Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue4 , 2026

A Comparative Assessment of Generative, Symbolic, and Machine Learning Tools in STEM Teaching Pedagogy

[*1]Hira Ashraf Baig, [2]Muhammad Arif Hussain, [3]Muhammad Rahim, [1]Nazia Majeed, [4]Muhammad Atif Idrees

[1]Department of Computer Science, Iqra University Karachi, Sindh, Pakistan.
[2]Department of. I. T and computer science, Karachi Institute of Economics and Technology, Sindh, Pakistan.
[3]Department of Science and Humanities, National University of Computer and Emerging Sciences, Karachi 75030, Pakistan
[4]Department of Basic Sciences, DHA SUFFA University, Karachi. Pakistan
[*1]hira.atif@iqra.edu.pk [2]arif.hussain@kiet.edu.pk, [3]muhammad.rahim@nu.du.pk,
[1]nazia.majeed@iqra.edu.pk, atif.idrees@dsu.edu.com

Keywords
Artificial Intelligence, MATLAB, Python, Symbolic Computation, Machine Learning, Polynomial Regression.

Article History
Received on 4 March, 2026
Accepted on 10 April, 2026
Published on 22 April, 2026

Corresponding Author:
Hira Ashraf Baig

Abstract

The rapid integration of Artificial Intelligence (AI) in engineering science requires a critical evaluation of tool reliability and algorithmic limits. Grounded in contemporary frameworks of AI literacy and the "plausibility trap" of using probabilistic engines for deterministic tasks, this paper investigates the performance profiles of generative and symbolic platforms. We evaluate ChatGPT and Wolfram Alpha across localized statistical and algebraic computations foundational to engineering curricula. The empirical results demonstrate that the symbolic computation engine significantly outperforms the probabilistic language model in arithmetic accuracy and graphical replication, underscoring the necessity of algorithmic verification frameworks. Additionally, this study benchmarks engineering software environments by developing ordinary least squares and polynomial regression models within MATLAB and Python. While linear models from both applications yielded comparable results, the nonlinear models differed significantly, with MATLAB's model demonstrating a higher R^2 value compared to Python's, with MATLAB's structural matrix capturing a higher coefficient of determination $(R^2 = 0.928)$ than the default Python setup $(R^2 = 0.883)$. The paper emphasizes the necessity for educators to understand and address the challenges associated with AI adoption, particularly in enhancing mathematical foundations and problem-solving skills.

1. Introduction

The rapid integration of artificial intelligence (AI) and machine learning (ML) environments has emerged as a critical catalyst for structural transformation across modern industrial workflows and scientific pedagogy [5]. Within contemporary engineering disciplines, technological implementations have rapidly transitioned from standalone computational delivery platforms to integrated, cross-phase systems that influence curriculum design, engineering analytics, and automated optimization workflows [4] In developing regions such as Pakistan, higher education institutions are increasingly exploring these digital configurations to optimize learning pathways, bridge domain knowledge gaps, and scale computational training frameworks. However, empirical observations indicate that the unchecked adoption of these tools operates as a double-edged sword; while accelerating data-processing workflows, it introduces significant concerns regarding data protection, cognitive disengagement, and academic integrity when students circumvent the underlying mathematical mechanics [3]. A primary technical and instructional challenge stems from what contemporary researchers define as the "Plausibility Trap"—a behavioral phenomenon where users reflexively deploy expensive, multi-billion-parameter probabilistic engines to solve deterministic micro-tasks that require absolute structural exactness [7]. Large language models (LLMs) such as the baseline variations of ChatGPT are trained as generative sequence-prediction networks designed to output plausible textual responses based on token distributions. Consequently, their lack of a deterministic symbolic processing layer frequently introduces algorithmic hallucinations and mathematical errors during rigorous scientific computation (1; 6). In direct contrast, symbolic computation utilities such as Wolfram Alpha interpret queries using non-probabilistic abstract syntax trees, ensuring mathematical reliability and exact graphical replication. This operational divergence underscores the critical need for "algorithmic verification literacy" within the engineering science curriculum, preparing scholars to audit generative outputs through deterministic tools [1]. Beyond automated conversational interfaces, the core technical competencies of engineering practitioners rely fundamentally on executing multivariable statistical modeling and regression within advanced numerical environments. While ordinary least squares (OLS) linear regression provides a transparent baseline for modeling linear trends, complex real-world systems often exhibit non-linear interactions that demand polynomial or non-linear formulations. Evaluating these non-linear behaviors routinely exposes hidden design variations across popular development environments. For instance, executing regression scripts within proprietary platforms like MATLAB versus open-source ecosystems like Python can result in disparate parameter outputs due to differences in baseline model specifications, design matrix configurations, and internal library defaults [2]. To demonstrate these software discrepancies within an authentic engineering framework, this study leverages the comprehensive Yacht Hydrodynamics dataset originally compiled at the Delft Ship Hydromechanics Laboratory. Predicting the residuary resistance per unit weight of displacement from geometric coefficients and velocity is a foundational requirement during the initial hull design phase to minimize required propulsive power. Because residuary hull resistance scales exponentially at higher velocity bounds, it presents an ideal non-linear microbenchmark to evaluate the mathematical convergence of competing development platforms. Discrepancies in fitting high-order parameters can drastically impact downstream workflows like hull shape optimization, emphasizing why software configuration limits must be rigorously documented alongside regression errors like the coefficient of determination (R^2). To systematically evaluate these overlapping technical boundaries, this paper presents a multi-layered empirical analysis. Section 2 investigates the accuracy and user-interface limitations of ChatGPT and Wolfram Alpha when solving foundational descriptive statistics problems and quadratic algebraic equations. Section 3 delineates the machine learning implementation details for linear and non-linear polynomial regression on the Delft Yacht Hydrodynamics dataset, directly contrasting the outputs produced by MATLAB and Python. Section 4 provides a critical discussion of the results, framing the findings around tool selection engineering, algorithmic transparency, and regression reporting guidelines. Finally, Section 5 concludes the paper by providing actionable recommendations for data science deployment in engineering science settings.

