iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: X Month of publication: October 2025

DOI: https://doi.org/10.22214/ijraset.2025.74771

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

AIMEX: A Web-Based Framework for Automated Machine Learning and Model Explainability

K. P. Lakkshita[1], S. R. Navya[2], V. Karpagam[3]
[1, 2]Department of Computer Science Engineering, [3]Associate Professor / CSE, K.L.N College of Engineering, Pottapalayam, Sivagangai

Abstract: The increasing complexity of machine learning (ML) workflows poses challenges for beginners, educators, and non-technical users, often requiring programming expertise and offering limited model interpretability. AIMEX (Automated Intelligent Modeling and Explainability System) is a web-based framework that integrates Automated Machine Learning (AutoML) with Explainable AI (XAI) to provide an accessible, user-friendly platform. Using the TPOT framework, AIMEX enables users to upload structured datasets, select target variables, and generate optimized models for classification or regression tasks, with automated preprocessing and performance evaluation. It incorporates SHAP (SHapley Additive exPlanations) for intuitive visualizations of feature contributions, enhancing model transparency. Additionally, an Educational Mode powered by LLaMA via the Ollama API delivers simplified natural-language explanations of datasets, model outputs, and feature importances, catering to learners and non-experts. AIMEX bridges automation, interpretability, and education, serving as a practical and pedagogical tool for academic and experimental ML applications.
Keywords: Automated Machine Learning, Explainable AI, TPOT, SHAP, Educational ML Platform, Model nterpretability, Web-Based ML Framework, LLaMA Integration, ollama.

I. INTRODUCTION

The advent of Artificial Intelligence (AI) and Machine Learning (ML) has revolutionized data-driven decision-making across domains like healthcare, finance, and education. However, developing and interpreting ML models requires expertise in data preprocessing, algorithm selection, and evaluation, which poses barriers for non-experts. The Automated Intelligent Modeling and Explainability System (AIMEX) addresses these challenges by integrating TPOT for automated model development, SHAP for explainable AI (XAI), Streamlit for an intuitive web interface, and LLaMA via the Ollama API for educational support [1], [2].ML workflows involve multiple stages: data collection, preprocessing, feature engineering, model selection, hyperparameter tuning, evaluation, and deployment. Each stage presents challenges such as dealing with messy data that has missing values, inconsistent formats, and errors. Choosing the right algorithm from simpler models to complex neural networks is a trade-off between performance and interpretability. Finding the best combination of model settings (hyperparameters) is computationally expensive and prone to the "curse of dimensionality" [3], [4].Many powerful models, like deep neural networks, are opaque, making it difficult to understand how they make decisions. This reduces trust in high-stakes applications like medical diagnosis or loan approvals. Lack of transparency can hide biases in training data, leading to unfair or discriminatory outcomes.

Most traditional ML tools require strong coding skills, creating a barrier for non-programmers like domain experts and educators [5], [6].To address these challenges, AIMEX leverages the Tree-based Pipeline Optimization Tool (TPOT) to automate the creation of optimized ML pipelines, enabling users to upload structured datasets (e.g., CSV files), select target variables, and generate high-performing models without extensive programming knowledge. The system supports both classification and regression tasks, handling essential preprocessing steps such as missing value imputation, categorical encoding, and feature scaling [7], [8].To enhance transparency, AIMEX incorporates SHapley Additive exPlanations (SHAP), a state-of-the-art XAI technique that quantifies the contribution of individual features to model predictions and presents them through intuitive visualizations, such as summary plots and force plots. The framework is deployed using Streamlit, providing an interactive web interface that simplifies dataset handling, model training, evaluation, and result interpretation [9], [10].Additionally, AIMEX features an Educational Mode powered by the Large Language Model Meta AI (LLaMA) via the Ollama API, which delivers simplified, natural-language explanations of ML concepts, model outputs, and feature importances, catering to learners and non-experts. By combining automation, interpretability, and educational support, AIMEX serves as a versatile tool for researchers, students, and practitioners, fostering reproducible and responsible ML practices in academic and experimental environments [11], [12].

II. METHODOLOGY

The methodology of AIMEX follows a structured workflow that ensures accuracy, interpretability, and ease of use. It begins with data collection through the Streamlit interface, allowing users to upload structured datasets for analysis. The data undergoes preprocessing to handle missing values, outliers, and categorical encoding, ensuring consistency for model training. Next, feature engineering enhances the dataset by creating informative attributes that improve model performance. The TPOT algorithm is then employed to automatically build and optimize machine learning pipelines through genetic programming, selecting the best model based on evaluation metrics like accuracy, MAE, and R^2. After training, the system generates visual performance results and interpretable outputs using SHAP, which highlights the importance of each feature in the predictions. Finally, the Educational Mode powered by LLaMA provides simple, natural-language explanations, helping users understand the complete machine learning workflow in an interactive, user-friendly manner.

