Machine learning surrogate models for Hertzian contact stress prediction in gear design: A comparative study of multiple approaches

NEXRES 2 (2025) 100940

Contents lists available at ScienceDirect

Next Research

journal homepage: www.elsevier.com/locate/nexres

Machine learning surrogate models for Hertzian contact stress prediction in gear design: A comparative study of multiple approaches

Fabio Bruzzone *, Daniele Fabbri, Carlo Rosso

Politecnico di Torino, Department of Mechanical and Aerospace Engineering, Corso Duca degli Abruzzi 24, Torino, 10129, Italy

ARTICLE INFO

Keywords:
Machine learning
Gear design
Hertzian stress
Neural networks

ABSTRACT

Accurately predicting contact stress in gears is essential for ensuring durability and optimizing performance during the design stage. This study investigates machine learning surrogate models for predicting Hertzian contact stress in involute gear pairs, aiming to accelerate the gear design process. Unlike approaches based on finite element simulations or experimental data, the proposed method relies solely on stress values from ISO 6336 formulations. A comprehensive dataset covering various macro geometries and loading conditions is used to train and evaluate several regression models, including Elastic Net, Support Vector Regressor, ensemble methods, and Neural Networks. To improve computational efficiency and address the high dimensionality of the input space, Principal Component Analysis is explored as a dimensionality reduction technique. The study provides a detailed comparison of surrogate modeling approaches based on predictive accuracy and generalization. Results show that surrogate models can accurately reproduce ISO based predictions, offering a faster alternative to traditional methods. Focusing on Hertzian stress, rather than root stress, offers deeper insight into surface fatigue and pitting resistance for gear life improvement. This work establishes the foundation for future surrogate models that optimize gear pairs by balancing multiple design objectives and constraints, aiding engineers in selecting optimal configurations.

1. Introduction

Machine learning (*ML*) surrogate models are becoming increasingly important tools in gear engineering, offering powerful capabilities for modeling, diagnosis, and optimization. Traditional approaches, such as analytical models, finite element simulations (*FEM*), or purely experimental validation are often resource intensive and limited in their flexibility to adapt to the wide variety of geometries and operating conditions encountered in modern gear systems. *ML* handles these challenges by understanding complex nonlinear mappings from input data to target variables, which enables accurate predictions, early fault detection, and effective exploration of the design space. There are typically two main categories for machine learning algorithms: supervised and unsupervised learning. Exploring patterns and structures in unlabeled data is possible through unsupervised methods. In this case, the model is not informed about which variables represent inputs or outputs; Instead, it uses techniques such as clustering or rule-based learning to discover hidden relationships or groupings [1]. Although unsupervised learning has interesting applications, particularly for anomaly detection [2] or pattern discovery, the most widely adopted framework in gear engineering design is supervised regression. This category includes both regression tasks where the output is a continuous variable (e.g., stress or noise level) and classification task, where the output is categorical (e.g., healthy vs. faulty gear) [3,4].

Supervised regression, in particular, has proven effective for predicting stress, wear, acoustic emissions, and other performance metrics under various operating conditions. To assess the accuracy and reliability of such predictive models, a range of evaluation metrics is typically employed. Mean Absolute Error (*MAE*), Mean Squared Error (*MSE*), Root Mean Squared Error (*RMSE*), and the coefficient of determination (R^2) are all commonly used evaluation metrics in regression tasks [5]. *MAE* measures the average absolute difference between predictions and true values, giving equal weight to all errors, whereas *MSE* and *RMSE* penalize larger errors more heavily, with *RMSE* expressed in the same units as the target variable. The R^2 score indicates the proportion of variance in the dependent variable that is predictable from the independent variables. In this study, *RMSE* was selected to quantify the deviation from true values of contact pressure, a parameter of critical importance in gears design, meanwhile R^2 was used to evaluate the overall explanatory capacity of the model. The combination of these metrics provides a comprehensive and interpretable assessment of model performance.

* Corresponding author.
E-mail address: fabio.bruzzone@polito.it (F. Bruzzone).

https://doi.org/10.1016/j.nexres.2025.100940
Received 18 August 2025; Received in revised form 4 October 2025; Accepted 13 October 2025
Available online 29 October 2025

In applied research, several recent works have demonstrated the effectiveness of these models in specific gear engineering contexts. Haefner et al. [6] developed a Neural Network (*NN*) based surrogate model to predict tooth root stress in micro gears using measured shape deviations. The model was trained on *FEM* outputs and incorporated uncertainty quantification via GUM [7] Monte Carlo simulations, resulting in a model capable of providing not only point predictions but also confidence intervals. This is particularly valuable for quality assurance and real time monitoring in micro scale components.

For non involute polymer gears (S-gears), where standard formulas are inapplicable, Urbas et al. [8] trained multiple *ML* regressors including random forest, AdaBoost, and *NN* on 456 *FEM* simulations to predict the shape factor Y_z. The best models achieved R^2 values over 0.9, showing the feasibility of real time surrogate models for stress estimation across a range of geometrical, materials and load variations.

On the experimental front, Hyoung et al. [9] addressed the problem of gear whine noise prediction based on micro geometry and test bench data. They collected a dataset from 12 transfer gear sets, each characterized by 24 features and recorded sound pressure levels under various loads. After *LASSO* regression for feature selection, they trained multiple models including random forest, gradient boosting, *LightGBM*, and *XGBoost*. The latter model consistently outperformed ordinary least squares (*OLS*) regression, improving R^2 by up to 63.5 % and reducing *RMSE* by up to 78.6 %. Moreover, their analysis revealed that the influence of specific geometric features on noise depended heavily on load conditions highlighting the importance of condition dependent modeling in gear design.

Machine learning has recently emerged as a powerful tool for predicting the Static Transmission Error (STE) of gears, which is one of the primary excitations leading to gear noise and vibration. Traditionally, STE is computed through finite element simulations or semi-analytical tooth contact analysis, both of which are computationally expensive when exploring large design spaces. Surrogate models based on ML provide an efficient alternative by learning the nonlinear mapping between gear geometry deviations and the resulting STE. For example, Willecke et al. [10] developed a deep neural network trained on thousands of ZAKO3D simulations [11], using a compact representation of flank deviations through the sum deviation surface. Their model was able to reproduce STE characteristics such as RMS, fluctuation width, and gear mesh orders with near real-time performance, achieving speed ups of several orders of magnitude compared to conventional simulations. These results demonstrate that ML based approaches can not only replicate high fidelity STE calculations with high accuracy, but also make it feasible to apply STE prediction in large scale design optimization or even real time gear pairing during production. Papalexis et al. [12] developed a neural network surrogate model for predicting static transmission error (STE) in polymeric gears, trained on 11,000 finite element simulations spanning geometry, material, and load parameters. By working in a dimensionless parametric space and preserving the periodic symmetry of meshing cycles, the model achieved a mean absolute percentage error of 0.49 % on unseen test curves about an order of magnitude more accurate than analytical formulas, while requiring only milliseconds per prediction. The study shows that data-driven surrogates can surpass physics-based methods in both accuracy and computational efficiency, offering a reliable tool for gear design and optimization 1

A significant body of research has focused on applying *ML* to fault diagnosis and condition monitoring of gear systems. These diagnostic models are predominantly trained on vibration, acoustic, or motor current data to classify the health state of a gear, identifying defects such as cracks, pitting, wear, and faulty teeth [13–15]. A variety of supervised and unsupervised algorithms, including Support Vector Machines (*SVM*) [16,17], *NN* and K-nearest neighbors [18], have been successfully employed for multi-fault classification. While invaluable for predictive maintenance and operational reliability, these diagnostic models are inherently reactive, as they are designed to detect faults that have already initiated. Another major application of computational methods is gear design optimization, where the goal is to find the optimal set of geometric parameters that minimizes a target objective, such as stress, noise, or vibration. Evolutionary methods, particularly Genetic Algorithms (*GAs*), have been widely used to explore the design space and identify superior gear micro-geometries [19]. A common theme in these approaches is their reliance on computationally expensive simulations, such as *FEM* or Loaded Tooth Contact Analysis (*LTCA*), to evaluate the fitness of each candidate design [20,21]. However the dependency on high-fidelity simulations can become a significant bottleneck, limiting the number of design iterations and disabling rapid optimization.

Recent developments in machine learning have led to the emergence of physics informed neural networks (*PINNs*) and hybrid modeling frameworks as powerful tools for simulating mechanical systems, particularly where traditional finite element methods (*FEM*) are computationally prohibitive. In gear dynamics, Tuiran et al. [22]. proposed a hybrid analytical FEM model to estimate vibration spectra under combined misalignments, balancing accuracy and computational cost for design stage fault prediction. Similarly, Monteiro et al. [23] leveraged a FEM analytical approach to synthetically generate vibration data for healthy and cracked gear states, subsequently used to train artificial neural networks (*ANNs*) for condition monitoring and fault classification. Expanding this paradigm, Yang et al. [24] introduced a temporally correlated *PINN* to estimate time varying meshing stiffness (*TVMS*) and static transmission error (*STE*) from noisy vibration signals. Their model enabled robust diagnosis of pitting and tooth crack failures by extracting pseudo *TVMS* features consistent with physical meshing laws. In parallel, Bolandi et al. [25] proposed *PINN Stress*, a spatiotemporal architecture capable of learning dynamic stress distributions from FEM data. By embedding the governing equations of motion into the loss function, their model achieved real time, generalizable stress prediction across varying geometries and load conditions, offering a viable surrogate for structural failure analysis. Finally, Sahin et al. [26]. extended *PINNs* to contact mechanics, formulating a mixed variable network that simultaneously learns displacements and stresses, while enforcing contact inequalities through complementarity functions. Their framework addressed forward simulations, inverse identification, and surrogate modeling of Hertzian contact, laying the groundwork for future *PINN* based modeling of gear tooth interactions. Collectively, these studies illustrate the versatility of hybrid and physics informed methods for gear diagnostics and dynamic analysis.

Taken together, these studies illustrate that machine learning is no longer a theoretical, but a practical and scalable tool for a wide range of tasks in gear design, monitoring, and optimization. From predicting root stress and diagnosing faults to anticipating noise emissions and guiding manufacturing decisions, *ML* models deliver not only high predictive accuracy but also actionable insights into the influence of key design parameters. With increasing access to high quality measurement data and simulation results, the integration of *ML* into gear engineering workflows is increasing. Also, the development of surrogate models, or metamodels, has become a cornerstone of modern engineering design, allowing for the rapid approximation of complex and computationally expensive analyses [27]. These models learn the input/output relationships of a high fidelity simulation or analytical formula, enabling real time prediction and efficient optimization. In gear engineering, this approach is particularly valuable for accelerating the iterative design process. Indeed the proposed methodology can be framed within the broader context of surrogate modeling or data driven model order reduction [28], where the objective is to replace a high fidelity, computationally intensive model with a fast and accurate approximation. The primary contribution of our study lies in leveraging a well established and validated analytical standard ISO 6336 [29] as data source for training the surrogate model. This approach combines the robustness and accumulated knowledge embedded in the ISO standard with the speed and interpolative power of machine learning. By focusing specifically on Hertzian contact stress, our model provides a tool for assessing surface durability and pitting resistance, which are critical failure modes in

Table 1
Comparison of recent ML applications in gear engineering.

Reference	Dataset	Method	Target	Key outcome
Haefner et al. [6]	FEM simulations of micro gears with measured deviations	Neural Network + GUM Monte Carlo	Tooth root stress	Accurate prediction with uncertainty quantification for micro-scale gears
Urbas et al. [8]	456 FEM simulations of S-gears (noninvolute polymer)	Random Forest, AdaBoost, NN regressors	Shape factor Y_z	Random forest and AdaBoost achieved $R^2 > 0.9$, showing feasibility in the design of nonstandard gear geometries
Lee & Park [9]	24 inspection features + noise bench data	LASSO + RF, GBM, LightGBM, XGBoost	Gear whine noise	XGBoost outperformed OLS with up to 63.5 % higher R^2 and 78.6 % lower RMSE
Willecke et al. [10]	3000 simulated variants (sum deviation surface, 18 parameters)	Neural Network	Transmission Error (TE) under load	Accurate prediction of RMS, fluctuation width, mesh orders; up to 10^6 faster than numerical simulations, enabling near real-time gear pairing
Papalexis et al. [12]	11,000 FEM-generated STE curves (17D parametric space)	Fully connected Neural Network surrogate	Static Transmission Error (STE)	Mean absolute percentage error of 0.49 %; surpasses analytical formulas and enables real-time prediction of full STE curves
This study	Synthetic dataset from ISO 6336 analytical stresses (5×10^6 samples)	Elastic Net, Ensemble methods, Neural Networks	Hertzian stress (ISO-compliant)	Foundational surrogate framework for gear design, enabling integration into multi-objective optimization and rapid evaluation of robust gear configurations

many gear applications, delivering proactive, design-stage prediction of stress to ensure durability and prevent failure from occurring in the first place. The ISO standard also provides the safety factor for the tooth root bending stress. Those factors are however much more linearly correlated to the input features and as such would not be a strong indicator for the performance, reliability and robustness of the different approaches that will be discussed.

The objective of this work is to establish a foundational methodology for creating comprehensive gear design surrogates models that can be integrated into multi objective optimization frameworks to quickly identify optimal and robust gear pair configurations and is structured as follows.

In Section 2, the procedure used to generate the dataset is described. The dataset is constructed entirely from analytical contact stress values computed using the ISO 6336 standard, covering a broad range of gear macro geometries and loading conditions. The same section presents the machine learning workflow, detailing the training and evaluation strategies adopted for various regression models, including linear regression, ensemble algorithms, and neural networks. All models were built using widely used machine learning libraries that are highly optimized for regression tasks, such as *scikit-learn* [30] and *Keras* [31] through *TensorFlow* [32]. In Section 3, the predictive performance of the trained models is assessed on a separate validation dataset, demonstrating their ability to accurately replicate ISO based stress estimates. Finally, the conclusions in Section 4 summarizes the key findings and outlines future directions in the development of surrogate models for gear design.

2. Methodology

The aim of the work is to develop surrogate models capable of accurately predicting Hertzian contact stress in spur gear pairs, using only input features related to gear macro geometry and loading conditions. The output variable is the Hertzian stress, denoted as $\sigma_{H1,2}$, which is calculated analytically based on the ISO 6336-2 formulation in which the subscripts $1, 2$ denotes the pinion and the driven gear respectively. This target feature was chosen because it is crucial in gear design for durability and reliability and has a strongly nonlinear relationship with the input parameters. To build the training dataset, a wide range of gear configurations was generated by systematically varying key macro geometric and load parameters. These include variables such as module, number of teeth, face width, pressure angle, helix angle, and applied torque. A full list of input features is provided in Table 2.

