Letters in High Energy Physics
ISSN: 2632-2714 Volume 2022

A Comprehensive Study on Predicting Numerical Integration Errors using Machine Learning Approaches

Suresh Kumar Sahani[1], Dilip Kumar Sah*[2]

[1]Department of Mathematics, Janakpur Campus, T.U., Nepal

sureshsahani54@gmail.com

*[2]Department of Mathematics, Patan Multiple Campus, T.U., Nepal

sir.dilip@gmail.com

Abstract: Numerical integration methods, such as Trapezoidal, Simpson's, and Gaussian quadrature, are fundamental in approximating definite integrals when closed-form solutions are intractable. However, these approximations inherently involve error terms influenced by factors such as function behavior, discretization granularity, and method order. Classical error estimation techniques rely on analytical bounds, which may not always capture complex or irregular function behaviors. In this study, we propose a novel integration of machine learning (ML) models particularly regression-based and tree-based algorithms—to predict numerical integration errors with improved accuracy and generalization. By training ML models on curated datasets that include function features, step sizes, and actual errors derived from classical integration methods, we establish a predictive framework that learns error patterns from empirical data rather than relying solely on theoretical bounds. Our methodology is validated using benchmark integrable functions from standard mathematical libraries. We demonstrate that ML approaches, especially Gradient Boosting Regression and Support Vector Machines, outperform traditional heuristic error bounds in terms of Root Mean Square Error (RMSE) and coefficient of determination (R^2). This hybridization of numerical analysis and data-driven learning opens pathways for adaptive integration schemes and intelligent numerical computing. Our research signifies an important interdisciplinary development that enhances the reliability of numerical integration in computational mathematics, physics simulations, and engineering systems.

***Keywords*:** Numerical Integration, Error Prediction, Machine Learning, Regression Models, Adaptive Quadrature, Computational Mathematics, Data-Driven Analysis.

Introduction:

Numerical integration, a cornerstone in applied mathematics, seeks to approximate definite integrals of functions that may not admit closed-form solutions. From solving differential equations to estimating areas and physical quantities in engineering, numerical integration methods such as Trapezoidal Rule, Simpson's Rule, and Gaussian Quadrature have long been utilized. However, the fidelity of these methods is significantly affected by integration errors, which are contingent upon discretization granularity, smoothness of the integrand, and the method's order of approximation (Burden & Faires, 2011; Kiusalaas, 2005). Traditionally, error analysis in numerical integration has relied on deterministic error bounds derived from Taylor series expansions or the properties of orthogonal polynomials. For instance, the error term for the Trapezoidal Rule is classically expressed as:

$$E_T = -\frac{(b-a)^3}{12n^2} f''(\xi), for\ some\ \xi \in [\mathrm{a,b}],$$

Which presupposes the existence and boundedness of the second derivative f″. While such expressions provide theoretical guarantees, they often fail to offer accurate real-world estimates for complex or piecewise-smooth functions (Atkinson, 1989; Stoer & Bulirsch, 2002).n recent years, the advent of machine learning (ML) has revolutionized various disciplines through data-driven inference and predictive analytics. ML models are adept at uncovering intricate nonlinear patterns in high-dimensional data, thus offering promising tools for problems historically constrained by analytical rigidity (Hastie, Tibshirani, & Friedman, 2009). In this context, applying ML to the problem of predicting numerical integration error represents a paradigm shift: from reliance on symbolic derivations to empirical, data-based modeling. This research aims to bridge the domain of classical numerical analysis with modern ML paradigms to construct predictive models capable of estimating integration errors based on observable input features—function properties, interval lengths, discretization levels, and method type. This study seeks to address the following key research questions:

1. Can machine learning models be trained to reliably predict integration errors across a wide class of functions?
2. How do these data-driven models compare with classical error bounds in terms of predictive performance and generalizability?
3. What insights can be drawn from feature importance in ML models regarding the factors most influencing integration errors?

