

Article

Prediction of Full-Load Electrical Power Output of Combined Cycle Power Plant Using a Super Learner Ensemble

Yujeong Song ¹, Jisu Park ², Myoung-Seok Suh ³  and Chansoo Kim ^{4,*}

¹ Data Service Division, Penta Systems Technology, 115 Yeouigongwon-ro, Seoul 07241, Republic of Korea; yjsong@penta.co.kr

² R&D Center, YOUNGJIN Technology Co., Ltd., 296 Sandan-ro, Ansan-si 15433, Republic of Korea; jspark@youngjintech.com

³ Department of Atmospheric Science, Kongju National University, 56 Gongjudaehak-ro, Gongju-si 32588, Republic of Korea; sms416@kongju.ac.kr

⁴ Department of Applied Mathematics, Kongju National University, 56 Gongjudaehak-ro, Gongju-si 32588, Republic of Korea

* Correspondence: chanskim@kongju.ac.kr

Abstract: Combined Cycle Power Plants (CCPPs) generate electrical power through gas turbines and use the exhaust heat from those turbines to power steam turbines, resulting in 50% more power output compared to traditional simple cycle power plants. Predicting the full-load electrical power output (P_E) of a CCPP is crucial for efficient operation and sustainable development. Previous studies have used machine learning models, such as the Bagging and Boosting models to predict P_E . In this study, we propose employing Super Learner (SL), an ensemble machine learning algorithm, to enhance the accuracy and robustness of predictions. SL utilizes cross-validation to estimate the performance of diverse machine learning models and generates an optimal weighted average based on their respective predictions. It may provide information on the relative contributions of each base learner to the overall prediction skill. For constructing the SL, we consider six individual and ensemble machine learning models as base learners and assess their performances compared to the SL. The dataset used in this study was collected over six years from an operational CCPP. It contains one output variable and four input variables: ambient temperature, atmospheric pressure, relative humidity, and vacuum. The results show that the Boosting algorithms significantly influence the performance of the SL in comparison to the other base learners. The SL outperforms the six individual and ensemble machine learning models used as base learners. It indicates that the SL improves the generalization performance of predictions by combining the predictions of various machine learning models.

Keywords: Combined Cycle Power Plants; electrical power output; machine learning; Super Learner; cross-validation; ensemble



Citation: Song, Y.; Park, J.; Suh, M.-S.; Kim, C. Prediction of Full-Load Electrical Power Output of Combined Cycle Power Plant Using a Super Learner Ensemble. *Appl. Sci.* **2024**, *14*, 11638. <https://doi.org/10.3390/app142411638>

Academic Editor: Gang Lei

Received: 14 October 2024

Revised: 26 November 2024

Accepted: 10 December 2024

Published: 12 December 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Gas turbines produce electrical power by utilizing the hot gases generated from combusting an air–fuel mixture. Although gas turbines are widely used, a significant amount of the fuel’s latent energy is wasted as thermal energy from hot exhaust gasses. To utilize this energy, Combined Cycle Power Plants (CCPPs) use the exhaust heat from the gas turbines to generate additional electricity through steam turbines. This approach increases the power plant output and boosts efficiency to more than 50%, which is substantially higher than that of the simple-cycle and even the supercritical pressure reheated steam power plants [1]. CCPPs also offer several additional features, including low environmental emissions; short construction times; fuel flexibility; and low initial investment, operation, and maintenance costs [2].

Accurately predicting the full-load electrical power output of a base-load power plant is crucial as it maximizes the income from the available megawatt hours, resulting in improved efficiency and economic operation of the plant [3]. Furthermore, integrating this prediction with a dynamic condition monitoring system allows for a real-time performance comparison between the power plant and the derived model [4]. Any deviations detected can be promptly diagnosed and inspected. This approach guarantees the plant's safety and reliability across diverse operating conditions.

Gas turbines are significantly influenced by ambient air compared to other conventional power generation systems, as they utilize ambient air directly as their working fluid [5]. Steam turbines are affected by exhaust steam pressure (or vacuum). Several studies have employed Artificial Neural Networks (ANNs) to predict the non-linear relationship between ambient conditions and the output of a CCPP. Roni and Khan [6] employed an ANN with a hidden layer consisting of 10 neurons to predict the power output of a CCPP. They analyzed the impact of prediction parameters on the power output and discovered that an increase in ambient temperature results in a decrease of approximately 14% in the power output. Zhao and Foong [7] showed that combining an electrostatic discharge algorithm with an ANN outperforms other ANN models. ANN models were applied to predict the output of various power plants [8–10]. Moreover, several studies have employed traditional machine learning algorithms to predict the electrical power of a CCPP. Tüfekci [3] showed that Bagging REP Tree (BREP), a type of ensemble learning, was the most successful method among some machine learning regression models evaluated. Janoušek et al. [11] utilized Random Forest to develop a model with lower errors and enhanced reliability in comparison to the methods proposed by [3]. Alketbi et al. [12] reported that Random Forest achieved the best performance compared to Linear Regression, Multilayer Perceptron, and K-Nearest Neighbors. Santarisi and Faouri [4] showed that the gradient boost regression model outperformed five regression models containing Random Forest. Previous studies have demonstrated that ensemble learning outperforms traditional single learners in predicting the power output of a CCPP. This study employs various machine learning algorithms and ensemble learning algorithms to predict the full-load electrical power output of a CCPP. Furthermore, we employ the Super Learner algorithm by combining them as base learners to enhance predictions compared to traditional single models.