The calculation of surface durability against pitting due to Hertzian contact stress, as prescribed by ISO 6336-2 and automatized in *GeDy TrAss* software [33–35], fundamentally compares the calculated contact stresses, $\sigma_{H1,2}$, with the material's permissible contact stress, σ_{HP}. The calculated stress for each gear is determined by scaling a nominal Hertzian stress, σ_{H0}, with factors accounting for operational conditions and specific gear geometry. The core equations are:

$$\sigma_H = (Z_B \text{ or } Z_D) \cdot \sigma_{H0} \sqrt{K_A \cdot K_v \cdot K_{H\beta} \cdot K_{H\alpha}} \tag{1}$$

where the nominal stress, σ_{H0}, integrates the main geometric and load parameters:

$$\sigma_{H0} = Z_E \cdot Z_\varepsilon \cdot Z_H \cdot Z_\beta \sqrt{\frac{F_t}{d_1 \cdot b} \frac{\tau + 1}{\tau}} \tag{2}$$

The tangential load, F_t, is directly derived from the applied torque T on the pinion and its pitch diameter d_1 via the relation $F_t = 2T/d_1$. The gear's geometry is fundamentally scaled by the normal module, m_n, which defines the pinion operating pitch diameter d_1 while the transmission ratio is represented by τ.

While not explicit in the main formula, detailed geometric parameters such as the addendum (h_a), dedendum (h_f), and the cutter's tip radius (ρ_{Fp}) are crucial as they govern the values of key modifying factors of the root radius ρ_a. Notably, the contact ratio factor, Z_ε, is a function of the transverse contact ratio, ε_α. This ratio is calculated from the active length of the path of contact, which is directly determined by the addendum circles (d_{a1}, d_{a2}) of the mating gears, themselves defined by $d_a = d + 2h_a$ and also by the profile shift coefficients x_1 and x_2 which also influences the working pitch diameter d_{w1} which is in turn dependent on the transversal pressure angle α_t. Furthermore, the single pair contact factors, Z_B for the pinion and Z_D for the driven gear, account for the stress concentration at the point of highest single tooth loading. These are calculated from the ratio of flank curvatures between the pitch point and the start/end of single-tooth contact, which are themselves a direct consequence of the detailed tooth profile shaped by h_a, h_f, and the tool geometry (ρ_{Fp}). The final design is validated if the condition $\sigma_H \leq \sigma_{HP}$ is met for both gears in the mesh, while usually the pinion is more critical and hence this study is focused on σ_{H1} [36].

For each combination of input parameters, the corresponding σ_{H1} value was computed using the ISO standard equations. The computational time for each combination of input parameters is negligible and this allows for the full exploration of the design space consisting of the factorial of all possible combinations and permutations in a feasible amount of computational time. Therefore no sampling strategy was

Table 2
Features and label.

Type	Symbol	Dataset label	Range
Target variable	σ_{H1}	sigma_H1	14 ÷ 792 MPa
Feature	T	Input_T	10 ÷ 3000 Nm
Feature	h_a	Input_ha	1.25 ÷ 1.8
Feature	h_f	Input_hf	1 ÷ 1.6
Feature	ρ_a	Input_rhoa	0.1 ÷ 0.33
Feature	τ	Input_tau,	2 ÷ 6
Feature	α_t	alphat	20 ÷ 21.17
Feature	b	b	10 ÷ 70 mm
Feature	β	beta	0 ÷ 20°
Feature	d_{w1}	dw1	15.28 ÷ 241.2 mm
Feature	i	i	25.19 ÷ 792.97 mm
Feature	m_n	mn	1 ÷ 6 mm
Feature	x_1	x1	−0.6 ÷ 0.9
Feature	x_2	x2	−0.6 ÷ 0.9
Feature	z_1	z1	17 ÷ 35

Material: 34CrNiMo6, ISO code V, Brinell Hardness 319HB, Hardened and Tempered.

employed in the current study for the generation of the dataset. The resulting dataset composed only of safe designs was then used to train multiple supervised regression models, including linear Elastic Net, ensemble methods (e.g., random forest, histogram based gradient boosting), and neural networks.

2.1. Dataset preparation

Before training, the dataset was cleaned by removing all non numeric entries the number of samples were approximately 5,350,000 for a total of 15 features and 1 label. As a fundamental step for a good machine learning process, the dataset has been analyzed in depth to understand the features and label distribution. The first passage was to analyze for the presence of effective outliers and in case delete them, then to evaluate the distributional properties of the input features. Skewness and kurtosis [37] were calculated for all numerical variables. This analysis was conducted to detect deviations from normality, which may negatively impact the performance of certain regression algorithms. Skewness measures the asymmetry of a distribution, while kurtosis assesses the heaviness of its tails relative to a normal distribution. Features were ranked according to the absolute value of their skewness, thereby identifying the most significantly skewed variables for potential correction.

For features exhibiting substantial skew, a data transformation procedure was employed to reduce asymmetry. Two transformation techniques were considered: the logarithmic transformation ($log1p$), applicable only to strictly positive values, and the *Yeo-Johnson* [38] transformation, which is suitable for variables containing both positive and negative values. This process allowed for systematic normalization of skewed features, improving the overall distributional properties of the dataset and enhancing the robustness of subsequent machine learning models (Fig. 1).

To improve dataset suitability for regression modeling, some techniques were applied to reduce multicollinearity [39,40] and select statistically relevant features. The correlation between each characteristic and the target variable (σ_{H1}) was computed, which allowed to know how strongly each individual feature was linearly associated with the output of interest. The correlation among the features was analyzed using a full Pearson correlation matrix [41]. Features exhibiting perfect correlation (equal to 1) were also examined, not just from a mathematical point of view, but also by their physical significance. This step enabled the identification of features that, despite having different names, have the same underlying physical meaning. To avoid singularity in matrix operations and avoid overfitting in subsequent models, redundant features were flagged and removed from the dataset. The new feature set was then combined with the target variable, and a heat map was generated to confirm the absence of excessively correlated feature pairs. The final dataset consisted of approximately 5,350,000 samples, with 14 features and 1 target label, all listed in Table 2.

Subsequently, Principal Component Analysis (*PCA*) [42,43] was employed to comprehend the structure of the input features and potentially reduce the dataset size. The process began by doing *PCA* to the scaled dataset, resulting in the transformation of the original features into a new set of variables called principal components. Each of these components captures a certain amount of the total variance present in the data.

An explained variance plot was created to display how much of the dataset's information is retained as more principal components are added. A total of 10 components were chosen and a new dataset was created based on the transformed features, he PCA analysis retained the 95 % of the variance as shows in 2

A new representation of the data was created by replacing each original input variable with its corresponding principal components, which means that each column (e.g. $PC1$, $PC2$) reflects a part of the original variance. The target variable σ_{H1} was then added back to this new feature set to preserve the original regression objective. The first 10 principal components are loaded with the original features, as reported in Table 3. Each loading shows how a given feature contributes to a particular principal component, allowing insight into how the original input space is transformed.

This *PCA* transformed dataset creates a distinct branch of the analysis. Two parallel modeling workflows were created, one using the original features after cleaning and multicollinearity analysis, and the other based on the new transformed dataset. This separation allows for a direct comparison of model performance between an approach based on physical feature interpretation and statistical filtering and a dimensionality reduction approach. The workflow is shown in Fig. 3

The following methodology was applied equally across all models and datasets to ensure a consistent and rigorous model development process. The first step was to perform a grid search [44] on a stratified subset of the training data to identify appropriate hyperparameters. This preliminary step was essential for tuning key parameters such as regularization strength, tree depth, learning rate, and network architecture, depending on the algorithm. The best hyperparameter configuration was selected by analyzing the training and validation loss curves, with particular attention to signs of instability such as oscillations or divergence. The third step involved the retraining of the model with the best hyperparameters on the full training dataset to make the most of all available data and improve generalization. Finally, evaluation was conducted on a test set that was held out to report all performance metrics and validate the final model's predictive capability. In addition to the predictive performance metrics, the training time for each machine learning model was also measured and reported. This provides a more comprehensive comparison between different algorithms, taking into account both their accuracy and computational efficiency. The training durations are included in the results tables. All computations and model training were carried out on a workstation equipped with a *i9-13900F* processor (2.00 GHz) and 64 GB of RAM. The models were trained using the standard CPU version of the machine learning frameworks. GPU acceleration was not employed in this study but is considered as a promising avenue for future work to further reduce training time, especially for more complex neural architectures or larger datasets. By using this workflow, it is possible to compare algorithms and feature sets with reproducibility.

2.2. Linear regression model

Linear regression is frequently employed as a baseline model. It estimates a linear relationship between inputs and an output by minimizing the least squares error. Although its simplicity limits its ability to capture nonlinear interactions, linear regression remains valuable due to its high interpretability and low computational cost. The inclusion of regularization techniques such as Ridge, Lasso, and Elastic Net [45] can

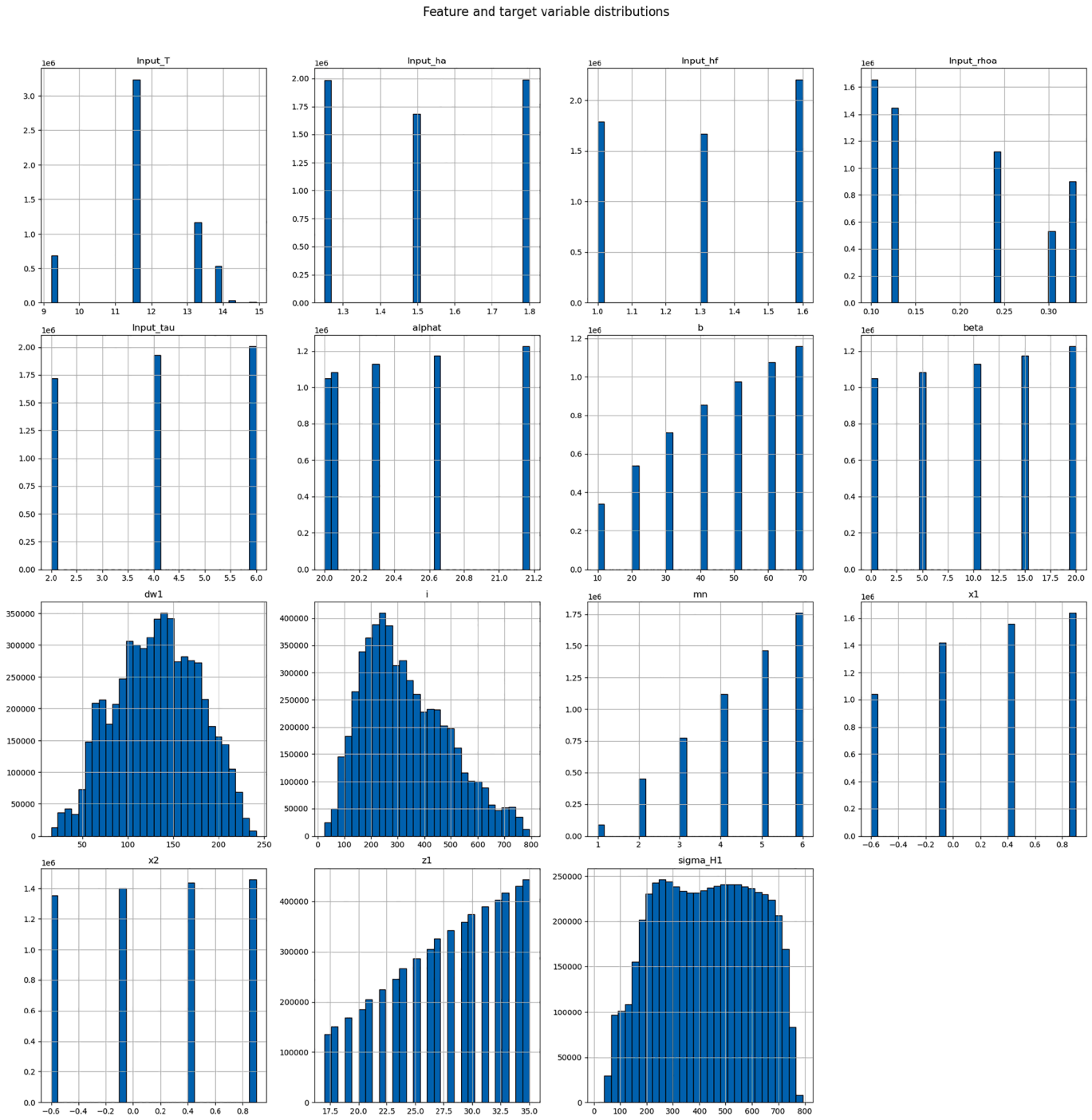


Fig. 1. Dataset after data transformation on the skewed variables.

significantly improve its robustness and performance. The methods used not only prevent overfitting but also help identify the most influential features in the dataset. Regularized linear models can frequently provide predictive performance that is comparable to more complex, less interpretable algorithms like *NN* [46,47] or ensemble models [48,49], making them an appealing choice in engineering contexts where transparency and simplicity are crucial.

2.2.1. *Elastic net model training and validation on the original dataset*

In this study, a linear regression model was trained using stochastic gradient descent (*SGD*) with Elastic Net regularization, aiming to balance sparsity and robustness in high dimensional input spaces. Elastic Net was selected over pure Lasso or Ridge regression due to its ability to combine the strengths of both approaches

The Elastic Net hyperparameters subjected to the grid search included the ℓ_1 ratio, regularization strength α, learning rate strategy *learning rate*, and initial learning rate η_0. However on the best parameters configurations, the loss function showed an unstable oscillatory behavior during the validation phase on the full dataset, with irregular fluctuations and an unstable trajectory. It is likely that this is connected to an inadequate adaptation of the learning rate and insufficient regularization. As a result, several modifications have been introduced to improve numerical stability and convergence. The initial learning rate was increased to accelerate early convergence and the learning rate

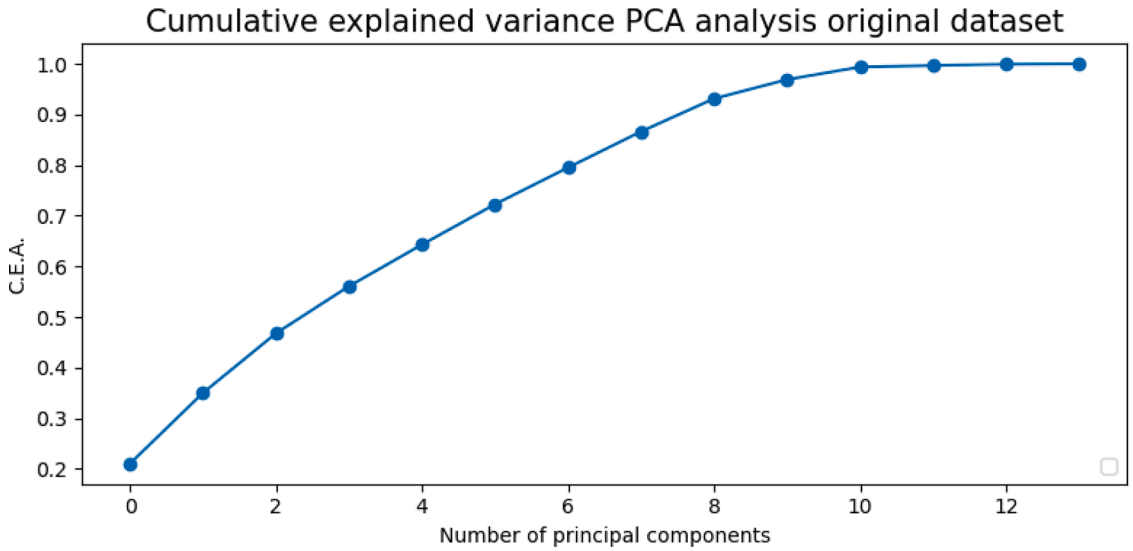


Fig. 2. *PCA* Explained variance.

Table 3
PCA loadings of the first 10 principal components.

Feature	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10
Input_T	0.361	−0.004	−0.093	−0.256	0.411	0.030	0.022	0.015	0.074	0.748
Input_ha	0.038	−0.011	−0.688	0.122	−0.089	0.013	−0.013	−0.018	−0.060	−0.225
Input_hf	−0.007	−0.007	−0.053	0.001	0.133	0.039	0.860	0.382	0.272	−0.127
Input_rhoa	−0.076	0.012	0.692	−0.081	0.022	−0.014	0.007	0.020	0.079	−0.151
Input_tau	0.171	−0.018	0.099	0.780	0.255	−0.119	−0.033	0.002	0.042	0.062
alphat	0.046	0.704	−0.009	0.003	−0.012	−0.013	−0.000	−0.004	0.025	−0.015
b	0.002	0.008	−0.051	−0.201	0.811	0.142	−0.133	−0.112	−0.084	−0.484
beta	0.045	0.704	−0.008	0.002	−0.010	−0.011	−0.001	−0.003	0.025	−0.011
*dw*1	0.537	−0.023	0.035	−0.236	−0.189	−0.006	−0.007	−0.003	0.035	−0.220
i	0.513	−0.026	0.101	0.370	0.028	−0.029	−0.014	0.004	0.000	−0.124
mn	0.460	−0.049	0.053	−0.234	−0.124	−0.314	0.137	−0.009	−0.376	−0.165
*x*1	−0.001	−0.041	−0.094	−0.122	0.043	−0.643	−0.190	−0.074	0.705	−0.103
*x*2	0.008	0.004	−0.019	−0.034	0.033	−0.043	−0.394	0.913	−0.067	−0.031
*z*1	0.256	−0.042	0.002	−0.027	−0.166	0.669	−0.172	−0.015	0.506	−0.097

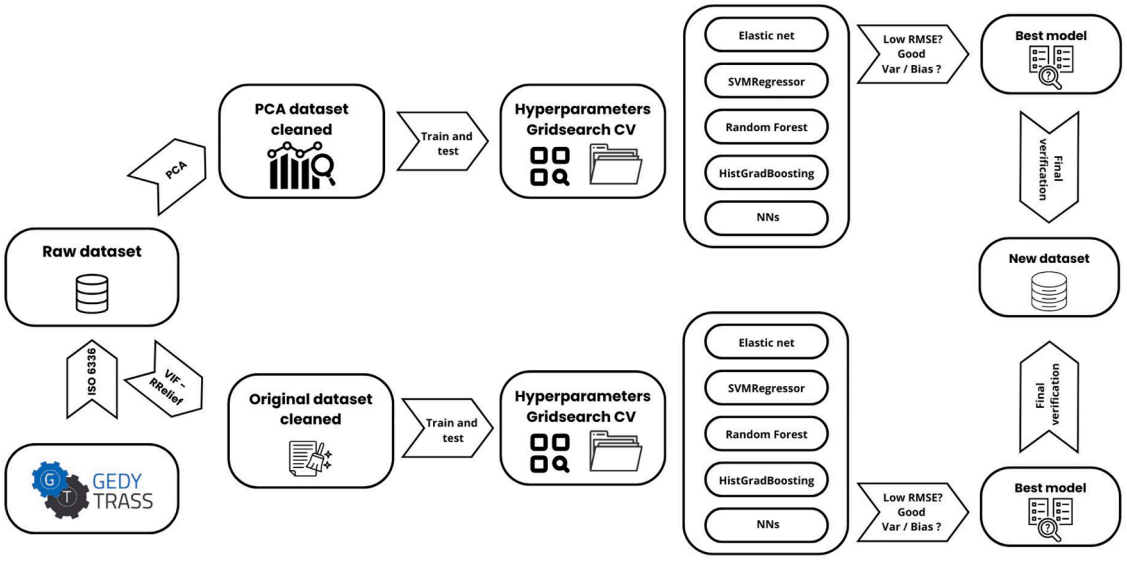


Fig. 3. Synthetic representation workflow.

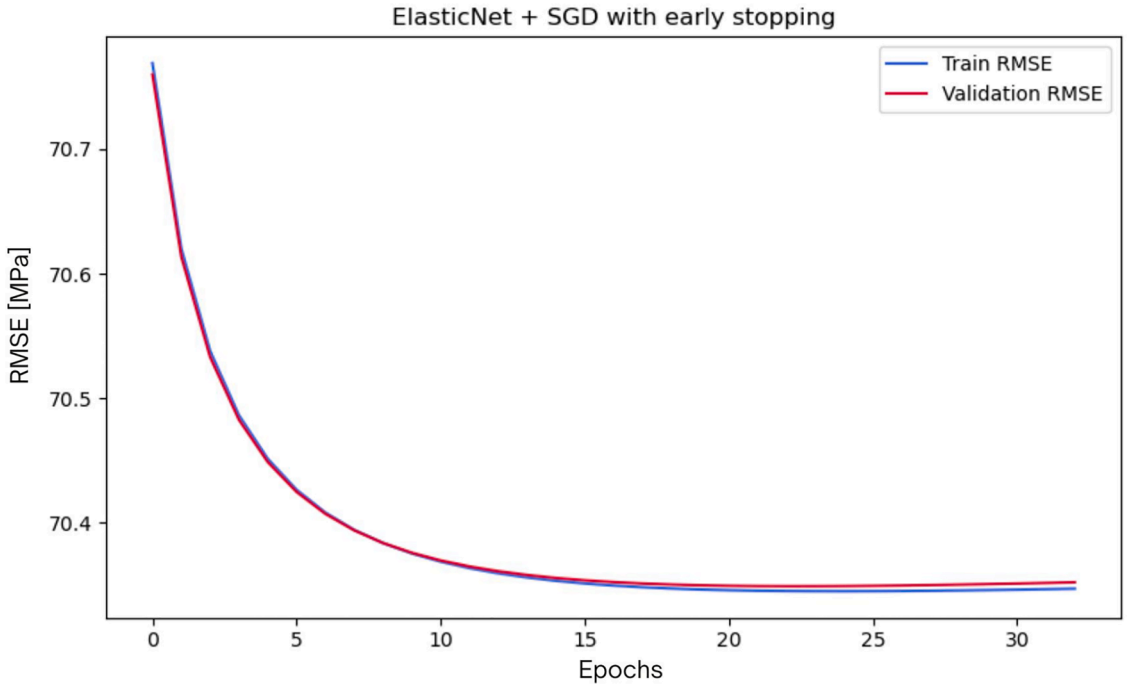


Fig. 4. *RMSE* curves over epochs - Elastic Net original dataset.

Table 4
Differences between the grid search hyperparameters and the parameters used in the final model.

Hyperparameters	Grid search parameters	Final parameters
learning rate η_0	0.01	0.1
regularization strength α	0.0001	0.01
ℓ_1	0.9	0.8
learning rate strategy	adaptive	invascaling
power t	none	0.25

Table 5
Final performance Elastic net SDG regression model on original dataset.

Training time [s]	38
Test *RMSE* [MPa]	70.29
Test *MAE* [MPa]	57.70
Test R^2	0.852

strategy was changed from adaptive to an inverse scaling schedule to ensure more precise decay and avoid unstable behavior. Also, a power factor was added to define the rate at which the learning rate decreases with each iteration. These changes were made to stabilize the training dynamics and to have a smoother convergence. A summary of the differences between the hyperparameters selected by grid search and those used in the final model is reported in Table 4.

The model was trained over multiple epochs on the scaled training set, which was obtained by applying a conventional 80–20 % split to the full dataset. Additionally, 20 % of the training set was used for validation during training. To prevent overfitting, an early stopping strategy was implemented. The validation set was used to monitor model performance during training. After each epoch, the (*RMSE*) for both training and validation was computed, if this value did not improve for 10 consecutive epochs, training was interrupted and the model parameters corresponding to the lowest validation *RMSE* were retained. Fig. 4 shows the *RMSE* curves, which exhibit convergence after approximately 30 epochs. The consistently narrow gap between the training and validation errors indicates stable learning and confirms the effectiveness of the early stopping strategy. The final performance of the model on the test dataset is reported in Table 5.

2.2.2. Elastic net model training and validation on the PCA dataset

The same workflow described for the full dataset was applied to the *PCA* dataset, starting from an initial grid search over hyperparameters. However, similar to the previous case, the best configuration selected by grid search led to unstable loss curves, characterized by oscillations during training. To address this issue, a new configuration was selected and the final set of hyperparameters is listed in Table 6.

Table 6
Differences between the grid search hyperparameters and the parameters used in the final model.

Hyperparameters	Grid search parameters	Final parameters
Initial learning rate η_0	0.05	0.1
regularization strength α	0.0001	0.001
ℓ_1	0.9	0.8
learning rate strategy	adaptive	invascaling
power t	none	0, 25

The training procedure was performed using the stochastic gradient descent (*SGDRegressor*), with a maximum of 500 epochs. An early stopping mechanism based on a patience parameter of 10 epochs was set, however, early stopping was not triggered. The choice of 500 epochs represented a compromise between model efficiency and computational cost, aiming to gain meaningful improvements in *RMSE* without incurring excessive training time or risking overfitting. After each epoch, the *RMSE* was evaluated on both the training and validation sets.

The *RMSE* curves, presented in Fig. 5, demonstrate smooth and stable convergence, characterized by a rapid initial decrease during the early epochs, followed by a more gradual reduction as the learning rate decays. Convergence appears to stabilize after approximately 300/400 epochs, confirming the effectiveness of the learning rate schedule and regularization configuration. The final model performance, evaluated on the *PCA* transformed test set, is summarized in Table 7.

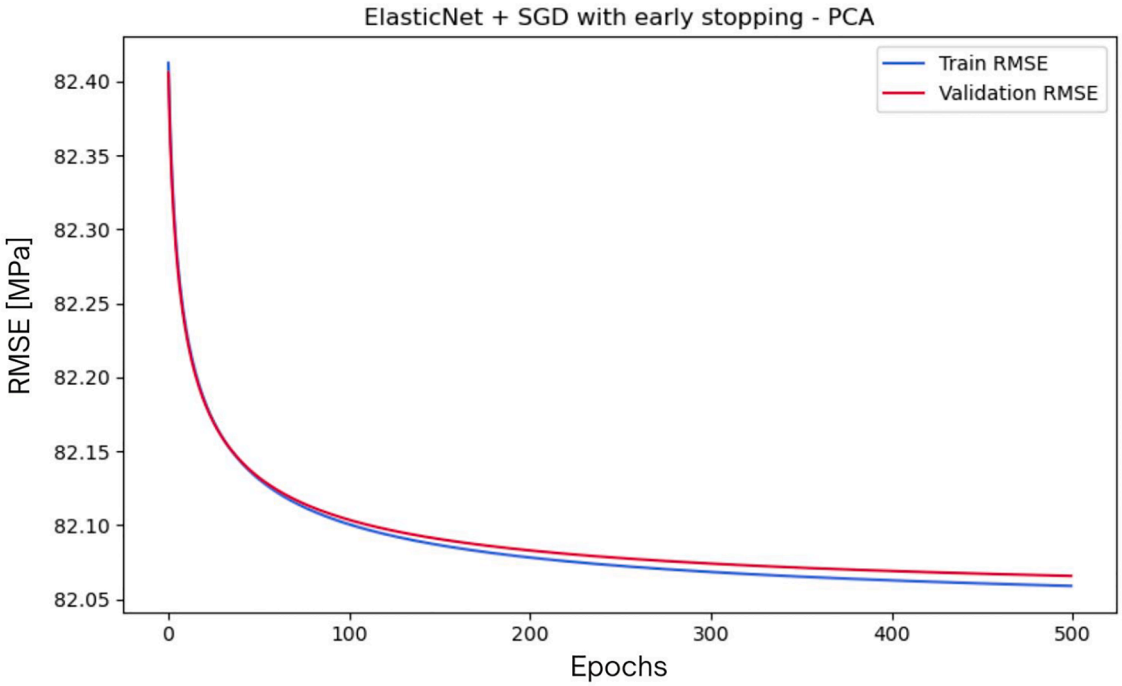


Fig. 5. *RMSE* curves over epochs - Elastic Net *PCA* transformed dataset.

Table 7
Final performance Elastic net SDG regression model on *PCA* dataset.

Training time [s]	300
Test *RMSE* [MPa]	97.56
Test *MAE* [MPa]	78.07
Test R^2	0.7174

Table 8
Hyperparameters and results of Support Vector Regression on the original dataset.

Kernel	*RBF*
Regularization parameter C	10
ϵ	0.1
Gamma	*scale*
Training time (on 50,000 samples) [s]	106
Test RMSE [MPa]	32.47
Test R^2	0.9684

Comparing the results coming from the original (Table 4) and the reduced dataset (Table 6), the *PCA* technique effectively reduced the dimensionality of the dataset. However, the Elastic Net model's performance declined significantly, with higher prediction errors and a lower R^2 score. This suggests that key information relevant to the regression task may have been lost during the transformation.

2.3. Support vector machines regression model

Support Vector Machines for regression (*SVR*) aims to find a function that approximates the data within a specified margin of error. Errors within this margin are not penalized, allowing the model to focus on fitting the most relevant deviations. Two main hyperparameters control the behavior of *SVR*: ϵ: which defines the width of the tolerance margin, and C: the regularization parameter that balances model simplicity with the penalty for errors outside the margin. A third important choice is the kernel, which determines how input data is mapped into a higher dimensional space to capture nonlinear relationships. Common choices include polynomial and radial basis function (*RBF*) kernels. Proper tuning of these hyperparameters is essential to achieve good generalization while avoiding overfitting or underfitting.

2.3.1. Model training and validation on the original dataset

SVR was employed to model the relationship between input features and the target variable [50]. Several kernel types (linear, polynomial, and radial basis function), regularization parameters C, and ϵ values were tried in the grid search procedure with 3-fold cross validation. *SVR*'s quadratic programming optimization problem and the kernel trick make it computationally impractical to train on the entire dataset of 5 million samples and 21 features.

