In-Processing Fairness Improvement Methods for Regression Data-Driven Building Models: Achieving Uniform Energy Prediction

Ying Sun[1], Benjamin C. M. Fung[2], Fariborz Haghighat*[1]

[1]Energy and Environment Group, Department of Building, Civil and Environmental Engineering Concordia, University, Montreal, Canada
[2]School of Information Studies, McGill University, Montreal, Canada

Abstract

In recent years, the massive data collection in buildings has paved the way for the development of accurate data-driven building models (DDBMs) for various applications. However, a model with a high overall accuracy would not ensure a good predictive performance on all conditions. The biased predictive performance for some conditions may cause fairness problems. Although pre-processing methods were proposed to improve predictive fairness by removing discrimination from training datasets for classification problems in building engineering domain, they lack the ability of achieving user-defined trade-off between fairness and accuracy for regression problems, such as energy prediction. To improve the predictive fairness of regression models in terms of having similar predictive performance between different conditions, this study proposes four in-processing methods, namely mean residual difference penalized (MRDP) regression, mean square error penalized (MSEP) regression, mean residual difference constrained (MRDC) regression, and mean square error constrained (MSEC) regression, to add fairness-related penalties or constraints to the loss function of regression models. Then, these proposed methods are applied to develop linear regression models for energy prediction of an apartment. In this case study, improving predictive fairness means to let the energy predictive accuracy be uniform no matter if there is occupancy movement. The result shows that MSEC is the most powerful method to improve fairness in terms of mean square error (MSE) rate and mean absolute error (MAE) rate, while MSEP is another good option to improve fairness without a significant decrease on the overall accuracy. MRDC is effective on improving the similarity of absolute mean residual difference (abs(MRD)) between different conditions, however, MRDP would not affect the predictive result.

Keywords: Fairness; Accuracy; Data-driven model; Energy prediction; Building

Corresponding Author*: Fariborz.Haghighat@Concordia.ca

1. Introduction

In recent years, the widespread installation of smart sensors, Internet-of-Things, and smart home energy management systems (HEMSs) makes buildings data-rich [1]. The abundant data could be utilized to train data-driven models to represent building states, such as indoor air temperature [2], indoor air quality [3], HVAC operation status [4,5], energy consumption [6,7], etc. Among these models, energy prediction models could be integrated into model predictive controller to provide optimal control signals for building energy management systems to achieve energy saving, cost saving, and/or peak shifting [8]. Accurate energy prediction could also benefit suppliers in energy generation and distribution planning [9].

Existing energy prediction models mainly treat energy prediction as regression problems whose outputs are continuous values [10]. The primary aim of these models is to be accurate enough so that predictive results are close enough to measured values. Commonly used accuracy measures for these regression models include MAE, MAPE, RMSE, CV(RMSE), R^2, etc. [11]. However, a high overall accuracy could not ensure the predictive performance is fairly perfect in different conditions. In fact, improving predictive performance similarity between distinct conditions could ensure that the predictive model provides a fair service to the users by ensuring them to receive a uniform predictive performance. For instance, if one energy predictive model is more accurate when some occupants are in the building than the period that some other occupants are in the building, energy scheduling service provided based on the model would be more efficient for the former occupants, thus, it would be unfair to other ones who receive less accurate information.

Fairness problems are varied and could mainly be classified into three categories based on the relationship between predictive results and the protected attribute(s) [12]: *Type I.* The predicted output is independent of the protected attribute(s). *Type II.* The predictive performance is similar across classes/conditions defined by the protected attribute(s). *Type III.* The predicted output is independent of the predictive probability score for samples coming from different classes/conditions defined by the protected attribute(s). Here, protected attributes are also called sensitive attributes. They define conditions in which the predictive result is not willing to be biased. For instance, race, age, gender, occupancy-related data (e.g., occupancy status, occupancy count, and occupants' occupation, etc.) are commonly used protected attributes because different predictive service for conditions defined by them could cause discrimination problems. Note that achieving different types of fairness at the same time is almost impossible [13] because of their distinct evaluation criteria. Thus, researchers are recommended to clearly define which type of fairness that they plan to achieve. In this study, achieving fairness refers to have uniform predictive accuracy among different conditions defined by the protected attribute (i.e., *Type II* fairness).

One probable reason behind the distinct predictive accuracy of different conditions is the imbalanced training dataset. For instance, in reality, most operation data of HVAC devices are of normal status, while only a few are in faulty condition. As a result, the data-driven model trained based on this imbalanced dataset would work perfectly on predicting normal status, but show worse performance for faulty detection. Therefore, to narrow the predictive performance between different conditions, one useful way is to reduce the discrimination among training dataset. In other word, it means to produce a balanced training dataset that contains a similar amount of data for all conditions. These methods belong to data pre-processing methods.

The easiest way to get a balanced training dataset is to oversample for minority conditions and/or undersample for majority conditions. For instance, the synthetic minority oversampling technique (SMOTE) that samples data for minority conditions by linear interpolation between minority samples has been used to oversample faulty samples to increase the fault detection accuracy of HVAC devices [14,15]. However, large oversampling size may increase the classification uncertainty because of the change in data distribution. Furthermore, this method has not been used to solve fairness problems. To eliminate the bias among conditions defined by the protected attribute and output label, Kamiran and Calders [16] proposed uniform sampling and preferential sampling: uniform sampling randomly duplicate data for minority conditions or delete data from majority conditions, while preferential sampling duplicate/delete data that closest to the decision boundary. However, these methods have not been applied to omit the bias among the training dataset of data-driven building models (DDBMs). To remove discrimination from the training dataset of DDBMs, we proposed four types of data pre-processing methods, namely sequential sampling, sequentially balanced preferential sampling, reversed preferential sampling, and sequential preferential sampling, to produce a balanced training dataset for classification problems. The generalizability of these proposed methods on fairness and accuracy of DDBMs is compared with uniform sampling and preferential sampling in [12,17].

Besides, generating representative data is an efficient way to enrich data points in minority conditions. For instance, Yan et al. [5] applied the generative adversarial network (GAN) to generate faulty training samples for fault detection and diagnosis (FDD) of air handling units (AHUs). Their study found that the re-balanced training dataset could improve the diagnostic accuracy of traditional data-driven models (e.g., random forest (RF), support vector machine (SVM), multi-layer perceptron (MLP), k-nearest-neighbor (KNN) and decision tree (DT)) from nearly 50% to almost 100%. Yan et al. [19] have also applied the GAN to re-balance the training dataset for automatic FDD for chillers. Li et al. [1] proposed a GAN to improve the diagnostic accuracy for building HVAC systems by taking advantage of the re-balanced labeled and unlabeled data. However, these data generative models usually face the non-convergence issue [20]. Besides, the representativity of created data would highly affect the predictive performance, including fairness and accuracy.

These data pre-processing methods could eliminate bias among training dataset, and thus, improve the predictive accuracy among minority conditions. However, it could not quantitively ensure that the predictive accuracy is similar enough between different conditions. Besides, data pre-processing methods are more suitable for classification problems because conditions defined by the output label and protected attribute are less and easier to be determined.

To quantitively define the required performance similarity for regression problems like building energy consumption, in-processing methods that set fairness-related constraints or penalties in the loss function of model training would be a good option [21]. In-processing methods could achieve specific fairness measures chosen by the programmer while preserving high accuracy. However, as the type of fairness measure could vary among different predictive tasks, the code may need to be changed accordingly.

There are mainly three types of fairness improvement in-processing methods: fairness constraints, prejudice remover regularizer, and adversarial debiasing. Among these methods, the fairness constraints method adds fairness constraints to the loss function of the training process [22]; the prejudice remover regularizer method applies a fairness regularizer to the loss function [23]; while the adversarial debiasing method develops a predictor and an adversary at the same time to weaken the power of predicting the protected attribute from the predictive outputs [24].

In-processing fairness improvement methods have been used in regression or classification problems to ensure similar income prediction performance [22,25,26], loan allocation result [22,23], or violent recidivism prediction performance [23,24,27,28] for people coming from different race or gender. Yet, they have never been applied to data-driven building models to achieve uniform energy predictive accuracy among end users.

In building engineering domain, the cost-sensitive algorithm, a kind of in-processing methods that increases classification accuracy of specified conditions through assigning higher misclassification cost for these conditions, has been used to increase the predictive accuracy of user-defined conditions [29] or faulty conditions [30–32]. However, they have not been applied to solve fairness problems. Further, most of existing studies on cost-sensitive algorithms focus on classification problems instead of regression problems.

