HyperProphet: an AI/ML Framework in Hypervelocity Impact Studies for Space Applications

1st Himanshi Sharma
B.Tech trainee, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
himanshisharma0512@gmail.com

2nd Ria Madan
B.Tech trainee, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
riamadan0109@gmail.com

3rd Vishal Sharma
B.Tech trainee, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
04vishalsharma04@gmail.com

4th Himansi Dhuri
B.Tech trainee, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
dhurihimansi@gmail.com

5th Anil Kumar
Scientist, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
anil.kumar.tbrl@gov.in

6th Gaurav Suman
Research Assistant, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
gaurav.suman976@gmail.com

7th Vikas Bhardwaj
Scientist, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
vbhrdwj.tbrl@gov.in

8th Pal Dinesh Kumar
Scientist, Ballistics Evaluation Hypervelocity Impact
TBRL
Chandigarh, India
dineshpalb@yahoo.com

doi: https://doi.org/10.37285/bsp.SACAD2025.21

***Abstract*—Extensive research into hypervelocity impact testing (HVIT) is necessary to develop protective measures for spacecraft and satellites due to the increasing threat posed by orbital debris in low Earth orbit (LEO). To replicate such impacts, two-stage light gas guns (2SLGGs) are frequently used; however, each trial entails significant time and costs. To offer a more affordable substitute for conventional trial-based testing, a machine learning (ML) based application, HyperProphet is put forth in this study to forecast the projectile velocity of 2SLGGs using experimental parameters. A dataset of 96 samples from past trials was used to assess regression-based, ensemble, and kernel-based regression models, which were further optimized. ElasticNet and Support Vector Regression (SVR) showed the highest post-optimization accuracy (~85%), showcasing their resilience to nonlinear trends and multicollinearity in small datasets. The findings advocate the potential of ML as an adjunct to physical HVIT systems, especially for initial testing and parameter estimation.**

Keywords—Hypervelocity impact testing; Machine learning in space engineering, Projectile velocity prediction, Two-stage light-gas gun, Regression models

I. INTRODUCTION

Gas guns are designed to achieve very high velocity by the compression of gases. Two-stage light gas gun (2SLGG) are excellent devices to achieve hypervelocity of around 8km/sec. Within the system, propellant burns, creating enough pressure to push the piston, which in turn compresses the light gas like hydrogen. When the pressure created due to compression of light gas becomes high enough, the petal valve bursts open, which propels the projectile towards the target. These systems are essential in aerospace, defence, and planetary science to simulate high-speed impacts, including micrometeoroid impacts and ballistic impacts. Hypervelocity testing in a controlled lab environment is important because the collision of spacecraft with even a tiny object can result in grave consequences due to their high kinetic energy. Therefore, to develop protective armour, it is essential to simulate objects at hypervelocity in a controlled lab environment[1]. To improve shielding technologies, like the Whipple shield, which is designed to safeguard against damage caused by debris clouds and shock waves, the impacts of such events are carefully examined[2]. However, this testing of materials requires a lot of resources and money. We aim to remediate these issues in an accessible fashion using an ML-based software to predict the muzzle velocity of the gun. This approach can be applied to develop and test spacecraft armour by optimising features of the gun, thereby reducing extensive trial factors and cutting costs.

II. LITERATURE REVIEW

This section reviews key literature relevant to hypervelocity impact testing and emerging machine learning based approaches. Table I outlines existing experimental methods, computational models, and gaps where data-driven solutions may offer improvements.

TABLE I. LITERATURE REVIEW

S.no	*Author*	*Technique*	*Description*	*Citation*
1	Murtaugh et al.	Regression models for predicting Muzzle Velocity	The research looked at regression models which the authors used to predict the muzzle velocity based on 10 launch parameters. Majority of the models indicated minimal improvements from the linear regression but the neural network was superior to both the Gaussian process and physics-based models.	3
2	Shojaei et al.	Random Forest Regression	The study uses machine learning models to predict velocity based on 211 experimental datasets available. Random Forest showed a maximum accuracy of 94%.	4
3	Whipple et al.	Analytical Estimation	This study performs a theoretical analysis to estimate the probability of meteorite penetration on a pressurized space vessel. It was concluded that a vessel with a 0.41-inch steel skin would face one penetration every 50 years. Therefore, a dual-layer shield design was proposed.	5
4	Raissi et al.	Traditional Simulations	Numerical methods are accurate but computationally demanding and less generalizable.	6
5	Tibshirani	Lasso Regression	This study introduces the Lasso method that performs shrinkage as well as variable selection in linear models. It sets certain coefficients to zero, reducing its error rate.	7

