ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

The Application of Machine Learning-Driven AI in Model Rocketry: Improving Flight Simulation and Data Analysis in TARC

Xinyu Du

Diamond Bar High School, Diamond Bar, California, USA

Abstract: *The Team America Rocketry Challenge (TARC) requires student teams to design rockets capable of meeting strict altitude and flight-time requirements. Traditional rocket prediction methods rely heavily on physics-based simulations and repeated experimental launches, which may not fully account for real-world aerodynamic variations. This study investigates the application of machine learning as a supplementary tool for predicting rocket flight performance in model rocketry. A physics-based flight simulator was first developed using classical mechanics, thrust modeling, and aerodynamic drag equations to estimate rocket trajectory, apogee, and flight time. A Random Forest Regression model was then trained using approximately 500 flight samples containing rocket parameters and measured apogees. The machine learning model used rocket weight, rocket height, launch angle, and wind speed as input variables. Model performance was evaluated using Mean Absolute Error (MAE). Results showed that the physics-based simulation predicted an apogee of approximately 697.77 ft, while the machine learning model achieved an MAE of approximately 21.18 ft and predicted an apogee of 759.52 ft for the tested rocket configuration. The relatively small difference between the two prediction approaches demonstrates that machine learning can effectively approximate complex flight behavior while significantly reducing computation time. The findings suggest that machine learning can serve as a practical supplementary tool alongside traditional physics-based simulations for improving rocket flight prediction and design optimization in TARC.*

Keywords: *Machine Learning, Model Rocketry, Flight Simulation, Artificial Intelligence, Rocket Apogee Prediction*

1. Introduction

The Team American Rocketry Challenge (TARC) is a contest where high school teams design, build and fly model rockets to some strict goals every year. For instance, rockets must carry a raw egg at least 750 ft high and land within 36-39 seconds this year [1]. To reach these goals, teams rely on computer simulation applications such as OpenRocket, and limited test flights to predict altitude, flight time, and stability. Machine learning and Artificial Intelligence offer new tools to improve these predictions and analyze flight data. Machine learning refers to algorithms that learn relationships from data rather than relying entirely on explicit programming rules [2]. In rocketry, this could mean teaching a computer from past flights how to adjust a simulation or control a flight.

Neural networks and other advanced machine learning models have already demonstrated applications in aerospace engineering. Researchers at the University of Texas at Austin used machine learning methods to accelerate rocket engine simulations that previously required extensive computational resources [3]. NASA-related research has also explored machine learning systems capable of identifying anomalies in rocket engine simulations. These studies demonstrate the growing importance of combining physics-based modeling with data-driven approaches.

Machine learning applications have also begun to appear in model rocketry. A 2025 study trained neural networks on approximately 10,000 simulated rocket flights to estimate drag coefficients and thrust factors from observed apogee data [4]. Their findings showed that machine learning models could improve prediction accuracy beyond standard simulation estimates. Such approaches are particularly valuable in TARC because teams often possess limited experimental flight data and must continuously adjust rocket parameters to meet competition requirements.

This research investigates whether machine learning can supplement traditional rocket simulations for predicting flight performance. A physics-based rocket simulation and a Random Forest Regression

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

model were developed and compared. The study evaluates the ability of machine learning to approximate rocket apogee while reducing computational complexity and improving adaptability to real-world conditions.

2. Application of Machine Learning in Rocketry

2.1 Aerospace examples

In professional rocketry, ML is already used to speed up design and find problems. NASA and university researchers blend physics and data-driven learning to make fast simulations of rocket engines. For instance, scientists at UT Austin trained ML models on high -fidelity engine data so simulations that once took weeks can now run in seconds. These reduced-order models match the key behavior with much less computation. Similarly, a NASA project built an ML system to analyze rocket engine simulation outputs and automatically identify faults in the engine data. These examples show ML can process complex rocket data from sensors to help engineers predict performance or catch abnormalities.

2.2 Model rocketry examples

In hobby/model rocketry, researchers and hobbyists have begun to apply ML ideas. An amortized inference study in 2025 took around 10,000 simulated model rocket flights to train a neural network. After training on synthetic data, their neural net could predict real rocket flights' apogee much better than the standard OpenRocket estimation. This model learned to estimate the rocket's drag coefficient and motor thrust factor from a single apogee reading, effectively calibrating the simulation with limited real data. In practical terms, a TARC team could use this approach to adjust for variations in motors or rocket finish, as simply plug in measured flight height and let the ML mode infer other factors to improve other simulations.

