EnginML: A Simplified Open-Source Framework for Teaching and Applying Machine Learning in Engineering

Amir Rafe
Research Assistant
Department of Civil and Environmental Engineering
Utah State University, Logan, UT, USA
Email: amir.rafe@usu.edu

Abstract—Despite the growing importance of machine learning (ML) in engineering applications, many engineers and students still face significant barriers to entry due to programming complexity. This paper presents EnginML, an open-source Python package designed to democratize access to machine learning for engineering students and professionals with minimal programming experience. EnginML provides a simplified interface for common machine learning workflows, enabling users to perform regression, classification, and clustering tasks with minimal code. The package features automatic data loading, model training, performance evaluation, and result visualization, along with model interpretability through SHAP (SHapley Additive exPlanations). We demonstrate the package's utility through practical engineering examples and discuss its educational benefits. EnginML aims to bridge the gap between traditional engineering education and modern data-driven methods, empowering engineers to incorporate ML techniques into their professional practice and research.

Index Terms—Machine Learning, Engineering Education, Python Package, Educational Software, SHAP, Model Interpretability

1 Introduction

The adoption of machine learning (ML) has accelerated across all engineering disciplines, from predicting material properties to optimizing structural designs and forecasting energy consumption [1]. Despite this growing importance, many engineers trained in traditional methodologies face significant barriers when attempting to incorporate these powerful techniques into their work [2].

While numerous ML libraries exist, most assume prior programming experience and statistical knowledge, presenting a steep learning curve for engineers focused on domain expertise rather than software development. This gap in accessibility limits the adoption of data-driven methods in engineering practice and education.

This paper introduces EnginML, an educational Python package designed specifically to address this gap by providing a streamlined interface for machine learning workflows commonly used in engineering applications. Unlike general-purpose ML libraries that prioritize flexibility and advanced features, EnginML emphasizes simplicity, transparency, and interpretability, qualities particularly valuable for engineering contexts where understanding model behavior is as important as performance.

The key contributions of this work include:

- A simplified Python API that reduces common ML workflows to single-function calls
- An intuitive command-line interface requiring no programming experience
- Automatic model evaluation and performance visualization
- Built-in model interpretability through SHAP
- Engineering-focused examples and documentation

EnginML is not intended to replace comprehensive libraries like scikit-learn [3] or TensorFlow [4], but rather to serve as an educational bridge that helps engineers begin applying ML techniques with minimal friction, eventually transitioning to these more powerful tools as their expertise grows.

2 Background

2.1 Machine Learning in Engineering

Machine learning has transformed numerous engineering fields by enabling data-driven insights and predictions. In civil engineering, ML models predict structural behavior and material properties [5]. Mechanical engineers use ML for predictive maintenance and design optimization [6]. Environmental engineers apply these techniques to model complex environmental systems and predict pollution patterns [7].

Despite these advances, the adoption of ML in engineering practice still lags behind research, partly due to the technical barriers faced by practitioners [8]. Traditional engineering education emphasizes domain-specific analytical methods rather than programming or statistical learning, creating a knowledge gap for many engineers.

2.2 Educational Software in Engineering

Educational software has a long history in engineering, from finite element analysis packages with educational licenses to specialized tools for teaching control systems or thermodynamics [9]. These tools typically prioritize conceptual understanding over industrial-scale capabilities, often featuring simplified interfaces and visual feedback.

In the ML domain, platforms like Orange [10] and Weka [11] provide graphical interfaces for machine learning, but they are not specifically tailored to engineering applications or workflows. There remains a need for educational tools that bridge the gap between engineering domain knowledge and machine learning capabilities.

3 EnginML Framework

3.1 Design Philosophy

EnginML was designed with the following principles:

- **Simplicity**: Reduce common ML workflows to minimal code
- **Transparency**: Make the underlying processes clear and explicit
- **Interpretability**: Prioritize understanding over black-box performance
- **Engineering relevance**: Focus on tasks and examples relevant to engineering applications

These principles informed both the API design and the selection of included algorithms and visualization methods.

3.2 Package Architecture

EnginML is built as a lightweight wrapper around established libraries, primarily scikit-learn [3]. The package structure separates core functionality (model fitting), explainability, reporting, and command-line interface components. This modular design maintains clarity while allowing for future extensions.