To resolve the issue, the *SVR* pipeline was trained on increasingly larger data subsets, which ranged from 1000 to 50,000 samples. The corresponding *RMSE* values were recorded during training, as shown in Fig. 6. Based on the observed *RMSE* plateau, a dataset of 50,000 samples, maintaining the same feature distribution as the full dataset, was selected as a suitable compromise between computational efficiency and model performance.

The final model results and the hyperparameters used in the model training in the whole dataset are listed in Table 8

2.3.2. Model training and validation on the PCA dataset

An *SVR* model was trained using the reduced *PCA* dataset, following the same methodology as the original dataset. Despite the reduction in feature dimensionality from 14 to 10, the high number of total samples still posed computational challenges. Training on the entire dataset was not an option due to this feature. To address this limitation, the *SVR* pipeline was trained on increasingly larger subsets of the *PCA* transformed dataset, ranging from 1000 to 50,000 samples. The corresponding *RMSE* values were recorded during training, as shown in Fig. 7. An effective compromise between computational cost and model accuracy was achieved by selecting a dataset of 50,000 samples.

The final model results and the optimized hyperparameters used during training on the *PCA* dataset are summarized in Table 9.

In this case, as the Elastic Net model, the results coming from *PCA* dataset showed higher *RMSE* and lower R^2 of the original dataset.

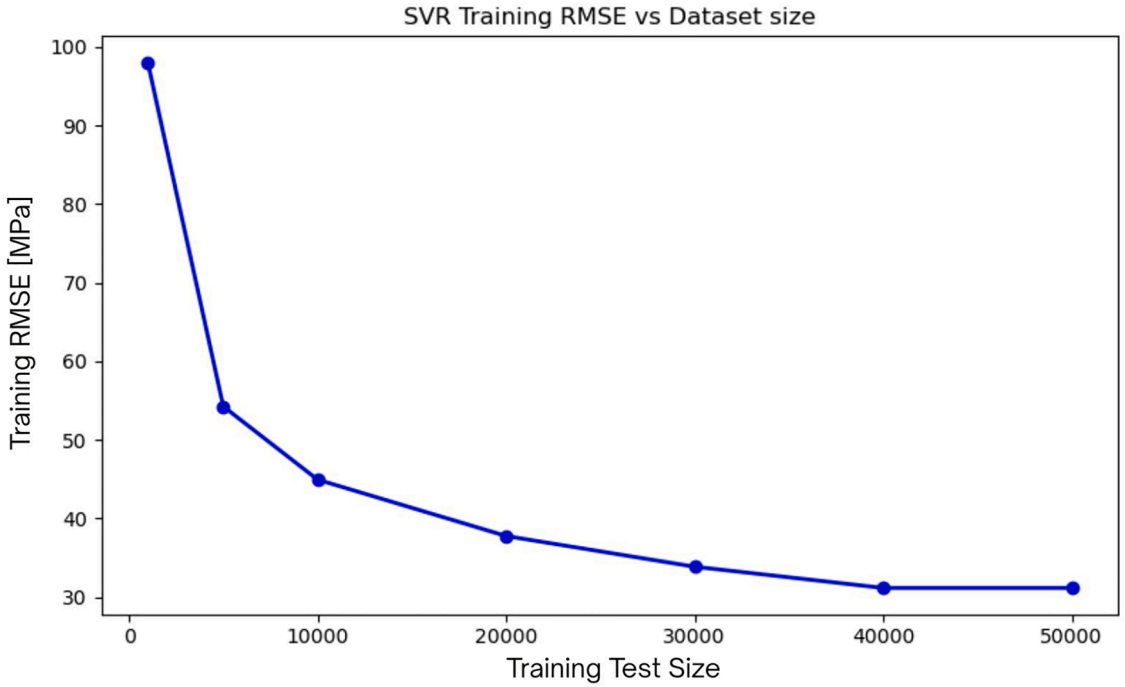


Fig. 6. Dataset size vs training *RMSE* for the *SVR* model on the original dataset.

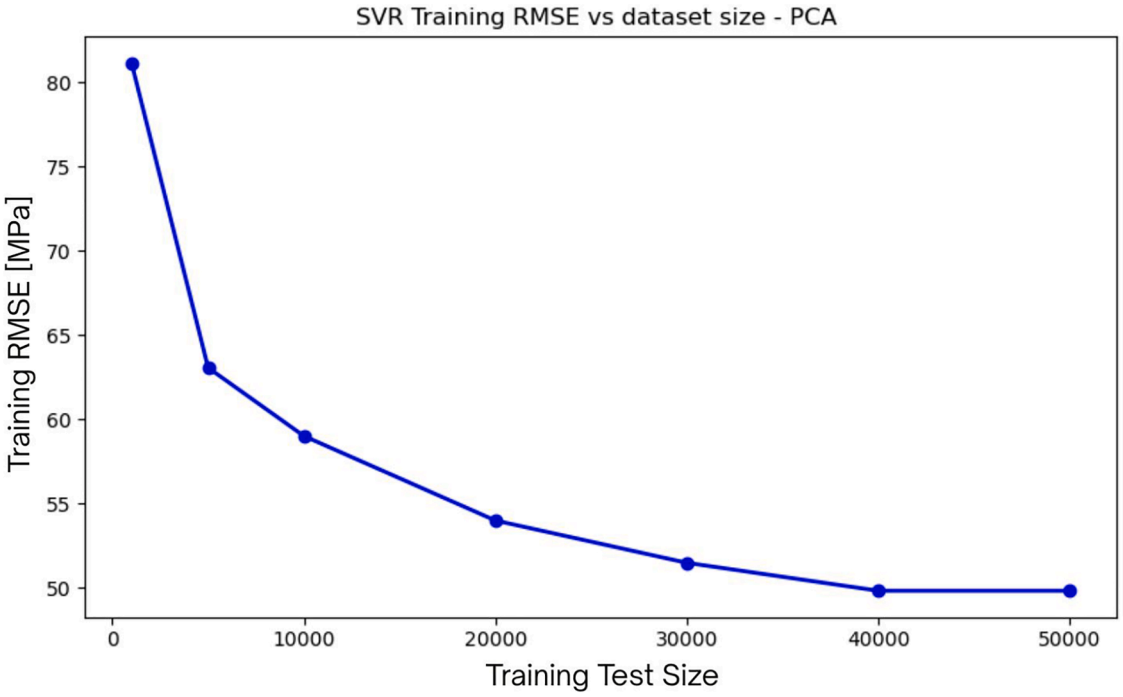


Fig. 7. Dataset size vs training *RMSE* for the *SVR* model on the *PCA* dataset.

2.4. Decision tree and ensemble methods

Decision trees have become popular because of their interpretability and low computational cost. However, individual trees often exhibit high variance and a tendency to overfit. Ensemble learning methods combine the outputs of multiple predictors to improve generalization performance by addressing the limitations of individual decision trees. Among these, bagging and boosting [48,49], represent two different strategies. Multiple base learners can be trained independently in parallel on random subsets of the training data sampled with replacement using bagging methods. To obtain the final prediction, the outputs in regression tasks are averaged. Boosting, instead, builds a base of learners sequentially, where each model is trained to focus on the errors made by its predecessor. Iterative refinement has the potential to reduce bias and generate highly accurate models, but if not regularized, it can also lead to noise sensitivity and overfitting.

In this work, both Random Forest and Histogram based Gradient Boosting were utilized to explore the trade offs between bagging and boosting based approaches. Random Forest [48,49], which is based on bagging, for boosting instead a Histogram based Gradient method has

Table 9
Hyperparameters and results of Support Vector Regression on the *PCA* dataset.

Kernel	*RBF*
Regularization parameter C	10
Epsilon ϵ	0.5
Gamma	*scale*
Training time (on 50,000 samples) [s]	71
Test *RMSE* [MPa]	49.99
Test R^2	0.9258

Table 10
Hyperparameters and results of Random Forest Regressor on the original data.

Max depth	100
Min samples per leaf	10
Min samples to split	10
Number of estimators	300
Train *RMSE* [MPa]	23.97
Training time (on 50,000 samples) [s]	5
Test *RMSE* [MPa]	29.62
Test R^2	0.9743

Table 11
Hyperparameters and results of Random Forest Regressor on the *PCA* dataset.

Max depth	100
Min samples per leaf	10
Min samples to split	10
Number of estimators	300
Train *RMSE* [MPa]	28.39
Training time (on 50,000 samples) [s]	8
Test *RMSE* [MPa]	36.99
Test R^2	0.9256

Table 12
Hyperparameters and results of HistGradBoosting on the original dataset.

Max depth	10
Max iter	300
Min sample leaf	20
Learning rate η	0.1
Training time (on 100,000 samples) [s]	46
Test *RMSE* [MPa]	14.8210
Test R^2	0.9935

been used, the reason for choosing it was its suitability for large scale datasets, as well described in Hang et al. [51]. By fitting new tree to the residuals of previous ones, the boosting method gradually corrects prediction errors. Histogram binning is used to improve computational efficiency by dividing continuous features into fixed bins, which greatly speeds up training and reduces memory consumption.

2.4.1. Random forest model training and validation on the original dataset

To develop and evaluate a Random Forest regression model, it was necessary to reduce the dataset size. A stratified sampling approach was applied to the full dataset to extract a representative subset of 50,000 samples, ensuring that the distribution of the target variable was preserved. This subset was then split into training and test sets using an 80–20 % ratio. Subsequently, the standard procedure described at the beginning of the section was followed. The corresponding results, along with the best hyperparameters obtained from the grid search, are reported in Table 10.

Additionally, the 44th decision tree from the ensemble is visualized in Fig. 8, showing the first few hierarchical splits to illustrate how the model divides the features space. This tree serves as an example to have an insights into the structure and logic of the Random Forest in this specific dataset.

Furthermore, global feature importance [52], is extracted from the trained model and ranked to quantify the relative influence of each predictor in the splitting process, and these values are visualized in Fig. 9. From the plot, it is possible to observe that the Torque (*Input_T* in the dataset) is by far the most influential predictor, contributing to over 50 % of the total decision making process in the forest. This suggests that its variation plays a dominant role in the model's output. The next most important features are *dw1* and *b*, which also show substantial importance, though considerably lower than *Input_T*. Together, these top three features account for the majority of the predictive power of the model. Other features, have much smaller importance scores, indicating they contribute marginally to the prediction. A long tail of features, shows near zero importance, suggesting they have little-to-no impact on the model's performance and may be candidates for removal in future dimensionality reduction steps. This plot helps in interpreting the Random Forest model by highlighting which variables drive the predictions.

2.4.2. Random forest model training and validation on the PCA dataset

To train and evaluate a Random Forest regression model on the *PCA* transformed dataset, the same workflow was adopted as for the original dataset. Given the large size of the dataset a stratified sampling strategy was applied to extract a representative subset of 50,000 samples. This sampling preserved the distribution of the target variable and ensured that the reduced dataset remained statistically consistent with the full data. The results, along with the optimized hyperparameters, are presented in Table 11. Compared to the model trained on the original dataset, the *PCA* based model showed a slightly reduced accuracy, as the other cases showed.

Feature importance analysis was also conducted to identify which principal components contributed most to the model predictions. The ranked importance values are shown in Fig. 10. This finding suggests that PC10 strongly influenced the splits during the model training, this is quite obvious since the results found before in Fig. 9, in which the torque has a large influenced on the splitting and since in the tenth components the torque is near to the unit (Table 2). Physically, since load condition is a key mechanical driver of root stress, its strong presence within PC10 likely explains the component's dominant role. The remaining components contribute progressively less, highlighting that certain lower variance directions can still find critical predictive information in engineering contexts.

2.4.3. HistGradBoosting model training and validation on the original dataset

To compare a bagging model with a boosting model, a Histogram based Gradient Boosting Regressor (HGBR) was employed. Due to the high cost of conducting an exhaustive grid search for the entire training set, a representative subset of 100,000 samples was chosen from the entire training data. Hyperparameter optimization was then carried out using grid search with 3-fold cross validation, evaluating combinations of maximum tree depth, $learningrate$, number of boosting iterations, ℓ_2 regularization, and minimum samples per leaf. The main training results, including the selected hyperparameters and the model's predictive performance on the test set in terms of *RMSE* and R^2 are summarized in Table 12. The high value of R^2 may also be a sign of initial overfitting. Although similar performance was observed across different hyperparameter configurations, which suggests a certain stability in the model's behavior, overfitting cannot be completely excluded. The most reliable way to assess the model's generalization capability is to evaluate it on completely new, unseen data. The external validation will be performed and discussed in Section 3.

To interpret the model, permutation feature importance [49] was computed on the test set. This method assesses feature relevance by measuring the increase in prediction error when a single feature values are randomly shuffled, thereby breaking its relationship with the target

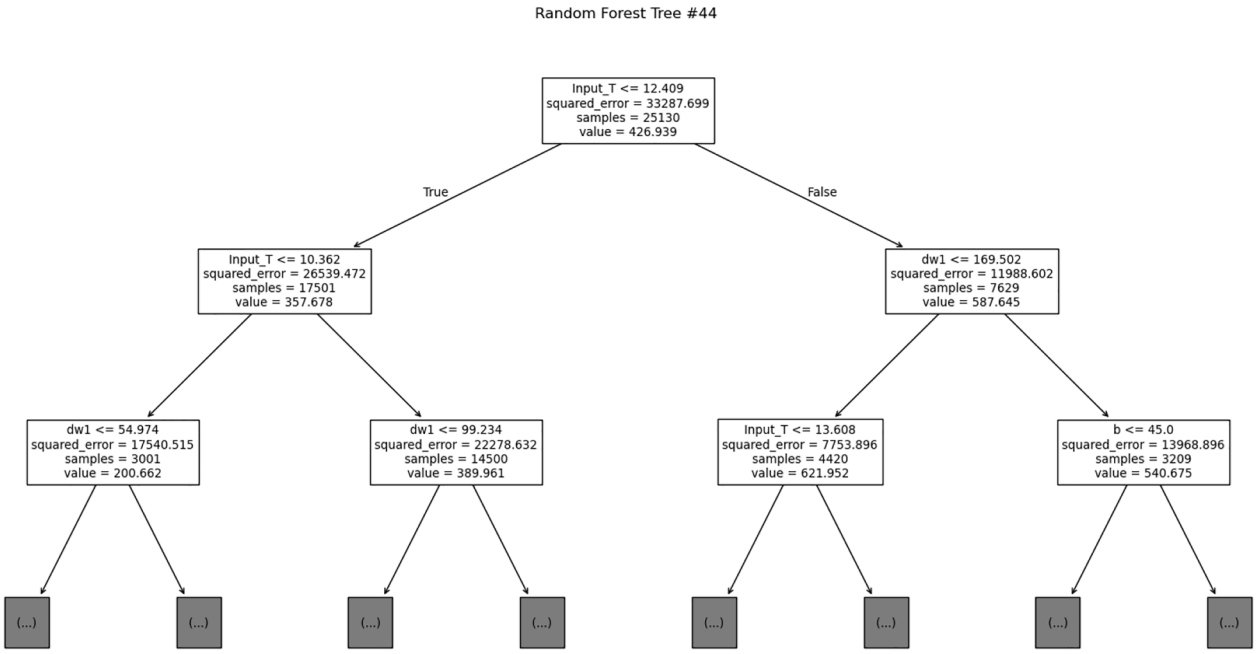


Fig. 8. Random forest 44 th tree.

