Wang *et al. J. Mater. Inf.* **2025**, *5*, 51
**DOI:** 10.20517/jmi.2025.45

**Journal of Materials Informatics**

**Research Article**

**Open Access**

Check for updates

# MatSci-ML Studio: an interactive workflow toolkit for automated machine learning in materials science

**Yu Wang[1], Fei Wang[2,*], Guangmao Yan[2], Jun Wang[2], Guodong Niu[1], Jing Feng[2], Jian Mao[1,*], Yan Zhao[1]**

[1]College of Materials Science and Engineering, Sichuan University, Chengdu 610065, Sichuan, China.
[2]Faculty of Materials Science and Engineering, Kunming University of Science and Technology, Kunming 650093, Yunnan, China.

[*]**Correspondence to:** Prof. Jian Mao, College of Materials Science and Engineering, Sichuan University, Chengdu 610065, Sichuan, China. E-mail: maojian@scu.edu.cn; Prof. Fei Wang, Faculty of Materials Science and Engineering, Kunming University of Science and Technology, Kunming 650093, Yunnan, China. E-mail: fwang@kust.edu.cn

**How to cite this article:** Wang, Y.; Wang, F.; Yan, G.; Wang, J.; Niu, G.; Feng, J.; Mao, J.; Zhao, Y. MatSci-ML Studio: an interactive workflow toolkit for automated machine learning in materials science. *J. Mater. Inf.* **2025**, *5*, 51. https://dx.doi.org/10.20517/jmi.2025.45

**Received:** 4 Jun 2025 **First Decision:** 30 Jun 2025 **Revised:** 14 Jul 2025 **Accepted:** 24 Jul 2025 **Published:** 5 Nov 2025

**Academic Editor:** Hao Li **Copy Editor:** Pei-Yun Wang **Production Editor:** Pei-Yun Wang

## Abstract

Machine learning (ML) has become a cornerstone of modern materials science, offering powerful tools for predicting material properties and accelerating experimental workflows. However, its widespread adoption is often hindered by the steep learning curve associated with programming languages such as Python, which presents a significant technical barrier for many domain experts. To address this challenge, we introduce MatSci-ML Studio: an interactive and user-friendly software toolkit designed to empower materials scientists with limited coding expertise. In contrast to traditional code-based frameworks, MatSci-ML Studio features an intuitive graphical user interface that encapsulates a comprehensive, end-to-end ML workflow. This integrated platform seamlessly guides users through data management, advanced preprocessing, multi-strategy feature selection, automated hyperparameter optimization, and model training, democratizing advanced computational analysis for the materials community. Notably, it incorporates advanced capabilities such as a SHapley Additive exPlanations-based interpretability analysis module for explaining model predictions and a multi-objective optimization engine for exploring complex design spaces. The practicality and effectiveness of MatSci-ML Studio are demonstrated through representative case studies, confirming its capacity to lower the technical barrier for ML applications, foster innovation, and significantly enhance the efficiency of data-driven materials science.

**Keywords:** Materials informatics, machine learning, materials science, automation tools, performance prediction

## INTRODUCTION

The proliferation of machine learning (ML) applications has profoundly transformed materials research by providing powerful methodologies for data-driven exploration, prediction, and optimization of material properties[1-3]. ML techniques are particularly effective at processing large-scale experimental and computational datasets, enabling the identification of complex relationships and patterns that might elude conventional analytical methods or trial-and-error experimentation[4,5]. For instance, Hao *et al.* employed the AdaBoost ensemble learning algorithm to analyze a dataset of 461 entries encompassing composition–process–property relationships, constructing both forward (composition → property) and inverse (property → composition) predictive models[6]. They systematically investigated the ultimate tensile strength (UTS) of Al–Si–Cu–Mg–Ni alloys at 300 and 350 °C. By employing polynomial feature engineering and feature selection, their model achieved a prediction accuracy ($R^2$) of 0.94 and a mean deviation of 7.75% for UTS, markedly outperforming single models such as random forest ($R^2$ = 0.84). Experimental validation confirmed that optimized compositions (e.g., Al–12.07Si–0.80Cu–3.70Ni) promoted the precipitation of $Al_9FeNi$ and $Al_3Ni$ phases. These thermally stable phases effectively hindered dislocation motion, yielding a UTS of 163.83 MPa at 300 °C (a 15% improvement over conventional alloys). Mo *et al.* focused on optimizing the strength–ductility balance in Al–Mg–Zn alloys and proposed an active learning strategy[7]. They selected a subset of 45 dual-aging (DA) samples from an initial dataset of 106 entries. To reduce small-sample noise, they employed a bagging ensemble model combined with the Non-dominated Sorting Genetic Algorithm II (NSGA-II) to construct the Pareto frontier between UTS and elongation (EL). Following two rounds of iterative optimization, an alloy with the composition Al–5.27Mg–2.8Zn–0.44Cu–0.19Ag was designed, which achieved a UTS of 602 MPa and an EL of 15.1%, thereby surpassing the traditional strength–ductility trade-off. Liu *et al.* integrated density functional theory (DFT) with ML to calculate the segregation energies of 42 elements at coherent and semi-coherent interfaces[8]. They found that segregation energy at semi-coherent interfaces strongly correlated with atomic size ($r^2$ = 0.659) and Ω-phase solubility. A Gaussian process regression model ($R^2$ = 0.956) further revealed that elements (such as B, Cd, and In) significantly reduced interfacial energy (segregation energy < -0.40 eV) owing to their small atomic radii or strong interfacial affinity, thereby inhibiting precipitate coarsening. These capabilities are particularly advantageous in domains such as alloy design, composite materials engineering, and materials characterization, which routinely involve large datasets and complex variable interactions.

Despite the considerable advantages of ML, its widespread adoption among materials researchers has been constrained by the steep learning curve associated with advanced programming skills. Proficiency in Python, the predominant language for data science, presents a substantial barrier for researchers whose expertise traditionally lies outside computer science. To address these challenges, several noteworthy tools have been developed to automate and streamline ML workflows in materials science. Frameworks such as Automatminer and MatPipe have emerged as powerful Python-based libraries that specialize in automating the process of featurization and model benchmarking[9]. These code-centric tools are invaluable for computational materials scientists requiring rapid feature generation from composition or structure and high-throughput model benchmarking. Similarly, foundational libraries such as Magpie offer robust command-line functionalities for generating a vast array of physics-based descriptors from elemental properties[10], and a detailed comparison of the features of these prominent frameworks is provided in Table 1.

