MDPI

Article

AI-Based Inference System for Concrete Compressive Strength: Multi-dataset Analysis of Optimized Machine Learning Algorithms

Carlos Eduardo Olvera-Mayorga[1], Manuel de Jesús López-Martínez[1,*], José Antonio Rodríguez-Rodríguez[2], Sodel Vázquez-Reyes[3,4], Luis Octavio Solis-Sánchez [4], José Ismael de la Rosa-Vargas[3], David Duarte-Correa [5], José Vidal González-Aviña[6], and Carlos A. Olvera-Olvera[1,*]

[1] Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica, Laboratorio de Invenciones Aplicadas a la Industria (LIAI), 98600, Zacatecas, México; liai@uaz.edu.mx
[2] Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería I, Laboratorio de Resistencia de Materiales y Mecánica de Suelos, 98600, Zacatecas, México
[3] Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica, 98600, Zacatecas, México
[4] Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica, Posgrados de Ingeniería y tecnología Aplicada, 98600, Zacatecas, México
[5] Tlachia Systems, Av. Felipe Carrillo Puerto 1001, Querétaro 76120, México
[6] Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias, Av. General Ramon Corona 2514, Zapopan, Jalisco, 45138, Mexico
* Correspondence: lopezmm@uaz.edu.mx; colvera@uaz.edu.mx; M.d.J.L.-M; C.A.O.-O

Abstract

The prediction of concrete compressive strength (CSMPa) is fundamental in experimental civil engineering, as it enables the optimization of mix design and complements laboratory testing through predictive tools. This study presents a systematic and reproducible methodology for comparing eight regression algorithms—including linear models, neural networks, and *boosting* methods—applied to three experimental datasets representing different production and compositional contexts. Through hyperparameter optimization using `RandomizedSearchCV` and homogeneous cross-validation, the metrics RMSE, MAE, MAPE, R^2, and nRMSE were evaluated. The *boosting* methods achieved the best performance, with CatBoost standing out by reaching R^2 values between 0.92–0.95 and RMSE between 3.4–4.4 MPa, confirming its inter-dataset stability and generalization capability. The high correlation values among performance rankings support the statistical consistency of the results. As an applied contribution, three interactive inference systems were developed in Google Colab to estimate compressive strength from mix parameters, promoting reproducibility, open access, and the potential to reduce costs and testing times associated with quality control in future practical applications.

Keywords: concrete compressive strength; machine learning; multi-dataset analysis; hyperparameter optimization; gradient boosting; neural networks; inference system

Received:
Revised:
Accepted:
Published:

Citation: Olvera, C.; López, M.; Rodríguez, J. AI-Based Inference System for Concrete Compressive Strength: Multi-dataset Analysis of Optimized Machine Learning Algorithms. *Appl. Sci.* **2025**, *1*, 0. https://doi.org/

1. Introduction

Concrete is one of the most widely used materials in construction due to its versatility, durability, and structural capacity. Among its mechanical properties, compressive strength (CSMPa) is the most relevant, as it determines the load-bearing capacity of elements such as columns, beams, and slabs [1]. Traditionally, CS determination is performed through standardized destructive tests—mainly the **ASTM C39/C39M** standard, which establishes

the test method for determining the compressive strength of cylindrical concrete specimens [2], and the **ASTM C31/C31M** standard, which regulates the preparation, curing, and handling of specimens to ensure controlled and reproducible conditions [3].

The specimens are typically cylindrical (150 mm $\times$ 300 mm), cured at a controlled temperature of $(23 \pm 2),^{\circ}$C and a relative humidity above 95,

The compression test consists of applying an increasing axial load until specimen failure, using a universal hydraulic press with automated control, a high-precision load cell, and safety mechanisms that stop the test upon detecting failure. Before load application, the cylinder surfaces are leveled through capping procedures with sulfur or other approved materials, ensuring a uniform stress distribution. Additionally, a compressometer–extensometer can be used to record axial and diametral deformation during the test, enabling the calculation of the elastic modulus and Poisson's ratio according to **ASTM C469** [7]. Together, these standards provide a unified experimental framework for correlating the elastic properties and compressive strength of concrete, and they also serve as the basis for laboratory validation of the inference system proposed in this study.

Despite the accuracy and reliability of the standardized method, this procedure involves high costs, long waiting times, and strong dependence on laboratory testing to complete the design and verification cycle. Alternative methods have been proposed to reduce these times, such as accelerated strength testing (**ASTM C684/C684M** [4]) or non-destructive tests based on ultrasonic pulse velocity and rebound hammer techniques (**ASTM C597** [5] and **ASTM C805** [6]). However, their application depends on specific experimental conditions, and their accuracy is not always comparable to standardized procedures. Consequently, engineers lack immediate feedback during the mix design stage, often waiting up to 28 days for laboratory results to validate the achieved strength and adjust mix proportions, which delays key decisions in the construction process.

In this context, machine learning (ML) has emerged as a powerful tool for estimating concrete compressive strength based on easily measurable mix and curing parameters, such as the water-to-cement ratio (W/C), aggregate proportions, or curing age. These techniques can capture complex nonlinear relationships among the components of concrete, offering a complementary alternative to traditional physical testing. Thus, ML-based models can support preliminary mix design, optimize quality control, and reduce exclusive dependence on destructive methods [9].

Since the late 1990s, artificial neural networks were the first to be successfully applied to this problem [8], laying the foundation for the use of more advanced methods such as *boosting* algorithms (CatBoost, XGBoost, and LightGBM), which have been widely adopted in civil engineering for their ability to improve accuracy and generalization in regression tasks [10,11].

Recent reviews highlight the trend toward more robust and interpretable models. Gamil [12] emphasized the importance of combining ensemble techniques with interpretability methods such as SHAP (SHapley Additive exPlanations) [13], while Zhang et al. [14] and Shaaban et al. [15] reported outstanding performance (R^2 between 0.91 and 0.94) in predicting compressive strength using *boosting*-based approaches. Complementarily, Olaye et al. [17] identified through SHAP interpretability that curing age and the water-to-cement ratio are the most influential variables affecting final strength.

Furthermore, recent studies have extended ML applications to High-Performance Concrete (HPC) and additional properties such as slump flow. Wang et al. [18] developed hybrid models optimized with metaheuristics (GWO and QPSO), achieving $R^2 > 0.99$, while Xu and Afzal [19] and Zhao et al. [20] used optimized neural network ensembles with comparable results. Nevertheless, the diversity of methodologies and validation criteria

reported in the literature makes it difficult to objectively compare model performance and limits their practical applicability in real-world environments [22,23].

Therefore, there is a clear need to establish a homogeneous, systematic, and reproducible methodological framework to compare the performance of different machine learning algorithms in predicting concrete compressive strength under equivalent experimental conditions and normalized metrics. This study addresses this need through the development of a comprehensive comparative methodology aimed not only at the objective evaluation of models but also at the construction of an inference system capable of estimating concrete strength from its mix variables, facilitating its practical application in mix design and quality control.

A comparative synthesis of the state of the art is presented in Table 1, summarizing the main experimental studies that applied machine learning to predict the compressive strength of concrete. The table highlights the models used, the optimization algorithms employed, the characteristics of the datasets, the reported performance metrics, and the main limitations identified in each case.

Table 1. Comparative summary of experimental studies applying machine learning to predict concrete properties (only studies cited in the Introduction).

Ref.	Models	Optimization algorithms	Data	Predicted properties	Performance metrics	Limitations
[14]	DeepForest (ensemble of 12 regression models)	Internal optimization (DeepForest framework)	200 HPC mixes	CS (MPa)	$R^2 = 0.91$	Small dataset; no external validation
[15]	XGBoost, Random Forest, SVR, ANN, Linear Regression, Ridge, Lasso, KNN	Default parameters; Grid Search (basic tuning)	180 high-strength concrete mixes	CS (MPa)	$R^2 \approx 0.94$	Small dataset; no interpretability; no external validation
[16]	ML models combined with fuzzy logic and simulated annealing	Simulated Annealing (SA)	Experimental engineering datasets	Decision-making in civil engineering	Relative error reduction (no R^2)	High implementation complexity
[17]	Gradient Boosting, XGBoost, Random Forest, SVR, ANN, MLP, Lasso, KNN	Grid Search + k-fold cross-validation	150 experimental mixes	CS (MPa)	R^2: XGB = 0.9349, GBR = 0.9209; MAE, RMSE also reported	Small dataset; limited generalization
[18]	Gradient Boosting + GWO; T-SFIS + QPSO; DGT	Metaheuristics (GWO, QPSO)	191 HPC mixes	CS (MPa), slump flow	$R^2 = 0.998$ (CS), 0.984 (slump); RMSE = 1.226 MPa (CS), 3.233 mm (slump)	Risk of overfitting; limited extrapolation; high computational cost
[19]	Ensemble ML models (RF, XGB, GBR, ANN hybrids)	Ensemble hybridization + k-fold cross-validation	200 HPC mixes	CS (MPa)	$R^2 \approx 0.96$	Dependent on lab data; no uncertainty analysis; no external validation
[20]	RBFNN optimized with IGWO and Dragonfly	Metaheuristics (IGWO, DA)	180 HPC mixes	CS (MPa)	$R^2 = 0.96$; RMSE = 2.5 MPa	Sensitive to data quality; complex hyperparameter tuning
[21]	RF, XGB, ANN, SVR, Linear Regression, Decision Tree, KNN	Cross-validated tuning (no metaheuristics)	220 conventional concrete mixes	CS (MPa), flexural strength, slump	$R^2 \approx 0.92$	No external validation; limited generalization
This work	Linear Regression, SVR, MLP, KNN, RF, XGBoost, LightGBM, CatBoost	`RandomizedSearchCV`	1030 UCI concrete mixes	CS (MPa)	$R^2 = 0.950$; RMSE = 3.48 MPa; MAE = 2.54 MPa; MAPE = 8.61%	No uncertainty analysis (PICP); experimental validation in progress

Objective of the Study.

The main purpose of this work is to develop a **homogeneous, systematic, and reproducible methodological framework** that enables the **evaluation, comparison, and implementation of machine learning models** for predicting concrete compressive strength (CSMPa). This framework aims not only to ensure a fair comparison among algorithms but also to **establish the technical foundation for developing a practical inference system** capable of assisting engineers in the rapid and reliable estimation of strength from mix parameters.

To this end, the performance of eight supervised regression algorithms was analyzed: **Linear Regression**, **Support Vector Regression (SVR)**, **Multilayer Perceptron (MLP)**, ***k*-Nearest Neighbors (KNN)**, **Random Forest**, **XGBoost**, **LightGBM**, and **CatBoost**.

Unlike previous studies focused on a single dataset, this research trains and evaluates the models on **three independent public datasets**: **(1)** the classic *Concrete Compressive Strength* dataset from the UCI Machine Learning Repository [9]; **(2)** the dataset of normal concretes with compressive strength and slump measurements compiled by Ke and Ming [24]; and **(3)** the experimental concrete dataset from the *Indian Institute of Technology Bhubaneswar* laboratory [25]. This experimental diversity allows for assessing the **generalization capability and stability** of the models under varying mix, curing, and material conditions.

Hyperparameter optimization was carried out using `RandomizedSearchCV`, ensuring comparability of model fitting across algorithms. Evaluation was based on well-established regression metrics [26]—RMSE, MAE, MAPE, R^2, and nRMSE—which quantify both absolute accuracy and relative consistency across datasets.

Contributions.

The main contributions of this research are as follows:

1. **To propose** a standardized comparison framework that unifies the preprocessing, optimization, and cross-validation procedures of eight regression algorithms applied to compressive strength prediction.
2. **To evaluate** the inter-dataset robustness and stability of the models, identifying those with greater generalization capability under varying experimental conditions.
3. **To develop** three AI-based interactive inference systems implemented in *Google Colab*, capable of estimating concrete strength from mix parameters as a complementary tool for quality control.
4. **To propose** the future experimental validation of the most promising system (CatBoost–Yeh) under ASTM C31/C39 standards, linking model predictions with actual compression test results and assessing its practical applicability.

Overall, this work provides both methodological and applied contributions by demonstrating that artificial intelligence can be reliably integrated into the design and evaluation of concrete, offering a first step toward intelligent automation of quality control in civil engineering.

2. Materials and Methods

2.1. Research Design and Workflow

The research was conducted following a structured, systematic, and reproducible methodology comprising **seven main stages**: **(1)** collection and integration of three public experimental concrete datasets; **(2)** review and selection of the most widely used regression algorithms for predicting concrete properties, based on their frequency of use and reported performance in recent literature; **(3)** preprocessing, normalization, and statistical analysis of the input variables; **(4)** training and optimization of eight supervised regression algorithms

through random hyperparameter search (`RandomizedSearchCV`); **(5)** comparative evaluation of model performance within each dataset using standardized metrics (RMSE, MAE, MAPE, R^2, nRMSE); **(6)** stability and global ranking analysis across datasets to identify the most consistent and generalizable models; and **(7)** implementation of an interactive inference system based on the best-performing models.

Figure 1 summarizes the proposed workflow, highlighting the main stages of the study and the iterative processes applied individually to each dataset. This methodological design ensured result traceability, algorithm and dataset comparability, and the practical transfer of knowledge toward the experimental validation of the developed inference system.

