A Sensitivity Analysis of Machine Learning Models on Fire-Induced Spalling of Concrete: Revealing the Impact of Data Manipulation on Accuracy and Explainability

Mohammad K. al-Bashiti[a1], M.Z. Naser[*1,2]

[1] *School of Civil and Environmental Engineering & Earth Sciences (SCEEES), Clemson University, Clemson, SC 29634, USA*
[2] *Artificial Intelligence Research Institute for Science and Engineering (AIRISE), Clemson University, Clemson, SC 29634, USA*

(Received keep as blank , Revised keep as blank , Accepted keep as blank)

Abstract. Using an extensive database, a sensitivity analysis across fifteen machine learning (ML) classifiers was conducted to evaluate the impact of various data manipulation techniques, evaluation metrics, and explainability tools. The results of this sensitivity analysis reveal that the examined models can achieve an accuracy ranging from 72-93% in predicting the fire-induced spalling of concrete and denote the light gradient boosting machine, extreme gradient boosting, and random forest algorithms as the best-performing models. Among such models, the six key factors influencing spalling were maximum exposure temperature, heating rate, compressive strength of concrete, moisture content, silica fume content, and the quantity of polypropylene fiber. Our analysis also documents some conflicting results observed with the deep learning model. As such, this study highlights the necessity of selecting suitable models and carefully evaluating the presence of possible outcome biases.

Keywords: concrete; fire; spalling; machine learning; deep learning; sensitivity analysis; feature importance.

1. Introduction

Because of its strength, durability, and versatility, concrete is one of the most widely used building materials in the construction industry. However, under elevated temperatures, it undergoes a series of adverse mechanical changes, such as strength loss and microstructure changes (Abo Sabah et al. 2019; Saberian *et al*. 2019). Along the same lines, fire-induced spalling of concrete is an unfavorable phenomenon that is also triggered by elevated temperatures (Dwaikat and Kodur 2010; Khoury 2000).

A few theories have been proposed to explain spalling. The first theory linked the formation of internal 3D stresses resulting from concrete exposure to intense heat. Under a high heating rate, a steep temperature gradient will develop uniaxial stresses that will cause spalling once they overcome the tensile strength of concrete (Zhang and Davie 2013; Zhao *et al*. 2014). A second theory connected water evaporation inside the concrete when heated into generating a pressure that keeps building up inside the concrete (Kanema *et al*. 2011; Ozawa and Morimoto 2014). Once the accumulated pressure exceeds the tensile strength of concrete, spalling occurs. A third one combined both theories (Khoury 2015, Mindeguia *et al*. 2015).

At the moment, codal provisions lack specific recommendations and prediction methods. Hence, we must consider an innovative approach apart from conducting expensive fire tests. This is where machine learning (ML) finds its relevance as a promising alternative modern approach. Over the past decades, the application of ML has grown significantly, and it is succeeding (Teymori *et al*. 2022; Thai 2022). In addition, only a few research works mainly focused on concrete spalling. McKinney and Ali (2014) pioneered by developing two supervised learning models for the spalling classification and failure prediction of high-strength concrete columns (HSCC) subjected to fire. In addition, Zhang and Liu (2020) proposed an ML model with an accuracy of more than 80% to assess explosive spalling risk and concluded that the ML models are a

*Corresponding author, Ph.D.,
E-mail: mznaser@clemson.edu
[a]Ph.D. Student,
E-mail: malbash@clemson.edu

promising method in the domain. Also, Naser and Kodur (2022) developed a machine-learning model capable of predicting RC column fire resistance and spalling. Panev *et al*. (2021), on the other hand, built an ML model to predict the fire resistance of a composite shallow floor system.

To ensure the robustness and reliability of the applications of ML, it is essential to integrate sensitivity analysis into these studies. Sensitivity analysis is crucial in validating the outcomes and findings of such ML approaches, especially when dealing with a complex phenomenon. It helps us understand how changes across the database can impact the outcomes, enabling us to validate and establish connections between these changes and domain knowledge (Ibrahimbegovic *et al*. 2010; Naser 2023, Seitllari *et al*. 2019). A sensitivity analysis could ultimately lead to gaining more confidence in providing a trustworthy model that showcases the limitations of using these models. Thereby enhancing the validity and applicability of using these models in our domain.

This study aims to strengthen ML implementation by creating various predictive models to evaluate their sensitivity to structural fire engineering phenomena. More specifically, targeting the vulnerability of these models to diverge in evaluation metrics or feature importance plots when exposed to data manipulation techniques. We benchmark an approach for identifying the best-performing and most reliable models over different ML algorithms. By examining these algorithms, we aim to showcase the top algorithms capable of accurately predicting concrete spalling and check the sensitivity of these models by varying the database sizes and input data size and factors. Also, identify the key factors influencing spalling by integrating an explainability tool and evaluate the model's sensitivity in generating a unified feature importance plot that we can rely on for our future research direction.

2. Statistical insights

This section demonstrates the statistical insights of the used database consisting of 22 factors and more than a thousand fire tests (about 1066 tests). In this sensitivity analysis, the selected factors were identified as influential based on existing literature. Hence, we decided to focus our analysis and discussion on these factors. Figure 1 shows the graphical distribution for each factor.

3. ML models

3.1 ML algorithms

In this section, a brief description will be provided for the fifteen ML algorithms that were used in this sensitivity analysis. Namely, Adaptive Boosting (ADA), Bernoulli Naive Bayes (Bernoulli NB), Categorical Boost (CatBoost), Decision Tree (DT), Extra Trees Classifier (ETC), Gaussian Naive Bayes (Gaussian NB), K-Nearest Neighbors (KNN), Light Gradient Boosting Machine (LGBM), Linear Discriminant Analysis (LDA), Logistic Regression (LR), Random Forest (RF), SPINEX Classifier (SPINEX), Support Vector Machine (SVM), Xtreme Gradient Boosting (XGBoost) and Deep learning (DL). It is worth noting that while we provide a brief overview here, more comprehensive details about these algorithms can be found in the cited references. Further, Table 1 outlines various characteristics and properties of the used ML algorithms. The selected algorithms were compared based on their broad types of ML, most common real-world applications, the complexity of its underlying mathematics, limitations, whether it is primarily used for classification or regression tasks, sensitivity to outliers, and the interpretability of the resulting model.

3.1.1 Adaptive Boosting (AdaBoost)

AdaBoost is an ensemble learning algorithm that reassigns higher weights to the misclassified instances (adapting) (Freund and Schapire 1997). It is a boosting algorithm; generally, all boosting algorithms generate an X number of decision trees during the training phase, and only the incorrect decisions will be sent to train the second model, which is common in all the boosting ML algorithms. However, AdaBoost creates only a decision tree with two leaves, and the decision of this tree (stump) is critical, as the subsequent weak learners will mainly depend on the decision made in the previous tree. Hence, it is sensitive to noisy data and outliers. In this work, the AdaBoost classifier is augmented with the following configurations: 400 estimators, a learning rate of 0.1, the SAMME algorithm, and a random state of 4.

3.1.2 Deep learning (DL)

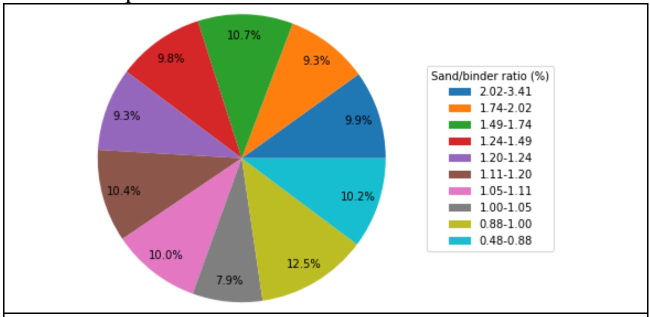
The DL algorithm is inspired by the human brain, which consists of countless neurons (McCulloch and Pitts 1943). Typically, inside this process (layers), the input will be multiplied by a weight, which will then be summed, and a bias constant will be added to that summation. Finally, the summed equation is passed over an activation function (i.e., Sigmoid), which will turn the input into the desired prediction. However, DL is a type of learning that is complex to interpret and understand its internal workings. In this work, the DL model consisted of 4 dense layers with tanh activation functions. The input layer has 128 neurons corresponding to the input dimensions. The subsequent layers have 64, 32, 16, and 8 neurons, respectively. Also, a dropout layer with a rate of 0.2 is added to minimize the overfitting of the model. Furthermore, the final dense layer has one neuron with a sigmoid activation function to produce a binary classification prediction.

3.1.3 Categorical Boosting (Catboost)

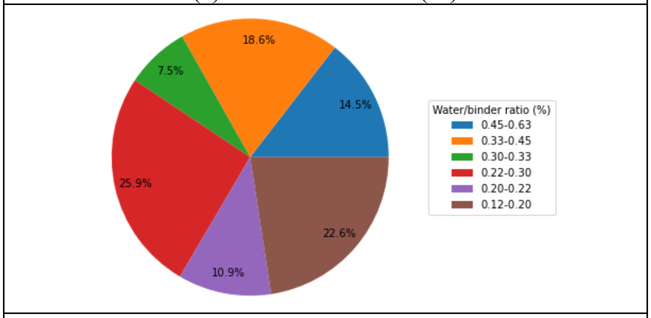
CatBoost (Prokhorenkova *et al*. 2017) is known for its excellent performance in various tasks, handling categorical variables and missing values using a symmetric weighted quantile sketch (SWQS) algorithm. However, like other gradient-boosting algorithms, it has many hyperparameters that require tuning and can be computationally expensive. The CatBoost classifier is initialized with the following hyperparameters: a learning rate of 0.5, a maximum depth of 7, a number of estimators used of 500, a random seed of 42, an evaluation metric of 'AUC,' and verbosity set to False.

