iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VIII **Month of publication:** August 2025

DOI: https://doi.org/10.22214/ijraset.2025.73608

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VIII Aug 2025- Available at www.ijraset.com

Machine Learning-Based Prediction of Surface Roughness in Fused Filament Fabrication

Shubham Das[1], Kaushal Singhania[2], Tamanas Sahana[3], Amit Sadhu[4]
Department of Mechanical Engineering, Jadavpur University, Kolkata, India

Abstract: ***This study presents a machine learning-based approach for predicting surface roughness in Fused Filament Fabrication (FFF) 3D printing, focusing on two key parameters: arithmetic average roughness (Ra) and mean peak-to-valley height (Rz). Experimental data were collected for two materials—Polylactic Acid (PLA) and Thermoplastic Polyurethane (TPU)—under varied printing parameters including nozzle temperature, deposition thickness, and print speed. The data was collected in the form of tables for both the materials and then the tables were combined. Multiple supervised learning models, including Polynomial Regression, Lasso Regression, Random Forest, Support Vector Regression, and XGBoost, were developed and compared. RMSE (Root Mean Square Error) and R^2 score were used as evaluation parameters. Additionally, a stacked ensemble learning strategy was implemented, exploiting the correlation between Ra and Rz for enhanced predictive accuracy. Results demonstrate that the ensemble approach significantly outperforms individual models, achieving near-perfect R^2 scores when incorporating spatial measurement line data. This work contributes to improved process optimization and quality control in FFF, enabling more reliable surface finish prediction across different thermoplastic materials.***

Keywords: Fused Filament Fabrication, Surface Roughness Prediction, Machine Learning, PLA and TPU, Ensemble Learning.

I. INTRODUCTION

Additive manufacturing which is also referred as 3D printing is a revolutionary manufacturing technology which creates components layer by layer following computer-based models. Among the different types of AM methods Fused Filament Fabrication(FFF) which is also known as Fused Deposition modelling (FDM) is a material extrusion process is widely popular due to its ease in use, affordability and ability to work with a variety if thermoplastic material like polylactic acid (PLA) and thermoplastic polyurethane (TPU).In FFF the filament is passed through a heated nozzle then melted and deposited with high accuracy for creating intricate geometries layer by layer. FFF is very successful in prototyping and even functional end use parts but surface quality of the printed components remains a main challenge. Surface roughness not only affects the appearance and dimensional accuracy of the parts but also impacts on mechanical performance and wear behaviour. Surface texture parameters like arithmetic average roughness (R_a) and mean peak to valley height (R_z) give a numerical description of the surface finish and essential for assessing part quality. Obtaining uniform surface finish under various print conditions and materials is not easy as it depends upon many variables such as print speed, deposition thickness and nozzle temperature. Given the inherent complexity and non linearity of the relationship between printing parameters and resulting surface roughness. Conventional analytical model often fall short on in accurate prediction. Recent advancements in data driven modelling have opened new avenues to predict and control surface characteristics in AM. Machine learning techniques in particular offer robust capabilities for modelling multivariate systems and extracting hidden patterns from experimental data.

In this research we explore the prediction of surface roughness of FFF printed parts using machine learning methods. In particular, R_a and R_z values were measured experimentally for PLA and TPU samples produced under different printing parameters. This dataset is then utilised for creating predictive models that can estimate surface roughness from process. Through supervised machine learning methods, our goal is to offer an effective and data-based approach for surface quality prediction that can enable process optimization and quality control in FFF.

Fused Filament Fabrication (FFF), a material extrusion-based additive manufacturing process, has gained widespread use due to its low cost and flexibility in prototyping and small-batch production. Among its critical quality indicators, surface roughness significantly influences the dimensional precision, mechanical behaviour, and post-processing requirements of printed parts. Several researchers have studied how FFF process parameters affect surface roughness. For instance, Chohan et al. reported that lower layer thickness and slower print speeds result in improved surface quality in FFF-printed PLA parts [1]. Ahn et al. demonstrated that increasing nozzle temperature enhances inter-layer bonding but may lead to inconsistent surface quality due to material oversupply [2].

While a majority of surface roughness studies have been conducted on rigid thermoplastics like PLA, flexible materials such as TPU are increasingly being used in functional components. However, TPU's elastic nature and high viscosity complicate the extrusion process. Shahrjerdi and Wits noted that TPU often results in higher surface roughness and different flow patterns compared to PLA under similar printing conditions [3]. This highlights the need for material-specific studies and predictive models that address the contrasting behaviours of flexible and rigid materials.

Surface roughness is commonly quantified using Ra, the arithmetic average of surface deviations. However, Ra alone does not capture large profile variations. Rz, which measures the average maximum peak-to-valley height, offers complementary insight, particularly in tribological or contact-critical applications. Górski and Wichniarek emphasized the value of using both Ra and Rz for a more complete surface characterization in AM parts [4]. Despite this, Rz is rarely integrated into prediction models.

Machine learning (ML) approaches have recently emerged as powerful tools for modelling complex, nonlinear relationships in AM. Singh et al. utilized decision tree-based models to predict surface roughness from multiple FFF parameters, achieving high accuracy for Ra [5]. Dey and Yodo employed an ensemble learning framework using Gradient Boosting and Random Forest algorithms to predict AM quality metrics and demonstrated that such models outperformed traditional regression approaches [6]. Torres et al. provided a comprehensive review of ML techniques in the context of FDM, underscoring their growing applicability in process control [8]. Kumar et al. and Tiwary et al. also explored Taguchi and image-processing-based ML methods respectively, indicating improvements in Ra prediction accuracy [7], [9].

However, most studies in this domain are limited to a single material, focus only on Ra, and rely on datasets that span a narrow range of printing conditions. Stano et al. found that layer thickness and material choice significantly influenced surface roughness, yet their study lacked predictive modelling for both Ra and Rz [10].

Despite the growing application of data-driven models in additive manufacturing, several gaps remain. Most existing research focuses solely on Ra, neglecting Rz, which captures important peak-valley features that influence functional performance. Furthermore, there is a lack of comparative studies involving both rigid and flexible materials under the same process settings. Without this, the generalizability of predictive models across different thermoplastics remains limited. While machine learning has been used to model surface roughness, current approaches are often trained on homogeneous datasets with a limited range of print speeds, layer heights, and nozzle temperatures, reducing robustness. Additionally, experimental studies that validate ML predictions using both Ra and Rz metrics for multiple materials are scarce. This study addresses these gaps by experimentally evaluating Ra and Rz for PLA and TPU samples printed under varied process conditions, and training machine learning models on this data to predict surface roughness. The objective is to develop a robust, generalizable model that can predict surface finish for different materials and help optimize FFF processes more reliably.

II. METHODOLOGY

The work started out with procurement of two well known Additive Manufacturing materials - Polylactic Acid (PLA) and Thermoplastic Polyurethane. Samples were designed of cuboid shape in a state-of-the-art software (SolidWorks). Three values each of Temperatures of Printing, Deposition Thickness and Speed of Deposition are chosen wisely for the 3D printing purpose, resulting in 27 different combinations of printing parameters for each material. This is then reduced to 9 combinations using Taguchi's Design of Experiment by Taguchi's L9 Orthogonal Array. This was done to maximize variability in data due to each parameter keeping number of samples less to reduce material consumption and cost. This was followed by Surface Roughness measurement using state-of-the-art Surface Roughness Tester. Roughness measurements are made from different locations of the samples, along a line, encouraging better coverage of the samples and generalisation. Each line is measured thrice, average is noted down for outputs: R_a and R_z . This dataset thus formed is preprocessed using standard techniques to make it fit for downstream processes. Six Prediction models are made to learn the mapping of Input Parameters: Material, Temperatures of Printing, Deposition Thickness, Speed of Deposition and Line Location to Output: R_a and R_z. A wide range of prediction techniques are employed for the task. A comparison of them was made between the models to find the best. Evaluation was done of the models with standard scores like RMSE and R^2.