The previous review of the literature shows that improving fairness for building energy prediction models is required to ensure uniform predictive performance under all conditions, which means making sure that services provided based on the predictive result are non-discriminatory. Besides, there are many fairness improvement methods. Among them, in-processing methods show the ability to achieve user-defined fairness. However, fairness-related constraints or penalty should be set based on the specific problem. To the best of the authors' knowledge, there is no existing study focusing on quantitively improving fairness among regression problems in building engineering domain. This study fills this gap and has the following major contributions:

- Investigate the possibility of improving fairness to have uniform predictive performance under different conditions for building energy prediction,
- Propose four in-processing fairness improvement methods for regression problems to improve fairness, by setting user-defined constraints or penalties for improving predictive performance similarity between different conditions, and
- Implement the proposed methods to develop fairness-aware linear regression models for building energy prediction. Both predictive accuracy and fairness are evaluated.

The outline of this paper is: Section 2 introduces the proposed in-processing techniques and optimization algorithm used for solving the optimization problem defined by in-processing methods. A case study that applies these methods to improve the energy predictive fairness while preserving predictive accuracy is designed and introduced in this section. In Section 3, results are analyzed in terms of accuracy measures and fairness measures. Then, Section 4 discusses the effect of loss functions and optimization algorithms. Finally, Section 5 summarizes the conclusion.

2. Methodology

2.1. In-processing fairness improvement methods

Training a model means learning proper model parameters to minimize a loss function (denote as *Loss*) that indicates the closeness of predicted values to their corresponding actual values. For regression models, commonly used *Loss* includes mean square error (MSE, see Equation 1) and mean absolute error (MAE, see Equation 2). MSE calculates the mean of squared error losses (also called L2 loss). Square loss is the square of residual difference, which is the difference between the actual value and the predicted value. MAE is the mean of absolute errors, which are also known as L1 losses. Absolute error is the distance between the actual value and predicted value. Generally, MSE loss function converges faster than MAE, because the quadratic function of MSE makes it easier to find the gradient. However, the MAE loss function shows the advantage of being more robust to outliers than MSE.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1:n}(y_i - \hat{y}_i)^2 \tag{1}$$

where n is the number of training data samples, y is the measured value, $\hat{y}$ is the predicted value.

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1:n}|y_i - \hat{y}_i| \tag{2}$$

However, minimizing MSE or MAE could not make sure that the predictive performance is similar among different conditions. To solve this problem, this section would present four in-processing fairness improvement methods that add penalties or constraints to *Loss* in order to narrow the predictive performance difference between conditions defined by the protected attribute. To make a simple explanation, the original loss function (such as MSE or MAE) without considering fairness is denoted by *Loss_ori*, and the protected attribute is assumed as a binary attribute. Note that these methods could be extended to problems with multi-class protected attribute through adding pair-wise constraints/penalties.

2.1.1. Mean residual difference penalized (MRDP) regression

The loss function of this method, as shown in Equation 3, comprises a *Loss_ori* that illustrates the overall predictive accuracy and a prejudice remover regularizer that indicates the difference magnitude between the mean residual difference when the protected attribute (denoted by *S*) is Positive and the mean residual difference when S = Negative. The difference magnitude is squared to avoid negative values. Besides, users could justify the trade-off between accuracy and fairness, though setting the multiplier λ for the regularizer. The bigger the λ, the more important the fairness.

$$min\, Loss\_ori + \lambda[\frac{1}{s0}\sum_{h=1:s0,S=negative}(y_h - \hat{y}_h) - \frac{1}{s1}\sum_{k=1:s1,S=positive}(y_k - \hat{y}_k)]^2 \quad (3)$$

where λ is the multiplier of the prejudice remover regularizer; S is the protected attribute, $S \in$ [Negative, Positive]; *s0* is the number of training data with S = Negative, *s1* is the number of training data with S = Positive.

The regularizer in Equation 3 could be rewritten as Equation 4. It explains that the regularizer calculates the square difference between the mean actual value difference and mean predicted value difference among conditions with S = Negative and S = Positive.

$$[\left(\frac{1}{s0}\sum_{h=1:s0,S=negative} y_h - \frac{1}{s1}\sum_{k=1:s1,S=positive} y_k\right) - \left(\frac{1}{s0}\sum_{h=1:s0,S=negative} \hat{y}_h - \frac{1}{s1}\sum_{k=1:s1,S=positive} \hat{y}_k\right)]^2 \quad (4)$$

2.1.2. Mean square error penalized (MSEP) regression

In this method, fairness significance is represented by the absolute difference between the MSE when S=Positive and MSE when S=Negative, see Equation 5. This method avoids the error cancellation within a condition caused by over-predicting for some samples and under-predicting for other samples in the same condition. Similar to MRDP, the trade-off between accuracy and fairness is justified by λ. Note that when λ=+∞, the MSEP could be considered as a method of Lagrange multiplier that is aimed at finding the minimum *Loss_ori* subject to the equality constraint that makes the MSE when S=Positive to be the same as the MSE when S=Negative.

$$min\, Loss\_ori + \lambda| \frac{1}{s0}\sum_{h=1:s0,S=negative}(y_h - \hat{y}_h)^2 - \frac{1}{s1}\sum_{k=1:s1,S=positive}(y_k - \hat{y}_k)^2 | \quad (5)$$

2.1.3. Mean residual difference constrained (MRDC) regression

MRDC is inspired by adding fairness-related constraints to *Loss_ori* to limit the predictive performance similarity between conditions defined by the protected attribute. The objective function of MRDC for model training, as shown in Equation 6, defines fairness by letting the absolute mean residual difference (denote as abs(MRD)) when S = Negative to be at least p or at most $\frac{1}{p}$ of abs(MRD) when S = Positive. When p =0.8, it infers that this method is trying to achieve fairness in terms of the "80 percent rule" [33]: the predictive result is fair when the predictive performance of any protected group is at least 80% of the highest predictive performance of the protected groups.

$$min\, Loss\_ori \tag{6}$$

Subject to $$\left|\frac{1}{s0}\sum\nolimits_{h=1:s0,S=negative}(y_h-\hat{y}_h)\right| \le \frac{1}{p} * \left|\frac{1}{s1}\sum\nolimits_{k=1:s1,S=positive}(y_k-\hat{y}_k)\right|$$

$$\left|\frac{1}{s0}\sum\nolimits_{h=1:s0,S=negative}(y_h-\hat{y}_h)\right| \ge p * \left|\frac{1}{s1}\sum\nolimits_{k=1:s1,S=positive}(y_k-\hat{y}_k)\right|$$

where p infers the similarity of predictive performance among different conditions defined by the protected attribute.

2.1.4. Mean square error constrained (MSEC) regression

The loss function of MSEC is present in Equation 7. Its fairness-related constrains are aimed at making the MSE when S = Negative to be at least p or at most $\frac{1}{p}$ of the MSE when S = Positive. It shows the advantage of considering the predictive error of each individual points, while mean residual difference in MRDC makes the overall predictive error be mitigated by different individuals in the same group.

$$min\, Loss\_ori \tag{7}$$

Subject to $$\frac{1}{s0}\sum\nolimits_{h=1:s0,S=negative}(y_h-\hat{y}_h)^2 \le \frac{1}{p} * \frac{1}{s1}\sum\nolimits_{k=1:s1,S=positive}(y_k-\hat{y}_k)^2$$

$$\frac{1}{s0}\sum_{h=1:s0,S=negative}(y_h-\hat{y}_h)^2 \ge p * \frac{1}{s1}\sum_{k=1:s1,S=positive}(y_k-\hat{y}_k)^2$$

2.2. Optimization algorithm

Considering the increased complexity of the loss function by in-processing methods, derivative-free optimization algorithms that do not use derivatives or finite differences [34] would be better options to train regression model parameters. Notable derivative-free optimization algorithms mainly include Bayesian optimization, adaptive coordinate descent, genetic algorithms (GA), differential evolution (DE), simulated annealing, particle swarm optimization (PSO), etc. Among these algorithms, DE has been proofed to be effective in solving constrained optimization problems [35]. Therefore, it will be selected as the solver for optimization problems set by in-processing methods.

DE is a heuristic approach that gets the global optimal solution by iteratively improving the candidate solution based on an evolutionary process [36]. Its general procedure is presented in Figure 1. Detailed description of each step is given as below:

Population Initialization: Generate a random or user-defined initial population that contains a set of candidate solutions.

Fitness assignment: Evaluate the fitness score of each solution through a fitness function to determine how fit the solution is.

Selection: Select a set of solutions (parents) based on some selection procedures for the mutation process to create the unit vector.

Mutation: Mutate a unit vector through adding a scaled differential vector to a target vector. Here, the differential vector is the difference between the two or more selected parents, while the target vector is the parent with prioritized direction of creating the unit vector.

Crossover: Generate new offspring by crossing over a selected 'major' parent (different from the parents used in mutation) and the unit vector created from mutation, and then, add the offspring to the population. Crossover methods mainly include average and intuitive.