Another discussion from the model rocketry community highlighted how ML could help teams with scarce flight data. Typically, a team simulates its rocket, using final weight, etc. to estimate apogee and time, then flies it, notes the real altitude and time and manually updates the model. Experts point out this loop is error prone when data are few, such as overfitting or underfitting. If teams equipped rockets with small flight computers, they could collect richer data on each flight. Then an ML anomaly detection system could compare the censored data to simulated flights and flag exactly where the real flight differs the most. Like this, it would automatically show if the actual drag was higher than assumed, or if stability was lower than expected. By automating this analysis, teams could update their simulation parameters more accurately between tests.

3. ML Models in Rocketry

Regression vs. Classification: In rocket data, regression could also be used whenever we want to predict a number, specifically, altitude. For instance, a team could build a regression model to estimate peak altitude from inputs like rocket weight, motor impulse, or wind speed. By fitting a curve to past test data, the model learns how each factor affects the altitude. For example, if heavier mass consistently gave lower apogee in tests, regression could quantify that drop so future predictions would be more precise.

Classification could label flights into categories as I have previously mentioned. For example, each test flight is classified as "reaching target altitude" versus "not reaching target altitude," or "flight is theoretically straight" versus "flight will be unstable." Classifiers such as the decision-tree classifier might learn from past flights: if wind speed is greater than 5 mph, then likely "too low time", etc. Classification helps teams quickly sort which design changes lead to hitting competition specs. It could also detect abnormalities such as if a flight's sensor pattern looks unlike any stable test, flag it for further reviews.

The great application actually highlighted the usage of Neural networks. Neural nets can do either regression or classification, but they actually pop off when relationships are complex. As one study noted, a neural network learns to invert the rocket's physics model: given an observed apogee and motor specs, it outputs estimates of drag and thrust efficiency. Because neural networks have many layers of weighted nodes, they can capture unobvious effects. A neural net could combine inputs like rocket geometry, weight, motor stats, weather to predict outcomes we needed (altitude and time) more flexibly than a simple formula. The problem is that they need more data to train, but simulated data can also be used as

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

a training set.

4. Methodology and Implementation

4.1 Physics-based flight simulation

Before applying machine learning, a physics-based rocket flight simulation was implemented to model the rocket's trajectory and estimate the expected apogee and flight time. The simulation uses classical mechanics and aerodynamic drag equations to compute rocket motion over time.

The model represents the rocket using several key physical parameters including dry mass, propellant mass, thrust, drag coefficient, and body diameter. The thrust profile was modeled based on the characteristics of an F41-5W motor, with a burn time of approximately 1.1 seconds and an average thrust of 42N.

During powered flight, the rocket's mass decreases linearly as propellant is consumed. After burnout, the rocket continues in a ballistic trajectory until gravity and drag slow the ascent and eventually cause descension.

The aerodynamic drag force was calculated using the standard drag equation:

$$F_d = \frac{1}{2}\rho C_d A v^2 \quad (1)$$

Where:

- ρ is air density
- C_d is the drag coefficient
- A is the reference area that air would have contacts in
- v is the velocity of the rocket

Air density varies with altitude and was modeled using an exponential atmosphere approximation. This accounts for decreasing air density as altitude increases.

The rocket trajectory was simulated using a semi-implicit Euler numerical integration method, updating velocity and height at minor time increments such as $\Delta t = 0.001\,s$. This time-stepping approach allows the model to approximate continuous rocket motion and two additional aerodynamic systems were included in the simulation for unique cases:

(1) Air brakes.

Two deployable airbrake panels were modeled to increase drag when the rocket reaches a predefined altitude threshold. When the rocket reaches approximately 720 ft in this test case, the airbrakes deploy and increase the effective drag area.

(2) Parachute deployment.

A parachute recovery system was also modeled. The parachute deploys at a specified time after launch and gradually inflates over a short inflation period. During inflation, the effective drag area increases smoothly to simulate realistic parachute deployment behavior.

The simulation records rocket altitude and time throughout the flight. The maximum altitude apogee is determined by taking the maximum value of the simulated height data.

This physics-based simulation provides a baseline prediction that can be compared with machine learning estimates.

4.2 Machine Learning Model for Apogee Prediction

To complement the physics simulation, a machine learning model was developed to predict rocket apogee directly from key flight parameters. The goal of the ML model is to learn relationships between rocket configuration variables and flight outcomes using historical or simulated flight data.