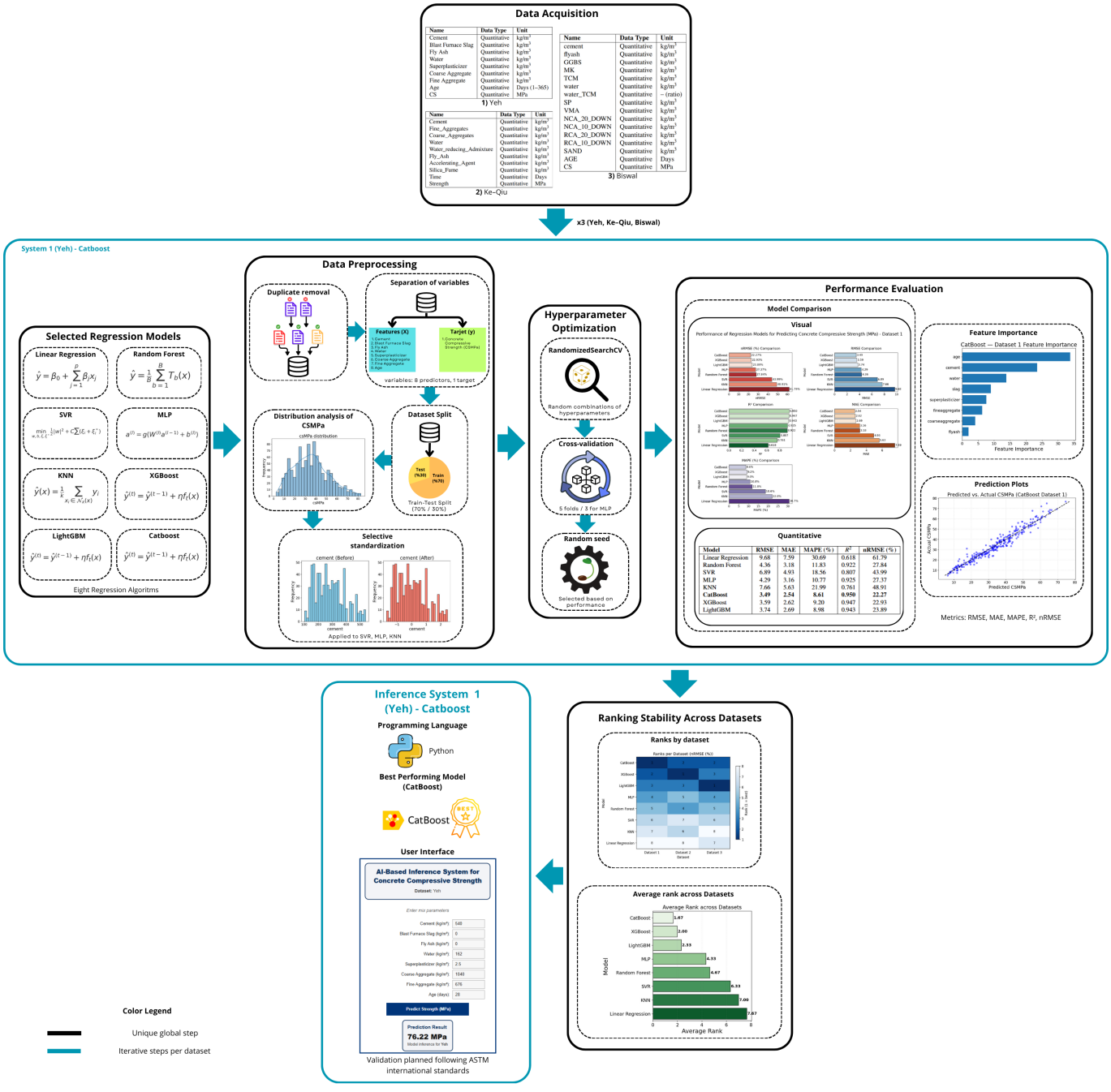

Figure 1. General diagram of the proposed methodology, composed of seven main stages: data collection, algorithm selection, preprocessing, optimized training, performance evaluation, stability analysis, and development of the inference system. The blue color indicates the iterative steps applied to each dataset, while the black color represents the global steps carried out only once in the study.

2.2. Data Acquisition

Three independent public datasets were used. In all cases, the original structures and values of the databases were preserved; only in the **Yeh (1998)** dataset were the column names simplified to facilitate the visualization of figures and tables, reducing lengthy labels such as `Cement (component 1)(kg in a m3 mixture)` to `Cement`. This modification was purely nominal and did not affect the information or data format.

2.2.1. Dataset 1: Yeh (1998)

A classic reference dataset with $n = 1030$ High-Performance Concrete (HPC) mixes [8,9]. The tests originated from an experimental program conducted at the *National Chiao Tung University* (Taiwan) and reported by **Yeh (1998)**. The mixes were prepared with ordinary Portland cement (OPC) and mineral additives (ground granulated blast-furnace slag and fly ash), under varying water-to-binder ratios and superplasticizer dosages. Curing was performed in water at 20 °C until the testing age, and compressive strength was determined on 100 × 200 mm cylindrical specimens following procedures equivalent to those established in **ASTM C39**. Although the author did not explicitly mention a curing standard, the described conditions are consistent with the standardized practices of **ASTM C31** for the preparation and curing of concrete specimens.

Eight predictive variables are recorded: `Cement`, `Blast Furnace Slag`, `Fly Ash`, `Water`, `Superplasticizer`, `Coarse Aggregate`, `Fine Aggregate`, and `Age` (days), with the target variable `Concrete Compressive Strength (CS)` expressed in MPa. The f'_c values range from **2.33–82.6 MPa**, covering concretes from low to very high strength.

The variables included in this dataset are described in Table 2.

Table 2. Variable description for **Dataset 1** (Yeh, $n = 1030$).

Name	Data Type	Unit	Description
Cement	Quantitative	kg/m^3	Mass of Portland cement per cubic meter of concrete
Blast Furnace Slag	Quantitative	kg/m^3	Mass of ground granulated blast-furnace slag
Fly Ash	Quantitative	kg/m^3	Mass of fly ash used as supplementary cementitious material
Water	Quantitative	kg/m^3	Mixing water mass per cubic meter of concrete
Superplasticizer	Quantitative	kg/m^3	High-range water-reducing admixture dosage
Coarse Aggregate	Quantitative	kg/m^3	Mass of coarse aggregate per cubic meter
Fine Aggregate	Quantitative	kg/m^3	Mass of fine aggregate (sand) per cubic meter
Age	Quantitative	Days (1–365)	Curing time at testing
CS	Quantitative	MPa	Uniaxial compressive strength measured at test age

2.2.2. Dataset 2: Ke–Qiu (2022)

This dataset was compiled from the *mix proportion ledger* of the *Guangxi Road and Bridge Group Pavement Branch* (China, November 2022) [24]. It includes $n = 1670$ mixes of conventional concrete (NC) used in pavements and highway engineering structures. Each mix was tested under real production conditions, with moist curing at controlled ambient temperature. It contains **9** predictive variables: `Cement`, `Fine_Aggregates`, `Coarse_Aggregates`, `Water`, `Water_reducing_Admixture`, `Fly_Ash`, `Accelerating_Agent`, `Silica_Fume` y `Time` (curing age in days). The target variable is `Strength` (MPa), with f'_c values ranging from

4.3 to 76.3 MPa. The repository also includes a complementary *slump* file (workability) not used in this study, but relevant for rheological behavior and consistency control analyses.

The contents of this dataset are described in Table 3.

Table 3. Variable description for **Dataset 2** (Ke–Qiu, $n = 1670$).

Name	Data Type	Unit	Description
Cement	Quantitative	kg/m^3	Portland cement dosage per cubic meter
Fine_Aggregates	Quantitative	kg/m^3	Fine aggregate (sand) mass per cubic meter
Coarse_Aggregates	Quantitative	kg/m^3	Coarse aggregate mass per cubic meter
Water	Quantitative	kg/m^3	Mixing water dosage per cubic meter
Water_reducing_Admixture	Quantitative	kg/m^3	Water-reducing admixture / superplasticizer dosage
Fly_Ash	Quantitative	kg/m^3	Fly ash dosage used as supplementary cementitious material (SCM)
Accelerating_Agent	Quantitative	kg/m^3	Set-accelerating admixture dosage
Silica_Fume	Quantitative	kg/m^3	Silica fume dosage used as SCM
Time	Quantitative	Days	Curing time (age) at compressive testing
Strength	Quantitative	MPa	Compressive strength measured at test age

2.2.3. Dataset 3: Biswal (2022)

An experimental dataset with $n = 188$ mixes tested in the laboratory at the *Department of Civil Engineering*, IIT Bhubaneswar (India) [25,27]. It focuses on *Recycled Aggregate Concrete* (RAC), assessing its mechanical behavior through 150 mm cubic specimens molded and water-cured at ambient temperature, with compression tests performed at 7, 28, and 56 days. Although the article does not explicitly mention a reference standard, the described procedure is consistent with the standard practices defined by **ASTM C31** for curing and **ASTM C39** for determining compressive strength. The experimental program investigates the water-to-binder ratio (0.25–0.75) and the influence of supplementary cementitious materials (*Fly Ash*, *GGBS*, *Metakaolin*) combined with chemical admixtures (*Superplasticizer*, *Viscosity Modifying Agent*, *Accelerator*). The target variable was obtained at different curing ages, with f'_c values ranging from **6.0–73.76 MPa**. This dataset represents a sustainable concrete scenario, featuring a more complex composition and a larger number of experimental variables (16 predictors in total).

The variables included in this dataset are described in Table 4.

Table 4. Variable description for **Dataset 3** (Biswal, $n = 188$).

Name	Data Type	Unit	Description
cement	Quantitative	kg/m^3	Portland cement dosage per cubic meter
flyash	Quantitative	kg/m^3	Fly ash dosage used as SCM
GGBS	Quantitative	kg/m^3	Ground-granulated blast-furnace slag dosage (SCM)
MK	Quantitative	kg/m^3	Metakaolin dosage used as SCM
TCM	Quantitative	kg/m^3	Total cementitious materials (cement + SCMs) per cubic meter
water	Quantitative	kg/m^3	Mixing water mass per cubic meter
water_TCM	Quantitative	– (ratio)	Water-to-binder ratio (water / TCM)
SP	Quantitative	kg/m^3	Superplasticizer dosage
VMA	Quantitative	kg/m^3	Viscosity-modifying admixture dosage
NCA_20_DOWN	Quantitative	kg/m^3	Natural coarse aggregate < 20 mm (mass per m^3)
NCA_10_DOWN	Quantitative	kg/m^3	Natural coarse aggregate < 10 mm (mass per m^3)
RCA_20_DOWN	Quantitative	kg/m^3	Recycled coarse aggregate < 20 mm (mass per m^3)
RCA_10_DOWN	Quantitative	kg/m^3	Recycled coarse aggregate < 10 mm (mass per m^3)
SAND	Quantitative	kg/m^3	Fine aggregate (sand) mass per cubic meter
AGE	Quantitative	Days	Curing time at testing
CS	Quantitative	MPa	Compressive strength of recycled-aggregate concrete at test age

2.3. Regression Models

For the prediction of CSMPa, eight supervised regression algorithms were selected, representing different machine learning approaches—from classical linear models to ensemble techniques and neural networks. Each algorithm is briefly described below, along with its basic mathematical formulation and relevant characteristics.

- **Linear Regression**: A statistical model that assumes a linear relationship between the independent variables $X = [x_1, x_2, \ldots, x_p]$ and the target variable y (CSMPa). Its formulation is:

$$\hat{y} = \beta_0 + \sum_{j=1}^{p} \beta_j x_j + \varepsilon$$

  where β_j are the coefficients to be estimated and ε is the error term. It is easy to interpret and computationally efficient but its predictive capacity decreases when the relationship among variables is nonlinear [26].
- **Random Forest**: An ensemble method composed of B decision trees $T_b{}_{b=1}^{B}$, trained on bootstrap samples and random subsets of features. The final prediction is:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} T_b(X)$$

  It reduces variance and captures nonlinear interactions. It is robust to outliers but may require more computational resources for very large datasets [28].

- **Support Vector Regression (SVR)**: Extends support vector machines to regression, seeking a function $f(x) = \langle w, x \rangle + b$ that minimizes:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

subject to $|y_i - f(x_i)| \leq \varepsilon + \xi_i$, with $\xi_i, \xi_i^* \geq 0$. Nonlinearities are handled through kernel functions. It performs well in high-dimensional spaces, although training can be slower for large datasets [29].
- **Multilayer Perceptron (MLP)**: A *feedforward* neural network with L layers, where each neuron computes:

$$a^{(l)} = g\left(W^{(l)}a^{(l-1)} + b^{(l)}\right)$$

where $g(\cdot)$ is the activation function (e.g., ReLU or sigmoid). The output $\hat{y}$ is obtained by minimizing a loss function such as the mean squared error. It can capture highly nonlinear relationships but is sensitive to hyperparameter choices and data normalization [44].
- **K-Nearest Neighbors (KNN)**: Predicts $\hat{y}$ as the average of the k nearest observations $\mathcal{N}_k(x)$ according to a distance metric $d(\cdot,\cdot)$:

$$\hat{y} = \frac{1}{k}\sum_{x_i \in \mathcal{N}_k(x)} y_i$$

It is simple and does not require explicit training, but performance depends on the chosen distance metric, the value of k, and the scale of variables [45].
- **XGBoost**: An optimized implementation of gradient boosting that builds the model additively:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i)$$

where f_t is a regression tree and η is the learning rate. The objective function combines a differentiable loss L with a regularization term Ω that penalizes model complexity:

$$\text{Obj} = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{t=1}^{T}\Omega(f_t)$$

Each new tree is fitted to minimize the gradient of the loss function with respect to previous predictions. It incorporates L_1/L_2 regularization, efficient handling of missing values, and parallel computation [30].
- **LightGBM**: A histogram-based gradient boosting method with a leaf-wise growth strategy that enhances computational efficiency. Its general mathematical formulation is the same as XGBoost, but it differs by using histograms to accelerate split point search and a leaf-wise growth strategy that selects the leaf with the highest information gain. It also includes memory reduction techniques and support for distributed training [31].
- **CatBoost**: A gradient boosting algorithm that shares the general formulation described in XGBoost but introduces key differences in the structure and training of base models. It employs symmetric decision trees (*oblivious trees*) and an *ordered boosting* scheme that prevents the use of future data in gradient calculation, thereby reducing overfitting. It also handles categorical variables natively through *target statistics* with smoothing schemes to mitigate noise [53].

Including these models enables a representative comparison across linear, distance-based, neural network, and ensemble approaches, both with and without boosting. Subse-

quently, each algorithm was tuned through hyperparameter optimization to maximize its predictive capability.