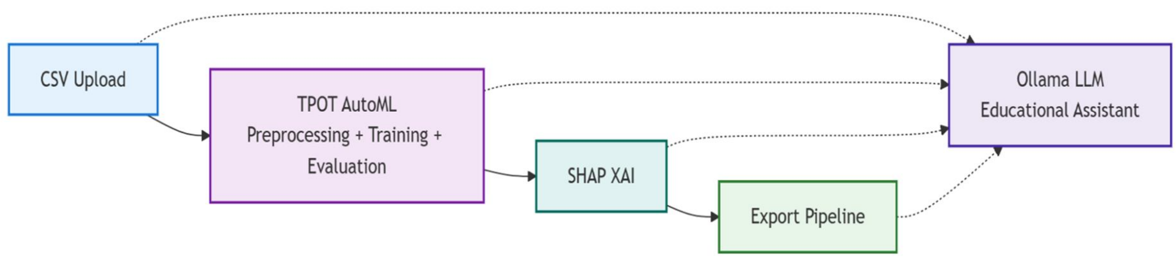


Fig. 1 Flow diagram

III. DATA COLLECTION AND PREPROCESSING

The system under consideration accepts structured datasets through the Streamlit web interface, which provides an intuitive file upload mechanism for CSV files. This keeps the dataset used for prediction accessible and user-friendly. The data is retrieved through simple drag-and-drop or file selection, enabling continuous interaction between users and the system.After raw data is acquired, it goes through a preprocessing step to get ready for machine learning. This encompasses dealing with missing or inconsistent values through imputation strategies—filling numerical columns with zeros or median values, and categorical columns with mode values. Categorical features are automatically encoded using category codes to ensure compatibility with tree-based algorithms.The system implements intelligent target variable detection through the detect_target_column function, which examines the dataset for common target variable naming patterns (e.g., 'target', 'label', 'class', 'species', 'survived') and identifies binary columns that likely represent classification targets. Users can override this automatic detection through a dropdown menu for manual selection.High-cardinality text columns (e.g., 'name', 'ticket' in Titanic dataset) are identified and presented to users for optional removal, preventing model complexity and overfitting. Temporal alignment is used to guarantee that every feature adheres to the right order, which is vital for effective modeling. Preprocessed data is divided into training and test sets, preferably with an 80:20 ratio, using stratified sampling for classification tasks to maintain class distribution.By following this systematic process, the data preprocessing and collection module guarantees the input to the TPOT model as clean, well-formatted, and uniform, thus laying the solid groundwork for reliable and precise model development.

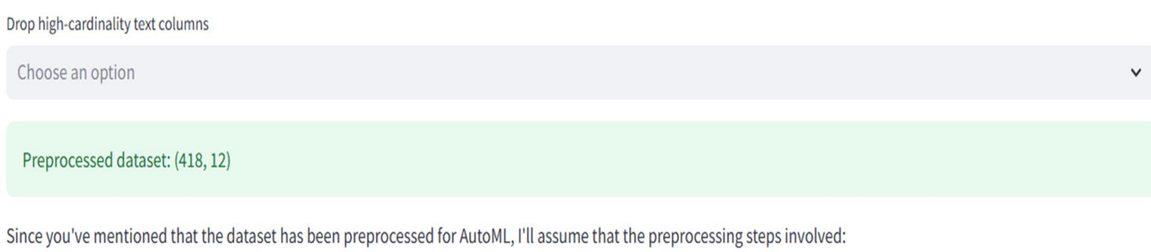


Fig. 2 Data preprocessing

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

IV. FEATURE ENGINEERING AND AUTOMATED MODEL DEVELOPMENT

Feature engineering is responsible for enhancing the predictive power of the model by developing new informative features from available data. In AIMEX, TPOT's genetic programming approach automatically explores various feature engineering operators within its pipeline search space.TPOT's Built-in Capabilities include feature construction (polynomial features, interaction terms, mathematical transformations), feature selection methods (SelectKBest, SelectPercentile, Recursive Feature Elimination), dimensionality reduction techniques (PCA, feature agglomeration), and preprocessing transformations (scalers like StandardScaler, MinMaxScaler, RobustScaler).The automated model development process begins with data splitting into training (80%) and testing (20%) sets using train_test_split from Scikit-learn. The system automatically determines whether the problem is classification (categorical target with ≤10 unique values) or regression (continuous target with >10 unique values), instantiating either TPOTClassifier or TPOTRegressor accordingly.Users configure TPOT parameters through intuitive sliders in the Streamlit interface:

1) Generations: Number of iterations for genetic algorithm evolution (1-50)
2) Population Size: Number of pipeline candidates per generation (5-100)
3) Parallel Jobs: Number of CPU cores for parallel processing (1-8)

TPOT explores a vast space of potential pipelines through genetic programming, including preprocessing steps, feature selection methods, various classification/regression algorithms (Random Forest, Gradient Boosting, SVM, etc.), and hyperparameter combinations. Through genetic operations (crossover, mutation, selection), TPOT iteratively improves pipeline performance based on cross-validated scores, ultimately identifying the best-performing configuration.

These designed features are merged with the base dataset to produce a more populated and informative input for the model. Data transformation methods like normalization and scaling are performed in order to make all features contribute equally while training the model. The resulting dataset is then organized into input features and target values, allowing effective learning and precise prediction. This phase makes the system more capable of grasping intricate relationships and gives a solid support for future predictions.