Despite the power of these code-centric frameworks, a significant accessibility gap remains. Tools such as Automatminer and Magpie are fundamentally designed for computational scientists and require a strong programming background. Consequently, their reliance on application programming interfaces (APIs) and script-based execution presents a substantial barrier for many experimental materials scientists who could

**Table 1. Detailed comparison of MatSci-ML Studio with other prominent materials informatics automation frameworks[9]**

| Aspect feature | MatSci-ML Studio (our work) | Automatminer | MatPipe | Magpie | Data platforms (e.g., Materials Project) |
|---|---|---|---|---|---|
| **1. Core paradigm and target audience** | | | | | |
| Primary interaction model | G | C | C | C | G |
| Primary target audience | Domain experts | Programming experts | Programming experts | Programming experts | Domain experts |
| **2. Workflow and project management** | | | | | |
| Visual workflow builder | G | - | - | - | - |
| Integrated project management system | G | - | - | - | - |
| Version control/snapshotting | G | - | - | - | - |
| Undo/redo in preprocessing | G | - | - | - | - |
| **3. ML pipeline** | | | | | |
| Featurization from composition/structure | - | C | C | C | C |
| Data quality and preprocessing | G | C | C | C | - |
| Hyperparameter optimization | G | C | C | C | - |
| Model training and validation | G | C | C | C | - |
| Model benchmarking and comparison | - | C | C | C | - |
| **4. Advanced analysis and design** | | | | | |
| SHAP interpretability | G | C | C | C | - |
| Target optimization (inverse design) | G | C | C | C | - |
| Multi-objective optimization | G | C | C | C | - |
| Active learning | G | C | C | C | - |

G: GUI-Driven. The feature is primarily accessed and controlled through a graphical user interface, designed for direct interaction with minimal to no coding. C: Code-Driven. The feature is primarily accessed programmatically via an API or command-line scripts. This approach offers high flexibility for users with programming skills. -: Not a primary focus of the tool's intended scope. ML: Machine learning; SHAP: SHapley Additive exPlanations.

otherwise benefit from ML. Overcoming this barrier necessitates a new class of user-friendly tools that replaces complex coding with an intuitive, visual, and interactive experience. A solution that minimizes coding requirements without compromising analytical power is therefore essential to democratize ML for the broader materials science community.

This work introduces and validates MatSci-ML Studio, a novel, code-free software toolkit developed to bridge this accessibility gap. We hypothesize that by integrating the entire ML workflow - from data ingestion and intelligent preprocessing to model training, interpretation, and inverse design - into a single, intuitive graphical user interface (GUI), the technical barrier for materials scientists can be significantly lowered. This paper first introduces the architecture and key functionalities of MatSci-ML Studio. We then demonstrate its practicality and effectiveness through two distinct case studies that serve as validation: a materials-centric regression task and a general classification task. Through this work, we aim to provide a robust platform that not only alleviates the technical burden on researchers but also fosters greater innovation in data-driven materials science.

## MATERIALS AND METHODS

### Design philosophy and scope

To realize the objective of creating an accessible yet powerful ML platform, MatSci-ML Studio was designed based on a clear philosophy. The primary goal is to empower materials scientists and domain experts who have limited programming backgrounds by automating the most laborious and technically demanding stages of the data-driven workflow.

The toolkit is specifically engineered to handle structured, tabular datasets, such as those used for modeling composition-process-property relationships. MatSci-ML Studio is not intended as a universal solution for all data types; it does not, for instance, natively process unstructured data such as microstructural images or spectra without prior, user-led feature extraction. Its core strength lies in providing a robust, end-to-end pipeline for the rich information contained within well-structured tables.

### The integrated workflow architecture

The software's modular architecture, presented in Figure 1, logically encapsulates each stage of the ML pipeline into a series of interconnected, user-friendly tabs. Built using PyQt5[11], the GUI systematically guides researchers through the entire discovery process. The core system is organized around the following key modules, each designed to address a specific challenge in the materials informatics workflow.

### Project management and collaboration

A core pillar of MatSci-ML Studio is its robust project management system, which is designed to ensure methodological rigor and enhance reproducibility. The module facilitates the creation of dedicated projects, each organized into a structured directory that isolates data, models, results, and configuration metadata. Building on this foundation, MatSci-ML Studio incorporates a critical version control feature, enabling users to create timestamped "snapshots" of the entire project state. This capability captures the exact data, preprocessing steps, and model parameters utilized at any given point. This allows researchers to readily revert to previous stages or compare different experimental workflows, thereby guaranteeing full traceability.

Export and Share: Projects can be exported as a single, self-contained archive, which simplifies sharing of complete workflows and findings with collaborators.

### Data management and quality assessment

The workflow commences with the data management module, which provides intuitive tools for data ingestion and initial exploration.

Flexible Data Import: The module supports loading data from various common formats, including CSV, Excel (.xlsx, .xls), and directly from the system clipboard, accommodating diverse data sources.

Initial Data Overview: Upon loading, the software automatically generates a statistical summary - including data dimensions, data types, missing value counts, and a preview table - providing an immediate snapshot of the dataset's characteristics.

### Advanced preprocessing with an intelligent assistant

Recognizing that data preprocessing is a critical and often complex step, a dedicated module with semi-automated, interactive capabilities has been developed.

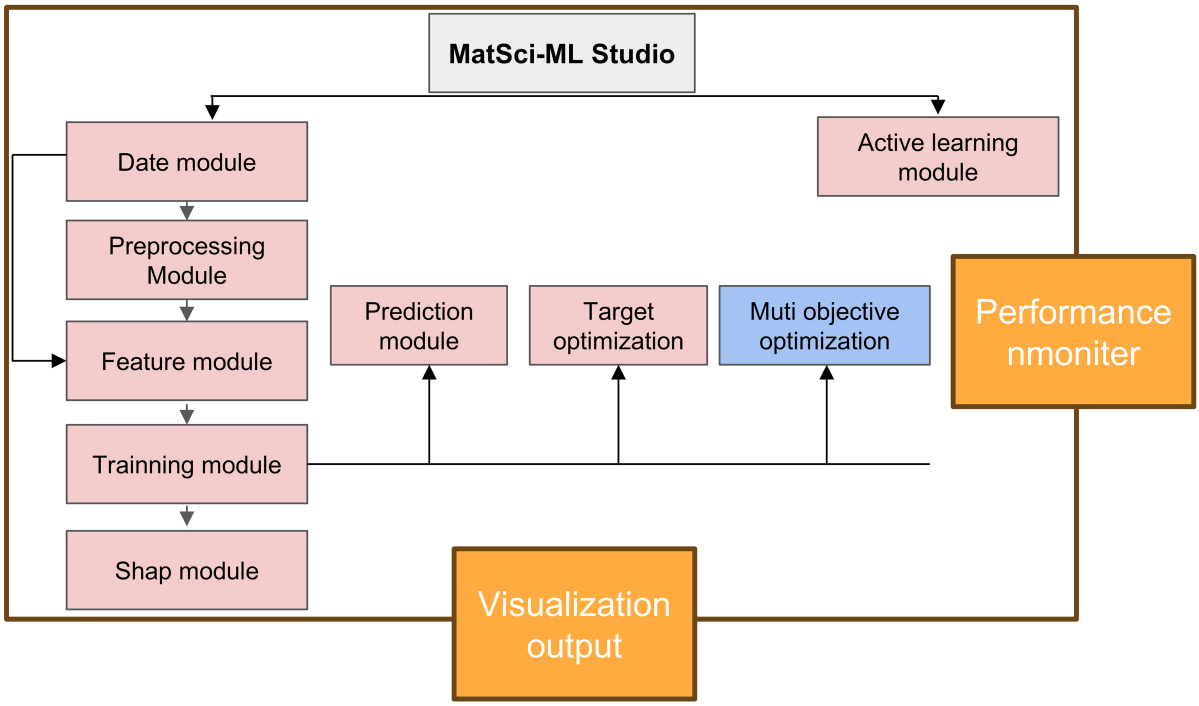


**Figure 1.** Flowchart for the MatSci-ML Studio. ML: Machine learning.

Intelligent Data Quality Analyzer: This core component performs a multi-dimensional analysis of the dataset, evaluating completeness, uniqueness, validity, and consistency. It generates an overall data quality score and provides a prioritized list of actionable recommendations for remediation.

