Degtyarev, V.V., Tsavdaridis, K.D., 2022, Buckling and ultimate load prediction models for perforated steel beams using machine learning algorithms. Journal of Building Engineering, 51, 104316. https://doi.org/10.1016/j.jobe.2022.104316.

Buckling and ultimate load prediction models for perforated steel beams using machine learning algorithms

Vitaliy V. Degtyarev[a,*], Konstantinos Daniel Tsavdaridis[b]

[a] *New Millennium Building Systems, LLC, 3700 Forest Dr. Suite 501, Columbia, SC 29204, USA*
[b] *Department of Civil Engineering, School of Mathematics, Computer Science and Engineering, City, University of London, Northampton Square, London EC1V 0HB, UK*

Abstract

Large web openings introduce complex structural behaviors and additional failure modes of steel cellular beams, which must be considered in the design using laborious calculations (e.g., exercising SCI P355). This paper presents seven machine learning (ML) models, including decision tree (DT), random forest (RF), k-nearest neighbor (KNN), gradient boosting regressor (GBR), extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), and gradient boosting with categorical features support (CatBoost), for predicting the elastic buckling and ultimate loads of steel cellular beams. Large datasets of finite element (FE) simulation results, validated against experimental data, were used to develop the models. The ML models were fine-tuned via an extensive hyperparameter search to obtain their best performance. The elastic buckling and ultimate loads predicted by the optimized ML models demonstrated excellent agreement with the numerical data. The accuracy of the ultimate load predictions by the ML models exceeded the accuracy provided by the existing design provisions for steel cellular beams published in SCI P355 and AISC De-

*Corresponding author
Email addresses: vitaliy.degtyarev@newmill.com, vitdegtyarev@yahoo.com (Vitaliy V. Degtyarev), konstantinos.tsavdaridis@city.ac.uk (Konstantinos Daniel Tsavdaridis)

sign Guide 31. The relative feature importance and feature dependence of the models were evaluated and discussed in the paper. An interactive Python-based notebook and a user-friendly web application for predicting the elastic buckling and ultimate loads of steel cellular beams using the developed optimized ML models were created and made publicly available. The web application deployed to the cloud allows for making predictions in any web browser on any device, including mobile. The source code of the application available on GitHub allows running the application locally and independently from the cloud service.
Keywords: Cellular beams, Perforated web, Elastic buckling, Ultimate strength, Predictive models, Machine learning

1. Introduction

Perforated steel beams with repeating web openings have been used in construction for more than a century [1]. They offer several advantages over steel beams with solid webs, including weight reduction, higher strength-to-weight ratio, integration of utilities, and improved aesthetics. Castellated beams with hexagonal openings, which were the first type of beams with perforated web sections, have practically been replaced in modern construction by cellular beams with circular openings [2]. Multiple large openings cause a significant reduction in the beam shear strength and introduce additional possible failure modes of the beams, which makes the flexural behavior and design of cellular beams complicated. A cellular beam may exhibit one of the following failure modes: global bending, lateral-torsional buckling, vertical shear, local Vierendeel bending, web post horizontal shear, web post bending, web post buckling, and shear buckling. Many researchers have contributed to the body of knowledge about the strength and structural behavior of steel cellular beams. Several research publications describe numerical studies on the lateral-torsional buckling of cellular beams [3–8], which allowed for determining the effects of different design parameters on the beam strength governed by the elastic and inelastic lateral-torsional buckling. It was found in particular that the cellular beam geometry affected the moment-gradient coefficient, which is not the case for the solid-web beams [4]. T-shaped stiffeners were proposed to improve the flexural stiffness of cellular beams and reduce the lateral-torsional buckling occurrence [5]. Modified calculations of the cross-sectional properties and a modified buckling curve selection were developed based on the existing European guidelines [6].

Web post buckling of cellular beams and beams with web openings of different shapes was studied in [9–11]. Tsavdaridis and D'Mello [9] demonstrated that particular non-standard opening shapes improved the beam structural performance compared with the beams with standard circular, hexagonal, and elongated web openings. They also proposed an empirical formula for predicting the ultimate vertical shear strength of web posts formed by the different opening shapes. Panedpojaman et al. [10] proposed design equations for predicting the shear strength of local web post buckling in symmetric and asymmetric cellular beams, which demonstrated improved accuracy in predicting the shear strength compared with BS EN 1993-1-1 [12] and AISC 360 [13].

Chung et al. [14] investigated the Vierendeel mechanism in cellular steel beams and found that shear yielding is more critical in steel beams with circular openings than in beams with rectangular openings. They proposed an empirical shear moment interaction curve at the perforated sections. Kang et al. [15] studied the shear behavior and strength of cellular beams and proposed a rational design model for predicting the beam shear strength, which showed a good agreement with the numerical and experimental results. Ellobody [16] investigated combined buckling modes of steel cellular beams and found that the failure load could be significantly reduced when the beams failed in combined web distortional and web post-buckling.

Several research papers have been dedicated to the optimal design of cellular beams [9, 17–24]. The studies demonstrated that the strength and weight of the beams with web openings could be significantly improved by using non-standard opening shapes [9, 17, 19, 21], applying special optimization techniques [18, 20, 22], and selecting specific sizes and spacing of web openings [21, 23, 24].

Akrami and Erfani [25] assessed design methodologies for perforated steel beams presented in ASCE 23-97 [26], SCI P100 [27], SCI P355 [2], Chung et al. [28], and Tsavdaridis and D'Mello [17]. The two latter methods were found least restrictive and produced the lowest errors. The authors proposed ASCE 23-97 modifications, which showed a good agreement with experimental and numerical data.

The presented literature review shows that the published research concentrated mainly on studying specific failure modes of cellular beams. To fill the gap in the information about the global response of such members, Rajana et al. [29] performed an extensive numerical parametric study of the elastic and inelastic buckling of cellular beams subjected to strong axis bending. The effects of various parameters on the elastic buckling and ultimate loads of cellular beams were investigated, and an extensive database of the FE simulation results was generated. The study showed that the elastic buckling was affected mainly by the web thickness and the flange geometry. The diameter of web openings, their spacing, flange geometry, and web thickness were the most critical parameters affecting the beam strength. It was also determined that the initial geometric imperfections had an insignificant effect on the predicted beam strength.

Artificial intelligence (AI) and machine learning (ML) are emerging fields of computer science that allow for developing machines with simulated human intelligence and creating data-based descriptive models capable of handling very complex problems efficiently. A properly developed ML model for engineering applications reveals hidden relations between the predicted variable and input parameters based on the underlying physics. Many industries have successfully adopted AI and ML [30–33], whereas their deployment in structural engineering is still somewhat limited despite many research publications demonstrating the accuracy and effectiveness of the AI and ML methods.

The number of research publications on ML applications in civil and structural engineering had increased exponentially since the second half of the 1980s, when the first papers on this topic were published [34–40]. Many publications described ML models considered in this study for predicting properties of concrete and reinforced concrete structures [41–66]. Fewer papers have been published on ML applications to steel structures, including buckling analysis of beam-columns [67], cold-formed steel (CFS) space structure optimization [68], web crippling strength prediction [69], elastic distortional buckling stress determination [70, 71], rotation capacity prediction [72], strength prediction of concrete-filled steel tubular columns [73], failure mode identification of column base plate connection [74], capacity prediction of cold-formed stainless steel tubular columns [75], seismic drift demand estimation for steel moment frame buildings [76], and shear strength of CFS channels with staggered perforated webs [77–80]. ML techniques were previously applied to steel cellular beams. Sharifi et al. [81] developed an artificial neural network (ANN) to predict the flexural strength of steel cellular beams governed by lateral-torsional buckling using a relatively small dataset with 99 samples. The predicting abilities of the developed ANN were superior to those by the Australian Standard [82]. Abambres et al. [83] developed an ANN model and an ANN-based formula for predicting the elastic buckling load of cellular beams using a large dataset of numerical results described in [29]. The ANN and the proposed formula showed an excellent agreement with the FE simulation results. An ANN and ANN-based formula for predicting the lateral-torsional buckling resistance of slender steel cellular beams were presented by Ferreira et al. [84]. Limbachiya and Shamass [85] presented an ANN and ANN-based formula for predicting web-post buckling shear strength of cellular beams, which demonstrated a higher level accuracy compared with the existing design provisions.

The presented review indicates that ML has excellent potential for developing structural engineering expert tools. ML models cannot currently be solely used for final designs because building codes do not permit them. However, they can be employed in the preliminary design stages to quickly evaluate and select options that may work and consider them in the detailed analysis and design per building codes. Due to the superior performance of ML models compared with conventional models demonstrated on many engineering problems, the question of what should be done to adopt them in building codes will need to be eventually answered. ML models are based on solid mathematical algorithms, well-described in the literature. The novelty of the algorithms, which structural engineers do not fully understand, is one of the significant barriers today to their adoption, which will eventually change with more education, research, and experience.

ML models built on top of the available test or numerical data are computationally efficient and often more accurate than the existing design methods based on the traditional approaches, which often rely on fewer data points and engineering intuition. They can replace computationally intensive finite element simulations when the design parameters are within the ranges of the data used for the ML model training. It should also be noted that accurate finite element simulations require advanced software resources, which are not always available to designers, and advanced techniques, thus skills and knowledge. Even when the appropriate software is available, and the designers possess the required expertise, it is impractical to perform advanced finite element simulations during the project design phase due to time constraints. Because of that, engineers are often using simplified FEA to perform stress analyses. The downside is that these are not very accurate due to the number of assumptions, leading to the very same initial problem: the need of higher safety factors and the excessive use of material where it is not needed.

This study aims to explore various ML algorithms for predicting the elastic buckling and ultimate loads of steel cellular beams. Considering the complexity of the structural behavior and design of such members, ML models are deemed to be a promising alternative to the existing design guidelines and computationally expensive FE modeling. The objectives of the study were as follows:

1. Develop and optimize ML models for predicting the elastic buckling and ultimate loads of steel cellular beams based on seven popular ML regressors, including decision tree (DT), random forest (RF), k-nearest neighbors (KNN), gradient boosting regressor (GBR), extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), and gradient boosting with categorical features support (CatBoost).
2. Interpret and explain the developed models using the permutations and SHapley Additive exPlanations (SHAP) [86] methods.
3. Compare predictions by the developed ML models with those per SCI P355 [2] and AISC Design Guide 31 [87].

ML models were trained using FE simulation results of steel cellular beams published in [29]. The elastic buckling load dataset included 3645 samples. The ultimate load (inelastic buckling load) dataset consisted of 78390 samples. All models were implemented in open-source Python-based frameworks, and their hyperparameters were optimized via an extensive tuning process. The ten-fold cross-validation method was employed for the model training and performance evaluation. The final evaluation of the models was performed on the data unseen by the models during training. The ML model predictions showed an excellent agreement with the FE simulation results. The ultimate loads of the cellular beams predicted by the models compared with the FE analysis data considerably better than the ultimate loads predicted by SCI P355 [2] and AISC Design Guide 31 [87]. The developed ML models allow for computing the elastic buckling and ultimate loads of cellular beams with a wide range of variables, including intermediate values of variables not considered in numerical studies used for the model training.

A web application for predicting the elastic buckling and ultimate loads of steel cellular beams was created in Streamlit. A light version of the application was deployed to the cloud at `https://scba-cb.herokuapp.com/`. It allows for making predictions in any web browser on any device, including mobile. The source codes of the full and lite application versions are available on GitHub at `https://github.com/vitdegtyarev/SCBA-Streamlit` and `https://github.com/vitdegtyarev/SCBA-Streamlit-CB`, respectively. They allow for running the application on a local machine. The scientific research part of this study consists of creating and optimizing ML models for predicting the behavior of cellular beams, while the web application is a convenient tool for obtaining predictions by the developed models.

The novelty of the presented work consists of the development of new optimized ML models for accurate and computationally efficient predictions of the elastic buckling and ultimate loads of steel cellular beams, interpretation and explanation of the developed models using the permutations and SHAP methods, comparison of the performance of seven different ML models, and development of a web application based on the optimized ML models for the ease of use by engineers in practice.

2. Datasets

The elastic buckling load, w_{cr}, and ultimate load (inelastic buckling load), w_{max}, datasets of FE simulation results from [29] were used for training and performance evaluation of the ML models. The FE models were validated against the experimental data, as described in [29]. The w_{cr} and w_{max} datasets consist of 3645 and 78390 samples, respectively. Fig. 1 shows dimensional parameters of the cellular beams considered in the numerical parametric study, including beam span length, L; beam height, H; flange width, b_f; flange thickness, t_f; web thickness, t_w; opening diameter, D_o; web post width, WP; and opening end distance, L_{ed}.

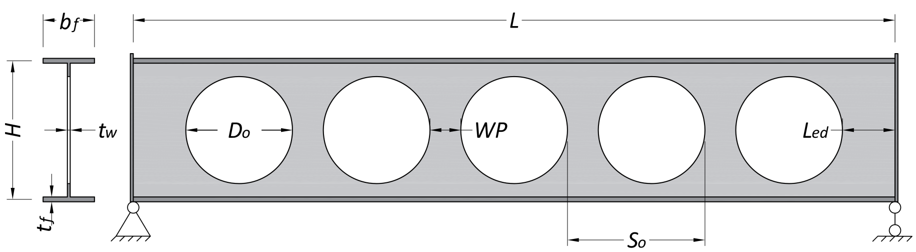

Figure 1: Dimensional parameters of steel cellular beams

In addition to the dimensional parameters of the beams, the ultimate load dataset included steel yield stress, F_y; steel ultimate stress, F_u; steel yield strain, ϵ_y; steel ultimate strain, ϵ_u; and initial geometric imperfections considered in the FE models. The dimensional beam characteristics shown in Fig. 1 and F_y (in the w_{max} models only) were considered the ML models' input parameters. The initial geometric imperfections were excluded from the input parameters because they have an insignificant effect on w_{max} [29], and their exact shape and magnitude are not usually known to the designer.

Distributions of the parameters in the elastic buckling and ultimate load datasets presented in Figs. 2 and 3 demonstrate that the datasets cover a wide range of the beams and represent the steel cellular beams used in construction.