The mathematical foundation of the core functionality can be summarized as follows:

For regression tasks with input features $X \in \mathbb{R}^{n \times p}$ and target values $y \in \mathbb{R}^n$, EnginML trains a model f to minimize:

$$\mathcal{L}(f) = \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - y_i)^2 \tag{1}$$

For classification with categorical targets $y \in \{1, ..., C\}^n$, the package optimizes:

$$\mathcal{L}(f) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(f(X_i) \neq y_i) \tag{2}$$

For clustering tasks, EnginML implements centroid-based approaches that minimize:

$$\mathcal{L}(C_1, ..., C_k) = \sum_{j=1}^{k}\sum_{X_i \in C_j}\|X_i - \mu_j\|^2 \tag{3}$$

where μ_j represents the centroid of cluster C_j.

3.3 Key Components

EnginML consists of four main components:

1) **Core fitting functions**: Simplified interfaces for regression, classification, and clustering
2) **Explainability module**: Integration with SHAP for feature importance and model interpretation
3) **Reporting functions**: Automatic generation of visualizations and HTML reports
4) **Command-line interface**: No-code access to all core functionality

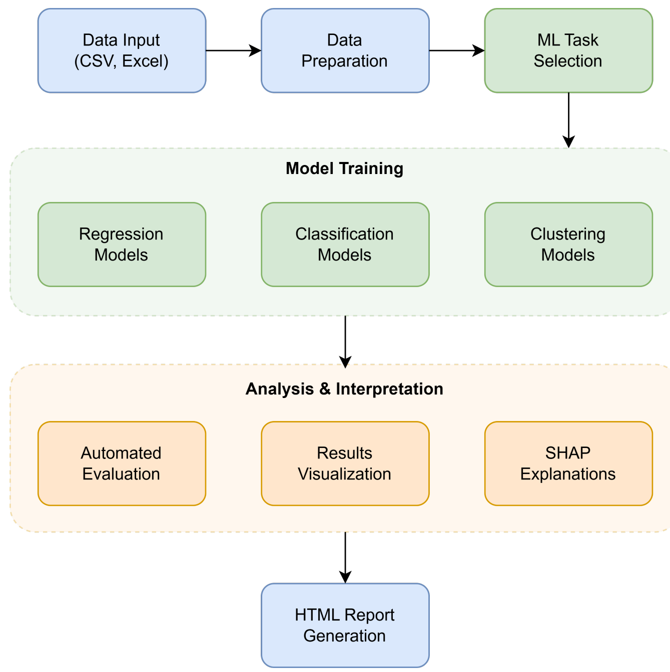

Fig. 1. EnginML workflow showing the progression from data input through model training to visualization and interpretation.

4 Implementation and Features

4.1 Simplified API

EnginML reduces common machine learning workflows to single function calls. For example, training a regression model requires just one line of code:

```
result = fit_regression(X, y, model='random_forest')
```

This approach abstracts away the underlying complexity while still providing access to key model parameters and outputs.

4.2 Model Selection

The package supports multiple models for each task type:

- **Regression**: Random Forest and K-Nearest Neighbors
- **Classification**: Random Forest and K-Nearest Neighbors
- **Clustering**: K-Means, BIRCH, and Gaussian Mixture Models

These algorithms were selected for their robustness, interpretability, and relevance to engineering applications. Future versions will incorporate additional algorithms based on user feedback.

4.3 Automated Evaluation

EnginML automatically performs:

- Train-test splitting
- Cross-validation
- Performance metric calculation

For regression tasks, the package reports R^2 and Mean Absolute Error (MAE). Classification models are evaluated using accuracy and F1 score. Clustering solutions are assessed with silhouette and Davies-Bouldin indices.

4.4 Model Interpretability

A core feature of EnginML is its integration with SHAP [12], which provides model-agnostic explanations of feature importance. This addresses a critical need in engineering applications, where understanding the reasoning behind predictions is often as important as the predictions themselves.

The SHAP values ϕ_i for each feature i are calculated as:

$$\phi_i(f,x) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!}[f_x(S \cup \{i\}) - f_x(S)] \tag{4}$$

where N is the set of all features, S represents subsets of features, and $f_x(S)$ is the expected model output when only features in set S are known.