Stop criteria: Terminate the algorithm if the population has converged (its offspring would not significantly increase the fitness) or if the maximum number of iterations has been reached.

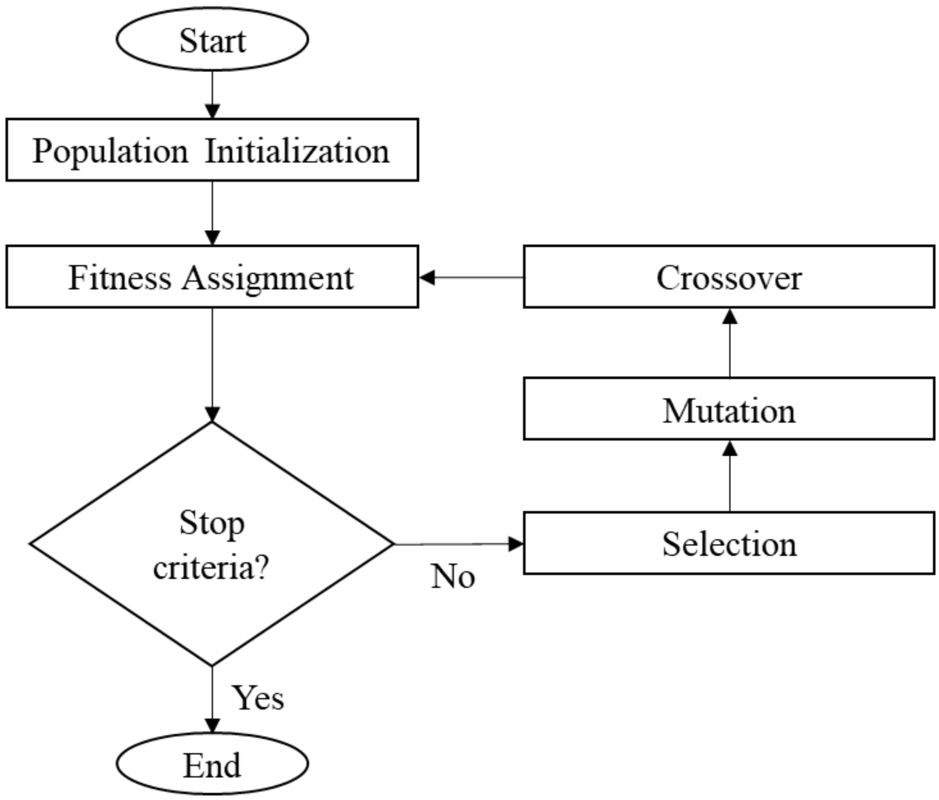


Figure 1: General procedure of a basic DE

2.3.Case study: Energy prediction

To investigate the applicability of proposed in-processing methods to solve fairness problems in building engineering domain, a case study is designed to apply these methods to train regressive models to predict hourly energy consumption of an apartment. In the case study, motion status is the binary protected attribute, which means the in-processing methods are aimed at presenting similar energy predictive performance no matter if there is occupancy movement in the apartment. Detailed description of data collection and feature selection are presented in Section 2.3.1, while the study cases are explained in Section 2.3.2.

2.3.1. Data description and feature selection

Building-related data used in this study was collected by sensors and HEMS in a three-bedroom apartment in Lyon, France, whose layout is shown in Figure 2. The data collection techniques and devices are explained in details in [37–40]. The original dataset was collected with one-minute time interval during the year of 2016. It contains information of time index (time of the day, day of the week), indoor temperature, indoor humidity, CO_2 concentration, motion status, window opening status, blind position, lighting status and lighting power consumption, as well as plug power consumption. Note that the summary of plug load and lighting energy consumption data (Wh) represents the building energy use in this study. Besides, weather information, such as ambient air temperature, and humidity, wind speed and direction, solar radiation, solar illuminance, etc., were collected with one-minute time interval from a local weather station in Vaulx-en-Velin, France.

As a larger time interval could increase the data representativity and acceptable predictive runtime [41], collected data was processed to one-hour resolution. Motion status was recorded as '**1**' if there was any movement detected by the corresponding presence sensor during the 60 minutes in that hour. One attribute called 'Motion status_Total' was added as a candidate input feature to represent if there is any movement detected in the studied apartment during one hour. It is assumed as the protected attribute, which means S=Positive is the condition that there is occupancy movement and S=Negative represents there is no detected movement in the apartment. Besides, the same sample strategy was applied to lighting status to evaluate if there was any light opening during one hour. Energy consumption data within one hour was averaged by minute data and normalized to be the range between **0** and **1**. For other attributes, the hourly value was sampled every 60 minutes. Besides, as energy consumption may belong to time series data that historical energy consumption would affect future values [11], previous 24 hours' normalized hourly energy consumption (NHEC) data and previous 168th NHEC are also added as candidate features. Overall, there are 106 candidate features. A list of these features are presented in the supplementary information.

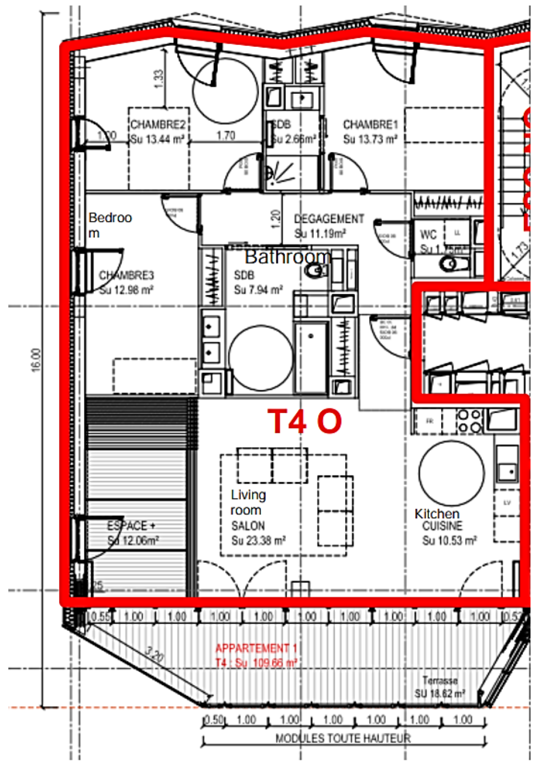


Figure 2: Footprint of the studied apartment

The NHEC of lighting and plug-ins is the output of data-driven models. Its distribution is shown in Figure 3. In the collected dataset, NHEC is lower than 0.7 most of the time. To select the most representative features for NHEC prediction, correlation between the candidate features and the output is calculated by Equation 8. Features whose correlation with the output is higher than 0.3 are selected as inputs. The selected 15 input features and their correlation with the output is present in Table 1.

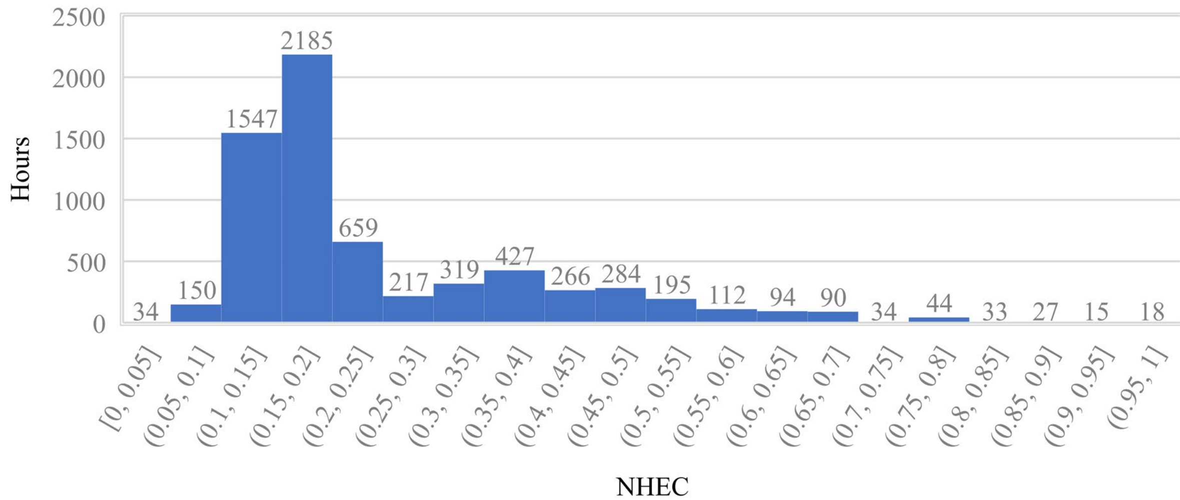


Figure 3: NHEC distribution

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}} \tag{8}$$

where r_{xy} is the Pearson correlation coefficient, x is the candidate input feature, $\bar{x}$ is the mean of corresponding x, $\bar{y}$ is the mean of y.