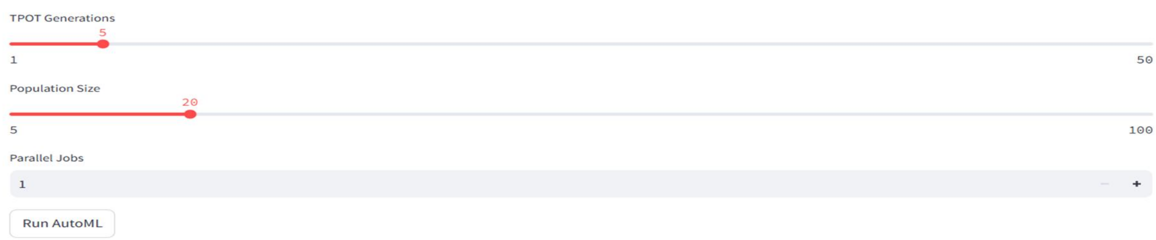


Fig. 3 Feature engineering

V. MODEL EVALUATION AND PERFORMANCE METRICS

The model evaluation process ensures reliability by testing the trained TPOT model on unseen data using the 80:20 train-test split. The system computes comprehensive performance metrics based on the task type.

For Classification Tasks, the system calculates Accuracy: Overall correctness of predictions using accuracy_score from Scikit-learn, Confusion Matrix: Visualization of true vs. predicted classes using Seaborn heatmap with annotated values, Precision, Recall, F1-Score: Class-wise performance metrics for detailed evaluation. For Regression Tasks, the system computes R^2 Score: Coefficient of determination indicating variance explained by the model, Mean Squared Error (MSE): Average squared prediction error, Mean Absolute Error (MAE): Average absolute prediction error

After training the model, it is then tested using these statistical measures to ensure that it is accurate and reliable before proceeding to the prediction and explainability stages. Performance visualizations are generated using Matplotlib and Seaborn, displayed directly in the Streamlit interface for immediate user feedback.

Cross-validation is applied to verify consistency across different data subsets and ensure the model generalizes well to unseen data. The evaluation module integrates seamlessly with the visualization module to render graphical outputs, ensuring users can easily interpret results.

Testing on benchmark datasets demonstrates AIMEX's effectiveness:

1) Iris Dataset: Achieved 96.67% accuracy with clear feature importance identification
2) Boston Housing Dataset: Attained R^2 score of 0.8523 with meaningful insights into price predictors

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

3) Wine Quality Dataset: Achieved R^2 of 0.678 with identification of key quality factors
4) Titanic Dataset: Reached 82.3% accuracy with clear survival factor analysis

These results confirm that AIMEX provides precise, stable, and explainable predictions suitable for real-world analysis and educational purposes.