3.1.4 Decision Tree (DT)

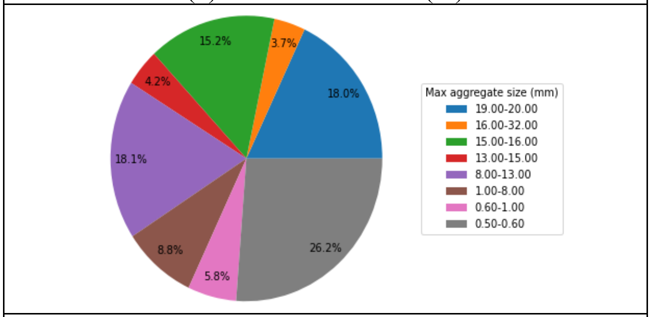
This algorithm works by building an inverted tree that splits the input into a conditional branch (Quinlan 1986). The root represents the variable being considered, while the branches represent the decision outcome of the variable



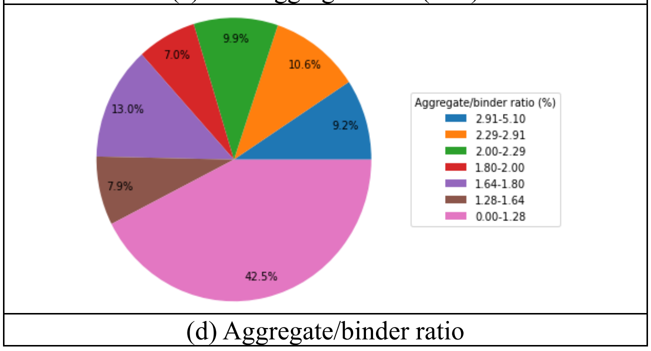
(a) Sand/binder ratio (%)



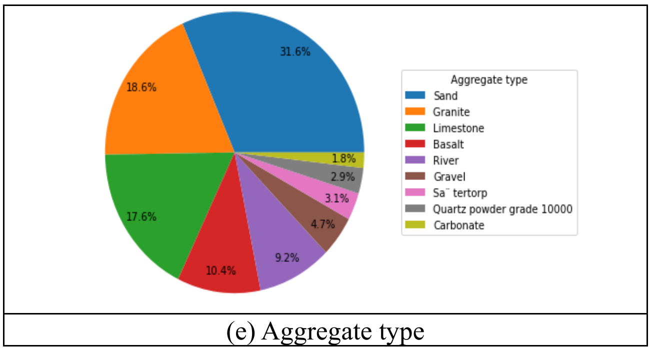
(b) Water/binder ratio (%)



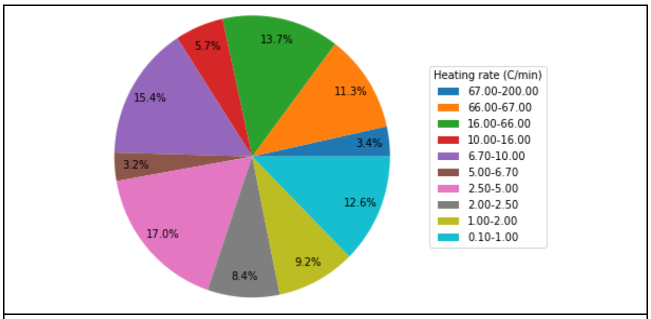
(c) Max aggregate size (mm)



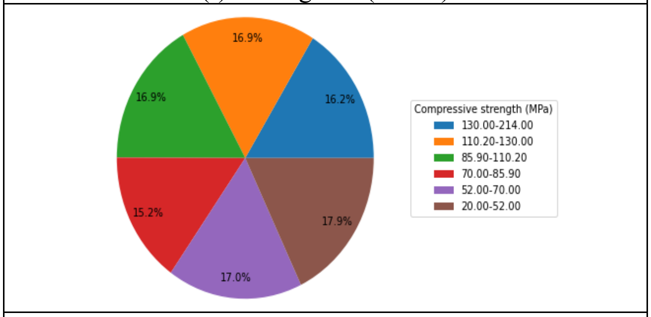
(d) Aggregate/binder ratio



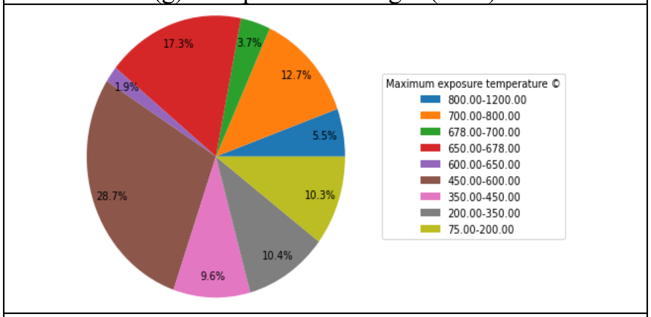
(e) Aggregate type



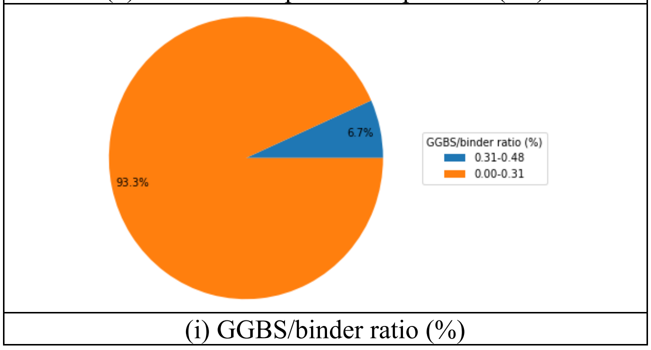
(f) heating rate (C/min)



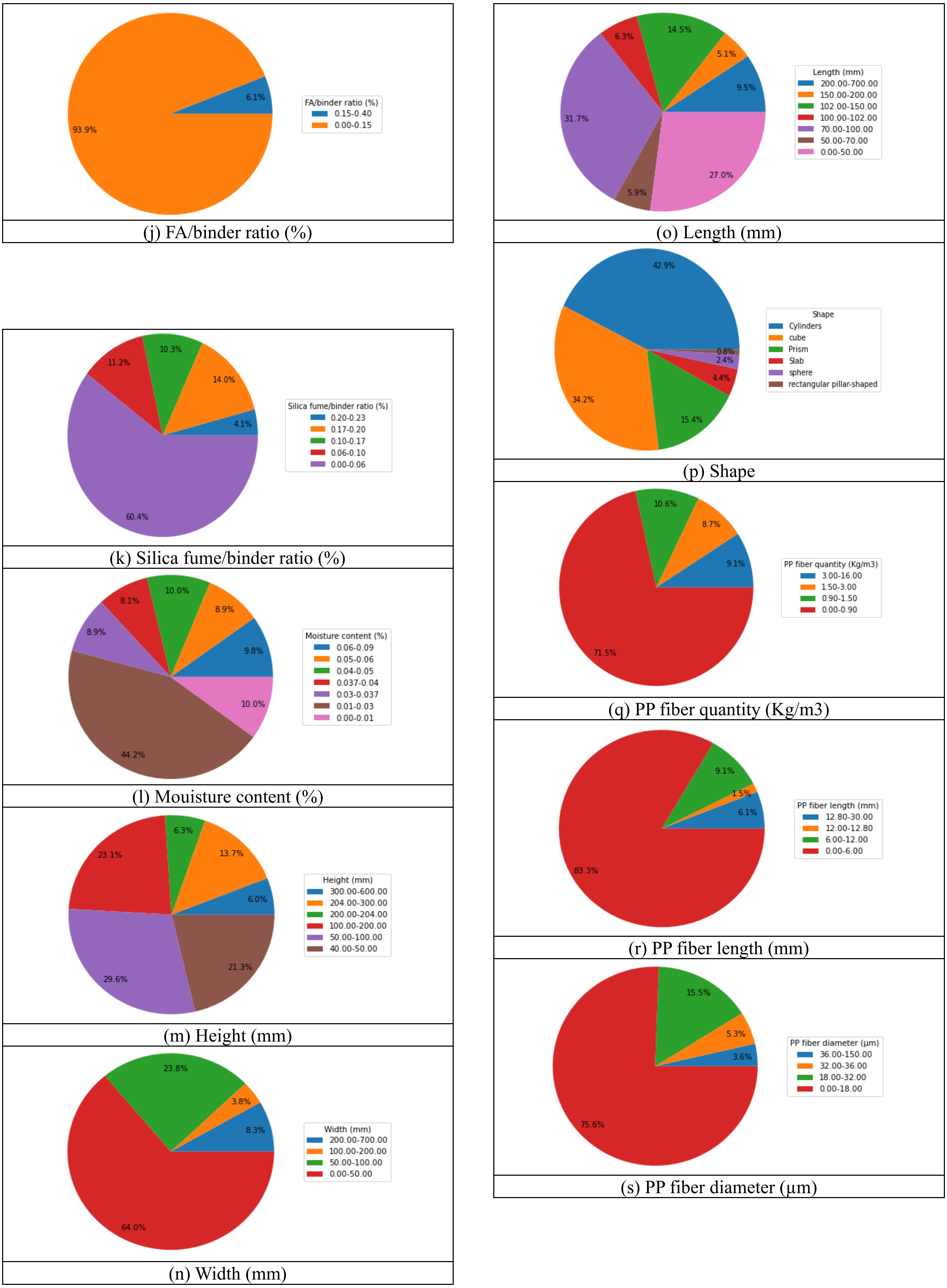
(g) Compressive strength (MPa)



(h) Maximum exposure temperature (oC)



(i) GGBS/binder ratio (%)



(j) FA/binder ratio (%)

(k) Silica fume/binder ratio (%)

(l) Mouisture content (%)

(m) Height (mm)

(n) Width (mm)