2. Problems and their Artificial Intelligence Solutions

To evaluate the practical capabilities, structural constraints, and analytical accuracy of contemporary computational assistants within scientific workflows, this section establishes a localized comparative benchmark. In this framework, ChatGPT and Wolfram Alpha, AI-driven tools are used here to assess their effectiveness in solving two simple problems [16-18]. By subjecting a probabilistic language model and a deterministic symbolic processing engine to identical mathematical tasks, this evaluation uncovers how their underlying algorithmic architectures influence calculation exactness and data interpretation.

Problem 1:

We input **summarize {1,2,5,7,8,9,4,5,7,8,9}** in the command area of ChatGPT. The input query did not specify any particular parameters or instructions to output specific metrics; however, the model autonomously selected a limited subset of basic descriptive values and returned only the mean, median, mode, and range.

Output from ChatGPT:

Here is the summary for the provided dataset:

1. **Mean**: The average value of the data:
   = 5.727
2. **Median**: The middle value when the data is sorted:
   = 7.00
3. **Mode**: The most frequently occurring value:
   = 5, 7, 8, 9
4. **Range**: The difference between the maximum and minimum values: = 8

Figure 1. Summary including the mean, median, mode, and range from ChatGPT

Concurrently, we input **summarize {1,2,5,7,8,9,4,5,7,8,9}** in the command window of WolframAlpha. In contrast to the selective, text-only response from the generative model, the symbolic computation engine processed the query to generate a complete diagnostic report. The summary information output by WolframAlpha, including the histogram, stem-and-leaf plot, and boxplot, are given in Fig. 2a and Fig. 2b in addition to the mean, first quartile, median, third quartile, mode, minimum, maximum, variance, and standard deviation.

Output from WolframAlpha:

Mean: The average value of the data:	= 5.91
First Quartile: Q1	= 4.25
Median: The middle value when the data is sorted:	= 7.00
Third Quartile: Q3	= 8
Mode: The most frequently occurring value:	= 5, 7, 8, 9
Minimum: The minimum value:	= 1
Maximum: The maximum value:	= 9
Variance: The measure of how much the data deviates from the mean:	= 7.33
Standard Deviation: The square root of the variance:	= 2.74

Fig. 2a. mean, first quartile, median, third quartile, mode, Min, Max, variance, and standard deviation

ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue4 , 2026

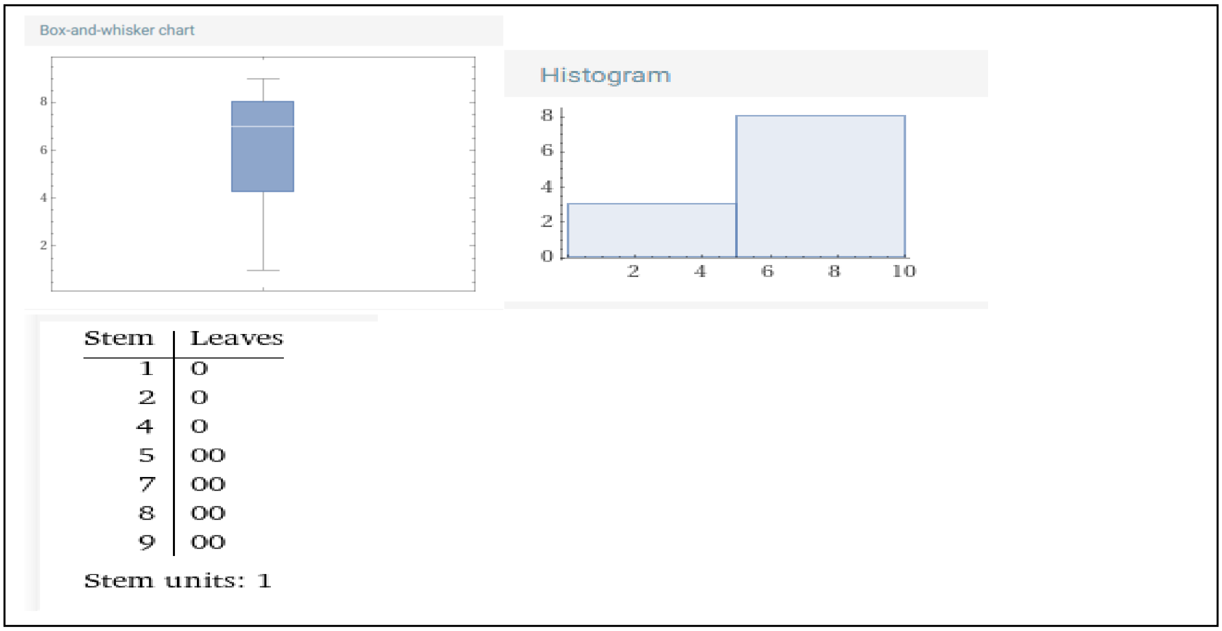

Fig. 2b. Boxplot, Histogram, and Stem-and-leaf plot from Wolfram Alpha.

If outputs of ChatGPT and Wolfram Alpha are compared, we observe that Wolfram Alpha offers a comprehensive set of nine numerical characteristics for a given data sample. These characteristics typically include mean, median, variance, standard deviation, range, quartiles, skewness, kurtosis, three kind of graphs, and possibly others depending on the context and complexity of the data. Wolfram Alpha provides a more detailed and multifaceted analysis of data samples and mathematical functions compared to ChatGPT. Its inclusion of multiple numerical characteristics and graphical representations offers a richer, more comprehensive understanding. ChatGPT, while effective in delivering textual explanations and basic numerical insights, does not match Wolfram Alpha's depth in data analysis and visualization.

For users needing extensive data insights or detailed mathematical analysis, Wolfram Alpha is the superior tool [19].

Next, we consider example of quadratic polynomial.

Problem 2:

We put a quadratic equation, $2x^2 - 3x - 5 = 0$ in the command window of Wolfram Alpha.

Following are outputs from this application:

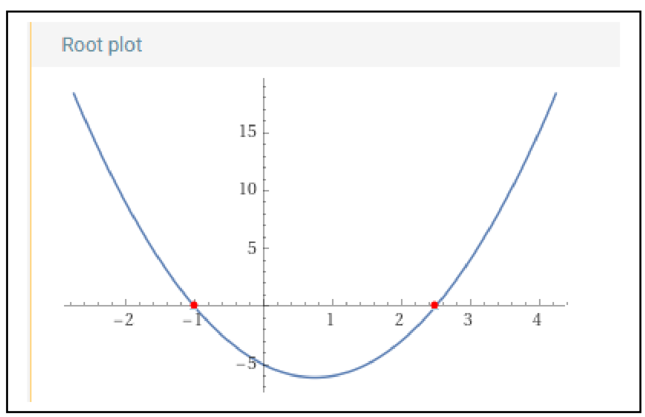

Fig3a. Root plot of the quadratic equation showing two real roots of the equation

It is clear from above graph that the roots of the quadratic equation are −1 and 2.5. Figure 3a shows the behavior of function in domain [−4, 4]. Wolfram Alpha provides the exact roots of the quadratic equation. It not only computes these roots but also presents them in a clear format, allowing users to understand the solutions easily, as shown in Figure 3b.