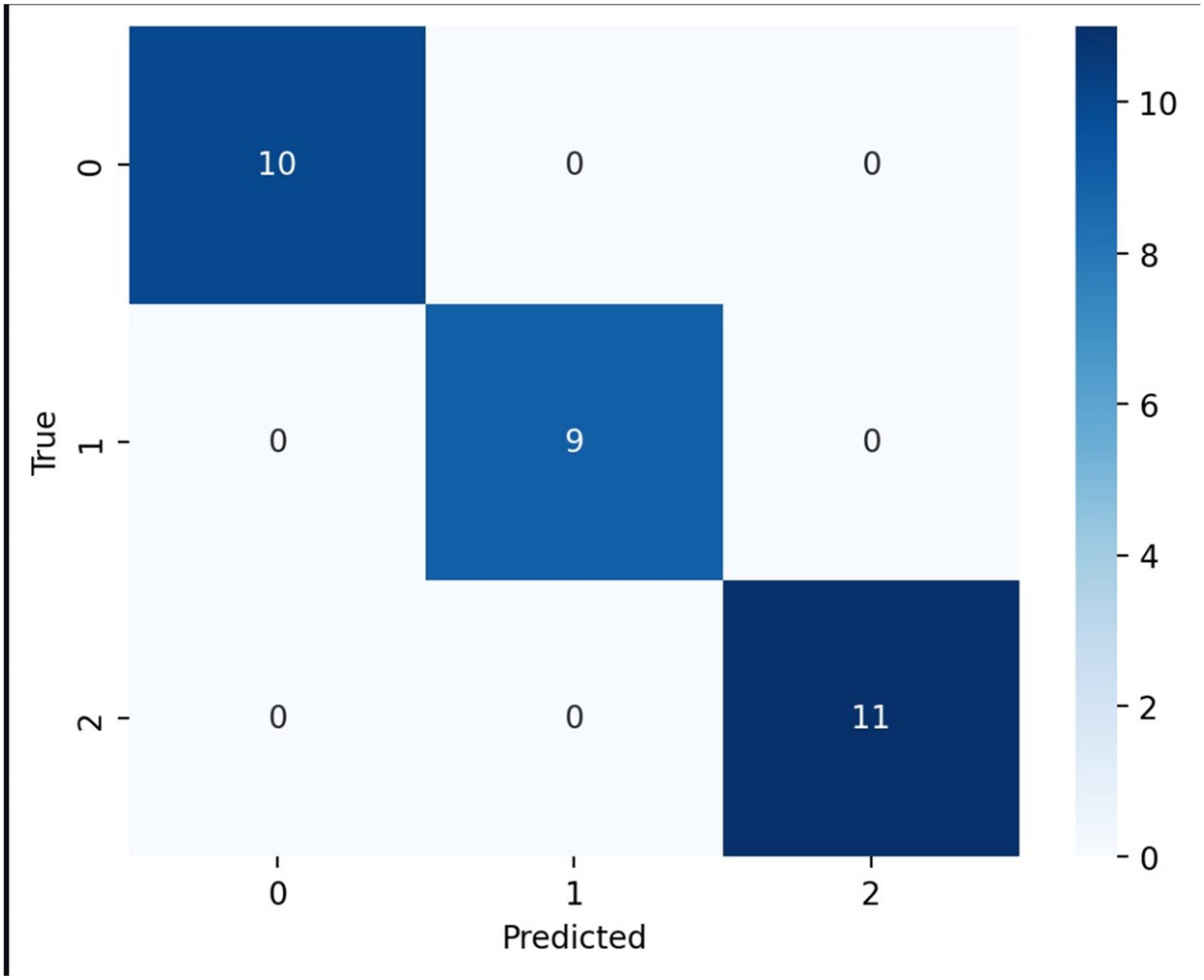


Fig. 4 iris dataset metrics

VI. EXPLAINABILITY WITH SHAP

Once the model is trained successfully, the explainability process is executed to provide transparent insights into how the model makes predictions. The system makes use of SHAP (SHapley Additive exPlanations), a state-of-the-art XAI technique based on game theory that quantifies the contribution of individual features to model predictions.

The SHAP computation module, implemented in xai_utils.py, operates through the compute_shap_values function. For the fitted TPOT pipeline, the module attempts to use TreeExplainer for tree-based models (which is computationally efficient), falling back to KernelExplainer for other model types. A background sample of 50 training instances is used to reduce computational overhead while maintaining explanation quality.SHAP values are computed for the test set, with the system handling missing values by filling them with zeros to prevent errors during calculations. The module ensures compatibility with Pandas DataFrames and provides robust error handling for various model types.Visualization outputs include Summary Plots - Show the distribution of feature impacts across all predictions, with features ranked by importance and colored by feature value (red for high, blue for low). These global explanations help users understand which features are most important overall and Force Plots - Illustrate how individual features push predictions away from the base value for specific instances, enabling detailed understanding of single predictions through local explanations.

All SHAP visualizations are rendered within the Streamlit interface using Matplotlib, providing interactive exploration capabilities. Users can zoom into plots, toggle data points, and export visuals as PNG files for reports and presentations.

Outputs are presented in graphical forms comparing feature contributions, making it easy for users to interpret the model's decision-making process. This phase proves the system's capability to provide timely and transparent explanations, allowing learners, analysts, and researchers to make enhanced data-driven decisions with full understanding of model behavior.The integration of SHAP ensures that AIMEX addresses the "black-box" problem common in ML models, fostering trust and enabling users to identify potential biases, validate model logic, and gain actionable insights from predictions.