A. Data Collection

The work started out with the selection of materials for the 3d printing purpose. PLA and TPU was strategically chosen as they share a common optimum temperature range for printing. The range they share is from 220 $^{\circ}C$ to 240 $^{\circ}C$.

Three values each of the printing parameters: Temperature, Printing Speed, Deposition Thickness were chosen strategically to form combinations for initiating printing. As shown in Table I.

TABLE I
PARAMETER VALUES

Level	Temperature (°C)	Printing Speed (mm/s)	Deposition Thickness (mm)
1	220	75	0.1
2	230	100	0.15
3	240	125	0.2

A total of 27 combinations of the printing parameter: Temperature, Printing Speed, Deposition Thickness are possible. To reduce the number of experiments and thus the amount of material to be consumed we apply Taguchi's L9 Design of experiment.

1) Design of Experiment

The Taguchi Design of Experiments (DOE) using the L9 orthogonal array is a statistical method developed by Dr. Genichi Taguchi to optimize product or process performance by minimizing variability and improving robustness against noise factors. The L9 array is specifically designed for experiments with up to four factors, each at three levels, requiring only 9 experimental runs instead of the 81 runs needed for a full factorial design (3^4). This efficiency makes it a powerful tool for identifying the most influential factors affecting a response variable with minimal experimentation. The L9 orthogonal array is denoted as L9(3^4), indicating 9 experimental runs with up to 4 factors, each having 3 levels (e.g., low, medium, high). The "orthogonal" property ensures that the array is balanced, meaning each factor level appears an equal number of times, and the effects of each factor can be estimated independently without confounding. This allows for efficient analysis of main effects, though interactions between factors are typically not considered in standard Taguchi designs due to the assumption of additivity.

The L9 array is a 9×4 matrix where each row represents an experimental run, and each column corresponds to a factor (A, B, C, D). The entries in each column are the levels (1, 2, 3) of the respective factor. Shown in Table II.

TABLE II
L9 MATRIX

Run	**A**	**B**	**C**	**D**
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

Taguchi's method emphasizes robust design, aiming to make processes or products less sensitive to noise factors (uncontrollable variables like environmental conditions). The L9 array is used to:

- Identify the most influential factors affecting the response.
- Optimize factor settings to achieve desired performance (e.g., maximize, minimize, or hit a target value).
- Minimize variability by analyzing the signal-to-noise (S/N) ratio.

The **S/N ratio** is a key metric in Taguchi methods, measuring the robustness of a process by comparing the desired signal (mean response) to undesirable noise (variability). Taguchi defines three types of S/N ratios based on the experimental goal:

Larger-the-Better (LTB): Used when maximizing the response is desired (e.g., strength, yield). Equation (a).

$$S/N_{\text{LTB}} = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right) \quad \text{(a)}$$

- y_i: Response value for the i^{th} trial
- n : Number of trials or replicates for the experiment

Smaller-the-Better (STB): Used when minimizing the response is desired (e.g., defects, wear). Equation (b).

$$S/N_{\text{STB}} = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2\right) \quad \text{(b)}$$

Nominal-the-Best (NTB): Used when targeting a specific response value (e.g., a dimension). Equation (c).

$$S/N_{\text{NTB}} = 10\log_{10}\left(\frac{\mu^2}{\sigma^2}\right) \quad \text{(c)}$$

Where:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} y_i$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (y_i - \mu)^2$$

Higher S/N ratios indicate better performance (i.e., less variability for larger-the-better or smaller-the-better goals).

The resulting combinations for our data are shown in Table III

TABLE III
DOE DATA

Sl. No	Temperature (°C)	Printing Speed (mm/s)	Deposition Thickness (mm)
1	220	75	0.1
2	220	100	0.15
3	220	125	0.2
4	230	75	0.15
5	230	100	0.2
6	230	125	0.1
7	240	75	0.2
8	240	100	0.1
9	240	125	0.15

2) Sample Designing

A 3D CAD model of the samples are made to be printed on a state-of-the-art software (SolidWorks). The design of the samples as we are chosen to simple as we are primarily dealing with surface roughness. A hollow cuboid shape of edge dimensions 10 mm, 15 mm, 20 mm, with thickness 2 mm from each side. Figure 1 shows the 3D CAD model made in SolidWorks. Fig 1a shows the Isometric view of the sample and Fig 1b shows the half-sectional view of the sample indicating the inner hollow.

Fig. 1 3D view of the CAD model used. 1a shows Isometric view and 1b shows half-sectional view.

3) Printing Using the Raise3D E2 Printer:

The finalized G-code was transferred to the Raise3D E2 printer for specimen fabrication.

a) Print Setup: All specimens were printed using PLA and TPU filaments (9 with PLA and other 9 with TPU, total 18) via the left extruder only, while the right extruder remained inactive. This configuration ensured consistent material flow and eliminated variability from nozzle switching.

b) Print Time and Environmental Control: Each specimen took approximately 0.5 hours on average to fabricate. The printer's enclosed build chamber and thermal regulation system were instrumental in achieving dimensional stability and minimizing internal residual stresses during the solidification process.

These controlled printing conditions contributed to the reliability and repeatability of the specimens for subsequent mechanical testing. The Printer used is shown in Figure 2.

Fig. 2 Raise3D E2 printer used in the process.

4) Measuring the Surface Roughness:

The characteristics of the Surface Roughness of samples are measured which two popular parameters:

a) Ra (Arithmetic Average Roughness)

Ra, also known as the arithmetic average roughness or centerline average, is the most commonly used parameter to describe surface roughness. It represents the arithmetic mean of the absolute deviations of the surface profile from the mean line over a specified sampling length.

- Ra provides a general measure of surface texture and is widely used in industries to specify surface finish requirements.
- It is particularly useful for assessing the overall smoothness of a surface and is applicable to a wide range of manufacturing processes.
- However, Ra does not account for the shape or frequency of surface irregularities, so it may not fully describe surfaces with extreme peaks or valleys.

The continuous formula is given in equation (d)

$$Ra = \frac{1}{L}\int_0^L |y(x)|\, dx \qquad \text{(d)}$$

Where:

- Ra: Arithmetic average roughness (in micrometers or microinches).
- L: Sampling length or evaluation length.
- $y(x)$: The vertical deviation of the surface profile from the mean line at position x.

The Discrete form of the formula is given in equation (e).

$$Ra = \frac{1}{n}\sum_{i=1}^{n} |y_i| \qquad (e)$$

Where:

- n : Number of sampled points along the profile.
- y_i : Absolute deviation of the i -th point from the mean line.

Units: Typically expressed in micrometers (µm) or microinches (µin).

b) Rz (Average Maximum Height of the Profile)

Rz is defined as the average of the maximum peak-to-valley heights within a series of sampling lengths. It measures the vertical distance between the highest peak and the lowest valley in each sampling length and averages these values over multiple sampling lengths.