Table 1: Selected input features and their correlation with the output attribute

Input feature	**Correlation with the output**	**Input feature**	**Correlation with the output**	**Input feature**	**Correlation with the output**
Sun altitude	0.30	Motion status 5	0.42	Motion status_Total	0.45
Motion status 1	0.58	Motion status 6	0.53	NHEC t-1	0.57
Motion status 2	0.59	Motion status 7	0.46	NHEC t-2	0.34
Motion status 3	0.57	Motion status 8	0.41	NHEC t-23	0.31
Motion status 4	0.39	Motion status 13	0.32	NHEC t-24	0.33

Note that the number after the name of motion status means the corresponding measurement device, while the time index after NHEC illustrates the normalized hourly energy consumption at the corresponding time, t is the current time.

2.3.2. Case description

As the start point of investigating in-processing fairness improvement methods in building energy application, a relatively simple regression model, i.e., linear regression (see Equation 9), is used in this study to predict the normalized hourly energy consumption (NHEC).

$$\hat{y} = w_0 + \sum_{j=1:m} w_j x_j \tag{9}$$

where w and w_0 are parameters that need to be estimated during model training, m is the number of input features.

In this study, a reference case that uses MSE as the loss function to learn parameters of the developed linear regression model is conduced to be the basis when evaluating the fairness improvement ability of in-processing methods. Then, other case studies are designed to investigate the effects of in-processing methods and their corresponding p or λ values on the predictive result, see Table 2. MSE is the *Loss_ori* for these cases.

Table 2: Description of study cases

Case name	**In-processing methods**	***p* or λ value**
Reference case		
MRDP_0.6	MRDP	0.6
MRDP_0.8	MRDP	0.8
MSEP_0.6	MSEP	0.6
MSEP_0.8	MSEP	0.8
MRDC_0.6	MRDC	0.6
MRDC_0.8	MRDC	0.8
MSEC_0.6	MSEC	0.6
MSEC_0.8	MSEC	0.8

Note that constraints that limit the predicted NHEC within [0, 1] are added to the loss function of all cases

The DE is coded using the scikit-opt package [42], and its hyperparameters are shown in Table 3. For all cases, a 10-fold cross validation process is used for training and validating. MAE and MSE are used to evaluate the predictive performance. To be more specific, 'MSE_TOTAL' and 'MAE_TOTAL' is the overall accuracy. '1-MSE' and '1-MAE' means MSE and MAE when S=Positive, respectively. '0-MSE' and '0-MAE' is MSE and MAE when S=Negative, respectively. Besides, as the goal of this study is to improve fairness in terms of increasing the similarity of predictive performance between different conditions, fairness could be evaluated by the difference between 1-MSE and 0-MSE or the difference between 1-MAE and 0-MAE. The smaller the difference, the better the predictive fairness. On the other hand, it could also be evaluated by MSE rate (the rate between 1-MSE and 0-MSE) or MAE rate (the rate between 1-MAE and 0-MAE). Higher MSE rate or MAE rate means a better fairness achievement. For example, if MSE rate or MAE rate is higher than 0.8, the "80 percent rule" is achieved.

Table 3: Hyperparameters of DE

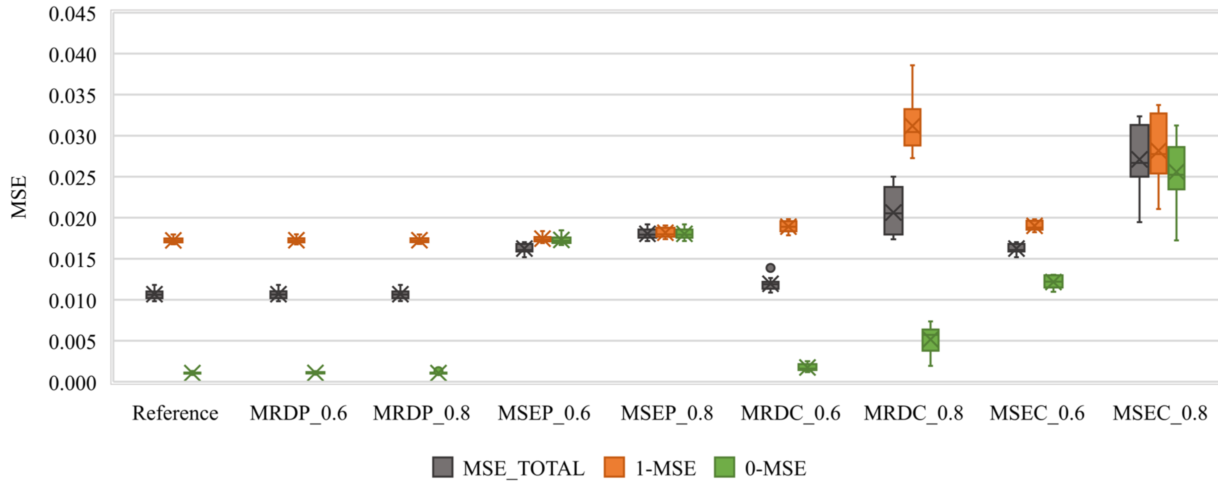
Hyperparameter	**Meaning**	**Value**
size_pop	Size of population	50
max_iter	Max iteration	1000
prob_mut	Probability of mutation	0.001
F	Coefficient of mutation	0.5

Simulations are run by Python 3.7 on a desktop with Intel Core i7-4790 CPU @3.60GHz and 8GB of RAM.

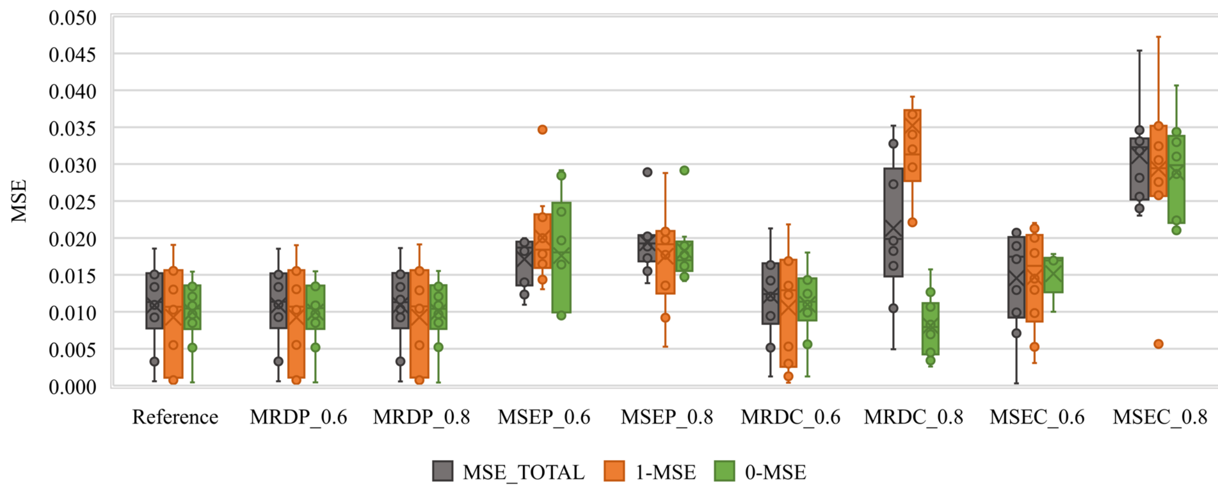
3. Results

3.1. Accuracy in terms of MSE and fairness in terms of MSE rate

The effect of four proposed in-processing methods on the predictive MSE during model training and validation are compared in Figure 4. It shows that MRDP would not affect the predictive accuracy in terms of MSE no matter during model training or model validation. As illustrated in Section 2.1.1, the regularizer added by MRDP tries to make the predictive result fair for conditions with S=Positive and S=Negative to have similar mean residual difference. In other words, it means to make the difference of mean measured value between the condition that S = Negative and the condition that S = Positive to be the same as the difference of mean predicted value between the condition when S = Negative and the condition when S = Positive. As shown in Figure 5, even in the reference case, the mean predicted NHEC is the same as the mean measured NHEC, irrelevance of S= Positive or S= Negative. Thus, the regularizer added by MRDP did not work and it is always almost equal to zero in this case study. Another interesting finding from Figure 5 is that the energy consumption when there are occupant activities in the apartment (when S= Positive) is more than twice of the energy consumed during the period that no occupancy movement is detected (S= Negative). It shows that occupancy-related data would be an important input for energy prediction of residential buildings.



