PLEA 2020 A CORUÑA
Planning Post Carbon Cities

Comparative Analysis of Machine Learning Models Optimized by Bayesian Algorithm for Indoor Daylight Distribution Prediction

LINHAI SHEN[1] YUNSONG HAN[2]

[1,2] School of Architecture, Harbin Institute of Technology, Harbin 150001, China
[1,2] Key Laboratory of Cold Region Urban and Rural Human Settlement Environment Science and Technology, Ministry of Industry and Information Technology, Harbin 150001, China

ABSTRACT: Daylight distribution evaluation is vital for daylight design. However, its application in the early design stage is limited due to the time-consuming simulation process. Many statistical models were proposed to reduce the prediction time, yet application of the machine learning model in daylight prediction was relatively rare and has very limited generalization capability. This paper aims to propose a new workflow for indoor daylight distribution prediction, and compare the performance of XGB, RF, SVR and MLP models with Bayesian optimization. The results showed the MLP based prediction model achieved best generalization performance for indoor daylight prediction, which reduced the simulation time to less than 1 second and maintained satisfactory accuracy.
KEYWORDS: Daylight Distribution Prediction, Artificial Neural Network, Bayesian Optimization, UDI, Machine Learning

1. INTRODUCTION

Annual daylight distribution metrics like UDI and DA, are widely used to evaluate the daylight design quality and indoor visual comfort. However, its application in the early design stage is limited due to long simulation time [1].

Machine Learning (ML) based models can be a viable solution for daylight prediction problem, which can make relatively accurate predictions in short time as it avoids the time-consuming raytracing process. Compared with raytracing-based daylight simulation tools, it's more suitable for early design stage as the rapid daylight evaluation can support the designers making design decision efficiently.

In the past few years, ML based methods such as Neural Network, Support Vector Machine, Random Forest, have been widely applied for building energy consumption prediction [2-20]. However, application of the such models in daylight prediction was relatively rare, and most of them focused on predicting real-time hourly data for built environment.

Existing annual daylighting performance prediction studies based on machine learning are still limited to the bottleneck of generalization ability. The related works are limited to several specific cases with fixed room size and orientation, without generic parametric space representation. Lorenz, C. L et.al used ANN to predict Daylight Autonomy for 28 rooms, all shared the same size but with different fenestration [2]. In the similar research, DA prediction model for rooms of varied size was developed [3]. The problem of such method is that each scenario was trained with different model. As a result, the prediction model failed to generalize in other cases and needs to be trained with corresponding dataset again with any other building configuration.

To overcome the generalization limitation and improve the prediction accuracy, we developed a parametric workflow to generate proper daylight distribution dataset and compared the performance of Multi-Layer Perceptron (MLP) with other popular machine learning algorithms such as Support Vector Regression (SVR), Random Forest (RF) and Gradient Boosting Tree (XGB) to support the selection of machine learning algorithms for daylight distribution prediction modelling. A daylight distribution prediction model with better generalization capability were proposed through systematic training data generation and model hyperparameter tuning with Bayesian optimization.

2. MACHINE LEARNING MODELS

In this section, we give a brief introduction to the MLP, XGB, RF and SVR machine learning models we used for comparison and the related works, as well as the Bayesian Optimization technique used for parametric study.

2.1. Multi-Layer Perceptron

Multi-Layer Perceptron is a kind of Neural Network that only consists of fully-connected layer. A typical MLP model has one input layer of m neurons, h hidden layers containing k neurons and one output layer with n neuron. Each layer calculates the

weighted sum of all inputs and applies a non-linear activation function. Many methods are available for network training, in this research we used the Broyden-Flatcher-Goldfarb-Shano (BFGS) algorithm to train the model, which works best on a small to medium size dataset. MLP has been intensively used in various building performance prediction task in the past decade. Kazanasmaz et al. applied single hidden layer ANN model to predict the hourly indoor illuminance values for an office building [4]. Biswas et al. used simple BPNN to predict the total energy consumption. The above works proved the reliability and accuracy of BPNN in the building energy prediction [5]. Zhou and Liu combined Principal Component Analysis with ANN and SVM and compared their performance on annual UDI classification problem, and shown ANN with outperformed SVM by large margin [6]. Sharifzadeh et al. used ANN along with SVR and Gaussian Process Regression to forecast wind and solar power and energy demand. The ANN performed well in all tasks and was superior at predicting energy demand [7]. Ahmad et al. compared ANN and RF in predicting hourly HVAC energy consumption of a hotel, and find that ANN performed marginally better than RF [8]. A detailed review about ANN in energy prediction can be referred in [9].

2.2. XGBoost

XGBoost (eXtreme Gradient Boost) algorithm was first proposed by Chen in 2016 as an improved version of gradient boosting tree with faster computing speed and better generalization performance [10]. By using boosting strategy, it is capable of fitting very complex function, but also prone to overfitting problem and hard to fine-tuning. XGB has been applied in many fields and achieved impressive success. Hadri et al. compared XGB and RF as well as mathematic method to explore the efficiency of different models. The result shown XGB not only outperformed other methods in terms of accuracy but also showed higher prediction efficiency [11]. Touzani et al. applied gradient boosting machine to develop building baseline energy consumption model using a dataset of 410 commercial buildings. Compared to linear regression and to random forest algorithm, the gradient boosting machine improved the R-squared and the CV(RMSE) in more than 80 percent [12]. Fan et al. made a comparative study for short-term building cooling load prediction considering XGB, SVR, DNN, ELN and the RF methods. The results shown the performance of RF is not as good as the above-mentioned four nonlinear methods [13].

2.3 Random Forest

Random Forest is a tree-based ensemble learning algorithm based on bagging strategy. It consists of many simple decision trees and uses different parts of data to train each individual tree in parallel, the result of these trees is weighted and combined to the final result. The RF algorithm overcomes the instability of each individual tree and thus has better generalization ability. Smarra, F et al. developed a RF based model predictive control system for heating system and found its performance comparable to a physics-based MPC controller [14]. Ahmad et al. compared the performance of ANN with random forest both in hourly energy consumption and daylight illuminance prediction for a classroom. The impact of two hyperparameters, max depth and max feature, were explored for RF. It's found that ANN performs better on energy consumption prediction but RF yield better result on daylight prediction task [15]. Wang et al. used RF and SVR model to predict hourly electricity consumption of two educational buildings and the results showed that showed better performance compared with SVR in building electricity consumption prediction [16].

2.4 Support Vector Regression

SVR is a classical machine learning algorithm based on Support Vector Machine algorithm. SVR has the flexibility to define error tolerence in prediction model and is more robust to outliers. By using kernel function, SVR can also be used for various nonlinear regression problem. Dong et al. applied SVR to monthly building electricity consumption prediction [17]. Li et al. compared the performance of SVR with BPNN, GRNN and RBFNN in predicting hourly cooling load for an office building and shown SVR and GRNN method had better prediction accuracy and generalization than the the other two prediction model [18]. Massana et al. compared ANN, SVR, and MLR in predicting short-term energy load for non-residential buildings. It showed SVR improved prediction accuracy than the other methods [19]. Fan et al. developed and compared SVM and XGBoost model for predicting global solar radiation and recommended XGBoost over SVM considering with the accuracy, stability and efficiency [20].

2.5 Bayesian Optimization

The selection of the most appropriate learning model and hyperparameters is difficult in ML-based prediction modelling, which depending on the specific problem to be predicted. Such model selection process requires rich experience with the relevant machine learning algorithms and adequate time for parameter fine-tuning. The concept of hyperparameter optimization was developed to alleviate the effort of fine-tuning, making such algorithm more accessible to non-expert user. By

iterating through all possible hyperparameter settings, one can find an optimal hyperparameter setting at the cost of very long training time. Bayesian optimization utilizes more sophisticated algorithm to minimize the trials required to find a near-optimal hyperparameter combination. Basically, it trains a surrogate probability model to predict the optimized model's performance based on performance of former hyperparameters, and then uses an acquisition function to determine subsequent sample point. It performs best for optimization problems that has less than 20 dimensions with continuous domains [21], and can find better hyperparameters faster than random search and grid search and can even outperformed manually tuning method by expert on certain dataset [22].

3. METHOD

In this section, we demonstrated the modelling method of machine learning based prediction model including training datasets generation and best model selection. Firstly, the parametric simulation model was developed to obtain large size of training samples. Afterwards, 4 algorithms were trained based on the generated training datasets and optimized with Bayesian optimization. Finally, the 4 prediction models were compared using Mean Square Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), R^2 metric.

3.1 Data Preparation

First, data acquisition and parametric simulation model was developed in grasshopper to generate large amounts of data for machine learning model training.

A room with rectangular plan and a single window located in Harbin was selected as the reference building. The parametric simulation model (Figure 1) was developed in Grasshopper, with variables listed in table 1. The model used Daysim as daylight simulation engine. whose simulation setting is detailed in table 2.

Table 1: Inputs for parametric model and prediction model

Input	Range*	Explanation
Width	4.0-8.0	Width of room(m)
Depth	4.0-8.0	Depth of room(m)
Height	2.8-4.2	Height of room(m)
WWR-X	0.4-0.9	window Width/room Width, centred
WWR-Y	0.4-0.8	window Height /room Height
WSR	0.2-0.4	windows sill Height / room Height
Orient	0-359	Angle to South direction in degree

*All inputs are normalized to 0-1 before training.

Table 2: Simulation parameters

ab	ad	as	ar	aa
5	1024	20	30	0.15
R_floor	R_wall	R_ceiing	T_window	
0.2	0.5	0.8	0.7	

**R_ refers to reflectance, T_ refers to transitivity*

The daylight sensor points were placed on the plane 0.8m above the floor. Each point was located at the centre of the 10x10 grid dividing the room equally, regardless of room size (Fig.2) as the number of outputs must remain the same for different samples to be utilized by any prediction model.

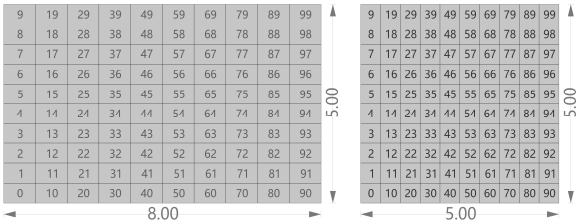

Figure 2: All rooms has same grid layout, regardless of their sizes.

The training data were created by sampling the parametric simulation model using Latin hypercube sampling method [23]. Latin hypercube sampling method was chosen because it can create a more representative sample space with fewer samples compared with the random sampling, which effectively reduce the total time required to acquire a representative training dataset. As the sampling method sampling each parameter independently, we filtered and discarded invalid Inputs where WWR-Y+WSR>0.9.

After obtaining the raw simulation data, we modified the data to make it better suited for machine learning task. Firstly, all inputs variables were remapped to [0, 1] with min-max scaling function. For tree-based model such as XGBoost and random forest, the result will not be influenced by the scale of input data, but for ANN and SVR this is mandatory. Secondly, the hourly illuminance value recorded at each test point were used to calculate 4 levels of UDI (Table.3) based on a working schedule from 9 a.m. to 5 p.m. Noting that rather than using averaged UDI, we explored more detailed daylight distribution information, which can capture the true spatial characteristic of daylight more accurately. This result in a large number of predicted targets for model, which is rare in previous literature. The intuition is to treat the room space as a whole, thus the relationship between all points in the space can be automatically learned by machine learning model.

Table 3: Simulation Results & Targets for prediction model

Target	Range*	Explanation
UDI100	3-37	UDI not achieved lx<=100
UDI300	0-85	Supplementary UDI where 100<lx<=300
UDI2k	0-92	UDI autonomous where 300<lx<2000
UDI2k+	0-92	UDI exceeded where lx>=2000

* All UDI are calculated based on workday, 9-17

1/5 of the samples were randomly chosen as testing data, others remained as training data. They

were processed individually in following step to avoid data leakage.

3.2 Model Training and Selection

After obtaining the dataset, 4 kinds of ML model were developed independently. As different machine learning model has varying hyperparameters that can drastically impact the training result, it's difficult to claim whose performance is better when only compared a certain one of these models to each other. With the help of Bayesian optimization, we can first identify the optimal hyperparameter for each model type, then compare the optimal model of each kind against each other to determine the best model.

All of these 4 models are implemented using "scikit-learn" library [24], and the name of each parameter corresponds to the name in library. The implementation of Bayesian optimization relies on the "scikit-optimize" python library.

Hyperparameters for different models and their range are listed in Table 4. These parameters are subset of all hyperparameter which are considered most impactful on performance. The upper and lower bounds of each parameter are chosen through preliminary trials and errors. These selected parameters were optimized sequentially by Bayesian optimization for 30 iterations. For each optimization literation of each model type, the whole training dataset was divided again into training and validation set using five-fold cross-validation method. The average MSE of five cross-validation model was used to rank the hyperparameter combination.

Table 4: Hyperparameters type and range for each model

MLP	SVR	GBT	RF
N_layers (1-4)	C (1-1000)	Max_depth (2-8)	Max_depth (2-20)
N_neurons (8-512)	Epsilon (0-1.0)	N_estimators (10-1000)	N_estimators (10-1000)
Alpha ($1e^{-5}$-0.1)	Gamma (0.01-10)	Gamma (0.01-10)	Min_samples_split (2-5)
		Learning_rate (0.001- 0.05)	Min_samples_leaf (1-5)

Finally, the optimized models were test on the reserved dataset to select the best model, using the following metric, MSE, MAE, MAPE, R^2, as formalised in Equations (1) to (4).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(predict_i - true_i)^2 \quad (1)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|(predict_i - true_i)| \quad (2)$$

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\left(\frac{predict_i - true_i}{true_i}\right)\right| \times 100\% \quad (3)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(predict_i - true_i)^2}{\sum_{i=1}^{N}(predict_i - \overline{true})^2} \quad (4)$$

4. RESULT AND DISCUSSION

As described in Section 3, we obtained 500 input samples and sent them to the parametric simulation pipeline. Then the simulation results were processed and divided into training set of size 400 and test set of 100, followed by the model development process where 4 ML models were trained and optimized.

MSE, MAE, MAPE and R^2 of 4 best models were shown in Figure 3. For training dataset, the performance rank is XGB>MLP>SVR>RF. XGB achieved lowest MSE (0.23) during training phase, marginally beating MLP (0.24 MSE). For test dataset though, MLP outperformed all other models by a large margin. Also, it showed much narrower gap between training and test performance, indicating good generalization capability.

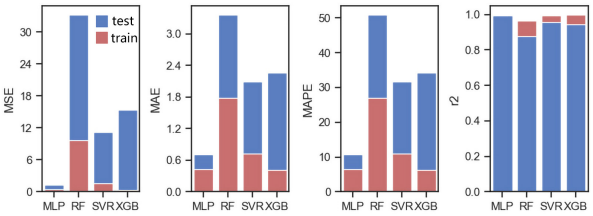

Figure 3: Performance on training and test data of 4 models

The difference of prediction models can be contributed to various reason:

1. Except for ANN and Random Forest, other two algorithm do not natively support the multi-output prediction problems. The multi-output was achieved by concatenating the result of individually trained model. Thus, the relationship between all output variables were not considered in these models, which could explain why they were outperformed by ANN.
2. In our experiment, the input features set are well defined as we already know which variable will impact the result, which is equivalent to manual featuring engineering. Thus, the featuring selection capability of each algorithm was not highly relevant. Also, the samples itself have contain few noises except for the inherent randomness in simulation engine.
3. While we replace the random search method with more advanced Bayesian optimization to reduce time cost for exhaustive search, the initial optimization boundary must be specified beforehand and chosen for this specific problem. Thus, different problem formation, such as using different weather data or parametric model, may require different parameter range, and possibly resulting in different performance. Also, the possible range of hyperparameter and the impact of certain parameter subsets were not discussed.

Although for this particular problem, other algorithms are unlikely to win against the MLP even with fine-tuning included judging from the training performance of each algorithm.

The optimal performance was achieved by a 3-hidden layer MLP with 496 neurons, converged within 10 minutes with $1e^{-5}$ learning rate. The MLP model generalizes on test dataset with R squared=0.98, MAE=0.76, MAPE=10.7% and MSE=1.27. In MAE distribution plot (Figure 4. Right), Most of errors (above 99%) are within a range of 5, and the error distribution have a centre very close to zero.

The learning curve (Figure 4. left) was created by testing model trained with different size training data on the test dataset. It indicated that the model has statured on the dataset of 400 samples, adding more data will not yield significant improvement.

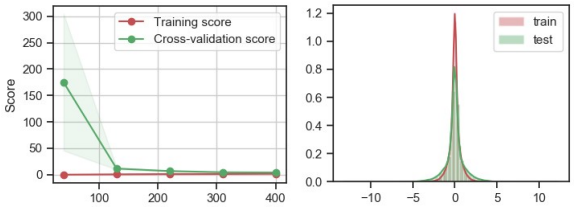

Figure 4: Learning curve (left) and error distribution (right) of MLP model

To take a closer look at the model performance, we created an error map of each individual value by averaging the mean absolute error of all samples. Figure 5. shown test MAE at each sensor points. Most of them have low error, except for some point near the windows. It may result from that the change of windows width will drastically influence the illuminance value near the window, causing them harder to predict.

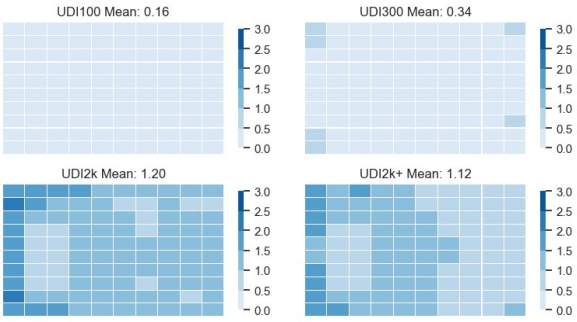

Figure 5. Test Mean Absolute Error for each sensor point

A case study of an office building was used to demonstrate the possible application (figure 6). The office plan consisted of simple rectangular office with one side opening, which can all be represented by the parametric prototype proposed in section 2. Even though the original parametric model is fairly simple, it can be matched and mixed to create various layout.

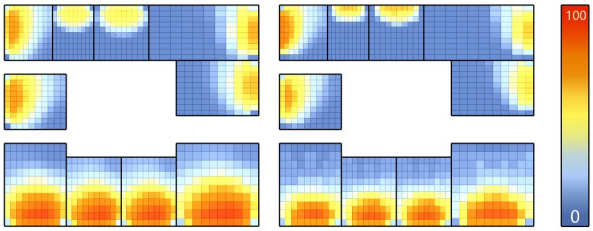

Figure 6: UDI2k+ of prediction results (left) and Daysim simulation results (right)

On change of any geometry, the MLP model can respond within 300ms (including data transmission time), while simulation by Daysim take about 10 minutes to recalculate on i7-4790 CPU with multiprocessing. As trade-offs, the result of the prediction model was not as accurate. The overexposure area is visibly larger which is corresponding to the error map, and the result seems more "averaged" than the actual case, which is caused by (1)the daylight simulation engine has inherent randomness which result in small noise in each sample, and the way statistical model was trained will average them. (2)the model learned to prefer continuous change from light to dark. While this may not suffice for a detailed light design, it accurately captures the distribution characteristic. In early design stage, this can serve as very fast and easy way to test the possible daylight performance.

5. CONCLUSION

In this paper, we proposed a machine learning based daylight prediction workflow, and benchmarked the performance of XGB, RF, SVR and MLP. Data simulated with Daysim from parametric study model were used as train and test dataset. The result shown that MLP achieved best generalization performance for daylight prediction models. The XGB model had very good performance on training dataset. But the generalization performance was way less ideal. We suggested that for similar problem with complex relationship between outputs, the ANN based models are better choice.

The proposed prediction model drastically reduces annually daylight prediction time under a second and maintains satisfactory accuracy, making daylight evaluation at design stage feasible. Compared with former researches, the output is the daylight distribution condition of the entire space instead of certain points, which is more intuitive for designer to understand. The predicted UDI can also be furthered processed to calculate Daylight Autonomy, weighted or averaged UDI to support design decision, or provided as extra input for various automatic indoor layout algorithm.

The main limitation of the research paper lies in that the urban context of studied room was not currently considered. Further researches can focus on

adding context inputs and extending the parametric model to cover more use case, such as room with multiple windows opening. The presented workflow to developed and compare different models can also be applied in other problem, such as yearly energy consumption.

For reference, the data, trained models and code will be available online at author's GitHub page [25].

ACKNOWLEDGEMENTS

This paper is funded by the National Natural Science Foundation of China (Grant No. 51708149), China Postdoctoral Science Foundation (Grant No. 2017M621276) and Heilongjiang Postdoctoral Science Foundation (Grant No. LBH-Z17076).

REFERENCES

1. Østergård, T., Jensen, R. L., & Maagaard, S. E. (2016). Building simulations supporting decision making in early design – A review. Renewable and Sustainable Energy Reviews, 61, 187–201.

2. Lorenz, C. L., Packianather, M., Spaeth, A. B., & De Souza, C. B. (2018, June). Artificial neural network-based modelling for daylight evaluations. In Proceedings of the Symposium on Simulation for Architecture and Urban Design (p. 2). Society for Computer Simulation International.

3. Lorenz, C. L., & Jabi, W. (2017). Predicting Daylight Autonomy Metrics Using Machine Learning. Sdbe: International Conference for Sustainable Design of the Built Environment. (SDBE), pp. 991-1002.

4. Kazanasmaz, T., Günaydin, M., & Binol, S. (2009). Artificial neural networks to predict daylight illuminance in office buildings. Building and Environment, 44(8), 1751–1757.

5. Biswas, M. A. R., Robinson, M. D., & Fumo, N. (2016). Prediction of residential building energy consumption: a neural network approach. Energy, 117, 84-92.

6. Zhou, S., & Liu, D. (2015). Prediction of Daylighting and Energy Performance Using Artificial Neural Network and Support Vector Machine. American Journal of Civil Engineering and Architecture, 3(3A), 1-8.

7. Sharifzadeh, M., Sikinioti-Lock, A., & Shah. (2019). Machine-learning methods for integrated renewable power generation: a comparative study of artificial neural networks, support vector regression, and gaussian process regression. Renewable & Sustainable Energy Reviews, 108(JUL.), 513-538.

8. Ahmad, M. W., Mourshed, M., & Rezgui, Y. (2017). Trees vs neurons: comparison between random forest and ann for high-resolution prediction of building energy consumption. Energy & Buildings, 147(jul.), 77-89.

9. Kumar, R., Aggarwal, R. K., & Sharma, J. D. (2013). Energy analysis of a building using artificial neural network: a review. Energy and Buildings, 65, 352-358.

10. Chen TQ, Guestrin C. XGBoost: a scalable tree boosting system. KDD 2016, San Francisco, CA, USA, August 13–17; 2016.

11. Hadri, S., Naitmalek, Y., Najib, M., Bakhouya, M., Fakhri, Y., & Elaroussi, M. (2019). A Comparative Study of Predictive Approaches for Load Forecasting in Smart Buildings. Procedia Computer Science, 160, 173–180.

12. Touzani, S., Granderson, J., & Fernandes, S., (2018). Gradient boosting machine for modeling the energy consumption of commercial buildings. Energy and Buildings, 158, 1533-1543.

13. Fan, C., Xiao, F., & Zhao, Y. (2017). A short-term building cooling load prediction method using deep learning algorithms. Applied Energy, 195, 222-233.

14. Smarra, F., Jain, A., de Rubeis, T., Ambrosini, D., D'Innocenzo, A., & Mangharam, R. (2018). Data-driven model predictive control using random forests for building energy optimization and climate control. Applied Energy, 226, 1252–1272.

15. Ahmad, M. W., Hippolyte, J. L., Mourshed, M., & Rezgui, Y. (2017). Random Forests and Artificial Neural Network for Predicting Daylight Illuminance and Energy Consumption. In Proceedings of the 15th IBPSA Conference (pp. 7-9).

16. Wang, Z., Wang, Y., Zeng, R., Srinivasan, R. S., & Ahrentzen, S., (2018). Random forest based hourly building energy prediction. Energy & Buildings, S0378778818311290.

17. Dong, B., Cao, C., & Lee, S. E. (2005). Applying support vector machines to predict building energy consumption in tropical region. Energy and Buildings, 37(5), 545–553.

18. Li, Q., Meng, Q., Cai, J., Yoshino, H., & Mochida, A. (2009). Predicting hourly cooling load in the building: a comparison of support vector machine and different artificial neural networks. Energy Conversion & Management, 50(1), p.90-96.

19. Massana, J., Pous, C., Burgas, L., Melendez, J., & Colomer, J., (2015). Short-term load forecasting in a non-residential building contrasting models and attributes. Energy and Buildings, 92, 322–330.

20. Fan, J., Wang, X., Wu, L., Zhou, H., Zhang, F., & Yu, X., et al. (2018). Comparison of support vector machine and extreme gradient boosting for predicting daily global solar radiation using temperature and precipitation in humid subtropical climates: a case study in china. Energy Conversion & Management, 164(MAY), 102-111.

21. Frazier, P. I. A tutorial on bayesian optimization. arXiv 2018. arXiv preprint arXiv:1807.02811.

22. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems (pp. 2951-2959).

23. Deutsch, J. L., & Deutsch, C. V. (2012). Latin hypercube sampling with multidimensional uniformity. Journal of Statistical Planning and Inference, 142(3), 763-772.Pedregosa,

24. F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

25. https://github.com/ForestoShen/PLEA2020-Indoor-Daylight-Distribution-Prediction