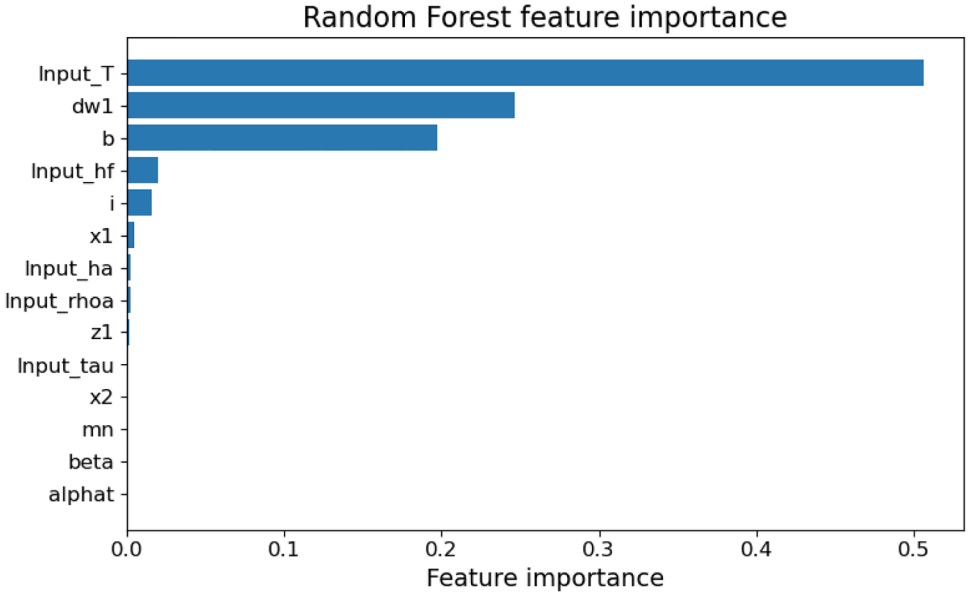


Fig. 9. Random forest feature importance.

variable. The results, shown in Fig. 11, highlight *Input_T*, *dw1*, and *b* as the most influential predictors, consistent with the findings from the Random Forest analysis. However, unlike Random Forests, which measure feature importance based on how much each feature helps to split the data during training, permutation importance looks at how much each feature actually affects the model's predictions after training. This makes permutation importance more directly related to how well the model will perform on new, unseen data.

2.4.4. HistGradBoosting model training and validation on the PCA dataset

HGBR was trained using the same workflow adopted for the original feature space, the parameters and the results on the test dataset are listed in Table 13.

The model metrics are very similar to those obtained on the original dataset, suggesting the need for further analysis to confirm generalization. Feature importance was further assessed using permutation importance on the test set. As shown in Fig. 12, PC10 emerged as the most influential component, followed by PC5 and PC3, this confirms

Table 13
Hyperparameters and results of HistGradBoosting on the *PCA* dataset.

Max depth	10
Max iter	300
Min sample leaf	100
Learning rate η	0.1
Training time (on 100,000 samples) [s]	42.3
Test *RMSE* [MPa]	23.64
Test R^2	0.98

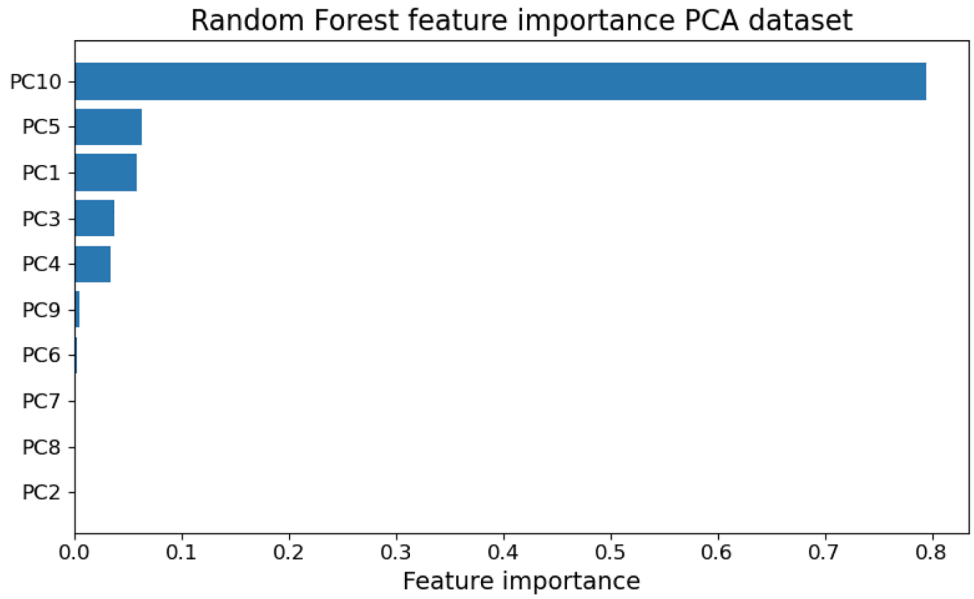


Fig. 10. Random forest feature importance on *PCA* transformed dataset.

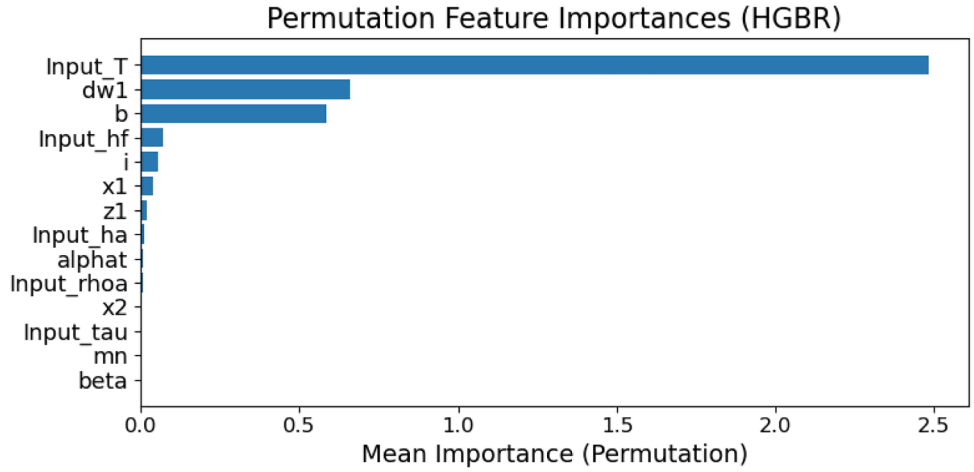


Fig. 11. HGBR Permutation feature importance on the original dataset.

previous findings that certain, lower variance principal components can retain highly predictive signals, particularly those strongly associated with torque and aligns with the results of Random Forest analysis.

2.5. Neural networks (NNs) regression model

Neural Networks are composed of layers of interconnected units called neurons, where each neuron receives input values, applies a weighted sum, passes the result through an activation function, and sends the output to the next layer. Although powerful, *NNs* require careful tuning of multiple hyperparameters (e.g., learning rate, number of layers, batch size, number of epochs) and are sensitive to overfitting without appropriate regularization. To explore the performance of neural networks in the regression task, two implementations were adopted: one using *scikit-learn MLPRegressor* and the other using a custom model built with *TensorFlow* and *Keras*. By using both frameworks, it is possible to compare high-level, user-friendly interfaces that are suitable for rapid experimentation (*MLPRegressor*), and a more flexible deep learning environment (*Keras*) that offers more control over model architecture, regularization, and optimization.

2.5.1. NNs model training and validation on the original dataset - MLPRegressor

To evaluate the suitability of neural networks for predicting the target variable σ_{H1}, a series of tests were conducted using the *MLPRegressor* class from *scikit-learn*. The standard modeling procedure described at the beginning of the section was followed, including data splitting, standardization, and hyperparameter tuning via grid search. The model selection was based on 3-fold cross validation using *RMSE* as the scoring metric.

In the second phase, the model architecture was fixed to three hidden layers with 200, 100, and 50 neurons respectively, and further attempts were conducted to compare the performance of different activation functions. Models were trained separately using *relu* and *tanh* activations to evaluate their effect on convergence and predictive performance. All models were trained with early stopping based on a validation set (20 % of the training data) to prevent overfitting. The model's performance was assessed on both training and test sets using *RMSE*, with the learning behavior visualized through loss curves over epochs. These experiments allowed us to compare the effect of architectural and training choices on model performance and generalization and served as a foundation for selecting the neural network configuration to be used in subsequent

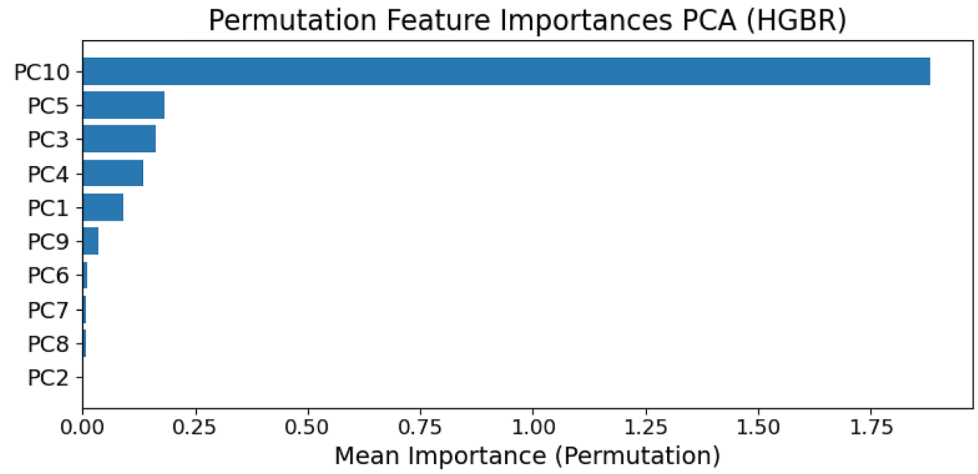


Fig. 12. HGBR Permutation feature importance on the *PCA* dataset.

Table 14
Hyperparameters and results of *NNs MLPRegressor* on the original dataset.

Hidden layer number	3
Hidden layer size	[200, 100, 50]
Regularization strength α	0.001
Initial learning rate η_0	0.001
Learning rate method (optimizer)	adam
Validation fraction	0.2
Training time [s]	560
Training *RMSE* [MPa]	3.34
Test *RMSE* [MPa]	3.43

analyses. The hyperparameters, the training settings and results of the model are summarized in Table 14.

Fig. 13 shows the loss curve of the model built using the *scikit-learn* environment. The plot illustrates the evolution of both the mean squared error (*MSE*, left axis) and its square root (*RMSE*, right axis) over 25 training epochs, these do not correspond to the total number of epochs set initially, as early stopping was triggered. A sharp reduction in error is observed within the first few epochs, indicating that the model rapidly captures the underlying data structure. After approximately 10 epochs, the loss curve decreases more gradually, eventually reaching a plateau. This behavior suggests that the model has converged to a stable solution and that further training would likely yield diminishing returns. The observed loss profile confirms the model's effective convergence and supports the suitability of the selected architecture and training parameters for this dataset.

2.5.2. NNs model training and validation on the original dataset - TensorFlow keras

As in the latter case, an optimization process was carried out to identify the most effective neural network architecture for predicting the target variable. *Keras* tuner was used with a random search strategy over a defined hyperparameter space. The search area included:

- The number of units in the input layer (ranging from 32 to 256 in steps of 32),
- The number of hidden layers (1 to 3)
- The number of units in each hidden layer (ranging from 32 to 256 in steps of 32),
- ℓ_2 regularization terms
- Learning rate for the Adam optimizer
- Batch size

The configuration that was most effective resulted in a *RMSE* of 10.08. Its architecture, which will be included among the three selected for further evaluation, is detailed below. Additional architectures were examined to evaluate the effectiveness of allocating more neurons in the second hidden layer than in the first. All the architectures explored, along with their corresponding hyperparameters, are reported in Table 15.

A limit of 100 training epochs was selected after observing that the loss function decreased very slowly beyond this point, indicating that the model had reached a performance plateau. While training was monitored using *MSE*, the root mean squared error (*RMSE*) was reported for evaluation, as it shares the same units as the target variable and is more interpretable in the context of regression tasks. Extending the training further would likely yield minimal improvement while increasing the risk of overfitting to the training data. As shown in the *RMSE* curves, all models exhibit rapid error reduction during the initial epochs, followed by a gradual convergence to stable values. Among the tested architectures, the configuration with 3 layers, called model 1 in Table 14, originally identified as optimal by the hyperparameter tuning process, achieved the lowest final error, confirming its better predictive capability. In contrast, although stable, the other configurations converge to slightly higher error values, reflecting lower accuracy. The results confirm that the additional depth lead to measurable improvements in predictive precision. The *RMSE* over epochs curves of the three *NNs* are shown in Fig. 14.

2.5.3. NNs model training and validation on the PCA dataset

To assess the suitability of neural networks for the *PCA* transformed dataset, the same two modeling strategies of the original dataset were employed. The first used the *MLPRegressor* from *scikit-learn* and the second used the *Keras Tuner* with a random search strategy to explore a broader space of configurations, including the number of hidden layers, neurons per layer, ℓ_2 regularization, learning rate, and batch size.

In this section, only the final selected architecture, corresponding to the best performing model, is presented, along with its training and validation loss curve. The full set of evaluated configurations is omitted for brevity.

The hyperparameters and the results of the *MLPRegressor* are summarized in Table 16.

Fig. 15 shows the loss curve of the model generated using the *scikit-learn* environment. The plot illustrates the evolution of *MSE* (left axis) and *RMSE* (right axis) over the course of training epochs. The limit of 100 epochs was not reached, as early stopping was triggered around the 35th epoch. A sharp decline in loss can be observed during the initial epochs, as in the normal dataset case, which indicates that the model quickly learned the underlying data structure.