2.4. Evaluation Metrics

The selected evaluation metrics quantify the prediction error and explanatory capability of the regression models. The root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and coefficient of determination (R^2) were considered, all of which are widely used in the literature [26]. Additionally, the normalized root mean square error (nRMSE) was included to facilitate the comparison of model performance and accuracy across the three datasets, as it expresses the error relative to the dispersion of each dataset. Let $\{(y_i, \hat{y}_i)\}_{i=1}^{n}$ be the actual and predicted values in the test set, and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ the mean of the observed values. The evaluation metrics are defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|,$$

$$\text{MAPE}(\%) = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|, \quad R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2},$$

$$\text{nRMSE}(\%) = \frac{\text{RMSE}}{\sigma_{y,\text{test}}} \times 100.$$

1. **nRMSE**. Corresponds to the normalized version of RMSE, calculated by dividing it by the standard deviation of the target variable in the test set. When expressed as a percentage, it enables direct comparison of model performance and accuracy across datasets with different scales or compressive strength ranges. This normalization approach is consistent with the concept of *Normalized Root Mean Square Error* commonly used in the literature to evaluate and compare models across different units or scales [32,33]. Lower nRMSE values indicate better relative performance and greater consistency across datasets.
2. **RMSE**. Measures the typical error in the same units as the target variable and quadratically penalizes large deviations. It is useful when it is desirable to penalize substantial errors and is standard in regression analysis [34,35]. In interpretative terms, lower RMSE values indicate better performance.
3. **MAE**. Summarizes the *average error* with a direct interpretation and lower sensitivity to outliers compared to RMSE. Several studies recommend MAE when a more robust measure of average error is required [36]. In this metric, lower values represent more accurate predictions.
4. **MAPE**. Expresses error as a percentage, facilitating comparison across ranges and communication to non-technical audiences. However, it is known that MAPE can become unstable when y_i is (near) zero; therefore, following the literature [37], terms with $y_i = 0$ (if any) are excluded to avoid division by zero. Lower MAPE values reflect lower relative error.
5. R^2. Indicates the proportion of explained variance and is widely reported as a measure of overall fit; however, it requires careful interpretation outside the linear model with intercept and may lead to misleading conclusions if used in isolation [38,39]. In this metric, higher values indicate better fit (upper-bounded by 1), whereas negative values indicate that the model performs worse than simply predicting the mean.

Together, these metrics provide a clearer and more balanced view of performance: RMSE and MAE indicate the model's deviation in absolute terms, MAPE expresses that

deviation as an easily interpretable percentage, and R^2 reflects how much of the data behavior is explained by the model. Using them jointly avoids relying on a single metric that could overlook specific weaknesses; therefore, the adopted criterion is to minimize RMSE/MAE/MAPE and maximize R^2, in accordance with common recommendations in the literature [35,37].

2.5. Data Preprocessing

Each dataset was individually inspected to ensure its integrity and consistency prior to modeling. In all cases, duplicate records were removed, and the coherence of numerical values and measurement units was verified. This preliminary cleaning ensures that the models operate only with representative information and prevents the results from being affected by redundant patterns.

In the Yeh dataset, **25** duplicate records were identified and removed, resulting in a total of **1,005** unique samples. In the Ke–Qiu dataset, **52** duplicates were removed, yielding a final total of **1,618** valid observations, while in the Biswal dataset, **3** duplicates were detected, resulting in **185** unique samples.

In all cases, this preliminary cleaning ensures that the models work exclusively with representative and redundancy-free information. Subsequently, the predictive variables (X) and the dependent variable (y) were separated, the latter corresponding to the concrete compressive strength (CSMPa). The distribution of y was evaluated using histograms with density fitting and the calculation of the skewness coefficient for each dataset. The obtained values were **0.395** for Yeh, **0.007** for Ke–Qiu, and **0.203** for Biswal, all classified as slightly or nearly symmetric. According to common statistical criteria [40–42], values close to zero indicate symmetry; between ± 0.5 and ± 1 indicate moderate skewness; and greater than ± 1 indicate high skewness. Since the three datasets showed approximately symmetric distributions, no transformations such as logarithmic, square root, or Box–Cox were applied. This decision is supported both by the statistical analysis and the visual inspection of the target variable distributions, shown in Figure 2, which presents the histograms corresponding to the three datasets: **(1)** Yeh, **(2)** Ke–Qiu, and **(3)** Biswal.

To assess the predictive performance of the models, the data from the three datasets were independently split into two subsets: **70%** for training and **30%** for testing, with a common random seed (123) to ensure experiment reproducibility.

A key aspect of the preprocessing stage was the standardization (zero mean and unit variance) of the predictive variables, applied exclusively to models sensitive to feature magnitudes: SVR, MLP, and KNN. This decision is based on the fact that certain algorithms are sensitive to variable scales, which may cause a feature with larger numerical values to exert greater influence on the result than another with smaller values, even if their true importance is the same [43].

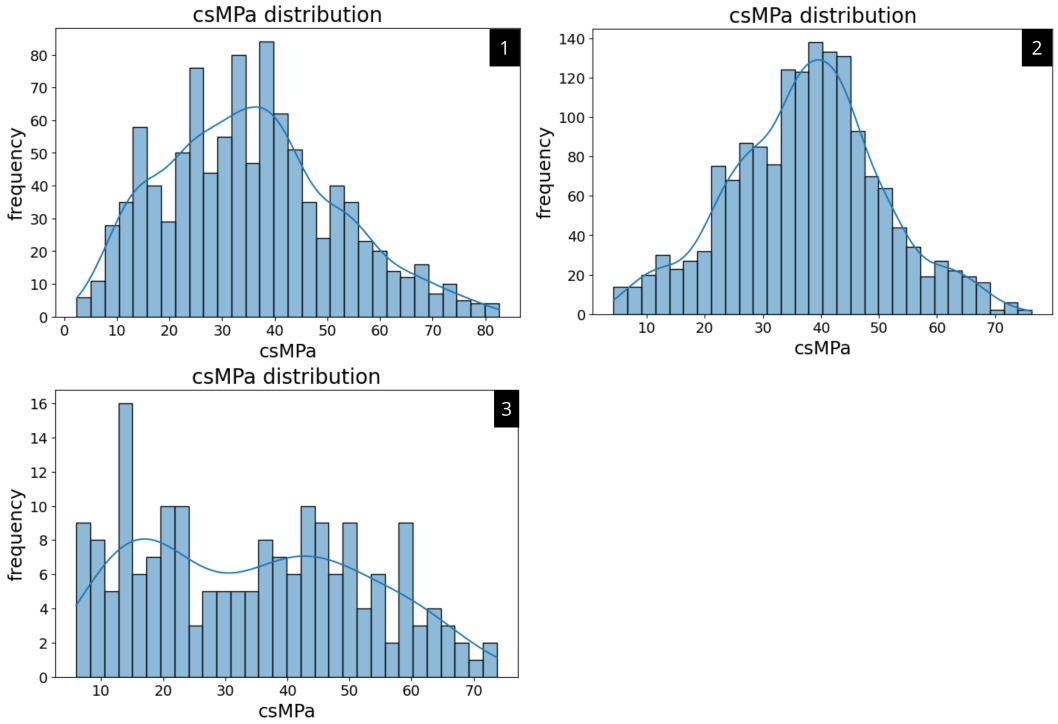

Figure 2. Distributions of the target variable (MPa) in the three datasets: **(1)** Yeh, **(2)** Ke–Qiu, and **(3)** Biswal. Approximately symmetric distributions are observed, with skewness coefficients of **0.395**, **0.007**, and **0.203**, respectively.

It is important to note that the output variable (CSMPa) was kept in its original scale, as it represents a physical magnitude whose value must be preserved to allow direct interpretation of the error metrics. Thus, the best test root mean square errors were **3.49 MPa** (Yeh), **3.88 MPa** (Ke–Qiu), and **3.76 MPa** (Biswal), values that directly reflect the magnitude of the prediction error with respect to the experimental tests. Such direct interpretation would not be possible if the target variable had been normalized.

Figure 3 illustrates, as an example, the distribution of the eight predictive variables from the Yeh dataset before and after the standardization process, showing how the transformation homogenizes magnitudes without altering the general shape of the data. The same procedure was applied identically to the Ke–Qiu and Biswal datasets.

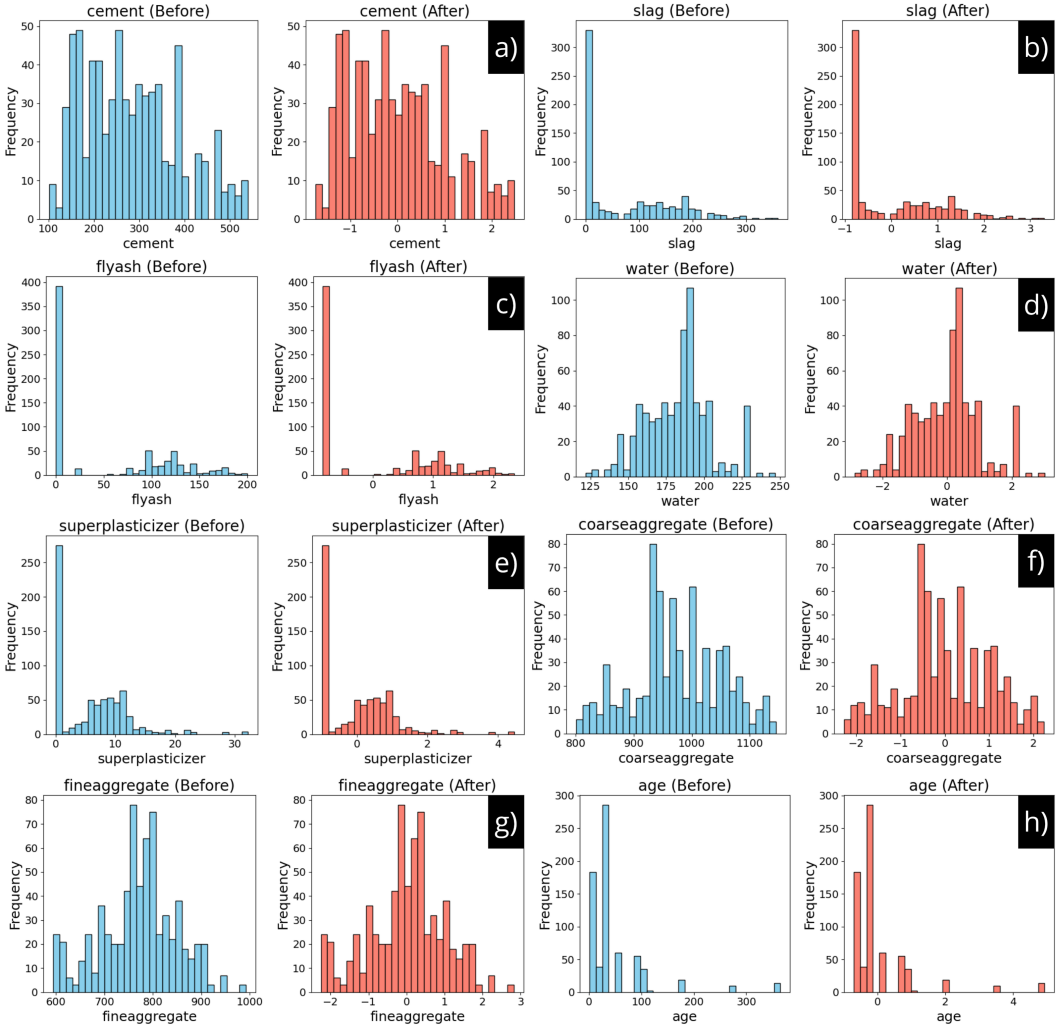

Figure 3. Distributions of the eight predictor variables of the Yeh dataset before and after standardization (zero mean and unit variance): **a)** Cement, **b)** Slag, **c)** Fly ash, **d)** Water, **e)** Superplasticizer, **f)** Coarse aggregate, **g)** Fine aggregate, and **h)** Age.

This scaling effect is particularly relevant in certain machine learning algorithms whose mathematical formulations depend directly on the magnitude of the input variables. Understanding this behavior helps justify why some models require prior normalization to achieve optimal performance. The most representative cases used in this study are described below:

- **SVR**: This method seeks to find a function that remains within an optimal tolerance margin around the actual data, maximizing the distance between the support vectors and the regression hyperplane. If the variables have very different scales, the internal metric (dot product or kernels) becomes distorted, affecting the correct placement of the margin and, consequently, the quality of the prediction [29].
- **MLP**: *Feedforward* neural networks use activation functions such as ReLU or *tanh* to transform inputs into internal signals. When input features are on very different scales, some neurons may receive excessively large or small values, causing activation function saturation and hindering gradient propagation during training, which slows down or even prevents convergence [44].
- **KNN**: This algorithm assigns predictions based on the distances between observations, typically using the Euclidean metric. If one variable has a much larger numerical range than the others, it will dominate the distance calculation, diminishing the influence of other variables that may be more relevant to the studied phenomenon [45].

In contrast, tree-based models—such as Random Forest, XGBoost, LightGBM, and CatBoost—do not require prior normalization, as they generate splits through partition rules based on absolute thresholds and are therefore invariant to the scale of the variables [26].

This selective preprocessing criterion, applied consistently across the three datasets, ensures that each algorithm receives the variables in the most suitable representation to maximize predictive performance while maintaining methodological homogeneity within the overall modeling pipeline.

2.6. Hyperparameter Optimization

Hyperparameter optimization was performed independently for each dataset using a standardized procedure with `RandomizedSearchCV`. This random search method efficiently explores large hyperparameter spaces at a lower computational cost compared to exhaustive search (*Grid Search*) [46]. The technique samples random combinations of hyperparameters from predefined distributions, evaluating their performance through k-fold stratified cross-validation. The use of this homogeneous approach across the three datasets allowed balancing search space coverage and computational time, while ensuring reproducibility by fixing random seeds.