4.5 Visualization and Reporting

EnginML generates task-appropriate visualizations automatically:

- **Regression**: Actual vs. predicted plots, residual analysis, and feature importance
- **Classification**: Confusion matrices, ROC curves, and SHAP summary plots
- **Clustering**: Cluster assignment visualizations and silhouette analysis

These visualizations are combined into comprehensive HTML reports that can be saved locally or shared.

4.6 Command-Line Interface

For users with no programming experience, EnginML provides a command-line interface that requires only a data file and basic configuration:

```
enginml data.csv --task regression --target y
```

This interface handles all steps from data loading to report generation, making ML accessible even to complete programming novices.

5 Educational Impact and Discussion

5.1 Bridging the Knowledge Gap

Initial feedback from engineering students and professionals suggests that EnginML effectively reduces the entry barrier to machine learning. By abstracting implementation details while maintaining transparency about the underlying processes, the package allows users to focus on problem formulation and result interpretation, skills more aligned with traditional engineering thinking.

5.2 Pedagogical Applications

EnginML has been integrated into several engineering courses, where it serves as:

- A demonstration tool for introducing ML concepts
- A platform for hands-on exercises without programming prerequisites
- A starting point for more advanced ML explorations

The package's design encourages a gradual transition from the simplified interface to direct use of underlying libraries, supporting a scaffolded learning approach.

5.3 Limitations and Future Work

While EnginML succeeds in its educational mission, it has several limitations:

- Limited model selection compared to comprehensive libraries
- Basic hyperparameter settings without tuning capabilities
- Simplified data preprocessing without advanced feature engineering

Future development will address these limitations while maintaining the core philosophy of accessibility. Planned enhancements include:

- Additional models (e.g., neural networks with simplified configurations)
- Basic hyperparameter tuning with educational visualizations
- Domain-specific examples for different engineering fields
- Interactive tutorials integrated with the package

6 Conclusion

EnginML demonstrates that machine learning can be made accessible to engineers and students without compromising educational value or practical utility. By reducing common workflows to minimal code while maintaining transparency and interpretability, the package bridges the gap between traditional engineering education and modern data-driven methods.

The adoption of tools like EnginML in engineering curricula and professional practice has the potential to accelerate the integration of machine learning into engineering workflows, ultimately leading to more informed design decisions and innovative solutions to complex problems.

EnginML is available as an open-source package with comprehensive documentation at https://github.com/pozapas/EnginML, with ongoing development guided by user feedback from the engineering education and professional communities.

References

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science, vol. 349, no. 6245, pp. 255-260, 2015.

[2] R. E. Baker et al., "Mechanistic models versus machine learning: A fight worth fighting for the biological community?," Biology Letters, vol. 14, no. 5, 2018.

[3] F. Pedregosa et al., ”Scikit-learn: Machine learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
[4] M. Abadi et al., ”TensorFlow: A system for large-scale machine learning,” in 12th USENIX Symposium on Operating Systems Design and Implementation, 2016, pp. 265-283.
[5] M. Z. Naser, ”Mechanistically informed machine learning and artificial intelligence in fire engineering and sciences,” Fire Technology, vol. 57, pp. 1-42, 2021.
[6] C. Zhang et al., ”Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms,” Automation in Construction, vol. 107, p. 102930, 2019.
[7] W. Wu et al., ”Application of machine learning methods to identify topographic and climatic factors associated with landslide occurrence,” in Engineering Geology for Society and Territory, vol. 1, Springer, 2014, pp. 367-371.
[8] Y. Reich, ”Machine learning in design: Opportunities and challenges,” Computer-Aided Design, vol. 115, pp. 1-16, 2019.
[9] R. M. Felder, D. R. Woods, J. E. Stice, and A. Rugarcia, ”The future of engineering education II. Teaching methods that work,” Chemical Engineering Education, vol. 34, no. 1, pp. 26-39, 2000.
[10] J. Demšar et al., ”Orange: Data mining toolbox in Python,” Journal of Machine Learning Research, vol. 14, pp. 2349-2353, 2013.
[11] M. Hall et al., ”The WEKA data mining software: An update,” ACM SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10-18, 2009.
[12] S. M. Lundberg and S. I. Lee, ”A unified approach to interpreting model predictions,” in Advances in Neural Information Processing Systems, 2017, pp. 4765-4774.