Interactive Cleaning Tools: The GUI provides tools for handling missing data and outliers. Users can select from a range of algorithms - from simple statistical methods (mean, median) to advanced techniques such as KNNImputer, IterativeImputer, and Isolation Forest - and preview their effects before application.

State Management and Reversibility: A built-in StateManager tracks every preprocessing operation. This provides full undo/redo functionality, empowering users to experiment with different cleaning strategies without the risk of irreversible changes.

### Feature engineering and selection

This module offers a comprehensive suite of tools for feature engineering and selection, processes that are particularly crucial in materials science.

Multi-Strategy Feature Selection: A multi-stage feature selection workflow is implemented, allowing for a systematic reduction of dimensionality:
Importance-based Filtering: Utilizes model-intrinsic metrics (e.g., .feature_importances_, .coef_) for rapid, initial feature filtering.Correlation-based Filtering: Identifies and removes highly correlated features based on a user-defined threshold, while prioritizing the preservation of features that are more important for model performance.
Advanced Wrapper Methods: For more rigorous selection, users can employ advanced search algorithms, including genetic algorithms (GA) and recursive feature elimination (RFE), which evaluate feature subsets based on model performance.

### Model training and hyperparameter optimization

The training module streamlines the process of constructing and validating predictive models.

Broad Model Library: It integrates a wide array of models from Scikit-learn[12], eXtreme Gradient Boosting (XGBoost)[13], Light Gradient Boosting Machine (LightGBM)[14], and Categorical Boosting (CatBoost)[15], supporting both regression and classification tasks. This diverse library was curated to include models, such as gradient boosting machines (XGBoost, LightGBM), known for their state-of-the-art performance on the structured, tabular data for which MatSci-ML Studio is designed.

Automated Hyperparameter Optimization: Hyperparameter tuning is automated using the Optuna library[16], which employs efficient Bayesian optimization to identify optimal model configurations. Optuna was selected for its efficient Bayesian optimization and pruning algorithms. These are crucial for automating this otherwise complex and computationally expensive task, thereby making best-practice modeling accessible to non-experts.

Robust Evaluation: The module enforces best practices by performing all training and evaluation within a cross-validation (CV) framework to prevent data leakage and provide reliable performance estimates. It automatically generates key metrics and visualizations, such as confusion matrices, receiver operating characteristic (ROC) curves, and residual plots.

### Model interpretability via SHapley Additive exPlanation analysis (shap_analysis.py)

To move beyond opaque "black-box" models, an integrated SHapley Additive exPlanations (SHAP) analysis module provides enhanced model interpretability.

Automated SHAP Calculation: Upon user request, the module calculates SHAP values to quantify the contribution of each feature to individual predictions. SHAP was selected due to its robust game-theoretic foundation and model-agnostic nature. This allows for a unified and reliable approach to interpreting any model within the library, which is a key requirement for building trust and generating scientific insight.

Interactive Visualizations: It automatically generates a suite of SHAP plots, including:Summary (Beeswarm) Plots: For a global view of feature importance and impact.Dependence Plots: To uncover complex, non-linear relationships between features and model output.Waterfall and Force Plots: For explaining individual predictions, enhancing local interpretability.

These visualizations are available both within the main interface and in separate, interactive windows.

### Prediction, optimization, and active learning

MatSci-ML Studio provides a suite of advanced modules for the application of trained models.

Prediction: Users can load a trained model and apply it to new, unseen data, either by importing a file or via manual input for single-point predictions.Target Optimization: For inverse design problems, this module employs algorithms such as GA and Particle Swarm Optimization to find the optimal feature combinations that maximize or minimize a predicted target property.Multi-Objective Optimization: Addressing the common need to balance competing properties (e.g., strength *vs.* ductility), this module uses algorithms such as NSGA-II, Non-dominated Sorting Genetic Algorithm III (NSGA-III)[17,18], Improved Strength Pareto Evolutionary Algorithm (SPEA2)[19], Multi-objective Evolutionary Algorithm Based on Decomposition (MOEA/D)[20], Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES)[21],

Differential Evolution[22], to identify the Pareto-optimal front, providing a set of non-dominated solutions. The inclusion of these powerful, well-established evolutionary algorithms directly addresses the complex, multi-faceted nature of real-world materials design problems. This enables users to move beyond single-property prediction toward practical, trade-off-aware optimization.Active Learning: To guide efficient experimental design, the active learning module uses Bayesian optimization to suggest the next most informative experiment to perform. It intelligently balances exploration (reducing model uncertainty) with exploitation (improving the target property).

### Performance monitoring and resource management

To ensure a stable and efficient user experience, the software incorporates a real-time performance monitor. This module tracks system resources, such as central processing unit (CPU) and memory utilization, and provides real-time feedback on task progress. It also issues alerts to prevent system overload, which is particularly critical during computationally intensive tasks such as hyperparameter optimization or feature selection.

## RESULTS AND DISCUSSION

The practicality and effectiveness of MatSci-ML Studio are validated in this section through two representative case studies: a regression task focused on predicting the strength of aluminum alloys, and a classification task.

### Case 1: strength prediction of aluminum alloys

In alloy property prediction, raw compositional data are seldom used directly as input features because they lack sufficient physical and chemical information[4,23]. Elemental percentages alone cannot capture critical factors such as thermodynamic interactions, atomic size mismatches, and electronic structure variations among constituent elements. To address this limitation, four feature mapping strategies were adopted to construct corresponding alloy descriptors, as detailed in previous studies[24-26]:

$$x_1 = \sum_{i=1}^{N} a_i x_i \tag{1}$$

$$x_2 = \left(\sum_{i=1}^{N} \frac{a_i}{x_i}\right)^{-1} \tag{2}$$

$$x_d = \sqrt{\sum_{i=1}^{N} a_i \left(1-\frac{x_i}{x_1}\right)^2} \tag{3}$$

$$x_r = max\left[a_i\left(1-\frac{x_i}{x_1}\right)^2\right] - min\left[a_i\left(1-\frac{x_i}{x_1}\right)^2\right] \tag{4}$$

where $a_i$ represents the atomic fraction of the $i$-th element, and $x_i$ denotes a specific elemental property (e.g., atomic radius or electronegativity). $N$ is the total number of elements in the alloy. Four types of descriptors are derived: the weighted average ($x_1$), the harmonic mean ($x_2$), the weighted variance ($x_d$), which reflects property dispersion, and the range of normalized squared deviations ($x_r$), which quantifies the degree of property mismatch among elements. In total, 66 elemental properties were considered, resulting in 264 input features and one target variable (UTS). These descriptors are widely adopted in feature engineering for materials informatics.