HVIT remains limited by costly and complex infrastructure. Traditional simulations, although accurate, are computationally intensive and not easily generalizable. Machine learning offers promise but is underutilized due to data paucity and limited domain-specific adaptation.

III. METHODOLOGY

This section briefly describes the steps followed to achieve the ML model for hypervelocity prediction. HyperProphet is an ML prediction software complete with a user-friendly graphical user interface (GUI). HyperProphet enables the user to predict the velocity of the projectile without the need for a 2SLGG laboratory setup. In fig.1 the algorithm discusses the working scheme of HyperProphet.

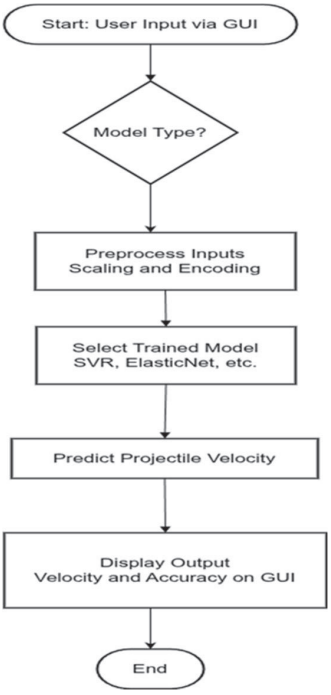


Fig. 1: Algorithm for HyperProphet

For prediction, the user must provide seven inputs: projectile shape, projectile dimensions, calibre, propellant mass, projectile mass, total mass with sabot, and sabot length. The development process is further explained in the following subsections.

A. Data Collection

The first step in the development of HyperProphet was data collection. Historical records of experimental data of two-stage light gas gun testing, dating back to 2006, were compiled into a dataset in a CSV format. Each data entry was carefully reviewed, and missing or incomplete values, such as projectile and sabot mass, were retrieved through lab measurements to avoid the use of imputation and keep high fidelity. This ensured a better level of data integrity than typical imputation.

B. Feature Engineering

Aspects of feature engineering and selection were critically undertaken to improve the performance and accuracy of the machine learning models. For example, the projectile and sabot masses were included to create "Moment of Inertia" and "Total mass", which were derived features that are highly related and significant to the activity of mass and velocity. In total, seven derived features were engineered from the fundamental features, i.e., shape-specific drag coefficient, surface area, volumes, ratio of different physical properties like surface_area/volume, which were used to account for the variation across different shapes and contributed to improvements in prediction and enhancement in correlativity. It provided an improved perspective on how shape-specific properties interact with velocity.

C. *Data Preprocessing and Feature Selection*

Certain preprocessing methods were used to prepare the

dataset for final model training. Initially, box plots were used to visualize feature distributions and outlier detection. Outlier removal was conducted using the Z-score method on the target variable, and all values beyond ±3 standard deviation were removed. Then, the numeric features were standardized to a zero mean and unit variance. Categorical variables (i.e., shape and material) were changed to one-hot encoding. These preprocessing steps improved the overall data quality to reduce bias and to improve model performance. Heatmaps were employed for correlation analysis to determine any multicollinearity between features, allowing the exclusion of redundant features. Finally, Recursive Feature Elimination (RFE) was applied to determine the most suitable inputs, and rank the features systematically to select the features that contribute most to enhancing model accuracy.