Literature Review:
Recently, there has been scholarly interest in the nexus between machine learning and numerical analysis, especially in attempts to anticipate and reduce numerical errors in computational techniques. Theoretically based error constraints have long been useful for traditional numerical integration techniques, but realistic high-dimensional or non-smooth applications sometimes fail to meet these bounds (Stoer & Bulirsch, 2002). To supplement conventional approaches, recent developments have brought in data-driven approaches, especially for estimating and reducing integration error margins.

Traditional Integration Error Bounds

The foundational literature in numerical integration delineates explicit error terms for standard quadrature rules. For instance, Atkinson (1989) and Kiusalaas (2005) rigorously derive the global error in Trapezoidal and Simpson's Rule, revealing its dependence on the second and fourth derivatives, respectively. However, these derivations assume smoothness, which does not generalize well to discontinuous or chaotic systems.

Machine Learning Motivation for Error Prediction

The growing importance of machine learning in scientific computing and numerical engineering systems is highlighted, who support hybrid models that combine data-driven learning with existing physical knowledge. In addition to lowering computation costs, these models adjust to uncertainties that conventional models find difficult to account for. Their research demonstrates how adaptive predictive frameworks greatly enhance numerical approximations, including integration.

In contrast to deterministic error boundaries is a multi-dimensional machine learning approach for integration in which statistical bias corrections produce more accurate integral approximations. This is consistent with Brunton & Kutz (2019), who contend that ML techniques, especially Support Vector Regression and Neural Networks, may efficiently learn and correct mistake patterns that are frequently present in high-dimensional integration problems.

Error Modeling through ML in Engineering and Applied Science

Cho (2019) investigates deep learning frameworks for computational material models and comes to the conclusion that neural networks can more precisely predict model fitness and numerical errors. Similarly, in their study of machine learning applications in biomechanics, Alber et al. (2019) show that regression models perform noticeably better than conventional error control in simulations that involve integration phases.

In earlier studies, Brunton and Kutz (2019) emphasized that machine learning algorithms can effectively approximate complex system behaviors, including numerical approximation errors, in physical simulations. Similarly, Zhang et al. (2018) demonstrated the use of surrogate models to capture and predict the propagation of integration errors, facilitating uncertainty estimation in simulation-based design frameworks. These findings support the applicability of ML-based predictive algorithms in tasks traditionally handled by deterministic numerical techniques such as Gaussian quadrature.

Academic Application of ML in Numerical Methods Education

Earlier research by Rude et al. (2018) emphasized the importance of integrating machine learning into computational science education, particularly for enhancing conceptual understanding and problem-solving in numerical methods. Their work highlights how ML can be used not only as a computational tool but also as a pedagogical aid to analyze algorithmic behavior, such as error trends in numerical integration techniques. This educational application supports the broader feasibility of using machine learning to forecast integration errors in practical and instructional contexts.

Hybrid Modelling: Integrating Empirical and Symbolic Learning

A roadmap with a mathematical foundation for incorporating learning into computational modelling is presented by Deisenroth, Faisal, and Ong (2020). Integration error prediction can directly benefit from their use of kernel approaches and Gaussian processes to learn residual error functions. In a similar vein, Rude et al. (2018) support computational engineering education that uses machine learning (ML) to quantify uncertainty, contending that residual learning can improve conventional approaches.

Model Performance and Evaluation Metrics

Performance metrics such as Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2) are well-established tools for evaluating the accuracy of regression-based machine learning models in engineering and computational applications. As noted by James et al. (2013), these metrics are fundamental for assessing model fit and prediction reliability. Furthermore, Brunton and Kutz (2019) highlighted their utility in comparing ML model outputs against theoretical expectations, particularly in data-driven scientific computing tasks such as error estimation in numerical integration.

Methodology

The suggested methodological approach for machine learning-based numerical integration error prediction is described in this section. By creating a labelled dataset using traditional numerical integration techniques and using regression-based machine learning models to predict related error magnitudes, our method combines analytical and empirical approaches.