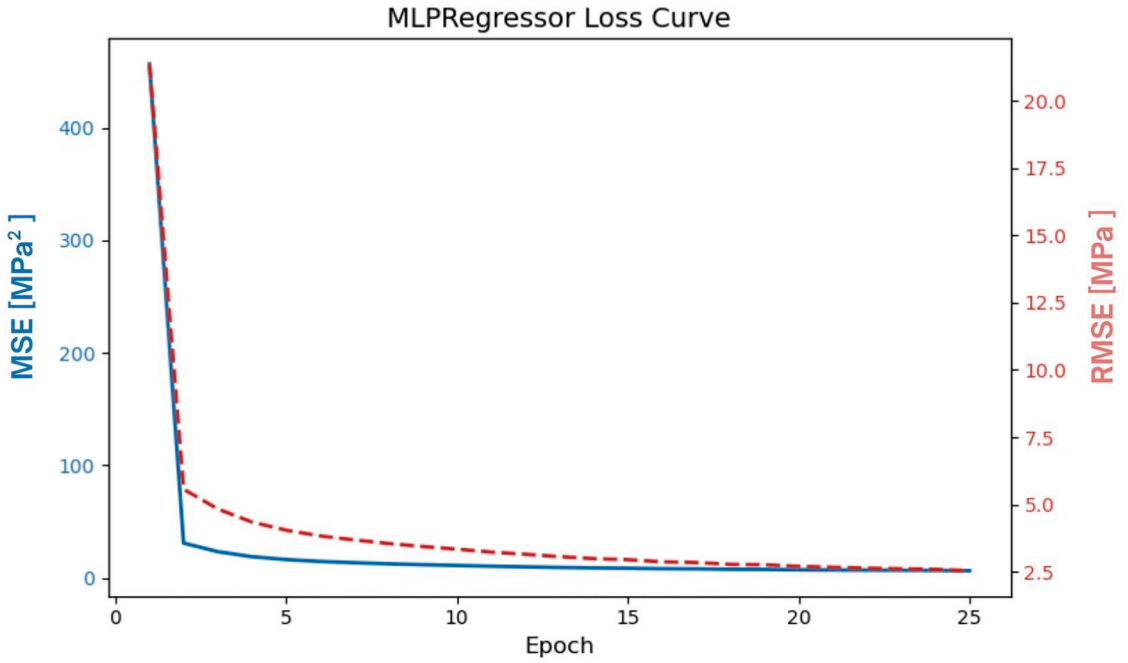
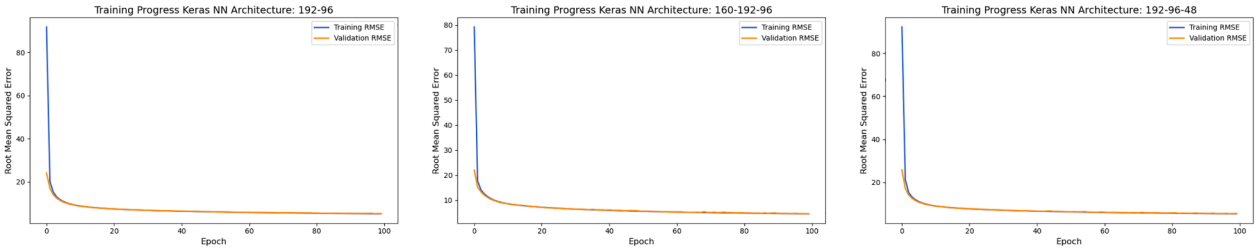


Fig. 13. Train MSE and *RMSE* across epochs for *MLPRegressor* 200-100-50 architecture - original dataset.



(a) 192-96 (b) 160-192-96 (c) 192-96-48

Fig. 14. Training and validation *RMSE* across epochs for three different *NN* architectures.

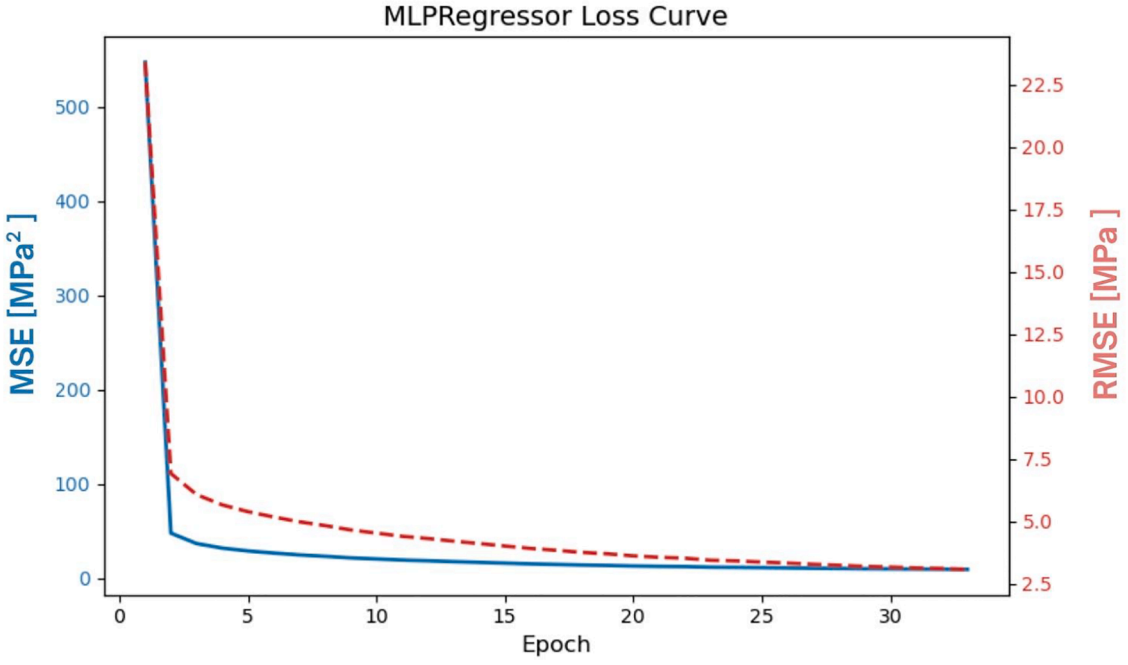
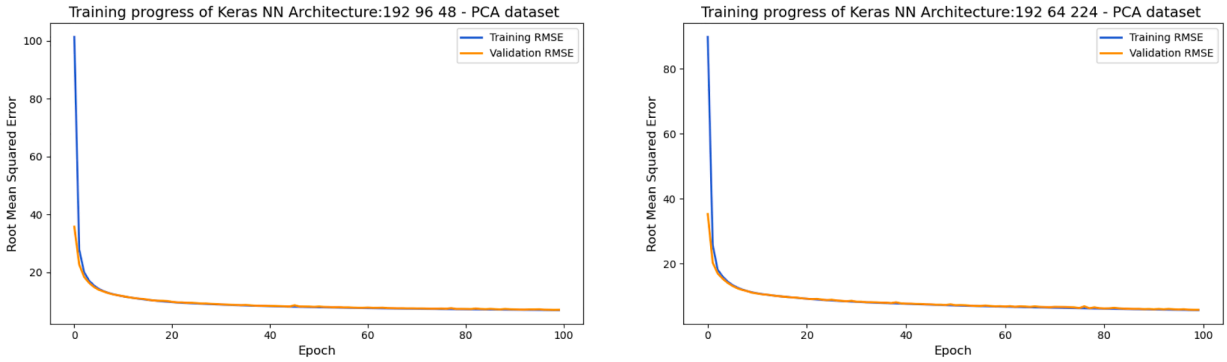


Fig. 15. Train MSE and *RMSE* across epochs for *MLPRegressor* 200-100-50 architecture on the *PCA* dataset.



(a) 192–96–48 (PCA dataset) (b) 192–64–224 (PCA dataset)

Fig. 16. Training and validation *RMSE* across epochs for two neural network architectures on the *PCA*-transformed dataset.

Table 15
Summary of the three tested neural network architectures, training times, and test *RMSE* values on the original dataset.

Model	Hidden Layers	Neurons per Layer	ℓ_2	Learning Rate	Batch Size	Test *RMSE* [MPa]	Training Time [s]
1	3	160, 192, 96	1×10^{-4}	5×10^{-5}	128	4.63	2880
2	3	192, 96, 48	1×10^{-4}	5×10^{-5}	128	5.39	2560
3	2	192, 96	1×10^{-4}	1×10^{-4}	128	5.34	2240

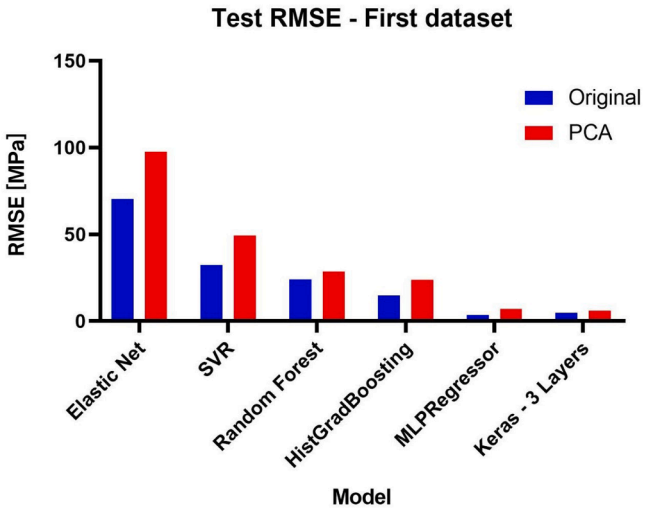


Fig. 17. Test *RMSE* results of surrogate models on the first dataset.

Table 16
Hyperparameters and results of *NNs MLPRegressor* on the *PCA* transformed dataset.

Hidden layer number	3
Hidden layer size	[200, 100, 50]
Regularization strength α	0.001
Initial learning rate η_0	0.01
Learning rate method (optimizer)	Adam
Validation fraction	0.2
Training time [s]	770
Training *RMSE* [MPa]	7.01
Test *RMSE* [MPa]	6.96

Instead, the results obtained from the *Keras Tuner* framework highlighted two *NNs* architectures that yielded the best performance. The details are reported in Table 17.

The loss curves for the training and validation phases of these two architectures are illustrated in Fig. 16, both models exhibit a similar convergence behavior.

2.6. Summary of results on the first dataset: original and PCA transformed features

This section summarizes the results of all trained regression models and allows for direct comparison of performance between models trained on the original feature dataset and those trained on the *PCA* transformed dataset. The evaluation is based on the test set of the first dataset. Aggregating test *RMSE* values in a single plot can enable visualization of relative performance differences and assessment of the impact of dimensionality reduction through *PCA*, it is evident that using the original dataset performs better than the *PCA* transformed one, as demonstrated in Figure 17. It is possible that the principal components do not fully capture the predictive structure of the input space, likely due to the loss of information relevant to the target variable. Among all models, the neural network with three layers achieves the lowest *RMSE*, demonstrating strong predictive capability and robustness. In contrast, linear models such as *Elastic Net* and *SVR* show significantly higher error, especially when trained on the *PCA* dataset, indicating a sensitivity to the transformed feature representation. Tree based models like *Random Forest* and *Histogram based Gradient Boosting* perform reasonably well, but they suffer from a decline in performance when used with *PCA*. The original features for this regression task are a more informative basis for learning, and *PCA* transformation may hinder performance, as confirmed by these results.

In the original dataset, the relative precision of the neural network was further assessed through statistical testing. The paired *t*-test [53] was applied to the error distributions coming from 10 fold cross-validation scheme, in order to determine whether the observed performance differences with respect to alternative models were statistically significant. In addition to linear, kernel based, and ensemble approaches, XGBoost was also included as a representative boosting method. This algorithm was evaluated following the same procedure adopted for the other models. However, since its performance did not exceed that of *HistGradientBoosting*, the latter was retained as the main ensemble method for comparison. As a representative of the neural network family, the multilayer perceptron (*MLP*) regressor was adopted as the benchmark model, given that it consistently achieved the best predictive performance. The same analysis was also carried out on the *PCA* transformed dataset. The results are presented separately in Table 18 for the original dataset and in Table 19 for the *PCA* transformed

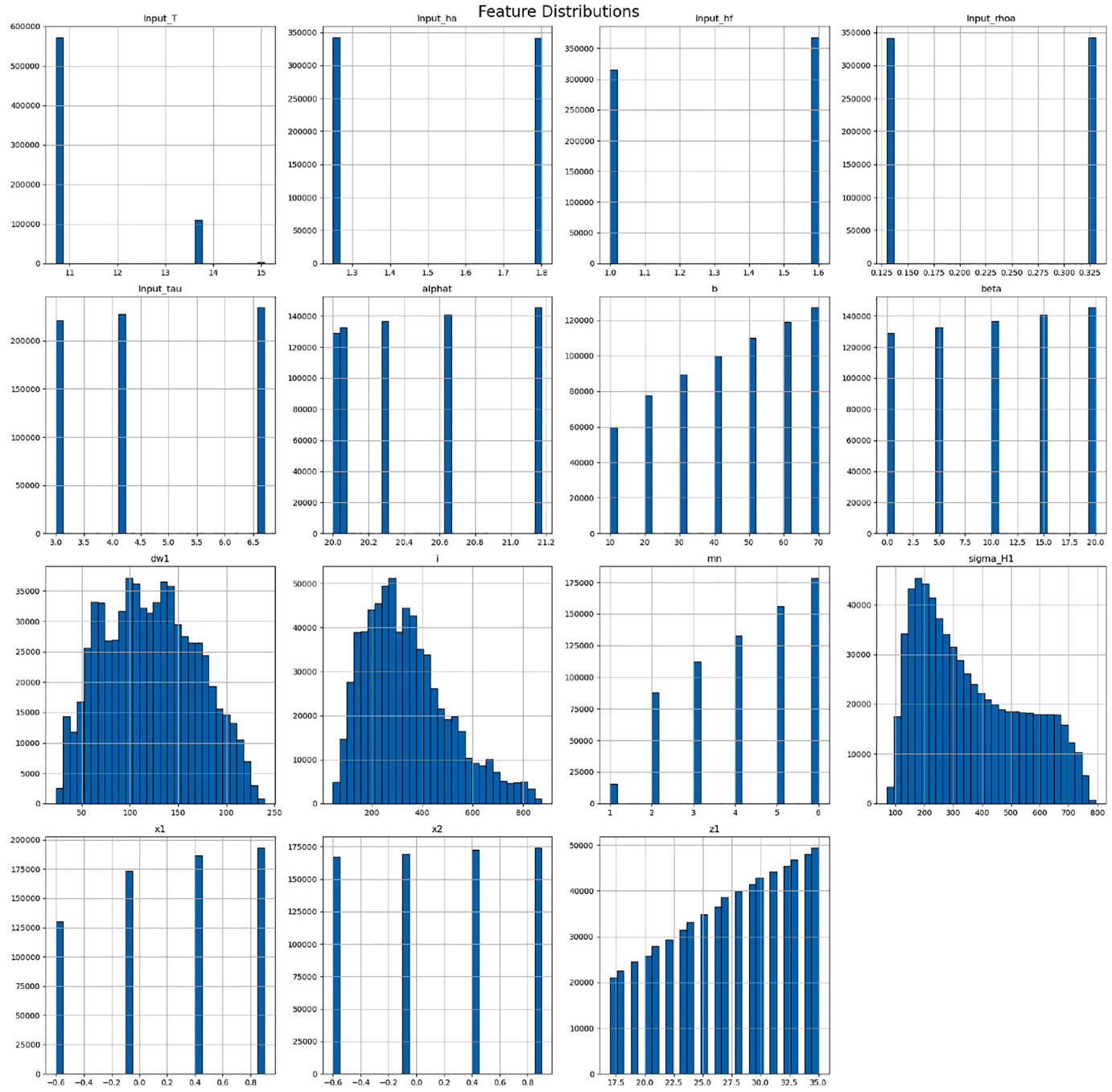


Fig. 18. New dataset after data transformation on the skewed variables.