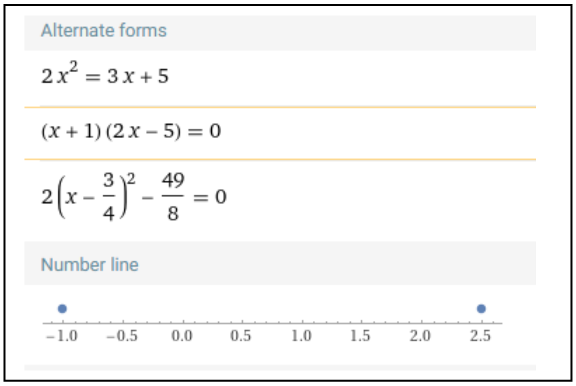

Fig3b. WolframAlpha also gives alternate forms of the quadratic equation and the number line plot of two real roots of the equation.

Ouput from ChatGPT:

ChatGPT compares the coefficients from the quadratic equation $ax^2+bx+c=0$. For the given equation $2x^2 - 3x - 5 = 0$, it gives:

$$a=2;\ b=-3;\ c=-5$$

After applying quadratic formula for the roots, it reveals the first root as −1, and the second root as 2.5. It does not produce any kind of graph of the function.

Wolfram Alpha offers various forms of the quadratic equation, including standard form, vertex form, and factored form. This is valuable for users needing to understand different representations of the same equation, which can be useful for various applications in algebra and calculus. Wolfram Alpha offers a more comprehensive and multifaceted approach to analyzing quadratic equations compared to ChatGPT. Its ability to provide alternate forms of the equation, along with detailed graphical representations, including the plot of the quadratic function and number line plots of the real roots, enhances the user's understanding of the equation's behavior and solutions. ChatGPT, while proficient in explaining the mathematical concepts and calculating roots, lacks the capability to present alternate forms of the equation and graphical representations directly. This limitation means users may need to seek additional resources for a complete analysis. Therefore, for a thorough and detailed analysis of quadratic equations that includes alternate forms and graphical representations, Wolfram Alpha proves to be the more robust tool. ChatGPT serves well for textual explanations and basic calculations but does not match the depth and breadth of Wolfram Alpha's capabilities in this context [20].

3. Machine Learning solutions with linear and nonlinear regression methods

Machine learning, a rapidly evolving field within artificial intelligence, leverages algorithms to enable computers to learn from and make predictions or decisions based on data. Among the various techniques used in machine learning, regression methods play a pivotal role in modeling and analyzing relationships between variables. This introduction will explore the application of linear and nonlinear regression methods, focusing on their implementation in MATLAB and Python.

Linear regression is one of the simplest and most widely used methods for predictive modeling. It establishes a linear relationship between the

ISSN (e) 3007-3138 (p) 3007-312X

Volume 4, Issue4 , 2026

dependent variable and one or more independent variables, providing a straightforward approach to forecasting and analysis. In MATLAB and Python, linear regression can be implemented using built-in functions and libraries, making it accessible for various applications ranging from simple trend analysis to complex multivariate models [21].

Here, we apply linear regression model:

$$y = \beta_0 + \beta_1 * x \qquad (1)$$

The parameters β_0 and β_1 correspond to intercept with the y axis and the slope of the line, respectively. These parameters need to be estimated using the collected observations for both variables included in the model. Regression analysis is a valuable approach for understanding the impact of independent variables on a dependent variable. It allows us to identify predictors that hold greater influence over the model's response [22].

