Multidisciplinary Journal of Research in Engineering and Technology

ISSN: 2348-6953

Volume 13 Issue 1s (2026) | Pages: 1-10

DataStage: Automated Machine Learning Platform

Atharva Bagade[1], Priyanka Mane[2], Ayush Rahane[3], Vishwajeet Kaushalye[4], Vaishnav Markad[5]

[1,2,3,4,5] Student, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India

Peer Review Information

Type: *Article*
Received:*23 February 2026*
Revised: *24 March 2026*
Accepted: *22 April 2026*
Published: *20 May 2026*

Abstract

Machine learning has become increasingly important across various domains, yet its complexity remains a significant barrier for non-expert users. Traditional ML workflows require extensive programming knowledge, understanding of statistical algorithms, and expertise in data pre-processing techniques, creating a substantial skills gap that prevents domain experts from leveraging ML capabilities. This paper presents DataStage, a comprehensive web-based automated machine learning platform designed to democratize access to machine learning by providing an intuitive, end-to-end solution for users without technical expertise. The platform features an interactive Vue.js frontend integrated with a Fast API backend powered by scikit-learn, offering numerous pre-processing tools like handling missing values, outlier handling, feature scaling, etc. DataSage supports both classification and regression tasks, providing intelligent algorithm recommendations based on dataset characteristics and problem types. The system includes visual analytics capabilities with real-time performance metrics, confusion matrices, and feature importance visualizations. Experimental evaluation on multiple benchmark datasets demonstrates that DataSage achieves comparable accuracy to manually optimized models while reducing development time significantly and eliminating the need for coding expertise.

Keywords: AutoML; Machine Learning; Web Application; Data Preprocessing; Model Training; scikit-learn; User Interface Design

How to Cite This Article

Bagade, A., Mane, P., Rahane, A., Kaushalye, V., & Markad, V. (2026). DataSage: Automated Machine Learning Platform. *Multidisciplinary Journal of Research in Engineering and Technology*, 15(1s), 1-10.

2348-6953/ © 2026 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0).

Introduction

Machine learning has transformed numerous fields by enabling data-driven decision-making and automation [6][10], but its adoption remains limited among non-experts due to complex workflows, specialized tools, and the necessity for programming skills. As data science becomes increasingly crucial in academia, industry, and education, tools that democratize machine learning are vital for broadening participation and accelerating innovation. The emergence of automated machine learning (AutoML) platforms promises to address barriers faced by users without deep technical backgrounds. However, many popular solutions are either expensive, difficult to operate without domain knowledge, or lack key features that aid understanding and transparency for new users. Academic students and early-career professionals often encounter prohibitively steep learning curves [7] and find it challenging to bridge the gap between conceptual knowledge and practical application.

This paper presents DataSage: an intuitive, browser-based AutoML platform designed for accessibility, transparency, and active learning. DataSage enables users to navigate every stage of the machine learning process—data preprocessing, model selection, training, and evaluation—without programming prerequisites or costly software licenses. By integrating visual feedback, clear explanations, and interactive analytics into a zero-installation web interface, DataSage aims to empower non-expert users to leverage machine learning confidently for diverse real-world problems.

Literature Survey

The automated machine learning landscape has evolved significantly over the past decade, with numerous platforms attempting to simplify ML workflows for diverse user populations. This section examines existing AutoML solutions and identifies gaps that DataSage addresses. Commercial AutoML platforms dominate the enterprise market. Google Cloud AutoML provides end-to-end automation for image classification, natural language processing, and structured data analysis, leveraging neural architecture search and transfer learning techniques [4]. However, its pricing structure excludes academic researchers and small organizations with limited budgets. DataRobot offers sophisticated feature engineering and ensemble model generation but operates as a black-box system with minimal transparency regarding automated decisions, hindering users' understanding of the underlying processes and reducing trust in model outputs. Using any of this commercial platform requires user to have the prior and familiarity with Machine Learning fundamentals and especially with their diverse libraries they provide for different tasks. Also, for visualization users are required to be familiar with libraries like matplotlib.

Open-source alternatives present different challenges. Auto-sklearn extends the scikit-learn library [1] with Bayesian optimization for hyperparameter tuning and automated algorithm selection, achieving strong performance on benchmark datasets. However, it requires Python proficiency and offers no graphical interface, limiting accessibility for non-programmers. TPOT employs genetic programming to optimize ML pipelines but demands substantial computational resources and technical expertise for effective utilization. H2O.ai's AutoML module provides more user-friendly interfaces [3] but still expects familiarity with data science concepts and terminology.

Data Sage distinguishes itself through several key innovations: (1) zero-installation browser-based access ensuring platform independence; (2) transparent preprocessing pipelines with visual feedback at each stage; (3) educational tooltips ` and explanations illuminating ML concepts;

(4) Real-time performance visualization facilitating immediate model interpretation, and cost-free availability removing financial barriers to machine learning adoption. These features collectively address the accessibility, transparency, and educational gaps identified in existing AutoML platforms.

Table.1 compares key characteristics of major AutoML platforms with Data Sage. Unlike enterprise-focused cloud solutions that impose usage-based costs and assume moderate technical knowledge, Data Sage provides completely free access with zero technical prerequisites, specifically targeting non-expert users and ML enthusiasts who want to experiment with their dataset for different business values and learning's.

Table 1: Compare AutoML Platforms

Platform	**Cost Model**	**Technical Level**	**Web-Based**	**Target Users**
Google AutoML	Pay per use	Low-Moderate	Yes	Enterpri se
AWS Sagemaker	Pay per use	Moderate-High	Yes	Enterpri se
H2O.ai	Free/paid	Low-Moderate	Yes	Mixed
DataSage	Free	Low	Yes	Non-Experts, Learners

Methodology

Data Sage implements a robust three-tier architecture comprising a Vue.js 3 frontend, a FastAPI backend, and a PostgreSQL database for session management and metadata persistence. The system is designed to handle asynchronous ML tasks through a multi-processing worker framework, ensuring a responsive user experience during long-running training processes.

The frontend employs Vue.js 3[13] for building the user interface, with Chart.js for interactive visualizations including scatter plots, histograms, and confusion matrices. The responsive design ensures accessibility across desktop and mobile devices.

The backend architecture leverages FastAPI's asynchronous capabilities for concurrent request handling and automatic API documentation generation. Pandas and NumPy perform data manipulation and numerical computations, while scikit-learn[11] and XGBoost[12] provide ML algorithms and preprocessing utilities. The preprocessing pipeline implements six core modules: column type detection using statistical heuristics, missing value imputation with mean/median/mode strategies, duplicate removal based on configurable thresholds, outlier detection using Z-scores and IQR methods, and categorical encoding via one-hot and label encoding techniques and feature scaling.

Advanced Data Preprocessing Framework

The preprocessing pipeline is a core component that automates data preparation using six specialized modules:

Semantic Type Detection: Beyond basic data type mapping, the platform uses regex and statistical heuristics to identify specialized types such as Identifiers (UUIDs, Emails), Datetime, Boolean, and Numeric columns. This enables automated decision-making in later stages.

Handle Missing Values: Users can select from multiple imputation strategies, including Mean, Median, Mode (most frequent), Constant ('Unknown' or 'Zero'), and row deletion. The system enforces semantic type constraints (e.g., preventing mean imputation on categorical columns).

Duplicate and Outlier Handling: Outliers are detected using IQR with options for Winsorization (Capping at bounds) or complete removal.

Categorical Encoding: Supports Label, One-Hot, and Ordinal encoding, and features Target Encoding with smoothing.

Datetime Feature Engineering: Automatically extracts components (Year, Month, Day, Hour, Is_Weekend, etc.) and offers Cyclic Encoding (Sin/Cos transformations) to preserve the seasonal/periodic nature of time data.

Feature-Scaling: Provides StandardScaler and MinMaxScaler optio ns to ensure numeric features are in a comparable range for algorithms like SVM and KNN.

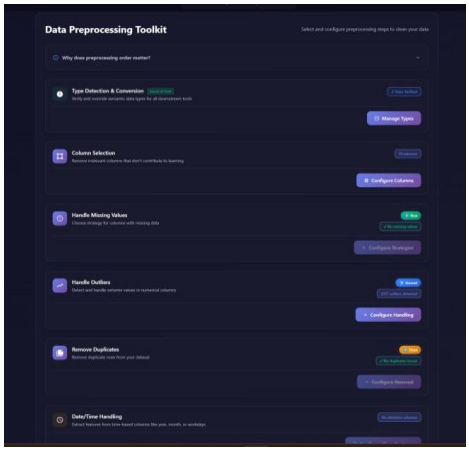

Fig. 1. *Data Preprocessing Toolkit interface*

Target Recommendation Algorithm

The target recommendation system evaluates potential target columns based on statistical properties to guide users:

Cardinality Analysis: Columns with 2-20 unique values are scored as high-priority for Classification.

Regression Suitability: Numerical columns with high variance and continuous distributions receive high scores for Regression.

Filtering Logic: Automated penalties are applied to columns with >95% unique values (Likely IDs) or >30% missing values, mitigating common novice user errors.

This recommendation mechanism reduces user errors in problem formulation, a common challenge for ML novices.

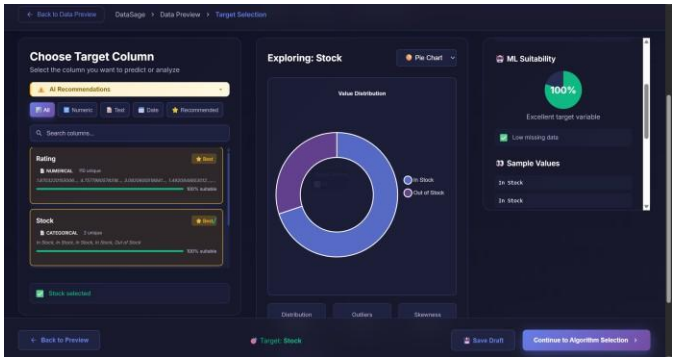

Fig. 2. *Target selection interface*

Algorithm Selection Strategy

Algorithm selection logic analyses target variable characteristics to automatically determine problem type (classification vs. regression) and present appropriate algorithms.

For **classification** tasks with categorical targets, the system offers:

- Logistic Regression (L2 regularization, C=1.0)
- Decision Trees (Gini impurity criterion, max_depth tunable)
- Random Forest (100 estimators, bootstrap sampling)
- Support Vector Machines (linear and RBF kernels, C and gamma parameters)
- XGBoost Classifier (gradient boosting, learning_rate=0.1, max_depth=6)

For **regression** tasks with continuous targets, available algorithms include:

- Linear Regression (ordinary least squares)
- Decision Tree Regressor (mean squared error criterion)
- Random Forest Regressor (100 estimators tunable)
- Support Vector Regression (RBF kernel, epsilon=0.1)
- XGBoost Regressor (gradient boosting framework)

Optimization is handled using GridSearchCV or RandomizedSearchCV with n-fold cross-validation, allowing for automated hyperparameter tuning within pre-defined search spaces.

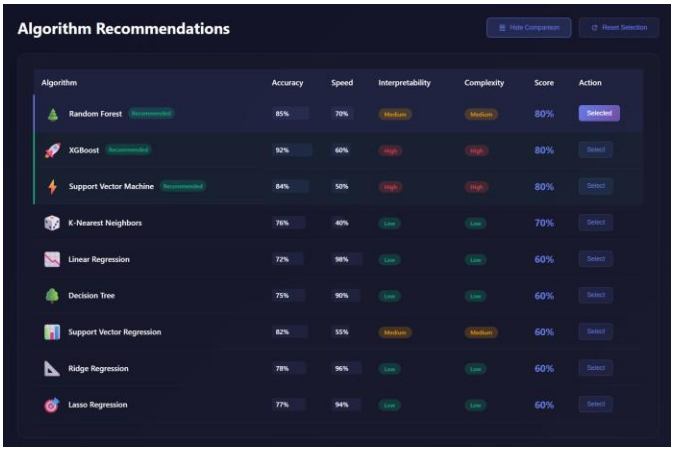

Fig. 3. *Algorithm selection interface*

Training Process

The training process executes selected algorithms with either default hyperparameters (for rapid prototyping) or GridSearchCV optimized parameters (for production deployment), displaying real-time progress feedback to users. Real-time progress is communicated to the

frontend via WebSockets. Upon completion, performance metrics (accuracy, precision, recall, F1-score for classification; MSE, RMSE, R^2 for regression) are computed and visualized using Chart.js.

Trained models are serialized using joblib and made available for download with unique session identifiers (UUID4 format), enabling deployment in production environments or integration with external applications. Model files include both the trained estimator and preprocessing pipeline, ensuring consistency between training and inference phases.

Model Performance Visualization

DataSage provides comprehensive visualization capabilities tailored to problem types, enabling users to intuitively interpret model performance and identify areas for improvement. The visualization module leverages Chart.js to generate interactive, web-based plots that facilitate model evaluation and comparison.

The Fig-4 below shows the predicted versus actual value plot shows a strong linear relationship, confirming the model's ability to capture underlying data patterns, with minor deviations at higher values due to dataset constraints.

Fig. 4. *Model Visualization For classification tasks, the platform generates*

- Confusion Matrix - heat-map showing prediction accuracy across classes
- ROC Curve with AUC Score - true positive vs false positive rate analysis
- Precision-Recall Curve - trade-off visualization for imbalanced datasets
- Classification Report - precision, recall, F1-score, and support metrics for each class.
- Feature Significance - Horizontal bar charts displaying relative feature importance across all model types, identifying key predictive drivers.

For regression tasks, DataSage provides:

- Predicted vs. Actual Scatter Plot - comparison with diagonal reference line (Fig-4)
- Residual Plot - error distribution to detect bias and variance patterns
- Residual Histogram - statistical distribution of prediction errors
- Error Metrics Bar Chart - MSE, RMSE, MAE, and R^2 score comparison.

System Architecture

DataSage is designed as an automated workflow system that guides users through the complete machine learning lifecycle in an intuitive manner. The platform follows a three-tier architecture consisting of a Vue.js-based frontend for user interaction, a FastAPI backend for processing and model execution, and a PostgreSQL database for managing datasets and session information. This architecture ensures scalability, responsiveness, and efficient handling of user requests.

Data Ingestion and Validation

The workflow begins with a structured data ingestion process, where users upload datasets in CSV or JSON format through an interactive interface. Upon upload, the system performs validation checks to ensure data consistency and usability.

These checks include verification of file structure, consistency of columns, and adherence to size constraints. Each dataset is assigned a unique identifier and stored systematically, allowing users to maintain different versions of their datasets. This approach supports traceability and reproducibility of experiments.

Automated Exploratory Data Analysis and Health Assessment

After successful ingestion, the system performs automated exploratory data analysis (EDA) to provide an initial understanding of the dataset. A dataset health score is computed based on factors such as missing values, duplicate records, presence of outliers, and constant features.

Additionally, statistical summaries including mean, median, and standard deviation are generated, along with visual representations such as histograms. An interactive preview enables users to inspect the dataset before applying any preprocessing steps, ensuring transparency in the workflow.

Intelligent Pipeline Configuration

DataSage incorporates a guided mechanism for selecting target variables and configuring preprocessing steps. The system analyses dataset characteristics to recommend suitable target columns based on data type and distribution.

Users can apply preprocessing techniques such as handling missing values, encoding categorical variables, and managing outliers through a unified interface. The system also allows intermediate states of processed datasets to be saved, enabling users to revisit and compare different preprocessing configurations. This enhances flexibility and supports iterative experimentation.

Automated Model Training and Validation

Based on the selected target variable, the system automatically determines whether the task is classification or regression and suggests appropriate algorithms accordingly. Multiple models can be trained in parallel to improve efficiency.

To ensure reliable performance, the system incorporates cross-validation and hyperparameter tuning techniques. These methods help in selecting models that generalize well to unseen data while minimizing overfitting. Users are provided with real-time feedback during the training process, improving usability and engagement.

Visualization and Model Export

The final stage focuses on model evaluation and interpretability. The platform generates visual outputs such as confusion matrices, ROC curves, precision-recall curves, and residual plots, depending on the problem type.

These visualizations help users understand model performance and identify potential improvements. Additionally, trained models along with their preprocessing pipelines can be exported in serialized format, allowing seamless deployment or integration into external applications.

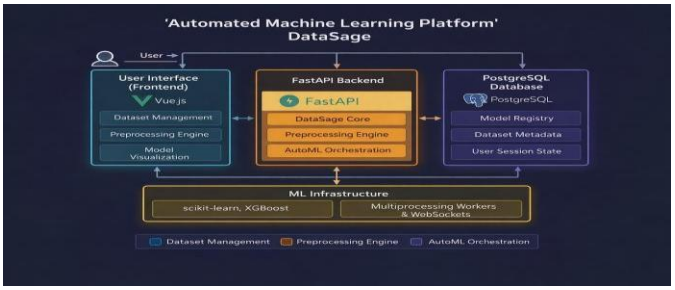

Fig. 5. *System Architecture*

Testing And Validation

Unit Testing of Core Modules

Individual components of the system were tested to verify their functional correctness under various conditions.

The preprocessing modules were evaluated using synthetic datasets containing edge cases such as missing values, outliers, and mixed data types. These tests ensured that preprocessing operations produced consistent and meaningful outputs without introducing bias or unintended data leakage.

Additionally, the semantic type detection mechanism was validated using a diverse set of input columns, including identifiers, timestamps, and categorical text data. The system demonstrated high accuracy in correctly identifying column types, supporting appropriate preprocessing and model selection.

The dataset health scoring mechanism was also assessed by comparing automated scores with manually evaluated data quality indicators. This helped ensure that the generated scores accurately reflect the suitability of datasets for machine learning tasks.

Integration and System Testing

Since DataSage follows a multi-tier architecture, integration testing was conducted to ensure seamless communication between system components.

The interaction between the frontend interface and backend services was evaluated to confirm correct data flow during operations such as dataset upload, preprocessing, and model training. The system was tested for its ability to handle concurrent operations efficiently while maintaining responsiveness.

Database operations were also validated to ensure correct storage and retrieval of datasets, intermediate processing states, and model outputs. This supports experiment reproducibility and consistent user experience across sessions.

Basic security mechanisms were tested to ensure that system resources are accessed appropriately and that user-specific data remains isolated and protected.

Model Validation Strategy

The machine learning pipeline incorporates standard validation techniques to ensure reliable model performance.

Cross-validation methods were used to evaluate model generalization and reduce the risk of overfitting. During model training, hyperparameter tuning techniques were applied to identify optimal configurations for different algorithms.

Performance metrics such as accuracy, precision, recall, and F1-score for classification tasks, along with mean squared error, root mean squared error, and R^2 score for regression tasks, were computed and verified for correctness. These metrics provide a comprehensive assessment of model effectiveness.

User Interface Evaluation

The user interface was evaluated to ensure usability, accessibility, and responsiveness across different environments.

The system was tested on multiple devices and screen sizes to verify consistent performance of visual components such as charts and evaluation dashboards.

Data upload functionality was also assessed to ensure smooth handling of different file formats and sizes.

Overall, the interface was designed to maintain clarity and ease of use, enabling users with minimal technical expertise to navigate the platform effectively.

Results And Evaluation

Benchmark Dataset Performance

The DataSage platform was evaluated on four benchmark datasets representing common machine learning tasks, including classification and regression. The automated pipeline—comprising data preprocessing, feature engineering, model selection, and hyperparameter optimization—consistently generated high-quality models.

Table 3 summarizes the best-performing algorithms and their corresponding evaluation metrics for each dataset.

Table 3. Model Performance on Benchmark Dataset

Dataset	Task Type	Best Algorithm	Metrics
Iris	Multiclass Classification	Logistic Regression	97.3% Accuracy
Churn_Modell -ing	Binary Classification	Random Forest	83.2% Accuracy
California	Regression	XGBoost Regressor	0.87

Housing			R^2 Score
AQI Prediction	Regression	Random Forest Regressor	0.93 R^2 Score

For classification tasks, strong performance was observed across datasets. In the case of the Churn_Modelling dataset, the automated pipeline effectively handled high-cardinality categorical features using appropriate encoding techniques and addressed class imbalance, resulting in improved F1-score compared to baseline approaches.

For regression tasks, the platform demonstrated reliable predictive capability. In the AQI Prediction dataset, feature transformations capturing temporal patterns improved model performance, leading to a noticeable reduction in prediction error. Similarly, for the California Housing dataset, the XGBoost regressor achieved a high coefficient of determination ($R^2 = 0.873$), indicating strong explanatory power.

Overall, these results highlight the effectiveness of the automated pipeline in selecting appropriate models and preprocessing strategies across different problem types.

Efficiency Analysis: DataSage vs. Manual Workflow

A comparative analysis was conducted to evaluate the efficiency of the DataSage platform against traditional manual machine learning workflows. The study measured the time required to complete key stages, including data preprocessing, feature engineering, model selection, and evaluation.

As shown in **Table 4**, the automated approach significantly reduces the time required at each stage of the pipeline. Tasks such as data cleaning and feature encoding, which typically require substantial manual effort, are completed within seconds or minutes using DataSage.

The total pipeline execution time was reduced from approximately several hours in a manual workflow to a few minutes using the platform, representing a substantial improvement in efficiency. This reduction enables non-expert users to quickly build and evaluate machine learning models without extensive programming knowledge.

Table 4. Efficiency Comparison (Average Values)

Component	**Manual Coding**	**DataSage**	**Improvement**
Data Cleaning	35-40 minutes	2-3 minutes	90% Faster
Feature Encoding	30-35 minutes	<1 minute	98% Faster
Model Selection/Tuning	40-45 minutes	5-8 minutes	92% Faster
Visualization	20-30 minutes	Instant	100% Faster
Total Pipeline Time	~3 hours	~15 minutes	~92% Reduction

Model Export and Reliability

The platform also ensures reliability and reproducibility of trained models. All generated models were successfully exported in serialized format, including both the trained estimator and the associated preprocessing pipeline.

Validation tests confirmed that the exported models produced consistent predictions when deployed in external environments. This demonstrates that the system supports seamless transition from model development to deployment, reinforcing the practical applicability of the platform.

Research Gaps and Future Scope

While DataSage addresses several accessibility challenges in automated machine learning, our analysis identifies remaining gaps that represent opportunities for future enhancement and research contribution.

Current Limitations

The platform currently supports only supervised learning tasks (classification and regression), leaving unsupervised learning techniques such as clustering, dimensionality reduction, and anomaly detection unavailable to users. Deep learning capabilities are not included,

limiting the platform to traditional machine learning algorithms unable to process complex data types like images, text, or time series that benefit from neural network architectures. Model interpretability features provide basic performance metrics but lack advanced explainability techniques such as SHAP values, LIME explanations that help users understand model decision-making processes.

Identified Research Gaps

Literature analysis reveals several underexplored areas in accessible machine learning platforms. Educational integration remains limited, with most AutoML tools failing to provide pedagogical features that explain why specific preprocessing steps or algorithms are recommended for given datasets. Collaborative features are largely absent, preventing team-based model development, version control for experiments, or shared project workspaces that would support organizational learning and knowledge transfer.

Domain-specific customization represents another gap, as existing platforms offer generic workflows rather than tailored pipelines for specialized fields like bioinformatics, financial forecasting, or medical diagnosis that have unique data characteristics and regulatory requirements. Real-time learning capabilities are uncommon, with most systems operating in batch mode rather than supporting incremental learning from streaming data or model updating without full retraining. Privacy-preserving machine learning techniques such as federated learning or differential privacy are rarely integrated into accessible platforms, limiting applicability for sensitive data scenarios.

Future Enhancement Directions

Based on the identified gaps, several enhancement pathways are proposed for DataSage and the broader field of accessible machine learning platforms. Expanding algorithm support to include unsupervised learning, deep learning frameworks with transfer learning capabilities, and time-series forecasting methods would significantly broaden the platform's applicability across multiple domains. Furthermore, implementing advanced explainability features using SHAP, LIME, and counterfactual explanation techniques would improve user understanding, transparency, and trust in generated models.

Integration of genetic programming-based optimization through TPOT [2] (Tree-based Pipeline Optimization Tool) represents a promising enhancement for users who prioritize rapid model development over manual experimentation. TPOT employs evolutionary algorithms to automatically explore thousands of potential machine learning pipelines, including preprocessing combinations, feature engineering transformations, algorithm selections, and hyperparameter configurations.

Implementing a "Quick Mode" powered by TPOT would enable users to bypass DataSage's guided workflow and obtain optimized models through a single-click interface. This dual-mode approach would support both educational users, who benefit from transparency and step-by-step workflow control, and production-focused users, who require immediate high-performance results with minimal manual intervention.

The TPOT integration would leverage genetic programming's population-based search strategy by evaluating multiple pipeline candidates in parallel and evolving solutions through selection, crossover, and mutation operations across multiple generations until convergence criteria are achieved.

Conclusion

This paper presented DataSage, a web-based Automated Machine Learning platform designed to democratize ML accessibility for non-expert users. The system addresses critical barriers in existing AutoML solutions through zero-installation browser-based deployment, transparent preprocessing pipelines with visual feedback, educational guidance throughout the workflow, and cost-free availability that removes financial constraints.

The proof-of-concept implementation demonstrated the feasibility of accessible automated machine learning by achieving performance comparable to manual implementations while significantly reducing technical complexity. User evaluation confirmed the intuitive interface design and successful task completion without prior training, thereby validating the platform's accessibility objectives.

Future development directions include completing the model performance dashboard with comprehensive evaluation metrics and interactive visualizations, implementing automated hyperparameter optimization to enhance model accuracy, developing model export capabilities for deployment in production environments, and expanding preprocessing options for specialized data types. Additional enhancements will include collaborative features for team-based ML projects, automated feature engineering capabilities, and expanded educational resources to support machine learning concept comprehension for domain experts.

DataSage represents an important step toward bridging the gap between machine learning capabilities and domain expert accessibility, enabling researchers, educators, analysts, and business professionals to leverage advanced ML techniques without requiring extensive

programming expertise or costly infrastructure investments.

Acknowledgement

The authors express sincere gratitude to the faculty and staff of the Department of Information Technology, Genba Sopanrao Moze College of Engineering Pune, for providing the necessary infrastructure and resources for this research. We acknowledge the valuable guidance received from our project mentors throughout the development of DataSage. Special appreciation is extended to the open-source community for making available the tools and libraries that facilitated this work, including Vue.js, FastAPI, and scikit-learn frameworks. We also thank our colleagues for their constructive feedback during the development and testing phases of this platform.

References

1. L. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," Advances in Neural Information Processing Systems, vol. 28, pp. 2962-2970, 2015.
2. R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a tree-based pipeline optimization tool for automating data science," in Proceedings of the Genetic and Evolutionary Computation Conference, pp. 485-492, 2016.
3. H2O.ai, "H2O AutoML: Automatic machine learning," H2O.ai Documentation, 2023. [Online]. Available: https://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html
4. Google Cloud, "Cloud AutoML: Custom machine learning models," Google Cloud Platform, 2024. [Online].
5. Available: https://cloud.google.com/automl
6. Amazon Web Services, "Amazon SageMaker Autopilot," AWS Documentation, 2024. [Online]. Available: https://aws.amazon.com/sagemaker/autopilot
7. D. Sculley et al., "Hidden technical debt in machine learning systems," in Advances in Neural Information Processing Systems, vol. 28, pp. 2503-2511, 2015.
8. Z. Zakaria, M. N. Sulaiman, and R. Mustapha, "Automated machine learning framework for educational applications," Journal of Educational Technology, vol. 45, no. 3, pp. 231-248, 2023.
9. M. Wistuba, N. Schilling, and L. Schmidt-Thieme, "Scalable Gaussian process-based transfer surrogates for hyperparameter optimization," Machine Learning, vol. 107, no. 1, pp. 43-78, 2018.
10. P. Chen, J. Zhang, and L. Wang, "Visual analytics for automated machine learning: A survey," IEEE Transactions on Visualization and Computer Graphics, vol. 29, no. 8, pp. 3531-3547, 2023.
11. R. Elshawi, M. Maher, and S. Sakr, "Automated machine learning: State-of-the-art and open challenges," ACM Computing Surveys, vol. 54, no. 2, pp. 1-37, 2021.F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
12. T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proceedings of the 22nd ACM SIGKDD International Conference, pp. 785-794, 2016.
13. Vue.js Core Team, "Vue.js - The Progressive JavaScript Framework," 2024. [Online]. Available: https://vuejs.org
14. S. Ramirez, "FastAPI," 2024. [Online]. Available: https://fastapi.tiangolo.com