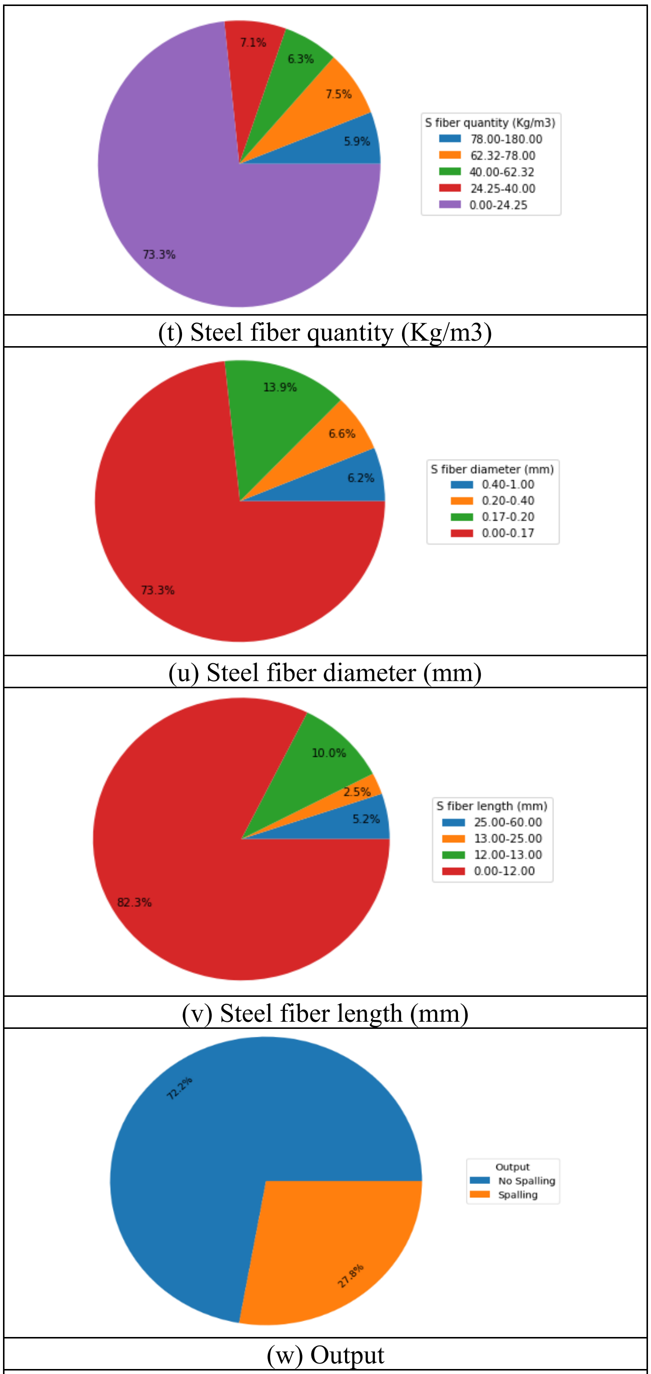
(o) Length (mm)

(p) Shape

(q) PP fiber quantity (Kg/m3)

(r) PP fiber length (mm)

(s) PP fiber diameter (μm)



(t) Steel fiber quantity (Kg/m3)

(u) Steel fiber diameter (mm)

(v) Steel fiber length (mm)

(w) Output

Fig. 1 Summary of graphical distribution of the fire-induced spalling of the concrete database

considered. It performs well for classification problems and can handle non-linear relationships and missing values. However, DT is sensitive to small changes in the data, which can severely affect the performance and destabilize the tree. Below are the settings used to develop the DT model: Criteria: Gini as the splitting criterion, a maximum depth of 10, no limit on the maximum number of features considered (None), a minimum of 2 samples required in each leaf, a minimum of 4 samples required to split an internal node, and a random splitter.

3.1.5 Extra Trees Classifier (ETC)

ETC is an ensemble learning method that can be used for classification and regression models (Geurts *et al*. 2006). It constructs multiple decision trees by using the entire dataset to select the feature to build the trees randomly but chooses the best split threshold randomly rather than optimizing it. This results in faster training times and increased diversity among the trees, which can help to reduce overfitting. The best parameters for the created model are identified as follows: the splitting criterion is set to ‘gini,’ the maximum depth of the tree is left unrestricted (None), the maximum features considered at each split are determined automatically (auto), and the number of estimators in the model is set to 100.

3.1.6 Gaussian naïve Bayes (GaussianNB) & BernonaïveNaive Bayes (BernoulliNB)

GaussianNB (Duda *et al*. 2001) and BernoulliNB (Domingos *et al.* 1996) are implementations of the Naive Bayes algorithm that is used particularly for classification problems.

On the one hand, GaussianNB performs well with classification tasks. The algorithm estimates the mean and standard deviation of each factor/class and incorporates them to calculate the probability of each class based on the examined factors. On the other hand, BernoulliNB is mainly used for binary classification tasks. Hence, it is expecting a binary outcome. One of the main concepts in BernoulliNB is that it penalizes the model for the NA values in any features, considering it as meaningful information rather than ignoring it, which is one of the differences between the two discussed Naive Bayes. Also, both algorithms may underperform when the Naïve’ independence assumption’ is severely violated. The GaussianNB was tuned with the following parameter: var_smoothing = 3.5e-08. While the BernoulliNB was tuned with the following parameters: alpha = 1.0 and binarize = 0.0.

3.1.7 K-Nearest Neighbors Algorithm (KNN)

This algorithm is compatible with classification and regression problems but is extensively used for classification tasks rather than regression (Fix 1985). The algorithms categorize the new data points by calculating the distances between the new data points and the training data. They then select the closest points and use their class to predict the new data point. It is a nonparametric algorithm that can handle non-linear relationships. Also, choosing the optimum k value severely influences the model’s performance. Hence, choosing the optimum K value can be a complex task. The K-Nearest Neighbors (KNN) model is applied with the following parameters: algorithm = 'ball_tree’, leaf_size = 30, metric = 'manhattan’, n_neighbors = 3, p = 1, and weights = 'uniform’.

3.1.8 Light Gradient Boosting Machine (LGBM)

LGBM was developed by Microsoft in 2016; since then, it has gained popularity due to its rapid processing speed and high performance, ability to handle missing data and categorical data, and ability to deal with non-linear relationships (Ke *et al*. 2017). It uses a gradient-boosting framework based on decision tree algorithms. However, it is designed to perform better because their trees expand from one leaf side (vertically), while other boosting trees split horizontally (one level of leaves by another). In addition, LGBM is known for its need for regularization techniques to minimize overfitting, which is a known downside for LGBM due to the vertical expansion of the tree, especially in small datasets. The LGBM classifier is set with the following parameters: a learning rate of 0.5, a maximum depth of 7, a binary classification objective, 500 estimators, and a random state of 42.

3.1.9 Linear Discriminant Analysis (LDA)

LDA is a linear classification algorithm that is popular in reducing the dimensionality reduction technique, which has become essential in the ML domain due to the existence of high dimensional databases (Fisher 1936). Ronald Fisher introduced the idea in the 1930s, which was applied to a 2-dimensional problem before developing it for multi-dimensional tasks. It aims at separating and reducing a database that consists of multi-features (each feature represents a dimensional plane), and the goal is to project the multi-dimensional features planes into 2-D or 3-D planes and then separate these planes based on the label class (outcome) and therefore, a human can easily understand the generated plots. It assumes that the data is normally distributed within each class and that the covariance matrices are equal for all classes, which may not always be accurate.

3.1.10 Logistic Regression (LR)

Unlike linear regression, LR works with classification problems when the output is categorical (usually a binary classification) (Cox 1958). It is a statistical method that determines the occurrence probability of the output class (i.e., fail, pass) given a set of input features. The LR model is configured with the following settings: the inverse of the regularization strength is set to 29.8, class weights are not specified (None), intercept fitting is enabled (fit intercept=True), the maximum number of iterations is set to 100, the penalty used is L1 regularization (penalty = l1), the solver used is Liblinear (solver = liblinear), the tolerance is set to 0.0001, and warm start is enabled (warm start = True).

3.1.11 Random Forest (RF)

An ensemble ML approach can be used for classification and regression problems by generating several decision trees on different levels and using the average of the trees' outcomes to improve the predictive accuracy instead of depending on a single decision tree's outcome (Breiman 2001). This algorithm uses a bagging approach that selects various training data to ensure the generated decision trees are universal (accounts for the maximum variation of data points). However, it has higher computational complexity than single decision trees and may be less interpretable. The random forest classifier is tweaked with the following configurations: 500 estimators, a maximum depth of 10, a minimum of 2 samples required to split an internal node, and a random state of 42.

3.1.12 SPINEXClassifier (SPINEX)

The SPINEX algorithm (Naser *et al*. 2023) is designed for interpretable regression and classification tasks. It starts by implementing essential preprocessing techniques to ensure the database is clean. Then, it calculates the distances between the two samples' features and assigns weights based on similarity using the Gaussian kernel function to accommodate single or ensemble models to distinguish between the complexity of the database. This algorithm builds on the importance of neighbor-based features to measure each feature's contribution, considering the influence of neighboring instances. SPINEX aims to provide transparent and interpretable predictions. The SPINEXClassifier is configured with a neighbor count of 3, a distance threshold of 0.05, no ensemble method (None), and a 'Manhattan' metric for distance calculation.

3.1.13 Support Vector Machine (SVM)

SVM is a family of algorithms mainly used for classification (Cortes and Vapnik 1995). It creates a plane with the number of features as the dimension of the generated plane and plots the raw data. The goal is to tie the data point to a specific coordinate to ease the complexity of classifying the data. SVM can handle non-linear correlations but uses kernel functions to plot the raw data points into a multi-dimensional plane. Although SVM is effective in high-dimensional spaces, large databases have high energy and time consumption.

3.1.14 Xtreme Gradient Boosting (XGBoost)

A decision-tree-based algorithm that uses a gradient boosting framework widely known for its high performance, speed, and scalability (Chen and Guestrin 2016). This algorithm starts by assigning a value to the tree leaves to build shallow decision trees. The error from these trees will be used to build the subsequent shallow tree, which will perform better because the model learns from the previous errors. Once the tree reaches the end of the training data, the model will use the predictions of the generated trees to make a prediction. In this analysis, the developed algorithm was tweaked with the following settings: Col sample by tree value of 1.0, a learning rate of 0.1, a maximum depth of 7, a minimum child weight of 1, 100 estimators, and a subsample of 0.8.

3.2 Technical details

Here, we delve into the technical aspects of the model deployment and discuss the metrics used to evaluate the performance of the models. Also, the approaches are taken to prevent unfavored biases that models can encounter during the training phase.

First, the algorithm's hyperparameter configurations should be optimized for each algorithm to achieve the optimum performance. Hence, these algorithms need to be fine-tuned. Grid Search Cross-Validation' GridSearchCV' technique (Sklearn 2023) was used to automate the search for the best hyperparameter values for a model to optimize its performance.

Along the same lines, the model's performance was evaluated based on four metrics: accuracy score, area under the curve (AUC), log loss, and cross-validation. The accuracy score is intuitive, representing the ratio of the prediction observation to the ground truth.

Table 1 Comparison of various ML algorithms

Algorithm	Family	Most common real-world applications	Complexity	Limitation	Classification or regression	Sensitive to outliers	Interpret-ability
Adaptive Boosting (Freund and Schapire 1997)	Boosting	Face detection, biology, speech processing	High	Sensitive to noisy data and outliers	Both	Yes	Low
Bernoulli Naive Bayes (Domingos *et al.* 1996)	Probabilistic	Text classification, spam filtering	Low	Assumes independence of features	Classification	No	High
Categorical Boost (Prokhorenkova *et al.* 2017)	Boosting	Tabular data, search ranking, ads ranking, recommendation systems	High	Requires parameter tuning	Both	Yes	Low
Decision Tree (Quinlan 1986)	Tree-based	Tabular data, customer segmentation, decision-making problems	Medium	Prone to overfitting	Both	No	High
Extra Trees Classifier (Geurts *et al.* 2006)	Tree-based	Tabular data, bioinformatics, genomics	High	Randomness can lead to lower accuracy	Both	No	High
Gaussian Naive Bayes (Duda *et al.* 2001)	Probabilistic	Text classification, spam filtering	Low	Assumes independence of features	Both	No	High
K-Nearest Neighbors (Fix 1985)	Instance-based	Tabular data, recommendation systems, concept search	Medium	Computationally intensive as the dataset grows	Both	Yes	Moderate
Light Gradient Boosting Machine (Ke *et al.* 2017)	Boosting	Tabular data, search ranking, ecology, anomaly detection	High	Requires parameter tuning	Both	Yes	Low
Linear Discriminant Analysis (Fisher 1936)	Discriminant analysis	Face recognition, image retrieval	Medium	Assumes normal distribution and equal covariance matrices	Both	Yes	Moderate
Logistic Regression (Cox 1958)	Regression	Tabular data, credit scoring, measuring campaign effectiveness	Low	Requires feature scaling, not suitable for non-linear problems	Classification	Yes	High
Random Forest (Breiman 2001)	Tree-based	Tabular data, banking, stock market, e-commerce	High	Requires parameter tuning	Both	No	Moderate
Support Vector Machine (Cortes and Vapnik 1995)	Kernel-based	Image recognition, text categorization, bioinformatics	High	Requires parameter tuning, not suitable for large datasets	Both	Yes	Low
Xtreme Gradient Boosting (Chen and Guestrin 2016)	Boosting	Tabular data, anomaly detection, predictive	High	Requires parameter tuning	Both	Yes	Low

		modeling, search ranking					
SPINEX Classifier (Naser *et al*. 2023)	Ensemble learning	Tabular data	Medium	Energy consumer	Both	High	Moderate
Artificial Neural Network (McCulloch and Pitts 1943)	Deep learning	Image recognition, natural language processing, speech recognition	Very high	Requires substantial data, difficult to interpret	Both	Yes	Low

Table 2 Summary of models' evaluation metrics scores for both training and testing sets

Complete database (100%) Testing Set (Stratified)							
	Testing set			Training set			Cross validation score
Model	Accuracy	AUC	Log-loss	Accuracy	AUC	Log-loss	
Light Gradient Boosting Machine	93.5%	91.3%	2.36	98.7%	98.1%	0.45	96.4%
Xtreme Gradient Boosting	93.8%	92.2%	2.24	98.9%	98.4%	0.40	96.1%
Random Forest	91.5%	89.6%	3.06	99.2%	98.8%	0.30	94.4%
Extra Trees Classifier	90.2%	87.7%	3.53	99.2%	98.6%	0.30	94.0%
Categorical Boost	92.5%	90.3%	2.71	98.2%	97.1%	0.66	92.0%
Adaptive Boosting	87.3%	83.1%	4.59	87.8%	81.9%	4.40	91.5%
SPINEX Classifier	85.6%	82.3%	5.18	98.5%	97.8%	0.56	89.8%
Decision Tree	87.3%	84.1%	4.59	96.9%	94.7%	1.11	88.4%
Linear Discriminant Analysis	80.7%	75.7%	6.95	79.5%	70.9%	7.38	83.9%
Support Vector Machine	81.0%	75.3%	6.83	79.5%	71.2%	7.38	82.9%
Gaussian Naive Bayes	76.5%	77.6%	8.48	75.2%	75.6%	8.95	78.6%
Bernoulli Naive Bayes	72.5%	58.4%	9.89	77.1%	61.3%	8.24	73.6%
Logistic Regression	83.3%	77.9%	6.01	81.1%	72.6%	6.82	70.3%
K-Nearest Neighbors	85.6%	80.7%	5.18	91.7%	88.4%	2.98	42.9%
Deep Learning	85.6%	92.0%	0.41	92.8%	98.2%	0.17	0.0%

Although users prefer higher accuracies, this metric might overestimate the model’s accuracy when using an imbalanced target feature. To overcome the imbalanced target feature issue, the AUC metric comes in handy, as it can differentiate between positive and negative classes, leading to an accuracy that accounts for both classes. In AUC, the score of 100% represents a perfect classification. In addition, the Log loss or cross-entropy loss metric was also used to quantify the model performance and show how surprised the model is with the predicted class. When a model makes a prediction, it provides a probability value for each class. A good model should have a lower log loss value, with 0 representing a perfect log loss.

The cross-validation technique checks the model’s vulnerability to failure in generalizing a pattern and reducing bias (i.e., overfitting), typically by splitting the entire database into K folds. In this validation process, each of the K folds will be used to test the model, and the remaining folds will be used to train and fit the model. Therefore, the model’s performance will be controlled. This work implemented the cross-validation technique to evaluate the model on a validation dataset by setting the K value to 5 folds.

All models were trained on split data with a training-to-testing sets ratio of 70%:30% to help us prevent any bias the model could encounter. The testing set is then used to evaluate the trained model and validate its results. In addition, each of the created models was implemented with one or more of the regularization techniques (i.e., early stopping, dropouts) to reduce the vulnerability of models to overfit.

4. Discussion

This section consists of six subsections. The first subsection presents our analysis using the whole database. In the subsequent subsections, subsets from the original database of 75% and 50% of the original database by maintaining their original data distribution (stratified database) are examined. These subsequent subsections focus on the behavior of the models when changing the distribution of the target variable to be normally distributed (customized database). Furthermore, a hybrid analysis will be repeated using the key factors influencing spalling for the top-performing models, pointing out the main differences and impacts on the model’s performance and robustness. Finally, the DL model was separately addressed in a subsection as the model showed biased insights.

4.1 Whole database (100%)

The evaluation results in Table 2 show a variation in the models’ performance. Based on the testing results, LGBM, XGBoost, and RF outperformed all the other models. This table also shows that the CatBoost, ETC models performed well, with an accuracy of 92.5% and 90.2%, AUC of 90.3%, 87.7%, Log loss of 2.71, 3.53, and a cross-validation score of 92.0%, 94.0%, respectively.

The DL model presented exciting results, with a relatively low accuracy score of 85.6% and a strong AUC score of 92%, and surprisingly, achieving the lowest log loss of 0.41 among all the models. This suggests that, despite relatively low accuracy, the model shows a high confidence level in its prediction. In contrast, models such as Gaussian NB and Bernoulli NB underperformed significantly, as indicated by their lowest accuracy and their highest log loss. The remaining models showed varied results. Some models, such as AdaBoost and DT, demonstrated strong performance, while others, like SVM and LR, performed poorly concerning other models.

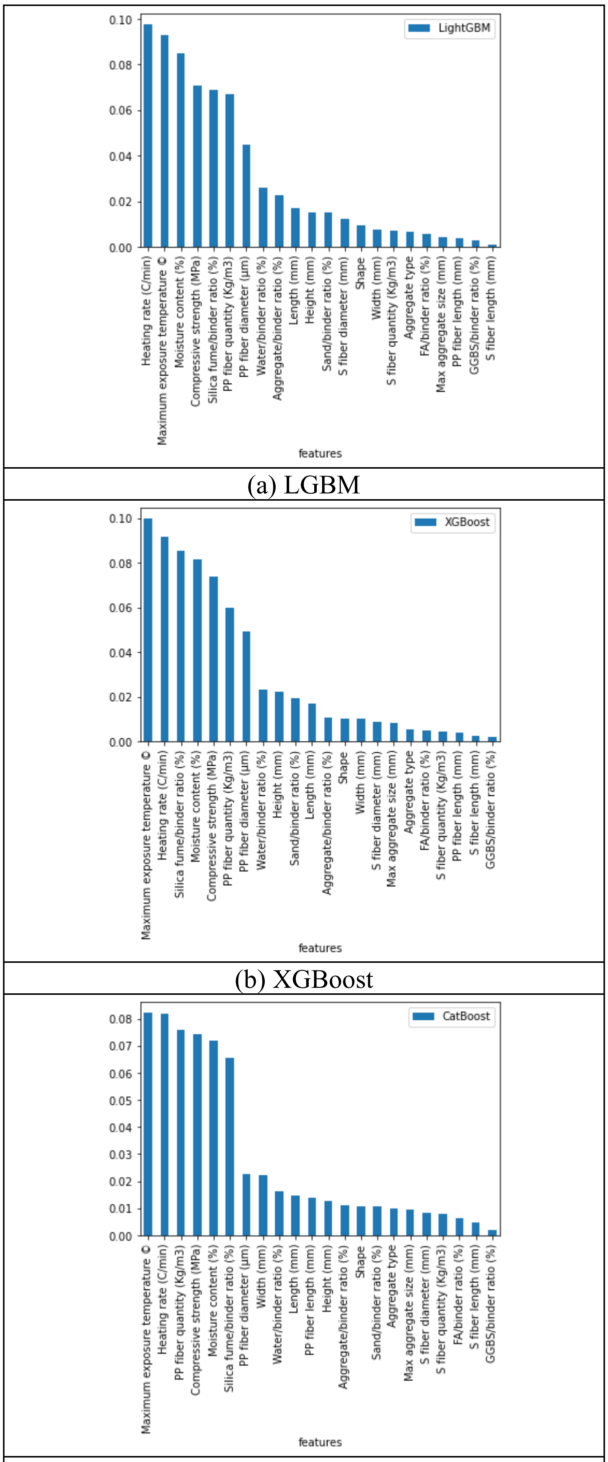
By looking at the training set’s evaluation metrics, results differ between training and testing scores, which can be explained by the large number of samples used for training purposes compared to the testing samples. The less accurate models such as LR, SVM (kernel = Linear), GaussianNB, and BernoulliNB metrics in both training and testing sets were not highly impacted. However, one can see that all the evaluation metrics of XGBoost, RF, LGBM, ETC, and SPINEX demonstrated strong results. It is noteworthy that, despite the regularity in the model’s performance across the evaluation metrics, none of the models tops all the metrics simultaneously, which validates the assumption of using more than one metric to evaluate the model.

Along the same lines, some models are more sensitive than others to the data being processed, which can affect the model’s behavior and performance. Therefore, we decided to evaluate feature importance based on the best-performing models. Two groups were considered, firstly, gradient-boosting algorithms, including LGBM, XGBoost, and CatBoost, because they are derived from the same gradient-boosting family. Secondly, RF, and ETC are learning ensemble methods; hence, using them in the same group is more rational.

4.1.1 LGBM, Catboost, and XGBoost

As discussed, the gradient boosting models share the same framework by building a series of learners’ decision trees; however, they differ in the expansion shapes and rates. These models provide feature importance plots based on how often a specific feature is used to decide on the generated decision trees. One can see exciting insight by looking at Fig. 2. All the models agreed to a large extent on the top 10 critical factors influencing fire-induced spalling of concrete and identifying maximum exposure temperature, compressive strength, heating rate, moisture content, PP fiber quantity, and silica fume/binder ratio as the top 6 influencing factors to the spalling.

Alternatively, by looking into the lower half of the top 10 parameters, one can see that the focus was shifted to the geometric and concrete mix properties factors. For example, LGBM and XGBoost considered water/ binder ratio, sand/binder ratio, length, height, and PP fiber diameter as the lower half of the top influencing factors. CatBoost, on the other hand, partially agreed with some factors, such as the length and the PP fiber diameter, and introduced the PP fiber length and the aggregate types as critical factors.



(a) LGBM

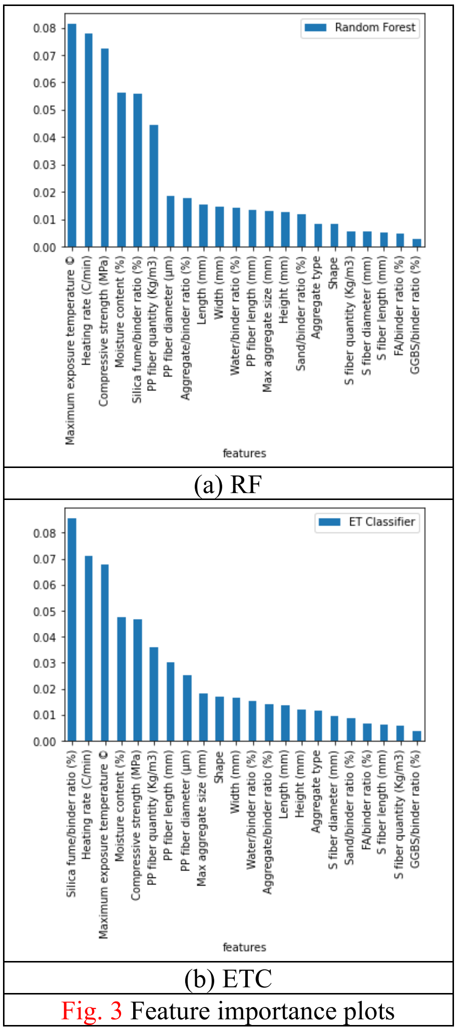
(b) XGBoost

(c) CatBoost

Fig. 2 Feature importance of top-performing models

4.1.2 RF, ETC

By considering the sequence of the feature importance of the RF, ETC models (Fig.3), it is evident that they share the same top 6 key factors influencing fire-induced concrete spalling with the top 3 models. Despite the slight changes in the factors' positions among RF, and ETC, they shared the same top 6 factors, which can be explained by the similarity in building the decision trees. Also, the lower half of the critical factors were aggregate/ binder ratio, the length, width, and PP diameter for the RF model, in contrast with the PP fiber length, PP fiber diameter, and Shape of the specimen and the maximum aggregate size for the ETC model seem to be given a less importance influence on the prediction.



(a) RF

(b) ETC

Fig. 3 Feature importance plots

4.2 75% of the database

To check the robustness of the developed models, we used a reduced dataset and evaluated the models' response through accuracy and feature importance. In this subsection, the data points were randomly selected from the database. However, the main criteria considered are maintaining the exact distribution of the target variable (spalling, no spalling) and the top 6 critical factors ─ see Fig. 4. Findings from this approach revealed a slight variability in the model's performance. The overall evaluation was not severely impacted by decreasing the number of selected samples, indicating that the models could capture the pattern of the prediction function extracted from the data.

The testing scores in Table 3 show that XGBoost, LGBM, and RF cross-validation scores were almost constant, with a slight increase in the RF score. They consistently maintain high accuracy, AUC, and low Log Loss scores on the testing and training sets. This suggests

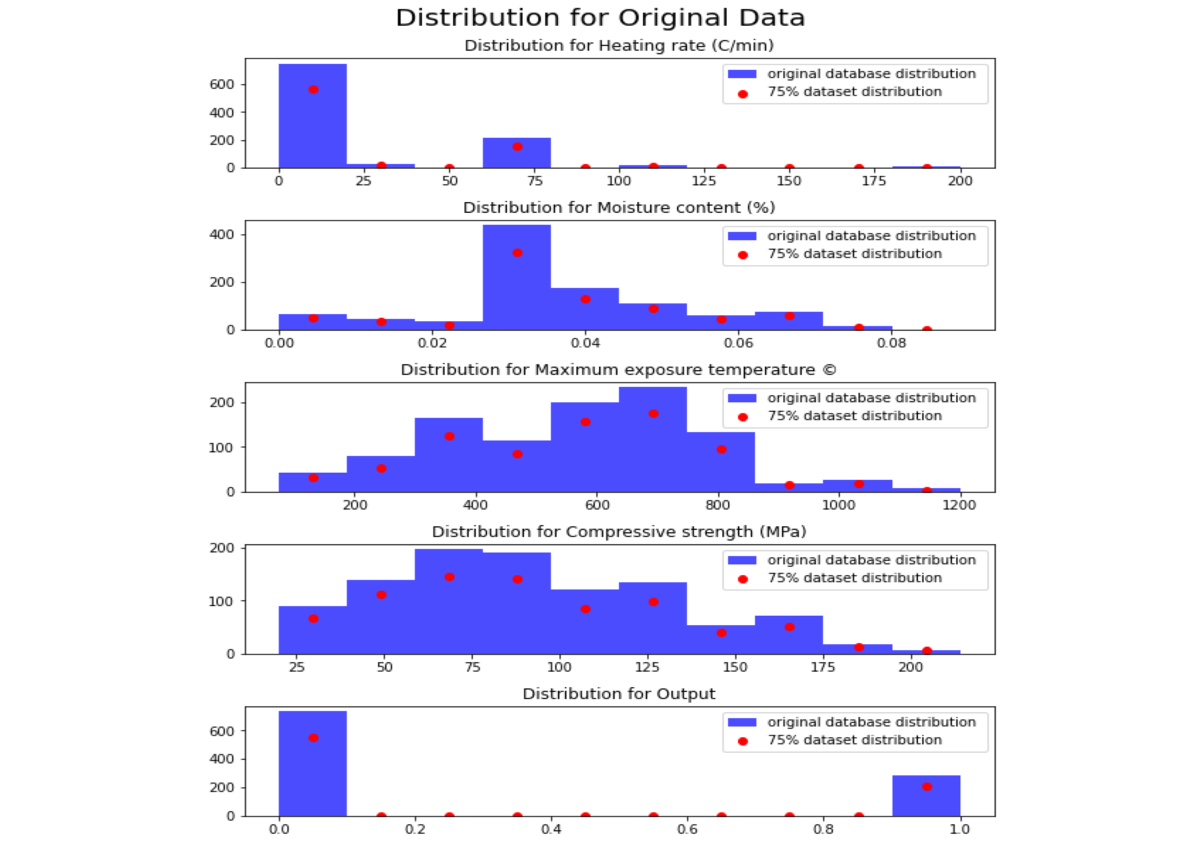


Fig. 4 Comparison between the database distribution when using 75% of the original database

Table 3 Summary of models' evaluation metrics scores for both training and testing sets

Reduced dataset (75%) Testing Set (Stratified)							
	Testing set			Training set			Cross validation score
Model	Accuracy	AUC	Log-loss	Accuracy	AUC	Log-loss	
Xtreme Gradient Boosting	90.4%	85.7%	3.45	98.9%	98.6%	0.40	96.0%
Light Gradient Boosting Machine	89.6%	84.1%	3.76	98.7%	98.1%	0.47	95.6%
Random Forest	89.1%	82.4%	3.92	99.1%	98.7%	0.34	94.9%
Extra Trees Classifier	88.3%	82.3%	4.23	99.1%	98.3%	0.34	94.6%
Adaptive Boosting	87.0%	80.9%	4.70	90.3%	87.0%	3.51	91.7%
Categorical Boost	87.0%	79.4%	4.70	96.4%	94.6%	1.28	91.1%
Decision Tree	87.8%	80.5%	4.39	97.9%	96.7%	0.74	87.5%
SPINEX Classifier	85.2%	80.6%	5.33	98.7%	98.7%	0.47	88.4%
K-Nearest Neighbors	81.3%	74.6%	6.74	91.8%	88.5%	2.97	87.5%
Support Vector Machine	81.3%	72.2%	6.74	79.8%	73.5%	7.29	82.5%
Linear Discriminant Analysis	79.1%	69.2%	7.52	80.3%	73.9%	7.09	83.8%
Gaussian Naive Bayes	77.0%	72.0%	8.31	76.0%	76.1%	8.64	78.6%
Logistic Regression	82.6%	74.5%	6.27	81.8%	75.8%	6.55	70.3%

Bernoulli Naive Bayes	71.3%	56.1%	10.3	75.3%	62.5%	8.91	73.0%
Deep Learning	85.7%	88.9%	0.48	94.4%	99.0%	0.12	0.0%

that these models effectively generalize the trend, reducing the likelihood of overfitting. The DL has maintained a relatively good accuracy, an impressive AUC, and a log loss score, indicating it provides highly confident predictions.

Despite their relatively acceptable performance, some models, such as ETC and CatBoost, experienced a retracement in the overall performance. Some models like GaussianNB, BernoulliNB, and LR show relatively lower performance than others. This could be due to the linear nature of these models; hence, they are struggling with the complexity of the data. In addition, models such as DT showed a performance metrics score significantly lower than in the training set, which might be a symptom of overfitting.

Overall, the model performances seem to have decreased slightly compared to the 100% dataset, which is expected given that the models now have less data to learn from. Still, the performances are relatively high, indicating that the models could learn effectively from a reduced dataset. Alternatively, the evaluation metrics of the training set remained within the range of over 95% for the three models.