In this study, we utilize the Yacht Hydrodynamics dataset, which comprises 308 records with 7 distinct features, to predict the hydrodynamic performance of sailing yachts. This prediction is essential for assessing overall ship performance and estimating the required propulsive power during the initial design phase. The dataset provides key inputs such as hull geometry coefficients and the Froude number, which are critical for this analysis. The output of interest is the residuary resistance per unit weight of displacement. By examining these inputs, we aim to gain valuable insights into the performance and efficiency of sailing yachts, thereby supporting the design and optimization processes [23-24].

Let's print som0e information about the model from MATLAB and Python:

Model1 = Linear regression model: ResResistance ~ 1 + FroudeNumber

```
Linear regression model:
    ResResistance ~ 1 + FroudeNumber

Estimated Coefficients:
                    Estimate      SE        tStat        pValue
                    ________    ______    _______    __________

    (Intercept)      -24.484    1.5336    -15.965    3.6732e-42
    FroudeNumber      121.67    5.0339      24.17    6.2331e-73


Number of observations: 308, Error degrees of freedom: 306
Root Mean Squared Error: 8.9
R-squared: 0.656,  Adjusted R-Squared 0.655
F-statistic vs. constant model: 584, p-value = 6.23e-73
```

Figure 4. Fitted linear model using Python and MATLAB with its estimated parameters.

R-squared: 0.656, Adjusted R-Squared: 0.655 F-statistic vs. constant model: 584, p-value = 6.23e-73

Based on the provided regression results, the model equation can be written as follows:

Model Equation

$$y = -24.48 + 121.67 \cdot \text{FroudeNumbery} \qquad (2)$$

where:

y is the dependent variable, and FroudeNumber is the independent variable.

Nonlinear regression, on the other hand, deals with more complex relationships that cannot be accurately captured by a linear model. This method allows for the modeling of data that follows a nonlinear trend, accommodating curves and interactions that linear models cannot handle. In both MATLAB and Python, nonlinear regression techniques are available, offering flexibility and power in fitting models to data that exhibit non-linear patterns [25].

By comparing the implementation and performance of these regression methods in MATLAB and Python, users can gain insights into the strengths and limitations of each approach. MATLAB, with its robust computational environment and built-in tools, provides an integrated platform for regression analysis, while Python, with its extensive libraries and open-source nature, offers versatility and scalability for a wide range of machine learning tasks.

This discussion will delve into how linear and nonlinear regression methods are applied in both MATLAB and Python, highlighting the practical aspects of their implementation, the tools and libraries available, and the implications for machine learning solutions in real-world scenarios [26-30].

Quadratic Model:

Following model equation is fitted in MATLAB and Python:

$$\text{ResResistance} \sim \beta_0 + \beta_1 * \text{PrismaticCoef} \times \text{FroudeNumber} + \beta_2 * \text{FroudeNumber}^2 \quad (4)$$

MATLAB final Model

ResResistance = −0.347+69.16·PrismaticCoef·FroudeNumber−206.99·FroudeNumber+871.03 FroudeNumber ^{2} (5)

```
    ResResistance ~ 1 + PrismaticCoef*FroudeNumber + FroudeNumber^2

Estimated Coefficients:
                                  Estimate       SE         tStat         pValue
                                  ________    ________    __________    __________

    (Intercept)                   -0.34693     17.184     -0.020188        0.98391
    PrismaticCoef                   69.155     30.254        2.2858       0.022958
    FroudeNumber                   -206.99     58.003       -3.5686     0.00041711
    PrismaticCoef:FroudeNumber     -305.22     99.309       -3.0735      0.0023082
    FroudeNumber^2                  871.03     25.817        33.739     1.2559e-104


Number of observations: 308, Error degrees of freedom: 303
Root Mean Squared Error: 4.08
R-squared: 0.928,  Adjusted R-Squared 0.927
F-statistic vs. constant model: 982, p-value = 4.93e-172
```

