17th World Conference on Earthquake Engineering, 17WCEE
Sendai, Japan - September 13th to 18th 2020

MACHINE LEARNING METHODS FOR PREDICTING SEISMIC RETROFIT COSTS

J.F. Fung(1), S. Sattar(2), D.T. Butry(3), S.L. McCabe(4)

(1) *Economist, National Institute of Standards and Technology, juan.fung@nist.gov*
(2) *Research Structural Engineer, National Institute of Standards and Technology, siamak.sattar@nist.gov*
(3) *Economist, National Institute of Standards and Technology, david.butry@nist.gov*
(4) *NEHRP Director, National Institute of Standards and Technology, steven.mccabe@nist.gov*

Abstract

Aging building clusters worldwide, especially in high seismic regions, will require a retrofit approach to improve the resilience of the built environment. One of the main challenges of retrofitting existing buildings is the associated cost. Fung et al. [1] develop a predictive modeling approach to estimating seismic retrofit costs. The predictive modeling approach uses historical data to predict retrofit costs for existing buildings based on observable building characteristics ("the features"). The advantages of this approach are that it is (1) cheap, (2) fast, and (3) can be applied to a single building or a large inventory of buildings. However, the approach relies on a linear model for prediction, which assumes a restrictive relationship between retrofit cost and the features. In this paper, we consider machine learning methods that can capture more complex, potentially nonlinear relationships between retrofit cost and the features. The paper considers ensemble methods (bagging and boosting) as well as neural networks and compares their performance to that of the linear model. The results show that a neural network provides the best performance in terms of prediction error. In applications, a user faces a tradeoff between accuracy and interpretability: while ensemble methods and deep neural networks may improve accuracy, they are not as easily interpretable as the linear model.

Keywords: seismic retrofit, risk reduction, resilience, retrofit cost estimation, building portfolio, prediction

© The 17th World Conference on Earthquake Engineering

1. Introduction

Aging building clusters around the world, especially in high-seismicity regions, will require a retrofit approach to improve the resilience of the built environment. One of the main challenges of retrofitting existing buildings is the associated cost. The reliable estimation of retrofit cost is valuable for decision makers, especially in the planning stage of a potential construction project. Machine learning methods have the potential to accurately predict retrofit cost for a single building and, more importantly, for a large inventory of buildings.

Fung et al. [1] present a predictive modeling approach that predicts retrofit cost based on observable building characteristics (e.g., building age and size) and the target performance objective. However, the approach relies on a linear model that essentially assumes a linear relationship between the predictors and retrofit cost (in particular, a *generalized linear model*, or GLM). In this paper, we consider machine learning methods that can capture more complex and possibly nonlinear relationships (in particular, ensemble methods and neural networks). Such methods are useful only if they can perform better than the linear model. However, the tradeoff is a lack of interpretability: for more complex methods, it is often difficult to understand how the model arrives at a prediction. The paper considers whether the performance gain is sufficient to warrant lack of interpretability.

The use of machine learning methods in the retrofit cost prediction literature is not new. Jafarzadeh et al. [2] use linear regression and Jafarzadeh et al. [3] use artificial neural networks. However, a comparison between methodologies is not made. Nasrazadani et al. [4] use Bayesian linear regression. However, the main objective is not to accurately predict cost but rather to quantify uncertainty in covariate effects (e.g., lateral strength). In the broader context of construction cost prediction, Elfaki et al. [5] provide a literature review of "intelligent methods," including machine learning methods.

The paper is organized as follows. Section 2 presents a discussion of the machine learning methods used in this paper. The main results comparing performance of machine learning methods is presented in Sec. 3. Finally, Sec. 4 presents concluding remarks and directions for future research.

2. Machine Learning Methods

This section describes the machine learning methods used in this paper to predict seismic retrofit costs. It is worth noting that the methods covered are not intended to be exhaustive. Rather, the methods are chosen in order to illustrate the tradeoff between model complexity and performance. A more complete treatment of machine learning is beyond the scope of this paper. An excellent reference is Mitchell [6].

At a high level, machine learning is a set of methods that finds patterns in data [7]. There are two main types of machine learning methods: supervised learning, which is *predictive*, and unsupervised learning, which is *descriptive* [7]. The key difference is that supervised learning has access to an outcome of interest (e.g., retrofit cost) and the goal is to predict the outcome from the input data (e.g., building characteristics) [8]. In contrast, there is no explicit outcome of interest under unsupervised learning and the goal is to explore patterns in the input data (e.g., clusters or latent features) [8].

This paper deals with a supervised learning problem. In particular, retrofit cost prediction is a *regression* problem because the outcome of interest, retrofit cost, is real-valued. If the outcome of interest is categorical (e.g., high cost vs low cost), the supervised learning problem is *classification*. In supervised learning, the model learns about the relationship between the input data (or features) and the outcome of interest. See Hastie et al. [8] for a treatment of supervised learning problems.

More precisely, suppose you have a set of features X (the covariates, or explanatory variables) associated with an outcome Y (the dependent, or response, variable). There is some relationship between X and Y, say $Y = f(X)$, and you would like to "learn" that relationship in order to predict new values of the outcome when presented with new instances of the features, say, $Y^*_{new} = f(X_{new})$.

© The 17th World Conference on Earthquake Engineering

In practical terms, to learn the function f means to fit a model (e.g., a GLM or a neural network) to data. How well the model captures the relationship between features and outcome depends on two key factors. The first is that the "training data" used to fit the model, $\{(X_i, Y_i)\}_{i=1}^{n}$, is sufficiently rich to credibly estimate the parameters of the underlying model. This means that a large number of examples n is not sufficient; the quality of the training data also matters. The second is the model itself, which may require "tuning" (i.e., the choice of hyperparameters such as the number of layers in a neural network). These issues, which are beyond the scope of this paper, are covered extensively in Hastie et al. [8].

In the context of cost prediction, X includes building characteristics (such as building age and size), and retrofit characteristics (in particular, the performance objective). The outcome Y may represent the actual cost from a retrofit or the estimated cost (e.g., a design or bid estimate) for a potential retrofit. As in Fung et al. [1], this paper uses data collected for FEMA 156 [9]. The training data is discussed in more detail in Sec. 3 below.

To determine how well a model captures the relationship between features and outcome, we need to quantify the quality of the predictions the model produces. This requires the choice of a loss function, $L(Y, f(X))$, that penalizes errors in prediction [8]. In other words, the criterion for choosing f should be to *minimize prediction error*. In regression problems, the most common and convenient way to quantify prediction error is squared error loss (or mean squared error):

$$L(Y, f(X)) = E[(Y - f(X))^2] \tag{1}$$

In this paper, we use the square root of Eq. (1), the *root mean squared error* (RMSE). The criterion is used to estimate the parameters of the underlying model (e.g., the parameters in a GLM may be estimated by the Newton-Rhapson method that minimizes Eq. (1)). The "trained" model is the model with parameters chosen to minimize RMSE, which is estimated by computing the empirical loss function.

Once the model is trained, it should be validated to estimate how good it is at making predictions. The goal is to estimate how the model will perform (in terms of prediction error) on new data examples. A model may perform very well on the training data while performing very poorly on new data, a problem called overfitting [7]. For instance, we train a GLM on FEMA 156 data so that we can predict retrofit cost for a building that we are interested in retrofitting (in other words, a building that is not in the training data). Validation is the key step to assessing generalizability of a model. Performance is measured by estimating RMSE on data that is *not used to train the model*. Otherwise, RMSE estimates will be biased because the model already knows the data! When the training data is not very large, as in our case, cross-validation can provide unbiased estimates of RMSE [8].

In the following subsections, we describe the particular machine learning methods we use in this paper, including the least complex method: the linear regression model.

2.1 (Generalized) Linear Model

In classical linear regression, the outcome is typically assumed to follow a normal distribution. The generalized linear model (GLM) extends the classical linear regression model by allowing the outcome to follow any distribution in the exponential family, such as the normal, binomial, Poisson, and gamma distributions [10]. Depending on the outcome distribution, the GLM can be used for regression or classification problems (for instance, the binomial distribution yields logistic regression, a widely used statistical model for binary classification).

As the name suggests, the GLM is essentially linear. While the GLM is not strictly linear, it nevertheless imposes a restrictive structure on the relationship between features and outcome. One might wonder if this restriction somehow hurts performance, in terms of high prediction error. On the other hand, as discussed in Fung et al. [1], the GLM has several strengths: relative to a more complex model such as a neural network, it is both easier to train and easier to interpret. The latter is particularly important if the ultimate objective is not just to predict Y but to understand what features are driving the prediction.

© The 17th World Conference on Earthquake Engineering

2.2 Ensemble Methods: Bagging and Boosting Trees

Tree-based methods, such as Classification and Regression Trees (CART) are a very different type of model to the GLM. At a high level, a decision tree partitions the feature space into different segments and uses the mean value of the outcome corresponding to a chosen segment. For instance, if X represents building age, we partition X into two segments, say $X \leq 50$ and $X > 50$. Given X_{new}, we predict Y_{new} based on the mean value of Y in the segment that X_{new} belongs to. If X represents multiple features, say building age and building historic status, then partitioning is recursive as shown in Fig. 1 (hence the tree structure) [11].

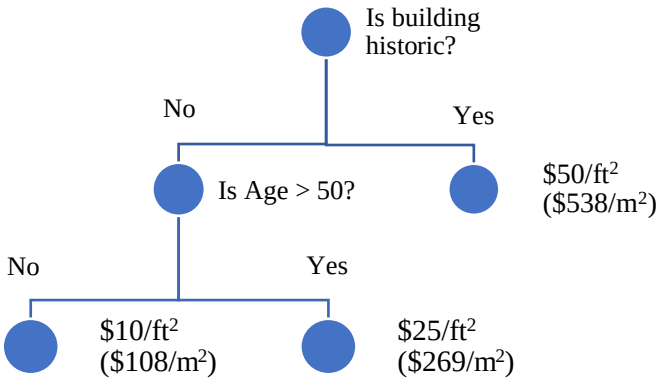

Fig. 1 – Example of a regression tree with two features: historic status of building and age of building. The terminal nodes represent predictions based on the mean retrofit cost in that branch of the tree. For instance, for a non-historic building that is over 50 years old, the predicted cost is $25 per square foot.

Representation of recursive partitioning as a tree makes these models interpretable, since they reflect the steps of a decision-making process. Moreover, the structure has the potential to easily capture interactions between features as well as non-linear relationships between features and outcome. Unfortunately, the hierarchical structure comes with a significant disadvantage: trees often have high variance [8]. Intuitively, this is because trees are highly sensitive to the partitioning rule, since any errors in the partitioning will be propagated down the tree. As a result, prediction error from a single tree can be large.

One way around these limitations is to combine the predictions of many trees. Ensemble methods leverage multiple individual algorithms to obtain better performance than each algorithm on its own. Two of the most common ensemble methods are *bagging* and *boosting*.

Bagging, short for bootstrap aggregating, resamples the training data (with replacement) B times, fitting the model to each of the B models and averaging the predictions across the B models. This process reduces the variance from fitting a single model to the training data without increasing the bias [12]. While bagging was originally applied to trees, the method can be applied to create an ensemble of any model. Bagging trees is robust to noise, outliers, and avoids overfitting. In this paper, we use *random forests*, a bagging method that additionally uses random subsamples of the features in order to reduce correlation across trees [13].

Boosting, on the other hand, fits a model sequentially M times to the training data. At each step m, the training data is reweighted based on the model's performance in the previous step, with the goal of focusing learning on the training examples that the model is predicting poorly, and at the last step M the final prediction is a weighted average of the M predictions [14]. Boosting was originally proposed for classification but has since been extended to regression. In contrast to bagging, boosted trees are not robust to noise or outliers. In this paper, we use a *gradient boosted model* (GBM), a boosting method that addresses some of the weaknesses of boosting, namely speed and robustness [15].

© The 17th World Conference on Earthquake Engineering

While boosting superficially resembles bagging, its objective makes the ensemble procedure very different. Boosting iteratively improves the performance of a "weak" learner (i.e., learner with high prediction error) to create a "strong" learner that has lower prediction error than the weak learners. Bagging, on the other hand, is a special case of model averaging: each model is fit independently of the others.

As is immediately obvious, the performance improvements provided by bagging and boosting come at a cost: combining multiple trees means the final "model" loses its interpretability. The question we ask in this paper is whether the improvement in performance (ie, lower prediction error) justifies the loss of interpretability: in other words, not just if such models are better but rather how much better.

2.3 Neural Networks

Another set of models that has recently received a lot of attention is the neural network. Neural networks are not new, however. The foundational theory dates back to the 1940s, with a mathematical model of how the brain processes complex information through the use of many simple neurons (hence the name) [18]. What is relatively novel is that the combination of large amounts of data and computing power in the early 21st century has allowed such previously intractable algorithms to be used broadly.

Despite the name and original motivation, neural networks have not been successful for plausibly modeling biological systems. Instead, neural networks have found success in statistical pattern recognition. The quintessential neural network is the feedforward neural network, or the multi-layer perceptron (MLP), essentially a "function approximation machine" [19]. A feedforward network learns a mapping $Y = f(X)$ by passing the inputs X through multiple hidden layers of neurons. Each layer finds a new representation of the features that is fed to the next layer, as shown in Fig. 1. Thus, the function f that is learned is simply a composition of many simpler functions. The term "feedforward" refers to information going in one direction (from inputs to output). If the output is fed back into the model, the network is called *recurrent*. Feedforward networks are most suitable for structured, tabular data [18].

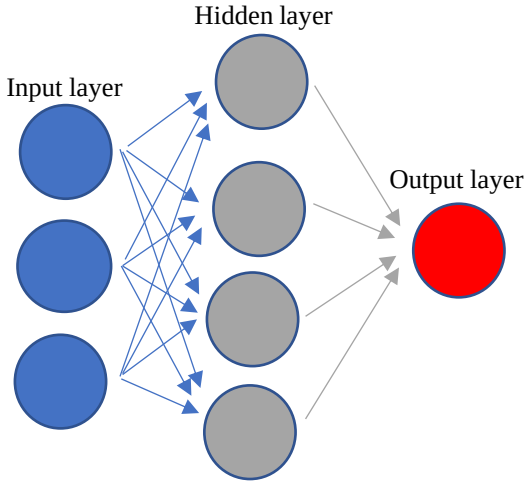

Fig. 2 – Example of a feedforward neural network with a single hidden layer. The input layer represents the features, which get passed on to the hidden layer. The hidden layer transforms the input features into a set of hidden features, which are passed on to the output layer to produce predictions. Note that the number of neurons in the hidden layer does not have to be the same as the number of features.

This simple idea belies the complexity of feedforward neural networks. The number of layers, including input layer, hidden layers, and output layer, defines the depth of the network. This is where the term deep learning originates. Moreover, the hidden layers can perform nonlinear transformations of the inputs and thus the network can learn a nonlinear and complex function f that is difficult to learn with a regression model.

© The 17th World Conference on Earthquake Engineering

Indeed, a neural network may be thought of as a nonlinear generalization of linear regression [8]; if the units in the hidden layers all perform an "identity" transformation, the entire model collapses to a linear regression model.

Estimation of the underlying parameters, however, requires solving a nonlinear, non-convex optimization problem. Fortunately, the gradients can be computed efficiently through *backpropagation* [18]. A thorough treatment of neural networks is far beyond the scope of this paper. For a deeper dive, see Goodfellow [19].

Like the ensemble methods discussed in Sec. 2.2, the complexity of a feedforward neural network makes interpretability (understanding how the algorithm produced a prediction) virtually impossible. Indeed, the impressive performance of neural networks has been criticized for coming at the expense of "black box learning." The interpretability problem has become a hot research topic in artificial intelligence (AI) more generally, as evidenced by the nascent field of Explainable AI; see Adadi and Berrada [20] for a survey.

3. The Training Data

In this section, we describe the training data used to predict seismic retrofit costs, originally collected for FEMA 156 [9]. The data is freely available online as part of FEMA's archived Seismic Rehabilitation Cost Estimator (SRCE) software (SRCE [16]). The data set includes retrofit cost and building characteristics for each of 1978 buildings in the United States and Canada. While the data set does not include primary building use, it nevertheless represents the most complete publicly available data set on seismic retrofit costs for North American buildings. The SRCE data is discussed in detail in Fung et al. [1].

In this paper, we use the features presented in Table 1. As shown in the Table, we focus on predicting *structural* retrofit cost (in particular, we predict the structural *unit* cost: structural cost per square foot). As we show in Fung et al. [1], predicting total retrofit cost is associated with much higher prediction error.

Table 1 – Description of the SRCE training data used in this paper

Variable	Definitions	Range of values
Structural retrofit cost	Cost to retrofit structural elements of a building, in millions of US dollars	[0.002, 157.161]
Area	Building floor area, in 1000ft^2 ($93m^2$)	[0.2, 1430.3]
Age	Building age, in years since construction	[2, 153]
Stories	Number of above and below ground stories	[1, 38]
Historic	Is building deemed historic?	Y, N
Occupancy	What happens to occupants during retrofit?	IP, TR, V
Seismicity	Site seismicity	L, M, H, VH
Performance objective	Retrofit target performance objective	LS, DC, IO
Building type	Building model type	See Table 2

Costs in the original SRCE data are normalized to average construction costs in California for 1993. Following Fung et al. [1], we normalize costs to US national average construction costs in 2016, using the Engineering News Record's Building Construction Index (BCI) [17]. It is worth noting that retrofit engineering

© The 17th World Conference on Earthquake Engineering

practice has evolved since the SRCE data was collected, likely decreasing the rate of growth in retrofit costs relative to the growth in the material and labor costs represented in the BCI.

Historic is a binary variable denoting whether the building is deemed historic and, thus, requires special care to preserve historic elements of the building. Occupancy denotes whether occupants are left in-place (IP), temporarily relocated to another part of the building (TR), or completely vacated (V) during the retrofit. Seismicity is measured using peak ground acceleration (*pga*) with a 5 % chance of exceedance in 50 years and is grouped into four categories [1]. In principle, any measure that captures the variation in seismic exposure will work. The performance objectives in the SRCE data are Life Safety (LS), the most common target, Damage Control (DC), and Immediate Occupancy (IO). For definitions, see FEMA 156 [9].

Finally, Table 2 presents the 15 building types represented in the SRCE data. In the models, we group building types based on structural similarities [1]. Note that, by far, unreinforced masonry is the most highly represented building type (32 %), followed by concrete frame with infill walls (C3) and concrete shear wall (C2), each at about 18 %. More details on the SRCE data are given in Fung et al. [1].

Table 2 – Building groups, building types, and their fraction in the SRCE data

Group	Building type	Building type name	Fraction in data
1	URM	Unreinforced Masonry	31.85 %
2	W1	Wood Light Frame	3.28 %
	W2	Wood (Commercial or Industrial)	4.85 %
3	PC1	Precast Concrete Tilt Up Walls	3.34 %
	RM1	Reinforced Masonry with Metal or Wood Diaphragm	5.24 %
4	C1	Concrete Moment Frame	7.54 %
	C3	Concrete Frame with Infill Walls	18.22 %
5	S1	Steel Moment Frame	4.98 %
6	S2	Steel Braced Frame	2.29 %
	S3	Steel Light Frame	1.31 %
7	S5	Steel Frame with Infill Walls	7.86 %
8	C2	Concrete Shear Wall	17.96 %
	PC2	Precast Concrete Frame with Infill Walls	0.98 %
	RM2	Reinforced Masonry with Precast Concrete Diaphragm	0.85 %
	S4	Steel Frame with Concrete Walls	2.11 %

4. Results

This section presents the results of training five different models on the SRCE data. In particular, we consider a linear regression model, a generalized linear model (GLM), a bagging model (random forests), a boosting model (GBM), and a neural network model (MLP).

© The 17th World Conference on Earthquake Engineering

We train each of the five models on the SRCE data using K-fold cross-validation in order to estimate out-of-sample performance, with $K = 10$. K-fold cross-validation randomly splits the data into K mutually exclusive subsets, iteratively using each as a validation set while training the model on the remaining data. Out-of-sample prediction error is estimated by averaging RMSE across the K iterations. Hyperparameters (e.g., tree depth, number of hidden units) are chosen using random grid search. Table 3 presents out-of-sample prediction error estimates, as well as training error (that is, average RMSE from training a model), for each of the models. The results suggest that the neural network, GBM, and random forest outperform the linear regression models. In particular, the prediction error for the linear regression model is \$3/ft2 (\$33/m2) higher than for the neural network.

Table 3 – Out-of-sample prediction error estimates (validation error), with their standard deviation. Based on K-fold cross-validation with $K = 10$. The training error represents the minimization of the loss function (in this case, RMSE), used to estimate the model parameters. Values in 2016 USD per ft2 (m2).

Model	Validation error	Standard deviation	Training error
Neural network	34.99 (376.64)	9.16 (98.60)	28.66 (308.50)
GBM	35.07 (377.50)	9.67 (104.09)	13.48 (145.39)
Random forest	35.14 (378.26)	9.93 (106.89)	36.27 (390.42)
GLM	37.13 (399.68)	9.10 (97.95)	38.57 (415.18)
Linear regression	38.04 (409.47)	8.59 (92.47)	38.63 (415.82)

Does the \$3/ft2 (\$33/m2) reduction in prediction error justify the loss in interpretability? Ultimately, this is subjective and depends on the problem context and the extent of the retrofit in terms of square footage. For a building owner that only cares about obtaining the most accurate estimate, the tradeoff may be highly valuable. In particular, for a building owner facing a 10,000ft2 (929m2) project, the potential cost from larger error is \$30,000. In contrast, for a consultant or any other user that must be able to provide an explanation as to *why* the cost estimate is *y*, a GLM may be desirable as it provides transparency and interpretability as to what features are driving the cost estimate. In this case, a \$2/ft2 (\$21/m2) reduction in prediction error may not be so appealing, especially for a smaller retrofit project.

4.1 Model Details

For completeness, we now describe each of the model architectures. The GLM uses a gamma distribution for the outcome with a log link function, i.e., $\ln(E[Y|X]) = X\beta$ [1]. The neural network includes one hidden layer with 500 neurons, with a rectified linear unit (ReLU) activation function. (The activation function transforms the weighted sum of the input to the layer into an output for the next layer; the ReLU activation function is the standard activation function for MLPs as it typically achieves better performance than other functions due to its handling of the vanishing gradient problem. For more details, see Goodfellow et al. [19].) The number of hidden layers and neurons is chosen by random grid search.

The GBM and random forest models can best be described in terms of number of trees and tree depth (i.e., the number of splits an individual tree performs), which are hyperparameters chosen by random grid search. For the GBM, the number of trees (that is, the number of boosting iterations) is M=46, with a maximum depth of 14. This results in an average of 547 leaves (i.e., terminal or decision nodes), with the smallest tree having only 174 leaves. The random forest model fits 37 separate trees, with a maximum depth of 29. This results in an average of 690 leaves, with the smallest tree having 550 leaves.

Each of the five models employs a form of *regularization*, a technique used to avoid overfitting when training a model. For the linear regression model and the GLM, regularization takes the form of adding a

© The 17th World Conference on Earthquake Engineering

penalty term to the objective function. Broadly, regularization penalizes model complexity. The linear regression model in Table 3 uses a quadratic penalty (this is commonly called ridge regression) [8]. The GLM, on the other hand, employs a linear penalty term (known as lasso). For the neural network, we use *dropout* for regularization [21]: in the hidden layer, 50% of the neurons are randomly dropped before passing the inputs to the activation function.

Regularization for ensembles of trees, on the other hand, is to a certain extent "built in." For GBM, the most straightforward method is to limit the number of boosting iterations *M*. For random forests, limiting tree depth is the easiest method of regularization. It is worth noting that in the random forest model trained in Table 3, all 37 trees have depth 29. For more advanced regularization techniques, see Hastie et al. [8].

It is worth noting that the neural network in Table 3 is not very deep, as it only contains a single hidden layer. A deeper neural network, with three hidden layers (each with 200 neurons, ReLU activation functions, and 40% dropout) performs worse than the neural network in Table 3, with out-of-sample prediction error of 37.28. While it does outperform the GLM and linear regression model, these results demonstrate that a more complex model is not always better.

5. Conclusion

In this paper, we considered the relative performance of machine learning methods, including linear regression models, ensembles of decision trees, and neural networks, as well as the tradeoff between prediction error and interpretability, in the context of predicting seismic retrofit costs. The results suggest that the gain in performance from more complex machine learning methods such as neural networks may not be significant to justify the loss of interpretability when a cost estimate may require some level of transparency. If the objective is to obtain an accurate estimate, especially when the building is very large, a neural network with a single hidden layer can reduce prediction error by \$3/ft$_2$ (\$33/m$_2$). Perhaps encouragingly, these results suggest that nonlinear relationships between features and the outcome are not hugely important for predicting retrofit cost.

It is worth noting that the ensemble methods we use combine the predictions of multiple "weak" learners in order to create a "strong" learner with better performance. An alternative ensemble method called *stacking* combines "strong" learners to create a "super learner" [22, 23]. An example of a super learner would be to stack random forests, GBMs, and neural networks into a single learner. Of course, the super learner would be inscrutable in terms of interpretability since each of the individual learners lacks interpretability. Nevertheless, it would be interesting to compare such a super learner to the models we consider in this paper. This is left for future work.

6. Acknowledgements

We are grateful to Shane Crawford and Stanley Gilbert for comments. All errors are our own.

7. Copyrights

17WCEE-IAEE 2020 reserves the copyright for the published proceedings. Authors will have the right to use content of the published paper in part or in full for their own work. Authors who use previously published data and illustrations must acknowledge the source in the figure captions.

8. References

[1] Fung JF, Butry DT, Sattar S, McCabe SL (2020). A predictive modeling approach to estimating seismic retrofit costs. *Earthquake Spectra*, **36** (2) *Forthcoming*. DOI: 10.1177/8755293019891716.

[2] Jafarzadeh R, Wilkinson S, Gonzalez V, Ingham J, Amiri GG (2013a). Predicting seismic retrofit construction cost for buildings with framed structures using multilinear regression analysis. *Journal of Construction Engineering and Management*, **140** (3) 04013062.

© The 17th World Conference on Earthquake Engineering

[3] Jafarzadeh R, Ingham J, Wilkinson S, Gonzalez V, Aghakouchak A (2013b). Application of artificial neural network methodology for predicting seismic retrofit construction costs. *Journal of Construction Engineering and Management*, **140** (2) 04013044.

[4] Nasrazadani H, Mahsuli M, Talebiyan H, Kashani H (2017). *Probabilistic modeling framework for prediction of seismic retrofit cost of buildings*. *Journal of Construction Engineering and Management*, **143** (8) 04017055.

[5] Elfaki AO, Alatawi S, Abushandi E (2014). Using intelligent techniques in construction project cost estimation: 10-year survey. *Advances in Civil Engineering*, **2014** 1-11.

[6] Mitchell T (1997): *Machine Learning*. McGraw Hill, 1st edition.

[7] Murphy KP (2012): *Machine Learning: A Probabilistic Perspective*. The MIT Press, 1st edition.

[8] Hastie T, Tibshirani R, Friedman J (2009): *The Elements of Statistical Learning*. Springer, 2nd edition.

[9] FEMA (1994): Typical costs for seismic rehabilitation of existing buildings, Volume 1: Summary. *FEMA 156*, Federal Emergency Management Agency, Washington, DC, USA.

[10] Nedler JA, Wedderburn RWM (1972): Generalized Linear Models. *Journal of the Royal Statistical Society. Series A*, **135** (3) 370-384.

[11] Breiman L, Friedman J, Olshen R, Stone C (1984): *Classification and Regression Trees*. CRC Press.

[12] Breiman L (1996): Bagging predictors. *Machine Learning*, **24** (2) 123-140.

[13] Breiman L (2001): Random forests. *Machine Learning*, **45** (1) 5-32.

[14] Schapire RE (2003): The boosting approach to machine learning: An overview. *Nonlinear Estimation and Classification*. Springer.

[15] Friedman JH (2001): Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, **29** (5) 1189-1232.

[16] FEMA (2013-2014): SRCE: Seismic Rehabilitation Cost Estimator. https://www.fema.gov/media-library/assets/documents/30220. Accessed: 2016-10-15.

[17] ENR (2017): Engineering News Record: Historical Indices. https://www.enr.com/economics/historical_indices. Accessed: 2017-03-03.

[18] Bishop CM (2006): *Pattern recognition and machine learning*. Springer. https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/.

[19] Goodfellow I, Bengio Y, Courville A (2016): *Deep Learning*. MIT Press. http://deeplearningbook.org.

[20] Adadi A, Berrada M (2018): Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, **6** 52138-52160. DOI: 10.1109/ACCESS.2018.2870052.

[21] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014): Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, **15** 1929-1958. http://jmlr.org/papers/v15/srivastava14a.html.

[22] Breiman L (1996): Stacked regressions. *Machine Learning*, **24** (1) 49-64.

[23] van der Laan MJ, Polley EC, Hubbard AE (2007): Super learner. *Statistical Applications in Genetics and Molecular Biology*, 6(1) 1544-6115.

© The 17th World Conference on Earthquake Engineering