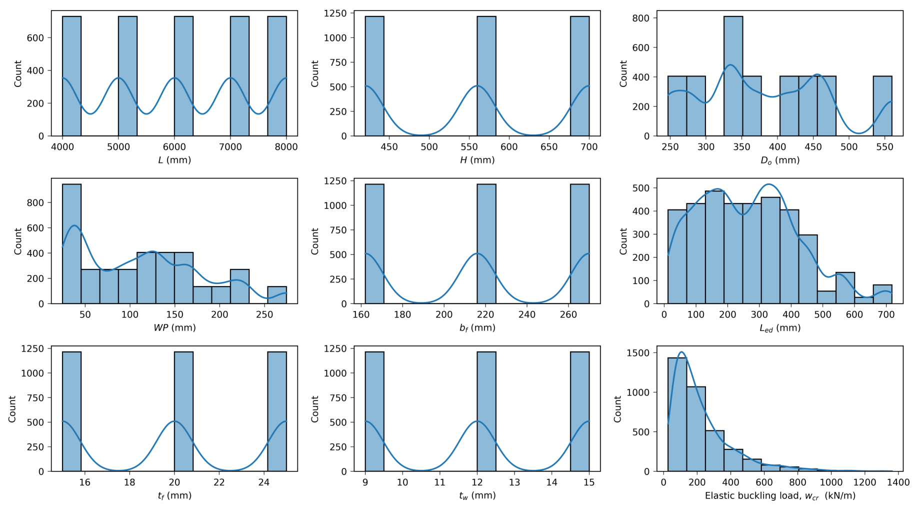

Figure 2: Distributions of variables of the elastic buckling load dataset

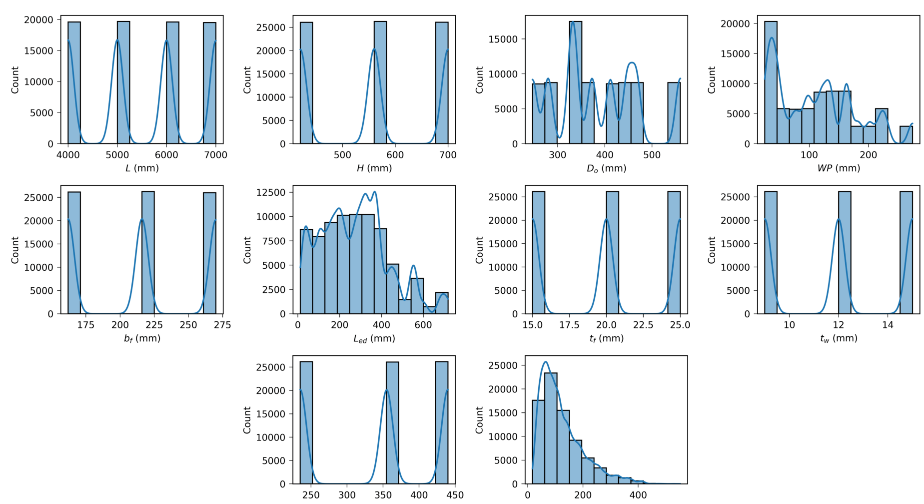

Figure 3: Distributions of variables of the ultimate load dataset

Fig. 4 and 5 show correlation matrices for the dataset variables. The beam span length, L, has the highest negative correlation with w_{cr} and w_{max}, characterized by moderate coefficients of correlation of -0.67 and -0.60, respectively.

All other variables have weak correlations with w_{cr} and w_{max}, with coefficients of correlation not exceeding 0.37. It is interesting to note that w_{max} has a considerably stronger correlation with WP than w_{cr}, which indirectly highlights the positive contribution of the web post plastic behavior to the ultimate load of the cellular beams. D_o has a strong positive correlation with H because the D_o values were set as fractions of the H values in the numerical parametric study. All other dataset variables have weak correlations between themselves.

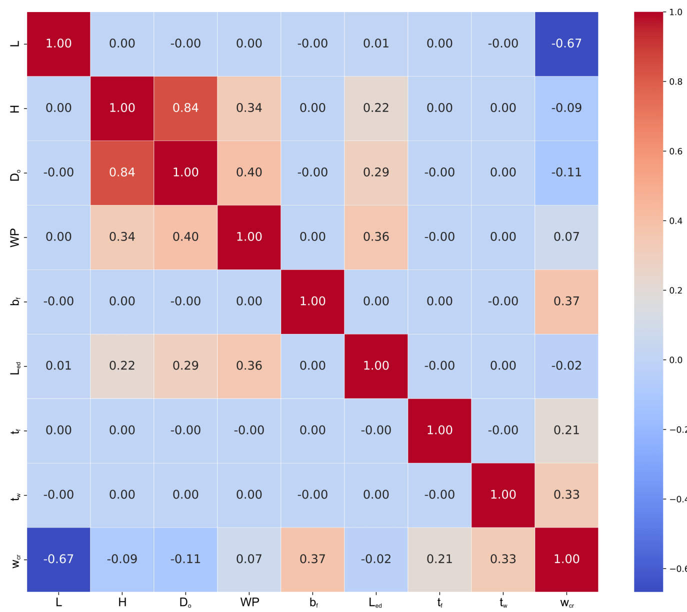

Figure 4: Correlation matrix for the elastic buckling load dataset

The datasets used in this study can be found at the following link: `https://www.kaggle.com/vitdegtyarev/buckling-and-ultimate-loads-of-cellular-beams`.

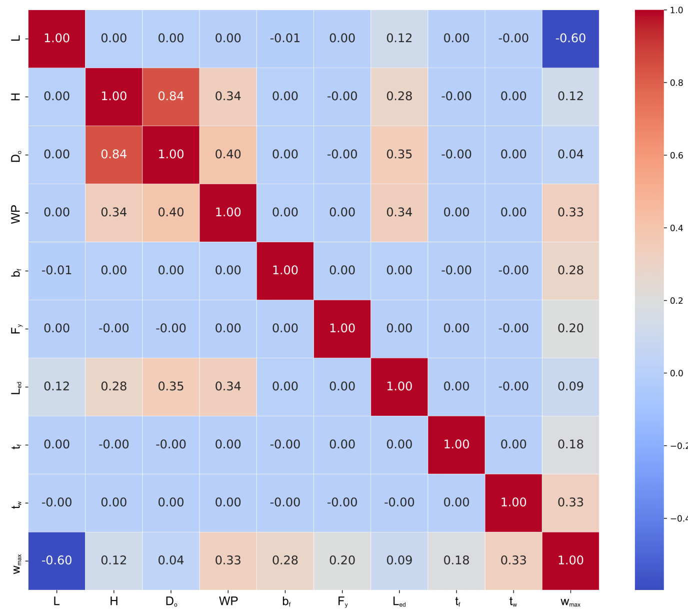

Figure 5: Correlation matrix for the ultimate load dataset

3. Review of machine learning algorithms

The abilities of seven popular supervised ML algorithms to predict the elastic buckling load, w_{cr}, and ultimate load, w_{max}, of steel cellular beams were evaluated. Supervised ML algorithms learn by example using labeled training data, which consist of input parameters (also known as features) and one or more output values (also known as targets). The evaluated ML algorithms included decision tree (DT), random forest (RF), k-nearest neighbors (KNN), gradient boosting regressor (GBR), extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), and gradient boosting with categorical features support (CatBoost). These algorithms are commonly employed to develop predictive ML models in civil/structural engineering (as was discussed in the Introduction section) and other domains. They are based on different principles and may result in different performances when used for different problems. One algorithm may demonstrate a better predictive accuracy than others on one problem and inferior performance on a different problem. Therefore, it is important to find an algorithm and its optimal hyperparameters that works the best for a given problem.

Fig. 6 demonstrates the schematic architecture of the considered ML models for predicting w_{cr} and w_{max}. The models consisted of features, ML algorithms, and targets. The features of the models for predicting w_{cr} were L, H, b_f, t_f, t_w, D_o, WP, and L_{ed}. The w_{max} models also included F_y as a feature.

All ML algorithms have hyperparameters, or the parameters specified before the model training to control the learning process and avoid overfitting or underfitting. Overfitting is characterized by the ability of a model to make good predictions for the samples used in training while making poor predictions on the new samples of data unseen by the model before. An underfitted model produces poor predictions on the seen and unseen data. The ability of an ML model to make good predictions for previously unseen data is referred to as generalization. Finding optimal hyperparameters is essential for obtaining a model with the best performance and generalization [54, 77]. It is equivalent to finding the form and coefficients of a regression equation that gives the best prediction accuracy for a given problem.

The following sections present a brief overview of each ML algorithm considered in the study. Detailed information about the ML algorithms and their practical implementation can be found in published literature, including [88] and [89].

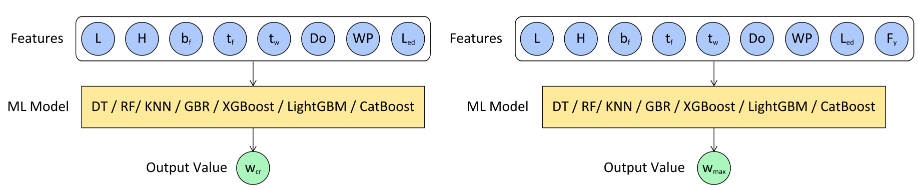

Figure 6: Architecture of ML models

3.1. Decision tree

The DT algorithm bears its name from its tree structure incrementally developed by splitting the dataset into smaller subsets. DT models have three types of nodes: root node, decision nodes, and terminal nodes (also known as leaves). The learning starts at the root node, which includes all training data. The root node splits into two or more decision nodes, which include subsets of the original training data. The splitting occurs based on a series of questions determined by the algorithm. It continues for the subsequent levels until a predefined maximum depth of the tree is reached or when the nodes have only one sample of the training data. The algorithm stops at the terminal nodes, which do not split.

Various algorithms for growing a DT exist. They differ by the possible tree structure, the split finding criteria, the splitting stoppage criteria, and the model estimation within the terminal nodes. The classification and regression trees (CART) algorithm [88] was used in this study. In this algorithm, a dataset (x_i, y_i) for $i = 1, 2, ..., N$, with $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$ is considered, where x_i and y_i are features and targets, N is the number of samples, and p is the number of features. The original dataset is split into M regions R_1, R_2,...,R_M. The model prediction in each region is a constant c_m described by Eq. 1.

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{1}$$

where $I(x \in R_m)$ is the identity function that returns 1 if x is in the subset R_m and 0 otherwise.

The best $\hat{c}_m$ is the average of y_i in region R_m when the sum of squared errors $\sum(y_i - f(x_i))^2$ is used as the criterion of minimization:

$$\hat{c}_m = \text{ave}(y_i \mid x_i \in R_m) \tag{2}$$

The following greedy algorithm is employed to find the best binary partition of each node in terms of the minimum sum of squared errors. The pair of half-planes partitioned by a splitting variable j and a split point s is defined as follows.

$$R_1(j,s) = \{X \mid X_j \leq s\} \text{ and } R_2(j,s) = \{X \mid X_j > s\} \tag{3}$$

The splitting variable j and the split point s that solve Eq. (4) are sought.

$$\min_{j,s}\left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2\right] \tag{4}$$

The inner minimization is solved by

$$\hat{c}_1 = \text{ave}(y_i \mid x_i \in R_1(j,s)) \text{ and } \hat{c}_2 = \text{ave}(y_i \mid x_i \in R_2(j,s)) \tag{5}$$

Once the best split is found, the dataset is partitioned into two resulting subsets, after which the splitting process is repeated for each subset and each node in the subsequent levels.

The DT advantages consist of the relative ease of data preparation, the ease of understanding and interpretation, and robustness against missing values. One of the main disadvantages of DT is their proneness to overfitting when the tree is very large [90, 91]. To avoid overfitting, the DT model should not be very large. At the same time, the model should be large enough to capture the important relationships between the features and targets to avoid underfitting. DT hyperparameters include the maximum depth of the tree, the minimum number of samples required to split an internal node, the minimum number of samples required to be at a leaf node, and others.

3.2. Random forest

RF is an ensemble of DTs generally trained via *bagging*, which stands for *bootstrap aggregating* [88]. In this method, the same algorithm (DT) is trained many times on different random subsets of the entire training set. The sampling is performed with replacement, meaning that the same sample may appear in different subsets. Predictions from multiple randomly generated DTs are averaged to obtain the final output value of the RF algorithm.

The RF regression algorithm consists of the following steps [88].

1. For b=1 to B, where b is an individual DT and B is the total number of DTs (estimators):
   (a) A bootstrap sample of size N is drawn from the training data.
   (b) A tree T_b is grown to the bootstrapped data by repeating the following substeps for each node until the maximum tree depth or the minimum node size is reached:
      i. m variables are randomly selected from p variables.
      ii. The best variable among m and the best split point is found.
      iii. The node is split into two nodes.
2. The ensemble of trees $\{T_b\}_1^B$ is output.
3. The final prediction is made as $\hat{f}_{RF}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

The RF advantages include those listed in Subsection 3.1 for DT and its robustness against overfitting due to the presence of multiple independent DTs making predictions. On the negative side, RF requires more computational power and resources to build numerous trees and combine their outputs compared with DT. The RF hyperparameters include those for DTs plus the number of trees in the forest.

3.3. K-nearest neighbors

The KNN regression algorithm predicts the output value by interpolating the output values of k nearest neighbors in the training set. The number of neighbors k is a hyperparameter set before training. The distance between the neighbors is defined by the distance function in the form of the Minkowski metric described by Eq. (6). The Euclidean and Manhattan distances, which are other typical distance metrics, can be obtained from the Minkowski metric by setting the power parameter, p, equal to 1 and 2, respectively.

$$D(X,Y) = \left(\sum_{i=1}^{k} |x_i - y_i|^p \right)^{\frac{1}{p}} \tag{6}$$

The output values are obtained by taking either an average (Eq.(7)) or an inverse distance weighted average (Eq.(8)) of the k nearest neighbors with similar features. In the latter approach, closer neighbors have a more significant influence on the target than the more distant neighbors.

$$\hat{f}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \tag{7}$$

$$\hat{f}(x) = \frac{\sum_{x_i \in N_k(x)} \frac{1}{d_i} y_i}{\sum_{x_i \in N_k(x)} \frac{1}{d_i}} \tag{8}$$

where $N_k(x)$ is the neighborhood of x defined by the k closest points x_i in the training data, d_i is the distance from the i^{th} point to the estimated point.