Many popular machine learning methods are ensemble algorithms such as Bagging, Random Forest, and Gradient Boosting. Ensemble learning is a method that combines prediction skill from multiple learning algorithms to generate more reliable predictions than a single learning algorithm. Bagging, called Bootstrap aggregating, is a method that combines multiple weak learners trained by random sampling with replacement [13]. Boosting is an ensemble method that reduces bias by iteratively combining multiple weak learners while reweighting the data [14]. Misclassified examples gain weights, while correctly classified ones lose weights. The next weak learner improves training error by focusing on the misclassified data from the previous weak learner. Stacking is also called "Stacked Generalization" and "Super Learning". Unlike Bagging and Boosting, Stacking aims to combine a diverse set of strong learners. First, base learners of Stacking learn the original data called level-zero data. Wolpert [15] introduced a method that utilizes the output values of base learners as new input components called level-one data. The final prediction is generated by a meta-learner that learns from the level-one data as its new input [16]. Breiman [17], known for his work on Classification and Regression Trees (CART) and the creator of Random Forest algorithm, formalized Stacking. This approach proposed using k-fold cross-validation in Stacking and improved performance by giving constraints when calculating weights of the level-one data. Until then, the mathematical reasons for why Stacking worked well were unknown, and Stacking was considered a "black art". However, Van der Laan et al. [18] mathematically showed the theoretical background of Stacking; hence, the algorithm took on the name "Super Learner". They proved that the Super Learner ensemble represents an asymptotically optimal system for learning.

In this study, we compare and analyze various machine learning algorithms to predict the full-load electric power of a CCPP and aim to identify the best prediction model. Several machine learning algorithms used in this paper include individual and ensemble models such as Support Vector Regressor (SVR), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), Gradient Boosting (GBM), and eXtreme Gradient Boosting (XGB). Additionally, we use Super Learner (SL), which combines individual and ensemble models. The machine learning algorithms and SL used in this study predict the average hourly power output using average hourly variables such as ambient temperature, atmospheric pressure, relative humidity, and vacuum. Furthermore, by analyzing the coefficients derived through SL, we gain insights into the model's contributions to predicting power output. Our method provides a more robust framework for power output prediction, especially under varying operational conditions.

This paper is organized as follows. Section 2 provides an overview of the CCPP structure and the dataset employed in this study. In Section 3, the machine learning algorithms and SL used for predicting the average hourly electrical power of the CCPP over six years are described. Section 4 presents a comparative evaluation of the prediction performance of these models. Finally, Section 5 offers a concise conclusion.

2. Data

The dataset used in this study was obtained from the UCI Machine Learning repository. It comprises data from a CCPP, which consists of two gas turbines, two Heat Recovery Steam Generators (HRSGs), and one steam turbine. The gas turbines consist of a compressor, combustor, and turbine. The compressor combines compressed air with fuel to generate a high-temperature flow through the combustor. The high-temperature flow drives the turbines, generating electrical power. The HRSGs capture heat from the gas turbines' hot exhaust gasses to produce steam, which is then supplied to a steam turbine to generate additional electrical power. Consequently, electricity is generated by both the gas and steam turbines within the CCPP. According to Baheta and Gilani [19], the performance of the gas turbine is affected by anything that changes the density and/or mass flow of air intake to the compressor such as ambient temperature, pressure, and humidity. The exhaust steam pressure from the steam turbine is converted back into water through the condenser. The higher the vacuum inside the condenser, the lower the outlet pressure of the steam turbine, and more steam is moved to the condenser. Thus, this enables the steam turbine to generate maximum power output. However, abnormal vacuum pressure can hinder the condensation process, ultimately resulting in decreased power output [20]. Therefore, a vacuum level must be maintained in the condenser so that the generator can work optimally.

The dataset contains 9,568 data points collected over six years (2006–2011), specifically during the 674 days when the CCPP operated at full load. Input variables, namely ambient temperature (AT), atmospheric pressure (AP), relative humidity (RH), and vacuum (V), were calculated as an average per hour and obtained from various sensors located around the plant that record the ambient variables every second. We use these input variables to predict the average hourly full-load electrical power output (P_E) received from a control system when the power plant is at base load. All the input variables and target variable are listed in Table 1.

Table 1. The hourly average ambient input variables and target variable in the dataset.

Type	Variable	Range
Input	Ambient temperature (AT)	1.81–37.11 °C
	Atmospheric pressure (AP)	992.89–1033.3 mbar
	Relative humidity (RH)	25.56–100.16%
	Vacuum (V)	25.36–81.56 cm Hg
Target	Full-load electrical power output (P_E)	420.26–495.76 MW

Figure 1 shows correlation coefficients among the variables. Among the input variables, AT and V exhibit a strong positive correlation of 0.844, indicating a significant positive relationship between them. The target variable (P_E) displays a strong negative correlation with AT and V and a weak positive correlation with AP and RH. Nevertheless, it suggests that they might all affect the power output.

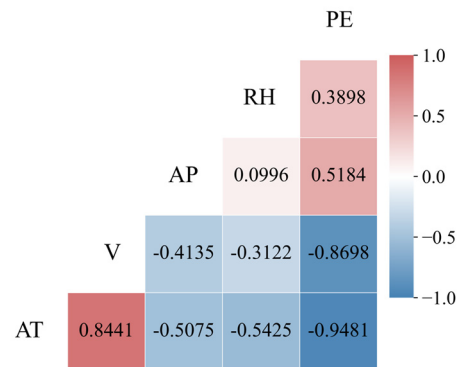


Figure 1. The correlation matrix showing the relationships between input and output variables.

3. Methods

A machine learning algorithm can estimate an unknown dependency between input variables and the output variable. In this study, six machine learning algorithms and a Super Learner ensemble learning method, which combines them, are employed to predict the hourly average full-load electrical power output of the CCPP. Here, we provide a summary of the learning methods.

3.1. Machine Learning Methods

3.1.1. Support Vector Regression (SVR)

SVR is based on the Support Vector Machine (SVM) algorithm proposed by [21] and is commonly applied to regression problems. Unlike other regression models that attempt to minimize the error on all the training sets, SVR relies only on a subset of the training data. The cost function in SVR disregards samples whose predictions are close to their targets.

Given a training dataset $\{(x_i, y_i), i = 1, \dots, N\}$, where $x_i \in R^M$ and $y_i \in R$, we construct a linear regression function $f(x_i)$:

$$f(x_i) = w^T x_i + b \quad (1)$$

where w is weight vector, and b is constant term.