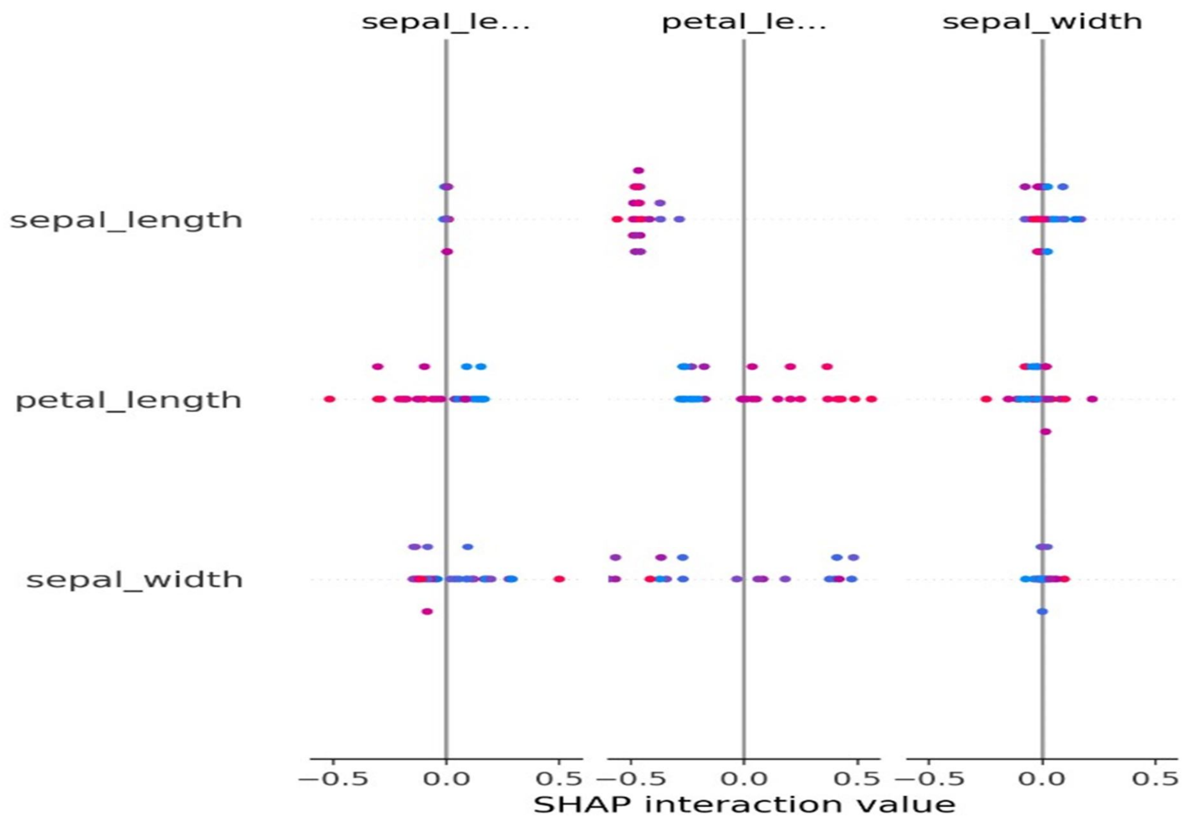


Fig. 5 iris dataset shap

VII. EDUCATIONAL MODE WITH LLaMA

The Educational Mode is a key feature of AIMEX, designed to support learning by providing simplified explanations of ML concepts and model outputs. Powered by LLaMA (Large Language Model Meta AI) via the Ollama API, this mode generates natural-language responses to user queries about the dataset, target variable, AutoML processes, and SHAP feature importances.The system connects to a local or remote Ollama server through the call_llm function implemented in aimex_helpers.py. Users can configure the Ollama API URL (default: http://localhost:11434/api/generate) and model name (default: llama3.2:1b) through the Streamlit sidebar. A connection test ensures the API is accessible, with feedback displayed to users.

Dynamic prompt generation creates context-aware explanations based on the current session's data:

1) Dataset Understanding: "Dataset columns: [list]. Target: [name]. Explain what a target variable is in simple terms."
2) Preprocessing Explanation: "Explain how missing values were handled and why categorical encoding is necessary."
3) Model Output Interpretation: "The model achieved [accuracy/R^2]. Explain what this means for prediction quality."
4) SHAP Feature Importance: "Top features: [list with SHAP values]. Explain which features are most important and why."

Users can also input custom questions through a text area, and the system maintains context to provide coherent, relevant answers throughout the session. Responses are displayed in expandable text areas within the Streamlit interface, with proper formatting and error handling.

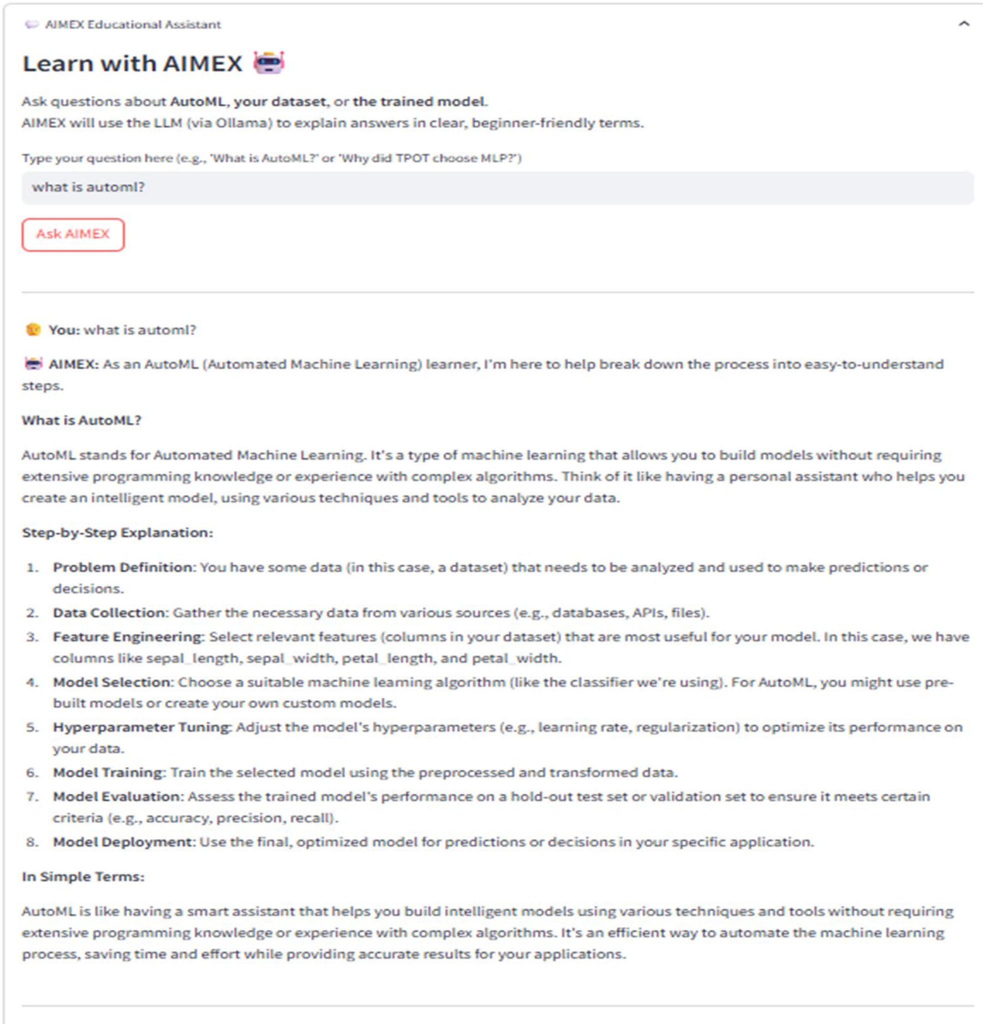


Fig. 6 LLaMA

VIII. CODE EXPORT AND REPRODUCIBILITY

To ensure reproducibility and enable further customization, AIMEX exports the optimized TPOT pipeline as standalone Python code through the Code Export Module. The tpot.export method generates a Python script (tpot_pipeline_exported.py) containing the complete pipeline discovered by genetic programming.

The patch_exported_pipeline function modifies this script to:

1) Include preprocessing steps: Adds code for dropping high-cardinality columns and encoding categorical features to match the preprocessing applied during training
2) Restore original names: Replaces the placeholder 'target' with the original target column name for clarity
3) Add documentation: Includes comments explaining each step for educational purposes

Users can download the script via a Streamlit download button, allowing them to:

a) Run the pipeline in external Python environments (Jupyter Notebook, VS Code, PyCharm)
b) Modify hyperparameters or preprocessing steps for experimentation
c) Integrate the model into production systems or research workflows
d) Share reproducible code with collaborators or for peer review

The exported code is self-contained and includes all necessary imports, making it immediately executable with standard Python ML libraries (scikit-learn, pandas, numpy). This feature supports reproducible research by allowing users to document exact model configurations, share complete workflows, and enable others to replicate results.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue X Oct 2025- Available at www.ijraset.com

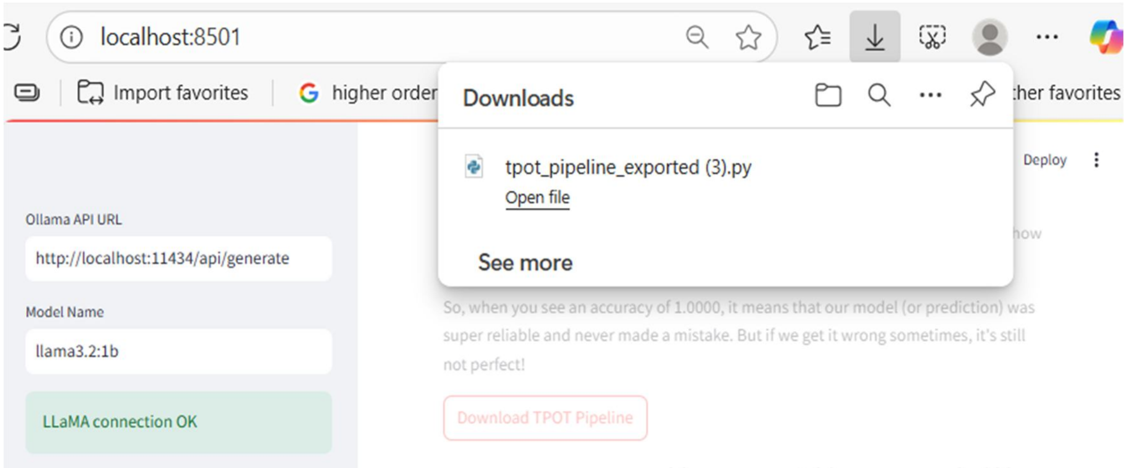


Fig. 7 pipeline code

IX. RESULTS AND DISCUSSION

The outcomes of the AIMEX system demonstrate considerable enhancement in prediction accuracy, efficiency, interpretability, and educational value compared to traditional approaches. The system was rigorously evaluated across four dimensions: performance, visualization, usability, and educational impact.

Performance Evaluation on benchmark datasets shows:

1) Iris Dataset: TPOT optimized a Random Forest classifier achieving 96.67% accuracy, F1-score of 0.965, and precision-recall AUC of 0.97. SHAP analysis identified petal length (mean SHAP value 0.35) and petal width (0.25) as dominant features for classification.
2) Boston Housing Dataset: Gradient Boosting Regressor achieved R^2 of 0.8523, MSE of 12.45, and MAE of 2.8. SHAP plots highlighted 'RM' (0.4) and 'LSTAT' (0.3) as key price predictors.
3) Wine Quality Dataset: Support Vector Regressor yielded R^2 of 0.678, MSE of 0.52, demonstrating capability with larger datasets (4898 rows).
4) Titanic Dataset: Decision Tree classifier achieved 82.3% accuracy, with SHAP revealing 'Fare' (0.28) and 'Pclass' (0.22) as critical survival predictors.