(a) Model training



(b) Model validation

Figure 4: Effect of in-processing methods on the predictive accuracy in terms of MSE during (a) model training and (b) model validation

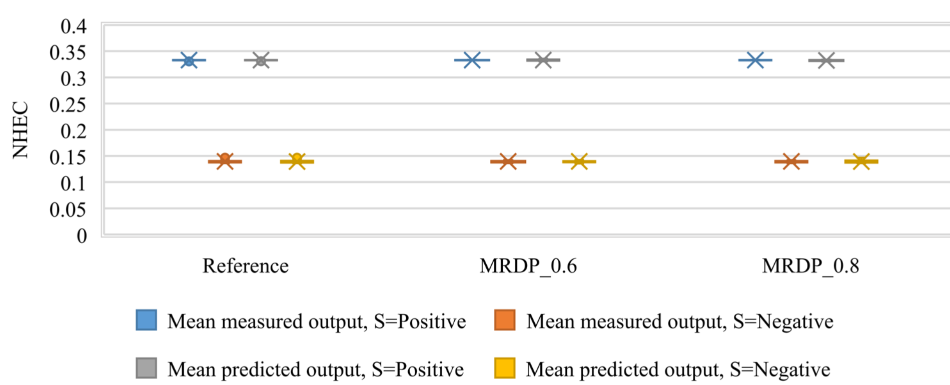
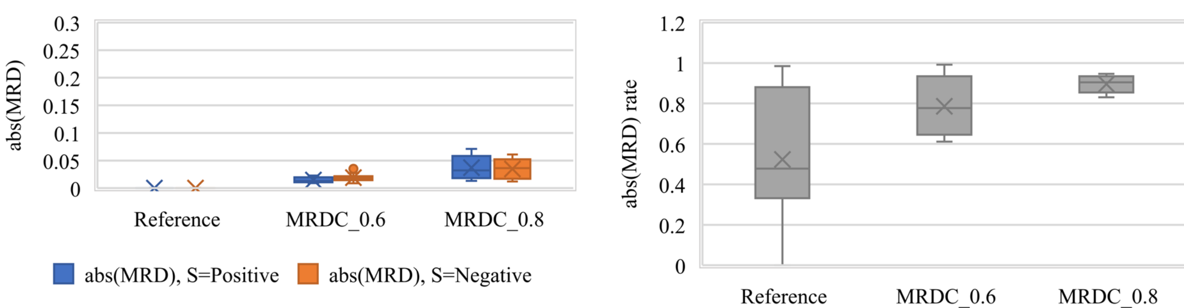


Figure 5: Mean measured NHEC and mean predicted NHEC for conditions S=Positive and S=Negative during model training

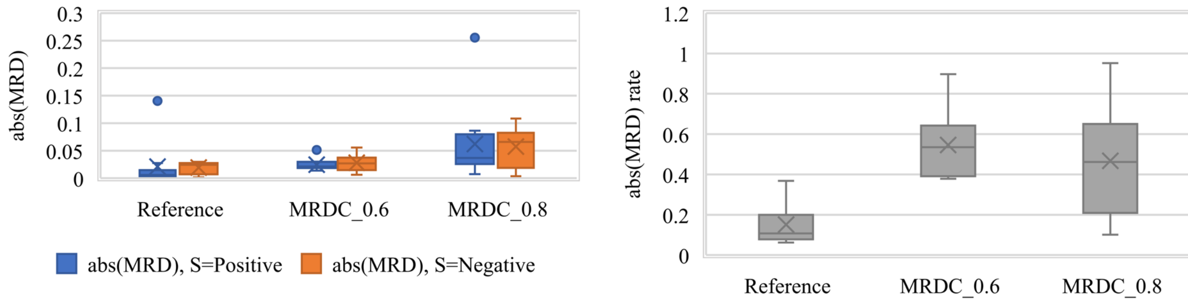
Besides, from Figure 4(a), during model training, MSEP with λ=0.6 could effectively make the 1-MSE to be similar to the 0-MSE. However, the predictive accuracy would be decreased as the overall MSE is increased from 0.01 to 0.016. Increasing λ from 0.6 to 0.8 would not significantly contribute to the similarity of between 1-MSE and 0-MSE. However, the overall predictive accuracy of MSEP with λ=0.8 is slightly worse than MSEP with λ=0.6. Furthermore, Figure 4(b) shows that the effect of MSEP on increasing the similarity between 1-MSE and 0-MSE could not be generalized to validation data.

Figure 4 also shows that MRDC with p=0.6 shows a slight effect on MSE, while MRDC with p=0.8 would significantly increase MSE_TOTAL, 1-MSE and 0-MSE. However, MRDC could not narrow the difference between 1-MSE and 0-MSE. This is because MRDC is aimed at making the abs(MRD) to be similar enough between S=Positive and S=Negative, instead of increasing the similarity in terms of MSE. Increasing p value for MRDC could effectively increase the fairness in terms of abs(MRD) rate during model training (see Figure 6(b)), however, the predictive accuracy would be decreased as the abs(MRD) would be increased (see Figure 6(a)). Even if increasing p value could increase the abs(MRD) rate, this pattern is not generalizable during model validation (see Figure 7). This problem may be caused by nonconvergence of the optimization algorithm when p=0.8. It could be solved by increasing the maximum iteration number of DE.



(a) (b)

Figure 6: Effect of MRDC on the (a) abs(MRD) and (b) abs(MRD) rate during model training

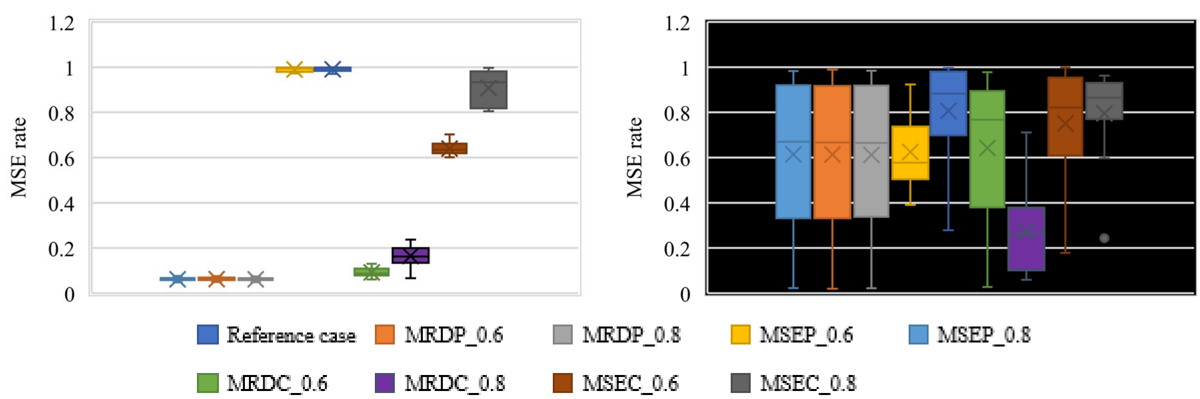


(a) (b)

Figure 7: Effect of MRDC on the (a) abs(MRD) and (b) abs(MRD) rate during model validation

Moreover, increasing p value of MSEC would increase the MSE (see Figure 4), however, the difference between 1-MSE and 0-MSE would be decreased. Figure 8 shows the effect of MSEC on the *Type II* fairness improvement: MSEC with p=0.6 could increase the MSE rate to be higher than 0.6 no matter during model training or model validation, while MSEC with p=0.8 could ensure the MSE rate to be higher than 0.8.

To improve *Type II* fairness in terms of having a high MSE rate, MSEP and MSEC would be suitable solutions. However, from Figure 8, MSEC shows better generalizability on the validation dataset. As Figure 8 shows MRDP does not affect the MSE rate. Although MRDC_0.6 shows a small MSE rate improvement ability during model training and validation, MRDC_0.8 significantly decreases the MSE rate from 0.61 to 0.27 during model validation.

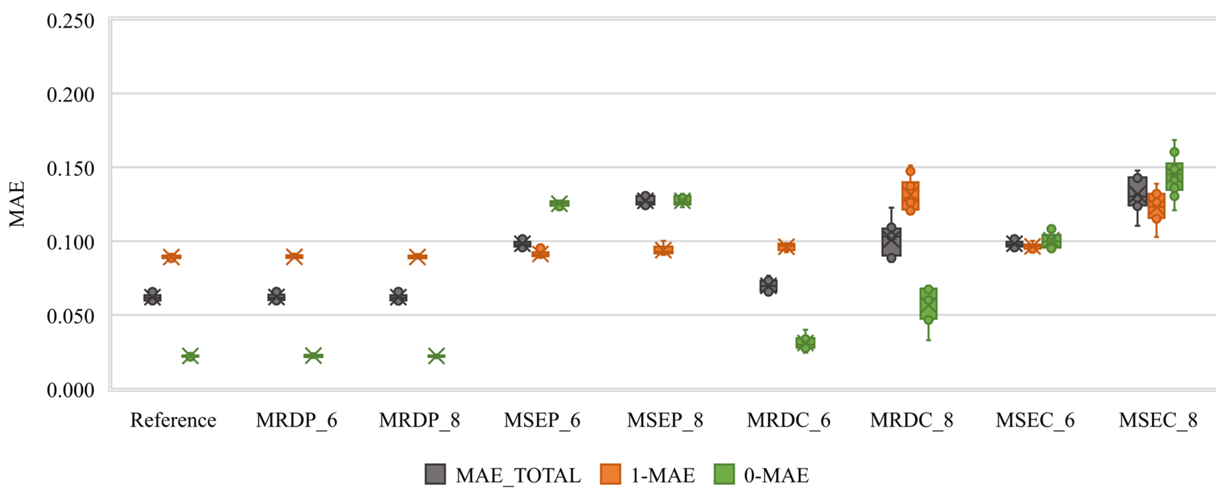


(a) (b)

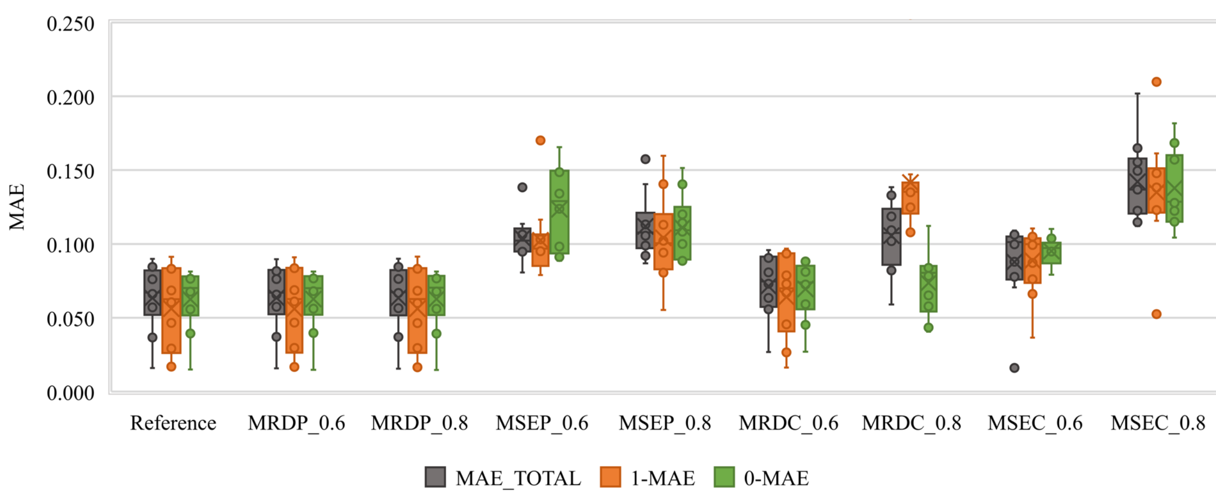
Figure 8: Effect of in-processing methods on the predictive fairness in terms of MSE rate during (a) model training and (b) model validation

3.2. Accuracy in terms of MAE and fairness in terms of MAE rate

As illustrated in Section 3.1, the regularizer added by MRDP is always almost equal to zero in this case study. Therefore, MRDP also does not present any effect on the MAE, as shown in Figure 9. Besides, MSEP with λ=0.6 could effectively decrease the difference between 1-MAE and 0-MAE during model training. Increasing λ from 0.6 to 0.8 would not further decrease the difference during model training, but the difference would be decreased during validation. Furthermore, increasing λ for MSEP would decrease the predictive accuracy because the overall MAE is increased.



(a) Model training

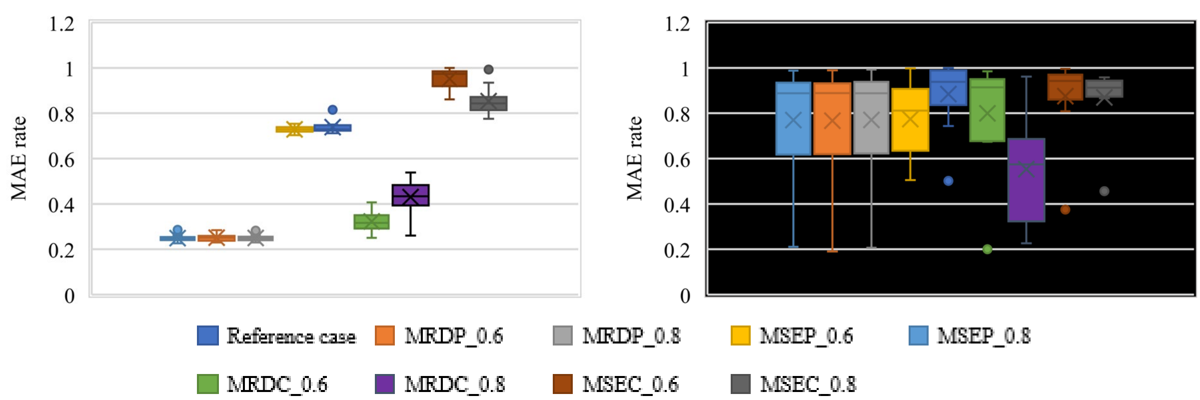


(b) Model validation

Figure 9: Effect of in-processing methods on the predictive accuracy in terms of MAE during (a) model training and (b) model validation

From Figure 9, MRDC could not decrease the difference between 1-MSE and 0-MSE, although the overall accuracy is decreased. However, MSEC with p=0.6 would effectively decrease the difference between 1-MAE and 0-MAE from ~0.067 to ~0.004 during model training. Increasing the p value would not contribute more to improving the similarity between 1-MAE and 0-MAE, but would show further harm to the overall predictive accuracy in terms of MAE.

The fairness improvement ability in terms of MAE rate is compared between these in-processing methods and is presented in Figure 10. Even if the reference case shows a MAE rate lower than 0.25 during model training, its MAE rate could reach 0.77 during model validation. MRDP does not affect the MAE rate during model training, while other in-processing methods would increase the MAE rate. Among them, MRDC with p=0.6 could slightly increase the average MAE rate during model training to 0.32, while MRDC with p=0.8 could increase this value to 0.43. However, even if MRDC_0.6 shows a slight increase on the MAE rate during model validation, MRDC_0.8 would significantly decrease it. MSEP could increase the MAE rate to be ~0.73 during model training, no matter λ=0.6 or λ=0.8. However, a higher λ value shows better MAE rate during validation. MSEC shows the best effect on increasing MAE rate, however, it shows the different pattern with MSE rate: increasing p value from 0.6 to 0.8 would not further improve the MAE rate.



(a) Model training (b) Model validation

Figure 10: Effect of in-processing methods on the predictive fairness in terms of MAE rate during (a) model training and (b) model validation

4. Discussion

4.1. Effect of *Loss_ori* selection

In this study MSE was selected as *Loss_ori* for in-processing methods, when MAE was another candidate. The reasons behind this selection include 1) MSE would be more efficient in term of computation time because the quadratic function of MSE makes it easier to find the gradient or the direction in which the value of loss function decreases. This reason is proofed through comparing the training time between using MSE and MAE as the loss function: the average runtime for using MSE is ~3,400s, while using MAE makes the runtime increase to ~3,600s; 2) MSE might be more powerful in predicting NHEC values that do not occur frequently in the training dataset. As illustrated in Section 2.1, MSE is more sensitive to outliers but MAE is more robust to outliers. It is because the square part of MSE makes it bigger than MAE when predicting an outlier as the same value. In other words, MSE tries harder to correctly predict unusual values. In the collected dataset, NHEC is lower than 0.7 most of the time (as shown in Figure 3), and high NHEC may be treated as outliers during model training, although it is not the case. Therefore, MSE is selected as *Loss_ori* to ensure the predictive accuracy for high NHEC values that are not common in the dataset and suffers a risk of considering as outliers by the data-driven model.

Nonetheless, when comparing the predicted NHEC and measured NHEC for linear regression models using MSE or MAE as the loss function in Figure 11, it is hard to conclude the better loss function. Both of them are likely to under-predict the NHEC when the corresponding ground truth value is higher than 0.7. More effective loss function that gives more weights to the unusual scenarios is still required to predict high NHEC. Further predictive performance comparison between MSE and MAE loss functions could be found in Table 4. It shows that using MSE as the loss function has higher predictive accuracy in terms of MSE, while selecting MAE as the loss function could ensure a lower predictive error in terms of MAE.