To guarantee the reproducibility of the experiments, specific random seeds were set for each model. Although a common seed (`123`)—widely used in the scientific community and in machine learning libraries for its ease of replication—was initially adopted, preliminary tests showed that using `42`—another seed frequently employed in the literature and programming environments—provided slightly superior and more stable performance in CatBoost, XGBoost, and Random Forest. Therefore, it was adopted for these cases. For SVR, MLP, and KNN, seed variation did not have a significant effect, so `42` was also used for methodological consistency. In the case of LightGBM, the original seed (`123`) was retained, as this configuration yielded the best results. This strategy balanced methodological consistency with the representativeness of the best performances achieved by each algorithm while maintaining full experiment traceability.

Each hyperparameter combination was evaluated through 5-fold stratified cross-validation, with random shuffling of samples to reduce partition bias. For the MLP model, 3-fold cross-validation was employed due to the high computational demand associated with neural network training during hyperparameter optimization, which reduces computation time without significantly compromising the statistical validity of the evaluation.

The number of random combinations evaluated in the search (n_{iter}) was adjusted according to the dimensionality of the hyperparameter space and the computational complexity of each model, ranging from 10 to 50 iterations.

The search intervals for each hyperparameter were defined based on recommendations from specialized literature and official algorithm documentation, complemented by prior empirical experience. The same approach was applied across all three datasets to maximize performance and generalization capability while maintaining a reasonable computational cost.

The optimized hyperparameters for each model are detailed below, along with their theoretical justification, selected range, and bibliographic support.

Random Forest Regressor

- *n_estimators* [100, 200, 300, 400, 500]: number of trees in the ensemble. A higher number reduces variance and improves prediction stability, although it increases computational cost [28].

- *max_depth* [5, 10, 15, 20, None]: maximum depth of the trees. Limiting depth decreases overfitting risk by controlling model complexity [43].
- *min_samples_split* [2, 5, 10]: minimum number of samples required to split a node. Higher values create more general partitions and reduce variance.
- *min_samples_leaf* [1, 2, 4]: minimum number of samples per terminal leaf. Prevents splits based on very few samples, improving generalization [26].

Support Vector Regressor (SVR)

- *C* [0.1, 1, 10, 100]: penalty parameter controlling the balance between training fit and regularization. Smaller values increase bias; larger values reduce bias but may increase variance [29].
- *epsilon* [0.01, 0.1, 0.5, 1]: defines the tolerance region (the ε-tube) where errors are not penalized, affecting the model's sensitivity to small fluctuations [47].
- *kernel* ['linear', 'rbf']: the RBF kernel captures complex nonlinear relationships, while the linear kernel is suitable for approximately linear data structures.

Multilayer Perceptron (MLP)

- *hidden_layer_sizes* [(50,), (100,), (50,50), (100,50)]: architectures with one or two hidden layers. Low-to-moderate capacity configurations are appropriate for tabular datasets without high noise levels [48].
- *activation* ['relu', 'tanh']: common activation functions. ReLU is computationally efficient and mitigates the vanishing gradient problem, while tanh is useful when data are centered around zero [44].
- *solver* ['adam', 'lbfgs']: optimization algorithms. Adam performs well on large or noisy datasets, while LBFGS converges rapidly on small, well-conditioned datasets [49].
- *alpha* [0.0001, 0.001, 0.01]: L2 regularization coefficient that penalizes large weights, reducing overfitting risk [50].

K-Nearest Neighbors (KNN)

- *n_neighbors* [3, 5, 7, 9]: number of neighbors used. Small values may lead to overfitting; higher values smooth predictions but may increase bias [45].
- *weights* ['uniform', 'distance']: uniform weights assign equal importance to all neighbors, while distance-based weighting gives greater influence to closer neighbors, which may improve performance on heterogeneous data [51].
- *metric* ['euclidean', 'manhattan']: standard metrics for continuous data. The Manhattan metric can be more robust to outliers in certain distributions [51].

XGBoost

- *n_estimators* [100, 200, 300]: number of sequential trees in the boosting process.
- *learning_rate* [0.01, 0.05, 0.1]: controls the contribution of each tree. Lower rates favor generalization but require more iterations [30].
- *max_depth* [3, 5, 7]: controls model complexity and its ability to capture nonlinear interactions.
- *subsample* [0.6, 0.8, 1.0]: proportion of data used in each iteration, useful for reducing overfitting [52].

LightGBM

- *n_estimators*, *learning_rate*, *max_depth*: same principles as in XGBoost.
- *num_leaves* [31, 50, 100]: controls the maximum number of leaves in each tree. Higher values increase complexity and reduce bias but may increase variance [31].

- *feature_fraction* [0.6, 0.8, 1.0]: fraction of features used by each tree, helping to reduce overfitting and accelerate training [31].

CatBoost

- *iterations* [100, 300, 500]: total number of trees in the model.
- *learning_rate* [0.01, 0.05, 0.1]: defines the magnitude of the update at each iteration.
- *depth* [4, 6, 8, 10]: depth of the symmetric (*oblivious*) trees used in CatBoost [53].
- *l2_leaf_reg* [1, 3, 5, 7]: L2 regularization applied to leaf values, reducing overfitting.
- *bagging_temperature* [0.2, 0.5, 1.0]: controls the randomness of weighted sampling; lower values reduce variance at the cost of increased bias [53].

Linear Regression

Ordinary linear regression does not include regularization hyperparameters by default, so its standard implementation was used as a baseline model. This model served as a reference for evaluating the relative improvement achieved by nonlinear and more complex methods.

This optimization process was designed to fit each model to the specific characteristics of each dataset, ensuring a fair comparison under homogeneous training conditions and preventing exposure of the test set during tuning.

2.7. Inference System Implementation

As the applied component of this study, three independent interactive inference systems were developed, each based on the best-performing model for its corresponding dataset: **CatBoost** for Yeh, **XGBoost** for Ke–Qiu, and **LightGBM** for Biswal. These systems enable the estimation of compressive strength (CSMPa) from the mix parameters available in each dataset, translating the predictive models into practical tools for concrete design and evaluation.

The main function, named `launch_dataset_inference()`, was developed and executed in the cloud computing environment `Google Colab`. It is designed to dynamically generate the inference interface corresponding to each dataset. Its modular structure allows the reuse of a single code block across the three systems, changing only the input parameters: the dataset name, the pre-trained model, and the ordered list of variables (*feature configuration*) associated with it. Based on this information, the function constructs numerical input controls, collects user-provided values, organizes them into a data frame, and sends them to the corresponding model to generate a prediction. The result is immediately displayed in the interface, expressed in megapascals (MPa) and rounded to two decimal places.

In the **System 1 (Yeh)**, based on CatBoost, the user inputs eight predictive variables: cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate, and concrete age. In the **System 2 (Ke–Qiu)**, implemented with XGBoost, nine parameters are included: cement, fine and coarse aggregates, water, water-reducing admixture, fly ash, accelerating agent, silica fume, and curing time. Finally, the **System 3 (Biswal)**, built with LightGBM, considers fifteen variables related to recycled concretes and supplementary admixtures, including cement, GGBS, metakaolin, water-to-total cementitious materials ratio (w/TCM), natural and recycled aggregates, superplasticizer, VMA, sand, and age.

In general, the inference performed by each system can be represented by the following function:

$$f'_c = f(x_1, x_2, \ldots, x_n),$$

where x_i corresponds to the specific mix parameters of each dataset, and $f(\cdot)$ represents the trained predictive model. The systems can be executed directly in Google Colab from any web browser without requiring advanced technical knowledge. Their modular architecture allows future integration into web platforms or mobile devices, facilitating use in laboratory or on-site applications.

Currently, the systems are presented as *proofs of concept*. Experimental validation is planned through compressive strength testing in accordance with ASTM C39, comparing the model predictions with the values measured in the laboratory. Initially, the system corresponding to the Yeh dataset will be evaluated to verify the accuracy of the methodology before extending testing to the other datasets.

Taken together, these systems constitute a proof of concept demonstrating the feasibility of integrating machine learning models into interactive inference environments, enabling the estimation of concrete compressive strength (f'_c) from its mix parameters. Their modular and reproducible implementation in `Google Colab` facilitates adaptation to future experimental and laboratory validation applications.

The characteristics and performance of the three developed systems are summarized in Table 5, which also presents the lowest RMSE value obtained on the test set.

Table 5. Summary of the inference systems developed for each dataset, indicating the best-performing model, number of input variables, concrete type, and lowest RMSE obtained on the test set. All systems were implemented in Google Colab and are planned to be validated experimentally according to ASTM C39.

System	Dataset	Model	# Variables	Concrete Type	RMSE (MPa)
1	Yeh	CatBoost	8	Conventional / HPC	**3.49**
2	Ke–Qiu	XGBoost	9	Normal concrete	**3.88**
3	Biswal	LightGBM	15	Recycled concrete	**3.76**

3. Results

3.1. Model Performance per Dataset

The performance of the eight regression algorithms was evaluated using the RMSE, MAE, MAPE, R^2, and nRMSE metrics. This analysis was applied independently to the three datasets in order to compare both absolute accuracy and relative consistency among them. The obtained results are presented in Tables 6, 7, and 8.

Table 6. Performance metrics for **Dataset 1** (Yeh, $n = 1030$). CatBoost achieved the best overall performance, obtaining the lowest RMSE and nRMSE and the highest R^2.

Model	RMSE (MPa)	MAE (MPa)	MAPE (%)	R^2	nRMSE (%)
Linear Regression	9.68	7.59	30.69	0.618	61.79
Random Forest	4.36	3.18	11.83	0.922	27.84
SVR	6.89	4.93	18.56	0.807	43.99
MLP	4.29	3.16	10.77	0.925	27.37
KNN	7.66	5.63	21.99	0.761	48.91
CatBoost	**3.49**	**2.54**	**8.61**	**0.950**	**22.27**
XGBoost	3.59	2.62	9.20	0.947	22.93
LightGBM	3.74	2.69	8.98	0.943	23.89

Table 7. Performance metrics for **Dataset 2** (Ke–Qiu, $n = 1670$). XGBoost achieved the lowest RMSE and nRMSE, showing the best trade-off between prediction error and explanatory power.

Model	RMSE (MPa)	MAE (MPa)	MAPE (%)	R^2	nRMSE (%)
Linear Regression	6.84	4.34	21.88	0.711	53.75
Random Forest	4.44	3.07	12.55	0.878	34.89
SVR	6.00	3.47	18.01	0.778	47.12
MLP	4.88	3.34	14.17	0.853	38.35
KNN	5.38	3.41	16.03	0.822	42.23
CatBoost	4.09	2.67	10.97	0.897	32.09
XGBoost	**3.88**	**2.66**	**10.61**	**0.907**	**30.47**
LightGBM	4.18	2.76	11.18	0.892	32.83

Table 8. Performance metrics for **Dataset 3** (Biswal, $n = 188$). LightGBM achieved the best performance, with the lowest RMSE and nRMSE and the highest R^2.

Model	RMSE (MPa)	MAE (MPa)	MAPE (%)	R^2	nRMSE (%)
Linear Regression	7.39	6.13	24.01	0.843	39.61
Random Forest	6.54	5.41	24.04	0.877	35.07
SVR	7.20	5.38	22.87	0.851	38.56
MLP	6.43	4.77	18.11	0.881	34.46
KNN	8.41	6.59	26.92	0.797	45.04
CatBoost	3.90	3.01	12.73	0.956	20.89
XGBoost	4.57	3.66	13.63	0.940	24.48
LightGBM	**3.76**	**3.16**	**13.66**	**0.959**	**20.16**

Across the three datasets, a consistent performance pattern was observed among the models. In the Yeh dataset, the **CatBoost** algorithm achieved the best results (*RMSE* = **3.49 MPa**, R^2 = 0.950, nRMSE = **22.27%**), slightly outperforming XGBoost and LightGBM. In the Ke–Qiu dataset, the best-performing model was **XGBoost** (*RMSE* = 3.88 MPa, R^2 = **0.907**, nRMSE = **30.47%**), followed closely by CatBoost. Finally, in the Biswal dataset, the best performance corresponded to **LightGBM** (*RMSE* = **3.76 MPa**, R^2 = **0.959**, nRMSE = **20.16%**). Overall, these results confirm that the *gradient boosting* algorithms achieved the best balance between accuracy and generalization capability across the three analyzed scenarios.

When comparing the minimum nRMSE values obtained by the best model for each dataset, the Biswal dataset exhibited the lowest relative error (20.16%), followed by Yeh (**22.27%**) and Ke–Qiu (**30.47%**). This suggests that the predictability of the phenomenon was higher in the Biswal dataset, possibly due to lower experimental dispersion or more controlled curing conditions, whereas Ke–Qiu exhibited greater heterogeneity among its samples.

A common pattern was also identified among the lower-performing algorithms. Linear and distance-based methods (**Linear Regression**, **KNN**, and **SVR**) showed clear limitations in capturing nonlinear relationships and interactions between mix and curing variables, reflected in high nRMSE values (between **39%** and **62%**) and lower coefficients of determination. In particular, Linear Regression was the weakest model in the Yeh and Ke–Qiu datasets, while KNN showed the poorest accuracy in the Biswal dataset.

This behavior highlights the inability of linear and proximity-based approaches to adequately represent the complexity of the problem, in contrast to the robustness of *gradient boosting* models, which demonstrated greater stability and generalization capability under different experimental conditions and compressive strength ranges.

The performance results were consolidated into a single comparative visualization (Figures 4–6), integrating the outcomes obtained for the three datasets. This figure illustrates the performance of the eight regression algorithms applied to datasets **(1)** Yeh, **(2)** Ke–Qiu, and **(3)** Biswal, distinguished by numerical labels in the chart. The figure allows visualization of the joint variation in model performance across the five evaluated metrics:

nRMSE, RMSE, MAE, MAPE, and R^2. For the error metrics (nRMSE, RMSE, MAE, and MAPE), lower values indicate better model fit, whereas for the coefficient of determination (R^2), higher values reflect greater model capability to reproduce concrete compressive strength from the predictive variables.