D. Model Selection and Training

A variety of machine learning algorithms, ranging from basic linear models to advanced ensemble and neural-network models, were evaluated to identify both linear and non-linear patterns in stereotypic projectile velocity data. The model was trained for a gas gun having two types of calibre, i.e., 40mm and 29mm, using powder mass ranging from 1300 to 3400grams. Moreover, the projectiles used were of different shapes, like a cube, cuboid, cylinder, sphere, and disc, with different dimensions weighing between 0.18g and 97.122 g when measured without a sabot, and when measured with a sabot, it range between 15.64g and 254.3g. The petal burst pressure throughout the data was mainly taken as 470 bar, while in some cases it was 1200 bar. The algorithms used to build predictive models were selected based on how well they could model the characteristics of the dataset:

- Linear Models: To establish baseline performances, we first used Ridge, Lasso, and ElasticNet regression models. These models were simple to interpret and were also effective in handling feature multicollinearity. Linear models provided high accuracy in terms of R2 score, while they performed well for only specific shapes when tested against the experimental values.
- Non-Linear Models: Non-linear algorithms were used to find more complex associations within the data. Models like SVR, Random Forest, and XGBoost were used to evaluate the performance of the non-linear models against the data. Most of these models demonstrated relatively lower accuracy, as indicated by the R2 score. However, when tested against experimental values, they exhibited a lower error margin across all shapes.

E. Optimisation of Models

Optimisation is a crucial step in the application of ML to improve the overall performance and generalisation capabilities of the model. The key is to fine-tune each aspect of the process, from the selection of relevant components to initial settings and internal parameters of the model. The work targeted optimising the feature set, tuning model parameters, and validating performance within a rather small computing environment.

- Grid Search: Hyperparameter tuning was implemented to use the model capacity for overfitting and to optimize accuracy. Grid Search was employed to search through multiple hyperparameter combinations. It uses brute force to pick the best combination of hyperparameters for a model. It tests all possible permutations and combinations for specific parameters and judges the model strength accordingly. It provides a systematic and robust idea to produce higher accuracy[8]. It was extensively used across all models to improve accuracy and overall model functionality to their optimal potential
- Recursive Feature Elimination (RFE): RFE allowed for the systematic reduction of the amount of input features to assist with learning and model generalisation by ranking features and recursively removing the features that contributed the least to the model efficiency. This method helps identify the key features and systematically ranks them based on model performance so that the model can direct its focus on features that are necessary for the prediction of projectile velocity[9]. This method was used across the models for achieving the optimal feature selection for improving accuracy.

F. Model Evaluation

Three different measures of performance were employed: R^2, the proportion of variance in the dependent variable that is explained by the independent variables. Mean squared error (MSE), which squares the errors, making larger errors more visible, and mean absolute error (MAE): the average of the absolute differences between predictions and true values. Most promising models are shortlisted and are explained further in detail in section 5.

G. *GUI Development and Integration*

To facilitate user interaction with the prediction and simulation models, the graphical user interface (GUI) for this project was designed in the Tkinter library in Python, which creates a platform for human engagement. The GUI allows for the input of variables such as projectile mass, shape, and material, user choice of available trained machine learning models, and produces real-time velocity prediction. While under the hood, the GUI presents the underlying principles and equations capturing performance measures and provides embedded visualisation tools, there is multi-page support driving front-end user interaction. This combines the backend technical work with a user-friendly front-end to allow for intuitive engagement with hypervelocity predictions.

IV. FUNCTIONAL OVERVIEW OF TWO-STAGE LIGHT GAS GUN

The most crucial part of HVIT is accelerating an object to the required velocities. To generate hypervelocity, the device used in this study is a 2SLGG. The 2SLGG can simulate hypervelocity collisions in controlled environments. A target is subjected to projectiles exceeding 3km/s shot from the gun.1 The impact is analysed according to perforation, fracture, and damage. Fig. 2. shows the schematic of the working of the 2SLGG. In the initial stage of the gun, a conventional explosive or powder charge is

ignited to drive a piston forward. This piston rapidly compresses a light gas, in this case hydrogen, terminally concentrating into a conical section. In the second stage, the air is forced through a petal valve (diaphragm), causing a significant rise in pressure. The highly pressurized gas then expands through a nozzle, propelling a projectile through the launch tube at hypervelocity[10]. 2SLGGs, while indispensable in HVIT, are expensive to maintain and operate due to consumables such as propellant, sabot, piston, etc.

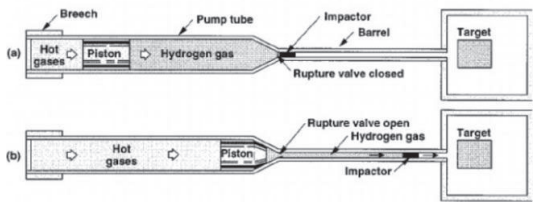


Fig. 2. Two-Stage Light Gas Gun[11].

Testing different combinations of inputs can take weeks of laboratory work. From experimental data, the duration of the trial is ~30ms, making minor misalignments, delays in detection a major issue. Thus, there is a need for a more robust and cost-effective solution. Traditionally, computationally heavy software with intensive physics is employed. These models, while reliable, fail at handling variation and lack adaptability[12]. Machine learning (ML) is a versatile tool that can be used to simulate laboratory-like conditions and provide a cost-effective estimate of the result.

V. ML MODEL OVERVIEW

Several ML models were implemented using the methods listed above. The accuracy and errors for each of the models were compared to find the superior model in this context.

A. Lasso Regression

Lasso regression is a type of linear regression that employs L1 regularisation, i.e., improving the model's generalisability. Lasso shrinks features that have a minimal contribution to prediction. This is useful to select relevant features when there are too many[13]. This model helped reduce overfitting. Grid Search was used to tune the ‘alpha’ parameter over the range of (-4,4,1000).

B. Ridge Regression

This model is a subset of linear regression as well, working on a similar penalizing concept as Lasso. Instead of shrinking irrelevant features, it adds a penalty to all coefficients of weight; this is called L2 regularisation. Larger coefficients or highly correlated features are penalised more heavily, to prevent the model from getting too complex. This model takes all features into account, which is extremely useful if the physics behind the mechanism isn't known well enough to know the interrelations[14]. This model helped reduce overfitting. Grid Search was used to tune the ‘alpha’ parameter over the range of (-4,3,100)

C. *ElasticNet*

ElasticNet is a combination of L1 and L2 regularisation, reaping benefits from both Lasso and Ridge models[15]. Lasso may be too aggressive with the exclusion of irrelevant features, and Ridge attempts to keep all instances in check. Parameters like ‘alpha’ were tuned over the range of(-4,3,50) while ‘l1_ratio’ was tuned over the range of (0,1,50).

D. Polynomial Regression

Polynomial regression enhances the linear assumption by introducing polynomial terms. This enables the model to handle non-linear relationships among the selected parameters. The introduction of curvature allows it to capture complex patterns between features. Consequently, with a small dataset such as in this test, non-linear models can face disruption due to overfitting and may fail[16]. It was noticed that polynomial regression gave the best result with degree = 2, i.e., 83% accuracy, and as this value was increased, the accuracy kept decreasing.

E. SVR

SVR is based on support vector mechanisms, which uses decision hyperplanes to define boundaries. Instead of minimising error between actual and predicted values like linear models, it searches for a function that fits the data best. With an error threshold, the focus is to minimise the error using this function used to define the hyperplane. It extends to higher dimensions using kernels, which map the relationship between input data[17]. It was configured with parameters like kernel = ‘rbf’, C = 10, epsilon = 0.1 and degree = 2. It demonstrated good generalisation and reduced overfitting.

F. Random Forest Regression

Random Forest is simply an ensemble of multiple decision trees. It creates multiple trees using bootstrap samples (subset samples taken at random with replacement) and at each node iteratively confirms the best split[18]. For a new input, it averages the predicted values of all the trees. This strategy of regularisation reduces the likelihood of overfitting.

G. XGBoost

XGBoost Extreme Gradient Boosting model was optimally set to enhance its predictive capability for projectile velocity. Some of the most important settings were applying objective='reg:gamma', which is a good setting for predicting always-positive values, such as velocity. It includes built-in mechanisms to prevent overfitting, such as regularisation, tree pruning, and learning rate control[19]. The learning rate was low at 0.01, so the model would learn very slowly and not overfit, and n_estimators=200 enabled the model to construct sufficient decision trees for effective learning. With these optimized parameters and other inherent features, XGBoost produced superior and consistent results.

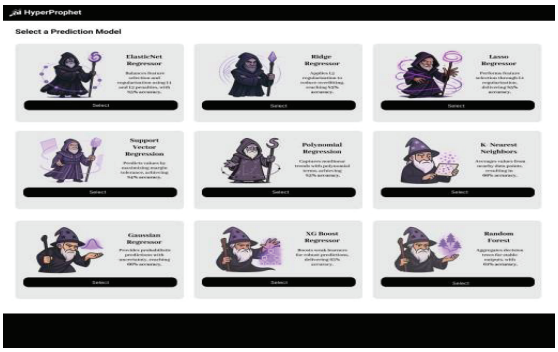
H. Gaussian Process Regression (GPR)

GPR is a nonparametric, kernel-based method that models a distribution over a viable function. It assumes a multivariate normal distribution by placing priors over functions and letting the data take shape[20]. To boost accuracy, hyperparameters were optimized. The kernel governed the complexity and smoothness of the model, while the alpha

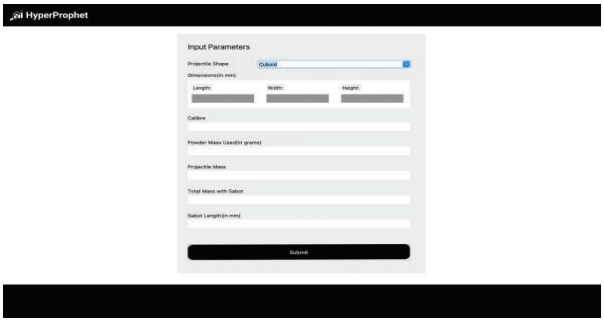
parameter managed the noise from the data. The n_restarts_optimizer attempted multiple optimization trials to circumvent cases where poor local minima could be reached. In conjunction with RFE, which only identified the top 17 features, these parameters provided the model with accurate predictions and the ability to generalize well.

VI. RESULTS AND DISCUSSION

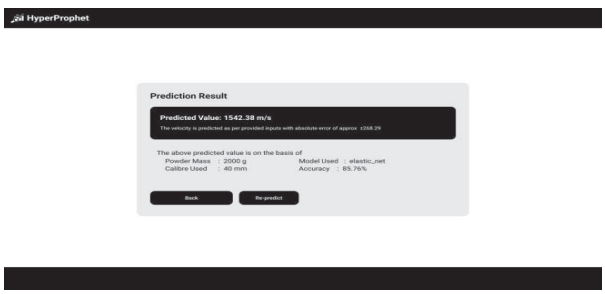
This section systematically evaluates the performance of various ML models based on the metrics mentioned in Section 3. The software is explored to demonstrate functionality and discuss its achievements. The following are the screenshots of the GUI for the discussed model.



3(a) Prediction result screen



3(b)Screen for input from user



3(c) Homepage with model selection Fig. 3 Snippets of HyperProphet software

The user is allowed to choose from nine different ML models on the homepage, as shown in fig. 3(a), depending on their specific requirements and objectives. User is redirected to the input form where they can insert required input parameters as shown in the fig. 3(b). In fig. 3(c), the prediction of projectile velocity is given, showcasing model's accuracy along with it in output format.

In Table II, the models with the highest performance after optimization are listed. The models are first implemented on the training set, followed by the test set. The following observations were made:

- Lasso Regression: The penalty applied to the weights of the features reduces issues caused due to overfitting or multicollinearity. The result showed a significant reduction in noise with an accuracy of 79.04% on the test set and 85.74% after optimization.
- Ridge Regression: Unlike Lasso, it takes correlated features into account, adding a nuanced layer, but performed with a minutely lesser accuracy of 77.29% which was later increased to 85.30% after optimization.
- ElasticNet: It finds a balance between shrinkage of irrelevant weights and handles multicollinearity. Model stability and accuracy were improved, with accuracy being upgraded from 66.01% to 85.7%.
- Polynomial Regression: Despite being simplistic, polynomial regression worked with a promising accuracy of 75.82% refined to 83.80% after optimization, considering nonlinearity in the model.
- SVR: Excellent for complex, nonlinear datasets with higher dimensions and showed a fair accuracy of 65.3% before optimization and 84.55% after.
- .Random Forest Regression: It works well on complex data with higher dimensions and nonlinearity and is reliable as compared to decision tree regression (DTR), while being slower. Post optimization accuracy was 61.24%, increased from an initial value of 39.06%.
- XGBoost: With strong learning prowess, it outperformed DTR and random forest with an accuracy of 46.63% on the test set before and 67.50% on the training set after optimization
- GPR: This approach is particularly desirable on a small data set exhibiting an accuracy of 66.88% optimised from 53.6%.

TABLE II. Model Performance Metrics

ElasticNet	269.011	144021.6	0.8576
Ridge	275.736	148606.1	0.853
Lasso	282.6	157206.5	0.8445
SVR	268.537	156192.7	0.8455
Polynomial	338.82	163776.2	0.838
Gaussian	409.64	334983.3	0.6688
XG Boost	410.637	318536.9	0.685
Random Forest	511.06	646285.4	0.6124

Table III. demonstrates a comparison of the experimental values and the predicted values for each model. The linear models showed a very small error difference in Cuboid and Cube, while they show a steep error during the prediction of the velocity of disc impact. Non-linear models, i.e., SVR, Gaussian Process Regressor, Random Forest regression, and XGBoost, showed the least deviation from the experimental values. Polynomial Regressor behaves similarly to the linear models, showing best results with cuboidal projectile, while deviating a little more in the case of disc and cube projectiles.

TABLE III. Comparison of predicted velocity for each model with relevant experimental values

Proj. Shape	Exp. Velocity	Predicted Velocity							
		Elastic Net	Ridge	Lasso	SVR	Poly	GPR	XGB	RF
Disk	5400	4807.02	4857.1	4931.3	5035.3	4855	5356.1	5047.05	5213.5
Cuboid	3000	3155.38	3060.8	3020.5	3116.5	3016.9	3004.1	3157.4	3056.3
Cube	4000	4024.11	4087.3	4166.5	4147.6	4399	4197.3	4003.6	4011.5

From Table IV, it is understood that random forest regression experienced the highest upgrade (+22.18%) in accuracy through RFE optimization. Optimisation methods and features chosen that showed a significant difference in accuracy are discussed.

TABLE IV. Optimization Summary and Parameters Used

Model	Key Tuned Aspects	Method Used	Accuracy Gain
Elastic Net	Alpha, L1 ratio	Grid Search	0.1975
Lasso Regression	Alpha	Grid Search	0.067
Ridge Regression	Alpha	Grid Search, RFE	0.0801
SVR	scaling, feature selection, Kernel, 'C', Epsilon, Gamma	Grid Search, RFE	0.2102
Random Forest	feature selection	RFE	0.2218
XGBoost	n_estimators, learning rate, objective	Grid Search	0.2187
GPR	Kernel, Alpha, n_restarts_optimiser	Grid Search, RFE	0.1322

We conclude that models based on linear regression or simpler models like polynomial regression outperformed more complex, nonlinear ones. This can be attributed to the fact that the small and noisy data is not ideal for deep learning models such as ANNs. Ensemble models like Random Forest and XGBoost showed the highest improvement in accuracy after optimisation, demonstrating the importance of the same.

VII. CONCLUSION

In summary, this study proposes a solid machine learning model for projectile velocity prediction for HVIT with a 2SLGG without resorting to deep learning methods. By exhaustive comparison of different regression models, ElasticNet and SVR were identified as the best-performing models, with the best trade-off between model complexity and generalization, especially when augmented with grid search and RFE. The relatively less optimal performance of decision trees and other models presents the difficulties with limited and noisy datasets, and the criticality of careful model selection and regularization. The incorporation of the best-performing models into the HyperProphet GUI tool illustrates the practical utility of this approach, with the ability of researchers and engineers to accurately estimate velocity. Although the framework does not replace traditional HVIT methods, it is a useful adjunct to pre-screening, design iteration, and parameter optimization. Future developments should include increasing the dataset, real-time diagnostics integration, and a hybrid approach with physics-based simulation and machine learning to further improve predictive accuracy and utility.

ACKNOWLEDGMENT

The authors would like to acknowledge Dr. M. Raghavendra Rao, Director of Terminal Ballistics Research Laboratory (TBRL), Ramgarh-Haryana, for his valuable guidance and constructive feedback, which significantly contributed to the quality of this work. His expertise and critical insights were instrumental in shaping the direction of the research.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest in this study.

REFERENCES

[1] Piekutowski AJ, Poormon KL. Meeting the challenges of hypervelocity impact testing at 10 km/s. Int J Impact Eng. 2023;180:104665. doi:10.1016/j.ijimpeng.2023.104665

[2] Pai A, Sharma A, Eby IM, Kini CR, Shenoy SB. A numerical approach for response of Whipple shields with coated and monolithic front bumper to hypervelocity impact by spherical projectiles. Acta Astronaut. 2023;202:433-441. doi:10.1016/j.actaastro.2022.10.041

[3] Murtaugh, Max & Rogers, Jacob & Allaire, Douglas & Lacy, Jr. (2024). Comparing Regression Methods for Two-Stage Light Gas Gun Muzzle Velocity Predictions. 10.31224/3433.

[4] Shojaei, P, Trabia, M, O'Toole, B, & Jennings, R. "Predicting the Projectile Velocity of a Two-Stage Gas Gun Using Machine Learning." Proceedings of the . Volume 3: Fluid-Structure Interaction; High Pressure Technology. Las Vegas, Nevada, USA. July 17–22, 2022. V003T05A014. ASME.

[5] Whipple FL. Shielding space vehicles against meteoroid damage. Proc Am Philos Soc. 1947;89(2):163-169.

[6] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys. 2019;378:686-707. doi:10.1016/j.jcp.2018.10.045

[7] Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc Series B. 1996;58(1):267-288. doi:10.1111/j.2517-6161.1996.tb02080.x

[8] Bergstra J, Bengio Y. Random search for hyper-parameter optimization. J Mach Learn Res. 2012;13:281-305. http://jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

[9] Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Mach Learn. 2002;46(1):389-422. doi:10.1023/A:1012487302797

[10] Suman G, Bhardwaj V, Kumar PD. Review of various techniques to achieve velocity unobtainable with two stage light gas gun. Adv Space Res. 2025;76(1):469-480. doi:10.1016/j.asr.2025.04.054

[11] Nellis W, Weir S, Mitchell AC. Minimum metallic conductivity of fluid hydrogen at 140 GPa (1.4 Mbar). Phys Rev B. 1999;59(5):3434-3437. doi:10.1103/PhysRevB.59.3434

[12] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys. 2019;378:686-707. doi:10.1016/j.jcp.2018.10.045

[13] Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc Series B. 1996;58(1):267-288. doi:10.1111/j.2517-6161.1996.tb02080.x

[14] Hoerl AE, Kennard RW. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics. 1970;12(1):55-67. doi:10.1080/00401706.1970.10488634

[15] Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc Series B. 2005;67(2):301-320. doi:10.1111/j.1467-9868.2005.00503.x

[16] 16. Cheng C, Wang C, Lee C. Polynomial regression as an alternative to neural nets. IEEE Access. 2018;6:30404-30412. doi:10.1109/ACCESS.2018.2841885

[17] Smola AJ, Schölkopf B. A tutorial on support vector regression. Stat Comput. 2004;14(3):199-222. doi:10.1023/B:STCO.0000035301.49549.88

[18] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016:785-794. doi:10.1145/2939672.2939785

[19] Breiman L. Random forests. Mach Learn. 2001;45(1):5-32. doi:10.1023/A:1010933404324

[20] Rasmussen CE, Williams CKI. Gaussian Processes for Machine Learning. MIT Press; 2006.