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

4.3 Dataset

The training data were stored in an around 500-sample-sized CSV dataset containing previous rocket flight parameters and their measured apogees. Each row of the dataset represents a single flight sample. The model uses the following input features that varies from the physics-based simulator:

- Rocket weight (grams)
- Rocket height (centimeters)
- Launch angle (degrees)
- Wind speed (m/s or mph)

Resulting variable:

- Apogee (ft)

Before training, the dataset was cleaned to remove unrealistic values and missing data. Flights with extremely low or extremely high apogees were filtered out to reduce noise and ensure stability.

4.4 Data Preprocessing

Feature normalization was performed using the StandardScaler function from the Scikit-learn library [5]. Standardization transformed each variable to a distribution with mean 0 and standard deviation 1. The dataset was divided into 80% training data, 20% testing data. The separation would ensure that the model was evaluated on data that had not been observed during training.

4.5 Machine Learning Model Architecture

The machine learning model used in this study is a Random Forest Regression model, implemented using the Scikit-learn library.

Random Forest is an ensemble learning method that combines predictions from many individual decision trees. Each tree learns a different set of decision rules from the data, and the final prediction is the average output of all trees. This approach improves prediction accuracy and reduces overfitting compared to a single decision tree.

The Random Forest model was configured with the following parameters:

- 300 decision trees ($n_{estimator} = 300$)
- Max tree depth = 6
- Fixed random seed for reproducibility

These parameters allow the model to capture nonlinear relationships between rocket parameters and flight altitude.

4.6 Model Training and Evaluation

The model was trained using supervised learning. During training, the algorithm learns patterns that relate the input rocket parameters to the resulting apogee. Model performance was evaluated using Mean Absolute Error (MAE), which measures the average absolute difference between predicted and actual apogee values.

$$MAE = \frac{1}{n}\Sigma|y_{true} - y_{predicted}| \quad (2)$$

A lower MAE represents that the model's predictions are closer to the actual measured values.

4.7 Model Deployment and Prediction

Once trained, the machine learning model and the data scaler were saved to disk using the Joblib serialization library. Saving the trained model allows it to be reused without retraining. During simulation, the trained model is loaded and used to predict apogee from new rocket parameters. The input variables are first scaled using the previously trained scaler, and then the model generates a predicted apogee value.[6]

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

4.8 Comparison between Physics Simulation and ML Prediction

To evaluate the effectiveness of the machine learning approach, the predicted apogee from the ML model is compared with the apogee calculated from the physics-based simulation. The comparison allows researchers to determine whether the machine learning model can approximate or improve upon traditional physics-based flight prediction methods. The physics simulation also generates a time vs altitude trajectory graph, which provides additional insight into the rocket's flight profile and recovery system behavior.

5. Results

5.1 Physics-based Prediction Performance

The physics-based rocket flight simulation generated a full trajectory of the rocket's altitude over time. Using the numerical integration model described in the methodology section, the simulation calculated velocity, drag forces, and gravitational effects throughout the entire flight. The apogee was determined by taking the peak value of the simulated height data. The simulation also provided the total flight duration from launch until the rocket returned to ground level. The simulation results indicated a predicted apogee of approximately **697.772 ft (214.21 m)** with a total flight duration of **31.233 seconds** based on the various inputs. The trajectory of the rocket's flight is shown in Figure 1, which plots altitude as a function of time.

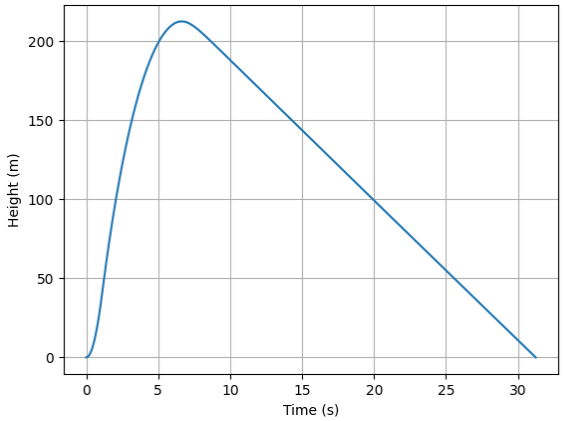

Figure 1. Rocket altitude (m) vs. time from physics-based simulation.

This graph illustrated the different phases of the rocket flight:

(1) Powered ascent, during which the rocket accelerates due to motor thrust.