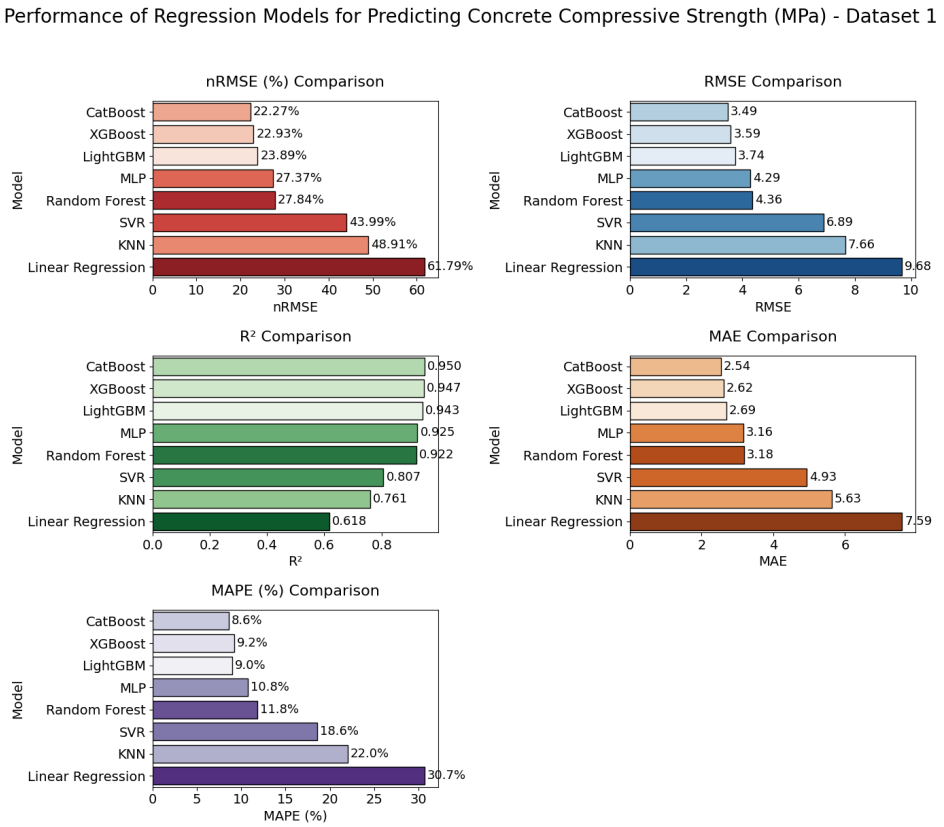

Figure 4. Global comparison of the regression models' performance for the **Yeh** dataset. The five evaluation metrics are shown: nRMSE, RMSE, MAE, MAPE, and R^2.

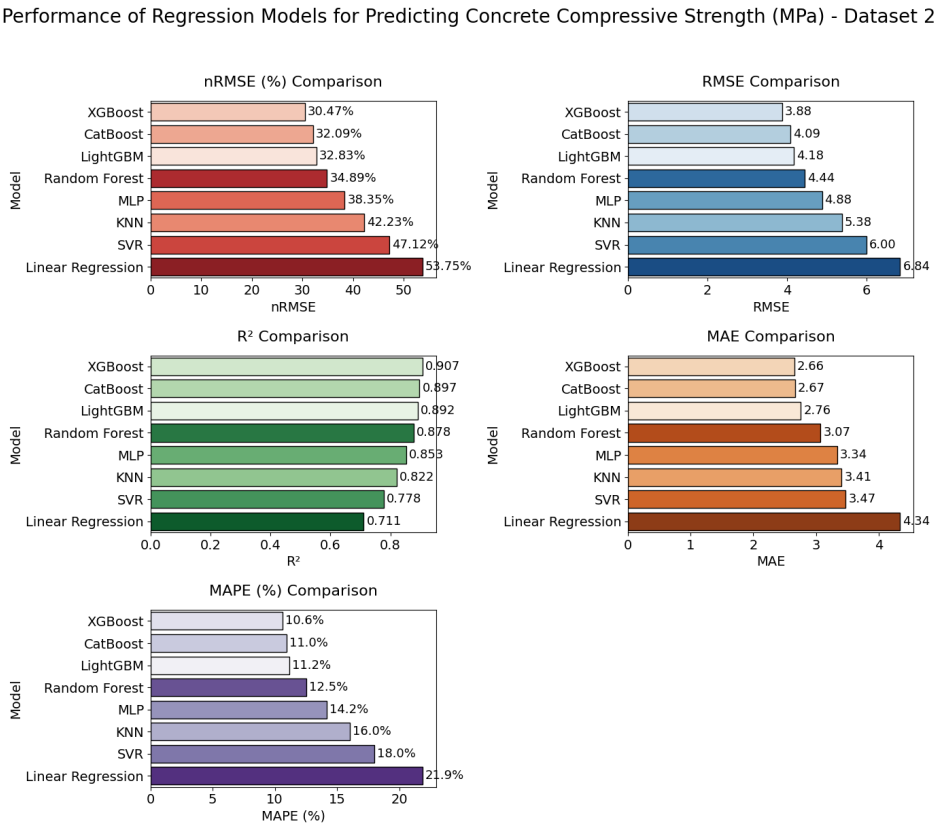

Figure 5. Global comparison of the regression models' performance for the **Ke–Qiu** dataset. The five evaluation metrics are shown: nRMSE, RMSE, MAE, MAPE, and R^2.

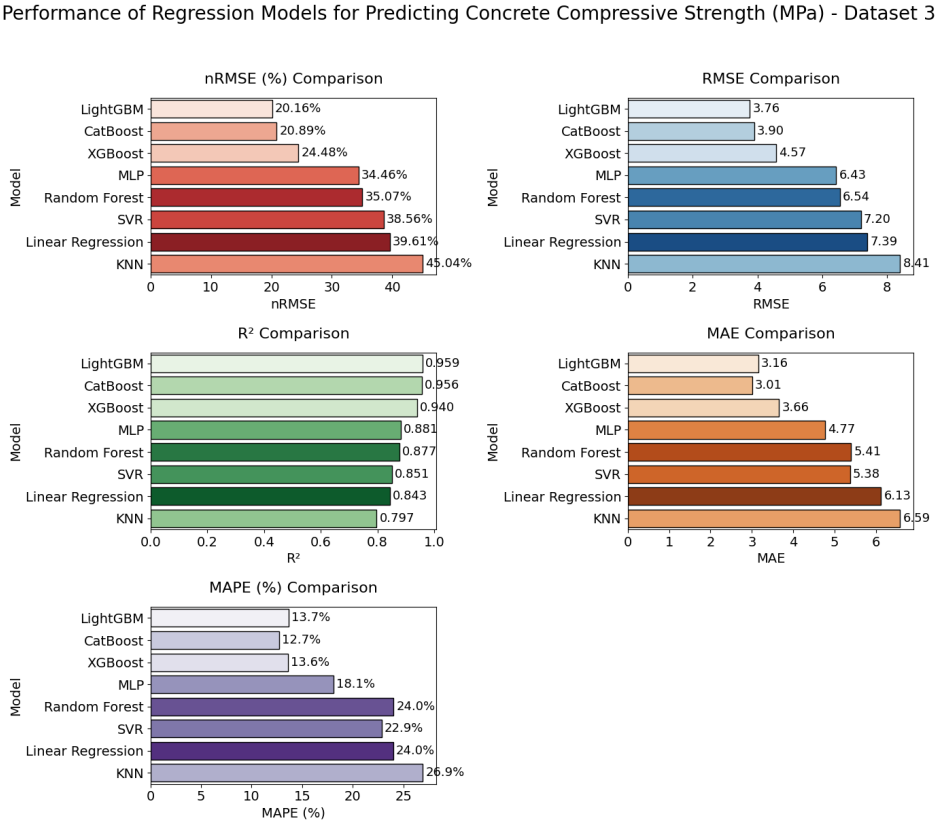

Figure 6. Global comparison of the regression models' performance for the **Biswal** dataset. The five evaluation metrics are shown: nRMSE, RMSE, MAE, MAPE, and R^2.

3.2. Ranking Stability Across Datasets

To analyze the consistency of algorithm performance across different experimental contexts, the **individual rankings** for each dataset were computed based on the nRMSE (%) metric. Subsequently, the **average ranking** was obtained to identify the models with the highest stability and generalization capability across datasets.

Figure 7 visually summarizes these results: **a)** shows the relative order of the eight algorithms within each dataset (*Ranks per Dataset*), while **b)** presents the overall average ranking (*Average Rank across Datasets*).

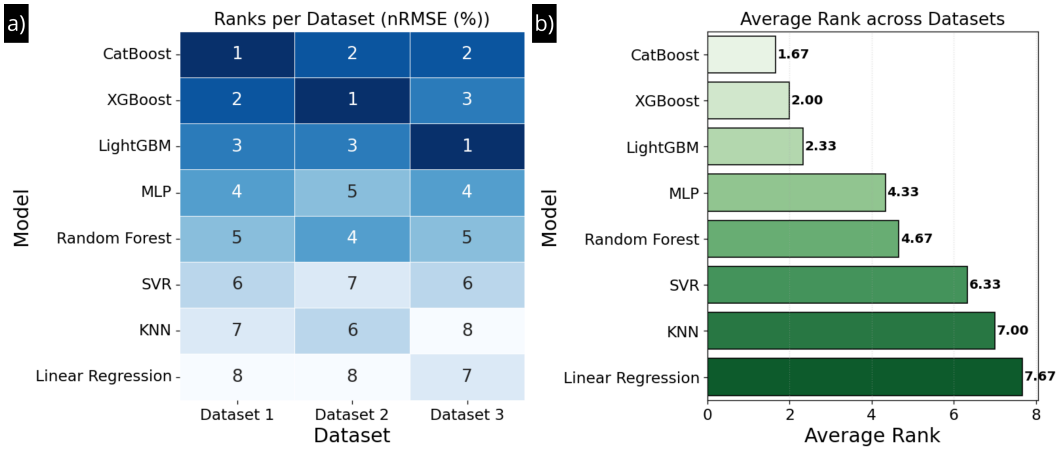

Figure 7. Performance stability across datasets. **a)**: heatmap showing the ranks per dataset based on nRMSE (1 = best). **b)**: bar chart of the *average rank* (lower is better) for the eight models across the three datasets.

The average ranking confirmed the overall superiority of the *boosting* methods, highlighting CatBoost (**1.67**), XGBoost (**2.00**), and LightGBM (**2.33**) as the most consistent models. Neural network– and tree-based algorithms (MLP and Random Forest) showed intermediate performance (**4.33** and **4.67**), while linear and nearest-neighbor methods ranked last, revealing their limited generalization ability under variations in concrete composition.

The Spearman correlation coefficients among the rankings of the three datasets were high—**0.93** between Yeh and Ke–Qiu, **0.90** between Yeh and Biswal, and **0.81** between Ke–Qiu and Biswal—indicating a **strong positive correlation** in the relative performance order of the models. This result suggests that the algorithms maintain stable behavior even under different experimental conditions, reinforcing the reliability of *boosting* methods for predicting compressive strength in concretes of varying nature.

Overall, these findings demonstrate a **strong statistical coherence among the performance rankings**, reflecting high inter-dataset stability and supporting the applicability of *boosting* methods as the most robust and consistent models for predicting concrete compressive strength.

3.3. Prediction Scatter Analysis

To complement the quantitative metrics presented in Tables 6–8 and the visual comparisons in Figures 4–4, scatter plots were generated between the actual and predicted values for the eight evaluated models in each dataset. Figures 8–10 group these results by dataset, showing the degree of alignment of the predictions with respect to the identity line ($y = x$).

This type of visualization is essential for analyzing model fit and detecting potential biases or error patterns that global metrics may conceal. Across all three datasets, the observed behavior is consistent with the numerical results: the *gradient boosting* models (**CatBoost**, **XGBoost**, and **LightGBM**) show a high concentration of points around the

identity line, indicating consistent predictions and low dispersion. In contrast, the **Linear**, **KNN**, and **SVR** models exhibit greater scatter and systematic deviations, also reflected in their lower R^2 values and higher absolute errors.

Overall, these visualizations confirm the superiority of boosting methods in accurately estimating concrete compressive strength and their ability to maintain stable performance under the different experimental conditions represented in the three datasets.

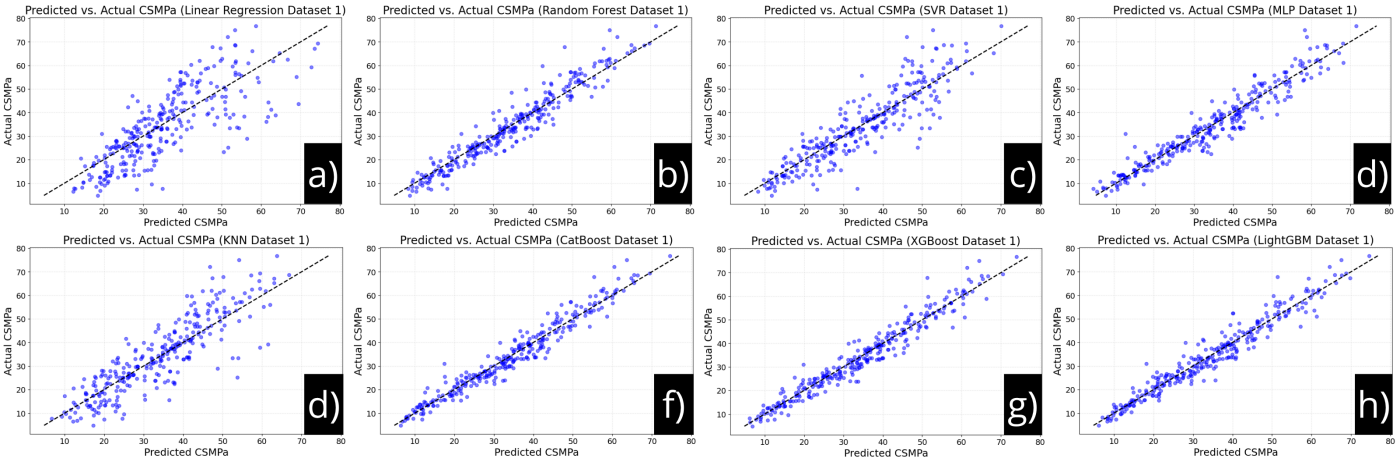

Figure 8. catter plots between actual and predicted values for the eight evaluated models in **Dataset 1** (UCI–Yeh). The subfigures show: **a)** Linear Regression, **b)** Random Forest, **c)** SVR, **d)** MLP, **e)** KNN, **f)** CatBoost, **g)** XGBoost y **h)** LightGBM. The diagonal line represents the identity ($y = x$).