In addition, by looking at Fig. 5, the feature importance plot shows that XGBoost, LGBM, and RF feature importance did not change by decreasing the database number of samples. Still, ETC and CatBoost followed the same observation on the key influencing factors. As one can see, maximum exposure temperature, compressive strength, heating rate, moisture content, PP fiber quantity, and silica fume/binder ratio were also identified as the top 6 influencing factors to the spalling. The above discussion reinforces the concept of conducting a sensitivity analysis as the divergent rankings of the influencing factors showcase the alternative ways the model can explain, interpret, and learn from data. Hence, engineers can observe these differences and analyze them. However, in this case, the top influencing factors remained consistent, but with slight changes in the magnitude of importance.

4.3 50% of the database

The same analysis was carried out with a 50% data reduction. The selection of the data points followed the same approach taken previously by maintaining the distribution of the target variable and the top influencing factors—see Fig. 6.

As expected, a general decline in the overall models' performance is seen in Table 4. Most of the top-performing models' accuracy scores retraced to the low 80s, a significant drop in the AUC score and log loss score indicating the need for more data to train the model. Despite their relatively good performance, there is still a significant gap between the training and testing scores, which might be concerning as it might indicate some form of overfitting. Interestingly, it is clear that the LGBM and DL models were the top performers, demonstrating relatively high accuracy and strong AUC scores. Surprisingly, DL distinguishes itself with the lowest log loss, indicating high confidence in its performance. Alternatively, the GaussianNB and BernoulliNB models demonstrated a low performance, suggesting they might struggle if used in a complex, non-linear dataset. Fig. 7 illustrates the feature importance for this subsection of the dataset and clearly shows that the key factors remained the same.

4.4 Customized database reduction

The main concept in this reduction approach is creating a 75% and 50% reduced dataset where the outcome (i.e., spalling, no spalling) is balanced and choosing a wide range of samples spanning over the factors' spectrum based on the key factors influencing concrete spalling. The aim is to compare the performance of the top performing models when the distribution of the database is maintained versus when the distribution is reduced in a systematic approach to achieving a normal distribution, and the outcome parameter is considered balanced.

On the one hand, the 75% data frame showed a relatively good performance. XGBoost, LGBM, ETC, and RF achieved the highest cross-validation score ranging between 93-95%, an accuracy score of more than 90%, and an AUC score of slightly under 90% with an exception for ETC, scoring 90% as the Highest AUC score among all the models. Despite the significant reduction in data size, the evaluation of the customized dataset was not severely impacted except for the AUC scores. Fig. 8 shows that the balanced 75% dataset achieved a relatively similar accuracy score and cross-validation score as opposed to the stratified 75% dataset. However, the stratified case's AUC scores significantly dropped to the mid-80s. In addition, the XGBoost model was the least impacted by the data customization regarding the AUC and accuracy scores.

On the other hand, a dataset with a 50% reduction has been created with the same above-described principle. The overall performance shows a similar trend as in the 75% scenario but with some variations. XGBoost, LGBM, and RF models still managed to secure top spots with accuracy and AUC scores near or above 87%. It is noteworthy that, despite the further reduction in data size, these models showed resilience, maintaining a relatively high level of performance. It is worth noting that the comparison between the 50% balanced and the stratified datasets (Fig. 9) reflects the same pattern as the 75% cases. However, the AUC scores were negatively impacted due to the variation of the data size and distribution, but the overall performance of the models improved compared to the stratified 50%.

4.5 Complete database (Top 6 factors)

In this subsection, the ML models were trained on the complete database by only considering the top 6 factors influencing concrete spalling: maximum exposure temperature, compressive strength, heating rate, moisture content, PP fiber quantity, and silica fume/binder ratio. To check the sensitivity of the models with fewer features, it would be interesting to compare the results from the full features set analysis versus the critical features set analysis — Table 5, models that perform well on both the full features set and the critical features set are likely to be capturing the underlying mechanism/relationships of the Spalling phenomenon. As shown in Fig. 10, it is clear that despite

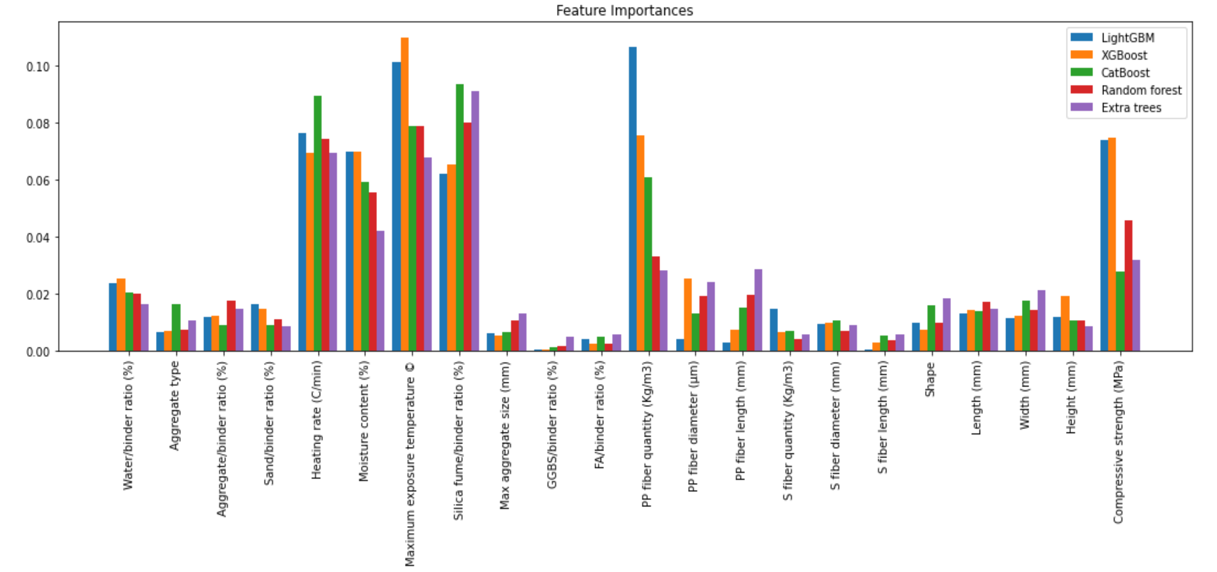


Fig. 5 Combined feature importance of the top-performing models (75% of the database)

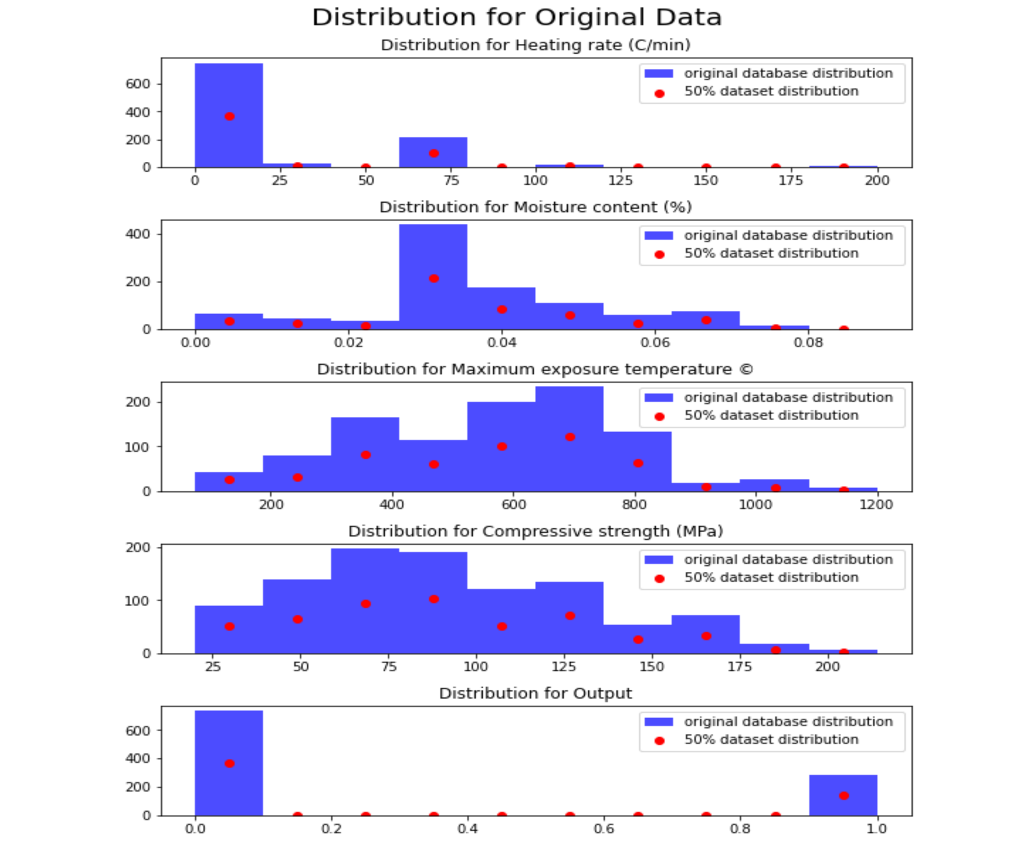


Fig 6 Comparison between the database distribution when using 50% of the original database

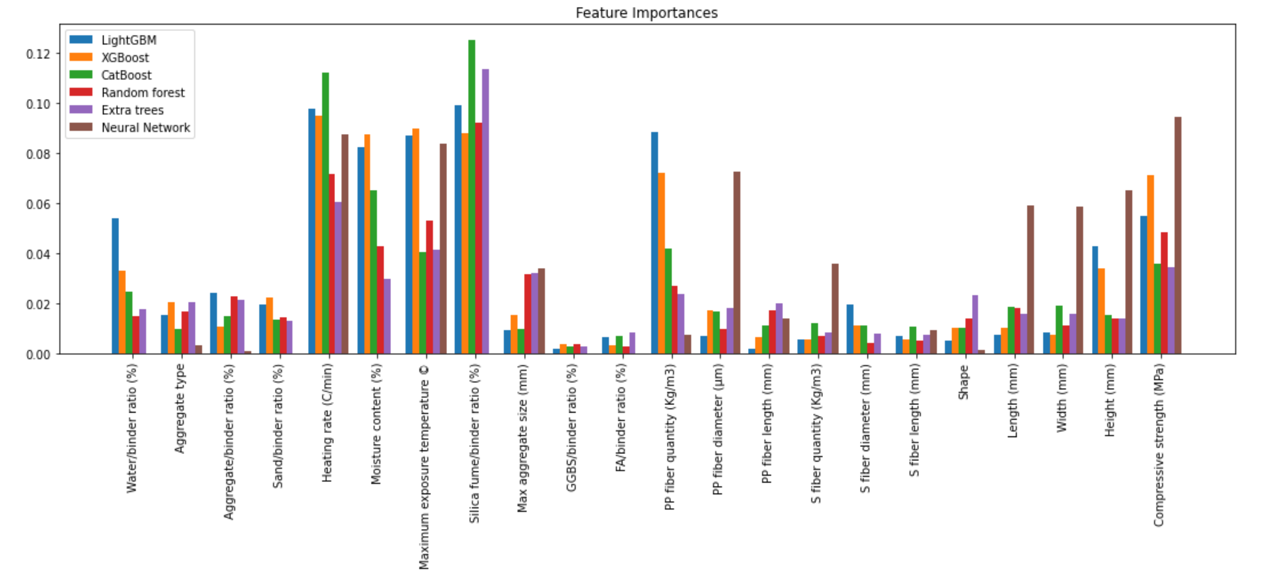


Fig. 7 Combined feature importance of the top-performing models (50% of the database)