Table 17
Summary of the tested neural network architectures, training times, and test *RMSE* values on the *PCA*-transformed dataset.

Model	Hidden Layers	Neurons per Layer	ℓ_2	Learning Rate	Batch Size	Test *RMSE* [MPa]	Training Time [s]
1	3	192, 96, 48	0	5×10^{-5}	128	6.95	2300
2	3	192, 64, 224	0	5×10^{-5}	128	5.68	2400

dataset, where mean RMSE values together with the test statistic (t) are reported.

In the original dataset, the validation results highlight the superior predictive performance of the neural network compared to all alternative models. ElasticNet-SGD performed poorly, confirming the limitations of linear methods for capturing the nonlinear relationships in the dataset. Kernel-based regression (*SVR*) and tree ensembles (*Random Forest*) improved accuracy but remained clearly inferior to boosting techniques. Both *HistGradientBoosting* and *XGBoost* achieved comparable performance, with similar mean *RMSE* values, confirming their competitiveness within the ensemble family. Nevertheless, the neural network consistently outperformed all models, achieving the lowest error and reduced variability. The same analysis was also conducted on the PCA transformed dataset and the same behavior has been found.

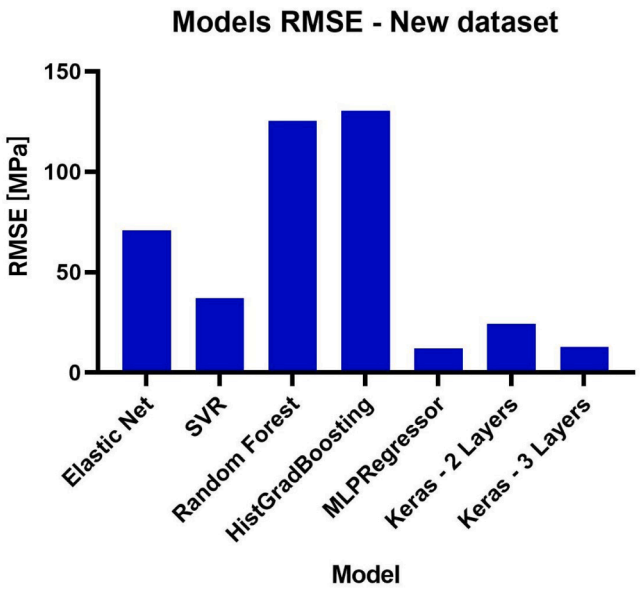


Fig. 19. *ML* models *RMSE* on the new dataset - Original features.

Table 18
Mean RMSE (10 fold CV) and paired *t*-test statistics comparing each model to the neural network (MLP) on the original dataset.

Model	Mean RMSE	*t*-statistic
ElasticNet_SDG	70.01 ± 0.37	−316.823
HistGradientBoosting	15.76 ± 0.57	−46.040
RandomForest	25.09 ± 0.53	−83.833
SVR	24.36 ± 0.65	−65.510
XGBoost	15.85 ± 0.69	−48.647
MLP	5.19 ± 0.28	–

Table 19
Mean RMSE (10 fold CV) and paired *t*-test statistics comparing each model to the neural network (MLP) on the PCA dataset.

Model	Mean RMSE	*t*-statistic
ElasticNet_SDG	81.98 ± 0.54	−331.450
HistGradientBoosting	24.87 ± 0.33	−112.191
RandomForest	32.28 ± 0.34	−134.970
SVR	46.37 ± 0.31	−191.426
XGBoost	24.90 ± 0.34	−115.054
MLP	5.77 ± 0.42	–

3. Results on a new dataset

The aim of this project is to create a surrogate model that can be used during the dimensioning phase of the gears design, but the performance metrics obtained during the training and test phases should not be completely indicative of the model's practical utility. To address this and test the robustness and effectiveness of the proposed approach, an entirely new dataset has been generated by altering the values of the input features, with particular attention given to those identified as most influential during model training. In this section, in addition to the dataset, the results coming from the use of the surrogate model trained and validated early on the new data will be shown as a more practical testing phase.

3.1. New dataset structures

The new dataset is again a series of results of verified dimensioning parameters of a gear couple in order to obtain the Hertzian contact stress σ_{H1} given a set of geometric and load inputs. This new dataset is made of 650,000 samples for a total of 14 features and 1 target variables. The main changes from the original one are in the torque distribution, as one of the most important feature found during the training phase of the models, however, also others geometric input has been changed. To ensure good replicability during the model training phases, the same logarithmic transformation has been applied to *Input_T* to adjust its skew behavior. The distribution of features and target variables in the new dataset is visible in 18.

Table 20
Model performance metrics on the new dataset.

Model	*RMSE* [MPa]	*MAE* [MPa]	R^2
Elastic Net	70.97	57.42	0.84
SVR	36.79	25.15	0.95
Random Forest	125.27	111.94	0.52
HistGradBoosting	130.47	118.01	0.48
MLPRegressor	12.04	8.58	0.99
Keras-2Layers	24.27	18.79	0.98
Keras-3Layers	12.63	8.83	0.99

3.2. Validation procedure

The validation workflow was repeated for each surrogate models trained and described in Section 2 to enable consistent performance comparison. The steps of the new validation phase are as follows:

- The new dataset, after fixed the skew variables is loaded and the target variable σ_{H1} is separated to the other features.
- A previously trained model is loaded (hyperparameters, weights and bias) and applied to the input features to produce predictions.
- Residuals are calculated as the difference between the true target values and the predicted one, an histogram of residuals is generated to inspect the error distribution.
- Standard regression metrics were computed to assess model performance (e.g. *RMSE*, R^2) .

3.3. Results of validation - original features

This section presents the results obtained using the original set of features. In Table 20 the performance (standard regression metrics) of regression models, evaluated on the new dataset, are summarized. A graphical representation of the results found is visible in Fig. 19.

For all the models used, the values found highlight that the *MLPRegressor* and *Keras* based neural networks achieved the best overall performance. The first recorded the lowest *RMSE* and *MAE*, as well as a near the unit R^2 value, which indicates highly accurate and consistent predictions. Similarly, the *Keras* models with two and three hidden layers both performed exceptionally well, with *RMSE* values below 25 MPa and coefficient of determination again close to the unit. These results confirm the suitability of deep learning approaches for modeling the nonlinear relationships present in the data. In contrast, the Elastic Net and *SVR* models achieved moderate performance. The latter in particular performed better than the Elastic Net even if less effectively than neural networks. The linear surrogate model, showed limitations in modeling complexity, as reflected by its higher *RMSE* and lower R^2 values. The ensemble models, namely Random Forest and HistGradientBoosting, performed poorly on this dataset. Both models had *RMSE* values exceeding 125 MPa and relatively low R^2 indicating a weak fit and less reliable predictions. This suggests that, under the current configuration and without additional tuning, these models may not be well suited for the specific case, the main reason should be the importance of *Input_T* as a feature, essentially changing the value of input Torque with ones never seen by the model causes poor results.

To further investigate model performance, the distribution of prediction errors was analyzed using kernel density estimation (*KDE*). It is a non parametric technique provides a smooth approximation of the

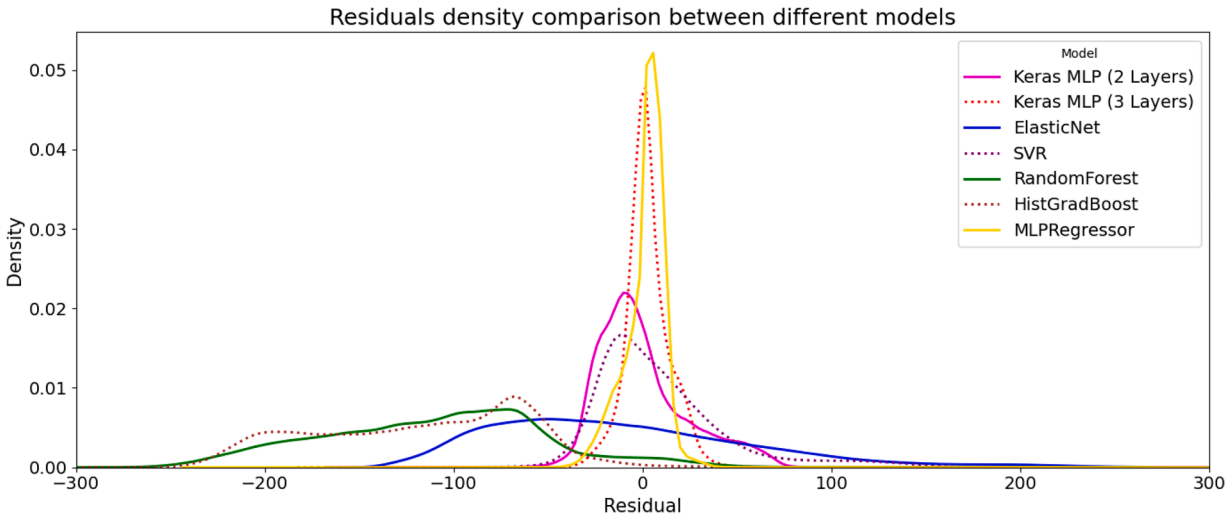


Fig. 20. Residual distributions of different models.

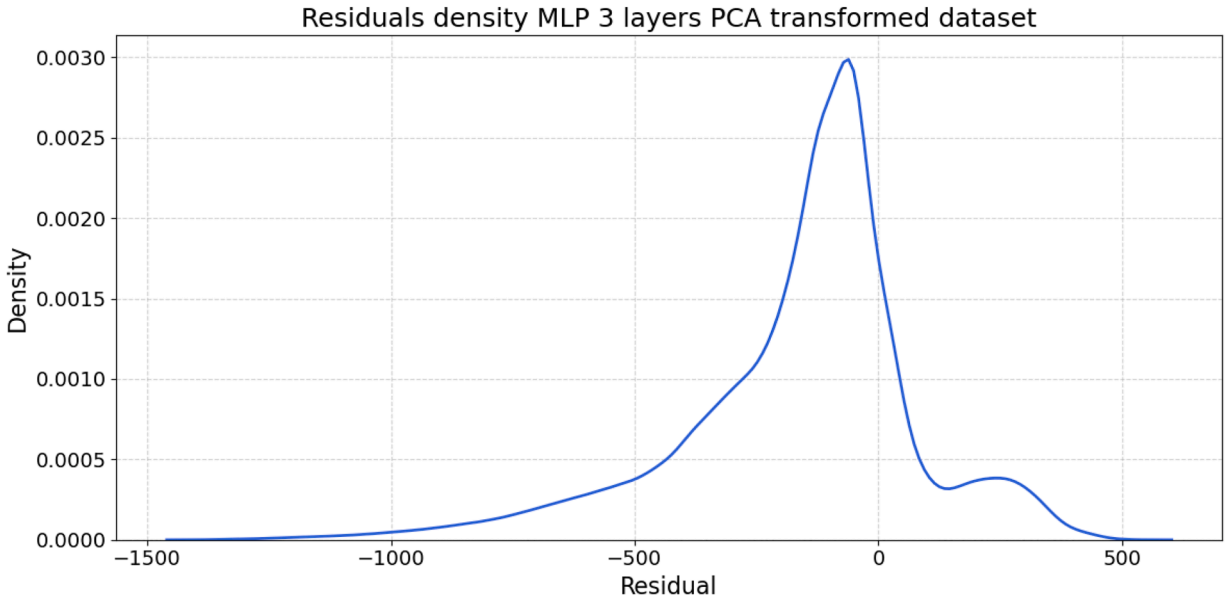


Fig. 21. Residual distributions of MLP *PCA* based model on the new dataset.

underlying probability density of the residuals without assuming a specific distributional form. Compared to histograms, KDE offers a more informative representation of the error structure, highlighting the concentration of values around zero as well as the presence of heavy tails or asymmetries. Models with narrow, symmetric KDE curves centered close to zero can be interpreted as achieving more accurate and stable predictions, whereas wider or skewed densities indicate larger variability and systematic deviations. Figure 20 illustrates the kernel density estimation (*KDE*) of residuals for the different regression models [54,55]. Residuals represent the difference between the predicted and true values of the target variable σ_{H1}, and analyzing their distribution provides insight into the accuracy and consistency of each model. Ideally, a model produces residuals tightly concentrated around zero with minimal spread, indicating both low bias and low variance. Among the models compared, the three layers Neural Networks, *Keras* (dotted red line) and the *MLPRegressor* (solid dark yellow line) exhibit the sharpest peaks centered near zero, reflecting the most accurate and consistent predictions. The *Keras NN* with two layers (dashed magenta line) also performs well, though its residuals are slightly more dispersed, indicating a marginal reduction in precision. The *SVR* model (dotted purple line) shows a moderately narrow distribution, suggesting reliable performance, though not as refined as the deep learning based models. In contrast, the ElasticNet (solid blue line), Random Forest (solid green line), and *HGBR* (solid brown line) models display broader distributions. These shapes indicate a wider spread in prediction errors, which may be attributed to either insufficient modeling capacity for the complexity of the data or a higher sensitivity to variance, as shown also by the bad values of the model metrics. Overall, the residual density analysis highlights that deeper neural architectures tend to achieve better predictive performance on this dataset, while linear or ensemble are comparatively less precise. The graph confirms the value of using advanced models to reduce error and improve generalization in complex regression tasks.

3.4. Results of validation - PCA transformed dataset

In order to validate all the models, the *PCA* based models, originally trained on the initial dataset were also tested on the new dataset. Specifically, the new data were projected onto the same principal components used during the original *PCA* transformation and predictions were made on this transformed input space. As results all *PCA* based models demonstrated poor performance in terms of both residual distributions and evaluation metrics. Fig. 21 shows the *KDE* of residuals for

the best performing based neural network model, clearly illustrating its inability to produce accurate predictions on the new data. Additional analyses involving features transformed models are omitted for brevity. This result highlights a fundamental limitation of applying fixed *PCA* transformations when the input distribution shifts between training and deployment. To obtain meaningful predictions, new models should be trained on *PCA* transformed datasets that reflect the variability of the new operating conditions, particularly with broader coverage of torque values and critical geometric parameters, so that the new components could capture sufficient variance. Nonetheless, dimensionality reduction through *PCA* remains a promising future perspective, especially in the context of gear dynamics simulations.

4. Conclusions

The study explored the application of machine learning surrogate models for the prediction of Hertzian contact stress (σ_{H1}) in involute gear pairs, with the aim of accelerating the early stage design process while maintaining analytical accuracy. A large dataset built using ISO analytical formulations was used for the initial training phase and multiple regression models were developed and evaluated using standard performance metrics (*RMSE, MAE*, R^2). The models included both linear and nonlinear approaches, ranging from Elastic Net and *SVR* to ensemble methods and deep neural networks. To investigate the influence of feature representation on model behavior, the study proceeded along two distinct branches: one using the original feature set and the other employing a *PCA* transformed version of the data.

To assess the practical utility of the model, a new dataset was generated with modified input distributions, particularly for torque values, which are one of the most influential features. Although all models were capable of replicating ISO based predictions on the training domain, only neural networks demonstrated strong generalization to the altered input space, as confirmed by the results. The *MLPRegressor* and deep *Keras* models achieved the lowest *RMSE* and highest R^2, with tightly distributed residuals centered around zero, indicating high predictive consistency and robustness to input variability.

Ensemble and linear models displayed significant performance degradation, which suggests limited extrapolation capability under unseen conditions.

The accuracy of the models based on *PCA* transformed features was not maintained on the new dataset, which highlights the limitations of fixed feature transformations when the input distribution changes. This outcome highlights the importance of building a broader and more representative dataset that varies influential features across a wider range, as one of the key directions for future work. The *PCA* transformation can maintain predictive performance under realistic variability by capturing the dominant variance modes of such a dataset

The models that were trained on *PCA* transformed features did not maintain accuracy on the new dataset, highlighting the limitations of fixed feature transformations when the input distribution changes. The construction of a broader and more representative dataset in which selected influential features are deliberately varied across a wider range is one of the key directions for future work, as highlighted in this outcome. With such a dataset, the *PCA* transformation can capture the dominant variance modes and maintain predictive performance under realistic variability

In general, the findings validate the usefulness of machine learning in creating precise and computationally efficient models that aim to predict gear contact stress. By focusing on surface fatigue mechanisms, the approach helps to make quicker and more informed decisions during early gear design stages during which the ISO standard is usually employed for preliminary fatigue verification. The usage of machine learning should not be restricted solely to stress prediction. In the future, this methodology will be extended to gear dynamics modeling and advanced optimization algorithms will be used to support multiobjective gear design, including trade-offs between strength, mass, efficiency, and dynamic behavior. Although we rely on learning directly from the outputs of the established ISO 6336 standard, new research is investigating hybrid methodologies that integrate physical laws directly into the machine learning framework. For example, physics-informed neural networks (*PINNs*) have been developed to ensure that the network's predictions remain consistent with the governing differential equations of the system being analyzed [56]. Such methods represent a promising future direction, while our work provides a practical and immediately applicable tool based on established industry standards.

CRediT authorship contribution statement

Fabio Bruzzone: Writing – review & editing, Visualization, Validation, Supervision, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Daniele Fabbri:** Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Funding acquisition, Formal analysis, Data curation, Conceptualization; **Carlo Rosso:** Writing – review & editing, Validation, Supervision, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Data availability

The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This publication is part of the project PNRR-NGEU which has received funding from the MUR – DM 117/2023.

References

[1] C.C. Aggarwal, Data Classification: Algorithms and Applications, Chapman & Hall/CRC, first ed., 2014.
[2] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: a survey, ACM Comput. Surv. (CSUR) 41 (3) (2009) 1–58. https://doi.org/10.1145/1541880.1541882
[3] S. Bansal, S. Sahoo, R. Tiwari, D.J. Bordoloi, Multiclass fault diagnosis in gears using support vector machine algorithms based on frequency domain data, Measurement 46 (9) (2013) 3469–3481. https://doi.org/10.1016/j.measurement.2013.05.015
[4] L.I. Xueyi, L.I. Jialin, Q.U. Yongzhi, H.E. David, Semi-supervised gear fault diagnosis using raw vibration signal based on deep learning, Chin. J. Aeronaut. 33 (2) (2020) 418–426. https://doi.org/10.1016/j.cja.2019.04.018
[5] A. Botchkarev, A new typology design of performance metrics to measure errors in machine learning regression algorithms, Interdiscip. J. Inf., Knowl., Manag. 14 (2019) 045–076. https://doi.org/10.28945/4184
[6] B. Haefner, M. Biehler, R. Wagner, G. Lanza, Meta-model based on artificial neural networks for tooth root stress analysis of micro-gears, Procedia CIRP 75 (2018) 155–160. https://doi.org/10.1016/j.procir.2018.04.031
[7] J.C. f. G.i. Metrology, JCGM 101: Evaluation of Measurement Data - Supplement 1 to the "Guide to the Expression of Uncertainty in Measurement" - Propagation of Distributions Using a Monte Carlo Method, Technical Report, JCGM, 2008.
[8] U. Urbas, D. Zorko, N. Vukašinović, Machine learning based nominal root stress calculation model for gears with a progressive curved path of contact, Mech. Mach. Theory 165 (2021) 104430. https://doi.org/10.1016/j.mechmachtheory.2021.104430
[9] L. Sun Hyoung, K.-P. Park, Development of a prediction model for the gear whine noise of transmission using machine learning, Int. J. Precis. Eng. Manuf. 24 (2023). https://doi.org/10.1007/s12541-023-00845-0
[10] M. Willecke, J. Brimmers, C. Brecher, Surrogate model based prediction of transmission error characteristics based on generalized topography deviations, Forsch. Ingenieurwes. 87 (1) (2023) 431–440. https://doi.org/10.1007/s10010-023-00647-w
[11] J. Henser, C. Gorgels, P. Kauffmann, T. Röthlingshöfer, A. Flodin, C. Brecher, ZaKo3D – simulation possibilities for PM gears, 2010. https://api.semanticscholar.org/CorpusID:208106560.
[12] C. Papalexis, E. Sakaridis, K. Terpos, C. Kalligeros, A. Tsolakis, V. Spitas, Neural network surrogates for finite element models in loaded tooth contact analysis of polymeric gears, Mech. Mach. Theory 214 (2025) 106127. https://doi.org/10.1016/j.mechmachtheory.2025.106127

[13] Z. Li, Z. Ma, Y. Liu, W. Teng, R. Jiang, Crack fault detection for a gearbox using discrete wavelet transform and an adaptive resonance theory neural network, Strojniški vestnik - J. Mech. Eng. 61 (1) (2015) 63–73. https://doi.org/10.5545/sv-jme.2014.1769
[14] A. Kumar, C.P. Gandhi, Y. Zhou, R. Kumar, J. Xiang, Latest developments in gear defect diagnosis and prognosis: a review, Measurement 158 (2020) 107735. https://doi.org/10.1016/j.measurement.2020.107735
[15] X. Li, J. Li, Y. Qu, D. He, Semi-supervised gear fault diagnosis using raw vibration signal based on deep learning, Chin. J. Aeronaut. 33 (2) (2020) 418–426. https://doi.org/10.1016/j.cja.2019.04.018
[16] D.J. Bordoloi, R. Tiwari, Optimum multi-fault classification of gears with integration of evolutionary and SVM algorithms, Mech. Mach. Theory 73 (2014) 49–60. https://doi.org/10.1016/j.mechmachtheory.2013.10.006
[17] W. Li, W. Lin, J. Yu, Predicting contact characteristics for helical gear using support vector machine, Neurocomputing 174 (2016) 1156–1161. https://doi.org/10.1016/j.neucom.2015.09.100
[18] D. Wang, K-nearest neighbors based methods for identification of different gear crack levels under different motor speeds and loads: revisited, Mech. Syst. Signal Process. 66–67 (2016) 576–589. https://doi.org/10.1016/j.ymssp.2015.10.007
[19] M. Cerrada, G. Zurita, D. Cabrera, R.-V. Sánchez, M. Artés, C. Li, Fault diagnosis in spur gears based on genetic algorithm and random forest, Mech. Syst. Signal Process. 66–67 (2016) 812–828. https://doi.org/10.1016/j.ymssp.2015.08.030
[20] A. Artoni, A methodology for simulation-based, multiobjective gear design optimization, Mech. Mach. Theory 133 (2019) 95–111. https://doi.org/10.1016/j.mechmachtheory.2018.11.013
[21] G. Bonori, M. Barbieri, F. Pellicano, Optimum profile modifications of spur gears by means of genetic algorithms, J. Sound Vib. 313 (3–5) (2008) 603–616. https://doi.org/10.1016/j.jsv.2007.12.013
[22] R. Tuirán, H. Maury, H. Águila, J. Oñate, H. Álvarez, Hybrid dynamic model of spur gears under combined misalignments, Adv. Mech. Eng. 17 (9) (2025). https://doi.org/10.1177/16878132251380302
[23] R. de Paula Monteiro, A. Marra, R. Vidoni, C. Garcia, F. Concli, A hybrid finite element method–analytical model for classifying the effects of cracks on gear train systems using artificial neural networks, Appl. Sci. 12 (2022) 7814. https://doi.org/10.3390/app12157814
[24] Y. Yang, X. Wang, J. Li, R. Ge, A data-physic driven method for gear fault diagnosis using PINN and pseudo-dynamic features, Measurement 236 (2024) 115124. https://doi.org/10.1016/j.measurement.2024.115124
[25] H. Bolandi, G. Sreekumar, X. Li, N. Lajnef, V.N. Boddeti, Physics Informed Neural Network for Dynamic Stress Prediction, 2022, https://arxiv.org/abs/2211.16190. 2211.16190
[26] T. Sahin, M. von Danwitz, A. Popp, Solving forward and inverse problems of contact mechanics using physics-informed neural networks, Adv. Model. Simul. Eng. Sci. 11 (1) (2024) 11. https://doi.org/10.1186/s40323-024-00265-3
[27] A.I.J. Forrester, A.J. Keane, Recent advances in surrogate-based optimization, Prog. Aerosp. Sci. 45 (1–3) (2009) 50–79. https://doi.org/10.1016/j.paerosci.2008.11.001
[28] D. Hartmann, M. Herz, U. Wever, Model order reduction a key technology for digital twins, in: Digital Twin: The Vision and the Future, Springer, Cham, 2018, pp. 167–179. https://doi.org/10.1007/978-3-319-75319-5-8
[29] Calculation of load capacity of spur and helical gears, 2006, (ISO 6336).
[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. VanderPlas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, CoRR abs/1201.0490 (2012). 1201.0490
[31] F. Chollet, et al., keras, 2015.
[32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, 2016, 1603.04467
[33] F. Bruzzone, T. Maggi, C. Marcellini, C. Rosso, C. Delprete, Proposal of a novel approach for 3D tooth contact analysis and calculation of the static transmission error in loaded gears, Procedia Struct. Integr. 24 (2019) 178–189.
[34] F. Bruzzone, T. Maggi, C. Marcellini, C. Rosso, Gear teeth deflection model for spur gears: proposal of a 3D nonlinear and non-Hertzian approach, Machines 9 (10) (2021). https://doi.org/10.3390/machines9100223
[35] C. Rosso, F. Bruzzone, On the influence of the actual load sharing factor in increasing the power density in gearboxes, in: V. Petuya, G. Quaglia, T. Parikyan, G. Carbone (Eds.), Proceedings of I4SDG Workshop 2023, Springer Nature Switzerland, Cham, 2023, pp. 420–429.
[36] F. Bruzzone, C. Rosso, Sources of excitation and models for cylindrical gear dynamics: a review, Machines 8 (3) (2020). https://doi.org/10.3390/machines8030037
[37] M.K. Cain, Z. Zhang, K.-H. Yuan, Univariate and multivariate skewness and kurtosis for measuring nonnormality: prevalence, influence and estimation, Behav. Res. Methods 49 (5) (2017) 1716–1735. https://doi.org/10.3758/s13428-016-0814-1
[38] I.-K. Yeo, R.A. Johnson, A new family of power transformations to improve normality or symmetry, Biometrika 87 (4) (2000) 954–959.
[39] N. Shrestha, Detecting multicollinearity in regression analysis, Am. J. Appl. Math. Stat. 8 (2) (2020) 39–42. https://doi.org/10.12691/ajams-8-2-1
[40] J.H. Kim, Multicollinearity and misleading statistical results, Korean J. Anesthesiol. 72 (6) (2019) 558–569. https://doi.org/10.4097/kja.19087
[41] K. Pearson, Note on regression and inheritance in the case of two parents, Proc. R. Soc. Lond. 58 (1895) 240–242.
[42] S. Marukatat, Tutorial on PCA and approximate PCA and approximate kernel PCA, Artif. Intell. Rev. 56 (6) (2023) 5445–5477. https://doi.org/10.1007/s10462-022-10297-z
[43] J. Shlens, A tutorial on principal component analysis, CoRR abs/1404.1100 (2014). 1404.1100
[44] J. Liu, G. Liang, K.D. Siegmund, J.P. Lewinger, Data integration by multi-tuning parameter elastic net regression, BMC Bioinform. 19 (1) (2018) 369. https://doi.org/10.1186/s12859-018-2401-1
[45] A.R. Nur, A.K. Jaya, S. Siswanto, Comparative analysis of ridge, LASSO, and elastic net regularization approaches in handling multicollinearity for infant mortality data in South Sulawesi, J. Mat., Stat. Komput. 20 (2) (2023) 311–319. https://doi.org/10.20956/j.v20i2.31632
[46] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Inc., USA, USA, 1995.
[47] I.H. Sarker, Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions, SN Comput. Sci. 2 (6) (2021) 420. https://doi.org/10.1007/s42979-021-00815-1
[48] D. Bhaduri, D. Toth, S.H. Holan, A review of tree-based methods for analyzing survey data, WIREs Comput. Stat. 17 (1) (2025-02).
[49] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32. https://doi.org/10.1023/A:1010933404324
[50] M. Awad, R. Khanna, Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers, Apress, Berkeley, CA, 2015. https://doi.org/10.1007/978-1-4302-5990-9
[51] H. Hang, T. Huang, Y. Cai, H. Yang, Z. Lin, Gradient Boosted Binary Histogram Ensemble for Large-scale Regression, 2021. 2106.01986
[52] A. Agarwal, A.M. Kenney, Y.S. Tan, T.M. Tang, B. Yu, Integrating Random Forests and Generalized Linear Models for Improved Accuracy and Interpretability, 2025. 2307.01932
[53] T.G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, Neural Comput. 10 (7) (1998) 1895–1923. https://doi.org/10.1162/089976698300017197
[54] B.W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman & Hall, London, London, 1986.
[55] A.Z. Zambom, R. Dias, A Review of Kernel Density Estimation with Applications to Econometrics, 2012. 1212.2812
[56] M. Parziale, L. Lomazzi, M. Giglio, F. Cadini, Physics-informed neural networks for the condition monitoring of rotating shafts, Sensors 24 (1) (2024) 207. https://doi.org/10.3390/s24010207