The automated workflow within MatSci-ML Studio and its comprehensive outputs are illustrated in Figure 2. To manage the high-dimensional input space of 264 features, the process begins with an automated feature importance analysis. Specifically, a cross-validated XGBoost model is employed to rank all features by their predictive power, with the results visualized in Figure 2A. This initial step immediately

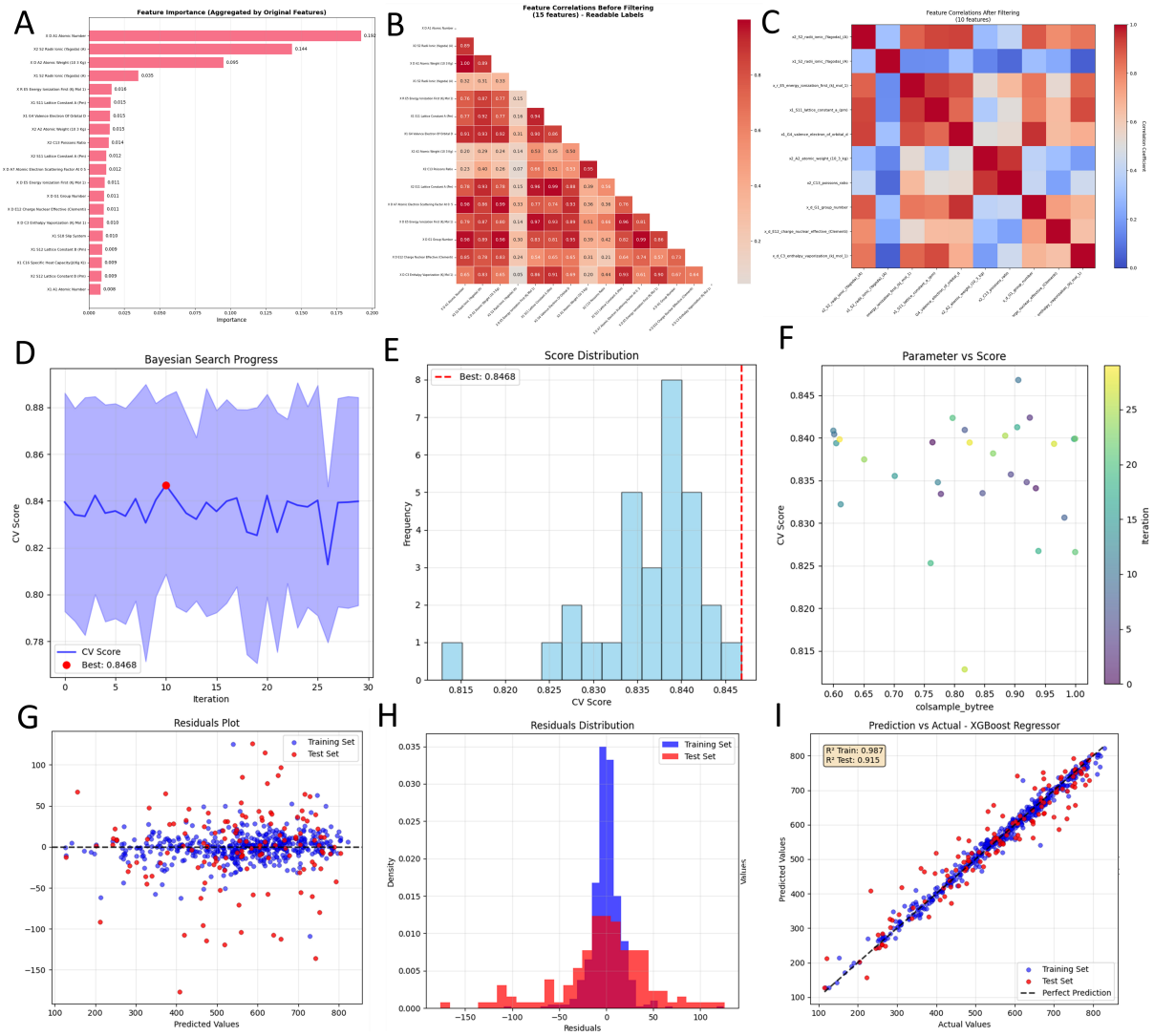


**Figure 2.** Comprehensive visualization of the automated regression workflow in MatSci-ML Studio for predicting aluminum alloy strength. (A) Feature importance plot ranking the most influential descriptors based on their predictive power. Error bars indicate the standard deviation of importance scores across CV folds; (B) Initial correlation matrix revealing multicollinearity among the 264 engineered features; (C) Correlation matrix of the refined subset of 13 features after automated filtering, displaying significantly reduced inter-correlation; (D) Progress of Bayesian hyperparameter optimization over 30 iterations, plotting the CV $R^2$ score and indicating the best score achieved (red dashed line); (E) Distribution of CV scores from the optimization search, confirming the stability of the optimal parameter set; (F) Parameter *vs.* score plot for the colsample_bytree hyperparameter, illustrating the optimization search path; (G) Residuals plot for the final model, showing the difference between predicted and actual values for both training (blue) and test (red) sets; (H) Distribution of residuals, centered around zero, indicating an unbiased model; (I) Predicted *vs.* actual UTS values for the training ($R^2$ = 0.987) and test ($R^2$ = 0.915) sets. The dashed black line represents perfect prediction (y = x). ML: Machine learning; CV: cross-validation.

provides valuable domain-specific insight, revealing that descriptors related to ionic radii [e.g., x2_S2_radii_ionic_(Yagoda)_(A)] are the most influential factors in determining UTS. The software then addresses the common issue of multicollinearity, visualized in the initial correlation heatmap [Figure 2B], where dense clusters of highly correlated features are evident. By applying a performance-aware correlation filtering algorithm (Pearson's r > 0.95), the system prunes redundant features. This reduces the feature set to a robust subset of 13 descriptors, a result confirmed by the significantly sparser correlation matrix in Figure 2C.

With a refined feature set, the workflow seamlessly transitions to hyperparameter optimization. MatSci-ML Studio utilizes a Bayesian search strategy, and its progress over 30 iterations is illustrated in Figure 2D. The plot reveals a clear convergence toward an optimal hyperparameter configuration, where the best CV $R^2$ score reached approximately 0.8468. The score distribution across all trials [Figure 2E] and the parameter-versus-score plot [Figure 2F] further validate the stability and effectiveness of the optimization process, confirming that the system successfully navigated the hyperparameter landscape to identify a high-performing model configuration.

The final stage involves training the optimized XGBoost model on the selected features and evaluating its performance on a held-out test set. The predictive accuracy is powerfully illustrated in the predicted-versus-actual plot [Figure 2I], where the model achieves an outstanding $R^2$ of 0.915 and a root mean square error (RMSE) of 47.7 MPa on the test set. This performance closely mirrors the training set metrics ($R^2$ = 0.987, RMSE = 26.1 MPa), indicating strong generalization capabilities. Furthermore, the residuals plot and its corresponding distribution [Figure 2G and H] exhibit a random pattern centered around zero, confirming that the model has captured the underlying data trends without systematic bias. These results are consistent with those reported by Jiang *et al.*, who achieved an $R^2$ of 0.93 on the same test dataset using a comparable methodology[25]. This validates that the MatSci-ML Studio automated pipeline can construct high-performing regression models that are competitive with manually curated workflows.