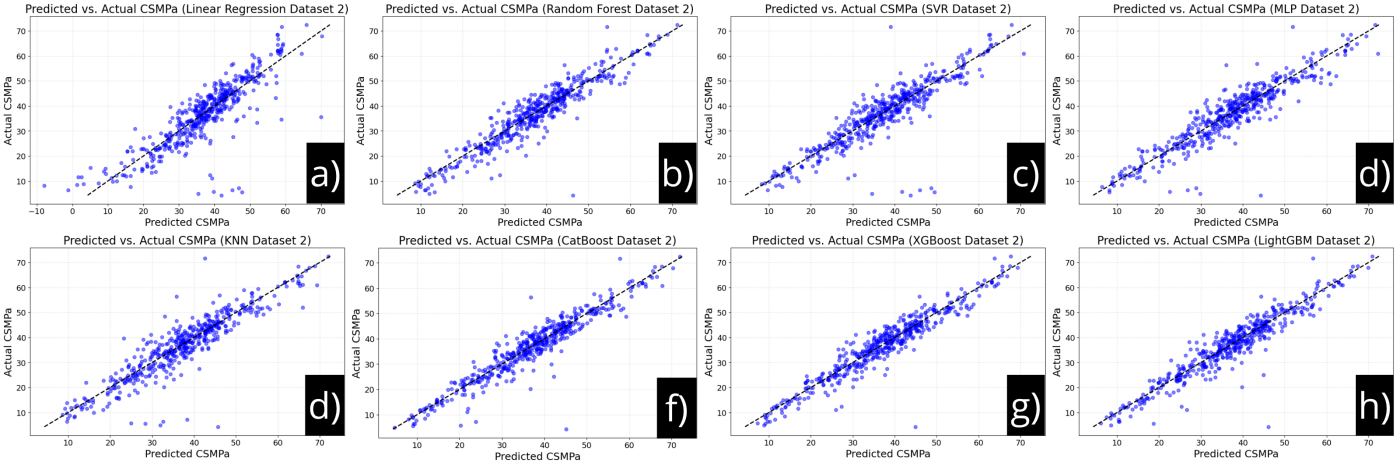

Figure 9. Scatter plots between actual and predicted values for the eight evaluated models in **Dataset 2** (Ke–Qiu). The subfigures show: **a)** Linear Regression, **b)** Random Forest, **c)** SVR, **d)** MLP, **e)** KNN, **f)** CatBoost, **g)** XGBoost y **h)** LightGBM. The diagonal line represents the identity ($y = x$).

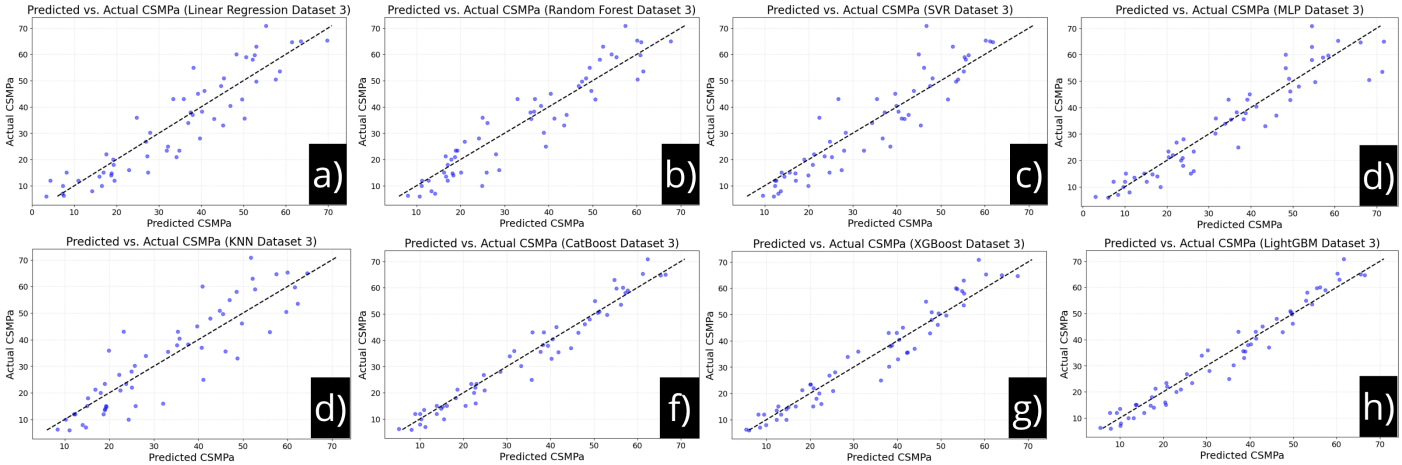

Figure 10. Scatter plots between actual and predicted values for the eight evaluated models in **Dataset 3** (Biswal). The subfigures show: **a)** Linear Regression, **b)** Random Forest, **c)** SVR, **d)** MLP, **e)** KNN, **f)** CatBoost, **g)** XGBoost y **h)** LightGBM. The diagonal line represents the identity ($y = x$).

3.4. Feature Importance Analysis

The feature importance analysis is presented in Figures 11–13 for the models that provide this measure directly: Linear Regression, Random Forest, XGBoost, LightGBM, and CatBoost. The KNN, MLP, and SVR models do not produce native importance estimates and were therefore excluded from this comparison. This analysis allows identifying the factors that most influence the prediction of concrete compressive strength and verifying the physical consistency of the relationships learned by the models.

Dataset 1 (Yeh, 1998).

In the UCI–Yeh dataset, the results were consistent across models, highlighting *age*, *cement*, and *water* as the dominant variables. In CatBoost, *age* contributed **33.8%** of the total importance, followed by *cement* (**23.4%**) and *water* (**13.8%**), together accounting for more than 70% of the overall influence. Tree-based models—Random Forest, XGBoost, and LightGBM—reflected the same pattern, with slight variations in relative weights (*age*: 30–35%, *cement*: 20–22%, *water*: 12–15%). In contrast, *fly ash*, *slag*, and *superplasticizer* showed lower importances (<10%), indicating their secondary role in strength development.

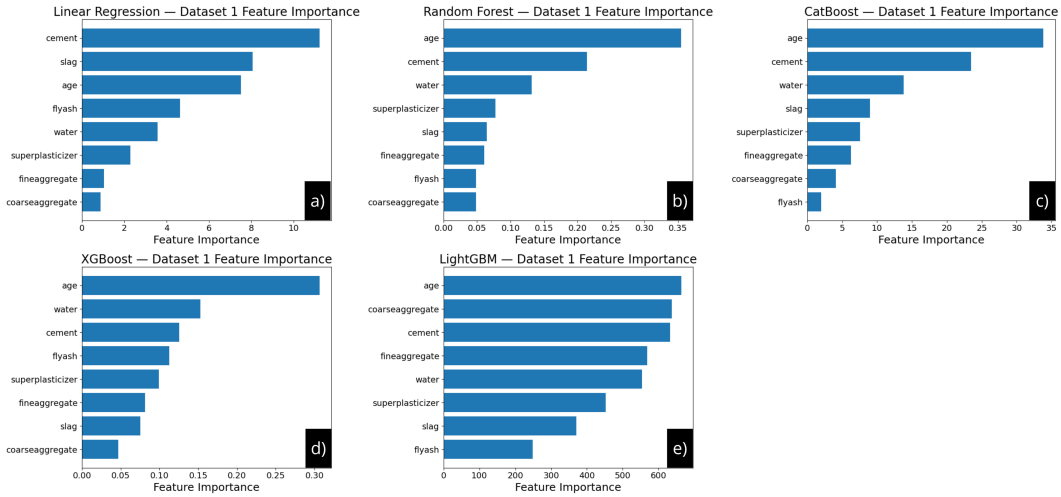

Figure 11. Relative importance of the predictive variables in **Dataset 1** (Yeh [8]). The subfigures show the results for: **a)** Linear Regression, **b)** Random Forest, **c)** CatBoost, **d)** XGBoost, and **e)** LightGBM.

Dataset 2 (Ke–Qiu, 2024).

In the Ke–Qiu dataset, corresponding to conventional concretes, the variable hierarchy remained similar. CatBoost identified *cement* (**30.8%**), *water* (**17.0%**), and *water_reducing_admixture* (**14.9%**) as the main determining factors, followed by *coarse_aggregates* (**12.7%**) and *time* (**9.8%**). XGBoost and LightGBM confirmed this structure, again ranking cement and water among the variables with the highest relative weights. The variables *silica_fume* and *accelerating_agent* showed marginal contributions (<2%), indicating their limited influence on the final strength of the tested mixes.

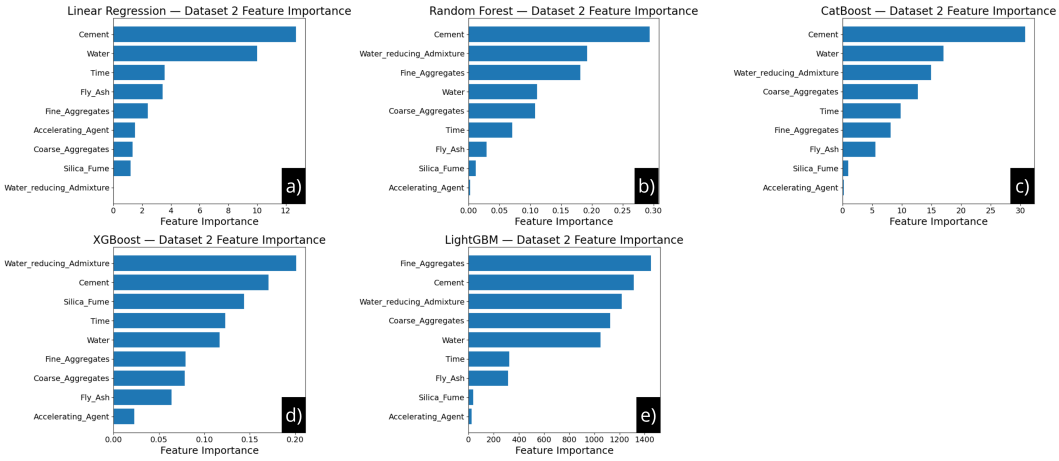

Figure 12. Relative importance of the predictive variables in **Dataset 2** (Ke–Qiu [24]). The subfigures show the results for: **a)** Linear Regression, **b)** Random Forest, **c)** CatBoost, **d)** XGBoost and **e)** LightGBM.

Dataset 3 (Biswal, 2022).

In the Biswal dataset, corresponding to concretes with recycled aggregates and more complex mix compositions, the importance values were distributed among a larger number of variables. CatBoost assigned the highest contributions to *age* (**26.3%**), *cement* (**17.6%**), and *flyash* (**16.3%**), followed by *water_TCM* (**9.6%**) and *TCM* (**9.5%**). To a lesser extent, variables associated with the granulometry of recycled aggregates (RCA, NCA) showed individual importances below 3%. The XGBoost and LightGBM models confirmed this behavior, maintaining the dominance of variables related to the cementitious matrix and curing age over those associated with aggregate characteristics.

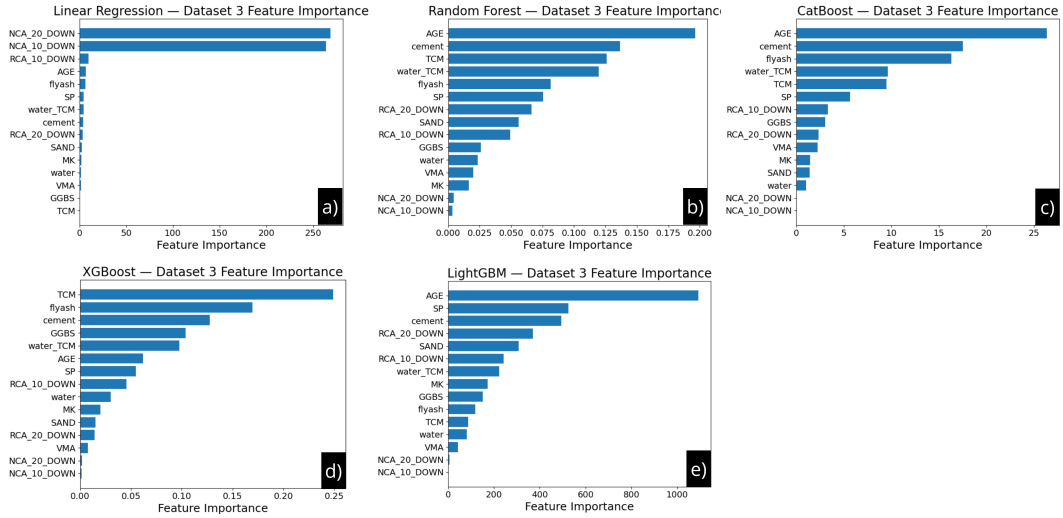

Figure 13. Relative importance of the predictive variables in **Dataset 3** (Biswal [25]). The subfigures show the results for: **a)** Linear Regression, **b)** Random Forest, **c)** CatBoost, **d)** XGBoost and **e)** LightGBM.

Taken together, the three analyses reveal a physically coherent pattern: cement hydration, the water-to-cement ratio, and curing age are the key factors governing the development of compressive strength [1]. The consistency of these results across algorithms and datasets reinforces the validity of *gradient boosting* models as interpretable and robust tools for identifying critical variables in concrete prediction systems.

