Prediction of Failures in Steel Frames

Jina Yang
jyang633@wisc.edu

Hyeyoung Koh
hyeyoung.koh@wisc.edu

Erika Yeonseo Park
ypark258@wisc.edu

Project Link
https://github.com/JinaYang777/Project_Detection_of_Lung_Diseases_Chest_X-ray

Abstract

Machine learning algorithms are developed to predict structural failures of steel frames. To design a safe structure, it must be understood how structures behave and fail under uncertainties in all the factors affecting system performance such as geometric and material imperfections. We build a classification system that classifies steel frames with the uncertainties into a frame that experiences failure or no failure. Two non-symmetric planar steel frames that have different failure modes are selected for the experiments. The Monte Carlo sampling method was utilized to generate random features and structural finite element analyses were conducted to obtain class labels. kNN, decision tree, and random forest algorithms are used in the experiments that predict the failures of steel frames. The results explain that the random forest algorithm showed the best results with the highest accuracy that exceeds the target accuracy, 93%. The two steel frames showed the best performance with the same model. Therefore, the best model is applicable to the steel frames with different failure modes.

1. Introduction

As the structural failure may cause substantial loss of human life, protection against possible failure is of primary concern and the structure must neither fail locally nor globally. Building codes and design specifications are provided to help engineers to achieve the level of safety based on the criteria that any structure must meet. The current method for designing steel structures, AISC 360 [1], involves checking of individual members at the deterministic cross-section and material property values. However, uncertainty in geometric and material properties is ever-present and it would affect system performance. Although there is now various sophisticated structural analysis software used by structural engineers, it is difficult to predict the actual performance of steel frames with certainty due to ever-present uncertainties in material and geometric properties and structural loads. Hence, the uncertainties in all the factors affecting system performance need to be considered in modeling when evaluating the system failure capacity.

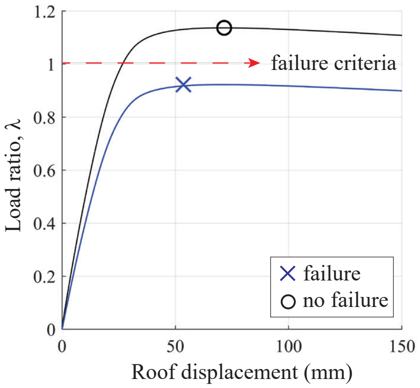

Figure 1. Failure criteria

Failure of a structure may either occur as the physical collapse, deflections that exceed predetermined values, or strength below the design values. The inelastic analysis provided in AISC 360 [1] requires modeling the structure and its members in detail to be able to track the response from the unloaded condition to the collapse load, as shown in Figure 1. Load ratio λ is defined as the ratio of applied load to the nominal strength. The structure does not experience failure if the ultimate strength of the structure reaches the nominal strength, whereas the insufficient ultimate strength causes the failure of a component or the entire structure. i.e., $\lambda = 1.0$ is the failure criterion because a frame has no failure and safe if λ is greater than 1.0 while it experiences failure and unsafe if λ does not reach 1.0.

The objective of this paper is to predict whether steel frames experience a failure or not under uncertainties in material and geometric properties and structural loads, by evaluating the ultimate strength of the frames. Two example frames are investigated that have the same layout but different failure modes, as shown in Figure 2, adopted from

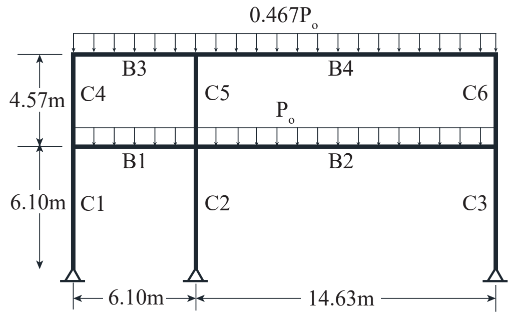

Figure 2. Dimensions and loads of the frames

Ziemian [21]. Each dataset has around fifty-thousand examples with six feature variables for each frame, which are labeled either with ”failure” or ”no failure” according to the ultimate load ratio that determines the failure status of the structure. Monte Carlo sampling method is utilized to incorporate the random variables into the simulation conducted in OpenSees [11]. To find the best model to predict the failure of steel frames, k-Nearest Neighbors (kNN), decision trees, and random forests are utilized. The model with the highest prediction accuracy are selected as the best model. The Frame 1 dataset is employed to find the best three models for each algorithm and the ultimate model among the three models. The best and ultimate models are applied to the Frame 2 dataset to confirm if they perform sufficiently well on the new dataset.