(2) Coasting phase, where the rocket continues upward after motor burnout.

(3) Apogee, the point at which vertical velocity becomes 0.

(4) Descent and recovery, when the parachute deploys and increases aerodynamic drag.

The deployment of airbrakes and the parachute increases the effective drag coefficient and significantly reduces descent velocity, producing the smooth recovery profile as shown. To note, the difference between the Physical simulation and the actual launch data collected with the settings incorporated was **22.264 ft.**

5.2 Machine Learning Based Prediction Performance

The random Forest Regression model described in the methodology section was trained using historical flight data and simulated data. The model used four input parameters according to the data recorded manually in the datasets: rocket weight, height, launch angle, and wind speed.[7]

After preprocessing and feature normalization, the dataset was divided into training and testing sets using an 80/20 split. The model was trained on the training subset and evaluated using the testing subset, which contained flight data that the model had not previously seen. This approach ensures that the evaluation reflects the model's ability to generalize to new rocket configurations rather than simply

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

memorizing the training data that led to overfitting that limits the validity.

Model accuracy was measured using the Mean Absolute Error (MAE) which represented the average absolute difference between the predicted apogee and the true measured apogee. The trained Random Forest model achieved an MAE of approximately **21.178 ft**, which corresponds to approximately 2.8% of the 750 ft target altitude. This means that, on average, the models predicted apogee differed from the actual value by **21.178 ft**.[8]

Considering that TARC competition flights in 2026 target altitudes near 750 ft, an error on the order of tens of feet represents a relatively small percentage of the total flight altitude. This indicates that the model successfully learned meaningful relationships between rocket parameters and flight outcomes. Random Forest models are particularly well suited for this task because they can capture nonlinear relationships between features, such as how increased rocket mass could reduce altitude.

After training, the machine learning model was integrated into the simulation program. For a test rocket configuration with the following parameters as default:

(1) Rocket weight: 520 g

(2) Rocket height: 71 cm

(3) Launch angle: 0°

(4) Wind speed: 0 m/s

The model generated a predicted apogee of approximately **759.52 ft**.

This value was then compared to the apogee predicted by the physics-based simulation. The physics simulation produced an estimated apogee of approximately **697.77 ft**, resulting in a difference of roughly **61.75 ft** between the two approaches. The relatively small difference between the predictions suggests that the machine learning model is capable of approximating the results of a physics-based flight simulation using only a small set of input variables. Unlike the physics model, which requires detailed aerodynamic equations and numeric integration over time, the ML model can generate predictions almost instantly once trained under relative precision.

Another advantage of the machine learning approach is its ability to incorporate real-world flight data. Physics simulations rely on assumed values for parameters such as drag coefficients, which may differ from actual conditions due to manufacturing differences, surface finish, or atmospheric variability. In contrast, a machine learning model can learn these effects from historical flight data. As more test flights are recorded and added to the dataset, the model can be retrained to further improve prediction accuracy.[9]

Overall, the results demonstrate that machine learning provides a practical supplementary tool for predicting rocket flight performance. While physics-based simulations remain essential for understanding the underlying mechanics of rocket motion, and pattern effectiveness of rocket factors.

6. Discussion

The results of this study demonstrate that both physics-based simulation and machine learning approaches are effective tools for predicting rocket flight performance, each with distinct advantages and limitations. The physics-based model provides a detailed representation of the rocket's motion by explicitly modeling forces such as thrust, gravity, and aerodynamic customizable deployable drag. This approach allows for a clear understanding of the underlying mechanics of flight and enables engineers to analyze how specific design parameters influence trajectory. However, the accuracy of physics-based simulations depends heavily on the precision of input parameters, such as the detail-calculation-required drag coefficient, air density, etc, which are often estimated and may differ from the real world condition.

In contrast the machine learning model offers a data-driven approach that learns patterns directly from historical or simulated flight data. The relatively low MAE indicates that the model is capable of capturing meaningful relationships between rocket parameters and apogee. Unlike the physics-based model, the machine learning approach does not require explicit knowledge of aerodynamic equations and constants while generating predictions rapidly once trained. This makes it particularly useful for iterative design processes, where quick adjustments to rocket parameters are needed to meet competition constraints.