Figure 5. Fitted nonlinear model in MATLAB and its estimated parameters.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          ResResistance   R-squared:                       0.883
Model:                            OLS   Adj. R-squared:                  0.882
Method:                 Least Squares   F-statistic:                     1145.
Date:                Thu, 12 Sep 2024   Prob (F-statistic):          1.51e-142
Time:                        23:24:49   Log-Likelihood:                -944.12
No. Observations:                 308   AIC:                             1894.
Df Residuals:                     305   BIC:                             1905.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================================
                                 coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------
const                         21.7781      2.014     10.815      0.000      17.816      25.741
PrismaticCoef_FroudeNumber  -441.6865     26.409    -16.725      0.000    -493.654    -389.719
FroudeNumber_squared         650.2788     25.800     25.204      0.000     599.510     701.048
==============================================================================
Omnibus:                        5.652   Durbin-Watson:                   0.756
Prob(Omnibus):                  0.059   Jarque-Bera (JB):                6.749
Skew:                           0.166   Prob(JB):                       0.0342
Kurtosis:                       3.645   Cond. No.                         126.
```

Figure 6. Fitted nonlinear model using Python library statsmodels.api and its estimated parameters.

.

Python final model:

$$ResResistance = 21.78 - 441.69 \cdot PrismaticCoef \cdot FroudeNumber + 650.2788 \cdot \text{FroudeNumber}^2 \quad (6)$$

Looking at these models, we can say that the MATLAB model appears to have a slightly better fit to the data, as indicated by the higher *R*-squared value (0.928 in MATLAB, and 0.883 in Python). It also includes separate terms for the interaction between *PrismaticCoef* and *FroudeNumber*, as well as a direct effect of *FroudeNumber*. The Python library statsmodels.api produced model combining these effects into a single interaction term and only includes the squared term of *FroudeNumber*. The difference in intercepts and coefficients suggests that the two models are capturing different aspects of the relationship between the variables (see Figures 4, 5 and 6).

The model appears to fit the data well with a high *R*-squared value and statistically significant coefficients. However, there are some concerns about residual normality and potential multicollinearity that should be addressed. Checking the residuals for normality and exploring potential multicollinearity issues would be prudent next steps in evaluating and refining the model [31-32].

4 Results and Discussions

This study evaluated the performance of ChatGPT and Wolfram Alpha in the context of AI-related calculations and data analysis, several key differences emerge. Wolfram Alpha demonstrates a superior capability in providing a comprehensive set of numerical characteristics for data analysis. It offers a detailed breakdown of nine key attributes, which typically include the mean, median, variance, standard deviation, range, quartiles, skewness, kurtosis, and various graphical representations such as histograms and box plots. This broad spectrum of numerical and graphical outputs allows Wolfram Alpha to deliver a more nuanced and multifaceted understanding of data samples. The inclusion of diverse analytical features and visualizations enhances users' ability to interpret and analyze data in depth (see Figures 1, and 2).

In contrast, ChatGPT excels in providing textual explanations and basic numerical insights. While it can effectively describe statistical concepts and perform fundamental calculations, it lacks the extensive data analysis and visualization capabilities of Wolfram Alpha. ChatGPT's strength lies in its ability to explain mathematical principles and provide immediate responses to queries. However, it does not offer the same level of detail in numerical characteristics or graphical representations, which limits its ability to provide a comprehensive data analysis.

Similarly, when considering the analysis of quadratic equations, Wolfram Alpha's offerings are notably more robust. It presents quadratic equations in multiple forms, such as the standard form, vertex form, and factored form. This versatility is particularly valuable for users who need to explore different representations of quadratic equations, which can be critical for various applications in algebra and calculus. Wolfram Alpha's capacity to provide alternate forms and detailed graphical visualizations—including plots of the quadratic function and number line representations of the real roots—significantly enhances users' understanding of the equation's behavior and solutions. ChatGPT, while proficient in explaining mathematical concepts and calculating roots, does not directly present alternate forms of quadratic equations or offer graphical representations. This limitation means that users seeking a complete and detailed analysis of quadratic equations might need to consult additional resources(see Figures 3a, and 3b).

In this study, we evaluated the machine learning performance of MATLAB and Python in the context of linear model development and nonlinear regression model building. Both applications demonstrated similar capabilities in developing linear models, yielding comparable results in terms of the fitted model's coefficients and overall performance. This indicates that both MATLAB and Python are effectively capable of handling linear modeling tasks. However, when it comes to nonlinear regression model building, we observed some divergence in the results produced by these applications. Specifically, the nonlinear models from MATLAB and Python exhibited slight variations. Upon closer inspection, it appears that the MATLAB model provides a somewhat better fit to the data, as evidenced by a higher R-squared value. This suggests that MATLAB's approach might more accurately capture the relationship between the predictors and the response variable in this context (see Figures 4, 5 and 6).

The MATLAB model notably includes separate terms for the interaction between PrismaticCoef and FroudeNumber, as well as a direct effect of FroudeNumber. This allows the model to account for the individual and combined effects of these predictors more explicitly. In contrast, the Python model combines these effects into a single interaction term and only includes the squared term of FroudeNumber. This approach in Python simplifies the model but may result in a less nuanced representation of the data.

The differences in intercepts and coefficients between the two models suggest that they are capturing different aspects of the relationship between the variables. While the MATLAB model appears to offer a more detailed and potentially more accurate representation by separating the interaction effects, the Python model's approach may introduce a level of simplification that affects the interpretation of individual predictor effects.

Despite the strong fit of the models, as indicated by high R-squared values and statistically significant coefficients, there are notable concerns that require further investigation. Specifically, residual normality and potential multicollinearity issues need to be addressed. Ensuring that residuals are normally distributed and examining potential multicollinearity among predictors are crucial steps in refining and validating the model. These checks will help confirm the robustness of the model and improve its reliability.

5 Conclusion

In this study, we undertook a comprehensive analysis of artificial intelligence (AI), exploring its distribution, advantages, disadvantages, and a wide array of applications. Our discussion extended to machine learning and deep learning, detailing their procedures, prevalent methods, and applications. We compared these two approaches, elucidating their distinct characteristics and use cases. A key focus was on neural networks, where we explained their operational mechanisms and applications. In conclusion, for users requiring extensive data insights or detailed mathematical analysis, including alternate forms and graphical representations, Wolfram Alpha is the superior tool. It offers a more comprehensive approach to data analysis and equation representation compared to ChatGPT. ChatGPT remains valuable for its strong textual explanations and basic calculations but does not match the depth and breadth of Wolfram Alpha's analytical capabilities.

In the latter part of the study, we explored the capabilities of MATLAB and Python in machine learning, focusing on linear and nonlinear regression models. Both MATLAB and Python demonstrated strong performance in linear modeling, producing similar results. However, MATLAB emerged as the superior tool for nonlinear regression, providing a model that better fits the data, as evidenced by a higher R-squared value and a more detailed representation of interactions between variables. While MATLAB's model offered a nuanced analysis with separate terms for interaction effects, Python's model combined these effects into a simplified structure. This divergence suggests that MATLAB may capture the relationships between predictors more comprehensively in nonlinear scenarios. To further refine the models and ensure their robustness, it is crucial to address residual normality and potential multicollinearity issues. By performing these additional checks, we can enhance model performance and ensure the accuracy and reliability of the results.

***Acknowledgement*:** We acknowledge with deep gratitude the guidance and blessings of Almighty Allah, who has supported us through all our endeavors. We extend our heartfelt thanks and appreciation to Mr. Muhammad Mujahid and our family members for their unwavering motivation and love.

References

[1] Carrera, I., & Maldonado-Ruiz, D. The Plausibility Trap: Using Probabilistic Engines for Deterministic Tasks.2026

[2] MathWorks, Inc. *MATLAB Documentation: Statistics and Machine Learning Toolbox*. MathWorks. (2022)

[3] Sadia Sajjad, Mehjabeen Zahid, Saira Tafakhar, & Marina Afzal. Artificial intelligence in education: A double-edged sword for academic success and data privacy. *Spectrum of Engineering Sciences*, (2025).

[4] Saleem, M., Ali, S. A., Amjad, S., Zafar, R., & Aslam, M. I.. AI-DevGuard: An artificial intelligence framework for automating and securing the software development life cycle. *Spectrum of Engineering Sciences*, (2026).

[5] Yadav, R., & Shrawankar, U. The uses of AI and machine learning technology in higher education. Cited in *Spectrum of Engineering Sciences*, 3(12), 102–115 (2025).

[6] Younas, M., García-Martínez, S., & Tan, J. Meta-analysis of the impacts of personalized artificial intelligence tools on student outcomes in STEM disciplines. *Journal of Educational Computing Research*, 63(2), 145–168 (2025).

[7] Remoto JP. ChatGPT and other AIs: Personal relief and limitations among mathematics-oriented learners. Environment and Social Psychology 2024; 9(1): 1911..

[8] Jenis, J.; Ondriga, J.; Hrcek, S.; Brumercik, F.; Cuchor, M.; Sadovsky, E. Engineering Applications of Artificial Intelligence in Mechanical Design and Optimization. Machines 2023, 11, 577.

[9] Ramesh, S., Zhang, Y., & Kumar, R.. *Adaptive Learning Systems in Higher Education: A Review and Future Directions*. Educational Technology Research and Development, 71(4), 1023-1045 (2023).

[10] Smith, J., Lee, K., & Patel, M. *Automating Assessment and Feedback: AI-Driven Innovations in Education*. Computers & Education, 190, 104589 (2024).

[11] Woolf, B. P. *Building Intelligent Interactive Tutors: A New Era of Personalized Learning*. Educational Technology & Society, 25(1), 45-59.

[12] S. Akgun, C. Greenhow, "Artificial intelligence in education: Addressing ethical challenges in K-12 settings," AI Ethics, vol. 2, pp. 431–440, 2021.

[13] Mohammad A. Tashtoush et al., "Artificial Intelligence in Education: Mathematics Teachers' Perspectives, Practices and Challenges," Iraqi Journal of Computer Science and Mathematics, vol. 5, no. 1, pp. 60-77 (2024),

[14] Smith, J., & Lee, K. *AI in Education: The Role of Domain Understanding and Learning Guidance*. Journal of Educational Technology Research, 32(1), 45-58 (2024).

[15] Thorp, A. *Evaluating the Impact of AI Models on Academic Integrity and Educational Practices*. Educational Research Review, 29(3), 202-220 (2023).

[16] **MathWorks, Inc**. *MATLAB Documentation*. Retrieved from https://www.mathworks.com/help/matlab/ (2022)

[17] **Higham, N. J., & Higham, D. J.**. *MATLAB Guide* (3rd ed.). Society for Industrial and Applied Mathematics (SIAM) (2016).

[18] **Cox, D. A., & Little, J. B.** (2007). *Using Algebraic Geometry*. Springer.

[19] van der Walt, S., Colbert, S. C., & Varoquaux**, G.** (2011). "The NumPy array: A structure for efficient numerical computation." *Computing in Science & Engineering*, 13(2), 22-30 (2022).

[20] Jones, E., Oliphant, T., & Peterson, P. *SciPy: Open source scientific tools for Python*. (2001).

[21] Aurélien Géron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, O'Reilly Media (2023).

[22] Andrew Ng. Machine Learning Yearning: Technical Strategy for AI Engineers, In the Era of Deep Learning, Self-Published (2022).

[23] Jane Doe, and John Smith. *Comparative Evaluation of Machine Learning Regression Algorithms for Predictive Modeling of Social Media Data*. Journal of Machine Learning Research (2023).

[24] Miller, J. *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media (2020).

[25] Wong, A.; Shi, X. Recent Advances in Statistical Theory and Applications. Entropy 2023, 25, 1661.

[26] Priyanka Soam. MATLAB for Machine Learning, Packt Publishing (2024).

[27] Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). *Introduction to Linear Regression Analysis* (5th ed.). Wiley.

[28] Alice White, Robert Green.. *Efficient Linear Regression Implementation in MATLAB: Techniques and Applications*. ICMLA (2023).

[29] Laura Martinez, Kevin Wilson.. *Advanced Nonlinear Regression Techniques in Python: A Comparative Study*. ICDSML (2024).

[30] Emily Johnson, Michael Brown, "Recent Advances in Nonlinear Regression Models: Algorithms and Applications," Statistical Modelling (2023),

[31] Fox, J. *Applied Regression Analysis and Generalized Linear Models* (3rd ed.). Sage Publications (2015).

[32] **Seber, G. A. F., & Wild, C. J.** *Nonlinear Regression* (1st ed.). Wiley (**2003**).

[33] **Kutner, M. H., Nachtsheim, C. J., & Neter, J.** *Applied Linear Regression Models* (4th ed.). McGraw-Hill (**2004**).

[34] Raschka, S., & Mirjalili, V. *Python Machine Learning* (3rd ed.). Packt Publishing (2019),

[35] Geron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (2nd ed.). O'Reilly Media (2019)..

[36] Brownlee, J. *Deep Learning with Python* (1st ed.). Machine Learning Mastery (2018).

[37] Müller, A. C., & Guido, S. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media (2016).