The goal is to find a function $f(x_i)$ that has at most ϵ deviation with the target value y_i for all the training data and is as flat as possible. For flatness, it is required to minimize the Euclidean norm $\|w\|^2$. Formally, this can be represented by a convex optimization problem:

$$\begin{aligned} \min & \frac{1}{2} \|w\|^2 \\ \text{s.t. } & |y_i - f(x_i)| \leq \epsilon, \quad \forall i \end{aligned} \quad (2)$$

If not all training data satisfy (2), f may not be generated. For the slack variables ξ_i and ξ_i^* , some errors are allowed [22]. SVR solves non-linear regression problems by mapping feature space to a high-dimensional feature space through kernel functions such as a polynomial function or radial basis function (RBF). It uses a kernel trick without converting the data into higher dimensions. In this study, SVR uses the RBF kernel function.

3.1.2. K-Nearest Neighbors (KNN)

KNN is one of the simplest machine learning algorithms, yet it is very effective for many applications and often offers better performance than SVR [23,24]. KNN predicts a

target variable based on a similarity measure, such as the distance between training and test data.

Given $x = (x_1, x_2, \dots, x_m)$ and $y = (y_1, y_2, \dots, y_m)$ that are, respectively, training and test data, Euclidean distance and Manhattan distance are as depicted below.

Euclidean distance is commonly used to calculate the distance between the training and test data as follows:

$$d_{euclidean}(x, y) = \sqrt{\sum_{m=1}^M (x_m - y_m)^2} \quad (3)$$

Manhattan distance uses absolute values of differences between the training and test data as follows:

$$d_{manhattan}(x, y) = \sum_{m=1}^M |x_m - y_m| \quad (4)$$

The selection of the number of neighbors (k) is a critical step. If k is set too small, the model is sensitive to neighbor points and becomes overly complex. If k is set too large, the model becomes monotonous. Usually, the best value for k is calculated using cross-validation [25]. KNN Regressor predicts a test example to an average output value of k neighbor points. In this study, KNN Regressor uses the inverse of the distance weight, not the average weight.

3.1.3. Decision Tree (DT)

DT generates tree-based rules and can be used for classification and regression. The algorithms used for DT include ID3, C4.5, and CART. CART is a binary tree structure that uses the Gini index for classification problems and the mean squared error between actual and predicted values for regression problems as splitting criteria [26]. In the DT structure, a node is a branching point that leads to two sub-branches. The first node of the tree is called the root node, and the last nodes are called leaf nodes [27]. Except for the root node and the leaf nodes, the other nodes are decision nodes that contain decision rules. A regression tree predicts the test data as the average of the leaf node. DT is intuitive and does not require pre-processing tasks such as feature normalization, while it may generate a tree that does not generalize well to the data.

3.1.4. Random Forest (RF)

RF is an algorithm that combines independently trained DT models using Bootstrap samples of training data [28]. It achieves a relatively fast processing speed among ensemble algorithms. RF uses the average and majority class of predicted values of DT models for regression and classification tasks, respectively, when generating a final prediction. Since it combines the results of various DT models, it is more stable than a single DT and reduces overfitting problems. However, if there are strong correlations between DT models, RF may not have an advantage over a single decision tree. To enhance predictive capability, Breiman [28] proposed a technique of randomly selecting features at each node during the construction of decision trees in RF. The random feature selection introduces diversity among the DT models and effectively reduces variance. In our study, RF is trained by randomly selecting two features at each node.

3.1.5. Gradient Boosting (GBM)

GBM is a powerful prediction model that combines multiple DT models as weak learners [29]. GBM trains the DT models sequentially to reduce the residuals of the previous model, while RF trains them independently. This sequential process may lead to slower learning for GBM, but it also contributes to its high accuracy. Unlike AdaBoost [30], GBM uses gradient descent to update weights. The weak learners in GBM are constructed in a greedy manner by selecting the best split points based on purity scores like Gini index

to minimize loss functions. GBM generally uses a logarithmic loss in classification and a mean squared error loss in regression. To prevent overfitting, GBM constrains the weak learners in various ways, such as limiting the maximum number of trees, nodes, splits, or leaf nodes. Since base learners in GBM consist of trees, it can overfit the data. To combat this, GBM predicts by scaling the trees with the same weight.

3.1.6. eXtreme Gradient Boosting (XGB)

XGB is an algorithm that improves both speed and performance based on GBM [31]. It can solve real-world scale problems using minimal resources. In contrast to GBM, XGB offers built-in regularization techniques to control the complexity of the model and prevent overfitting. By incorporating L1 regularization and L2 regularization, it can adaptively shrink the weights of less important features and even remove irrelevant features altogether. XGB also supports internal cross-validation for each iteration to explore the number of optimized iterations, which supports early stopping. XGB can provide parallel tree boosting, which makes tree learning faster. Moreover, XGB uses a regularized loss function to prevent overfitting. Given sparse data included in many real-world problems, XGB deals with it by adding a default direction in each tree node to handle missing values.

3.2. Super Learner (SL)

SL is a prediction algorithm designed to find the optimal combination of base learners [32]. The base learners should come from contextual knowledge and a large set of default algorithms, ranging from a simple linear regression model to a multi-step algorithm [33]. To combine the prediction results of various algorithms and eliminate the inherent bias of the algorithm, we use six base learners: SVR, KNN, DT, RF, GBM, and XGB [34].

Base learners that predict the same validation set are strongly correlated. To explore the coefficients of the base learners, Leo Breiman (1996) proposed non-negative constraints that ensure the coefficients are not sensitive to small changes in data. The predicted values of the SL are also constrained to be between the minimum and maximum predicted values of base learners. The coefficients selected by cross-validation are combined with the predictions of base learners on the original dataset, not the cross-validated one. In this study, we calculate the coefficients using 5-fold cross-validation to generate SL. Algorithm 1, presented below, describes the SL algorithm that utilizes these six base learners.