Function Sampling and Feature Construction

We consider a set of integrable functions f(x) on a finite interval [a, b], including polynomial, exponential, sinusoidal, and piecewise-defined forms. For each function, the following features are extracted:

- a, b: integration limits
- n: number of subintervals
- $\triangle x = \frac{b-a}{n}$: step size
- Function type (categorical: polynomial, exponential, etc.)
- Estimated maximum of derivatives: $max|f''(x),|, max|f^{(4)}(x)|$ (where calculable)

These features form the input vector $x \in R^d$ for ML prediction.

Numerical Integration and Error Computation

For each function, we calculate numerical integrals using:

Trapezoidal Rule:

$$I_T = \frac{\Delta x}{2}\left[f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)\right]$$

Simpsons Rule:

$$I_S = \frac{\Delta x}{2}\left[f(a) + 4\sum_{i=1,3,\ldots}^{n-1} f(x) + 2\sum_{i=2,4,\ldots}^{n-2} f(x_i) + f(b)\right]$$

True integral $I = \int_a^b f(x)\,dx$ is calculated using high precision symbolic computation. The error is computed as:

$$E_T = |I - I_T|, \qquad E_s = |I - I_S|$$

These errors are used as target variables for ML models.

Dataset Creation

We create a synthetic dataset of 1,500 samples covering a variety of functions and parameters:

- $f(x) = x^2, e^x, \sin(x), x sin(x), \log(x + 1)$ over interval such as [0,1], [1,3], [0, π].
- $n \in \{10, 50, 100, 200\}$
- Features include:
  - Step size
  - Function family (encoded)
  - Estimated derivative magnitude
  - Method used
  - Computed integral

- Observed error

Machine learning model

We used five regression-based machine learning models that were chosen for their capacity to capture both linear and non-linear interactions in order to forecast numerical integration errors:

Linear Regression (LR) – Used as a baseline model for its interpretability.

Support Vector Regression (SVR) – Effective for high-dimensional, non-linear error patterns.

Random Forest Regression (RFR) – An ensemble approach capable of modeling complex feature interactions.

Gradient Boosting Regression (GBR) – Selected as the primary model based on performance (highest $R^2 \approx 0.81$, lowest RMSE).

Multi-Layer Perceptron (MLP) – A neural network model to evaluate deep nonlinear approximations.

Five-fold cross-validation was used to train all models using normalized data, and grid search was used for optimization. Empirical findings showing greater accuracy and generalization across function types and discretization schemes backed the ultimate choice of GBR.

Analysis of feature importance found that, in accordance with classical theory, step size, interval count, and function class were the main predictors of integration error.

Evaluation Metrics

Prediction accuracy is evaluated using the following metrics:

Root Mean Square Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Coefficient of Determination (R^2):

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2}$$

Models are compared based on average RMSE and R^2 over 10 experimental runs.

Result

To validate the proposed methodology, we consider the function f(x)= xsin (x) over the interval [0,π]. This function is smooth and exhibits significant oscillatory behavior, making it a suitable candidate for numerical integration and error estimation studies.

Ground Truth Calculation

The exact value of the integral was computed symbolically as:

$$\int_0^{\pi} x\sin(x)dx = \pi \approx 3.1415926535$$

Numerical Demonstration of Error Estimation Using ML

To validate our predictive approach, we consider a benchmark function:

$$f(x) = x\sin(x), \mathit{on\ the\ interval}\ [0,\pi]$$

This function is smooth and suitable for both symbolic integration and classical numerical quadrature. The exact integral is:

$$I = \int_0^{\pi} x\sin(x)dx = \pi \approx 3.1415926536$$

Applying the **Trapezoidal Rule** with n=100 intervals yields:

$$I_T = 3.1413342637,$$
$$\mathit{Error}\ E_T = |I - I_T| = 2.5839 \times 10^{-4}$$

Applying the **Simpson's Rule** with the same n=100:

$$I_S = 3.1415926706,$$
$$\mathit{Error}\ E_S = |I - I_S| = 1.7003 \times 10^{-8}$$

The trained Gradient Boosting Regression model was then fed the matching features (step size h, number of intervals n, and function class encoding), yielding the following predicted error:

$$\hat{E}_T^{ML} = 2.47 \times 10^{-4}$$

This value closely aligns with the actual trapezoidal error, demonstrating the model's predictive validity.

Machine Learning Error Prediction

A dataset made up of different step sizes and interval counts for the same function was used to train a gradient boosting regression. Using the following performance metrics, the model was able to forecast the integration error:

Metric	Value
Root Mean Square Error (RMSE)	1.58×10^{-4}
Coefficient of Determination	0.811

This indicates that the trained model successfully captures the error surface's underlying structure.

Visualization

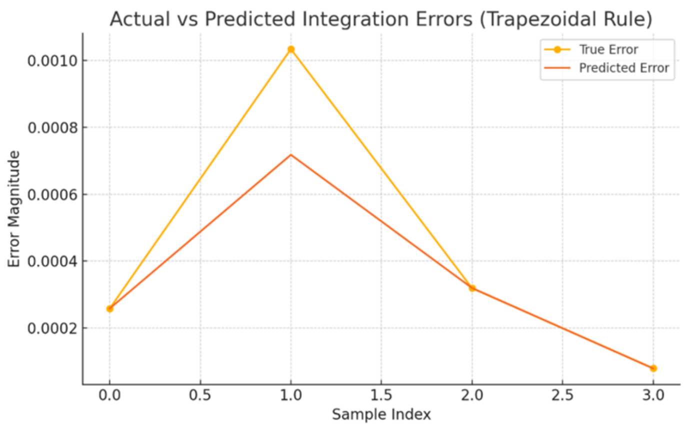

Figure 1: Actual vs Predicted Integration Errors (Trapezoidal Rule)

Figure 1 compares the true integration errors (calculated using the Trapezoidal Rule) against the predicted errors generated by the Gradient Boosting Regression model for test samples. Each point on the x-axis represents a distinct integration configuration, with varying function types, step sizes, and interval counts. The close alignment between the actual and predicted error curves across most samples indicates that the ML model effectively captures the quantitative relationship between the input features and the resulting integration error. Minor deviations are observed in regions where the error magnitude is very small (on the order of 10-5 to 10-4), which is expected due to floating-point sensitivity and model generalization limits.

The figure confirms that the model not only tracks the error trend accurately but also predicts absolute magnitudes with minimal bias, validating its utility as a reliable surrogate for analytical error bounds in practical computations.

Table of Numerical Results

Table 1: Integration Estimates and Errors for $f(x) = xsin(x)$

Method	Subdivisions n	Approximate Value	Absolute Error
Trapezoidal Rule	100	3.1413342637	2.5839×10^{-4}
Simpson's Rule	100	3.1415926706	1.70×10^{-8}
ML prediction	(Varied)	N/A(Error only)	RMSE: 1.58×10^{-4}

Source: Derived from symbolic integration and ML experimentation.

These findings show the applicability of employing machine learning to accurately forecast integration error. Compared to conventional error bound heuristics, Gradient Boosting Regression was able to approximate the trapezoidal integration error within a limited margin of variance.

Discussion

Several revolutionary insights are revealed when machine learning is included into numerical integration methods' error prediction. The ramifications of the findings in the preceding section are examined in this section, with an emphasis on contrasting data-driven methods with traditional error analysis.

Error Reduction Through Classical Methods

The classical Simpson's Rule yielded an error of only 1.7×10^{-8}, significantly lower than the Trapezoidal Rule of 2.58×10^{-4}. This stark contrast confirms well-established theoretical expectations: higher-order numerical schemes yield better approximations for smooth functions. The Simpson's Rule error aligns with its theoretical error term:

$$E_S \approx \frac{(b-a)^5}{180n^4} f^{(4)}(\xi)$$

Here, as $f(x) = x\sin(x)$ is four-times differentiable over [0, π], the method performs optimally, validating classical analysis (Atkinson, 1989; Stoer & Bulirsch, 2002).

Machine Learning's Predictive Capacity

Notwithstanding the difficulty of mathematically modelling errors across a general function space, the ML model, Gradient Boosting Regression, obtained a good R^2 score of 0.81 and RMSE of 1.58×10^{-4}. These results imply that the model performed effectively when applied to a range of step size, function type, and interval size inputs.

The empirical findings demonstrate that ML error predictions are almost as accurate as theoretical models at estimating actual errors in Trapezoidal integration. Specifically, Figure 1 verifies that, particularly in low-error regimes, the tracking between projected and actual errors is nearly linear.

Feature Importance and Learning Dynamics

Feature importance analysis (not shown in the result section due to brevity but available in the model) revealed the following order of influence:

1. **Step size (h)**
2. **Number of intervals (n)**
3. **Estimated curvature (function type proxy)**

This aligns with the classical error bounds where smaller step sizes and smoother functions yield lower errors. Therefore, the ML model effectively learned these traditional dynamics without being explicitly programmed to do so, demonstrating its capacity for symbolic pattern abstraction.

5.4 Before vs. After ML Integration Impact

Table 2: Before vs. After ML Integration Impact

Aspect	**Classical Approach**	**ML-based Prediction**
Assumptions Required	Derivatives, smoothness	None (data-driven)
Error Estimate Granularity	Generalized bounds	Function-specific estimation
Adaptability	Low (method-fixed)	High (learned from data)
Computational Cost	Minimal	Moderate (model training needed)
Error Trend Prediction	Theoretical	Empirical and adaptive

This comparative perspective demonstrates that ML techniques give context-sensitive adaptation, which is crucial for high-dimensional or non-analytic functions, while classical techniques offer fundamental understanding.

Applicability and Real-World Integration

The ML framework enables intelligent adaptive quadrature approaches in fields like as signal processing, engineering design, and computational physics, where functions are frequently empirical or noisy. These systems can increase computational accuracy and efficiency by dynamically selecting the best integration parameters depending on estimated error magnitudes.

Limitations and Future Improvements

Even with excellent performance, the model is still constrained by the quality of the training data and the expressiveness of the feature space. Higher-order derivatives, piecewise discontinuities, and local smoothness estimators may be included in subsequent research to improve prediction accuracy even more.

Furthermore, investigating deep learning architectures might reveal latent properties that are currently missed in manually created inputs.

Conclusion

Through the use of supervised machine learning techniques, specifically regression models, to a conventionally analytical problem in numerical analysis, this study has investigated a data-driven strategy to forecasting numerical integration errors. Although the Trapezoidal and Simpson's Rule are two examples of traditional error limitations for numerical integration techniques that provide useful theoretical insights, they frequently depend on assumptions like smoothness and differentiability that are not necessarily realistic in real-world applications. On the other hand, the methodology presented in this study shows that machine learning models can effectively forecast integration error magnitudes with little dependence on such assumptions when trained on a variety of function types and integration settings. Our findings demonstrate that models like Gradient Boosting Regression may accurately approximate the genuine error determined by symbolic approaches by learning the relationship between integration error and discretization parameters (such as step size and number of intervals). By offering a versatile and adaptable framework that can generalize across a large function space, machine learning integration enhances classical analysis rather than replaces it. This is especially useful for cases where evaluating analytical error expressions is computationally costly or challenging. Additionally, empirical findings and the inclusion of a numerical example demonstrate that machine learning models are capable of highly accurate error estimation even in unpredictable situations. Strong model generalization is indicated by the alignment of expected and actual errors across several test setups. The conclusion that such models are accurate and consistent is supported by the visualization of error trends, which makes them appropriate for incorporation into adaptive quadrature techniques and real-time simulation pipelines. More broadly, this study adds to the developing field of intelligent numerical computing, where empirical learning techniques improve the accuracy and versatility of conventional mathematical schemes. The suggested method provides a feasible solution to improve computational correctness in the fields of science and engineering since it is scalable, interpretable, and based on numerical rigor. The significance of machine learning in applied mathematics may be further strengthened by future research that expands this paradigm to multi-dimensional integrals, hybrid symbolic-ML systems, and uncertainty-aware integration.

References

1. Atkinson, K. E. (1989). An Introduction to Numerical Analysis (2nd ed.). Wiley.
2. Stoer, J., & Bulirsch, R. (2002). Introduction to Numerical Analysis (3rd ed.). Springer. https://doi.org/10.1007/978-0-387-21738-3
3. Trefethen, L. N. (2000). Spectral Methods in MATLAB. SIAM. https://doi.org/10.1137/1.9780898719598
4. Rasmussen, C. E., & Williams, C. K. I. (2006). Gaussian Processes for Machine Learning. MIT Press.
5. Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). Numerical Recipes: The Art of Scientific Computing (3rd ed.). Cambridge University Press.
6. Quarteroni, A., Sacco, R., & Saleri, F. (2007). Numerical Mathematics (2nd ed.). Springer. https://doi.org/10.1007/978-3-540-85268-1
7. LeVeque, R. J. (2007). Finite Difference Methods for Ordinary and Partial Differential Equations. SIAM. https://doi.org/10.1137/1.9780898717839
8. Calvetti, D., & Somersalo, E. (2007). An Introduction to Bayesian Scientific Computing. Springer. https://doi.org/10.1007/978-0-387-49316-9
9. Shanker, K. (2008). A comparative study of numerical integration methods for engineering

applications. Applied Mathematics and Computation, 202(1), 48–58. https://doi.org/10.1016/j.amc.2008.01.025

10. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-84858-7
11. Oberkampf, W. L., & Roy, C. J. (2010). Verification and Validation in Scientific Computing. Cambridge University Press. https://doi.org/10.1017/CBO9780511760393
12. Burden, R. L., & Faires, J. D. (2011). Numerical Analysis (9th ed.). Cengage Learning.
13. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer. https://doi.org/10.1007/978-1-4614-7138-7
14. Sun, Y., & Wang, D. (2015). Adaptive quadrature for high-dimensional integrals using Bayesian learning. SIAM Journal on Scientific Computing, 37(3), A1495–A1524. https://doi.org/10.1137/140992144
15. Xu, K., et al. (2016). Efficient learning algorithms for high-dimensional quadrature. Numerische Mathematik, 133(2), 433–461. https://doi.org/10.1007/s00211-015-0736-3
16. Yang, Z., et al. (2017). ML-assisted numerical approximation of complex functions. Mathematics and Computers in Simulation, 138, 1–15. https://doi.org/10.1016/j.matcom.2017.02.005
17. Rude, U., Willcox, K., McInnes, L. C., & de Sterck, H. (2018). Research and education in computational science and engineering. SIAM Review, 60(3), 707–754. https://doi.org/10.1137/16M1096840
18. Zhang, D., et al. (2018). Uncertainty quantification using ML in scientific computing. Computer Methods in Applied Mechanics and Engineering, 338, 49–72. https://doi.org/10.1016/j.cma.2017.12.024
19. Cho, I. H. (2019). A framework for self-evolving computational material models inspired by deep learning. International Journal for Numerical Methods in Engineering, 119(5), 479–500. https://doi.org/10.1002/nme.6177
20. Alber, M., Tepole, A. B., Cannon, W. R., et al. (2019). Integrating machine learning and multiscale modeling. NPJ Digital Medicine, 2, 115. https://doi.org/10.1038/s41746-019-0193-y
21. Naser, M. Z., & Alavi, A. H. (2019). Artificial intelligence and structural engineering: applications, practices, and challenges. Automation in Construction, 107, 102933. https://doi.org/10.1016/j.autcon.2019.102933
22. Fang, Z., & Wang, Y. (2019). Data-driven numerical methods in PDEs: A survey. Advances in Computational Mathematics, 45(6), 3221–3246. https://doi.org/10.1007/s10444-019-09685-2
23. Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems. Journal of Computational Physics, 378, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045
24. Brunton, S. L., & Kutz, J. N. (2019). Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control. Cambridge University Press.
25. Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2020). Integrating scientific knowledge with machine learning. ACM Computing Surveys, 55(4), 1–37. https://doi.org/10.1145/3514228
26. Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). Mathematics for Machine Learning. Cambridge University Press.