Beyond predictive accuracy, MatSci-ML Studio provides deeper scientific insight via an automated SHAP analysis [Figure 3]. The global feature importance plot [Figure 3A], which ranks features by their mean absolute SHAP value, offers a model-agnostic confirmation of the primary physical drivers of UTS. This validates that descriptors related to atomic and ionic properties - notably ionic radii [x2_S2_radii_ionic_(Yagoda)_(A)] and lattice constant [x1_S11_lattice_constant_a_(pm)] - are the most influential factors. Moving from global importance to local explanations, the SHAP summary plot [Figure 3B] visualizes how each feature's value impacts individual predictions. For instance, it reveals a clear positive correlation for x2_S2_radii_ionic_(Yagoda)_(A), where higher feature values (red points) consistently yield positive SHAP values, indicating a contribution to increased strength. Conversely, a feature such as x_r_E5_energy_ionization_first exhibits an inverse trend, where lower values (blue points) positively impact the predicted strength. Furthermore, the plot uncovers complex interaction effects; the wide vertical dispersion of SHAP values for x1_S11_lattice_constant_a_(pm) suggests that its influence is highly context-dependent and modulated by other features. This dual-level interpretability, generated automatically, empowers researchers not only to trust model predictions but also to formulate new hypotheses about the underlying material physics.

### Case 2: lung cancer prediction

To demonstrate the framework's versatility beyond regression and its capability to handle heterogeneous data, MatSci-ML Studio was applied to a publicly available lung cancer prediction dataset. This dataset contains a mix of categorical (e.g., "SMOKING") and numerical features, making it an excellent test case for our automated preprocessing and classification pipeline. The complete set of results is summarized in Figure 4.

The workflow, similar to Case 1, commences with feature analysis. The feature importance plot [Figure 4A], which aggregates the impact of one-hot encoded variables, immediately identifies SMOKING as the most critical predictive factor. The initial correlation heatmap [Figure 4B] reveals very low multicollinearity among the features, suggesting that most provide independent information. Consequently, the automated correlation filtering step [Figure 4C] retains the majority of the features, a decision

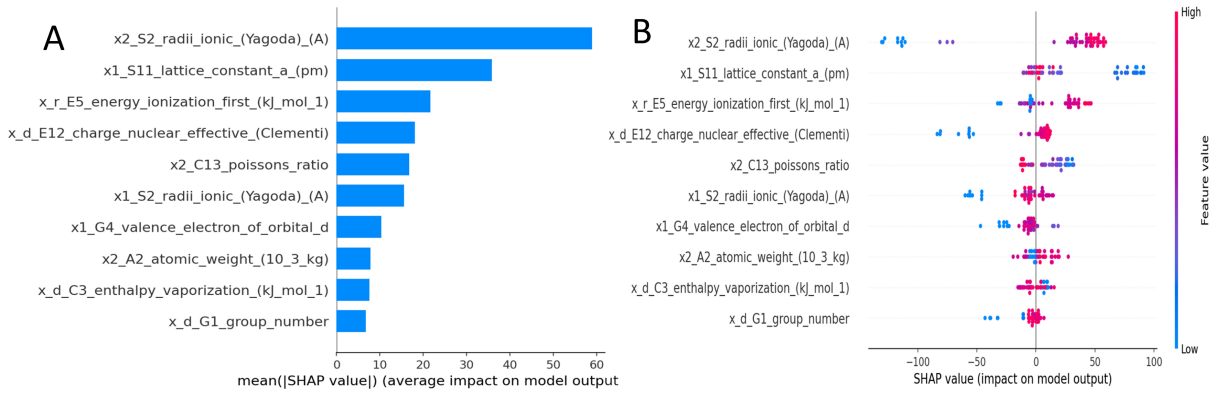


**Figure 3.** SHAP analysis results for interpreting the final XGBoost model predicting aluminum alloy strength. (A) Global feature importance ranked by mean absolute SHAP value, indicating the average magnitude of impact on model predictions; (B) SHAP summary plot (beeswarm) illustrating the distribution, direction (positive/negative SHAP value), and magnitude of impact for each feature across individual predictions, colored by the feature's original value (red = high, blue = low). SHAP: SHapley Additive exPlanations; XGBoost: eXtreme Gradient Boosting.

intelligently made by the system based on the data's statistical properties.

Even with relatively independent features, hyperparameter optimization remains crucial for maximizing model performance. MatSci-ML Studio's automated Bayesian search process, visualized in Figure 4D, effectively maximizes the F1 score, converging to a peak CV score of approximately 0.9123. The stability of this optimum is confirmed by the score distribution plot [Figure 4E], while Figure 4F offers a granular view of the search, plotting the CV score against the min_samples_leaf hyperparameter. The final evaluation of the optimized classifier on unseen test data showcases its excellent performance. The ROC curve [Figure 4G], yields an area under the curve (AUC) of 0.917, indicating excellent class separability. This is corroborated by the Precision-Recall curve [Figure 4H], which demonstrates a high average precision of 0.885, particularly important for imbalanced datasets. The confusion matrix [Figure 4I] details the model's performance, recording 544 true negatives and 356 true positives against a relatively small number of misclassifications.

In summary, the automated workflow achieved an accuracy of 90% and an F1-score of 88%, which represents excellent predictive performance on this dataset. These results closely align with those reported by Ahmed [Figure 5], further validating the applicability and reliability of the proposed framework[27].

To ensure the final classifier is not merely a "black box", MatSci-ML Studio integrates an automated SHAP analysis to provide transparent and interpretable insights into the model's decision-making process. The results, shown in Figure 6, illuminate the key factors driving the lung cancer predictions. The global feature importance plot [Figure 6A], ranked by the mean absolute SHAP value, unequivocally identifies smoking-related features as the most dominant predictors. The one-hot encoded features SMOKING_0 (non-smoker) and SMOKING_1 (smoker) exert the largest average impact on the model's output, far surpassing other factors such as ENERGY_LEVEL and THROAT_DISCOMFORT_0. This provides immediate, quantifiable evidence of the most critical risk factors learned by the model from the data.

However, the SHAP summary plot [Figure 6B] provides a richer, more directional understanding. Each point on this beeswarm plot corresponds to a single prediction, colored by the feature's value (red for present/high, blue for absent/low). This visualization clearly reveals the learned relationships. For instance,

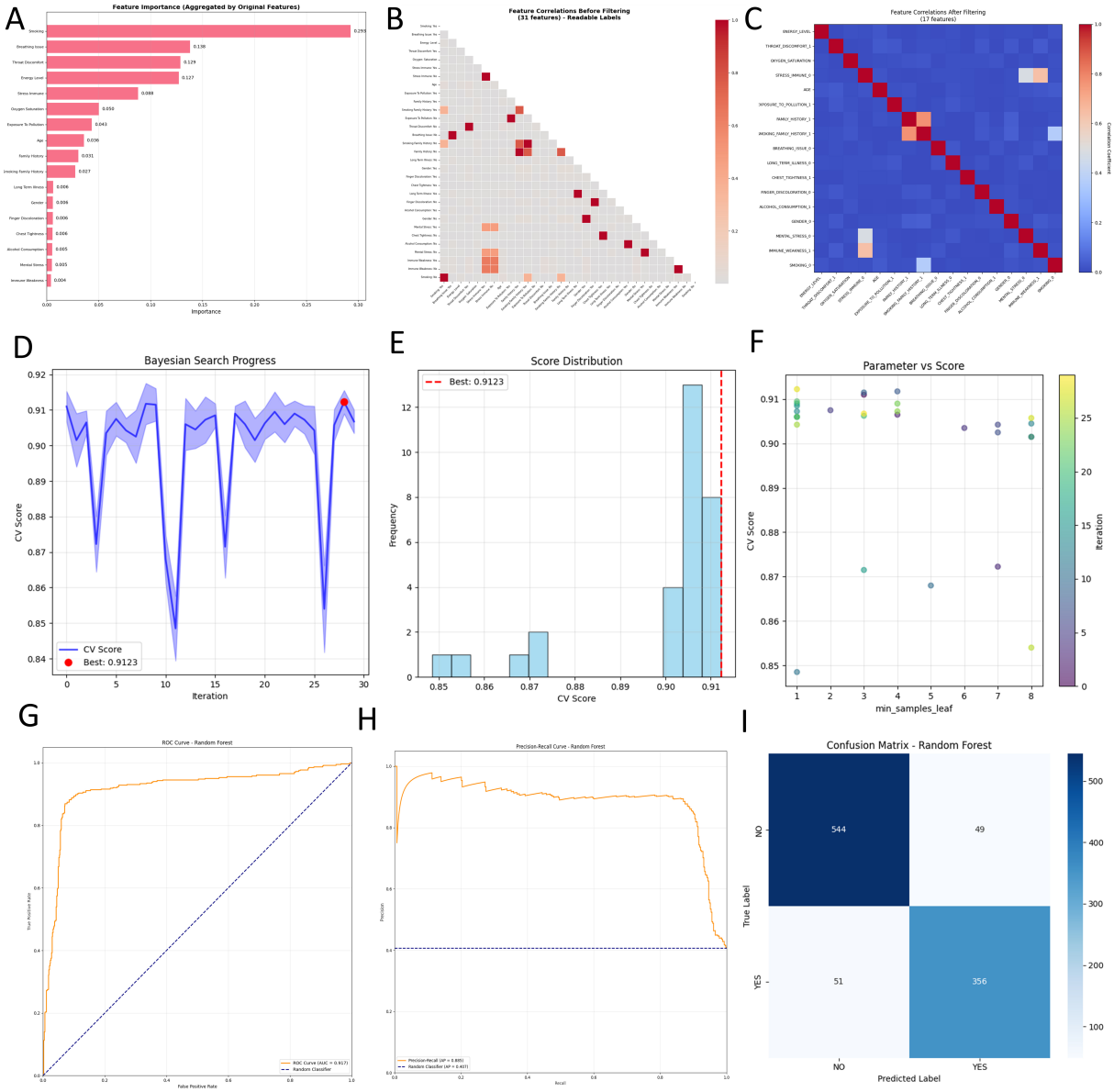


**Figure 4.** Complete workflow visualization generated by MatSci-ML Studio for the lung cancer prediction (classification) task. (A) Feature importance plot, where SMOKING is identified as the most influential factor. The importance of one-hot encoded features has been aggregated back to the original categorical feature; (B) Initial correlation matrix showing low inter-correlation among features; (C) Correlation matrix after filtering, demonstrating that most original features were retained; (D) Progress of Bayesian hyperparameter optimization maximizing the F1-score, which converges to a best value of approximately 0.9123; (E) Score distribution from the hyperparameter search; (F) Relationship between the min_samples_leaf hyperparameter and the CV score; (G) ROC curve for the final model on the test set, achieving an AUC of 0.917; (H) Precision-Recall curve, showing a high average precision (AP = 0.885); (I) Confusion matrix for the test set, detailing the classification performance (TN = 544, FP = 49, FN = 51, TP = 356). ML: Machine learning; CV: cross-validation; ROC: receiver operating characteristic; AUC: area under the curve.

the presence of SMOKING_1 (value = 1, red points) consistently produces large positive SHAP values, strongly pushing the prediction toward the "Cancer" class (class 1). Conversely, the presence of SMOKING_0 (red points, which means the patient is a non-smoker) yields strongly negative SHAP values, driving the prediction towards the "No Cancer" class (class 0).