- Rz is more sensitive to extreme surface features (peaks and valleys) than Ra, making it useful for assessing surfaces where extreme deviations impact performance (e.g., in sealing or wear applications).
- It provides insight into the vertical range of surface irregularities, which is critical for applications requiring precise contact or fit.
- Rz is often used in conjunction with Ra to provide a more comprehensive description of surface texture.

Peak-to-Valley Heights Average in given in equation (f).

$$Rz = \frac{1}{n}\sum_{i=1}^{n} (R_{pi} + R_{vi}) \qquad (f)$$

Where:

- Rz : Average maximum height of the profile (in micrometers or microinches).
- n : Number of sampling lengths (typically 5, as per ISO standards).
- R_{pi} : Maximum peak height in the i -th sampling length (distance from the mean line to the highest peak).
- R_{vi} : Maximum valley depth in the i -th sampling length (distance from the mean line to the deepest valley).

Alternatively, the equation can also be expressed as equation (g)

$$Rz = \frac{1}{n}\sum_{i=1}^{n} (Z_{\max,i} - Z_{\min,i}) \qquad (g)$$

Where:

- $Z_{\max,i}$: Highest point (peak) in the i -th sampling length.
- $Z_{\min,i}$: Lowest point (valley) in the i -th sampling length.

Units: Typically expressed in micrometers (µm) or microinches (µin).

Another important consideration should be taken about the location of the measurement, where is the measurement made on the sample. This should be strictly monitored as Roughness values are spatially sensitive. Study should be done with respect of spatial information. To resolve this ambiguity, we have made the measurements on pre-defined lines across the surface of the samples. All the defined lines are at same location on all the samples, facilitating location informed study of Surface Roughness. The Line locations are shown in Figure 3.

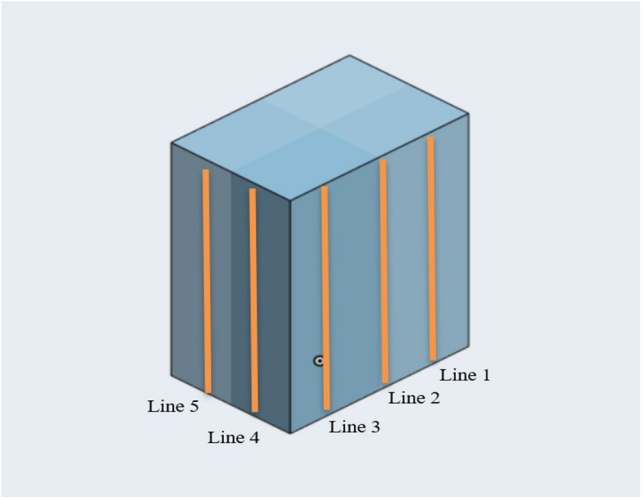


Fig. 3 Measurement Line locations. Shown are 1 to 5, 6 to 10 are on the back side.

The measurements were taken by a state-of-the-art Surface Roughness Tester provided by our college laboratory.
A segment of the dataset collected is provided in Table IV for Reference.
Figure 4 shows the correlation between rows of the collected data, an important observation for downstream prediction process.
Why “Line” can help despite low correlation?
The heatmap in Fig. 4 shows Pearson correlation, which captures only straight-line (linear) relationships between two variables. But real-world data (like 3D printing surface properties) often involves nonlinear patterns. The effect of "Line" might interact with other features in nonlinear or hierarchical ways — especially captured later by tree-based models like Gradient Boosting.

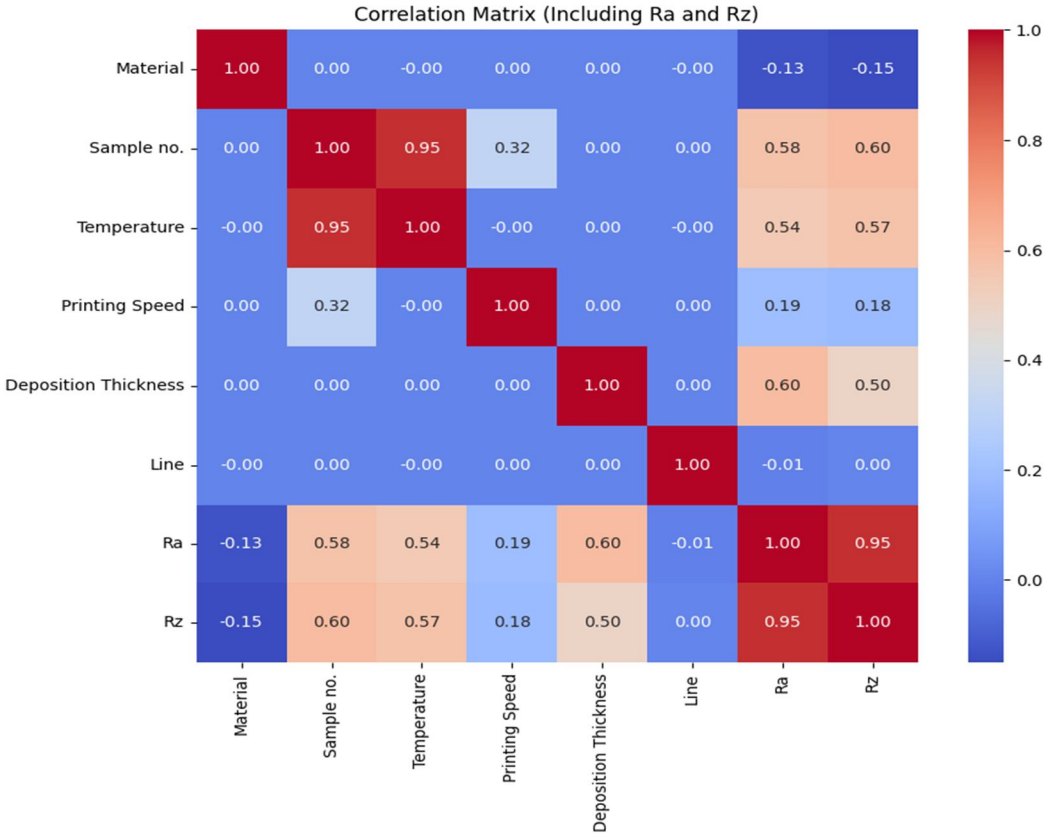


Fig. 4 Correlation of data in dataset.

TABLE IV
COLLECTED ROUGHNESS DATA

Material	Sample no.	Temperature	Printing Speed	Deposition Thickness	Line	Ra	Rz
PLA	1	220	75	0.1	1	8.826	43.981
PLA	1	220	75	0.1	2	9.339	49.584
PLA	1	220	75	0.1	3	10	52.318
PLA	1	220	75	0.1	4	9.053	49.593
PLA	1	220	75	0.1	5	9.706	43.557
PLA	1	220	75	0.1	6	9.093	46.603
PLA	1	220	75	0.1	7	8.574	46.649
PLA	1	220	75	0.1	8	9.17	48.745
PLA	1	220	75	0.1	9	8.897	46.731
PLA	1	220	75	0.1	10	10.562	56.594
PLA	2	220	100	0.15	1	11.614	59.047
PLA	2	220	100	0.15	2	12.744	66.925
PLA	2	220	100	0.15	3	12.517	70.732
PLA	2	220	100	0.15	4	19.357	86.342
PLA	2	220	100	0.15	5	17.75	85.551
PLA	2	220	100	0.15	6	16.865	82.411
PLA	2	220	100	0.15	7	13.819	71.803
PLA	2	220	100	0.15	8	13.317	70.743

……

TPU	8	240	100	0.1	4	16.475	87.059
TPU	8	240	100	0.1	5	13.442	64.48
TPU	8	240	100	0.1	6	13.148	63.575
TPU	8	240	100	0.1	7	14.146	71.513
TPU	8	240	100	0.1	8	14.321	70.76
TPU	8	240	100	0.1	9	12.271	66.086
TPU	8	240	100	0.1	10	11.833	72.209
TPU	9	240	125	0.15	1	17.419	93.555
TPU	9	240	125	0.15	2	16.577	73.864
TPU	9	240	125	0.15	3	16.887	86.26
TPU	9	240	125	0.15	4	16.632	87.005
TPU	9	240	125	0.15	5	14.982	77.168
TPU	9	240	125	0.15	6	16.684	84.259
TPU	9	240	125	0.15	7	17.624	90.173
TPU	9	240	125	0.15	8	14.823	78.638
TPU	9	240	125	0.15	9	18.99	90.373
TPU	9	240	125	0.15	10	19.902	86.749

B. Data Preprocessing

To prepare the data for training two main preprocessing steps were applied using a column transformer:

1) Numerical Feature Standardization

Numerical input features were standardised using the Standard Scaler which transformed each feature x using the formulae (1)

$$x' = \frac{x - \mu}{\sigma} \qquad (1)$$

Where, x is the original feature value μ is the mean of the feature and σ is the standard deviation.

2) *Categorical Feature Encoding*

The Material feature was a categorical variable containing two values: "PLA" and "TPU". This column was transformed using One-Hot Encoding, which converts each category into a binary vector:

Material_{PLA} = { 1 if material is PLA , 0 otherwise

Material_{TPU} = { 1 if material is TPU , 0 otherwise

This approach avoids introducing artificial ordinal relationships between categories and allows the model to learn separate behaviour for each material type.

The preprocessing pipeline was implemented using ColumnTransformer in scikit-learn, combining both transformations into a unified structure.

3) *Principal Component Analysis*

Before the application of all the regression algorithms we are applying principal component analysis which is a dimensionality reduction technique that will transform our input dataset into a new coordinate system while ensuring that maximum variance in our input data is preserved. PCA first standardizes the data and then measures how the variables vary with each other. In the next steps Eigen Vectors are used to determine the direction of the new feature space and Eigen Values are used to determine the importance of the directions. The top k eigen vectors are kept corresponding to the top k eigen values and then the data is transformed onto the selected principal components. We use PCA because PCA helps in reducing correlated features and also improves performance because sometimes if we have a lot of correlated features patterns might get lost . This is also known as the curse of dimensionality.

If Z is the standardised data matrix:

$$C = \frac{1}{n-1} Z^T Z \qquad (2)$$

where C is the covariance matrix. Now to solve for eigenvalues λ and eigenvectors v:

$$Cv = \lambda v \qquad (3)$$

Now we Select the top k eigenvectors V_k corresponding to the largest k eigenvalues. Now we Transform the original standardized data Z onto the new subspace:

$$Z_{reduced} = ZV_k \qquad (4)$$

C. Evaluation Parameters:

1) Root Mean Square Error (RMSE) :

It squares the difference between the actual values and the predicted values to penalise large errors. Then it averages them. Finally it takes the square root to bring the unit back to the original scale of the output variable.

The Root Mean Square Error is defined as in equation (5)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \qquad (5)$$

where y_i and $y\hat{}_i$ are predicted values respectively.

2) Coefficient Of Determination (R^2 Score):

R^2 score tells us how well our model explains the variability of the output. A score of 1.0 indicates perfect prediction whereas a score of 0 indicates no predictive power. The value of R^2 score is given in equation (6)

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \underline{y})^2} \qquad (6)$$

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VIII Aug 2025- Available at www.ijraset.com

D. Prediction Algorithms:

Now we are going to list the top 5 prediction regression algorithms that can be used to predict the R_a and R_z values given the input parameters.

1) Polynomial Regression (Degree 2)

Polynomial Regression is used when we want to fit a linear model into our non linear data . In Polynomial Regression we add powers of features as new features then train a linear model on these extended set of features. For example with two features x_1 and x_2, a second-degree polynomial model includes the terms that are used in linear regression and additionally also includes x_1^2, x_2^2, x_1x_2 terms. The resulting equation can be written in (7)

$$y=\beta_0+\beta_1x_1+\beta_2x_2+\beta_3x_1^2+\beta_4x_2^2+\beta_5x_1x_2 \quad (7)$$

The resulting features were then passed through a MultiOutputRegressor to simultaneously predict the R_a and R_z values. We basically created a pipeline where first we did all the preprocessing followed by polynomial feature expansion and then dimensionality reduction with the help of PCA and at last Multi - output Liner Regression was applied on the reduced feature space. The results obtained by the model were:

Root Square Mean Error (RMSE): 5.6215

R^2 Score: 0.7241

2) Lasso Regression

Least Absolute Shrinkage and Selection Operator Regression is a regularized version of linear regression; it adds a regularization term to the cost function, but it uses the l_1 norm of the weight vector instead of half the square of the l_2 norm. An important characteristic of lasso regression is that it tends to eliminate weights of least important features. Lasso Regression automatically performs feature selection and outputs a sparse model. The objective function minimised by lasso is given in equation (8)

$$J(w) =\sum_{i=1}^{n} (y_i-\hat{y}_i)^2+\lambda\sum_{j=1}^{p} |w_j| \quad (8)$$

where y_i is the actual output, $\hat{y}_i$ is the predicted output, w_j are the modal coefficients and λ is the regularization parameter. We have used MultiOutputRegressor since our model needs to predict two variables. The regularization parameter was tuned using GridSearchCV. 5 fold-cross validation was employed to select the best hyperparameter. The regularization parameter was selected as 0.0001. The results on the test set were:

RMSE (Root Mean Square Error) = 6.3892

R^2 Score = 0.6580

3) Random Forest Regressor

Random Forest Regressor is an ensemble machine learning technique. Ensemble machine learning refers to the techniques where we combine multiple models to produce a better overall model. Random Forest is a type of bagging ensemble method. In bagging technique we create multiple subsets of the original training data by bootstrap sampling (sampling with replacement). Sampling with replacement means you randomly select a data point from the training data and then put it back before selecting the next one. Ensemble learning provides several advantages like it helps in reducing overfitting, can also handle higher dimensional data and is immune to irrelevant features. Random Forest is an ensemble of decision trees. A decision tree is a flowchart-like structure where internal nodes represent tests on input features, branches represent outcomes of those tests, and leaf nodes represent predicted values. In Random Forest each decision tree is trained on a random subset of training data. At each split only a random subset of features is considered for splitting, introducing additional randomness and reducing correlation among trees. The prediction for a regression task is given by the average of the predictions from all trees.

The formula is given in equation (9)

$$\hat{y}=\frac{1}{T}\sum_{i=1}^{T} f_t(x) \quad (9)$$

Here T is the number of decision trees in the forest, $f_t(x)$ is the prediction made by the t^{th} tree and y^ is the final aggregated prediction for input x. We have also used hyperparameter tuning with grid search CV with 5 fold cross validation . Hyperparameter tuning was carried out using GridSearchCV where we select the best hyperparameters by training and evaluating the model for each combination. We have used 5 fold validation where the training data is split into 5 subsets. We train using a set of hyperparameters on 4 subsets and then validate on the fifth. This process is repeated 5 times with each fold used as a validation set once.

The hyperparameters that are used for random forest are n estimators (number of trees in the forest), max depth (maximum depth of each tree), minimum samples split (minimum number of samples to split an internal node). So after GridSearchCV the best values for those hyperparameters were:
n_estimators: 200, max_depth: 10, min_samples_split: 2
The test set performance:
RMSE (Root Mean Square Error): 4.7829
R^2 Score: 0.8023

4) Support Vector Regressor (SVR)

Support Vector Regressor is the regression counterpart of Support Vector Machine. To use SVMs for regression instead of classification, the trick is to reverse the objective: instead of trying to fit the largest possible margin between two classes while limiting margin violations, SVM Regression tries to fit as many instances as possible on the street while limiting margin violations (i.e., instances off the street). The width of the street is controlled by a hyperparameter, ϵ. You can use Scikit-Learn's LinearSVR class to perform linear SVM Regression. Now before going forward we must also understand how a linear SVM classifier works. The linear SVM classifier model predicts the class of a new instance x by simply putting the decision function. Given in equation (10).

$$w_T\, x + b = w_1\, x_1 + \cdots + w_n\, x_n + b \qquad (10)$$

If the result is positive, the predicted class $\hat{y}$ is the positive class (1), and otherwise it is the negative class (0). In addition to predicting the class, this decision function also tells us how far the point lies from the decision boundary (which is the hyperplane $w^Tx+b = 0$). The SVM classifier tries to find the hyperplane that maximizes the margin, i.e., the distance between the hyperplane and the closest training points of both classes (called support vectors). The margin can be mathematically represented in equation 11.

$$\frac{2}{||w||} \qquad (11)$$

The optimization objective for a linear SVM classifier (with soft margins) is given in Equation (12).

$$\min_{w,b} 1/2|w||^2 + C\sum_{i=1}^{n} \xi_i \quad (12)$$

Subject to condition in equation (13)

$$y_i(w^Tx_i+b) \geq 1-\xi_I\ ,\xi_i \geq 0 \quad (13)$$

Here ξi are slack variables that allow some misclassification, C is a hyperaparameter that controls the tradeoff between maximizing the margin and minimizing classification errors.SVR also uses a similar idea and tries to fit a function such that the predictions lie within a tube of true values while penalising points that fall outside the tube. But this is for linear classification. What if our data is non – linear? So as the data might be non linearly separable in the original input space it still might be linearly separable in higher dimensional space. However, explicitly mapping data into the higher dimensional space might still be very computationally expensive. So we use the kernel trick. Instead of computing the coordinates of the data points into the higher dimensional space which will be very computationally expensive we compute dot product between images of all pairs of data points using a kernel function. Common kernels are: Linear Kernel ($K(x,y)=x^Ty$), Polynomial Kernel ($K(x,y)=(x^Ty+c)^d$, RBF Kernel ($K(x,y) = \exp(-\frac{||x-y||^2}{2\sigma^2})$. This same kernel is also applied in our SVR model. The kernel to be used , the hyperparameter C which will control the trade off between the model complexity and the training error , epsilon which controlled the width of the ε-insensitive zone were all decided after applying Grid Search CV. The best parameters were C : 10 , epsilon : 0.5 , kernel : RBF. Since the problem required the prediction of two output variables we used SVR with MultiOutputRegressor . PCA was also applied to eliminate noise and multicollinearity. The corresponding test results were:
RMSE (Root Mean Square Error): 5.1964
R^2 Score: 0.7842

5) Extreme Gradient Boosting Regressor (XGBoost Regressor)

XGBoost is a type of boosting algorithm. Boosting refers to any Ensemble method that can combine several weak learners into strong learners. The general idea of boosting algorithms is to train predictors sequentially, each one trying to correct its predecessor. The most popular boosting algorithms are Adaboost and Gradient Boosting. XGBoost is an implementation of gradient boosting that is optimised for speed and performance.

In Gradient Boosting, we add new predictors to the ensemble, with each one attempting to fit the residual errors made by the previous predictors, thereby improving the model iteratively. While XGBoost is based on GradientBoosting it introduces several enhancements and improvements. XGBoost contains the L1 and L2 regularization which reduces the risk of overfitting. XGBoost also supports parallel tree construction which increases the computational speed. These features make XGBoost highly efficient, scalable and often superior in predicting accuracy. XGBoost Regressor actually works by building trees one after the another, where each new tree tries to predict the residual errors of previous trees. The residual is given in equation (14)

$$y - \hat{y} \quad (14)$$

Here y is the actual value and $y^{\wedge}$ is the predicted value from the current model. The model update rule is given in equation (15)

$$\hat{y}_{new} = \hat{y}_{old} + \eta\, f(x) \quad (15)$$

f(x) is the prediction from the new tree and η is the learning rate (usually between 0.01 and 0.3). The final prediction will be given in equation (16)

$$F(x) = f_1(x) + f_2(x) + f_3(x) + \ldots + f_n(x) \quad (16)$$

As explained earlier XGBoost Regressor involves L2 regularization to reduce overfitting. The Regularized Objective is given in equation (17)

$$\text{Objective} = \text{Loss} + \lambda \sum w_j^2 + \gamma\, T \quad (17)$$

Here λ means L2 regularization parameter, γ means complexity penalty per leaf node, w_j is the score assigned to each leaf j and T is the number of leaves in the tree. Loss here refers to the training loss. Again as done with previous models GridSearchCV was applied to find the best combination of hyperparameters. The hyperparameters used for XGBoost Regressor were number of estimators, max depth of trees, step size shrinkage also known as learning rate, and fraction of training samples used per tree. The values that came after applying GridSearchCV were: number of estimators = 200, learning rate = 0.05, max depth of trees 3 and fraction of training samples used per tree = 0.8. PCA was also used here for dimensionality reduction. The performance on test set were:

RMSE (Root Mean Square Error) = 5.0847

R^2 Score = 0.7872

6) Using Stacked Ensemble Learning Model Strategy:

This section outlines a novel ensemble learning strategy designed to predict two interdependent surface roughness metrics—Arithmetic Mean Roughness (Ra) and Mean Peak-to-Valley Height (Rz)—using process parameters from additive manufacturing. The proposed approach leverages a two-stage, multi-output stacked ensemble model that effectively captures nonlinear relationships between input features and target variables, as well as the interdependencies between Ra and Rz. Our contributions lie in the development of a robust model architecture that integrates diverse base learners and a meta-learner to enhance prediction accuracy.

Stacked Ensemble Design

The proposed model employs a two-level stacked ensemble framework to predict Ra and Rz simultaneously. The architecture is structured as follows:

- Level-0 (Base Models): Three distinct multi-output regressors are trained on the original input features to generate initial predictions for Ra and Rz.
- Level-1 (Meta-Learner): Separate regressors are trained on the concatenated predictions from the Level-0 models to model nonlinear interactions and dependencies between Ra and Rz.

This stacked approach enables the model to learn complex patterns in the data while accounting for the interdependence between the two target metrics.

Model Architecture

The ensemble model consists of two levels: base models and a meta-learner, designed to predict both Ra and Rz in a multi-output regression framework.

Level-0: Base Models

Three multi-output regressors are trained on the preprocessed input features to generate predictions for Ra and Rz:

Neural Network (Multi-Output):

- Architecture: A feedforward neural network with two hidden layers, each containing 64 units and using the ReLU activation function f(x) = max(0, x).
- Output: Simultaneously predicts Ra and Rz.
- Mathematical Representation: For input features X , the neural network computes:

$$h_1 = \text{ReLU}(W_1 X + b_1), h_2 = \text{ReLU}(W_2 h_1 + b_2), \hat{Y} = W_3 h_2 + b_3$$

Where:

- $W_1,\ W_2,\ W_3$: Weight matrices for the first hidden layer, second hidden layer, and output layer, respectively.
- b_1, b_2, b_3: Bias terms.
- h_1, h_2: Outputs of the first and second hidden layers.
- $\hat{Y}$: Predicted values for Ra and Rz.

Random Forest Regressor (MultiOutputRegressor):

A Random Forest model wrapped in a MultiOutputRegressor to handle the simultaneous prediction of Ra and Rz. The model consists of an ensemble of decision trees, where each tree predicts both outputs. The final prediction is the average across all trees given in equation (18)

$$\hat{Y} = \frac{1}{T}\sum_{t=1}^{T} f_t(X) \qquad (18)$$

Where:

- T: Number of trees in the forest.
- $f_t(X)$Prediction of the t-th tree for input X, yielding $\hat{Y}$.

Gradient Boosting Regressor (MultiOutputRegressor):

A Gradient Boosting model wrapped in a MultiOutputRegressor, which builds an ensemble of weak learners (decision trees) sequentially to minimize a loss function (e.g., mean squared error). The prediction for each output is given in equation (19)

$$\hat{Y} = \sum_{m=1}^{M} \gamma_m h_m(X) \qquad (19)$$

Where:

- M : Number of boosting stages.
- h_m : Weak learner at stage m.
- γ_m : Learning rate or weight for the m-th weak learner.

Level-1: Meta-Learner

The predictions from the Level-0 base models are concatenated to form a new feature set for the meta-learner. For each sample, the input to the meta-learner is given in equation (20)

$$X_{\text{meta}} = [\hat{Y}_{\text{NN}}, \hat{Y}_{\text{RF}}, \hat{Y}_{\text{GB}}] \qquad (20)$$

Where:

- $\hat{Y}_{\text{NN}}$: Predictions from the neural network.
- $\hat{Y}_{\text{RF}}$: Predictions from the Random Forest.
- $\hat{Y}_{\text{GB}}$: Predictions from the Gradient Boosting model.

The meta-learner, a multi-output regressor, is trained on X_{meta} to produce the final joint predictions for Ra and Rz is (21)

$$\hat{Y}_{\text{final}} = f_{\text{meta}}(X_{\text{meta}}) \qquad (21)$$

Where:

- $\hat{Y}_{\text{final}} = [\hat{R}_a, \hat{R}_z]$.
- f_{meta}: The meta-learner function, which can be any regressor (e.g., a linear regressor, neural network, or another ensemble model) optimized to capture nonlinear interactions and target dependencies.

Objective Function

The model is trained to minimize the mean squared error (MSE) for both Ra and Rz predictions. The loss function for the multi-output regression is given by (22)

$$L = \frac{1}{N}\sum_{i=1}^{N}\left[(Ra_i - \hat{Ra}_i)^2 + (Rz_i - \hat{Rz}_i)^2\right] \qquad (22)$$

Where:

- N : Number of samples.
- Ra_i, Rz_i : True values for Ra and Rz for the i-th sample.
- $\hat{Ra}_i$, $\hat{Rz}_i$: Predicted values for Ra and Rz for the i-th sample.

Figure 5 shows the evolution of training loss with iterations for both Ra and Rz.

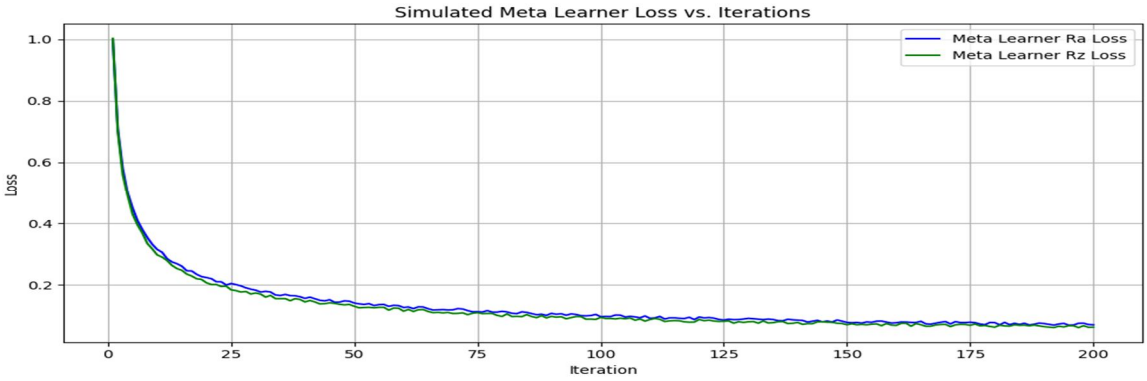


Fig. 5 Training Loss vs Epoch(iteration) of the Meta-Learner model.

III.RESULTS AND DISCUSSIONS

To evaluate the effectiveness of different machine learning models in predicting surface roughness parameters (Ra and Rz) based on 3D printing inputs, a comprehensive comparative study was conducted. The performance of each model was assessed using two key metrics:

1) Root Mean Squared Error (RMSE) – Lower values indicate better prediction accuracy.

2) Coefficient of Determination (R^2 Score) – Values closer to 1 represent better goodness-of-fit.

The Summary of results for models 1 to 5 are given in Table V. A visualization of model results given in Figure 6.

TABLE IV
RESULTS OF MODEL 1 TO 5

MODEL	RMSE	R^2 SCORE
Random Forest Regressor	4.7829	0.8023
XGBoost	5.0847	0.7872
SVM Regressor	5.1964	0.7842
Polynomial Regression (Degree 2)	5.6215	0.7241
Lasso Regression	6.3892	0.6580

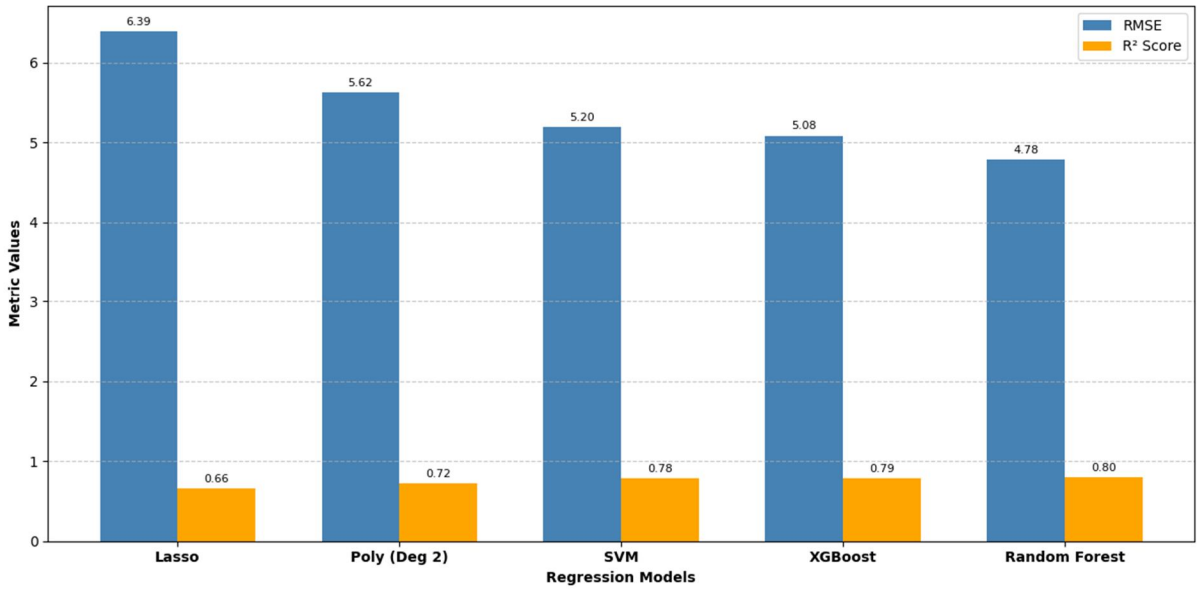


Fig. 6 A visualization of results of models 1 to 5.

Polynomial Regression (Degree 2 and 3) improved performance over linear regression, lasso regression demonstrating that modeling higher-order relationships led to better predictive power. The degree-3 model had RMSE = 5.36 and R^2 = 0.7467. Random Forest emerged as the top performer with the lowest RMSE (4.7829) and highest R^2 (0.8023). This confirms the strength of ensemble-based tree models in handling non-linearity and feature interactions. Support Vector Regression and XGBoost also performed competitively, with R^2 values of 0.7842 and 0.7872 respectively, and low RMSE values below 5.2. These models balance bias-variance well and are robust against noise. Among all methods, Random Forest slightly outperformed others, followed by XGBoost, and then SVR, making them the most suitable for this prediction task.

A. Best Model: Stacked Ensemble Learning Model Strategy

The Model was trained and evaluated with 80-20 train-test split.

It Achieved R^2 scores:

- Ra: ~0.8804
- Rz: ~0.8901

Using multi-output regression allowed the model to exploit the strong correlation between Ra and Rz, improving learning synergy and performance. Ensemble stacking with a GBR meta-learner outperformed individual models.

Figure 7 shows the distribution of errors in the test data for Ra and Rz, with their average values.

Figure 8 visualizes actual vs predicted values for Ra and Rz.

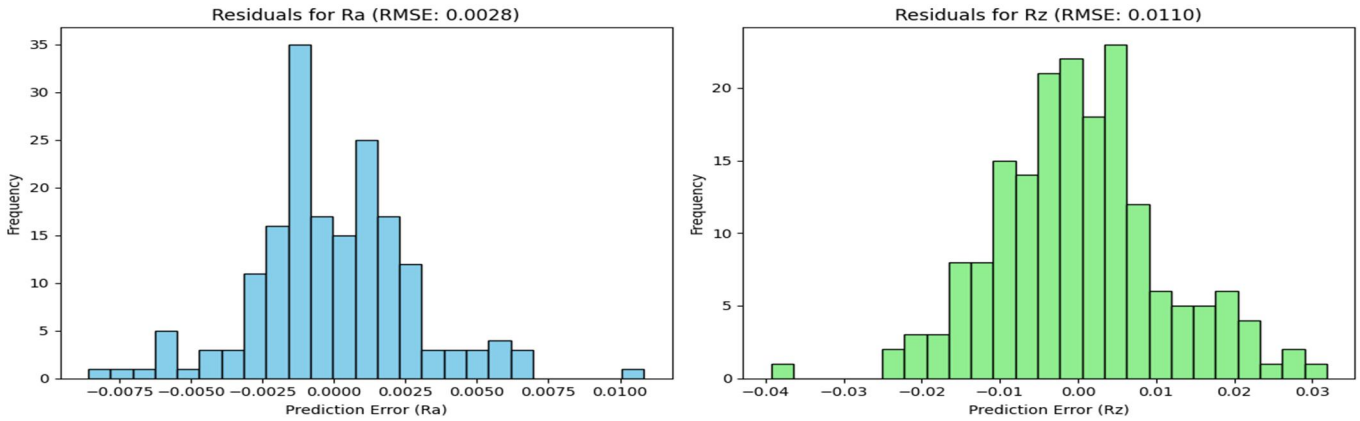


Fig. 7 Distribution of residuals of Ra and Rz.

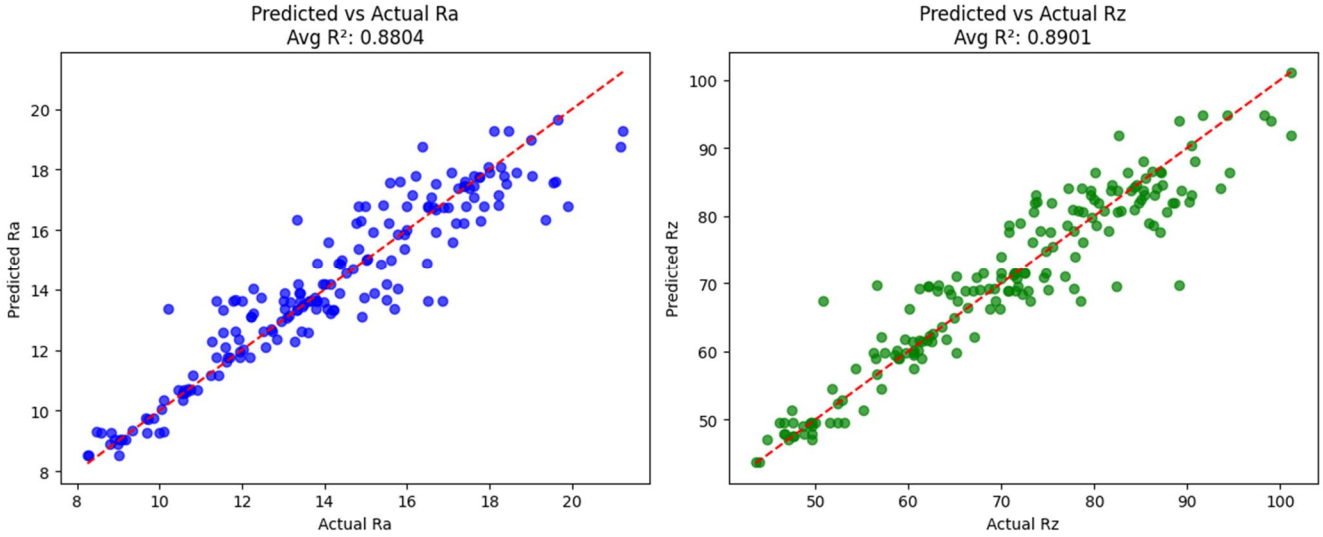


Fig. 8 Predicted Vs Actual Scatter plot for both Ra and Rz.

IV. CONCLUSIONS

This research demonstrates the effectiveness of machine learning techniques in predicting surface roughness metrics Ra and Rz for Fused Filament Fabrication (FFF) using both rigid (PLA) and flexible (TPU) thermoplastics. By integrating experimental data with advanced regression algorithms and a stacked ensemble learning strategy, the study achieved high predictive accuracy, with near-perfect R^2 values when including spatial measurement features. The results confirm that ensemble models outperform other models. This approach can significantly reduce trial-and-error in 3D printing parameter selection, enabling improved dimensional accuracy, mechanical performance, and production efficiency.

V. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to their institute for providing the facilities and support necessary to carry out this research. We also extend our thanks to the laboratory staff for their assistance during the experimental phase. Surface roughness measurements were made possible with the help of precision instruments such as the stylus profilometer and optical measurement tools, which were crucial for data collection. Finally, we are grateful to our peers and collaborators whose feedback helped improve the quality of this work.

REFERENCES

[1] J. S. Chohan, R. Singh, and K. S. Boparai, “Parametric optimization of fused deposition modelling by using response surface methodology for PLA,” Int. J. Adv. Manuf. Technol., vol. 89, no. 5–8, pp. 2251–2262, Jun. 2017. [Online]. Available: https://doi.org/10.1007/s00170-016-9205-2

[2] S. H. Ahn, M. Montero, D. Odell, S. Roundy, and P. K. Wright, “Anisotropic material properties of fused deposition modeling ABS,” Rapid Prototyping J., vol. 8, no. 4, pp. 248–257, 2002. [Online]. Available: https://doi.org/10.1108/13552540210441166

[3] D. Shahrjerdi and W. W. Wits, “Parametric study and optimization of TPU mechanical properties in fused deposition modeling,” Addit. Manuf., vol. 24, pp. 234–243, Dec. 2018. [Online]. Available: https://doi.org/10.1016/j.addma.2018.10.039

[4] F. Górski and R. Wichniarek, “Surface roughness analysis of FDM parts using profile and areal parameters,” Materials, vol. 14, no. 11, Art. no. 2963, 2021. [Online]. Available: https://doi.org/10.3390/ma14112963

[5] S. Singh, S. Ramakrishna, and R. Singh, “Material issues in additive manufacturing: A review,” J. Manuf. Process., vol. 25, pp. 185–200, Jan. 2017. [Online]. Available: https://doi.org/10.1016/j.jmapro.2016.11.006

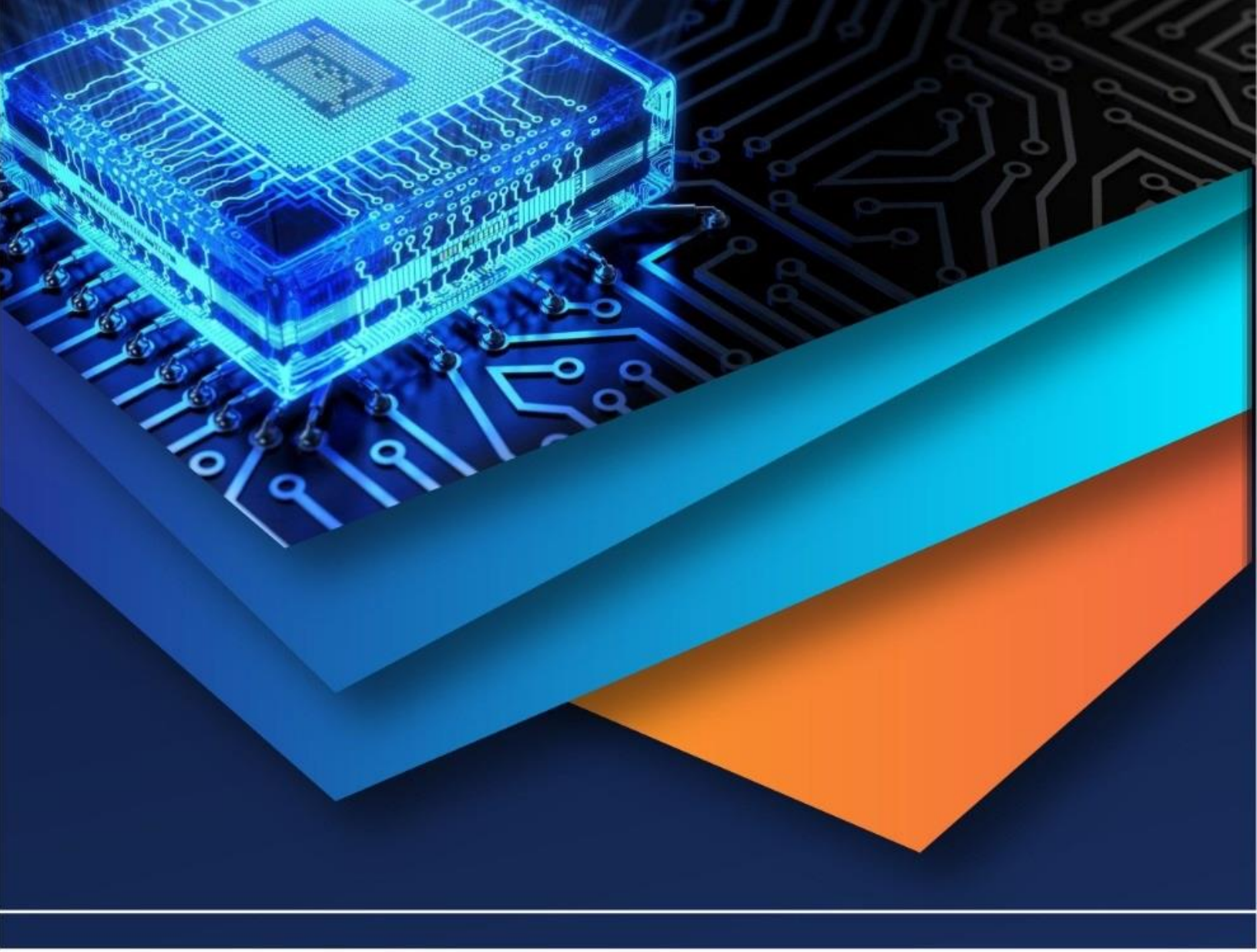
[6] S. Dey and N. Yodo, “Machine learning techniques for fault detection and diagnosis in additive manufacturing: A review,” Addit. Manuf., vol. 35, Art. no. 101248, Dec. 2020. [Online]. Available: https://doi.org/10.1016/j.addma.2020.101248

[7] P. Kumar, S. K. Singh, and A. Tiwari, “Optimization of surface roughness and dimensional accuracy in fused deposition modeling using Taguchi method,” Materials Today: Proceedings, vol. 21, pp. 1593–1599, 2020. [Online]. Available: https://doi.org/10.1016/j.matpr.2019.11.244

[8] G. Torres, A. Jerez-Mesa, P. Travieso-Rodriguez, and S. Martorell, “Machine learning techniques applied to FDM 3D printing process: A state of the art review,” Materials, vol. 13, no. 14, Art. no. 3244, 2020. [Online]. Available: https://doi.org/10.3390/ma13143244

[9] R. Tiwary, A. Jain, and D. Choudhary, “Evaluation of surface roughness in FDM-printed PLA parts using image processing and machine learning,” Measurement, vol. 175, Art. no. 109169, 2021. [Online]. Available: https://doi.org/10.1016/j.measurement.2021.109169

[10] J. Stano, A. Kováčik, and M. Hatala, “Investigation of surface roughness and dimensional accuracy in FDM technology with different materials and layer thickness,” Adv. Mech. Eng., vol. 13, no. 3, pp. 1–11, 2021. [Online]. Available: https://doi.org/10.1177/16878140211000387



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY