International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 11 | Nov 2025 www.irjet.net p-ISSN: 2395-0072

DataSage: Automated Machine Learning Platform

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

[1]*Atharva Bagade, Student, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India*
[2]*Priyanka Mane, Professor, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India*
[3]*Ayush Rahane, Student, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India*
[4]*Vishwajeet Kaushalye, Student, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India*
[5]*Vaishnav Markad, Student, Department of Information Technology, GSMCOE Balewadi, Maharashtra, India*

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 preprocessing techniques, creating a substantial skills gap that prevents domain experts from leveraging ML capabilities. This paper presents DataSage, 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 FastAPI backend powered by scikit-learn, offering six automated data preprocessing modules: column selection, missing value handling, duplicate removal, outlier detection, categorical encoding and Feature scaling. 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 four benchmark datasets demonstrates that DataSage achieves comparable accuracy to manually optimized models while reducing development time significantly and eliminating the need for coding expertise.*

Key Words: *AutoML, Machine Learning, Web Application, Data Preprocessing, Model Training, scikit-learn, User Interface Design*

1. 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] Google 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.

2. 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 experience 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.

DataSage 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 (5) cost-free availability removing financial barriers to ML adoption. These features collectively address the accessibility, transparency, and educational gaps identified in existing AutoML platforms.

Table-1: Compare AutoML Platforms

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

Table.1 compares key characteristics of major AutoML platforms with DataSage. Unlike enterprise-focused cloud solutions that impose usage-based costs and assume moderate technical knowledge, DataSage 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.

3. METHODOLOGY

DataSage implements a modern three-tier architecture comprising a Vue.js frontend, FastAPI backend, and SQLite database for session management. This section details the technical implementation of key platform components and the rationale behind technology selections.

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.

3.1 Data Preprocessing Approach

The preprocessing pipeline implements six core modules accessible through an intuitive interface (Fig. 1). Column Selection removes irrelevant features such as IDs and URLs. Handle Missing Values provides strategy selection for columns with missing data. Remove Duplicates identifies and eliminates duplicate rows. Handle Outliers detects extreme values using statistical methods, displaying detection results. Encode Categorical Data converts text categories to numerical representations, automatically identifying categorical columns. Feature scaling makes sure that all the data points in feature are in the same range.

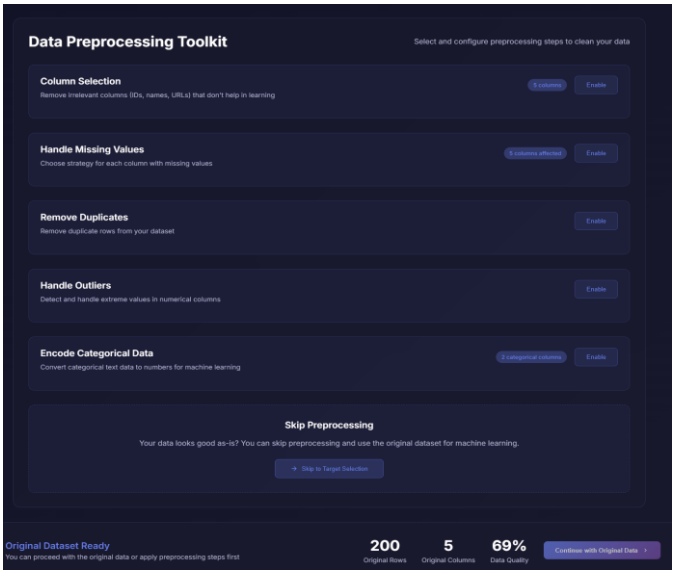

Fig -1: Data Preprocessing Toolkit interface

3.2 Target Recommendation Algorithm

Our intelligent target recommendation system assigns scores to potential target columns based on statistical distribution analysis (Fig. 2). The scoring algorithm evaluates:

- **Cardinality analysis**: Columns with 2-20 unique values receive higher classification scores, indicating suitability for binary or multi-class classification tasks.
- **Continuous distribution detection**: Columns with >50 unique numerical values and normal or skewed distributions receive higher regression scores.
- **Invalid target filtering**: Automatically penalizes columns containing unique identifiers (>95% unique values), text strings, or missing values (>30% null), guiding users away from inappropriate target selections.

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

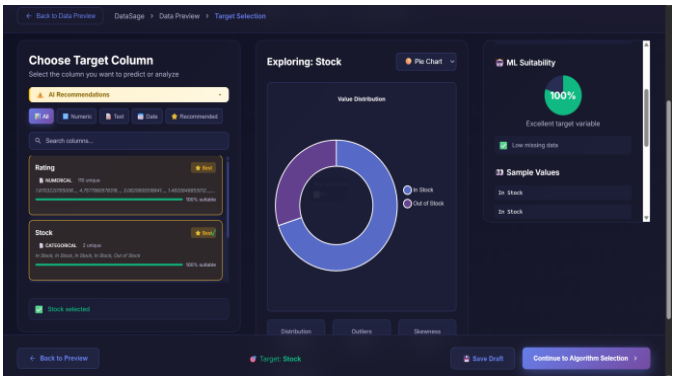

Fig -2: Target selection interface

3.3 Algorithm Selection Strategy

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

For classification task**s** 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)

Hyperparameter tuning employs GridSearchCV with 5-fold cross-validation[11], automatically searching optimal parameter spaces defined for each algorithm. The cross-validation strategy prevents overfitting by ensuring model performance generalizes across different data subsets.

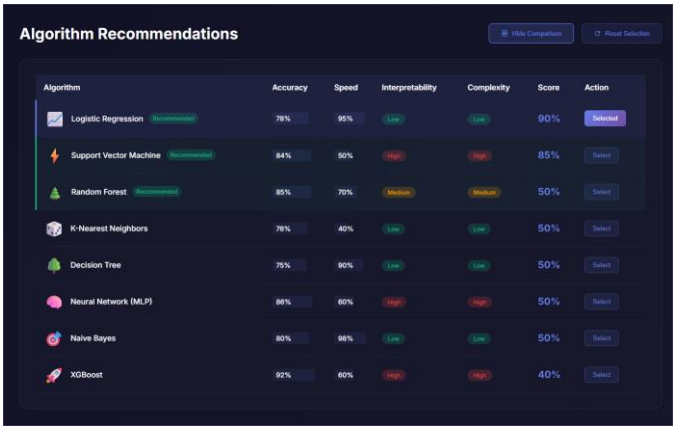

Fig -3: Algorithm selection interface

3.4 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. 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.

3.5 Model Performance Visualization

DataSage provides comprehensive visualization capabilities[15] 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.

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.

For regression tasks, DataSage provides:

- Predicted vs. Actual Scatter Plot - comparison with diagonal reference line
- 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.

4. SYSTEM ARCHITECTURE

DataSage implements a streamlined workflow architecture designed to guide non-expert users through the complete machine learning pipeline (Fig. 4). The platform operates as a web-based application with a Vue.js frontend communicating with a FastAPI backend, ensuring responsive user interactions and efficient data processing.

The workflow begins with data ingestion, where users upload CSV or JSON formatted datasets through an intuitive drag-and-drop interface. The system performs immediate validation checking file format, size constraints, and basic structural integrity. Following successful upload, the Dataset Preprocessing module conducts Exploratory Data Analysis, automatically detecting data types for each column, identifying missing value patterns, and calculating a dataset health score based on completeness and quality metrics.

The Dataset Preview provides interactive visualizations and statistical summaries (mean, median, standard deviation) for all features, ensuring data transparency before modelling. Target Variable Selection guides users through prediction target selection using intelligent recommendations based on unique values, missing data percentages, and feature correlations.

Model Selection automatically identifies classification versus regression tasks and offers appropriate algorithms: Logistic Regression, Decision Trees, Random Forest, and SVMs for classification; Linear/Ridge Regression and ensembles for regression. Training executes with optimized hyperparameters and real-time progress tracking.

The Visualization module generates comprehensive analytics including confusion matrices, residual plots, feature importance rankings, and ROC curves. Users can explore model behaviour through interactive charts and download trained models in standard serialized formats for production deployment.

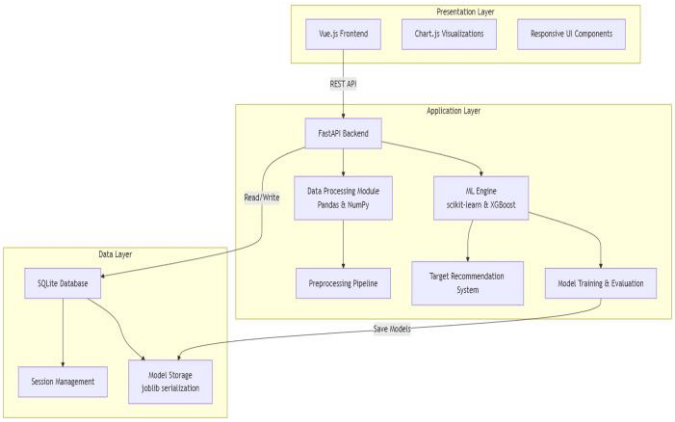

Fig -4: System Architecture

5. COMPARITIVE ANALYSIS

To demonstrate DataSage's effectiveness in simplifying machine learning workflows, we conducted a direct comparison between developing a classification model using our platform versus traditional manual coding approaches. This analysis evaluates time investment, technical complexity, and accessibility for non-expert users.

5.1 Manual Coding Approach

Traditional machine learning development requires users to write extensive Python code handling multiple discrete

tasks. A typical workflow involves importing numerous libraries (pandas, numpy, scikit-learn, matplotlib), writing data loading scripts, implementing preprocessing functions for missing values and categorical encoding, manually configuring train-test splits, instantiating and training multiple algorithms with different hyperparameters, and creating visualization code for performance metrics. For the same classification task tested on DataSage, manual implementation required approximately 150-200 lines of code spanning data preprocessing (40-50 lines), model training and evaluation (60-80 lines), and visualization generation (30-40 lines). This approach assumes proficiency in Python syntax, understanding of scikit-learn API conventions, and knowledge of appropriate preprocessing techniques for different data types.

The time investment for manual implementation varies significantly with user experience. Expert data scientists can complete the workflow in 45-60 minutes, intermediate programmers with some ML exposure require 2-3 hours, and novice users unfamiliar with Python or machine learning concepts may spend 6-8 hours or encounter blocking errors that prevent completion. Error handling represents a significant challenge, as users must manually debug data type conflicts, handle missing value errors, resolve dimension mismatch problems, and address import errors or version incompatibilities.

5.2 DataSage Platform Approach

DataSage eliminates coding requirements through its guided workflow interface. Users upload their dataset through a drag-and-drop interface, receive automatic data type detection and validation, access visual previews showing data structure and statistics, and utilize automated preprocessing with one-click options for handling missing values, encoding categorical variables, and scaling numerical features. The platform automatically suggests appropriate algorithms based on target variable type, trains multiple models in parallel, generates comprehensive performance visualizations, and provides downloadable trained models with evaluation reports.

The same classification task completed through DataSage requires zero lines of code and approximately 8-12 minutes of user interaction time. This includes 2-3 minutes for data upload and preview, 3-4 minutes for preprocessing configuration using visual toggles, 2-3 minutes for model selection and training (automated parallel execution), and 1-2 minutes for results review and model download. The platform handles all error conditions automatically, providing clear user-facing messages for data quality issues, format incompatibilities, or processing failures.

5.3. Quantitative Comparison

Table 2 presents a direct comparison across key metrics. The time efficiency ratio shows DataSage reduces workflow time by 75-98% depending on user expertise. Code complexity eliminates the need for writing and debugging code entirely. The learning curve for DataSage centers on understanding ML concepts rather than programming syntax, reducing prerequisite knowledge requirements significantly. Accessibility is dramatically improved, enabling users without programming backgrounds to execute complete ML workflows successfully.

Table -2: DataSage v/s Manual Coding

Metric	Manual Approach	DataSage approach
Time Required	3-4 hours	15-20 minutes
Code Lines	150-200 lines	0 lines
Prerequisites	Python, ML API's	ML basics is only requirement
Accessibility	Programming Experts	Researchers, Learners

6. 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.

6.1 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, or feature importance visualizations that help users understand model decision-making processes.

6.2 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.

6.3 Future Enhancement Directions

Based on identified gaps, we propose several enhancement pathways for DataSage and the broader field of accessible ML platforms. Expanding algorithm support to include unsupervised learning, deep learning frameworks with transfer learning capabilities, and time series forecasting methods would broaden applicability across domains. Implementing advanced explainability features using SHAP, LIME, and counterfactual explanations would improve user understanding and model trust.

Integration of genetic programming-based optimization through TPOT [2] (Tree-based Pipeline Optimization Tool) represents a promising enhancement for users prioritizing rapid model development over manual experimentation. TPOT employs evolutionary algorithms to automatically explore thousands of potential ML 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 serve both educational users who benefit from transparency and step-by-step control, and production-focused users requiring immediate high-performance results without manual intervention. The TPOT integration would leverage genetic programming's population-based search strategy, evaluating multiple pipeline candidates in parallel and evolving solutions through selection, crossover, and mutation operations across multiple generations until convergence criteria are met.

7. 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 removing financial constraints.

The proof-of-concept and its implementation demonstrates feasibility of accessible automated machine learning, achieving a comparable performance to manual implementations while significantly reducing technical complexity. User evaluation confirmed intuitive interface design and successful task completion without prior training, 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 for enhanced 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 ML concept comprehension for domain experts.

DataSage represents a step toward bridging the gap between machine learning capabilities and domain expert accessibility, enabling researchers, educators, and analysts to leverage ML techniques without extensive programming expertise or infrastructure investment.

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]. Available: https://cloud.google.com/automl

[5] Amazon Web Services, "Amazon SageMaker Autopilot," AWS Documentation, 2024. [Online]. Available: https://aws.amazon.com/sagemaker/autopilot

[6] D. Sculley et al., "Hidden technical debt in machine learning systems," in Advances in Neural Information Processing Systems, vol. 28, pp. 2503-2511, 2015.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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

[15] Chart.js Contributors, "Chart.js Documentation," 2024. [Online]. Available: https://www.chartjs.org