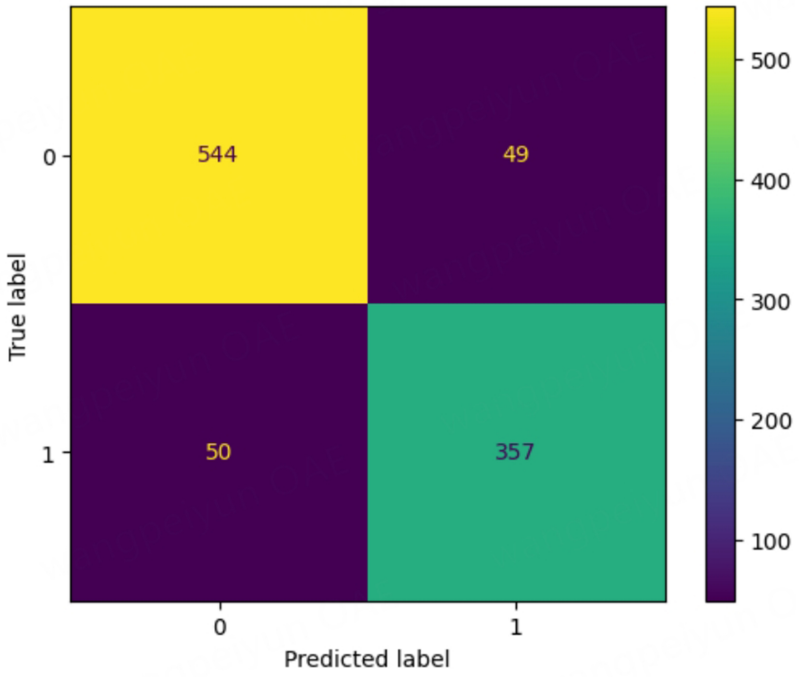


**Figure 5.** Confusion matrix of the classification model on lung cancer prediction dataset obtained by Ahmed[27].

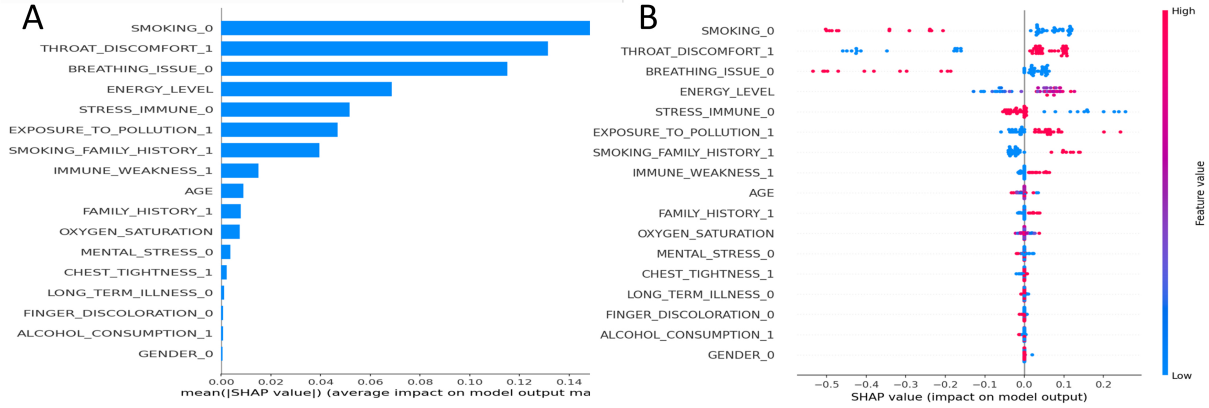


**Figure 6.** SHAP interpretability analysis for the final classification model. (A) Global feature importance ranked by mean absolute SHAP value (|SHAP value|), indicating the average impact magnitude on the model output; (B) SHAP summary plot (beeswarm) showing the distribution of SHAP values for each feature across individual predictions. Each point represents a single prediction; its position on the x-axis denotes the impact on the model output (positive values push toward class 1, negative toward class 0), and its color represents the original feature value (red = high/present, blue = low/absent). SHAP: SHapley Additive exPlanations.

This plot also uncovers more subtle patterns. For ENERGY_LEVEL, a clear inverse relationship is visible: high energy levels (red points) are associated with negative SHAP values (lower cancer risk), while low energy levels (blue points) contribute positively to the prediction (higher cancer risk). Similarly, the presence of THROAT_DISCOMFORT_1 and BREATHING_ISSUE_1 (not shown in the top features but visible in the beeswarm) consistently pushes predictions towards a positive diagnosis. The vertical spread of SHAP values for a given feature, such as EXPOSURE_TO_POLLUTION_1, indicates interaction effects, where the impact of pollution exposure may be amplified or mitigated by other factors. By automatically generating and presenting this dual-level interpretability, MatSci-ML Studio enables users to validate that

the model has learned medically intuitive and scientifically sound relationships, thereby fostering trust in its predictions.

### Synthesis and implications from case studies

The two case studies, while demonstrating the versatility of MatSci-ML Studio, originate from fundamentally different contexts, as astutely noted by our reviewers. Case 1 (Alloy UTS) exemplifies a typical materials science problem that involves what can be termed the "compression of a hierarchical structure", where low-level physical descriptors are engineered to predict a high-level macroscopic property. In contrast, Case 2 (Lung Cancer) represents a clinical problem where features exist on a "single hierarchical level" of empirical observation.

This distinction highlights a key strength and a clear limitation of our toolkit. The utility of MatSci-ML Studio lies in its domain-agnostic architecture: it provides a robust, automated workflow for any problem that can be formulated with structured, tabular data, regardless of feature origin. Its limitation, however, is that it operates on this final tabular representation and does not automate the complex, domain-specific process of feature engineering itself (e.g., deriving the physical descriptors in Case 1). The toolkit empowers researchers once these foundational features are established, accelerating the subsequent steps of model building, interpretation, and optimization.

### Advantages

It is crucial to first clarify that MatSci-ML Studio, as an induction-based toolkit, operates on correlations within data and does not inherently model underlying physical or biological causal mechanisms. Its primary role is to serve as a convenient and powerful platform for researchers to strategically execute data-driven discovery. Within this well-defined scope, MatSci-ML Studio represents a significant methodological advancement.

MatSci-ML Studio delivers significant methodological advantages by targeting three key challenges in computational materials science: accessibility, reproducibility, and analytical depth. First, it champions accessibility. Unlike traditional script-based frameworks, MatSci-ML Studio is built entirely around an intuitive GUI. This no-code design fundamentally democratizes advanced ML. It empowers domain experts to perform sophisticated analyses without the hurdle of extensive programming, thereby shifting their focus from technical implementation to scientific inquiry. Second, it enhances end-to-end reproducibility. We address the common challenge of fragmented workflows by encapsulating the entire research pipeline - from data ingestion to model interpretation - within a single, cohesive application. This seamless integration is powerfully enhanced by a built-in Project Management and Version Control system, a feature that facilitates traceable and reproducible research by allowing users to snapshot and restore entire experimental states.

Furthermore, the software is designed with embedded intelligence to guide the user. The Advanced Preprocessing module, for instance, does not merely provide a set of tools; it includes an EnhancedDataQualityAnalyzer that automatically assesses the dataset and offers actionable, prioritized recommendations. This semi-automated guidance, combined with an interactive interface featuring an undo/redo state manager, fosters an exploratory and risk-free environment for data preparation. This approach contrasts sharply with the "black-box" nature of fully automated scripts, placing the domain expert firmly in control while alleviating the technical burden.

Finally, the most significant strategic contribution of MatSci-ML Studio is its ability to logically integrate multiple tools into a seamless, end-to-end design loop. The platform transforms the ML model from a passive predictor into a proactive engine for discovery. This is achieved by integrating the forward modeling workflow with inverse design capabilities. Researchers can first build and validate a predictive model, then use the integrated SHAP analysis to understand its driving factors, and finally, employ this same model as a high-fidelity surrogate within the target optimization and multi-objective optimization modules. This seamless chaining of analysis, interpretation, and optimization is the core of the toolkit's strategic power, enabling researchers to efficiently navigate complex design spaces and accelerate the discovery of novel materials.

## CONCLUSIONS

In this work, we developed and presented MatSci-ML Studio, a novel interactive toolkit designed to bridge the gap between advanced ML methodologies and their practical application in materials science. The core contribution of MatSci-ML Studio is its fully integrated, GUI-driven workflow, which encapsulates the entire ML pipeline from project management and intelligent preprocessing to advanced model training and interpretation. By replacing the command-line with an intuitive interface, it substantially lowers the technical barrier for researchers, enabling them to execute complex analyses without extensive programming expertise. As demonstrated through both materials-centric regression and general classification case studies, MatSci-ML Studio proved to be a versatile and effective tool for generating high-fidelity, interpretable models. By automating laborious technical processes while maintaining user control, our platform empowers researchers to focus on scientific insight and innovation, thereby accelerating data-driven discovery in materials science.

## DECLARATIONS

### Authors' contributions

Conceptualization, methodology, software (lead), validation, writing - original draft: Wang, Y.
Conceptualization, resources, supervision, funding acquisition, writing - review and editing: Wang, F.
Software (testing), data curation, visualization: Yan, G.
Software (testing): Wang, J.
Validation, formal analysis, visualization: Niu, G.
Data curation, investigation: Feng, J.
Conceptualization, project administration, supervision, funding acquisition, writing - review and editing: Mao, J.
Validation, resources: Zhao, Y.

### Availability of data and materials

The source code for the MatSci-ML Studio software is openly available in a public GitHub repository at https://github.com/wy314159-lyq/MatSci-ML-Studio1.git. This release represents a stable version that has resulted from extensive development and testing.
However, as with any complex software project, we anticipate that there may still be undiscovered bugs or areas for improvement. We therefore warmly welcome and highly encourage user feedback through the GitHub repository. This community engagement will be instrumental in guiding the ongoing maintenance and enhancement of the toolkit in future versions.
The datasets used for the case studies in this work are publicly available from their original sources. The aluminum alloy dataset was adapted from the work of Jiang *et al.*[25], and the lung cancer prediction dataset is available from the Kaggle platform[27]. Further details on data access are provided in the main text.

### Financial support and sponsorship
This study was financially supported by the National Natural Science Foundation of China (No. 52402365), Yunnan Fundamental Research Projects (202401BE070001-010), and Sichuan Science and Technology Program (No. 24NSFSC3150).

### Conflicts of interest
All authors declared that there are no conflicts of interest.

### Ethical approval and consent to participate
Not applicable.

### Consent for publication
Not applicable.



## REFERENCES
1. El Naqa, I.; Murphy, M. J. What is machine learning? In: El Naqa I, Li R, Murphy MJ, editors. Machine learning in radiation oncology. Cham: Springer International Publishing; 2015. pp. 3-11. DOI
2. Jordan, M. I.; Mitchell, T. M. Machine learning: trends, perspectives, and prospects. *Science* **2015**, *349*, 255-60. DOI PubMed
3. Butler, K. T.; Davies, D. W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547-55. DOI PubMed
4. Durodola, J. Machine learning for design, phase transformation and mechanical properties of alloys. *Prog. Mater. Sci.* **2022**, *123*, 100797. DOI
5. Liu, X.; Zhang, J.; Pei, Z. Machine learning for high-entropy alloys: progress, challenges and opportunities. *Prog. Mater. Sci.* **2023**, *131*, 101018. DOI
6. Hao, C.; Sui, Y.; Yuan, Y.; Li, P.; Jin, H.; Jiang, A. Composition optimization design and high temperature mechanical properties of cast heat-resistant aluminum alloy via machine learning. *Mater. Design.* **2025**, *250*, 113587. DOI
7. Mo, W.; Xiao, Y.; Huang, Y.; et al. Active learning-based alloy design strategy for improving the strength-ductility balance of Al-Mg-Zn alloys. *Mater. Design.* **2025**, *252*, 113772. DOI
8. Liu, Y.; Zhang, Y.; Xiao, N.; Li, X.; Dai, F. Z.; Chen, M. Investigating interfacial segregation of Ω/Al in Al–Cu alloys: a comprehensive study using density functional theory and machine learning. *Acta Mater* 2024;279:120294. DOI
9. Dunn, A.; Wang, Q.; Ganose, A.; Dopp, D.; Jain, A. Benchmarking materials property prediction methods: the Matbench test set and Automatminer reference algorithm. *npj. Comput. Mater.* **2020**, *6*, 406. DOI
10. Xu, Z.; Jiang, F.; Niu, L.; et al. Magpie: alignment data synthesis from scratch by prompting aligned LLMs with nothing. *arXiv* **2024**, arXiv:2406.08464. https://doi.org/10.48550/arXiv.2406.08464. (accessed 31 Jul 2025)
11. Willman, J. Overview of PyQt5. In: Modern PyQt. Berkeley: Apress; 2021. pp. 1-42. DOI
12. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; et al. Scikit-learn: machine learning in Python. *J. Mach. Learn Res.* 2011;12:2825-30. https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?source=post_page. (accessed 31 Jul 2025)
13. Chen, T.; Guestrin, C. Xgboost: a scalable tree boosting system. *arXiv* **2016**, arXiv:1603.02754. https://doi.org/10.48550/arXiv.1603.02754. (accessed 31 Jul 2025)
14. Ke, G.; Meng, Q.; Finley, T.; et al. LightGBM: a highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*. 2017. https://proceedings.neurips.cc/paper_files/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html. (accessed 31 Jul 2025)
15. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A. V.; Gulin, A. CatBoost: unbiased boosting with categorical features. *arXiv* **2017**, arXiv:1706.09516. https://doi.org/10.48550/arXiv.1706.09516. (accessed 31 Jul 2025)
16. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: a next-generation hyperparameter optimization framework. *arXiv* **2019**, arXiv:1907.10902. https://doi.org/10.48550/arXiv.1907.10902. (accessed 31 Jul 2025)
17. Mkaouer, W.; Kessentini, M.; Shaout, A.; et al. Many-objective software remodularization using NSGA-III. *ACM. Trans. Softw. Eng. Methodol.* **2015**, *24*, 1-45. DOI
18. Ishibuchi, H.; Imada, R.; Setoguchi, Y.; Nojima, Y. Performance comparison of NSGA-II and NSGA-III on various many-objective test problems. In: *2016 IEEE Congress on Evolutionary Computation (CEC)*, Vancouver, Canada. Jul 24-29, 2016. IEEE; 2016. pp. 3045-52. DOI
19. Zitzler, E.; Laumanns, M.; Thiele, L. SPEA2: improving the strength pareto evolutionary algorithm. 2001. https://www.research-

collection.ethz.ch/handle/20.500.11850/145755. (accessed 31 Jul 2025)

20. Zhang, Q.; Li, H. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE. Trans. Evol. Computat.* **2007**, *11*, 712-31. DOI
21. Hansen, N.; Müller, S. D.; Koumoutsakos, P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol. Comput.* **2003**, *11*, 1-18. DOI PubMed
22. Das, S.; Suganthan, P. N. Differential evolution: a survey of the state-of-the-art. *IEEE. Trans. Evol. Computat.* **2011**, *15*, 4-31. DOI
23. Boyce, B.; Dingreville, R.; Desai, S.; et al. Machine learning for materials science: barriers to broader adoption. *Matter* **2023**, *6*, 1320-3. DOI
24. Zhang, H.; Fu, H.; He, X.; et al. Dramatically enhanced combination of ultimate tensile strength and electric conductivity of alloys via machine learning screening. *Acta. Mater.* **2020**, *200*, 803-10. DOI
25. Jiang, L.; Fu, H.; Zhang, Z.; et al. Synchronously enhancing the strength, toughness, and stress corrosion resistance of high-end aluminum alloys via interpretable machine learning. *Acta. Mater.* **2024**, *270*, 119873. DOI
26. Li, H.; Li, X.; Li, Y.; et al. Machine learning assisted design of aluminum-lithium alloy with high specific modulus and specific strength. *Mater. Design.* **2023**, *225*, 111483. DOI
27. Ahmed, I. Lung cancer prediction dataset. 2025. https://www.kaggle.com/datasets/shantanugarg274/lung-cancer-prediction-dataset. (accessed 31 Jul 2025)