2. Related Work

The use of the inelastic method in practical design has increased with significant advances in computerized structural analysis. Previous studies have investigated the system reliability of steel frames for system-based design by the inelastic method. Buonopane and Schafer [4] compared the system reliabilities of a series of 2-dimensional steel frames with the same layout shown in Figure 2. The frames were controlled by gravity loads and the study considered structural loads and yield strength only as random variables. Buonopane [3] presented sensitivity analyses of two planar steel frames designed using first-order and second-order analysis. The uncertainties in yield strength, elastic modulus, residual stresses, and geometric imperfections were considered in analyses for symmetric frames. Zhang et al. [20] compared the system reliabilities of five planar steel structures. The study considered uncertainties in structural loads, strength and stiffness, cross-sectional dimensions, and sway imperfection. This study considers all the uncertainties covered in the previous studies [4, 3, 20].

The application of machine learning techniques to structural modeling with a consideration of uncertainties has recently increased. Hwang et al. [6] examined the effect of uncertain modeling parameters related to the seismic performance of reinforced concrete building frames by using multiple machine learning methods. The study developed a predictive model of maximum story drift of frames resulting from seismic loading. Both regression-based and classification-based tools are carried out to compare the collapse status of the model. All maximum story drifts are labeled either collapse or non-collapse and the total number of collapses is used to accomplish the collapse fragility curves. The results emphasized the importance of consideration of the effect of parameter uncertainties because ignoring the uncertainties in structural modeling underestimated the collapse risk that could lead to unsafe structural design. Kiani et al. [7] developed the framework for deriving seismic fragility curves with the implementation of classification-based tools. The influence of twenty-one uncertainty in the seismic input such as ground acceleration and spectrum intensity. Besides, the impact of an imbalanced dataset by performing the sensitivity of the applied tools on the size and type of dataset. The random forest method showed the best performance when validating against the experimental results. This study uses classification-based methods to examine the failure status of structural systems similar to both studies [6, 7], but applies the methods to steel frames subjected to gravity loads that is distinguished from the two studies.

3. Proposed Method

The main task of the project is to predict no failure ($\lambda > 1.0$) or failure ($\lambda < 0.0$) of the steel frames. With the labeled data, supervised learning is employed, which focuses on learning a classification model. The Frame 1 dataset is used to select the best models and algorithms by examining different models of kNN, decision trees, and random forests algorithms from $scikit - learn$ [12].

kNN algorithm is one of the simplest algorithms, that is widely used in practice. kNN predicts the target class label that is most often represented among the k most similar training examples for a given query point. The kNN algorithm in the classification setting is employed for the goal of this project, which is to predict no failure or failure of a given labeled data. Decision Tree algorithms are a type of supervised and eager learning, where the data is continuously splitted according to a certain parameter. The decision tree algorithm for the given task is classification trees since the class labels are ‘0’, and ‘1’, which are two discrete values. Random forest is implemented, which is known to perform well without hyperparameter tuning. Random forest uses different bootstrap samples to fit different trees and it uses random feature subsets on each node when splitting. Finally, the sum of the results from each tree is used to make predictions.

By using nested cross-validation (CV), different models for the three algorithms, kNN, decision trees, and random forests are evaluated with the Frame 1 dataset and the models with the highest generalization performance for each algorithm are selected to be compared with the Frame 2 dataset. Among the three models with different algorithms, the first two models of the three listed in descending order of generalization performance are compared by using CV to select the final model that maximizes the performance; prediction accuracy (Equation 1 [14]) is used as a metric for estimating generalization performance for each model and algorithm. Then, the chosen model is evaluated as the fitted model with the test set by using a bootstrap method.

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n} L(\hat{y}^{[i]}, y^{[i]}) \times 100, \tag{1}$$

$$L(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} = y, \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where n = the number of examples in the test set, $\hat{y}$ = predicted class label, y = the ground truth

Despite the selection of the best model, the two unselected models are evaluated for their functionality. Thus, each model that corresponds to each algorithm is evaluated. Also, the three models are applied to the different dataset, which is from Frame 2, to check on the assumption that the best model chosen from the first data is still the best model on the second data. Also, the performance of the three models obtained from the first data is examined to check whether the three models provide sufficient accuracy.

3.1. Feature Selection

Prior to comparing models and algorithms, z-score standardization (Equation 3) and feature selection of the data is proceeded to complement kNN algorithm's shortcomings. The kNN algorithm is profoundly affected by the distances between samples that standardization of the data points is necessary.

$$x_{std}^{[i]} = \frac{x^{[i]} - \mu_x}{\sigma_x} \tag{3}$$

where $x^{[i]}$ = a data point, μ_x = the population mean of x, σ_x = the population standard deviation of x

kNN algorithm is also susceptible to the number of dimensions because higher numbers of dimensions require the larger volume in the hyperspace, which is the curse of dimensionality. With the larger volume in the hyperspace, the neighbors become less similar to the query point; it leads to lower accuracy of the model. Although all features in the dataset may be significant, not all features may have a major impact on the model. Thus, Principal Component Analysis (PCA) via Scikit-learn [12] and Sequential Feature forward Selection (SFS) via Mlxtend [13] are employed to select the best features that would enhance the performance of the kNN model.

Principal Component Analysis The basic idea of using PCA is to select variables according to the magnitude from largest to smallest in absolute values of their coefficients through ranking principal components by importance through their explained variance. However, it may be difficult to interpret PCA because it consists of a complex mixture of the original features. Thus, the information of classes is not considered because the principal components are calculated only from features.

Sequential Feature Forward Selection To supplement the flaws of PCA, SFS from Mlxtend [13] is employed to select the best features among the six features in the original dataset. The SFS algorithm starts by having no features in the model, then adds a subset of features in each iteration. The iteration continues until a new variable fails to improve the model performance. SFS supplements the flaws of PCA by providing information of chosen subsets. SFS not only provides CV scores by using each subset, but also provides the feature indices. This method is used to select the features that are most significant to the data.

The scaled and reduced dataset is only used for kNN because scaling is not necessary for tree-based algorithms. Also, having a reduced dataset for tree-based algorithms does neither change the model efficiency nor improve the performance estimation. The decision tree and random forest use information gain and impurity measure that non-informative features will not be selected to split the data. Thus, the scaled and reduced data is not used for tree-based algorithms.

3.2. Model and Algorithm Selection

To select the best models for each algorithm, different parameters are evaluated by using Grid-search in the 5 by 2 nested CV. First, the lists of different hyperparameter values are defined in the hyperparameter grid. Thus, three-parameter grids are defined. Then, the parameter grids are passed down to the inner loop of 5 by 2 nested CV and grid search to choose the best hyperparameter setting for each algorithm. After comparing 5 average accuracies, the algorithm with 2 best generalization performance with selected hyperparameters is compared by 10-fold CV and the best model is chosen.

Setting Hyperparameter Grid It is necessary to compare the model with different hyperparameters to determine the best model for the kNN algorithm. The number of neighbors (k) between 3 to 15 are measured to inspect in a kNN model. The distance metrics that determine the validity of a neighbor are measured, which are Manhattan

(Equation 4) and euclidean (Equation 5) distance [15]. Furthermore, different algorithms that are used to compute the nearest neighbors of kNN models are considered, which are 'auto', 'kd tree', and 'ball tree'. They are considered to improve the computational performance of the kNN model by having different data structures instead of comparing every point in the data.

$$Manhattan = \sum_{j=1}^{m} |x_j^{[a]} - x_j^{[b]}| \quad (4)$$

$$Euclidean = \sqrt{\sum_{j=1}^{m} (x_j^{[a]} - x_j^{[b]})^2} \quad (5)$$

where m = the number of features, $x_j^{[a]}$, $x_j^{[b]}$ = feature vectors

Since the dataset has six predictor variables, it is necessary to consider a splitting criterion to optimize the decision tree; it is important to decide on splitting options to avoid the errors that would result from a few observations in a node that would eventually lead to a lack of statistical significance. For the splitting options, Shannon Entropy (Equation 6) and Gini impurity (Equation 7) are compared as the splitting criterion that measures the amount of information gain (Equation 8) of each feature [16].

$$Entropy\ H(p) = -\sum_{i} p_i \log_2(p_i) \quad (6)$$

$$Gini = 1 - \sum_{i} (p(i|x_j)^2) \quad (7)$$

$$Gain(D, x_j) = H(D) - \sum_{\nu \in Values(x_j)} \frac{|D_\nu|}{|D|} H(D_\nu) \quad (8)$$

where p_i = the probability of i-th class label, D_v = dataset at a child node, D = dataset at a parent node

The maximum depth of the tree is also considered, ranging from 1 to 20 and none; none means to split until all leaves are pure. Furthermore, the minimum number of samples required to split a node is considered from 2 to 4. The maximum depth of the tree and the minimum number of samples to split a node are considered to decide when it is optimal to stop growing a tree. When the tree is too deep, it may result in overfitting while a large minimum number of samples to split may lead to underfitting. Hence, different numbers are utilized as a criterion to estimate the performance of the decision tree corresponding to each of the hyperparameters.

Although the random forest algorithm is known to perform well without hyperparameter tuning, it may perform differently depending on the datasets. Thus, a few parameters are examined. Similar to the decision tree, the maximum number of depth and minimum number of splits is evaluated, but with a smaller number of values due to computational complexity.

5 x 2 Nested Cross Validation First, 5 by 2 nested CV is implemented to select the best models for each algorithm. In the nested CV, the training fold is split again to perform 2 CVs for each round, which corresponds to the inner loop. In this two-step procedure, the hyperparameters for each algorithm are evaluated through a grid search. Then, the model with the best hyperparameter setting is fit on the training set, which is one of the five bigger folds and they are evaluated on the validation fold again. Then, this gives 5 performance estimations with hyperparameter settings for each algorithm. Among the five performance estimates and hyperparameter settings, the best model of each algorithm is chosen manually by looking at the repeating settings. The three algorithms are compared by examining the average of the five estimations for each algorithm. By using 5 by 2 nested CV, the best model for each algorithm is selected, and the three algorithms are compared by using the average accuracies of the five performance estimates.

10 - Fold Cross validation After comparing three algorithms through the nested CV, the two best models corresponding to the two algorithms are compared by using 10-fold CV with grid search. 10-fold CV used in this process is similar to the inner loop of the nested CV, in that the hyperparameter grids are passed down and evaluated 10 times through grid search. This process provides the model with the highest performance estimate. By using 10-fold CV, the best algorithm resulting from the nested CV is verified and the best model returned by 10-fold CV is used.

3.3. Evaluation

The best model obtained by performing 5 by 2 nested CV and 10-fold CV is evaluated in three different ways. First, the 0.632 bootstrap method is used by fitting the model on the training set, which returns average bootstrap scores and confidence intervals. Then, the model is once again fitted on the training set and evaluated on the test set. Lastly, the whole dataset is used for evaluation through CV with stratified K-fold CV.

0.632 bootstrap The bootstrap method is a resampling technique for estimating a sampling distribution. By using the bootstrap method, the uncertainty of performance estimation for the final model is estimated. Specifically, the idea of the bootstrap method is to generate new data from a population by repeated sampling from the original dataset with replacement. The samples in the original sample that are not included in the bootstrap sample are used as a validation set, which is for the evaluation. 0.632 bootstrap method contains '0.632' in its name because bootstrap samples only contain approximately 63.2% of the unique samples. However, the 0.632 method Equation 9 addresses the pessimistic bias of the Out-Of-Bag method by combining pessimistic and optimistic estimates to get more realistic estimates of neither being overly pessimistic nor being overly optimistic. This method provides the average of the bootstrap accuracies and the confidence interval.

$$ACC_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632 \cdot ACC_{h,i} + 0.368 \cdot ACC_{r,i}), \quad (9)$$

[17] where $ACC_{r,i}$ = resubstitution accuracy and $ACC_{h,i}$ = accuracy on the out-of-bag sample

The model is fitted on the training set and evaluated on the test set. The prediction accuracy is used for evaluation. Furthermore, the stratified k-fold CV for 10 folds is used to evaluate the chosen model. Stratified k-fold cross validation shuffles the original data before splitting. Because 10 folds are used, the data will be splitted into 10, in which the test folds do not overlap. By using the stratified 10-fold CV, the folds preserve the proportions of the classes in the sample.

3.4. Using different dataset

Although the best model and algorithm for the given task is chosen, the models selected for each algorithm are compared with the Frame 2 dataset in order to see the models' performance with another dataset, expecting them to work sufficiently well. For this purpose, The three best models chosen for the algorithms used in this project are applied to the Frame 2 dataset. The test accuracy is computed by fitting the best models chosen from the nested CV on the training set. The evaluation for the new dataset consists of two parts: the stratified k-fold CV with 10 splits and the test accuracies. Finally, the results from the two datasets are compared.

4. Experiments

4.1. Dataset

The original dataset has 50,000 examples labeled either with "failure" or "no failure" according to the value of λ. Six feature variables consist of the uncertainties including yield strength (F_y), elastic modulus (E), sway imperfection (ψ), residual stress (X), dead load (D), and live load (L). For each feature variable, 50,000 random samples were generated by using Monte Carlo simulation based on statistical information summarized in Table 1. Samples are normalized by dividing nominal values into features. Distributions of random features are shown in Figure 3. The two frames utilized the same random samples in the simulation, except dead load and live load. A few examples that contain feature variables in a negative value, which is not reasonable in practice and results in simulation convergence issues, are excluded from the dataset. Therefore, the number of examples is 49,443 and 49,329 for Frame 1 and Frame 2, respectively. Finite element analyses were conducted in OpenSees [11], which is the computational platform for building finite element applications in structural systems, to obtain the ultimate strength ratio (λ) and thereby classify the class label as shown in Figure 4. The number of Class 0 and Class 1 for Frame 1 is 2,424 and 47,019, respectively. The Frame 2 dataset has 7,351 Class 0 and 41,978 Class 2.

Table 1. Description of feature variables

Variable	Mean	COV	Distribution	References
F_y	$1.1F_{yn}$	0.06	Lognormal	Bartlett et al. [2]
E	E_n	0.04	Lognormal	Bartlett et al. [2]
ψ	1/770	0.875	Lognormal	Lindner and Gietzelt [9]
X	1.064	0.27	Normal	Shayan et al. [19]
D	$1.05D_n$	0.1	Normal	Ellingwood et al. [5]
L	L_n	0.25	Extreme Value	Ellingwood et al. [5]

Nominal values: F_{yn} = 248 MPa, E_n = 200 GPa, D_{n1} = 31.1 kN/m, L_{n1} = 46.6 kN/m, D_{n2} = 30.4 kN/m, L_{n2} = 45.6 kN/m

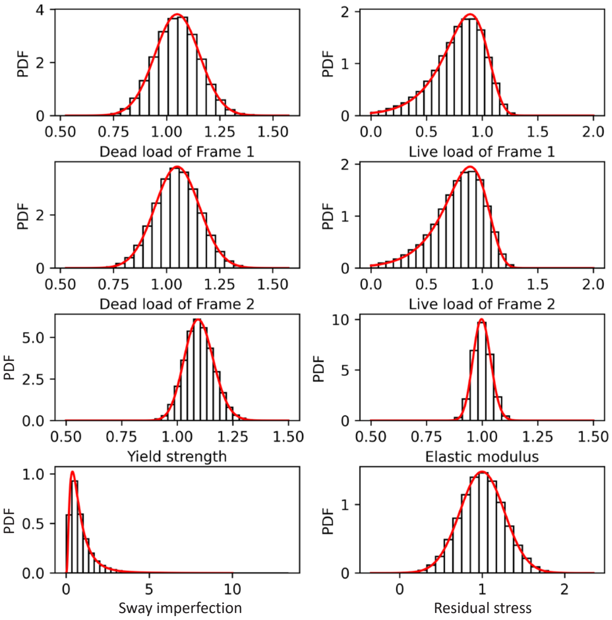

Figure 3. Distributions of feature variables

4.2. Software

Finite element analyses were conducted in *OpenSees* [11], and Monte Carlo sampling method was utilized by *MATLAB* [10]. *Python* 3 [18] is used through *Jupyter Notebook* [8] for the general procedure to perform the

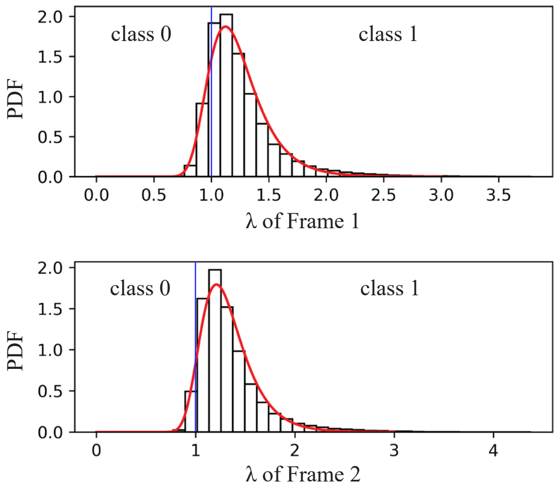

Figure 4. Distributions of λ

given task. It is used to enforce specific libraries and packages for feature selection, model and algorithm selection, and implementation of the three algorithms.

4.3. Hardware

Since more than one minute is taken for running one structural analysis, High Throughput Computing resources from the Center for High Throughput Computing (CHTC) of the university were utilized to obtain the two datasets which consist of 50,000 examples each. The computer hardware for running Python is going to be each member's labtop such as a Macbook pro.

5. Results and Discussion

5.1. Feature Selection

After splitting the data from Frame 1 to training and test set, z-score standardization (Equation 3) and feature selection are carried through before implementing model and algorithm selection, expecting for improved efficiency of the kNN algorithm. Through z-score standardization, the dataset is scaled with a mean of zero and a variance of one that the data is centered.

Instead of directly implementing PCA on the dataset, four cases of the data preprocessing steps for kNN are taken into consideration; using scaled or unscaled dataset and dataset with a reduced number of features or original features are taken into consideration. It is checked that the case with using scaled data and six feature numbers result in the highest prediction accuracy of 97.18%, but the downside is that it is computationally inefficient, which took four times longer than the other cases. The unscaled data and three features from PCA showed 93.09% prediction accuracy and the unscaled data using six features resulted in 97.14%. Through comparing the four cases, it is found that the condition that results in the most accurate performance estimation would be using scaled data with six features.

To increase computational efficiency, the dimensionality reduction is performed through the use of PCA and SFS. By applying PCA on the scaled data, three principal components (PC 1: 56.8% + PC 2: 21.4% + PC 3: 16.5%) explain 94.7% of the whole dataset, which exceeds the target accuracy of 93%. With the reduced size of the data through PCA, computational efficiency is significantly increased. Thus, it is concluded to use three of the six feature values. SFS is implemented to examine which features are most significant. By setting the number of features as three decided through the use of PCA, SFS informed that the best three features are indices 0, 1, and 5 among the features, which correspond to dead load, live load, and residual stress. By having 95.2% accuracy with the three feature values, it is concluded to use the three features for the kNN model.

5.2. Model selection & Algorithm selection result

For the model and algorithm selection, 5 by 2 nested CV and 10 fold CV are implemented. To select models corresponding to each algorithm, grid-search in the inner loop of nested CV is utilized to compare all of the possible combinations of the parameters, which are specified in the parameter grids. With the five best parameter settings with performance estimates for each algorithm, the best hyperparameter setting is chosen based on the repeating parameters instead of choosing the model with higher accuracy. By comparing the five cases for each algorithm, three models in total are selected with the best settings. The best settings for kNN, decision tree, and random forests are {'algorithm':'auto','n_neighbors':15,'p':2}, {'criterion':'entropy','max_depth':15,'min_samples_split}, and {'max_depth':None,'min_samples_split':3}, respectively.

The average of the five performance estimates for each algorithm is used to compare the algorithms. The average of five performance estimates are as follows: 92.80% ± 0.28 for kNN, 95.19% ± 0.28 for the decision tree, and 97.43% ± 0.11 for the random forest. By looking at the average of the performance estimates, the random forest algorithm has the highest performance estimate around 97.43%. However, it is doubtful to choose the best model and algorithm at this moment by having an ideal performance estimate for the decision tree, which is around 95.19%. Both the decision tree and random forest models have high accuracies over the target accuracy, 93%.

10-Fold Cross Validation is implemented with Grid Search to compare the models of decision trees and random forests. The 10-Fold CV ensured that random forest is still the best algorithm with the best CV accuracy of 97.5%, while it is 95.4% for the decision tree. The best model for the random forest is thus chosen from the result of 10-fold CV with setting the maximum depth as None and minimum samples required for splitting as 2. This model is also chosen as an optimal model for the prediction of no failure or failure in steel frames.

5.3. Evaluation of chosen model

The model chosen for the given task is evaluated in two different ways. First, the 0.632 bootstrap method is utilized in the training set to evaluate the chosen model. A bootstrap score is 98.46% with 95% confidence interval between 98.32% and 98.60%. Next, the chosen model is evaluated by evaluating the test set after fitting the model on the training set. The performance estimate of the training set is 100% and the test set accuracy is 97.77%, which shows that the model would overfit. However, more confidence in the model is gained by a narrow confidence interval. Both evaluation methods demonstrated performance estimates that exceed the goal accuracy for the given task. As a notion of reconfirmation, the model is fitted on the whole dataset with 10 CV, which demonstrates 97.76% of the performance estimate.

5.4. Evaluation of other models

Despite the best model and algorithm for the given task is already chosen and evaluated, the other models that are selected as the best models for corresponding algorithms are evaluated since all models had performance estimates that are higher than 90%. The nested CV provided that the best model for kNN has the setting of {'algorithm':'auto','n_neighbors':15,'p':2} and 93.26% test accuracy.

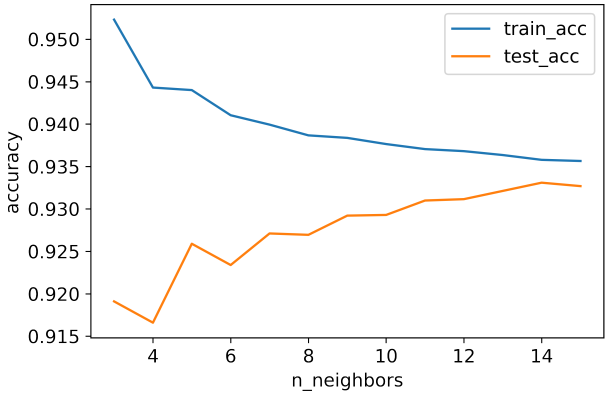

Figure 5. Accuracy of kNN

To check whether the model overfits or underfits, the training and test accuracies were compared based on the number of k as shown in Figure 5. Although the least difference between training and test accuracy occurred when n_neighbors (k) equals 14, the difference of accuracies when k = 14 and k = 15 from the best model are in the decimal unit. Therefore, the best model obtained for kNN is a complete model for the given task even though it is not the optimal choice.

The best model for decision tree selected through nested CV is with the setting of 'criterion': 'entropy', 'max_depth': 15, 'min_samples_split': 3 and it has 95.19% accuracy. When implementing the 10 fold CV to compare decision tree and random forest, the parameters were the same as the selected model from the nested CV has, except min_samples split being 4. The accuracy from 10 fold CV is 95.43%, which is slightly higher than the accuracy from the nested CV, which is 95.19%. The setting slightly differs, but both settings give prediction accuracy around 95%, which exceeds 93% of the goal accuracy score.

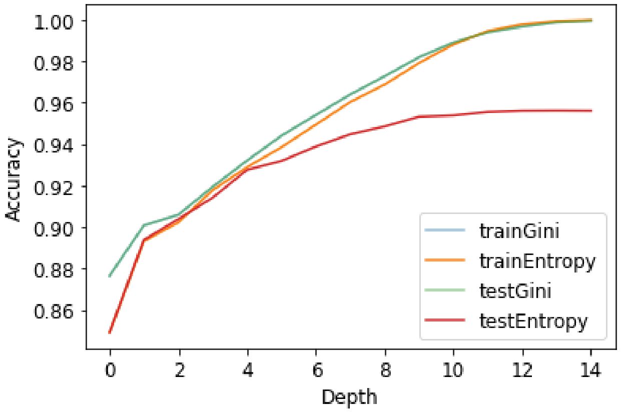

Figure 6. Gini vs. Entropy with the maximum depth

Figure 6 compares the performances of Gini and entropy in order to learn which index performs better in the model. The model is consistent with the parameters except for the two distance metrics. It is checked that Gini performed better than entropy when the maximum depth was small, but entropy performs slightly better as the depth increased. Also, the accuracy is the highest when the maximum depth is 15, regardless of the distance metric that corresponds to the chosen model for the decision tree. The accuracies of the training set and test set show almost the same and high, indicating that neither overfitting nor underfitting. As a result, it is concluded that the best model for decision tree from nested CV is still a good model even if it is not the best model chosen.

5.5. Models applied to the Frame 2 dataset

The best models chosen with the Frame 1 dataset are applied to the Frame 2 dataset to check if the models perform sufficiently well on different data. Before fitting the models, the feature selection was also performed on the new data for kNN. Then, the dimension of the data is reduced from 6 to 3 features by using PCA. Through performing SFS, three features are selected, which are dead load, live load, and yield strength. Through the use of SFS, it is checked that the three features account for approximately 98.62% of the total for the new data.

The three models obtained with the Frame 1 dataset are now applied to the Frame 2 dataset. The 10-fold CV accuracy for kNN, decision tree, and random forest corresponds to 97.98%, 98.13%, and 97.99%. The test set accuracies for the same models are 97.95%, 98.09%, and 98.84%, respectively. The random forest has the highest performance estimate and test accuracy. The test set accuracy confirms that the best model selected in the Frame 1 dataset is still the best model for the Frame 2 dataset. Also, all the three models exceed 97%, which is much higher than 93%, the target accuracy.

6. Conclusions

The main objective of this study was to examine the application of classification-based models for predicting structural failures in steel frames. kNN, decision tree, random forest were implemented with the datasets that consider the uncertainties such as material properties, and geometrical imperfections, and structural loads. The uncertainties should be included in structural analysis because they affect structural responses and even cause structural failures. The datasets used in this study obtained from the two frames with different structural failure modes. The developed models were applied to the two datasets to compare and analyze the results.

Based on a comprehensive analysis of the experimental results, the random forest algorithm showed the best results with the highest accuracy in the Frame 2 dataset, and all three models exceed 93% which is the target accuracy. For the weaknesses of the three models, for kNN, it was difficult to visualize in 3-dimensions. For random forest and decision tree methods, they showed higher accuracy than kNN, but interpretability was poor if the number of data points and variables is large. Overall, compared to the three models on the Frame 1 and Frame 2 datasets, accuracies of Frame 2 was higher than that of Frame 1 with values over 93%. As we expected, the random forest is the best algorithm for both datasets. Although the frames have different structural failure modes that would determine the number of failures (Class 0) in a dataset, the same models were selected as the best models for both frames. Therefore, we suggest using the random forest model with the setting chosen as the best model for the prediction of steel frames with different failure modes.

For future directions, the effects of class imbalance should be investigated. Since structures are usually designed not to be collapsed, the number of failures is fewer than the number of no failures. i.e., the occurrences of Class 0 and Class 1 have a huge difference that leads to class imbalance. We should be careful in using imbalanced data because machine learning methods are sensitive to class imbalance. Compensation methods for class imbalance can be used in the future study, such as oversampling, undersampling, and synthetic oversampling methods.

References

[1] ANSI/AISC, Chicago, IL. *Specification for Structural Steel Buildings*, 2016.

[2] F. M. Bartlett, R. J. Dexter, M. D. Graeser, J. J. Jelinek, B. J. Schmidt, and T. V. Galambos. Updating standard shape material properties database for design and reliability. *Engineering Journal-American Institute of Steel Construction*, 40(1):2–14, 2003.

[3] S. G. Buonopane. Strength and reliability of steel frames with random properties. *Journal of Structural Engineering*, 134(2):337–344, 2008.

[4] S. G. Buonopane and B. W. Schafer. Reliability of steel frames designed with advanced analysis. *Journal of Structural Engineering*, 132(2):267–276, 2006.

[5] B. Ellingwood, J. G. MacGregor, T. V. Galambos, and C. A. Cornell. Probability based load criteria: load factors and load combinations. *Journal of the Structural Division*, 108(5):978–997, 1982.

[6] S. Hwang, S. Mangalathu, J. Shin, and J. Jeon. Machine learning-based approaches for seismic demand and collapse of ductile reinforced concrete building frames. *Journal of Building Engineering*, page 101905, 2020.

[7] J. Kiani, C. Camp, and S. Pezeshk. On the application of machine learning techniques to derive seismic fragility curvess. *Computers and Structures*, pages 108–122, 2019.

[8] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing. Jupyter notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press, 2016.

[9] J. Lindner and R. Gietzelt. Imperfektionsannahmen für stützenschiefstellungen. *Der Stahlbau*, 52(4):97–101, 1984. [in German].

[10] MATLAB. *9.7.0.1190202 (R2019b)*. The MathWorks Inc., Natick, Massachusetts, 2018.

[11] S. Mazzoni, F. McKenna, M. Scott, and G. L. Fenves. *OpenSees command language manual*, 2007.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[13] S. Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *The Journal of Open Source Software*, 3(24), Apr. 2018.

[14] S. Raschka. Lecture 1: What is machine learning? *STAT451 Lecture Notes*, 2020.

[15] S. Raschka. Lecture 2: Nearest neighbor methods. *STAT451 Lecture Notes*, 2020.

[16] S. Raschka. Lecture 6: Decision trees. *STAT451 Lecture Notes*, 2020.

[17] S. Raschka. Lecture 9: Model evaluation 2: Confidence intervals and resampling techniques. *STAT451 Lecture Notes*, 2020.

[18] V. G. Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[19] S. Shayan, K. J. R. Rasmussen, and H. Zhang. Probabilistic modelling of residual stress in advanced analysis of steel structures. *Journal of Constructional Steel Research*, 101:407–414, 2014.

[20] H. Zhang, H. Liu, B. R. Ellingwood, and K. J. Rasmussen. System reliabilities of planar gravity steel frames designed by the inelastic method in aisc 360-10. *Journal of Structural Engineering*, 144(3):04018011, 2018.

[21] R. D. Ziemian. *Advanced methods of inelastic analysis in the limit states design of steel structures*. PhD thesis, Cornell University, Ithaca, NY, 1990.