3.5. Inference Systems

Three interactive inference systems based on machine learning algorithms were developed for predicting the compressive strength of concrete, each built from the best-performing model obtained for its corresponding dataset. Figure 14 shows the general interface of the three systems, labeled **1**, **2**, and **3**, corresponding to the **Yeh**, **Ke–Qiu**, and **Biswal** datasets, respectively. These tools enable the immediate estimation of concrete compressive strength (CSMPa) from mix proportions and curing age, providing users with an interactive and reproducible experience through Google Colab.

The development of these systems represents an **applied phase of technological transfer**, aimed at bridging predictive modeling from theoretical research to practical applications. Although **functional and experimental validation** is not yet part of the present study, future work includes implementing a validation program based on the **Yeh (Dataset 1)** configuration, whose mix and curing conditions are well-documented [8]. This will allow comparing the model predictions with laboratory results obtained in accordance with **ASTM C31/C31M** and **ASTM C39/C39M** standards.

Overall, the developed systems represent a **first step toward the intelligent automation of concrete design**, providing an accessible, reproducible, and low-cost tool that complements traditional physical testing. These platforms can support preliminary mix design, experimental planning, and quality control, helping to reduce time, cost, and reliance on destructive testing.

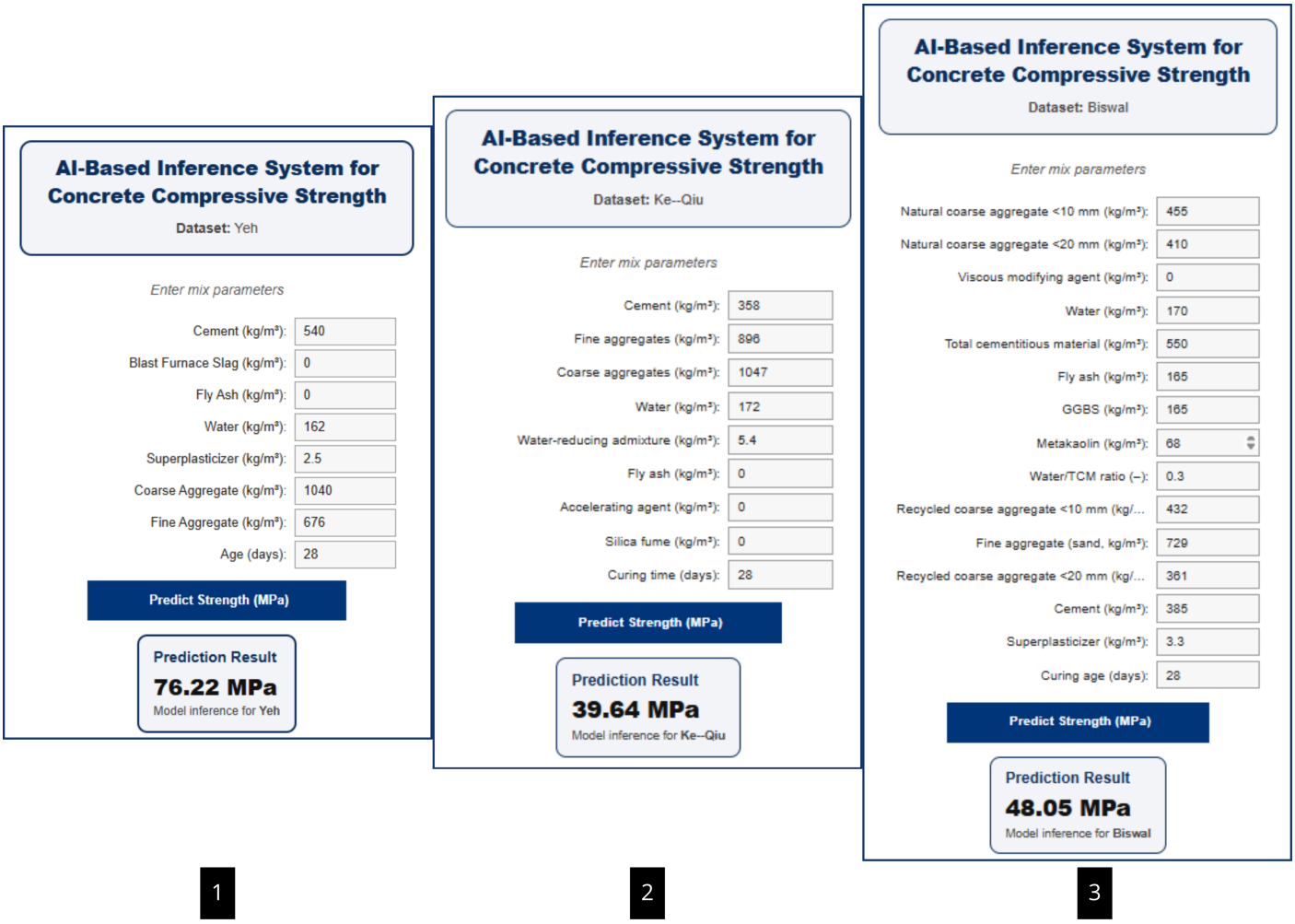

Figure 14. General interface of the developed inference systems. The labels identify the three datasets: **1–Yeh**, **2–Ke–Qiu**, and **3–Biswal**. Each system allows users to input mix variable values and instantly obtain the predicted compressive strength (CSMPa).

Discussion

The proposed approach was based on a systematic methodology applied to three experimental datasets with distinct characteristics: **(1)** the Yeh dataset [8], corresponding to High-Performance Concrete (HPC) produced with ordinary Portland cement; **(2)** the Ke–Qiu dataset [24], representative of conventional concretes with lower water-to-cement ratios and a wide range of curing ages; and **(3)** the Biswal et al. dataset [25], which incorporates recycled aggregates and supplementary cementitious materials such as fly ash, GGBS, and metakaolin. This diversity enabled the analysis of model stability under different experimental contexts and levels of material heterogeneity, establishing a solid basis for assessing inter-dataset generalization capability.

Hyperparameter optimization using `RandomizedSearchCV` was a key element of the methodology. This technique allowed efficient exploration of a wide hyperparameter space at moderate computational cost, enabling each algorithm to reach its optimal configuration under a homogeneous cross-validation scheme. This procedure ensured comparability and reproducibility—essential conditions for establishing a reliable benchmark across models and datasets with different distributions and scales.

The results showed that *gradient boosting* methods—particularly CatBoost, XGBoost, and LightGBM—maintained consistently superior performance across all three datasets, both in absolute terms (RMSE and R^2) and relative terms (nRMSE). For the Yeh dataset, CatBoost achieved R^2 = **0.950** and RMSE = **3.48 MPa**; for Ke–Qiu, the same model reached

R^2 = **0.932** with RMSE = **4.12 MPa**; and for Biswal, R^2 **= 0.921** with RMSE = **4.39 MPa**. This stability across error ranges confirms the robustness of the model under variations in composition, aggregate type, and the inclusion of supplementary materials. Moreover, the high Spearman correlation coefficients among performance rankings ($\rho > 0.8$) indicate statistical coherence across datasets, supporting the generalization capability of the *boosting* methods.

In contrast, linear algorithms (Linear Regression, Ridge, Lasso) and distance-based methods (KNN, SVR) showed greater sensitivity to differences in variable scale and dispersion, as well as a significant performance drop when transferred across datasets. This indicates that simple parametric models fail to capture the nonlinear interactions between the physicochemical variables of concrete, particularly in mixes containing alternative materials or higher granulometric variability. Meanwhile, decision tree–based methods (Random Forest) and neural networks (MLP) exhibited intermediate results, achieving good fits for more homogeneous datasets (Yeh and Ke–Qiu) but showing a slight loss of accuracy in Biswal, where the compositional complexity was higher.

From an engineering standpoint, the RMSE values obtained—ranging between **3.4** and **4.4 MPa**—fall within the experimental variability margins established by ASTM C39, which allows differences of 5–10% between nominally identical specimens [1,2,54]. This finding reinforces the practical applicability of the proposed models, as the magnitude of the predictive error is comparable to the inherent variability of laboratory testing. Consequently, the developed inference systems can be considered complementary tools for early-age strength estimation, mix design optimization, and quality control support.

The feature importance analysis consistently confirmed across datasets that the most influential variables were concrete age, cement content, and water content, followed by the water-to-cement ratio and secondary mineral components. The agreement between model interpretation and theoretical principles of hydration and strength development [1] reinforces the physical validity of the model's inferences. In the Biswal dataset, the relative importance of mineral additions (fly ash and GGBS) further reflected their role in pozzolanic reactivity and strength gain at later ages—a phenomenon coherently captured by the model in line with experimental evidence.

The inter-dataset behavior also revealed that metric normalization through nRMSE was essential for comparing model performance across different strength ranges. While the HPC concretes in the Yeh dataset exhibit average strengths of 45–80 MPa, the conventional concretes in Ke–Qiu reach mean values of 30–60 MPa, and the Biswal dataset shows higher dispersion due to partial replacement of aggregates and cement. Despite these differences, *boosting* methods consistently maintained leading positions across all scenarios, suggesting that the learned relationships are structurally transferable and not confined to the specific training domain.

From a reproducibility perspective, the random seed (`random_state`) had a notable impact mainly on stochastic methods, although variations were below 2% in global metrics. Comparing results obtained with `random_state = 123` and `random_state = 42`, it was verified that algorithms such as SVR, MLP, and KNN produced nearly identical metrics, while CatBoost and XGBoost showed slight improvements with `random_state = 42`, achieving R^2 values of **0.950** and **0.947**, respectively. Conversely, LightGBM performed marginally better with `random_state = 123`. Although subtle, these differences highlight that some tree-based ensemble algorithms are sensitive to internal random sampling processes, reinforcing the importance of fixing and reporting random seeds in reproducible studies.

The implementation of interactive inference systems in Google Colab—one for each dataset—represents a practical and distinctive contribution. These tools allow real-time

predictions using the trained models without requiring local infrastructure or advanced programming knowledge. Their open and reproducible design aligns with the FAIR principles (*Findable, Accessible, Interoperable, Reusable*) and provides a starting point for developing web or mobile platforms applicable to concrete mix design and production control.

Nevertheless, certain limitations are recognized. Although the three analyzed datasets encompass different experimental contexts—including laboratory mixes and real production records—their size and strength range remain limited compared to the variability observed in industrial practice. In particular, the datasets used represent specific intervals of composition and strength, constraining the model's ability to extrapolate to very high- or low-strength mixes. Moreover, the absence of environmental variables (such as temperature, humidity, or *in situ* curing conditions) and long-term performance indicators (e.g., elastic modulus or shrinkage) limits the predictive scope of the models for broader operational contexts.

For future work, it is proposed to conduct the experimental validation of **Inference System 1**, based on the CatBoost model trained with the Yeh dataset [8]. This system is the most suitable for laboratory implementation due to its smaller number of input variables and the comprehensive methodological documentation of the original study, which details the mix, curing, and testing conditions under ASTM C31/C39 standards. These characteristics facilitate experimental reproducibility and will enable comparison between model predictions and actual compressive strength measurements, strengthening the practical and scientific validity of the proposed framework.

Overall, the findings confirm the effectiveness of machine learning for modeling concrete compressive strength and provide three main contributions: **(1)** the establishment of a reproducible comparative methodology applied to multiple datasets; **(2)** the identification of statistical stability and inter-dataset coherence of *boosting* models; and **(3)** the development of interactive inference systems demonstrating the feasibility of translating predictive modeling into practical engineering applications.

Conclusions

This study developed and applied a systematic and reproducible methodology to compare the performance of eight supervised regression algorithms in predicting the compressive strength of concrete, using three experimental datasets representative of different material and production contexts. The optimization strategy based on `RandomizedSearchCV`, combined with normalized performance metrics, ensured a fair and statistically coherent comparison among models and datasets, providing a comprehensive evaluation of their stability and generalization capability.

The results consistently demonstrated that *gradient boosting* algorithms—particularly CatBoost, XGBoost, and LightGBM—outperformed linear, neighbor-based, and neural network methods across all analyzed scenarios. Among them, CatBoost achieved the best overall performance, reaching R^2 values between **0.92** and **0.95** and RMSE values between **3.4** and **4.4 MPa**, depending on the dataset. These error margins are comparable to the experimental variability reported in ASTM C39 tests, validating their practical applicability in real-world civil engineering environments.

The inter-dataset analysis revealed a high coherence in performance rankings (Spearman coefficients $\rho > 0.8$), confirming the statistical stability of *boosting* models under variations in concrete composition, aggregate type, and inclusion of supplementary materials. Furthermore, the feature importance analysis showed a direct correspondence between the most influential variables (age, cement content, water, and water-to-cement

ratio) and the theoretical fundamentals of hydration and strength development, reinforcing the physical validity of the model inferences.

The implementation of three interactive inference systems—one for each dataset—represents a significant practical contribution, translating predictive modeling outcomes into an accessible environment via Google Colab. These tools enable real-time estimations without requiring local infrastructure or advanced programming skills and align with the FAIR principles of openness and reproducibility. Their potential applications in mix design, quality control, or early-age strength prediction mark an important step toward integrating artificial intelligence into concrete engineering practice.

Despite these advances, the study presents limitations related to the size and range of the employed datasets, which cover restricted intervals of strength and composition. Although they include both laboratory mixes and real production records, the absence of environmental variables (temperature, humidity, curing type) and long-term properties (elastic modulus, shrinkage) constrains the model's extrapolation to broader contexts.

As future work, the experimental validation of **Inference System 1**, based on the CatBoost model trained with the Yeh dataset [8], is proposed. This system is the most suitable for laboratory implementation due to its smaller number of input variables and the comprehensive methodological documentation of the original study, which facilitates experimental reproduction under ASTM C31/C39 standards. The direct comparison between model predictions and experimental results will strengthen the scientific and practical validity of the proposed framework, establishing a tangible link between statistical modeling and the actual behavior of concrete.