Algorithm 1: Super Learner

Input : The dataset $\mathcal{D} = \{(X_i, Y_i) : i = 1, \dots, n\}$

Output: A super learner

Step 1: Train a base learner h_k with the dataset $\mathcal{D}$ for $k = 1, \dots, 6$.

Step 2: Split the dataset into 5 equal-size subsets

for $v = 1, \dots, 5$:

Set the v -th fold as a validation set $V(v)$ and the rest folds as a training set $T(v)$,

$T(v) = \mathcal{D} \setminus V(v)$.

end for

Step 3: For each base learner h_k , create the prediction set $h_k(\mathcal{D}^*)$ by aggregating predictions on $V(v)$, i.e., $h_k(V(v))$ in order $v = 1, \dots, 5$.

for $v = 1, \dots, 5$:

Train a base learner h_k on $T(v)$ and save the prediction set $h_k(V(v))$.

end for

Step 4: Let α_k ($k = 1, \dots, 6$) be the coefficient giving weight to $h_k(\mathcal{D}^*)$ under the constraints.

$$\sum_{k=1}^6 \alpha_k h_k(\mathcal{D}^*), \quad \text{s.t. } \alpha_k \geq 0 \quad \forall k, \quad \sum_{k=1}^6 \alpha_k = 1$$

Step 5: Determine the weight vector α that minimizes the cross-validation error using the following optimization.

$$\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}} \sum_{i=1}^n \left(Y_i - \sum_{k=1}^6 \alpha_k h_k(\mathcal{D}_i^*) \right)^2$$

Step 6: To create Super Learner, combine $\hat{\alpha}$ with the prediction of base learners that are generated in line 1.

4. Results

In this study, we used machine learning models and SL to predict the average hourly electrical power output of the CCPP and identified the most successful regression model by comparing their performances. We employed the dataset from [3], which comprised five randomly shuffled files, and performed two-fold cross-validation on each file. Hence, we evaluated and compared the prediction performance on 10 validation folds. The prediction performance was assessed by Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE).

(1) RMSE

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2} \quad (5)$$

(2) MAE

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - f(x_i)| \quad (6)$$

4.1. Pre-Processing of the Data

Due to the presence of outliers and significant variations in scales, the prediction performance of several machine learning algorithms decreases. Additionally, the computation of gradient-based estimators becomes more complicated and takes longer to converge.

Figure 2 illustrates the boxplots of the features, namely AT, V, AP, and RH. These input variables have different scales that can affect gradient-based estimators like SVR and KNN. Specifically, AP has a large scale compared to AT, V, and RH. To address this issue, we performed feature scaling to normalize all features to the same scale. While tree-based models such as DT, RF, GBM, and XGB do not require feature scaling, SVR and KNN do. In this study, we used standardization for feature scaling, which is unaffected by the presence of outliers. It is calculated using the mean and standard deviation of each feature, as follows:

$$\hat{x}_i = \frac{x_i - \mu}{\sigma} \quad (7)$$

where x_i is i -th raw data, and μ, σ are, respectively, mean and standard deviation of feature values.

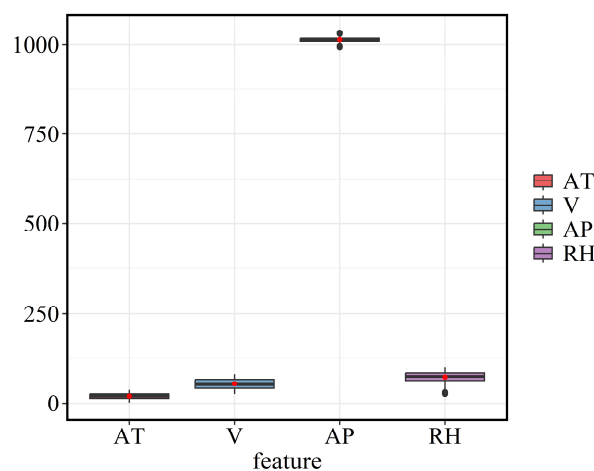


Figure 2. Box plot for input variables.

4.2. Information of Fitted Machine Learning Algorithms

We employed grid search cross-validation to identify the optimal parameter combination through cross-validation. Specifically, we adjusted hyper-parameters for the machine

learning algorithms using five-fold cross-validation based on mean squared error. Table 2 presents the hyper-parameters information of each model and the search range utilized in the grid search cross-validation.

Table 2. Hyper-parameter information of the machine learning models.

Model	Parameter	Description	Range
SVR	C	Regularization parameter	[0.1, 1, 10, 100]
	gamma	Kernel coefficient	[1, 0.1, 0.01, 0.001]
KNN	n_neighbors	The number of neighbors	range(3, 20, 2)
	p	Distance method (1: Manhattan, 2: Euclidean)	[1, 2]
	weights	Weight function used in prediction	[uniform, distance]
DT	min_samples_leaf	The minimal number of samples at the leaf node	range(5, 50)
RF	max_features	The number of random features when setting node	range(1, 5)
	n_estimators	The number of trees	range(100, 301, 100)
GBM	learning_rate	Learning rate	range(0.01, 0.5, 0.05)
	max_depth	The maximum depth of the tree	range(3, 11)
	n_estimators	The number of trees	range(100, 301, 100)
XGB	learning_rate	Learning rate	range(0.01, 0.5, 0.05)
	max_depth	The maximum depth of the tree	range(3, 11)
	n_estimators	The number of trees	range(100, 301, 100)

4.3. Information of the Fitted Super Learner

SL is the prediction model that employs six trained base learners, namely SVR, KNN, DT, RF, GBM, and XGB. Each base learner stores the predictions of five-fold cross-validation to determine the coefficient in step 3 of Algorithm 1. The stored predictions are utilized to calculate the coefficients of base learners in step 4. These coefficients are then combined with the predicted values of the base learners on the original dataset in step 6. In this study, SL uses base learners trained with the optimal hyper-parameters previously obtained.

Figure 3 shows the coefficients of base learners for 10 validation folds, which were created using two-fold cross-validation on each of five files. The sum of the coefficients for each fold remains at 1. There is a difference in coefficients for each fold, but XGB has the highest coefficient for all folds, followed by GBM. In several folds, SVR, KNN, and RF have coefficients of 0, and DT has all coefficients at 0. SL is more significantly affected by XGB and GBM than by the other models and is hardly affected by DT.

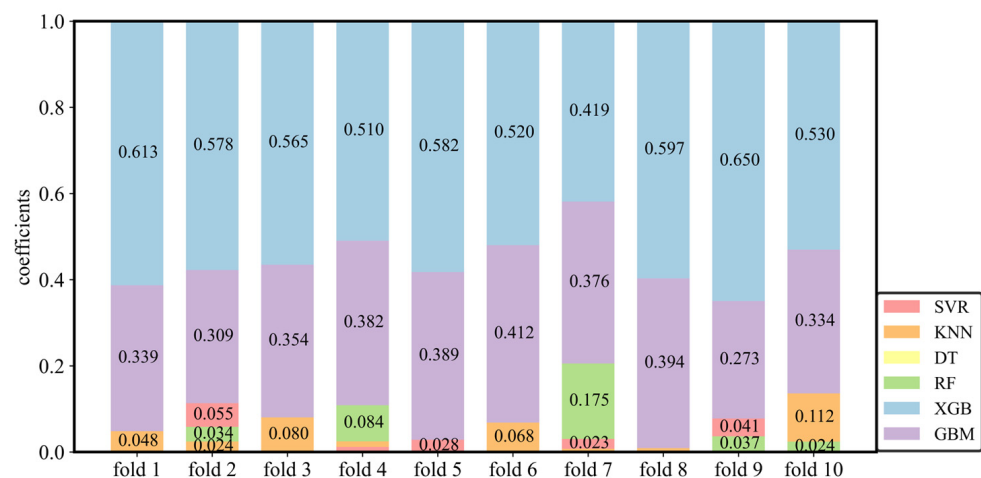


Figure 3. The coefficient-stacked bar displays the optimal weight combinations calculated from the SL for each of the 10-fold cross-validations. Coefficients below 0.02 are not displayed in the text.

4.4. Comparison of Model Performance

Figure 4 displays the distributions of actual and predicted values of the power output, estimated using a kernel density estimation technique. The solid black and red lines represent the distribution of the observation and prediction of SL, respectively. The remaining colored lines represent the distribution of the predicted values of the other machine learning models. All distributions exhibit a multimodal shape. The distributions of the machine learning models and SL are quite similar. However, it can be seen that the distribution of the models has relatively lower variability than the observed distribution.

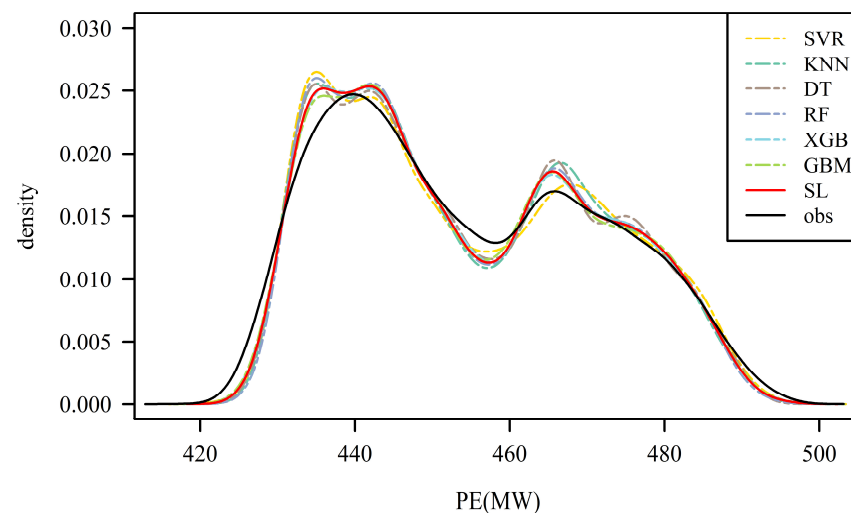


Figure 4. Kernel density estimation of the predicted values for each model and its corresponding actual values.

Figure 5 and Table 3 present the performance of each model in terms of RMSE and MAE for 10 validation folds, as well as their average performance. For each model, the black points represent the performance values for 10 validation folds, while the red plus sign indicates the average performance of 10 validation folds. Ensemble learning models performed better than traditional single models for all folds. Among all the models, SL achieved the best performance in terms of the average of RMSE and MAE. Except for Super Learner, XGB showed better average performance compared to the other base learners, followed by GBM and RF. Table 3 compares the average performance of the regression methods in our study and a previous article by [3]. KNN, RF, GBM, and XGB show better prediction skills than BREP on the average of RMSE and MAE. Furthermore, SL outperformed BREP by combining models that were either better or not better than BREP. This implies that SL is a stable model by combining the predictions of various models. SL demonstrated superior prediction skills compared to base learners, with the averages of RMSE and MAE at 3.1772 and 2.2537, respectively.

Figure 6 presents the predictive performance of SL and base learners in terms of RMSE and MAE for each fold. If the points are distributed close to the 45-degree line for the model, it indicates that the model's performance is similar to SL. Ensemble learning models are more closely aligned with the line compared to traditional single models, with XGB demonstrating the highest resemblance to SL. However, SL outperforms the base learners on all folds.

Figure 7 and Table 4 present the average RMSE and MAE obtained from two-fold cross-validation for five files. The machine learning models and SL demonstrate comparable performance across the files. Ensemble models exhibit significantly superior prediction performance compared to traditional machine learning models. Among the ensemble models, SL consistently achieves the lowest RMSE and MAE, followed by XGB, GBM, RF, KNN, SVR, and DT for all files.