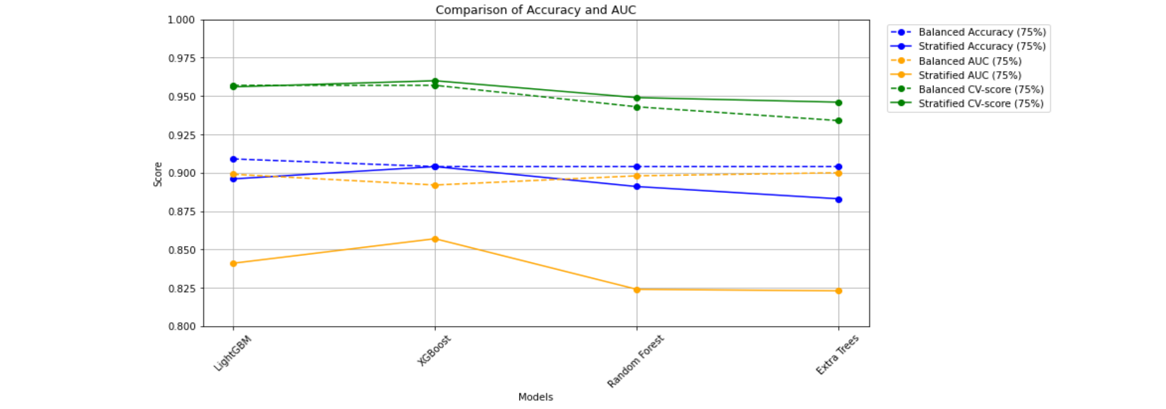


Fig. 8 Comparison between the top-performing models in the two cases of 75% of the database input

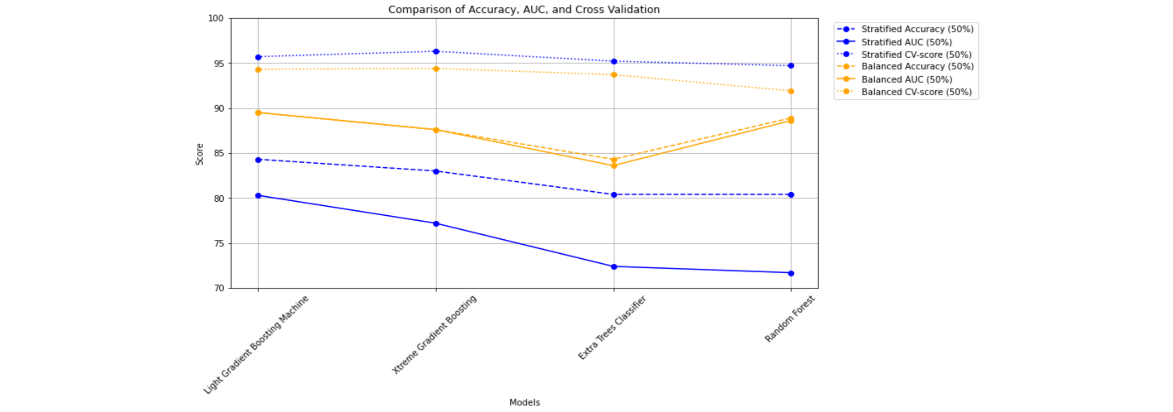


Fig. 9 Comparison between the top-performing models in the two cases of 50% of the database input

Table 4 Summary of models' evaluation metrics scores for both training and testing sets

Reduced dataset (50%) Testing Set (stratified)							
	Testing set			Training set			Cross validation score
Model	Accuracy	AUC	Log-loss	Accuracy	AUC	Log-loss	
Light Gradient Boosting Machine	84.3%	80.3%	5.65	99.4%	99.0%	0.20	95.7%
Xtreme Gradient Boosting	83.0%	77.2%	6.13	99.4%	99.0%	0.20	96.3%
Decision Tree	85.0%	79.3%	5.42	97.2%	94.9%	1.01	88.2%
Categorical Boost	83.0%	76.4%	6.13	96.3%	95.0%	1.32	91.5%
Extra Trees Classifier	80.4%	72.4%	7.07	99.7%	99.5%	0.10	95.2%
Random Forest	80.4%	71.7%	7.07	99.7%	99.5%	0.10	94.7%
Adaptive Boosting	79.1%	73.0%	7.54	93.5%	90.9%	2.33	91.8%
SPINEX Classifier	80.4%	72.4%	7.07	99.4%	99.3%	0.20	88.3%
K-Nearest Neighbors	79.7%	72.0%	7.30	90.7%	87.4%	3.34	88.3%
Support Vector Machine	77.8%	72.8%	8.01	86.5%	83.2%	4.86	82.3%
Linear Discriminant Analysis	76.5%	69.7%	8.48	82.9%	77.6%	6.18	83.9%
Logistic Regression	78.4%	71.8%	7.77	84.8%	79.2%	5.47	70.3%
Gaussian Naive Bayes	67.3%	65.6%	11.78	79.2%	76.6%	7.49	78.6%
Bernoulli Naive Bayes	71.9%	60.6%	10.1	78.1%	69.3%	7.90	72.9%
Deep Learning	84.3%	83.7%	0.72	95.8%	99.4%	0.09	0.0%

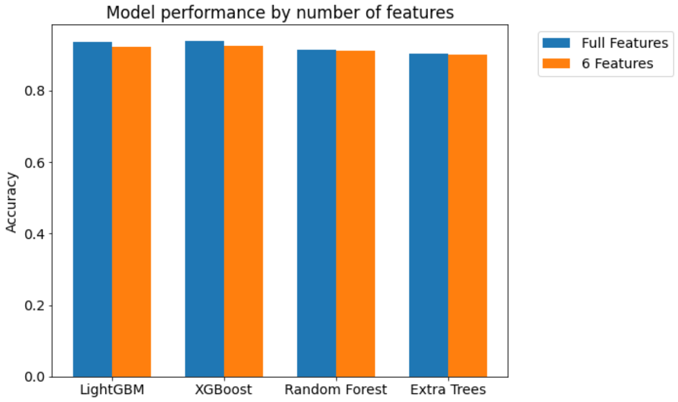


Fig. 10 Comparison between the top-performing models in the full and reduced features databases

Table 5 Summary of models' evaluation metrics scores for both training and testing sets of the complete database (key features)

Complete Database (top 6 features)							
	Testing set			Training set			Cross validation score
Model	Accuracy	AUC	Log-Loss	Accuracy	AUC	Log-Loss	
Light Gradient Boosting Machine	92.2%	89.4%	2.83	97.5%	96.0%	0.91	95.1%
Xtreme Gradient Boosting	91.5%	89.0%	3.06	97.9%	96.4%	0.76	94.7%
Random Forest	90.5%	87.6%	3.42	98.3%	97.5%	0.61	93.5%
Extra Trees Classifier	89.2%	85.2%	3.89	98.3%	97.2%	0.61	94.1%
Categorical Boost	89.9%	86.3%	3.65	93.5%	89.6%	2.33	89.5%
SPINEX Classifier	85.6%	81.4%	5.18	97.3%	96.0%	0.96	87.4%
Decision Tree	85.3%	82.7%	5.30	96.4%	94.4%	1.31	86.2%
Adaptive Boosting	83.0%	77.6%	6.13	84.3%	77.2%	5.66	89.9%
K-Nearest Neighbors	83.3%	78.5%	6.01	90.9%	87.3%	3.29	86.9%
Linear Discriminant Analysis	78.8%	68.1%	7.66	78.1%	66.1%	7.89	80.7%
Logistic Regression	80.1%	69.3%	7.19	78.0%	66.2%	7.94	77.6%
Gaussian Naive Bayes	76.1%	71.5%	8.60	76.4%	69.3%	8.49	78.3%
Support Vector Machine	76.1%	63.4%	8.60	76.4%	60.8%	8.49	80.2%
Bernoulli Naive Bayes	69.9%	50.0%	10.8	73.2%	50.0%	9.66	69.1%
Deep Learning	82.7%	89.2%	0.38	83.6%	88.3%	0.37	0.0%

the reduction in features, the top-performing models for the complete database with all features and the critical features set remain largely the same with a general slight decline in performance. Still, the models continued to perform well, indicating the impact of the selected features on the spalling prediction. Even with a reduction in features, LGBM, XGBoost, RF, ETC, and CatBoost models have shown resilience and robustness, dominating the overall performance of the other models in this study. Also, the comparison between the training and testing sets of the DL model seems to be close enough, which eliminates any symptoms of overfitting or data memorization issues. Similarly, models such as GaussianNB and BernoulliNB continue to perform poorly in both scenarios, confirming that they are unlikely to be appropriate for this complex problem or database.

4.6 DL investigation

One model of particular note was the DL. We evaluated the model across various scenarios, similar to the previous subsections, using the complete database, 75% and 50% of the original database size. In addition, we maintained the original distribution of the data (stratified) and examined the performance by customizing the data distribution (customized). Remarkably, the model's accuracy remained consistently high across these scenarios. According to evaluation metrics from Table 2 - Table 5, one can see that, despite changes in data size or distribution, the accuracy scores slightly declined to 84% when using the stratified 50% of the database while remaining at 87% in both the 75% and the complete database scenarios.

Moreover, similar accuracy scores were observed when customizing the reduction of data to become a normally distributed dataset. In line with this, the log loss metric achieved the lowest score among all models in all scenarios, remaining below 1, indicating that the model was very confident when making the prediction. Also, the AUC scores were high, remaining above 83% for the 50% data scenario and above 87% for other types of data reduction.

Despite the high and robust performance presented by the model, a 'non-sense' divergence was observed upon examining the DNN's feature importance. In the complete database scenario, the feature importance plot for the DNN model (see Fig. 11) comes with a different insight. The model identified the heating rate, maximum exposure temperature, and compressive strength of concrete as three of the top 6 key factors matching the other models. Contrary to the widely accepted theories and domain knowledge, the DNN highlighted the specimen height, PP fiber diameters, maximum aggregate size, length, width, steel fiber quantity, and PP fiber length as 7 of the top 10 influencing factors. Notwithstanding their importance, the model did not highlight other factors, such as the moisture content. Instead, it was identified as the least important among all the 22 factors.

Also, these outcomes contradict the findings from the top-performing models. One can see that although the DL model performed well, the results are unexpected, which highlights the essential need for a detailed check and confirmation of the data used in predictive modeling. Also, we suspect that the significant divergence of the critical factors in the model is mainly due to the complexity of interpreting the DL models because of the complex learning techniques.

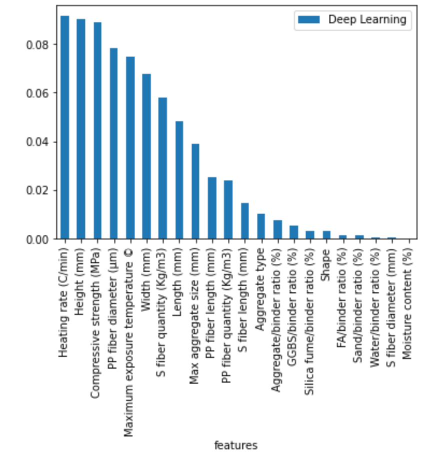


Fig. 11 DNN feature importance plot of the DL model

5 Conclusions

Overall, this work emphasizes the importance of conducting a sensitivity analysis of various ML algorithms and examining their robustness against data manipulation techniques. Particularly, LGBM, XGBoost, RF, ETC, and CatBoost showed robustness in accurately predicting concrete spalling due to fire. In addition, these models performed well across various data scenarios and effectively identified the key factors influencing the fire-induced spalling of concrete. On the contrary, other algorithms that adopt more straightforward learning approaches performed poorly against the same data scenarios, such as SVM, GussianNB, and BernoulliNB. These results underline the need for careful model selection, data preprocessing, and the usage of multiple approaches to verify the outcomes and the evaluation scores and validate them.

Hence, we believe that demonstrating that some models are more capable of predicting spalling and showing a consistently high performance is an indication of the robustness of these models and can be perceived as an opportunity to improve our spalling domain knowledge by integrating explainability and causality tools. This also opens a wide door for collaboration between structural fire engineers and data scientists to develop new algorithms that focus on the strong parts of the adopted models.

- XGBoost, LGBM, and RF were the best-performing models across the entire analysis, achieving an accuracy score of 93.5%, 93.8%, and 91.5%, respectively.
- The analysis shows that, in general, the evaluation metrics increase when using 100% of the data and decline with data reduction, which indicates the

need for more tests to improve the model's performance and provide a robust predictive model.

- While the balanced and stratified models yielded comparable accuracy and cross-validation scores, the AUC was significantly impacted by the 50% data reduction.
- The key factors influencing spalling are maximum exposure temperature, heating rate, moisture content, compressive strength, PP fiber quantity, and silica fume /binder ratio.
- The top-performing models achieved a relatively similar evaluation when training the models on the key factors and eliminating the other factors, indicating their high influence on the predictions.
- The sensitivity analysis revealed consistent critical factors across all the top-performing models with slight changes in the magnitude of importance, validating their critical role in understanding and predicting the spalling phenomena.
- DL model's feature importance diverged from those identified by other models and existing literature, which might open the doors for further investigation regarding the data parameters and sizes.
- The deep learning model provided a rare case where the model shows a high performance and results that do not match the existing literature.

References

Abo Sabah, S. H., N. L. Zainal, N. Muhamad Bunnori, M. A. Megat Johari, and M. H. Hassan. (2019), "Interfacial behavior between normal substrate and green ultra-high-performance fiber-reinforced concrete under elevated temperatures." *Structural Concrete*, 20 (6): 1896–1908.

Breiman, L. (2001), "Random forests." *Mach Learn*, 45 (1): 5–32.

Chen, T., and C. Guestrin, "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 10.1145/2939672.

Cortes, C., and V. Vapnik. (1995), "Support-vector networks." *Mach Learn*, 20 (3): 273–297.

Cox, D. R. (1958), "The Regression Analysis of Binary Sequences." *Journal of the Royal Statistical Society: Series B (Methodological)*, 20 (2): 215–232.

Domingos, P., M. P.-Proc. (1996), "Beyond independence: Conditions for the optimality of the simple bayesian classifer." *13th Intl. Conf. M. Learning, and undefined.*

"Duda, R. O., Hart, P. E., & Stork, D. G. (2001), - Google Scholar." Accessed June 17, 2023. https://scholar.google.com/scholar?hl=en&as_sdt=0%2C41&q=Duda%2C+R.+O.%2C+Hart%2C+P.+E.%2C+%26+Stork%2C+D.+G.+%282001%29.+Pattern+Classification+%282nd+ed.%29.+Wiley.&btnG=.

Dwaikat, M. B., and V. K. R. Kodur. (2010), "Fire induced spalling in high strength concrete beams." *Fire Technol*, 46 (1).

eugenics, R. F.-A. of, and undefined (1936), "The use of multiple measurements in taxonomic problems." *Wiley Online Library*, 7 (2): 179–188.

Fix, E. (1985), "Discriminatory analysis: nonparametric discrimination, consistency properties".

Freund, Y., and R. E. Schapire. (1997), "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting." *J Comput Syst Sci*, 55 (1): 119–139.

Geurts, P., D. Ernst, and L. Wehenkel. (2006), "Extremely randomized trees." *Mach Learn*, 63 (1): 3–42. *Springer*.

Ibrahimbegovic, et al (2010), "On modeling of fire resistance tests on concrete and reinforced-concrete structures." *techno-press.org.*

Kanema, M., P. Pliya, A. N.-J. of M. (2011), "Spalling, thermal, and hydrous behavior of ordinary and high-strength concrete subjected to elevated temperature." *American Society of Civil Engineers ascelibrary.org,* 23 (7): 921–930.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. (2017), "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." *Adv Neural Inf Process Syst*, 30.

Khoury, G. A. (2000), "Effect of fire on concrete and concrete structures." *Progress in Structural Engineering and Materials*, 2 (4): 429–447.

Khoury, G. A. (2015), "Passive fire protection of concrete structures." 2008.161.3.135. *Thomas Telford Ltd.*

Liu, J. C., and Z. Zhang. (2020), "A machine learning approach to predict explosive spalling of heated concrete." *Archives of Civil and Mechanical Engineering*, 20 (4): 1–25. Springer Science and Business Media Deutschland GmbH.

McCulloch, W. S., and W. Pitts. (1943), "A logical calculus of the ideas immanent in nervous activity." *Kluwer Academic Publishers*, 5 (4): 115–133..

McKinney, J., and F. Ali. (2014), "Artificial neural networks for the spalling classification & failure prediction times of high strength concrete colunms." *Journal of Structural Fire Engineering*, 5 (3): 203–214.

Mindeguia, J. C., H. Carré, P. Pimienta, and C. La Borderie. (2015), "Experimental discussion on the mechanisms behind the fire spalling of concrete." *Fire Mater*, 39 (7): 619–635.

Naser, M. (2023), Machine Learning for Civil and Environmental Engineers: A Practical Approach to Data-Driven Analysis, Explainability, and Causality. ISBN: 978-1119897606.

Naser, M. Z., and V. K. Kodur. (2022), "Explainable machine learning using real, synthetic and augmented fire tests to predict fire resistance and spalling of RC columns." *Eng Struct*, 253: 113824.

"Naser, M.Z. & albashiti, M. & Naser, A.. (2023), SPINEX: Similarity-based Predictions and Explainable Neighbors Exploration for Regression and Classification Tasks in Machine Learning."

Ozawa, M., and H. Morimoto. (2014), "Effects of various fibres on high-temperature spalling in high-performance concrete." *Constr Build Mater*, 71: 83–92.

Panev, Y., P. Kotsovinos, S. Deeny, and G. Flint. (2021), "The Use of Machine Learning for the Prediction of fire Resistance of Composite Shallow Floor Systems*." Fire Technol*, 57 (6): 3079–3100.

Prokhorenkova, L., G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. "CatBoost: unbiased boosting with categorical features." *proceedings.neurips.cc.*

Quinlan, J. R. (1986), "Induction of decision trees." Mach Learn, 1 (1): 81–106. *Springer Science*.

Saberian, M., L. Shi, A. Sidiq, J. Li, S. Setunge, and C. Q. Li. (2019), "Recycled concrete aggregate mixed with crumb rubber under elevated temperature." *Constr Build Mater*, 222: 119–129. Elsevier.

Seitllari, A., M. N.-Comput. Concr, and undefined (2019), "Leveraging artificial intelligence to assess explosive spalling in fire-exposed RC columns." *Comput. Concr, researchgate.net.*

"sklearn.model_selection.GridSearchCV — scikit-learn 1.3.0 documentation." Accessed July 7, (2023). https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

Teymori, A., G. Tapeh, · M Z Naser, and M. Z. Naser. (2022), "Artificial Intelligence, Machine Learning, and Deep Learning in Structural Engineering: A Scientometrics Review of Trends and Best Practices." *Archives of Computational Methods in Engineering* 30:1, 30 (1): 115–159.

Thai, H. T. (2022), "Machine learning for structural engineering: A state-of-the-art review." *Structures*, 38: 448–491.

"Welcome to LightGBM's documentation! — LightGBM 3.3.5 documentation." Accessed June 17, 2023. https://lightgbm.readthedocs.io/en/v3.3.5/index.html.

Zhang, H. L., and C. T. Davie. (2013), "A numerical investigation of the influence of pore pressures and thermally induced stresses for spalling of concrete exposed to elevated temperatures." *Fire Saf* J, 59: 102–110.

Zhao, J., J. J. Zheng, G. F. Peng, and K. Van Breugel. (2014), "A meso-level investigation into the explosive spalling mechanism of high-performance concrete under fire exposure." *Cem Concr Res*, 65: 64–75