The comparison between the machine learning prediction and the physics-based simulation shows

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

that the two methods produce similar apogee estimates, with only a modest difference between them. This suggests that the machine learning model can approximate the results of a more computationally intensive simulation using a limited set of input variables. However, it is important to note that the machine learning model's performance is highly dependent on the quality and diversity of the training dataset. If the dataset does not adequately represent a wide range of flight conditions, the model may struggle to generalize to new scenarios, particularly under extreme conditions such as high wind speeds or irregular launch angles.

Another key advantage of the machine learning method is its ability to incorporate real-world flight data over time. While physics-based models rely on fixed assumptions, machine learning models can be continuously improved by retraining on new data collected from test launches. This allows the model to implicitly account for factors that are difficult to model analytically, such as minor variations in rocket construction, surface roughness, or environmental conditions. As a result, machine learning has the potential to reduce systematic errors that arise from imperfect assumptions in physics-based simulations.

Despite these advantages, machine learning should not be viewed as a replacement for physics-based modeling, but rather as a complementary tool. Physics models provide interpretability and theoretical grounding, while machine learning offers adaptability and speed. A hybrid approach, where machine learning is used to measure or correct physics-based simulations, may provide the most effective solution for improving prediction accuracy in model rocketry.

Overall, this study highlights the practical value of integrating machine learning into model rocketry workflows. By combining data-driven methods with traditional simulation techniques, TARC teams can achieve more accurate predictions, optimize rocket design more efficiently, and better adapt to real-world flight conditions.

7. Conclusion

This study investigated the application of machine learning as a supplementary tool for predicting rocket flight performance in the Team America Rocketry Challenge (TARC). By developing both a physics-based simulation and a Random Forest regression model, the research evaluated the effectiveness of traditional analytical methods compared to data-driven approaches. The physics-based simulation produced a predicted apogee of approximately 697.77 ft with a realistic flight trajectory, while the machine learning model achieved a Mean Absolute Error of 21.178 ft, and generated a predicted apogee of 759.52 ft for a given test configuration.

The results indicate that the machine learning model is capable of producing reasonably accurate predictions, with errors on the order of only a few percent relative to the target altitude of 750 ft. Additionally, the relatively small difference between the machine learning prediction and the physics-based simulation demonstrates that data-driven models can approximate complex aerodynamic behavior without requiring explicit physical equations. This highlights the potential of machine learning to significantly reduce computation time while maintaining useful predictive accuracy. However, each approach has inherent limitations. Physics-based models provide interpretability and a deeper understanding of rocket dynamics but rely on assumptions that may not fully reflect real-world conditions. In contrast, machine learning models depend heavily on the quality and diversity of training data and may struggle to generalize beyond the range of observed inputs. Therefore, machine learning should not replace physics-based simulation, but rather complement and assist it.

To wrap up, this research demonstrates that integrating machine learning into model rocketry workflows can enhance prediction accuracy, improve design efficiency, and better account for real-world variability. Future work may explore more advanced models such as neural networks, larger and more diverse datasets, and hybrid approaches that combine physics-based simulations with machine learning corrections to further optimize rocket performance in competitive settings.

References

[1] Aerospace Industries Association. "Competition Guidelines." Team America Rocketry Challenge, Available from: https://www.rocketcontest.org
[2] IBM. "Machine Learning." IBM Technology Topics, Available from: https://www.ibm.com/topics/machine-learning.
[3] Fears, Darrell. "Scientific Machine Learning Paves Way for Rapid Rocket Engine Design." UT

ISSN 2616-5775 Vol. 9, Issue 6: 1-8, DOI: 10.25236/AJCIS.2026.090601

Austin News, 16 Apr. 2020, Available from: https://news.utexas.edu/2020/04/16/scientific-machine-learning-paves-way-for-rapid-rocket-engine-design.
[4] Kamath, Sreejith, et al. "Amortized Inference for Rocket Flight Performance." arXiv, 2025, Available from: https://arxiv.org/html/2512.22248v1.
[5] Pedregosa, Fabian, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 2011. Available from: https://scikit-learn.sourceforge.net/dev/about.html
[6] McKee, Martin Jay. "Newsletter #616." Apogee Rockets Education, Available from: https://apogeerockets.com/education/downloads/Newsletter616.pdf.
[7] Harris, Charles R., et al. "Array Programming with NumPy." Nature, 2020, 585(7825): 357–362.
[8] Hunter, John D. "Matplotlib: A 2D Graphics Environment." Computing in Science & Engineering, 2007, 9(3): 90–95.
[9] Virtanen, Pauli, et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." Nature Methods, 2020, 17(3): 261–272.