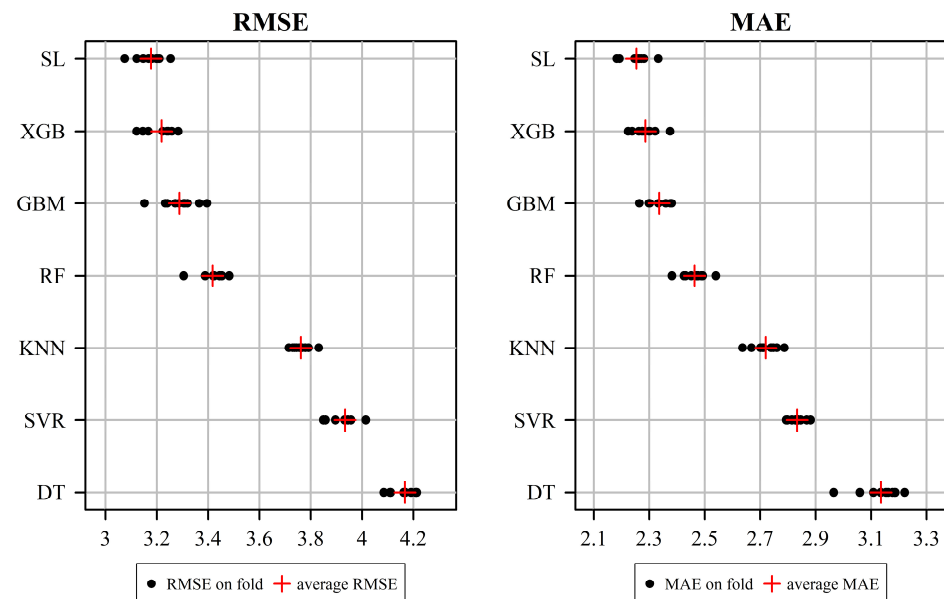


Figure 5. Comparisons of the prediction performances for each 10 validation folds and averaged performance in terms of RMSE and MAE.

Table 3. The average RMSE and MAE for SL ensembles and BREP.

Regression Method	Average RMSE	Average MAE
BREP [3]	3.787	2.818
SL	3.1772	2.2537
XGB	3.2188	2.2858
GBM	3.288	2.3356
RF	3.4176	2.4632
KNN	3.7613	2.7202
SVR	3.9336	2.8332
DT	4.1673	3.1356

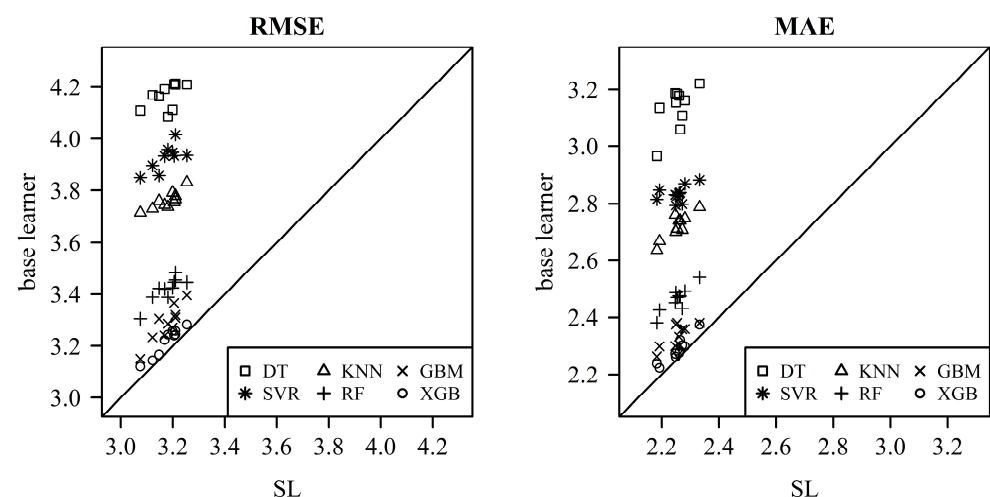


Figure 6. Comparison of the prediction performances of SL and base learners.

In Figure 8, the distribution of actual and predicted values of SL, which had the model with the most successful prediction performance, is presented. The predicted values of SL closely follow the observed pattern, indicating that the model predicts the power output of the CCPP using the four variables (AT, V, AP, and RH) similarly.

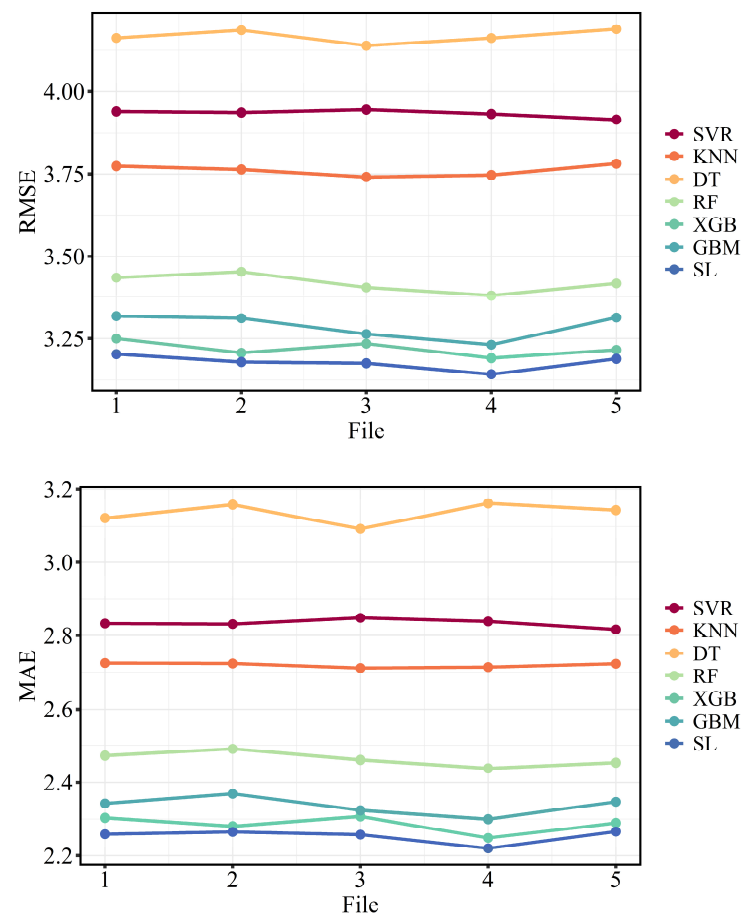


Figure 7. The average performances of 2-fold cross-validation on 5 files for each model.

Table 4. Prediction performance results of each model on 5 files.

	File 1		File 2		File 3		File 4		File 5		Mean	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SL	3.202	2.259	3.179	2.265	3.175	2.258	3.143	2.220	3.188	2.266	3.177	2.254
SVR	3.939	2.833	3.936	2.831	3.945	2.848	3.932	2.838	3.916	2.816	3.934	2.833
KNN	3.774	2.726	3.764	2.724	3.741	2.712	3.747	2.715	3.781	2.724	3.761	2.720
DT	4.162	3.122	4.186	3.158	4.138	3.094	4.161	3.161	4.189	3.143	4.167	3.136
RF	3.435	2.472	3.453	2.490	3.404	2.461	3.380	2.439	3.417	2.454	3.418	2.463
GBM	3.319	2.342	3.313	2.368	3.264	2.323	3.231	2.299	3.315	2.346	3.288	2.336
XGB	3.250	2.303	3.206	2.281	3.234	2.307	3.190	2.249	3.215	2.289	3.219	2.286

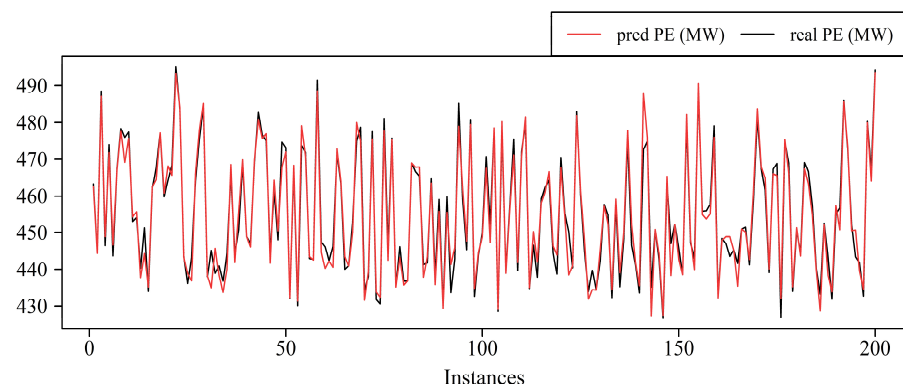


Figure 8. The plot of the real value and predicted value with SL for 200 points.

5. Conclusions

CCPPs generate additional electrical power from steam turbines by utilizing the exhaust heat from gas turbines, making them more efficient than simple cycles. Accurately predicting the full-load P_E of CCPP is crucial to ensure efficient and economical operation. In this study, six machine learning models, namely SVR, KNN, DT, RF, GBM, and XGB, were employed to predict the hourly electrical power output of a CCPP. These models were also utilized as base learners to generate SL. All models predict the hourly average P_E using input variables that affect the power output of gas and steam turbines, such as AT, V, AP, and RH. To generate 10 validation folds, we utilized five files from the randomly shuffled dataset and performed two-fold cross-validation on each file. Performance metrics for comparison are RMSE and MAE on the 10 validation folds. As a result, the prediction distributions of all models exhibit relatively lower variability compared to the distribution of the target P_E . Ensemble learning models, including RF, GBM, XGB, and SL, outperformed traditional single machine learning models. SL achieves the best prediction performance among the ensemble models, yielding a RMSE of 3.1772 and a MAE of 2.2537. It represents a more stable ensemble learning model by combining various models. Additionally, the prediction error incurred by SL is lower than that of BREP, which was previously proposed as the optimal model in a different study [3].

When estimating the coefficients of base learners, constraints were employed to minimize sensitivity to small fluctuations in data. In a future study, we aim to explore different methods for combining the predicted values of base learners in various aspects.

This study demonstrates that the SL model significantly improves the accuracy of CCPP power output predictions, contributing to enhanced operational efficiency and better-informed decision-making. The proposed framework offers a robust approach for optimizing CCPP performance under diverse operational conditions.

Author Contributions: Conceptualization, Y.S. and J.P.; methodology, C.K.; software, Y.S. and J.P.; validation, Y.S. and J.P.; writing—original draft preparation, Y.S.; writing—review and editing, Y.S. and J.P.; project administration, C.K.; funding acquisition, M.-S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Korea Meteorological Administration Research and Development Program, grant number RS-2023-00239653 and was supported by a research grant from the Kongju National University in 2024.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available in UCI Machine Learning Repository at <https://doi.org/10.24432/C5002N> (Tfekci and Kaya, 2014).

Conflicts of Interest: Author Yujeong Song was employed by the company Penta Systems Technology; Author Jisu Park was employed by the company YOUNGJIN Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Polyzakis, A.L.; Koroneos, C.; Xydis, G. Optimum gas turbine cycle for combined cycle power plant. *Energy Convers. Manag.* **2008**, *49*, 551–563. [\[CrossRef\]](#)
2. Shin, J.Y.; Jeon, Y.J.; Maeng, D.J.; Kim, J.S.; Ro, S.T. Analysis of the dynamic characteristics of a combined-cycle power plant. *Energy* **2002**, *27*, 1085–1098. [\[CrossRef\]](#)
3. Tüfekci, P. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *Int. J. Electr. Power Energy Syst.* **2014**, *60*, 126–140. [\[CrossRef\]](#)
4. Santarisi, N.S.; Faouri, S.S. Prediction of combined cycle power plant electrical output power using machine learning regression algorithms. *East.-Eur. J. Enterp. Technol.* **2021**, *6*, 114. [\[CrossRef\]](#)
5. Erdem, H.H.; Sevilgen, S.H. Case study: Effect of ambient temperature on the electricity production and fuel consumption of a simple cycle gas turbine in Turkey. *Appl. Therm. Eng.* **2006**, *26*, 320–326. [\[CrossRef\]](#)

6. Roni, M.H.K.; Khan, M.A.G. An artificial neural network based predictive approach for analyzing environmental impact on combined cycle power plant generation. In Proceedings of the 2nd International Conference on Electrical & Electronic Engineering (ICEEE), Rajshahi, Bangladesh, 27–29 December 2017; pp. 1–4. [\[CrossRef\]](#)
7. Zhao, Y.; Foong, L.K. Predicting electrical power output of combined cycle power plants using a novel artificial neural network optimized by electrostatic discharge algorithm. *Measurement* **2022**, *198*, 111405. [\[CrossRef\]](#)
8. Mabel, M.C.; Fernandez, E. Analysis of wind power generation and prediction using ANN: A case study. *Renew. Energy* **2008**, *33*, 986–992. [\[CrossRef\]](#)
9. Amirkhani, S.; Nasirivatan, S.; Kasaeian, A.B.; Hajinezhad, A. ANN and ANFIS models to predict the performance of solar chimney power plants. *Renew. Energy* **2015**, *83*, 597–607. [\[CrossRef\]](#)
10. Smrekar, J.; Pandit, D.; Fast, M.; Assadi, M.; De, S. Prediction of power output of a coal-fired power plant by artificial neural network. *Neural Comput. Appl.* **2010**, *19*, 725–740. [\[CrossRef\]](#)
11. Janoušek, J.; Gajdoš, P.; Dohnálek, P.; Radecký, M. Towards power plant output modelling and optimization using parallel Regression Random Forest. *Swarm Evol. Comput.* **2016**, *26*, 50–55. [\[CrossRef\]](#)
12. Alketbi, S.; Nassif, A.B.; Eddin, M.A.; Shahin, I.; Elnagar, A. Predicting the power of a combined cycle power plant using machine learning methods. In Proceedings of the 2020 International Conference on Communications Computing, Cybersecurity, and Informatics (CCCI), Sharjah, United Arab Emirates, 3–5 November 2020; pp. 1–5. [\[CrossRef\]](#)
13. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [\[CrossRef\]](#)
14. Schapire, R.E. The strength of weak learnability. *Mach. Learn.* **1990**, *5*, 197–227. [\[CrossRef\]](#)
15. Wolpert, D.H. Stacked generalization. *Neural Netw.* **1992**, *5*, 241–259. [\[CrossRef\]](#)
16. Alexandropoulos, S.A.N.; Aridas, C.K.; Kotsiantis, S.B.; Vrahatis, M.N. Stacking strong ensembles of classifiers. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 545–556. [\[CrossRef\]](#)
17. Breiman, L. Stacked regressions. *Mach. Learn.* **1996**, *24*, 49–64. [\[CrossRef\]](#)
18. Van der Laan, M.J.; Polley, E.C.; Hubbard, A.E. Super Learner. *Stat. Appl. Genet. Mol. Biol.* **2007**, *6*, 25. [\[CrossRef\]](#)
19. Baheta, A.T.; Gilani, S.I.U.H. The effect of ambient temperature on a gas turbine performance in part load operation. *AIP Conf. Proc.* **2012**, *1440*, 889–893. [\[CrossRef\]](#)
20. Sulistiyo, E.; Kristina, E.A.; Muhammad, R. Analysis of the effect of changes in vacuum pressure on condenser effectiveness before and after overhaul in Blok 1 PLTGU UP Gresik. *Int. J. Soc. Rev.* **2024**, *2*, 1106–1117.
21. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [\[CrossRef\]](#)
22. Basak, D.; Pal, S.; Patranabis, D.C. Support vector regression. *Neural Inf. Process. Lett. Rev.* **2007**, *11*, 203–224.
23. Rahman, F.I. Short term traffic flow prediction using machine learning—KNN, SVM and ANN with weather information. *Int. J. Traffic Transp. Eng.* **2020**, *10*, 371–389. [\[CrossRef\]](#)
24. Beskopylny, A.N.; Stel'makh, S.A.; Shcherban', E.M.; Mailyan, L.R.; Meskhi, B.; Razveeva, I.; Chernil'nik, A.; Beskopylny, N. Concrete strength prediction using machine learning methods CatBoost, k-Nearest Neighbors, Support Vector Regression. *Appl. Sci.* **2022**, *12*, 10864. [\[CrossRef\]](#)
25. Goldberger, J.; Hinton, G.E.; Roweis, S.; Salakhutdinov, R.R. Neighbourhood components analysis. *Adv. Neural Inf. Process. Syst.* **2004**, *17*.
26. Breiman, L.; Friedman, J.H.; Olshen, R.A.; Stone, C.J. *Classification and Regression Trees*; Routledge: Oxford, UK, 2017. [\[CrossRef\]](#)
27. Shabani, S.; Samadianfard, S.; Sattari, M.T.; Mosavi, A.; Shamshirband, S.; Kmet, T.; Várkonyi-Kóczy, A.R. Modeling pan evaporation using Gaussian process regression K-nearest neighbors random forest and support vector machines; comparative analysis. *Atmosphere* **2020**, *11*, 66. [\[CrossRef\]](#)
28. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [\[CrossRef\]](#)
29. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [\[CrossRef\]](#)
30. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [\[CrossRef\]](#)
31. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794. [\[CrossRef\]](#)
32. Lee, S.; Nguyen, N.H.; Karamanli, A.; Lee, J.; Vo, T.P. Super learner machine-learning algorithms for compressive strength prediction of high performance concrete. *Struct. Concr.* **2023**, *24*, 2208–2228. [\[CrossRef\]](#)
33. Polley, E.C.; Van der Laan, M.J. *Super Learner in Prediction*; U.C. Berkeley Division of Biostatistics Working Paper Series; Berkeley Graduate Division: Berkeley, CA, USA, 2010; Volume 266.
34. Zhang, Y.; Wen, M.; Sun, Y.; Chen, H.; Cai, Y. Black carbon emission prediction of diesel engine using stacked generalization. *Atmosphere* **2022**, *13*, 1855. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.