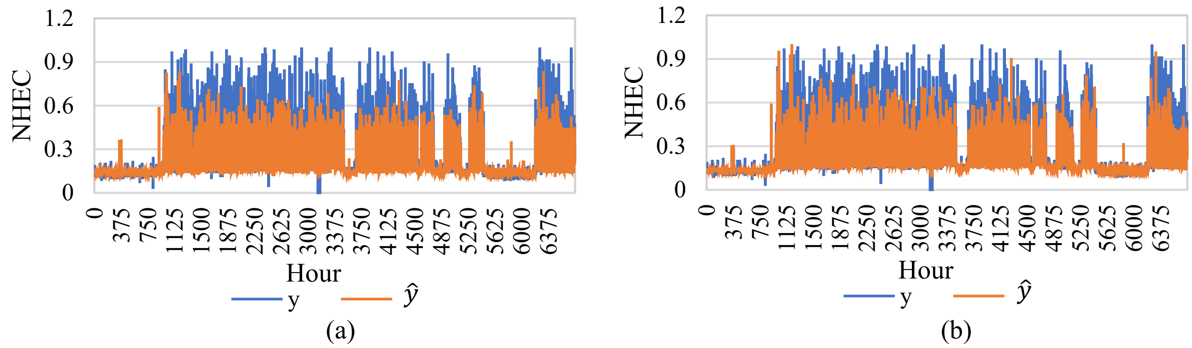


Figure 11: Comparison between y and $\hat{y}$ when using (1) MSE or (2) MAE as the loss function

Table 4: Predictive accuracy when using MSE or MAE as the loss function

		Performance criteria	
		MSE	MAE
Loss function	MSE	0.0109	0.0634
	MAE	0.0114	0.0605

4.2. Effect of optimization algorithms

Although DE is powerful in solving loss functions of the proposed in-processing algorithms, there are other optimization algorithms that may work well on these constrained optimization problems. For example, genetic algorithm (GA) is a commonly used derivative-free optimization algorithms in building engineering domain. It has been used to do optimal design [43], optimal control [44,45], and predictive model training [46,47], etc.. Therefore, in this section, the runtime and predictive accuracy of reference cases with MSE as the loss function would be compared between DE and GA.

GA is a metaheuristic inspired by the process of natural selection that selects the fittest individual to produce the next generation [48]. It could solve both constrained and unconstrained optimization problems, even if their objective function is discontinuous, nondifferentiable, stochastic, or nonlinear [49]. The general procedure of a basic GA is presented in Figure 12. Note that unlike DE, crossover is processed before mutation in GA. Besides, in the selection step, GA selects two fittest solutions based on their fitness scores, while DE selects a set of parents. Further, GA mutates new offspring based on a probability distribution to maintain the diversity within the population, while the mutation in DE is processed to create a unit vector based on the differential vector and target vector.

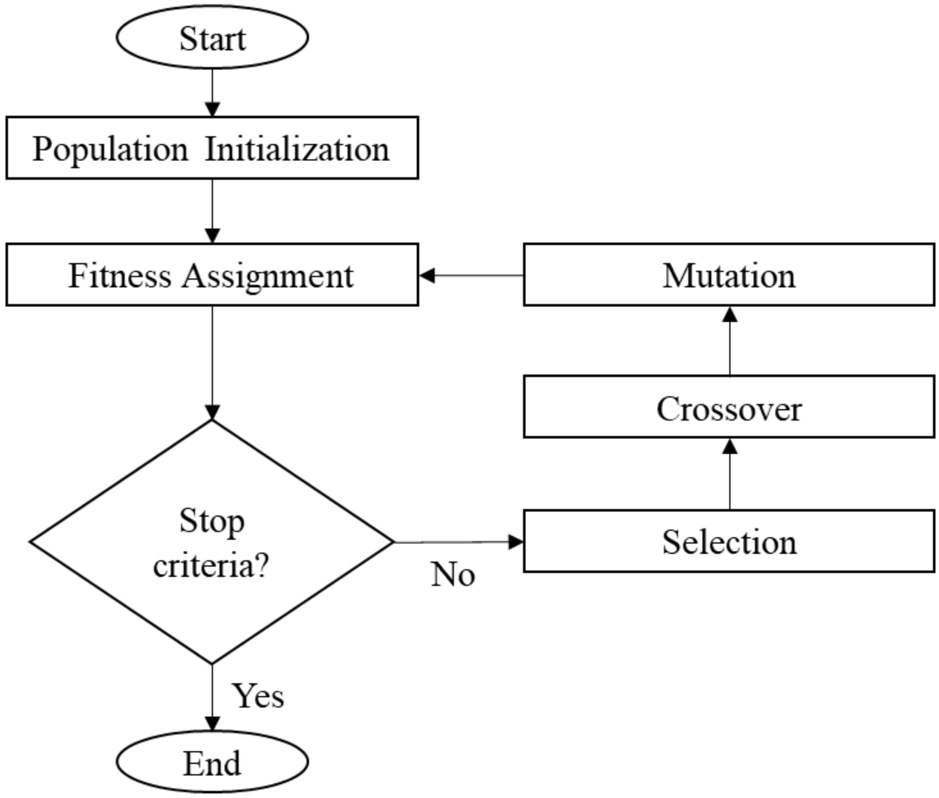


Figure 12: General procedure of a basic GA

In this section, DE and GA have the same population size and maximum iteration time. Their predictive accuracy for the reference case is compared in Table 5. It shows that DE always show a better accuracy than GA no matter in terms of MSE or MAE. However, GA is much faster than DE, as training time of each fold in GA is ~1,700s while DE needs ~3,400s. Therefore, GA would be recommended to solve the optimization problem during model training if the runtime is an important factor.

Table 5: Predictive accuracy comparison between DE and GA

		Model training		Model validation	
		MSE	MAE	MSE	MAE
Optimization algorithm	DE	0.0107	0.0620	0.0109	0.0634
	GA	0.0109	0.0639	0.0112	0.0647

5. Conclusion

To improve predictive fairness of regression DDBMs to have uniform predictive accuracy between different conditions, this study proposed four in-processing methods—MRDP, MSEP, MRDC, MSEC—to achieve the user-defined trade-off between predictive fairness and overall accuracy. The fundamental of these methods is to set fairness-related penalties or constraints in the objective function of model training.

A case study was done to apply these in-processing methods to develop linear regression models for the energy prediction of an apartment. The effect of p/λ values of these methods on the predictive accuracy and fairness were investigated. Conclusions draw from this case study include:

- MRDP would not affect the predictive result, because the mean predicted values are almost equal to the mean measured values under the same condition (S=Positive or S=Negative).
- MSEP with λ=0.6 could significantly decrease the accuracy difference between the situation that S= Positive and the situation with S=Negative. Increasing λ from 0.6 to 0.8 for MSEP would not narrow the accuracy difference too much, but it would decrease the overall accuracy.
- MRDC does not present the ability to decrease the accuracy (MAE or MSE) difference between different conditions defined by the protected attribute. However, it works good on increasing the similarity of abs(MRD) between S=Positive and S=Negative.
- MSEC could decrease the difference between 1-MSE and 0-MSE. However, MSEC with p=0.6 results in competitive 1-MAE and 0-MAE similarity compared to MSEC with p=0.8. The overall predictive accuracy in terms of MSE and MAE would be decreased when increase the p value.
- MSEC is the most powerful in-processing methods to improve *Type II* fairness in terms of MSE rate and MAE rate. Besides, MSEP is another good option. It shows better performance on preserving the overall predictive accuracy than MSEC. However, MRDC with a high p value could even destroy the fairness.

As the proposed methods show different effects on the accuracy and fairness, researchers are recommended to select proper methods based on their research objectives. For example, if improving the MSE rate is the main concern, MSEC would be the best option; if the main objective is to improve fairness to have a high abs(MRD) rate, MRDC could be selected. Furthermore, this study shows some drawbacks: 1) Linear regression models are relatively simple compared with other regression models, such as deep learning. The simple structure of linear regression makes it hard to provide high predictive accuracy. Therefore, in the future, integrating the proposed in-processing methods into more complex and powerful data-driven models would be an interesting topic. 2) Finding fast and effective optimization algorithms to solve complex objective functions caused by integrating in-processing methods would be a potential research direction, as the shorted runtime would make the fairness-aware regression models applicable to develop model predictive controllers.

Acknowledgments

The authors would like to express their gratitude to Concordia University for the support through the Concordia Research Chair in Energy & Environment.