The KNN advantages include the ease of implementation, the ability to add new data without the effect on the algorithm's accuracy, and the training period absence, which makes the KNN algorithm significantly faster than other ML algorithms when the dataset size and the number of input variables are relatively small. The KNN disadvantages consist of sensitivity to noisy data, missing values and outliers, and slow predictions for large datasets and datasets with a large number of features. The KNN hyperparameters are the number of neighbors, the weight function (uniform or inverse distance weighted), and the distance metric.

3.4. Gradient boosting

Boosting algorithms, or boosting machines, are ensemble methods that combine several weak learners (usually DTs) to produce a strong learner. Boosting machine predictors are trained sequentially, with each subsequent learner improving the predecessor's predictions. The algorithm stops when a predefined number of predictors is reached or when the perfect fit is achieved. The two common boosting algorithms are gradient boosting and adaptive boosting.

In gradient boosting, the boosting algorithm is combined with gradient descent, which is an iterative optimization algorithm for finding a local minimum of a function. New predictors are fitted to the residual errors from the previous predictors. The gradient boosting algorithm includes the following steps [92]:

1. For a training set $(x_i, y_i)_{i=1}^{n}$, the model is initialized with a constant value of

$$F_0(x) = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, \gamma) \tag{9}$$

   where i and n denote the i^{th} sample and the total number of samples

2. For m=1 to M, where m and M are the m^{th} iteration and the total number of iterations:

   (a) Pseudo-residuals are computed as follows:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \tag{10}$$

   (b) The training set $(x_i, r_{im})_{i=1}^{n}$ is used to fit a predictor $h_m(x)$ to pseudo-residuals.

(c) Multiplier γ_m is computed by solving the following optimization problem:

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L[y_i, F_{m-1}(x_i) + \gamma h_m(x_i)] \tag{11}$$

(d) The model is updated using the following equation:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x) \tag{12}$$

3. $F_M(x)$ is obtained.

The gradient boosting algorithm's advantages include high accuracy, flexibility, and the ability to handle missing data. It is generally considered resistant to overfitting due to many weak learners involved in the prediction. However, the algorithm may overfit when its hyperparameters are poorly selected. The disadvantages of gradient boosting are computation cost, the number of hyperparameters that require proper tuning, and limited interpretability.

The gradient boosting hyperparameters include learning rate, the number of boosting iterations, maximum depth of the individual regression estimators, the minimum number of samples required to split an internal node, the minimum number of samples required to be at a leaf node, and others.

The gradient boosting algorithms have been implemented in several frameworks: GBR [93], XGBoost [94], LightGBM [95], and CatBoost [96]. XGBoost, LightGBM, and CatBoost are improved implementations of GBR. XGBoost was optimized for more accurate and faster predictions via regularization, custom loss functions, parallel processing, and other algorithm improvements. LightGBM offers improved training speed, higher efficiency, better accuracy, lower memory use, and the ability to process large datasets by applying the Gradient-based One-Side Sampling (GOSS) method and parallel learning. CatBoost can process categorical features to improve accuracy for datasets with categorical features, ordered boosting to fight overfitting, missing value support, and others.

4. Implementation and results

The ML algorithms were implemented in the following Python-based open-source libraries: *scikit-learn* (DT, RF, KNN, and GBR) [93], *XGBoost* [94], *LightGBM* [95], and *CatBoost* [96]. The models were optimized, validated, and tested using the ten-fold cross-validation method. The w_{cr} and w_{max} datasets were randomly divided into training and test sets in the 80/20 proportion. The training set of each dataset was partitioned into ten groups. The models were trained on nine groups of the training set and validated on the remaining group. The process was repeated for the remaining groups of the training set until each group had served as the validation set. The final performance of the models was evaluated on the test data unseen by the models in training. Compared with the hold-out method, where the dataset is divided into training, validation, and test sets, with each set used for its purpose only, the ten-fold cross-validation method makes more samples available for model training and excludes model dependence on a particular random choice of the samples selected for the training, validation, and test sets. As a result, the ten-fold cross-validation method usually produces more accurate models with better generalization performance.

Figs. 2 and 3 demonstrate that the numerical ranges of the features ranged widely in the datasets, which is not ideal for ML models, as it might cause difficulties for the algorithms in finding optimal model parameters. Each feature value in the training set was standardized using Eq. (13) to make the features' scales uniform. Each feature in the test set was also standardized using the mean and standard deviation values of the feature obtained for the training set.

$$x' = \frac{x - \mu}{\sigma} \tag{13}$$

where x' is the standardized value of the input parameter, x is the original (non-standardized) value of the input parameter, μ is the mean of the original values of the input parameter, and σ is the standard deviation of the original values of the input parameter.

Performance of the ML learning models was evaluated based on the mean squared error (MSE) values obtained for the test set calculated as follows.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2 \tag{14}$$

where n is the number of samples, y is the output value, and $\hat{y}$ is the predicted output value.

Mean absolute error (MAE), mean absolute percentage error (MAPE), the coefficient of determination (R^2), the minimum, maximum, mean, and coefficient of variation values of the prediction-to-FEA ratios, which are metrics commonly used for performance evaluation of ML models [97], calculated using the following equations were also determined for the training and test sets.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y-\hat{y}| \tag{15}$$

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y-\hat{y}}{y}\right| \tag{16}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y-\hat{y})^2}{\sum_{i=1}^{n}(y-\bar{y})^2} \tag{17}$$

where $\bar{y}$ is the mean of the y values.

Extensive hyperparameter tuning was carried out for each ML model using the grid and random searches to find optimal hyperparameter values that give the best model performance. The obtained optimal hyperparameters for each ML model are listed below. The hyperparameter designations used in the Python libraries [93–96] are shown in parentheses. The hyperparameters not presented below had default values.

- DT:
  - the maximum depth of the tree (`max_depth`): None for w_{cr} and w_{max},
  - the minimum number of samples required to split an internal node (`min_samples_split`): 4 for w_{cr} and 2 for w_{max},
  - the minimum number of samples at a leaf node (`min_samples_leaf`): 2 for w_{cr} and w_{max}.
- RF:
  - the number of trees in the forest (`n_estimators`): 80 for w_{cr} and 200 for w_{max},
  - the maximum depth of the tree (`max_depth`): None for w_{cr} and w_{max},
  - the minimum number of samples required to split an internal node (`min_samples_split`): 2 for w_{cr} and w_{max},
  - the minimum number of samples at a leaf node (`min_samples_leaf`): 1 for w_{cr} and w_{max}.
- KNN
  - the number of neighbors (`n_neighbors`): 5 for w_{cr} and 4 for w_{max},
  - weight function (`weights`): uniform for w_{cr} and w_{max},
  - the power parameter for the Minkowski metric (`p`): 1 for w_{cr} and w_{max},
  - leaf size (`leaf_size`): 20 for w_{cr} and 30 for w_{max}.
- GBR:

- learning rate (`learning_rate`): 0.1 for w_{cr} and w_{max},
- the number of boosting stages (`n_estimators`): 200 for w_{cr} and 1300 for w_{max},
- maximum depth of individual regression estimators (`max_depth`): 5 for w_{cr} and 9 for w_{max},
- the minimum number of samples required to split an internal node (`min_samples_split`): 2 for w_{cr} and w_{max},
- the minimum number of samples at a leaf node (`min_samples_leaf`): 3 for w_{cr} and 4 for w_{max}.

• XGBoost:

- learning rate (`eta`): 0.2 for w_{cr} and w_{max},
- minimum loss reduction required to make a further partition on a leaf node of the tree (`gamma`): 1 for w_{cr} and w_{max},
- the maximum tree depth of base learners (`max_depth`): 5 for w_{cr} and 12 for w_{max},
- the minimum sum of instance weight (hessian) needed in a child (`min_child_weight`): 3 for w_{cr} and 6 for w_{max}.

• LightGBM:

- learning rate (`learning_rate`): 0.1 for w_{cr} and w_{max},
- the number of boosting iterations (`num_iterations`): 100 for w_{cr} and 3800 for w_{max},
- maximum tree leaves for base learners (`num_leaves`): 50 for w_{cr} and w_{max},
- the minimum number of observations that must fall into a tree node for it to be added (`min_data_in_leaf`): 10 for w_{cr} and w_{max},

  - maximum tree depth for base learners (`max_depth`): -1 (None) for w_{cr} and w_{max},
  - the maximum number of bins (`max_bin`): 100 for w_{cr} and w_{max}.

- CatBoost
  - learning rate (`learning_rate`): 0.03 for w_{cr} and w_{max},
  - the number of iterations (`iterations`): 850 for w_{cr} and 4000 for w_{max},
  - tree depth (`depth`): 6 for w_{cr} and 11 for w_{max},
  - L2 regularization term coefficient of the cost function (`l2_leaf_reg`): 3 for w_{cr} and 1 for w_{max},
  - the amount of randomness to use for scoring splits when the tree structure is selected (`random_strength`): 1 for w_{cr} and 2 for w_{max}.

Figs. 7 and 8 show the performance of the developed ML models with the optimal hyperparameters for predicting w_{cr} and w_{max}, respectively. The comparisons of the ML model predictions with FE simulation results are demonstrated for the training and test datasets in each figure. The values of R^2, minimum, maximum, mean, and coefficient of variation of the prediction-to-FEA ratios are presented in Figs. 7 and 8.The MSE, MAE, and MAPE values are given in Tables 1 and 2.

Table 1: Performance metrics of ML models for predicting elastic buckling loads of steel cellular beams, w_{cr} (Train/Test)

Model	**MSE ((kN/m)^2)**	**MAE (kN/m)**	**MAPE (%)**
DT	155.94/1538.26	6.04/19.07	2.35/7.05
RF	83.79/682.24	4.44/13.00	1.83/5.27
KNN	1320.87/2617.14	21.09/28.99	10.22/12.66
GBR	39.68/319.27	3.85/7.66	1.99/3.24
XGBoost	48.44/294.91	4.51/8.59	2.51/3.95
LightGBM	41.88/366.81	3.57/7.56	1.82/3.06
CatBoost	33.90/295.43	3.16/6.15	1.52/2.48

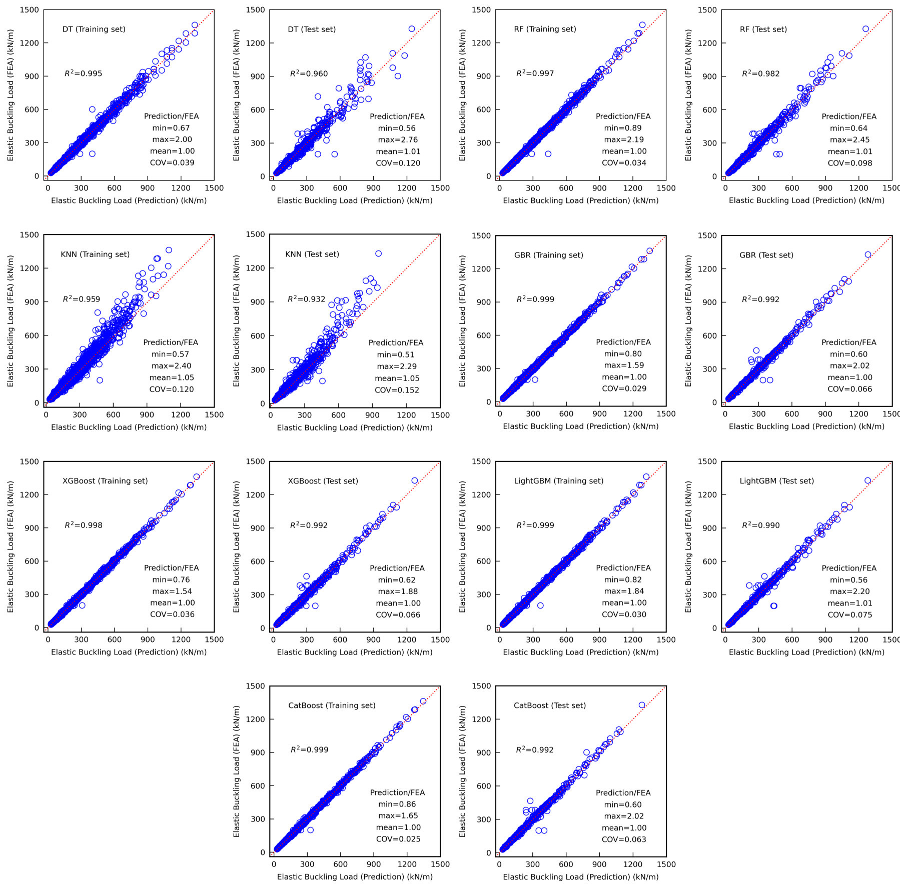

Figure 7: Performance of ML models for predicting elastic buckling load of steel cellular beams

Table 2: Performance metrics of ML models for predicting ultimate loads of steel cellular beams, w_{max} (Train/Test)

Model	MSE ((kN/m)^2)	MAE (kN/m)	MAPE (%)
DT	15.99/20.25	2.08/2.37	1.90/2.18
RF	16.01/20.17	2.08/2.38	1.90/2.18
KNN	18.43/22.61	2.18/2.49	1.99/2.27
GBR	16.01/20.24	2.08/2.38	1.90/2.18
XGBoost	16.03/20.13	2.09/2.39	1.92/2.19
LightGBM	16.10/20.10	2.11/2.40	1.94/2.20
CatBoost	16.05/20.17	2.10/2.39	1.92/2.19

As can be seen from Fig. 7 and Table 1, CatBoost, XGBoost, and GBR demonstrated comparable performances in predicting w_{cr} for the test set. The

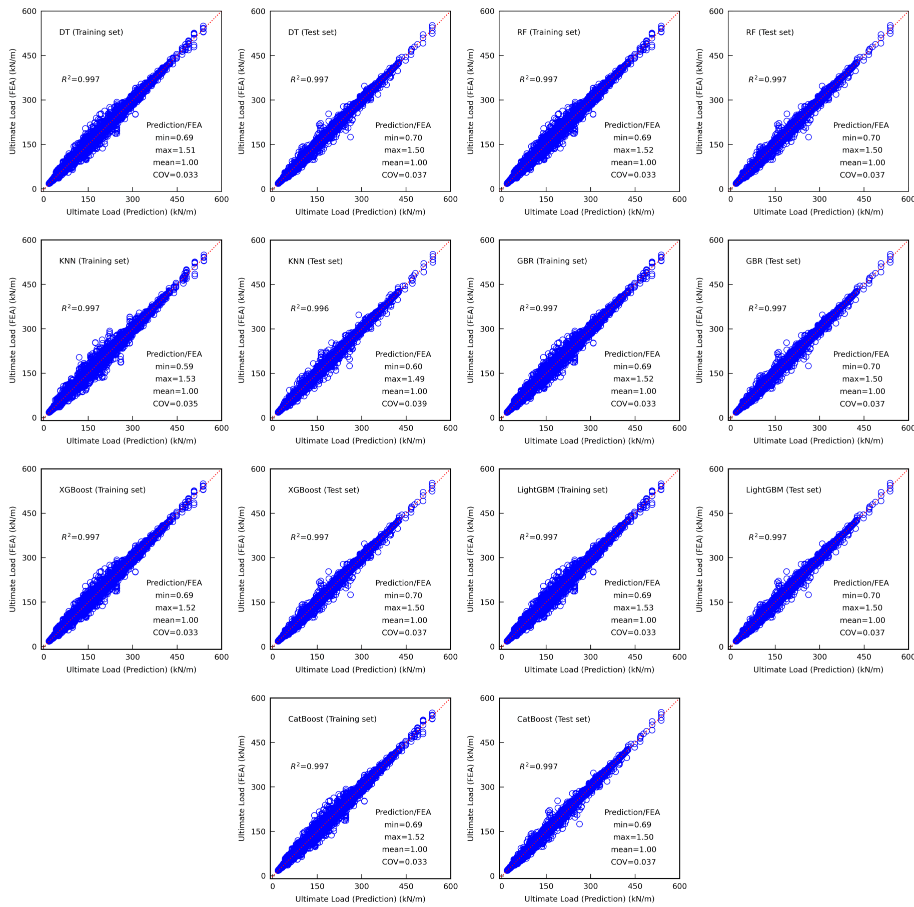

Figure 8: Performance of ML models for predicting ultimate load of steel cellular beams

performance metrics for LightGBM were slightly worse than those for CatBoost, XGBoost, and GBR. KNN provided inferior performance compared with other considered ML models. Fig. 8 and Table 2 show that all models performed well in predicting w_{max}, with KNN providing slightly worse metrics than other algorithms. It is worth reminding that the ultimate load dataset included 78390 samples and was significantly larger than the elastic buckling load dataset with 3645 samples. The good predictions of the ultimate load by all considered ML models, which was not the case for the elastic buckling dataset, highlight that good data with a very large number of samples contributes more to the accuracy of ML models than the ML algorithm differences. It can also be seen from comparisons of the performance metrics for the training and test sets that the created ML models with the optimal hyperparameters have a reasonably good generalization performance.

Developed ML models can be accessed at the following link: `https://www.kaggle.com/vitdegtyarev/cellular-beams-ml-models`. An interactive notebook for predicting the elastic buckling and ultimate loads of steel cellular beams with the developed ML models can be found at the following link: `https://www.kaggle.com/vitdegtyarev/ml-models-for-cellular-beams?scriptVersionId=63075739`.

5. Relative feature importance and feature dependence

Structural engineers often perceive ML methods as black boxes because humans cannot easily explain and interpret ML predictions. To remove this barrier to adopting ML methods, several ML explainability and interpretability techniques are available, including relative feature importance, partial dependence, feature interactions, and SHAP [98]. These techniques shed light on why and how an ML model made its predictions and expose how ML model predictions compare with mechanics-based knowledge. The application of the explainability and interpretability methods to the developed ML models is described in this section.

Relative effects of the features on the w_{cr} and w_{max} predictions by each considered ML model were analyzed using the permutation and SHAP methods. The permutation feature importance is a decrease in a model score when the feature values are randomly shuffled (permuted). A feature with a more significant score decrease is more important than others. The model score in the form of coefficient of determination, R^2, was used in this study. The random shuffling of values is repeated several times for each feature to obtain the mean and the standard deviation of the permutation importance score.

The SHAP method [86] aims to explain a prediction for a sample by determining the contribution of each feature to the prediction by computing Shapley values from coalitional game theory [99]. The Shapley value represents the average contribution of one player, which is a model feature in our case, to the model predictions taken for all possible combinations, which may consist of all dataset samples or a predefined portion of them. SHAP uses an additive feature attribution method – a linear explanation model of the summation of present features. The feature importance is determined based on the absolute average Shapley values. Features with larger Shapley values are more important than others. SHAP feature importance is based on the magnitude of feature attributions, while permutation feature importance is based on the decrease in the model performance. Thus, the relative feature importance predicted by these two methods might be different.

The relative feature importance was determined for all considered ML models using both methods. The relative feature importance plots were similar for all models. Therefore, the relative feature importance for the CatBoost models, which are ones of the most accurate models for predicting w_{cr} and w_{max}, is presented and discussed hereafter. Fig. 9 shows permutation and SHAP feature importance plots for the optimized CatBoost models for predicting w_{cr} and w_{max}.

The relative feature importance in predicting the elastic buckling load, w_{cr}, of steel cellular beams is discussed first. The span length, L, has the most significant importance according to both methods, which was expected. The next important beam parameters are the flange width, b_f, the web thickness,

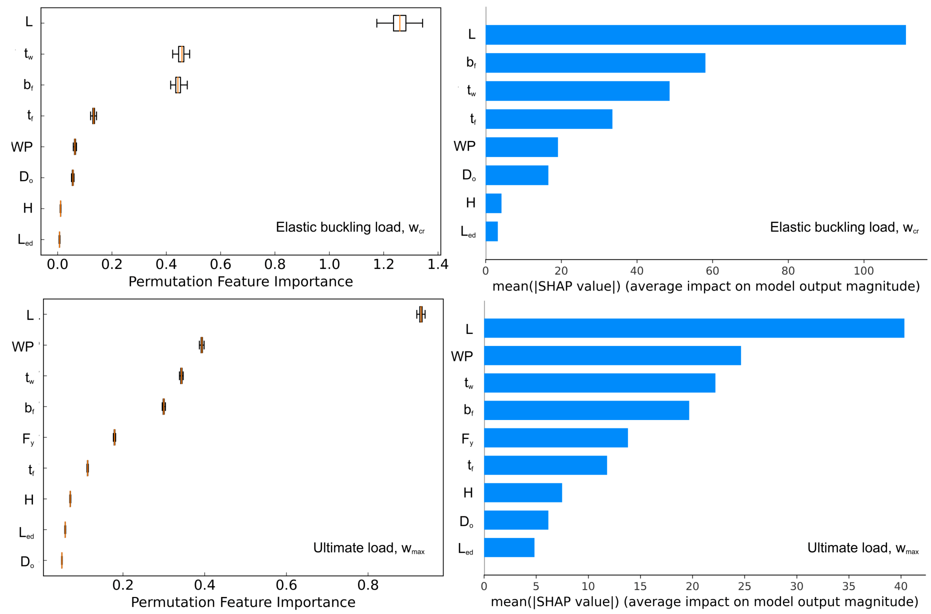

Figure 9: Permutation and SHAP feature importance for CatBoost models

t_w, and the flange thickness, t_f. The permutation method indicated that t_w is more important than b_f in predicting w_{cr}, while the SHAP method showed b_f above t_w. However, the difference in the importance scores for b_f and t_w is small, especially per the permutation method. These results compare well with the conclusions made in [29], indicating that the CatBoost model can capture the mechanics of the cellular beam behavior. The relative importance of the web post width, WP, the opening diameter, D_o, the beam height, H, and the opening end distance, L_{ed}, have relatively small importance in predicting w_{cr} according to both methods.

The permutation and SHAP relative feature importance plots for w_{max} demonstrate that the span length, L, is the most important feature, followed by WP, t_w, b_f, F_y, and t_f. It should be noted that WP has a more significant impact on w_{max} than on w_{cr}. These results align with the conclusion made in [29] and confirm the positive contribution of the web post plastic behavior to the strength of steel cellular beams mentioned in Section 2. The relative importance of H, D_o, and L_{ed} for predicting w_{max} is minor.

SHAP feature importance plots provide useful information, which is, however, somewhat limited. SHAP summary plots shown in Fig. 10 are more informative as they combine feature importance and feature effects. Each point on the summary plots represents a Shapley value for a dataset sample. The color shows the feature value from low (blue) to high (red). Points with the same Shapley values are scattered vertically to demonstrate their distribution for each feature. The order of the features follows their importance; so, it is the same as shown in Fig. 9. The SHAP summary plots presented in Fig. 10 indicate that w_{cr} and w_{max} increase when the beam span reduces and vice versa. Wide web posts have higher w_{cr} and w_{max}, which decrease when the web post width reduces. Greater values of t_w, b_f, t_f, and L_{ed} produce higher w_{cr} and w_{max}, whereas an increase in the opening diameter D_o results in w_{cr} and w_{max} reductions. The beam height H affects w_{cr} and w_{max} differently: w_{cr} goes down when H increases, while w_{max} increases when H goes up. The reduction of w_{cr} with an increase in H can be explained by an increase in the web post slenderness, which results in the elastic buckling load reduction.

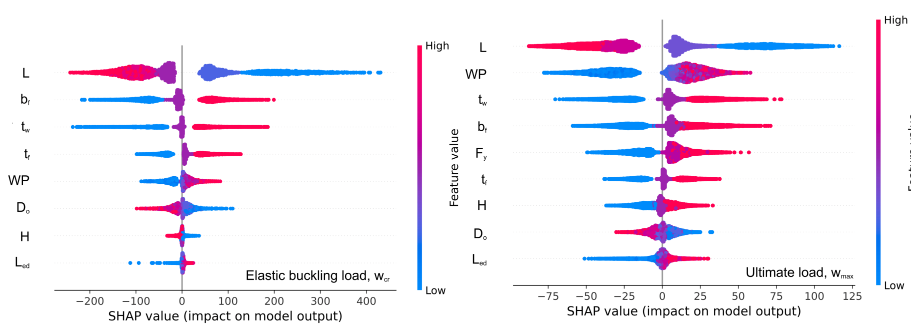

Figure 10: SHAP summary plots for CatBoost models

SHAP dependence plots given in Figs. 11 and 12 for w_{cr} and w_{max} illustrate exact relationships between feature values and predictions. Each point represents a prediction for a dataset sample. Feature values are shown on the horizontal axes, while SHAP values are given on the vertical axes. The SHAP values demonstrate the magnitude of change in w_{cr} and w_{max} when the feature's value is known. The color of each point corresponds to the second feature, which was determined by the algorithm to have the highest interaction with the considered feature shown on the horizontal axis.

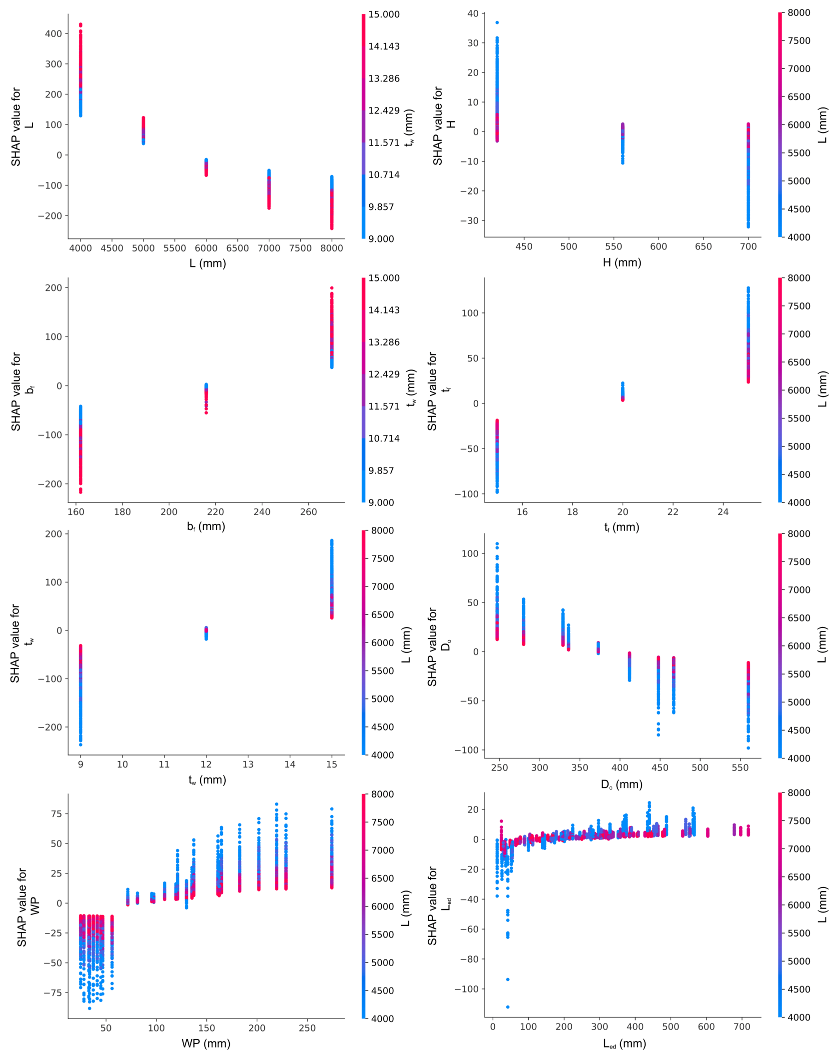

Figure 11: SHAP dependence plots for CatBoost model for predicting elastic buckling load

The SHAP dependence plots for w_{cr} show that L and b_f have the highest interactions with t_w, while other features interact with L most frequently. An increase in L results in an exponential decrease in w_{cr}, which is more pronounced for the cellular beams with thicker webs. An increase in H results in reductions of w_{cr} for the beams with short spans and smaller reductions or no reduction for the beams with long spans, which can be seen from the comparison of the SHAP values for the beams with long spans (red dots) with the SHAP values for the beams with short spans (blue dots). These results indicate that the elastic buckling load of the beams with short spans was likely governed by the local web buckling. In contrast, the elastic buckling load of the beams with long spans was likely governed by the global lateral-torsional buckling of the beams, which was less sensitive to the changes in H for the beams considered in this study. The elastic buckling loads increase when b_f increases, especially for the beams with thicker webs. The elastic buckling load, w_{cr}, increases when t_f and t_w increase, especially in the beams with short spans. The elastic buckling loads reduce when D_o goes up. The w_{cr} reduction is more significant due to the D_o increase for the cellular beams with short spans. When WP increases, w_{cr} increases, especially for the beams with short spans. The plot also shows that web posts with widths of 56 mm and narrower contribute to w_{cr} reductions, indicated by the negative SHAP values, while web posts of 72 mm wide and wider contribute to w_{cr} positively. An increase in L_{ed} results in a more significant increase in w_{cr} for the beams with short spans and a smaller increase in w_{cr} for those with long spans. It should be noted that the beam elastic buckling loads in the dataset were obtained for different buckling modes, including global lateral-torsional buckling of the beams, local buckling of the web posts, and their interaction. Therefore, the effects of the ML model features discussed above reflect possible changes in the buckling modes when the beam geometry changes.

The SHAP dependence plots for w_{max} demonstrate that the strongest interactions are between L and t_w, H and D_o, b_f/t_f and WP, and L_{ed} and b_f. All other features of the ultimate load model interact with L the most. An increase in L results in a w_{max} reduction, which is more pronounced in the cellular beams with thicker webs. Increases in H, b_f, t_f, and t_w cause an increase in w_{max}, which goes down when D_o increases. An increase in WP makes the beam ultimate load higher, which is more pronounced for the beams with short spans. Similar to the observed effect of WP on w_{cr}, web post widths up to 56 mm have a negative contribution to the beam ultimate load, while web posts of 72 mm wide and wider contribute to the beam ultimate load positively. It implies that the beam ultimate load in the dataset was governed by the web post strength when WP was 56 mm or lower. The beam ultimate load becomes higher when L_{ed} and F_y increase. The positive effect of F_y on w_{max} is more significant in the beams with short spans and when F_y increases from 235 to 355 MPa compared with the F_y increase from 355 to 440 MPa.

Fig. 13 shows contour plots of w_{cr} predicted by the developed CatBoost model as functions of H/D_o and S_o/D_o (where S_o is the center-to-center spacing of the web openings) for the beams with different span lengths and cross-section dimensions. The beam designations are presented in the L-H-t_w-b_f-t_f format, with all dimensions in mm. Fig. 13 demonstrates that S_o/D_o has a greater influence on w_{cr} than H/D_o for the beams with short spans. For many short-span beams, an increase in S_o/D_o from 1.1 to 1.3 results in a greater increase in w_{cr} than a further increase in S_o/D_o from 1.3 to 1.49. It indicates that web opening spacing of approximately $1.3D_o$ is optimal for many short-span beams. Only short-span beams with H=420 mm and t_w=9 mm demonstrate an approximately uniform w_{cr} increase when S_o/D_o increases from 1.1 to 1.49. The long-span beams show a wider variety of the w_{cr} contour shapes. For example, H/D_o has a more significant effect on w_{cr} than S_o/D_o for the beam with H=420 mm, t_w=9 mm, b_f=162 mm, and t_f=15 mm compared with other analyzed beams. The w_{cr} values for the 8000-420-9-162-15 and 8000-700-15-162-15 beams reduce slightly when S_o/D_o increase from 1.1 to 1.3 and increase with the further increase in S_o/D_o from 1.3 to 1.49. However, it should be noted that the absolute magnitude of the w_{cr} change is relatively small in those cases. The contour plots also show the effects of the cross-section dimensions on the w_{cr} values of the cellular beams discussed earlier in the paper.

Figs. 14 and 15 present contour plots of w_{max} predicted by the CatBoost model as functions of H/D_o and S_o/D_o for the cellular beams made from steel with F_y of 235 and 440 MPa, respectively. The beam designation format is as described previously, with the steel yield strength added at the end. Similar to w_{cr}, S_o/D_o shows a more significant effect on w_{max} than H/D_o for most of the considered beams. The effect of the opening diameter on w_{max} is more pronounced in the H/D_o range from 1.25 to approximately 1.45 for many beams. A further increase in H/D_o at a constant S_o/D_o value changes w_{max} insignificantly. It can also be seen from the contour plots that for some beams (see 4000-420-15-162-15-235, 7000-420-15-162-15-235, 4000-420-15-270-15-440, and 7000-700-9-270-15-440, for example), the effects of H/D_o varying in the range from 1.25 to 1.45 on w_{max} are relatively small when S_o/D_o is low. They become more pronounced as S_o/D_o increases. The S_o/D_o of approximately 1.3 is optimal for many considered beams. The effects of the cross-section dimensions on w_{max} discussed earlier in the paper can also be seen from Figs. 14 and 15.

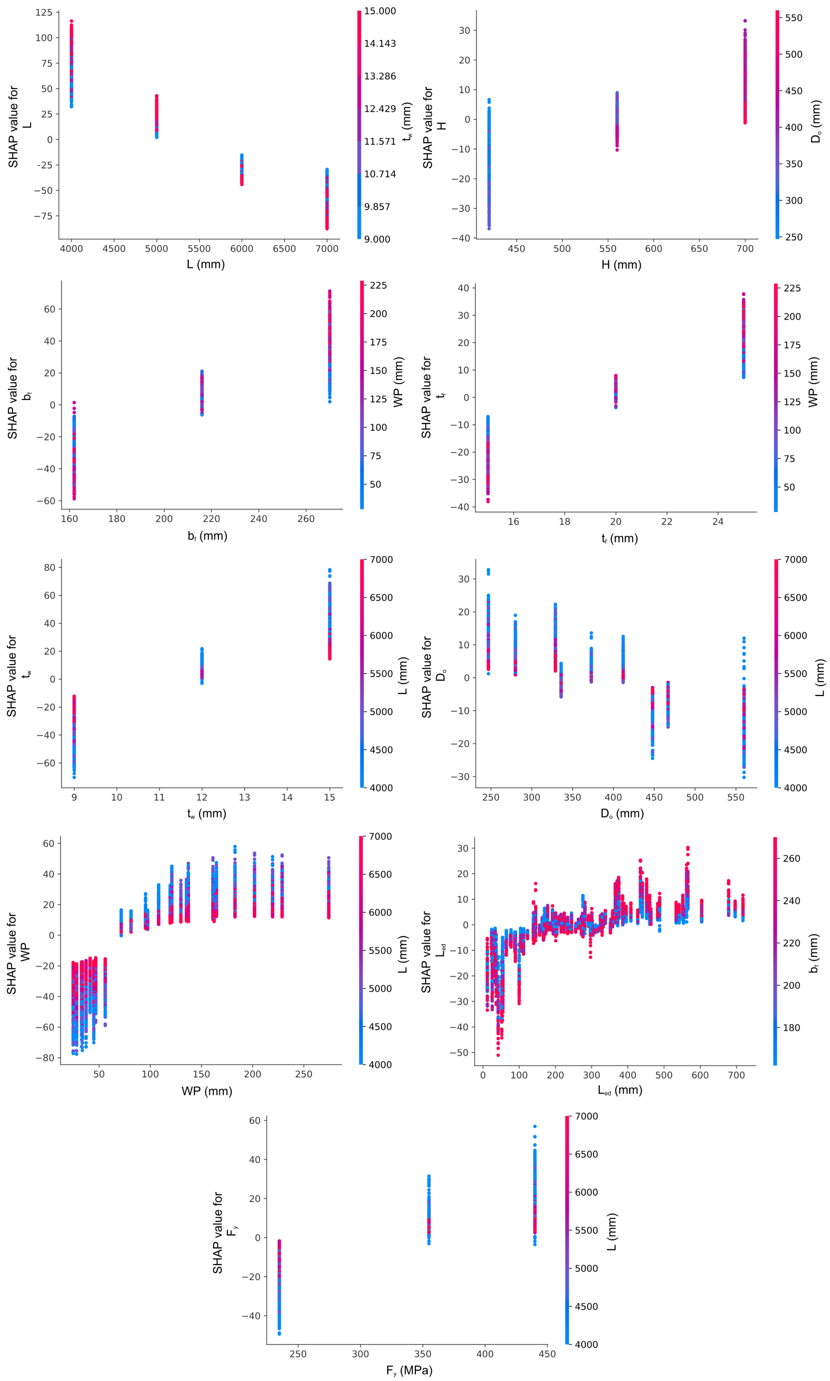

Figure 12: SHAP dependence plots for CatBoost model for predicting ultimate load

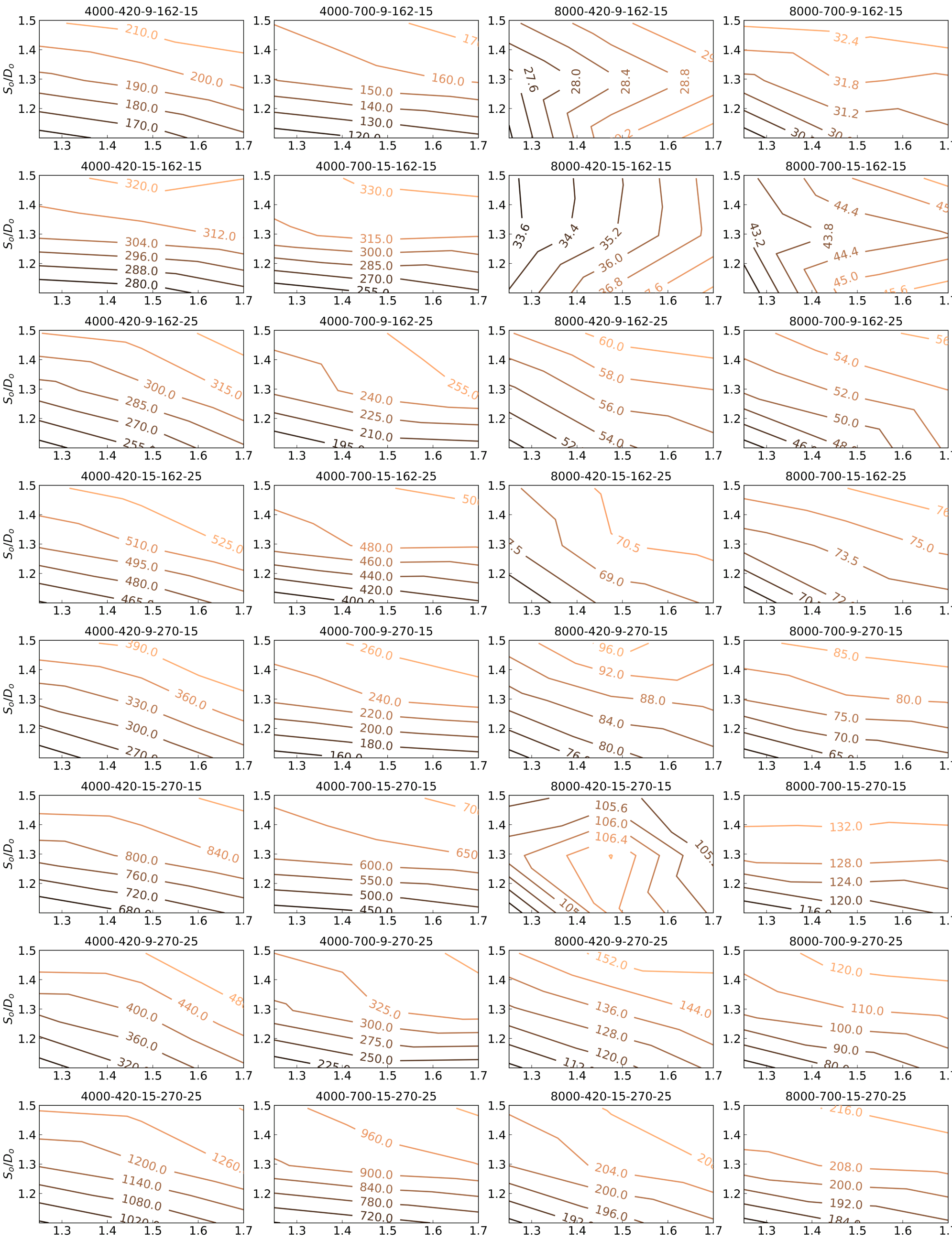

Figure 13: Contour plots of w_{cr} (kN/m) as functions of H/D_o and S_o/D_o

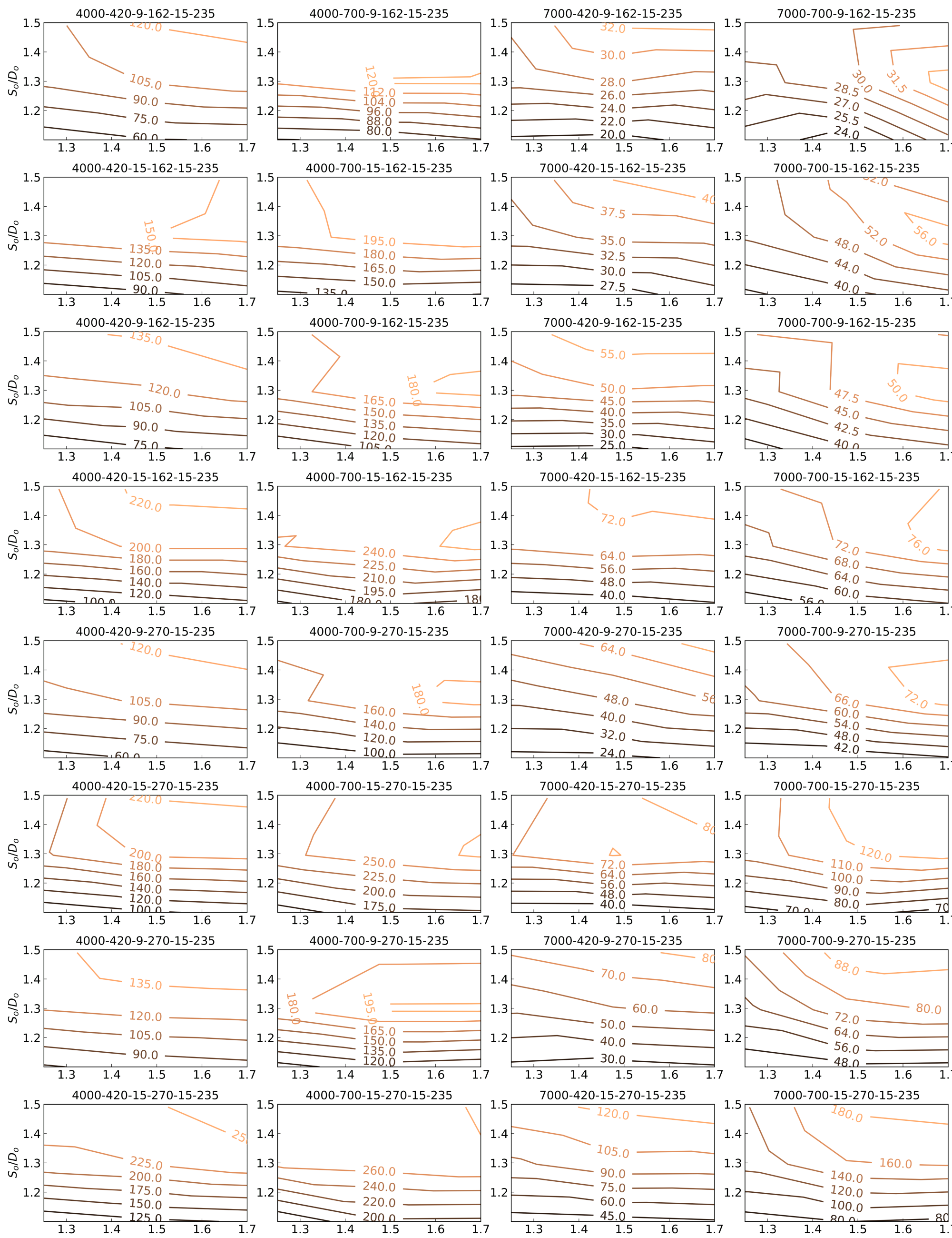

Figure 14: Contour plots of w_{max} (kN/m) as functions of H/D_o and S_o/D_o for beams made from 235 MPa steel

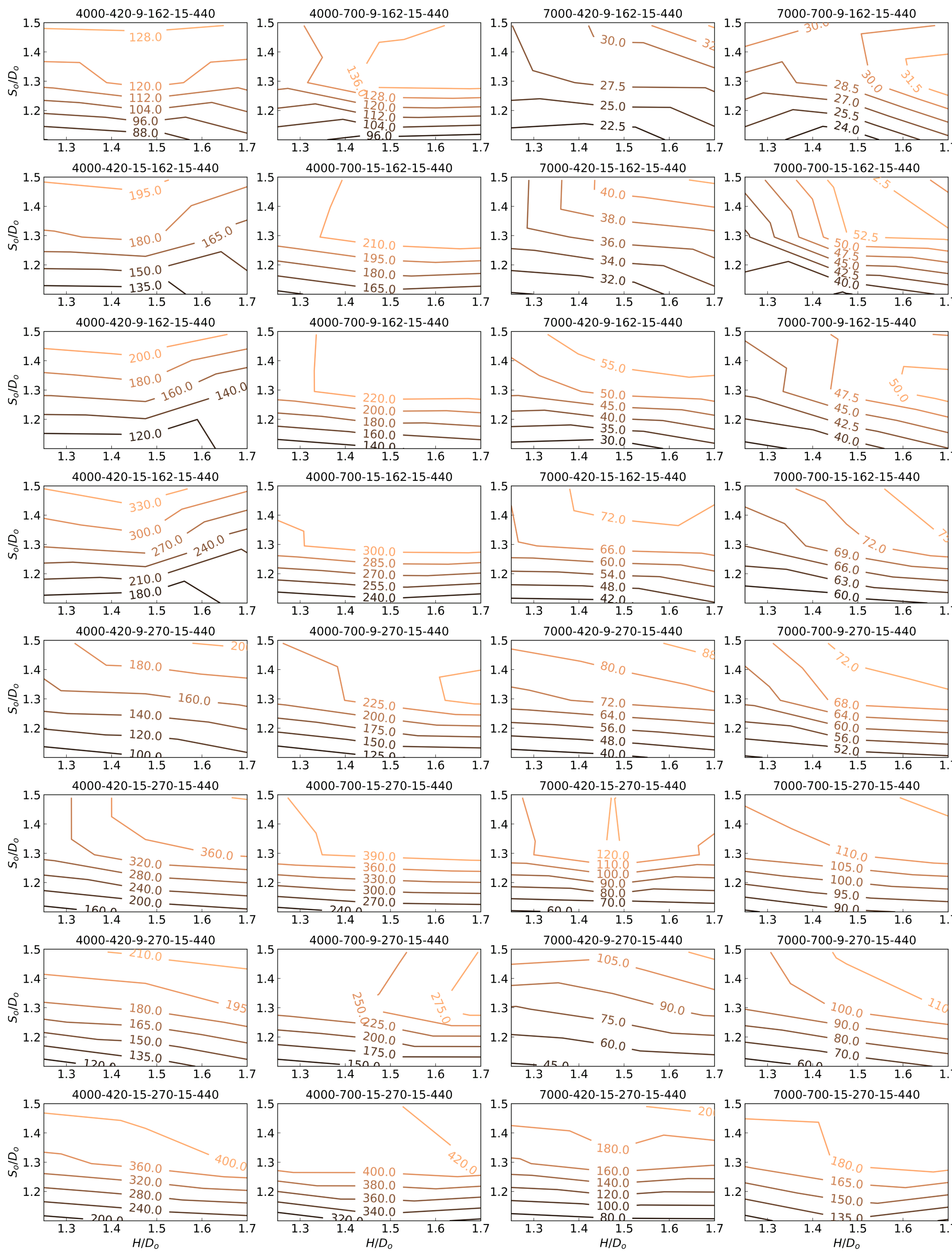

Figure 15: Contour plots of w_{max} (kN/m) as functions of H/D_o and S_o/D_o for beams made from 440 MPa steel

6. Comparisons of ultimate loads of cellular beams predicted by ML models, SCI P355, and AISC Design Guide 31

The ultimate loads of cellular beams predicted by the developed ML models were compared with the nominal beam strengths determined per SCI P355 [2] and AISC Design Guide 31 [87]. According to SCI P355 and AISC Design Guide 31, the cellular beam strength may be governed by shear resistance of perforated beam section, shear resistance of solid beam section, shear buckling resistance of perforated web, bending resistance of beam at the centerline of opening, bending resistance of tees, web post shear resistance, and web post buckling resistance. The beam, tee, and web post resistances are computed per EN 1993-1-1 [12] and EN 1993-1-5 [100] in SCI P355 and per AISC 360 [101] in AISC Design Guide 31. The most significant differences between the SCI P355 and AISC Design Guide 31 provisions are in the web post buckling resistance and lateral-torsional buckling calculations [102]. In SCI P355, the web post buckling resistance is calculated using analytical equations, which account for the web post slenderness, while AISC Design Guide 31 adopted empirical equations from SCI P100 [27]. SCI P355 also requires checking web shear buckling near openings, whereas AISC Design Guide 31 does not include such a requirement.

The SCI P355 provisions apply to cellular beams with the following geometric limits: $H/D_o \geq 1.25$, $2.0 \geq S_o/D_o \geq 1.3$, $L_{ed}/D_o \geq 0.5$, and the depth of tees not less than t_f+30 mm. The beams considered in the study had the following parameters: $1.70 \geq H/D_o \geq 1.25$, $1.49 \geq S_o/D_o \geq 1.10$, $1.49 \geq L_{ed}/D_o \geq 0.04$, and the depth of tees between t_f+29.5 mm and t_f+136.5 mm. S_o/D_o, L_{ed}/D_o, and the depth of tees of some beams did not comply with the SCI P355 limits. Therefore, the SCI P355 predictions were compared with the FE simulation results for all beams and 17,982 beams that met the geometric limits. AISC Design Guide 31 applies to cellular beams with $1.75 \geq H/D_o \geq 1.25$ and $1.50 \geq S_o/D_o \geq 1.08$. All beams considered in the present study complied with the AISC Design Guide 31 limits.

Fig. 16 compares the ultimate loads of the cellular beams from the FE simulations with those predicted by the developed ML models, SCI P355, and AISC Design Guide 31. Fig. 16 clearly shows that the developed ML models predict the ultimate loads of the cellular beams considerably better than SCI P355 and AISC Design Guide 31. For the best models, the mean ratio and the coefficient of variation of the ML predictions to the FE simulation results are 1.00 and 0.034, respectively. The coefficient of determination, R^2, is 0.997. The corresponding metrics for SCI P355 are 0.75, 0.253, and 0.638 for all beams and 0.75, 0.298, and 0.535 for the beams meeting the geometric limits. It is interesting to note that the SCI P355 provisions demonstrate better comparison with the FE simulation results when all beams are considered neglecting the geometric limits. AISC Design Guide 31 showed even worse accuracy than SCI P355, characterized by the mean ratio and the coefficient of variation of the prediction-to-FEA ratios of 0.69 and 0.429, and the coefficient of determination of 0.416.

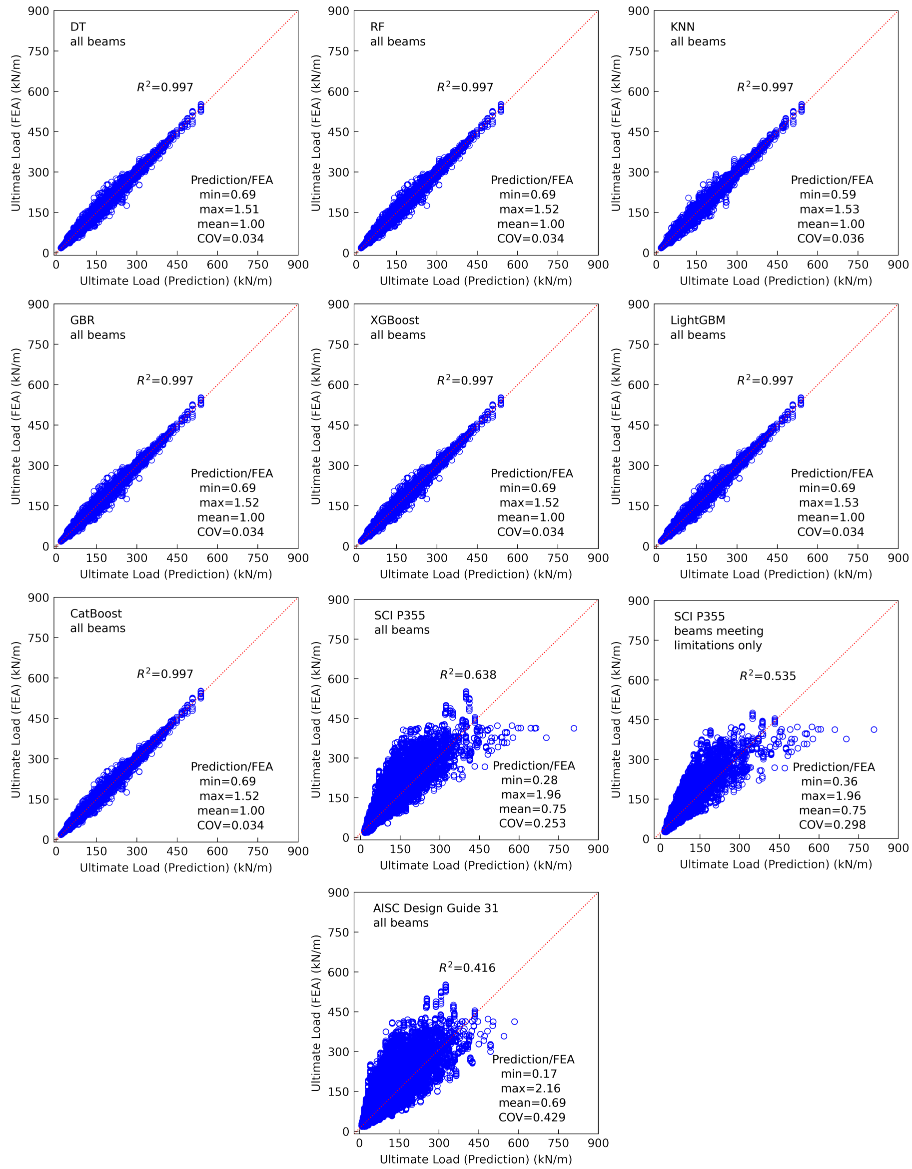

Figure 16: Comparisons of ultimate loads of steel cellular beams predicted by ML models, SCI P355, and AISC Design Guide 31 with FE simulation results

7. Web application

A user-friendly web application was created in the Streamlit framework (`https://streamlit.io`) to predict the elastic buckling and ultimate loads of steel cellular beams with the ML models developed in the present work. Fig. 17 demonstrates a flow chart of the web application.

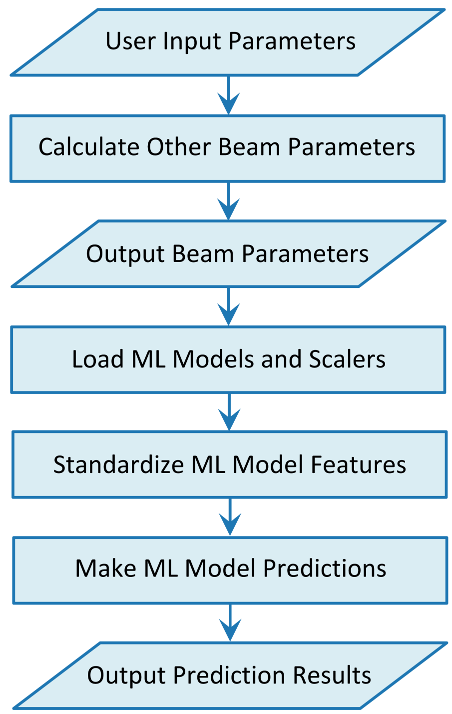

Figure 17: Flow chart of the web application

In the beginning, the user specifies the following parameters via the web application sliders and radio buttons: L, H, b_f, t_f, t_w, H/D_o, S_o/D_o, and F_y. Ranges of the parameters available in the application correspond to the feature ranges in the datasets used for the ML training. At the next step, the following parameters are computed: D_o, S_o, L_{ed}, the number of openings evenly spaced along the beam length, the cellular beam weight, and the percentage of the beam weight reduction due to the openings compared with the identical solid-web beam. The parameters specified by the user and the computed ones are displayed on the screen. Next, the developed ML models and scalers are loaded; the features are standardized, and predictions by all ML models considered in this study are made and displayed. The code runs automatically after any change of input variables. The prediction process takes only several seconds. The graphical user interface of the web application is presented in Fig. 18.

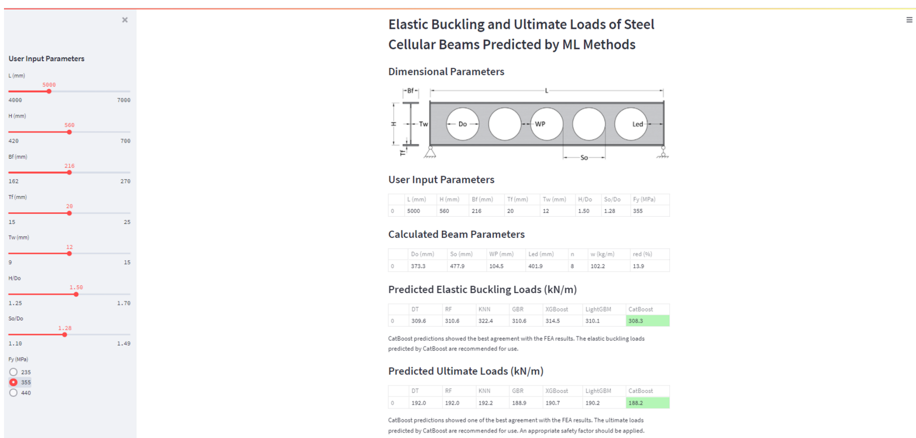

Figure 18: Graphical user interface of the developed web application

It was found challenging to deploy the web application with all considered ML to cloud platforms due to the application size and required computational resources, which exceeded the limits of free cloud accounts. Therefore, a lite version of the application based on CatBoost predictions was created and successfully deployed on Heroku at the following address: `https://scba-cb.herokuapp.com/`. The deployed lite version of the application opens and runs in any web browser on any device, including mobile.

It should be noted that the computational resources provided by the free Heroku account are sufficient for running the application by one user at a time. Multiple users can open the application, but it crashes when two or more users run the computations simultaneously. If that happens, it is recommended to close the application and use it later. The use of a paid Heroku account, which offers more powerful computational resources, would resolve this issue.

The source codes of the full and lite application versions can be accessed on GitHub at `https://github.com/vitdegtyarev/SCBA-Streamlit` and `https://github.com/vitdegtyarev/SCBA-Streamlit-CB`, respectively. The GitHub pages include instructions on how the web applications can be used independently from the cloud services on a local machine.

8. Conclusions

ML models for predicting the elastic buckling and ultimate loads of steel cellular beams were developed and optimized using the following algorithms: decision tree (DT), random forest (RF), k-nearest neighbor (KNN), gradient boosting regressor (GBR), extreme gradient boosting (XGBoost), light gradient boosting machine (LightGBM), and gradient boosting with categorical features support (CatBoost). Large datasets of FE simulation results from the literature [29], validated against experimental data, were employed to train and evaluate the ML models implemented in open-source Python-based libraries.

The ML models were optimized by tuning their hyperparameters via extensive grid and random searches and validated through the ten-fold cross-validation method. The final evaluation of the ML models was performed on the test sets unseen by the models during training. The elastic buckling and ultimate loads predicted by the optimized ML models demonstrated excellent agreements with the numerical data. The accuracy of the ultimate load predictions by the ML models exceeded the accuracy provided by the existing design provisions for steel cellular beams. An interactive Python-based notebook for predicting the elastic buckling and ultimate loads of steel cellular beams using the developed optimized ML models was created and made publicly available at the following link: `https://www.kaggle.com/vitdegtyarev/ml-models-for-cellular-beams?scriptVersionId=63075739`.

The developed ML models were explained and interpreted by evaluating the relative feature importance using the permutations and SHAP methods. SHAP feature dependence was also determined and discussed. It was demonstrated that the beam span length, beam flange width, and beam web thickness are the most important features in predicting the elastic buckling by the developed models, with the opening end distance and beam height being the least important parameters. The most important features in predicting the ultimate load are the beam span length, web post width, beam web thickness, and beam flange width. The opening end distance and opening diameter are the least important characteristics. These results align well with the mechanics-based knowledge demonstrating that the developed ML models can capture the web opening effects from the data used for their training. Contour plots of w_{cr} and w_{max} predicted by the CatBoost model as functions of H/D_o and S_o/D_o were presented and discussed. For most beams, S_o/D_o affects w_{cr} and w_{max} more significantly than H/D_o, with S_o/D_o=1.3 being the optimal value.

A web application for predicting the elastic buckling and ultimate loads was created in Streamlit. The lite version of the application has been deployed to the cloud at: `https://scba-cb.herokuapp.com/`. It can be opened and run in any web browser on any device, including mobile. The source codes of the full and lite application versions can be accessed on GitHub at `https://github.com/vitdegtyarev/SCBA-Streamlit` and `https://github.com/vitdegtyarev/SCBA-Streamlit-CB`, respectively.

The presented study demonstrates the opportunities for using ML methods for predicting the elastic buckling and ultimate loads of cellular beams. However, it should be noted that the developed models are based on the data for cellular beams with relatively short spans, not exceeding 8 m in the elastic buckling load dataset and 7 m in the ultimate load dataset. Therefore, the developed models are limited to beams with such spans. In modern construction, cellular beams are often used for spans ranging from 9 to 18 m [103]. Future work should concentrate on extending the datasets to the beams with longer spans and retraining the ML models using the extended data. The reliability of the ultimate load predictions by the ML models should also be evaluated, and an appropriate safety factor determined.

References

[1] P. K. Das, S. L. Srimani, Handbook for the design of castellated beams, Central Mechanical Engineering Research Institute, Oxford & IBH Publishing Company, New Delhi, Delhi, India, 1984.

[2] R. Lawson, S. Hicks, Design of composite beams with large web openings SCI P355, Steel Construction Institute, Berkshire, UK, 2011.

[3] M. I. M. H. Martini, Elasto-plastic lateral torsional buckling of steel beams with perforated web, Ph.D. thesis, United Arab Emirates University (2011).

[4] A. M. Sweedan, Elastic lateral stability of I-shaped cellular steel beams, Journal of Constructional Steel Research 67 (2) (2011) 151–163. doi:10.1016/j.jcsr.2010.08.009.

[5] M. M. Sehwail, Lateral torsional buckling of steel i-section cellular beams, Ph.D. thesis, Eastern Mediterranean University (EMU)-Doğu Akdeniz Üniversitesi (DAÜ) (2013).

[6] D. Sonck, J. Belis, Lateral–torsional buckling resistance of cellular beams, Journal of Constructional Steel Research 105 (2015) 119–128. doi:10.1016/j.jcsr.2014.11.003.

[7] P. Panedpojaman, W. Sae-Long, T. Chub-uppakarn, Cellular beam design for resistance to inelastic lateral–torsional buckling, Thin-Walled Structures 99 (2016) 182–194. doi:10.1016/j.tws.2015.08.026.

[8] F. P. V. Ferreira, A. Rossi, C. H. Martins, Lateral-torsional buckling of cellular beams according to the possible updating of EC3, Journal of Constructional Steel Research 153 (2019) 222–242. doi:10.1016/j.jcsr.2018.10.011.

[9] K. D. Tsavdaridis, C. D'Mello, Web buckling study of the behaviour and strength of perforated steel beams with different novel web opening shapes, Journal of Constructional Steel Research 67 (10) (2011) 1605–1620. doi:10.1016/j.jcsr.2011.04.004.

[10] P. Panedpojaman, T. Thepchatri, S. Limkatanyu, Novel design equations for shear strength of local web-post buckling in cellular beams, Thin-Walled Structures 76 (2014) 92–104. doi:10.1016/j.tws.2013.11.007.

[11] L. F. Grilo, R. H. Fakury, G. de Souza Veríssimo, et al., Design procedure for the web-post buckling of steel cellular beams, Journal of Constructional Steel Research 148 (2018) 525–541. doi:10.1016/j.jcsr.2018.06.020.

[12] BS EN 1993-1-1: 2005. Design of steel structures–Part 1-1 General rules and rules for buildings, British Standards Institution, London, UK, 2005.

[13] ANSI/AISC 360-10. Specification for structural steel buildings, American Institute of Steel Construction, Chicago, Illinois, USA, 2010.

[14] K. F. Chung, T. Liu, A. Ko, Investigation on Vierendeel mechanism in steel beams with circular web openings, Journal of Constructional Steel Research 57 (5) (2001) 467–490. doi:10.1016/S0143-974X(00)00035-3.

[15] L. Kang, S. Hong, X. Liu, Shear behaviour and strength design of cellular beams with circular or elongated openings, Thin-Walled Structures 160 (2021) 107353. doi:10.1016/j.tws.2020.107353.

[16] E. Ellobody, Nonlinear analysis of cellular steel beams under combined buckling modes, Thin-Walled Structures 52 (2012) 66–79. doi:10.1016/j.tws.2011.12.009.

[17] K. D. Tsavdaridis, C. D'Mello, Vierendeel bending study of perforated steel beams with various novel web opening shapes through nonlinear finite-element analyses, Journal of Structural Engineering 138 (10) (2012) 1214–1230. doi:10.1061/(ASCE)ST.1943-541X.0000562.

[18] N. D. Lagaros, L. D. Psarras, M. Papadrakakis, G. Panagiotou, Optimum design of steel structures with web openings, Engineering Structures 30 (9) (2008) 2528–2537. doi:10.1016/j.engstruct.2008.02.002.

[19] K. Tsavdaridis, C. D'Mello, FE investigation of perforated sections with standard and non-standard web opening configurations and sizes, in: The 6th International Conference on Advances is Steel Structures (ICASS 2009), 16-18 December 2009, pp. 213–220.

[20] F. Erdal, M. P. Saka, Ultimate load carrying capacity of optimally designed steel cellular beams, Journal of Constructional Steel Research 80 (2013) 355–368. doi:10.1016/j.jcsr.2012.10.007.

[21] A. Jamadar, P. Kumbhar, Parametric study of castellated beam with circular and diamond shaped openings, International Research Journal of Engineering and Technology 2 (2) (2015) 715–722.

[22] K. D. Tsavdaridis, J. J. Kingman, V. V. Toropov, Application of structural

topology optimisation to perforated steel beams, Computers & Structures 158 (2015) 108–123. doi:10.1016/j.compstruc.2015.05.004.

[23] S. G. Morkhade, L. M. Gupta, An experimental and parametric study on steel beams with web openings, International Journal of Advanced Structural Engineering (IJASE) 7 (3) (2015) 249–260. doi:10.1007/s40091-015-0095-4.

[24] K. D. Tsavdaridis, G. Galiatsatos, Assessment of cellular beams with transverse stiffeners and closely spaced web openings, Thin-Walled Structures 94 (2015) 636–650. doi:10.1016/j.tws.2015.05.005.

[25] V. Akrami, S. Erfani, Review and assessment of design methodologies for perforated steel beams, Journal of Structural Engineering 142 (2) (2016) 04015148. doi:10.1061/(ASCE)ST.1943-541X.0001421.

[26] SEI/ASCE 23-97. Specifications for structural steel beams with web openings, American Society of Civil Engineers, Reston, VA, USA, 1999.

[27] J. Ward, Design of composite and non-composite cellular beams SCI P100, Steel Construction Institute, Berkshire, UK, 1990.

[28] K. F. Chung, C. Liu, A. Ko, Steel beams with large web openings of various shapes and sizes: an empirical design method using a generalised moment-shear interaction curve, Journal of Constructional Steel Research 59 (9) (2003) 1177–1200. doi:10.1016/S0143-974X(03)00029-4.

[29] K. Rajana, K. D. Tsavdaridis, E. Koltsakis, Elastic and inelastic buckling of steel cellular beams under strong-axis bending, Thin-Walled Structures 156 (2020) 106955. doi:10.1016/j.tws.2020.106955.

[30] F. Jiang, Y. Jiang, H. Zhi, Y. Dong, H. Li, S. Ma, Y. Wang, Q. Dong, H. Shen, Y. Wang, Artificial intelligence in healthcare: past, present and

future, Stroke and Vascular Neurology 2 (4). doi:10.1136/svn-2017-000101.

[31] N. R. Tadapaneni, Artificial intelligence in finance and investments, International Journal of Innovative Research in Science, Engineering and Technology 9 (5) (2019) 2792–2795.

[32] Y. Ma, Z. Wang, H. Yang, L. Yang, Artificial intelligence applications in the development of autonomous vehicles: a survey, IEEE/CAA Journal of Automatica Sinica 7 (2) (2020) 315–329. doi:10.1109/JAS.2020.1003021.

[33] T. Wuest, D. Weimer, C. Irgens, K.-D. Thoben, Machine learning in manufacturing: advantages, challenges, and applications, Production & Manufacturing Research 4 (1) (2016) 23–45. doi:10.1080/21693277.2016.1192517.

[34] N. Kartam, I. Flood, J. H. Garrett, Artificial Neural Networks for Civil Engineers: Fundamentals and Applications, American Society of Civil Engineers, 1997.

[35] H. Adeli, Neural networks in civil engineering: 1989–2000, Computer-Aided Civil and Infrastructure Engineering 16 (2) (2001) 126–142. doi:10.1111/0885-9507.00219.

[36] H. Salehi, R. Burgueño, Emerging artificial intelligence methods in structural engineering, Engineering Structures 171 (2018) 170–189. doi:10.1016/j.engstruct.2018.05.084.

[37] H. Sun, H. V. Burton, H. Huang, Machine learning applications for building structural design and performance assessment: state-of-the-art review,

Journal of Building Engineering (2020) 101816doi:10.1016/j.jobe.2020.101816.

[38] M. Z. Naser, Systematic Integration of Artificial Intelligence Toward Evaluating Response of Materials and Structures in Extreme Conditions, Springer Transactions in Civil and Environmental Engineering, Springer, 2021, p. 183–212. doi:10.1007/978-981-15-5772-9_10.

[39] M. Naser, Mechanistically informed machine learning and artificial intelligence in fire engineering and sciences, Fire Technology (2021) 1–44doi:10.1007/s10694-020-01069-8.

[40] H.-T. Thai, Machine learning for structural engineering: A state-of-the-art review, Structures (2022) 448–491doi:10.1016/j.istruc.2022.02.003.

[41] A. Çevik, A. E. Kurtoğlu, M. Bilgehan, M. E. Gülşan, H. M. Albegmprli, Support vector machines in structural engineering: a review, Journal of Civil Engineering and Management 21 (3) (2015) 261–281. doi:10.3846/13923730.2015.1005021.

[42] C. Liu, C. Liu, C. Liu, X. Huang, J. Miao, W. Xu, Fire damage identification in RC beams based on support vector machines considering vibration test, KSCE Journal of Civil Engineering 23 (10) (2019) 4407–4416. doi:10.1007/s12205-019-2353-7.

[43] J. Zhang, G. Ma, Y. Huang, F. Aslani, B. Nener, et al., Modelling uniaxial compressive strength of lightweight self-compacting concrete using random forest regression, Construction and Building Materials 210 (2019) 713–719. doi:10.1016/j.conbuildmat.2019.03.189.

[44] S. Mangalathu, S.-H. Hwang, J.-S. Jeon, Failure mode and effects analysis of rc members based on machine-learning-based shapley additive ex-

planations (SHAP) approach, Engineering Structures 219 (2020) 110927. doi:10.1016/j.engstruct.2020.110927.

[45] R. Solhmirzaei, H. Salehi, V. Kodur, M. Naser, Machine learning framework for predicting failure mode and shear capacity of ultra high performance concrete beams, Engineering Structures 224 (2020) 111221. doi:10.1016/j.engstruct.2020.111221.

[46] M. S. Ahmad, S. M. Adnan, S. Zaidi, P. Bhargava, A novel support vector regression (SVR) model for the prediction of splice strength of the unconfined beam specimens, Construction and Building Materials 248 (2020) 118475. doi:10.1016/j.conbuildmat.2020.118475.

[47] S. Mangalathu, H. Jang, S.-H. Hwang, J.-S. Jeon, Data-driven machine-learning-based seismic failure mode identification of reinforced concrete shear walls, Engineering Structures 208 (2020) 110331. doi:10.1016/j.engstruct.2020.110331.

[48] M. Naser, H. Salehi, Machine learning-driven assessment of fire-induced concrete spalling of columns, ACI Materials Journal 117 (6) (2020) 7–16. doi:10.14359/51728120.

[49] D.-C. Feng, Z.-T. Liu, X.-D. Wang, Y. Chen, J.-Q. Chang, D.-F. Wei, Z.-M. Jiang, Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach, Construction and Building Materials 230 (2020) 117000. doi:10.1016/j.conbuildmat.2019.117000.

[50] T. Nguyen-Sy, J. Wakim, Q.-D. To, M.-N. Vu, T.-D. Nguyen, T.-T. Nguyen, Predicting the compressive strength of concrete from its compositions and age using the extreme gradient boosting method, Construction and Building Materials 260 (2020) 119757. doi:10.1016/j.conbuildmat.2020.119757.

[51] M. R. Kaloop, D. Kumar, P. Samui, J. W. Hu, D. Kim, Compressive strength prediction of high-performance concrete using gradient tree boosting machine, Construction and Building Materials 264 (2020) 120198. doi:10.1016/j.conbuildmat.2020.120198.

[52] A. Marani, M. L. Nehdi, Machine learning prediction of compressive strength for phase change materials integrated cementitious composites, Construction and Building Materials 265 (2020) 120286. doi:10.1016/j.conbuildmat.2020.120286.

[53] W. B. Chaabene, M. Flah, M. L. Nehdi, Machine learning prediction of mechanical properties of concrete: Critical review, Construction and Building Materials 260 (2020) 119889. doi:10.1016/j.conbuildmat.2020.119889.

[54] H. Nguyen, T. Vu, T. P. Vo, H.-T. Thai, Efficient machine learning models for prediction of concrete strengths, Construction and Building Materials 266 (2021) 120950. doi:10.1016/j.conbuildmat.2020.120950.

[55] M.-C. Kang, D.-Y. Yoo, R. Gupta, Machine learning-based prediction for compressive and flexural strengths of steel fiber-reinforced concrete, Construction and Building Materials 266 (2021) 121117. doi:10.1016/j.conbuildmat.2020.121117.

[56] M. Su, Q. Zhong, H. Peng, S. Li, Selected machine learning approaches for predicting the interfacial bond strength between FRPs and concrete, Construction and Building Materials 270 (2021) 121456. doi:10.1016/j.conbuildmat.2020.121456.

[57] O. B. Olalusi, P. O. Awoyera, Shear capacity prediction of slender reinforced concrete structures with steel fibers using machine learning, Engi-

neering Structures 227 (2021) 111470. doi:10.1016/j.engstruct.2020.111470.

[58] H. D. Nguyen, G. T. Truong, M. Shin, Development of extreme gradient boosting model for prediction of punching shear resistance of r/c interior slabs, Engineering Structures 235 (2021) 112067. doi:10.1016/j.engstruct.2021.112067.

[59] D.-C. Feng, W.-J. Wang, S. Mangalathu, G. Hu, T. Wu, Implementing ensemble learning methods to predict the shear strength of RC deep beams with/without web reinforcements, Engineering Structures 235 (2021) 111979. doi:10.1016/j.engstruct.2021.111979.

[60] J. Rahman, K. S. Ahmed, N. I. Khan, K. Islam, S. Mangalathu, Data-driven shear strength prediction of steel fiber reinforced concrete beams using machine learning approach, Engineering Structures 233 (2021) 111743. doi:10.1016/j.engstruct.2020.111743.

[61] D.-C. Feng, B. Cetiner, M. R. Azadi Kakavand, E. Taciroglu, Data-driven approach to predict the plastic hinge length of reinforced concrete columns and its application, Journal of Structural Engineering 147 (2) (2021) 04020332. doi:10.1061/(ASCE)ST.1943-541X.0002852.

[62] S. Mangalathu, H. Shin, E. Choi, J.-S. Jeon, Explainable machine learning models for punching shear strength estimation of flat slabs without transverse reinforcement, Journal of Building Engineering 39 (2021) 102300. doi:10.1016/j.jobe.2021.102300.

[63] S.-H. Hwang, S. Mangalathu, J. Shin, J.-S. Jeon, Machine learning-based approaches for seismic demand and collapse of ductile reinforced concrete building frames, Journal of Building Engineering 34 (2021) 101905. doi:10.1016/j.jobe.2020.101905.

[64] B. Fu, D.-C. Feng, A machine learning-based time-dependent shear strength model for corroded reinforced concrete beams, Journal of Building Engineering 36 (2021) 102118. doi:10.1016/j.jobe.2020.102118.

[65] Z. Liu, A. Guo, Empirical-based support vector machine method for seismic assessment and simulation of reinforced concrete columns using historical cyclic tests, Engineering Structures 237 (2021) 112141. doi:10.1016/j.engstruct.2021.112141.

[66] V. V. Degtyarev, Machine learning models for predicting bond strength of deformed bars in concrete, ACI Structural Journal (in press).

[67] M. Kumar, N. Yadav, Buckling analysis of a beam–column using multilayer perceptron neural network technique, Journal of the Franklin Institute 350 (10) (2013) 3188–3204. doi:10.1016/j.jfranklin.2013.07.016.

[68] A. Tashakori, H. Adeli, Optimum design of cold-formed steel space structures using neural dynamics model, Journal of Constructional Steel Research 58 (2002) 1545–1566. doi:10.1016/S0143-974X(01)00105-5.

[69] I. H. Guzelbey, A. Cevik, A. Erklig, Prediction of web crippling strength of cold-formed steel sheetings using neural networks, Journal of Constructional Steel Research 62 (10) (2006) 962–973. doi:10.1016/j.jcsr.2006.01.008.

[70] M. Pala, A new formulation for distortional buckling stress in cold-formed steel members, Journal of Constructional Steel Research 62 (7) (2006) 716–722. doi:10.1016/j.jcsr.2005.09.011.

[71] M. Pala, N. Caglar, A parametric study for distortional buckling stress on

cold-formed steel using a neural network, Journal of Constructional Steel Research 63 (5) (2007) 686–691. doi:10.1016/j.jcsr.2006.07.005.

[72] M. D'Aniello, E. M. Güneyisi, R. Landolfo, K. Mermerdaş, Analytical prediction of available rotation capacity of cold-formed rectangular and square hollow section beams, Thin-Walled Structures 77 (2014) 141–152. doi:10.1016/j.tws.2013.09.015.

[73] S. Lee, T. P. Vo, H.-T. Thai, J. Lee, V. Patel, Strength prediction of concrete-filled steel tubular columns using Categorical Gradient Boosting algorithm, Engineering Structures 238 (2021) 112109. doi:10.1016/j.engstruct.2021.112109.

[74] M. A. B. Kabir, A. S. Hasan, A. M. Billah, Failure mode identification of column base plate connection using data-driven machine learning techniques, Engineering Structures 240 (2021) 112389. doi:10.1016/j.engstruct.2021.112389.

[75] Y. Xu, B. Zheng, M. Zhang, Capacity prediction of cold-formed stainless steel tubular columns using machine learning methods, Journal of Constructional Steel Research 182 (2021) 106682. doi:10.1016/j.jcsr.2021.106682.

[76] X. Guan, H. Burton, M. Shokrabadi, Z. Yi, Seismic drift demand estimation for steel moment frame buildings: From mechanics-based to data-driven models, Journal of Structural Engineering 147 (6) (2021) 04021058. doi:10.1061/(ASCE)ST.1943-541X.0003004.

[77] V. V. Degtyarev, Neural networks for predicting shear strength of CFS channels with slotted webs, Journal of Constructional Steel Research 177 (2021) 106443. doi:10.1016/j.jcsr.2020.106443.

[78] V. V. Degtyarev, M. Z. Naser, Boosting machines for predicting shear strength of CFS channels with staggered web perforations, Structures 34 (2021) 3391–3403. doi:10.1016/j.istruc.2021.09.060.

[79] M. Z. Naser, V. Kodur, H.-T. Thai, R. Hawileh, J. Abdalla, V. V. Degtyarev, Structuresnet and firenet: Benchmarking databases and machine learning algorithms in structural and fire engineering domains, Journal of Building Engineering 44 (2021) 102977. doi:10.1016/j.jobe.2021.10297.

[80] V. V. Degtyarev, Predicting shear strength of cfs channels with slotted webs by machine learning models, Architecture, Structures and Constructiondoi:10.1007/s44150-021-00001-0.

[81] Y. Sharifi, A. Moghbeli, M. Hosseinpour, H. Sharifi, Neural networks for lateral torsional buckling strength assessment of cellular steel I-beams, Advances in Structural Engineering 22 (9) (2019) 2192–2202. doi:10.1177/1369433219836176.

[82] AS 4100:1990. Steel structures, Standards Association of Australia, Sydney, Australia, 1990.

[83] M. Abambres, K. Rajana, K. D. Tsavdaridis, T. P. Ribeiro, Neural network-based formula for the buckling load prediction of I-section cellular steel beams, Computers 8 (1) (2019) 2. doi:10.3390/computers8010002.

[84] F. P. V. Ferreira, R. Shamass, V. Limbachiya, K. D. Tsavdaridis, C. H. Martins, Lateral–torsional buckling resistance prediction model for steel cellular beams generated by Artificial Neural Networks (ANN), Thin-Walled Structures 170 (2022) 108592. doi:10.1016/j.tws.2021.108592.

[85] V. Limbachiya, R. Shamass, Application of artificial neural networks for

web-post shear resistance of cellular steel beams, Thin-Walled Structures 161 (2021) 107414. doi:doi.org/10.1016/j.tws.2020.107414.

[86] S. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, arXiv preprint arXiv:1705.07874.

[87] S. Fares, J. Coulson, D. Dinehart, Design Guide 31: Castellated and Cellular Beam Design, American Institute of Steel Construction, Chicago, IL, USA, 2016.

[88] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning: data mining, inference, and prediction, Springer Science & Business Media, 2009.

[89] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems, O'Reilly Media, 2019.

[90] L. Breiman, Random forests, Machine learning 45 (1) (2001) 5–32. doi:10.1023/A:1010933404324.

[91] T. K. Ho, Random decision forests, in: Proceedings of 3rd international conference on document analysis and recognition, Vol. 1, IEEE, 1995, pp. 278–282. doi:10.1109/ICDAR.1995.598994.

[92] J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of Statistics (2001) 1189–1232.

[93] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in Python, the Journal of Machine Learning Research 12 (2011) 2825–2830.

[94] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794. doi:10.1145/2939672.2939785.

[95] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, LightGBM: A highly efficient gradient boosting decision tree, Advances in Neural Information Processing Systems 30 (2017) 3146–3154.

[96] A. V. Dorogush, V. Ershov, A. Gulin, CatBoost: gradient boosting with categorical features support, arXiv preprint arXiv:1810.11363.

[97] M. Naser, A. Alavi, Insights into performance fitness and error metrics for machine learning, arXiv preprint arXiv:2006.00887.

[98] M. Z. Naser, An engineer's guide to eXplainable Artificial Intelligence and Interpretable Machine Learning: Navigating causality, forced goodness, and the false perception of inference, Automation in Construction 129 (2021) 103821. doi:10.1016/j.autcon.2021.103821.

[99] B. Peleg, P. Sudhölter, Introduction to the theory of cooperative games, Vol. 34, Springer Science & Business Media, 2007.

[100] BS EN 1993-1-5: 2006. Design of steel structures–Part 1-5 Plated structural elements, British Standards Institution, London, UK, 2006.

[101] ANSI/AISC 360-16. Specification for structural steel buildings, American Institute of Steel Construction, Chicago, Illinois, USA, 2016.

[102] C. Topkaya, S. Şahin, A comparative study of AISC-360 and EC3 strength limit states, International Journal of Steel Structures 11 (1) (2011) 13–27. doi:10.1007/S13296-011-1002-x.

[103] S. J. Hicks, R. M. Lawson, J. W. Rackham, P. Fordham, Comparative structure cost of modern commercial buildings, Steel Construction Institute, 2004.