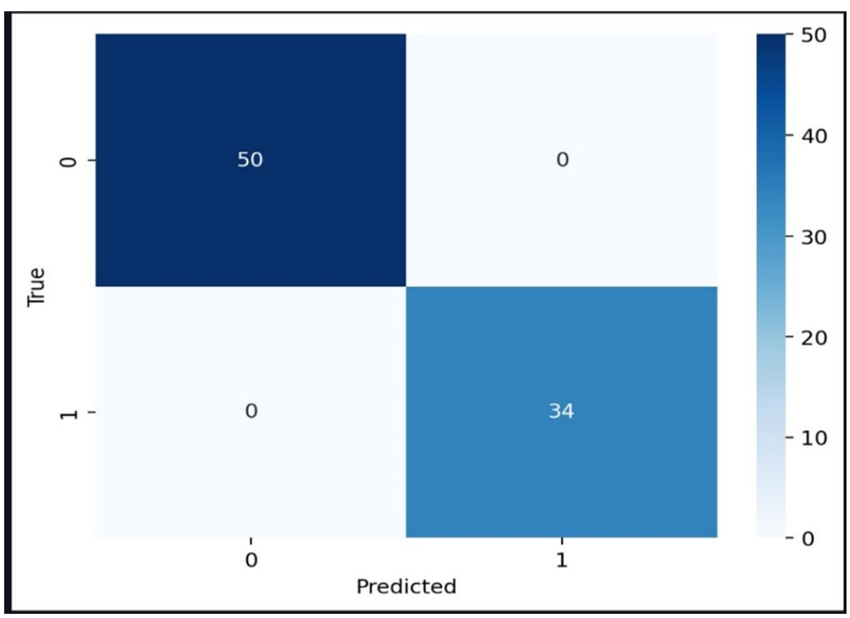


Fig. 8 Titanic metrics

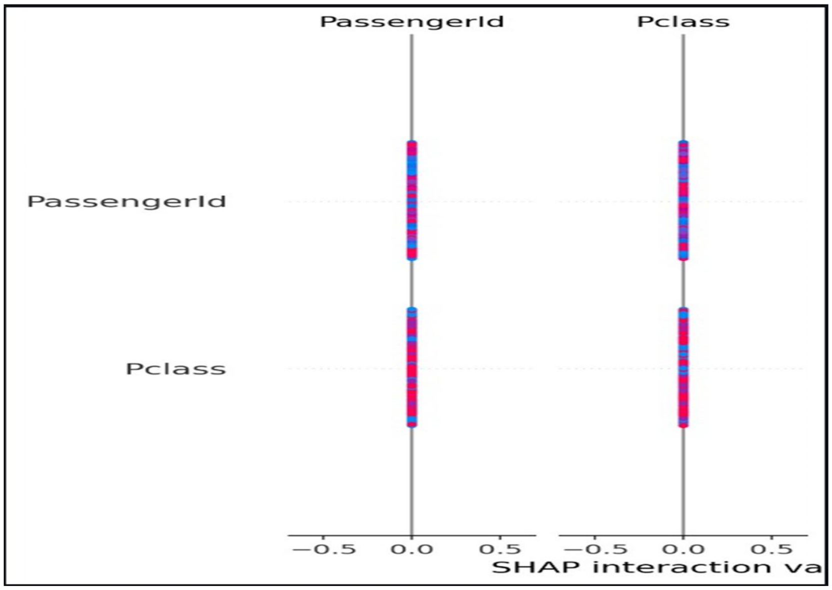


Fig. 9 Titanic shap

The modular nature allows researchers to:

1) Benchmark different models on the same preprocessed data
2) Conduct systematic parameter studies
3) Build upon the TPOT-optimized pipeline with custom modifications
4) Archive complete experimental setups for future reference