In summary, this research provides a robust, comparative, and transparent methodological framework for applying machine learning to the prediction of concrete mechanical properties. By integrating a multi-dataset approach, normalized metrics, stability analysis, and practical inference tools, this work lays the foundation for developing more generalizable, reproducible, and transferable models, contributing to advances in materials engineering and the digitalization of construction processes.

Author Contributions: "Conceptualization, C.E.O.-M.; methodology, C.E.O.-M.; software, C.E.O.-M. and M.d.J.L.-M.; validation, C.E.O.-M; formal analysis, C.E.O.-M. and C.E.O.-O.; investigation, C.E.O.-M.; resources, L.O.S.-S., S.V-R., D.D.-C. and J.V.G.-A; data curation, C.E.O.-M.; writing—original draft preparation, C.E.O.-M. and J.A.R.-R.; writing—review and editing, C.A.O.-O. and J.I.d.l.R.-V; visualization, C.E.O.-M.; supervision, C.A.O.-O. and M.d.J.L.-M.; project administration, C.A.O.-O. and M.d.J.L.-M.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: The datasets analyzed in this study are publicly available and can be freely accessed as follows: Dataset 1 — Concrete Compressive Strength (Yeh, 1998), available at the UCI Machine Learning Repository: https://archive.ics.uci.edu/dataset/165/concrete+compressive+strength. Dataset 2 — Concrete Compressive Strength and Slump Dataset (Ke & Lu, 2024), available at Mendeley Data: https://data.mendeley.com/datasets/zrsbhndz9f/1. Dataset 3 — Recycled Aggregate Concrete with Fly Ash, GGBS, and Metakaolin (Biswal et al., 2022), available at Mendeley Data: https://data.mendeley.com/datasets/5wkxzmzwnz/2.

No new experimental data were generated in this study. The code used for model training, optimization, and comparative analysis is available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
ML	Machine Learning
CS	Compressive Strength
MPa	Megapascals
W/C	Water-to-Cement Ratio
HPC	High-Performance Concrete
RAC	Recycled Aggregate Concrete
SCM	Supplementary Cementitious Materials
TCM	Total Cementitious Materials
GGBS	Ground-Granulated Blast-Furnace Slag
SP	Superplasticizer
VMA	Viscosity-Modifying Agent
NCA	Natural Coarse Aggregate
RCA	Recycled Coarse Aggregate
SVR	Support Vector Regression
MLP	Multilayer Perceptron
KNN	*k*-Nearest Neighbors
RF	Random Forest
ANN	Artificial Neural Network
XGBoost	Extreme Gradient Boosting
LightGBM	Light Gradient Boosting Machine
CatBoost	Categorical Gradient Boosting
SHAP	SHapley Additive exPlanations
GWO	Grey Wolf Optimizer
IGWO	Improved Grey Wolf Optimizer
QPSO	Quantum-behaved Particle Swarm Optimization
RBFNN	Radial Basis Function Neural Network
DA	Dragonfly Algorithm
SA	Simulated Annealing
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
nRMSE	Normalized Root Mean Square Error
R^2	Coefficient of Determination
PICP	Prediction Interval Coverage Probability
FAIR	Findable, Accessible, Interoperable, Reusable
UCI	UCI Machine Learning Repository
IIT	Indian Institute of Technology
ASTM	ASTM International (Standards Organization)
Google Colab	Google Collaboratory

References

1. Neville, A.M. *Properties of Concrete*, 5th ed.; Pearson Education Limited: Harlow, U.K., 2011.
2. ASTM International. Standard Test Method for Compressive Strength of Cylindrical Concrete Specimens (ASTM C39/C39M). ASTM International: West Conshohocken, PA, USA, 2023. https://doi.org/10.1520/C0039_C0039M-23.
3. ASTM International. Standard Practice for Making and Curing Concrete Test Specimens in the Field (ASTM C31/C31M). ASTM International: West Conshohocken, PA, USA, 2022. https://doi.org/10.1520/C0031_C0031M-22B.
4. ASTM International. *Standard Test Method for Compressive Strength of Hydraulic Cement Mortars (Using Portions of Prisms Broken in Flexure) (ASTM C684/C684M)*. ASTM International: West Conshohocken, PA, USA, 2020. https://doi.org/10.1520/C0684_C0684M-20.

5. ASTM International. *Standard Test Method for Pulse Velocity Through Concrete (ASTM C597)*. ASTM International: West Conshohocken, PA, USA, 2016. https://doi.org/10.1520/C0597-16.
6. ASTM International. *Standard Test Method for Rebound Number of Hardened Concrete (ASTM C805/C805M)*. ASTM International: West Conshohocken, PA, USA, 2018. https://doi.org/10.1520/C0805_C0805M-18.
7. ASTM International. Standard Test Method for Static Modulus of Elasticity and Poisson's Ratio of Concrete in Compression (ASTM C469/C469M). ASTM International: West Conshohocken, PA, USA, 2021. https://doi.org/10.1520/C0469_C0469M-21.
8. Yeh, I.-C. Modeling of strength of high-performance concrete using artificial neural networks. *Cem. Concr. Res.* **1998**, *28*, 1797–1808. https://doi.org/10.1016/S0008-8846(98)00165-3.
9. Dua, D.; Graff, C. UCI Machine Learning Repository: Concrete Compressive Strength Data Set. 2017. Available online: https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength (accessed on 16 July 2025).
10. Rathakrishnan, M.; Lavanya, C.; Ganesan, K. Comparative analysis of machine learning models to predict compressive strength of slag-based concrete. *Mater. Today Proc.* **2022**, *62*, 5286–5293. https://doi.org/10.1016/j.matpr.2022.03.730.
11. Xu, Y.; Zhang, S.; Liu, C. Prediction of concrete strength using machine learning with ensemble methods. *Eng. Struct.* **2020**, *223*, 111136. https://doi.org/10.1016/j.engstruct.2020.111136.
12. Gamil, M. Machine learning techniques for predicting the compressive strength of concrete: A review. *Front. Built Environ.* **2023**, *9*, 1145591. https://doi.org/10.3389/fbuil.2023.1145591.
13. Lundberg, S.M.; Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*; Curran Associates: Red Hook, NY, USA, 2017; Volume 30. Available online: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf (accessed on 11 October 2025).
14. Zhang, C.; Li, B.; Wang, J.; Zhou, Y. DeepForest-based ensemble learning model for predicting high-performance concrete compressive strength. *Sci. Rep.* **2024**, *14*, 69616. https://doi.org/10.1038/s41598-024-69616-9.
15. Shaaban, M.; Amin, M.; Selim, S.; Riad, I.M. Machine learning approaches for forecasting compressive strength of high-strength concrete. *Sci. Rep.* **2025**, *15*, 25567. https://doi.org/10.1038/s41598-025-10342-1.
16. Chou, J.-S.; Pham, A.-D. Enhanced artificial intelligence for ensemble approach to predicting civil infrastructure costs. *Constr. Build. Mater.* **2013**, *49*, 554–563. https://doi.org/10.1016/j.conbuildmat.2013.08.078.
17. Olawale, O.A.; Akinosho, T.D.; Oyedele, L.O.; Ajayi, A.O.; Akanbi, L.A.; Delgado, J.M.D. Explainable artificial intelligence framework for predicting compressive strength of concrete with SHAP-based model interpretation. *AI Civ. Eng.* **2025**, *1*, 4. https://doi.org/10.1007/s43503-025-00061-x.
18. Wang, Q.; Zhang, X.; Li, H.; Zhou, Y. Hybrid machine learning models for predicting compressive strength and slump flow of high-performance concrete. *Sci. Rep.* **2025**, *15*, 10860. https://doi.org/10.1038/s41598-025-10860-y.
19. Xu, Y.; Afzal, A.; Li, Z.; Wang, Y. Applying machine learning techniques in the form of ensemble and hybrid models to appraise hardness/strength properties of high-performance concrete. *J. Intell. Fuzzy Syst.* **2024**, *46*, 2749–2764. https://doi.org/10.3233/JIFS-234409.
20. Zhao, P.; Li, Y.; Chen, H. Predicting the compressive strength of high-performance concrete by using Radial Basis Function with optimization (Improved Grey Wolf optimizer and Dragonfly algorithm). *J. Intell. Fuzzy Syst.* **2023**, *45*, 7917–7932. https://doi.org/10.3233/JIFS-224382.
21. Vargas, A.; Martínez, L.; López, J. Machine-learning-based predictive models for compressive strength, flexural strength, and slump of concrete. *Appl. Sci.* **2024**, *14*, 4426. https://doi.org/10.3390/app14114426.
22. Cheng, Y.; Liu, J.; Wu, T. Data-driven modeling and uncertainty analysis for concrete compressive strength prediction using ensemble learning. *Eng. Appl. Artif. Intell.* **2021**, *100*, 104214. https://doi.org/10.1016/j.engappai.2021.104214.
23. Bitencourt, L.; Barbosa, F.; Monteiro, D. A reproducibility analysis of ML models for concrete strength: Challenges and recommendations. *J. Build. Eng.* **2024**, *87*, 107142. https://doi.org/10.1016/j.jobe.2024.107142.
24. Ke, L.; Ming, Q. Dataset of compressive strength and slump of normal concrete. *Mendeley Data* **2024**, *V1*. https://doi.org/10.17632/zrsbhndz9f.1.
25. Biswal, U.S.; Pasla, D.; Mishra, M. Experimental dataset for concrete compressive strength prediction from IIT Bhubaneswar concrete laboratory. *Mendeley Data* **2022**, *V2*. https://doi.org/10.17632/5wkxzmzwnz.2.
26. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed.; Springer: New York, NY, USA, 2009.
27. Biswal, U.S.; Mishra, M.; Singh, M.K.; et al. Experimental investigation and comparative machine learning prediction of the compressive strength of recycled aggregate concrete incorporated with fly ash, GGBS, and metakaolin. *Innov. Infrastruct. Solut.* **2022**, *7*, 242. https://doi.org/10.1007/s41062-022-00844-6.
28. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. https://doi.org/10.1023/A:1010933404324.
29. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88.

30. Chen, T.; Guestrin, C. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794. https://doi.org/10.1145/2939672.2939785.
31. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.-Y. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 4–9 December 2017. Available online: https://papers.nips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html (accessed on 11 October 2025).
32. ScienceDirect Topics. Root Mean Square Error—Overview. 2024. Available online: https://www.sciencedirect.com/topics/engineering/mean-square-error (accessed on 11 October 2025).
33. Lightning AI. Normalized Root Mean Squared Error (NRMSE). 2024. Available online: https://lightning.ai/docs/torchmetrics/latest/regression/normalized_root_mean_squared_error.html (accessed on 11 October 2025).
34. Montgomery, D.C.; Peck, E.A.; Vining, G.G. *Introduction to Linear Regression Analysis*, 5th ed.; Wiley: Hoboken, NJ, USA, 2012.
35. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning: With Applications in R*, 2nd ed.; Springer: New York, NY, USA, 2021. https://doi.org/10.1007/978-1-0716-1418-1.
36. Willmott, C.J.; Matsuura, K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Clim. Res.* **2005**, *30*, 79–82. https://doi.org/10.3354/cr030079.
37. Hyndman, R.J.; Koehler, A.B. Another look at measures of forecast accuracy. *Int. J. Forecast.* **2006**, *22*, 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001.
38. Kvalseth, T.O. Cautionary note about R^2. *Am. Stat.* **1985**, *39*, 279–285. https://doi.org/10.1080/00031305.1985.10479448.
39. Chicco, D.; Warrens, M.J.; Jurman, G. The coefficient of determination (R^2) is more informative than SMAPE, MAE, MAPE, MSE, and RMSE in regression analysis evaluation. *PeerJ Comput. Sci.* **2021**, *7*, e623. https://doi.org/10.7717/peerj-cs.623.
40. Shapiro, S.S.; Wilk, M.B. An analysis of variance test for normality (complete samples). *Biometrika* **1965**, *52*, 591–611.
41. Kim, H.-Y. Statistical notes for clinical researchers: assessing normal distribution (2) using skewness and kurtosis. *Restor. Dent. Endod.* **2013**, *38*, 52–54. https://doi.org/10.5395/rde.2013.38.1.52.
42. Doane, D.P.; Seward, L.E. Measuring skewness: A forgotten statistic? *J. Stat. Educ.* **2011**, *19*, 2. https://doi.org/10.1080/10691898.2011.11889611.
43. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning: With Applications in R*, 2nd ed.; Springer: New York, NY, USA, 2021. https://doi.org/10.1007/978-1-0716-1418-1.
44. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010; pp. 249–256. Available online: https://proceedings.mlr.press/v9/glorot10a.html (accessed on 11 October 2025).
45. Zhang, Z. Introduction to machine learning: k-nearest neighbors. *Ann. Transl. Med.* **2016**, *4*, 218. https://doi.org/10.21037/atm.2016.03.37.
46. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
47. Drucker, H.; Burges, C.J.C.; Kaufman, L.; Smola, A.; Vapnik, V. Support vector regression machines. In *Advances in Neural Information Processing Systems 9 (NeurIPS)*, Denver, CO, USA, 2–5 December 1996; pp. 155–161.
48. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
49. Scikit-learn Developers. Multi-layer Perceptron Documentation. 2024. Available online: https://scikit-learn.org/stable/modules/neural_networks_supervised.html (accessed on 11 October 2025).
50. Ng, A.Y. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, Banff, AB, Canada, 4–8 July 2004; p. 78. https://doi.org/10.1145/1015330.1015435.
51. Tan, P.-N.; Steinbach, M.; Kumar, V. *Introduction to Data Mining*, 2nd ed.; Pearson Education: London, UK, 2018.
52. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. https://doi.org/10.1214/aos/1013203451.
53. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems (NeurIPS)*, Montréal, QC, Canada, 3–8 December 2018. Available online: https://papers.nips.cc/paper/2018/hash/14491b756b3a51daac41c24863285549-Abstract.html (accessed on 11 October 2025).
54. Mindess, S.; Young, J.F.; Darwin, D. *Concrete*, 2nd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2003; ISBN 978-0130646323.