**Ensemble machine learning-based models for estimating the transfer length of strands in PSC beams**

Viet-Linh Tran1,2 and Jin-Kook Kim1\*

1 Department of Civil Engineering, Seoul National University of Science and Technology, 232 Gongneung-ro, Nowon-gu, Seoul 01811, Republic of Korea

2 Department of Civil Engineering, Vinh University, Vinh 461010, Vietnam

\* Corresponding author : [jinkook.kim@seoultech.ac.kr](mailto:jinkook.kim@seoultech.ac.kr).

This supplemental material briefly presents the overview of four single ML algorithms, including Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), K-nearest Neighbors (KNN), and Decision Tree (DT). They were implemented in the Scikit-learn package (Pedregosa et al., 2011).

*1. Support Vector Machine (SVM)*

SVM uses statistical learning theory to minimize both the empirical risk and the confidence interval, and achieve a good generalization capability (Vapnik, 1995). SVM is a highly efficient and robust algorithm for regression (Smola & Schölkopf, 2004) and classification (Lauer & Bloch, 2008) problems. The basic idea behind the SVM algorithm is to map the original data sets from the input space to a high- or infinite-dimensional feature space to simplify the problems. To minimize the model complexity and prediction error, SVM uses kernel tricks to build expert knowledge about a problem (Raghavendra. N & Deka, 2014).

*2. Multi-Layer Perceptron (MLP)*

MLP (Adeli, 2001; Rumelhart et al., 1986) is a particular class of deep neural network algorithms. The MPL structure consists of an input layer, hidden layer(s), and an output layer. The nodes in the layers are interconnected and have associated thresholds and weights. The training process involves assigning values to these weights. The nodes' weights are constantly updated to reduce the difference between the predicted and target values.

*3. K-nearest Neighbors (KNN)*

KNN (Aha et al., 1991) locates the nearest data points in the training set to the point where a target value is missing and applies the approximate value of the identified data sets to it. It has no assumptions about the data distribution. Thus, it is efficient for extensive amounts of training data.

*4. Decision Tree (DT)*

DT (C.J. Stone, 1984) develops a tree-like structure from the training data. This process comprises three main steps. At first, the training data from the root node is recursively partitioned into subsets or branches using the Gini or entropy index criterion. Then, tree pruning is adopted to handle the overfitting issue during tree construction. Finally, a smoothing operation compensates for the sharp discontinuities between adjacent linear models at the pruned tree leaves.

**Table S-1** Hyperparameters of single ML models

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **No.** | **Hyperparameters** | **Range** |
| SVM | 1 | C | (0.01,1.0) |
|  | 2 | gama | (0.01,1.0) |
|  | 3 | degree | (1,5) |
|  | 4 | epsilon | (0.01,1.0) |
|  | 5 | tol | (0.01,1.0) |
|  | 6 | kernel | {'linear', 'poly', 'rbf', 'sigmoid' } |
| MLP | 1 | activation | {'logistic', 'tanh', 'relu'} |
|  | 2 | alpha | (0.00001,1.0) |
|  | 3 | batch\_size | (1,100) |
|  | 4 | hidden\_layer\_sizes | (1,100) |
|  | 5 | learning\_rate | {'constant', 'invscaling', 'adaptive'} |
|  | 6 | momentum | (0.1,1.0) |
| KNN | 1 | algorithm | {'auto', 'balll\_tree', 'kd\_tree', 'brute'} |
|  | 2 | leaf\_size | (1,100) |
|  | 3 | n\_neighbors | (1,50) |
|  | 4 | p | (1,2) |
|  | 5 | weights | {'uniform', 'distance'} |
| DT | 1 | criterion | {'squared\_error', 'friedman\_mse', 'absolute\_error', 'poisson' } |
|  | 2 | max\_depth | (1,50) |
|  | 3 | min\_samples\_leaf | (1,10) |

**Table S-2** Optimal hyperparameters of single ML models

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Data set 1** |  | **Data set 2** |  |
| **Model** | **No.** |  |  |  |  |
| SVM | 1 | 1.0 | 1.0 | 1.0 | 1.0 |
|  | 2 | 1.0 | 1.0 | 0.03512 | 0.01 |
|  | 3 | 5 | 5 | 5 | 5 |
|  | 4 | 1.0 | 1.0 | 1.0 | 0.01 |
|  | 5 | 1.0 | 0.01 | 0.01 | 1.0 |
|  | 6 | 'linear' | 'linear' | 'linear' | 'linear' |
| MLP | 1 | 'tanh' | 'relu' | 'relu' | 'relu' |
|  | 2 | 0.00029 | 1.0 | 0.93438 | 1.0 |
|  | 3 | 1 | 1 | 1 | 1 |
|  | 4 | 98 | 95 | 100 | 100 |
|  | 5 | 'adaptive' | 'adaptive' | 'adaptive' | 'adaptive' |
|  | 6 | 1.0 | 1.0 | 0.90486 | 0.23062 |
| KNN | 1 | 'brute' | 'kd\_tree' | 'auto' | 'kd\_tree' |
|  | 2 | 100 | 97 | 100 | 1 |
|  | 3 | 48 | 3 | 6 | 7 |
|  | 4 | 2 | 2 | 1 | 1 |
|  | 5 | 'distance' | 'distance' | 'distance' | 'distance' |
| DT | 1 | 'friedman\_mse' | 'absolute\_error' | 'friedman\_mse' | 'absolute\_error' |
|  | 2 | 9 | 42 | 21 | 9 |
|  | 3 | 5 | 10 | 1 | 3 |

**References**

Adeli, H. (2001). Neural Networks in Civil Engineering: 1989–2000. *Computer-Aided Civil and Infrastructure Engineering*, *16*(2), 126–142. https://doi.org/10.1111/0885-9507.00219

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*(1), 37–66. https://doi.org/10.1007/bf00153759

C.J. Stone, R. A. O. L. B. J. F. (1984). Classification and regression trees Boca Raton. In *Fl: Chapman and Hall/CRC* (Chapman an).

Lauer, F., & Bloch, G. (2008). Incorporating prior knowledge in support vector machines for classification: A review. *Neurocomputing*, *71*(7–9), 1578–1594. https://doi.org/10.1016/j.neucom.2007.04.010

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://doi.org/https://doi.org/10.48550/arXiv.1201.0490

Raghavendra. N, S., & Deka, P. C. (2014). Support vector machine applications in the field of hydrology: A review. *Applied Soft Computing*, *19*, 372–386. https://doi.org/10.1016/j.asoc.2014.02.002

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, *14*(3), 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer New York. https://doi.org/10.1007/978-1-4757-2440-0