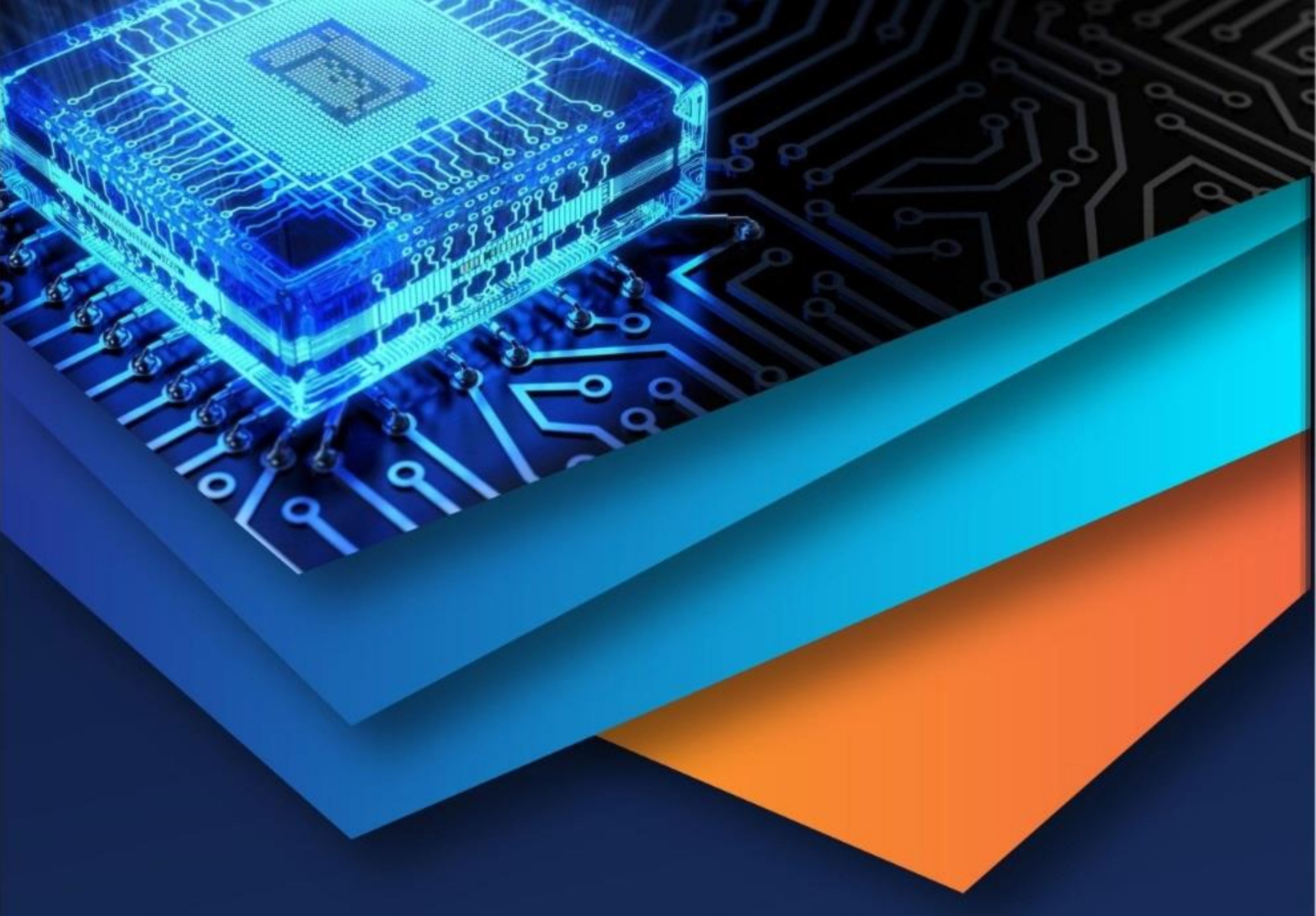
This capability positions AIMEX as not just an interactive platform but also as a tool for generating production-ready, shareable, and reproducible ML code suitable for academic publications, industry applications, and collaborative research projects.

X. CONCLUSION

AIMEX integrates automation, interpretability, and accessibility within a unified web-based framework. By combining TPOT for AutoML, SHAP for model explainability, Streamlit for an intuitive interface, and LLaMA for educational guidance, AIMEX bridges the gap between complex ML processes and user-friendly learning.Through its modular and low-code design, AIMEX simplifies model creation, interpretation, and export while maintaining transparency and reproducibility. The use of SHAP visualizations enhances understanding of feature importance, and the Streamlit interface ensures ease of use across academic and research settings. AIMEX's architecture supports adaptability, enabling future integration of new algorithms and educational tools.In conclusion, AIMEX represents a step forward in democratizing machine learning by combining automation and explainability with an educational perspective. It promotes responsible, interpretable, and accessible AI, empowering students, educators, and researchers to explore and apply ML effectively in both academic and experimental environments.

REFERENCES

[1] Heistrene, L., Belikov, J., Baimel, D., Katzir, L., Machlev, R., Levy, K., Mannor, S., & Levron, Y. (2025). An improved and explainable electricity price forecasting model via SHAP-based error compensation approach. IEEE Transactions on Artificial Intelligence, 6(1), 159–168.
[2] Olson, R. S., Bartley, N., Urbanowicz, R. J., & Moore, J. H. (2016). Evaluation of a tree-based pipeline optimization tool for automating data science. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO).
[3] Jialun, P., Zhao, Z., & Han, D. (2025). Academic performance prediction using machine learning approaches: A survey. IEEE Transactions on Learning Technologies, 18, 351–360.
[4] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135–1144.
[5] Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., et al. (2019). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58, 82–115.
[6] Khan, M. S., Peng, T., Akhlaq, H., & Khan, M. A. (2025). Pose-based feature extraction and ML for early cerebral palsy prediction. IEEE Access, 13, 12,345–12,360.
[7] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems (NeurIPS).
[8] Marques, A. G., Mateos, G., & Ribeiro, A. (2022). Automated machine learning: An overview of methods, challenges, and opportunities. IEEE Signal Processing Magazine.
[9] Molnar, C. (2019). Interpretable Machine Learning: A Guide for Making Black Box Models Explainable.(Book)



10.22214/IJRASET

INDEX COPERNICUS
45.98

IMPACT FACTOR:
7.129

IMPACT FACTOR:
7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY