F., & Simperl, E. (2023). A survey of data quality requirements that matter in ML development pipelines. ACM Journal of Data and Information Quality, 15(2), Article 11. https://doi.org/10.1145/3592616
15. Raymaekers, J., & Rousseeuw, P. J. (2022). Automatic robust Box–Cox and extended Yeo–Johnson transformations in regression. Statistical Methods & Applications. https://doi.org/10.1007/S10260-022-00640-7
16. Salehin, I., Islam, M. S., Saha, P., Noman, S. M., Tuni, A., Hasan, M. M., & Baten, M. A. (2024). AutoML: A systematic review on automated machine learning with neural architecture search. Journal of Information and Intelligence, 2(1), 52–81. https://doi.org/10.1016/j.jiixd.2023.10.002
17. Staerman, G., Laforgue, P., Mozharovskyi, P., & d'Alché-Buc, F. (2021). When OT meets MoM: Robust estimation of Wasserstein distance. Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, 130, 136–144. https://proceedings.mlr.press/v130/staerman21a.html
18. Tu, D., He, Y., Cui, W., Ge, S., Zhang, H., Shi, H., Zhang, D., & Chaudhuri, S. (2023). Auto-validate by-history: Auto-program data quality constraints to validate recurring data pipelines. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.

19. Verma, N., Bifet, A., Pfahringer, B., & Bahri, M. (2024). ASML: A scalable and efficient AutoML solution for data streams. Proceedings of the Third International Conference on Automated Machine Learning, 256, 11/1–11/26. https://proceedings.mlr.press/v256/verma24a.html
20. Yeo, I.-K., & Johnson, R. A. (2000). A new family of power transformations to improve normality or symmetry. Biometrika, 87(4), 954–959. https://doi.org/10.1093/biomet/87.4.954