Abbreviations

abs(MRD)	Absolute Mean Residual Difference
AHUs	Air Handling Units
CV(RMSE)	Coefficient of Variation of the Root Mean Square Error
DDBMs	Data-Driven Buildings Models
DE	Differential Evolution
DT	Decision Tree
FDD	Fault Detection and Diagnosis
GA	Genetic Algorithm
GAN	Generative Adversarial Network
HVAC	Heating Ventilation and Air-Conditioning
HEMS	Home Energy Management System
kNN	k-Nearest-Neighbor
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLP	Multi-Layer Perceptron
MRDC	Mean Residual Difference Constrained Regression

MRDP	Mean Residual Difference Penalized Regression
MSE	Mean Square Error
MSEC	Mean Square Error Constrained Regression
MSEP	Mean Square Error Penalized Regression
NHEC	Normalized Hourly Energy Consumption
PSO	Particle Swarm Optimization
R^2	R Square
RF	Random Forest
RMSE	Root Mean Square Error
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine

Nomenclature

$Loss$	Loss function
$Loss\_ori$	The original loss function without considering fairness
p	Weights in MRDC and MSEC. It infers the similarity of predictive performance among different conditions defined by the protected attribute
r_{xy}	Pearson correlation coefficient between input feature x and target output y
S	Protected attribute
$s0$	The number of training data with S = Negative
$s1$	The number of training data with S = Positive
w_0	Bias term
w	Weight matrix
x	Input feature
$\bar{x}$	Mean value of input feature
y	Measured value
$\bar{y}$	Mean value of measured value
$\hat{y}$	Predicted value
λ	Multiplier of the prejudice remover regularizer (MRDP and MSEP)

Reference:

[1] Li B, Cheng F, Cai H, Zhang X, Cai W. A semi-supervised approach to fault detection and diagnosis for building HVAC systems based on the modified generative adversarial network. Energy Build 2021;246:111044. https://doi.org/10.1016/j.enbuild.2021.111044.

[2] Sun Y, Joybari MM, Panchabikesan K, Moreau A, Robichaud M, Haghighat F. Heating demand and indoor air temperature prediction in a residential building using physical and statistical models: a comparative study. IOP Conf Ser Mater Sci Eng 2019;609:072022. https://doi.org/10.1088/1757-899X/609/7/072022.

[3] Wei W, Ramalho O, Malingre L, Sivanantham S, Little JC, Mandin C. Machine learning and statistical models for predicting indoor air quality. Indoor Air 2019;29:704–26. https://doi.org/10.1111/ina.12580.

[4] Wang Y, Li W, Zhang Z, Shi J, Chen J. Performance evaluation and prediction for electric vehicle heat pump using machine learning method. Appl Therm Eng 2019:113901.

[5] Mirnaghi MS, Haghighat F. Fault detection and diagnosis of large-scale HVAC systems in buildings using data-driven methods: A comprehensive review. Energy Build 2020;229:110492. https://doi.org/10.1016/j.enbuild.2020.110492.

[6] Adams S, Greenspan S, Velez-Rojas M, Mankovski S, Beling PA. Data-driven simulation for energy consumption estimation in a smart home. Environ Syst Decis 2019:1–14.

[7] Ahmad AS, Hassan MY, Abdullah MP, Rahman HA, Hussin F, Abdullah H, et al. A review on applications of ANN and SVM for building electrical energy consumption forecasting. Renew Sustain Energy Rev 2014;33:102–9. https://doi.org/10.1016/j.rser.2014.01.069.

[8] Afram A, Janabi-Sharifi F, Fung AS, Raahemifar K. Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system. Energy Build 2017;141:96–113.

[9] Wang Z, Hong T, Li H, Ann Piette M. Predicting city-scale daily electricity consumption using data-driven models. Adv Appl Energy 2021;2:100025. https://doi.org/10.1016/j.adapen.2021.100025.

[10] Ahmad T, Chen H. Nonlinear autoregressive and random forest approaches to forecasting electricity load for utility energy management systems. Sustain Cities Soc 2019;45:460–73. https://doi.org/10.1016/j.scs.2018.12.013.

[11] Sun Y, Haghighat F, Fung BC. A review of the-state-of-the-art in data-driven approaches for building energy prediction. Energy Build 2020;221:110022.

[12] Sun Y, Haghighat F, Fung BC. Trade-off Between Accuracy and Fairness of Data-driven Building and Indoor Environment Models: A Comparative Study of Pre-processing Methods. Energy 2021:122273. https://doi.org/10.1016/j.energy.2021.122273.

[13] Kleinberg J, Mullainathan S, Raghavan M. Inherent trade-offs in the fair determination of risk scores. ArXiv Prepr ArXiv160905807 2016.

[14] Fan Y, Cui X, Han H, Lu H. Chiller fault diagnosis with field sensors using the technology of imbalanced data. Appl Therm Eng 2019;159:113933. https://doi.org/10.1016/j.applthermaleng.2019.113933.

[15] Zhou Z, Chen H, Li G, Zhong H, Zhang M, Wu J. Data-driven fault diagnosis for residential variable refrigerant flow system on imbalanced data environments. Int J Refrig 2021;125:34–43. https://doi.org/10.1016/j.ijrefrig.2021.01.009.

[16] Kamiran F, Calders T. Data preprocessing techniques for classification without discrimination. Knowl Inf Syst 2012;33:1–33. https://doi.org/10.1007/s10115-011-0463-8.

[17] Sun Y, Fung BCM, Haghighat F. The generalizability of pre-processing techniques on the accuracy and fairness of data-driven building models: A case study. Energy Build 2022;268:112204. https://doi.org/10.1016/j.enbuild.2022.112204.
[18] Yan K, Huang J, Shen W, Ji Z. Unsupervised learning for fault detection and diagnosis of air handling units. Energy Build 2020;210:109689.
[19] Yan K, Chong A, Mo Y. Generative adversarial network for fault detection diagnosis of chillers. Build Environ 2020;172:106698. https://doi.org/10.1016/j.buildenv.2020.106698.
[20] Disadvantages of GANs || Am I real or a Trained Model to write? OpenGenus IQ Comput Expert Leg 2020. https://iq.opengenus.org/disadvantages-of-gans/ (accessed November 4, 2021).
[21] Mary J, Calauzènes C, Karoui NE. Fairness-Aware Learning for Continuous Attributes and Treatments. Proc. 36th Int. Conf. Mach. Learn., PMLR; 2019, p. 4382–91.
[22] Zafar MB, Valera I, Rogriguez MG, Gummadi KP. Fairness constraints: Mechanisms for fair classification. Artif. Intell. Stat., PMLR; 2017, p. 962–70.
[23] Berk R, Heidari H, Jabbari S, Joseph M, Kearns M, Morgenstern J, et al. A convex framework for fair regression. ArXiv Prepr ArXiv170602409 2017.
[24] Wadsworth C, Vera F, Piech C. Achieving fairness through adversarial learning: an application to recidivism prediction. ArXiv Prepr ArXiv180700199 2018.
[25] Kamishima T, Akaho S, Sakuma J. Fairness-aware Learning through Regularization Approach. 2011 IEEE 11th Int. Conf. Data Min. Workshop, 2011, p. 643–50. https://doi.org/10.1109/ICDMW.2011.83.
[26] Zhang BH, Lemoine B, Mitchell M. Mitigating unwanted biases with adversarial learning. Proc. 2018 AAAIACM Conf. AI Ethics Soc., 2018, p. 335–40.
[27] Fitzsimons J, Al Ali A, Osborne M, Roberts S. A general framework for fair regression. Entropy 2019;21:741.
[28] Olfat M, Mintz Y. Flexible Regularization Approaches for Fairness in Deep Learning. 2020 59th IEEE Conf. Decis. Control CDC, IEEE; 2020, p. 3389–94.
[29] Li Z, Zhu H, Ding Y, Xu X, Weng B. Establishment of a personalized occupant behavior identification model for occupant-centric buildings by considering cost sensitivity. Energy Build 2020;225:110300. https://doi.org/10.1016/j.enbuild.2020.110300.
[30] Tang M, Chen Y, Wu H, Zhao Q, Long W, Sheng VS, et al. Cost-Sensitive Extremely Randomized Trees Algorithm for Online Fault Detection of Wind Turbine Generators. Front Energy Res 2021;9.
[31] Qu Z, Liu H, Wang Z, Xu J, Zhang P, Zeng H. A combined genetic optimization with AdaBoost ensemble model for anomaly detection in buildings electricity consumption. Energy Build 2021;248:111193. https://doi.org/10.1016/j.enbuild.2021.111193.
[32] Li K, Zhou G, Zhai J, Li F, Shao M. Improved PSO_AdaBoost Ensemble Algorithm for Imbalanced Data. Sensors 2019;19:1476. https://doi.org/10.3390/s19061476.
[33] Biddle D. Adverse Impact and Test Validation: A Practitioner's Guide to Valid and Defensible Employment Testing. 2 edition. Aldershot, Hampshire, England : Burlington, VT: Routledge; 2006.
[34] Conn AR, Scheinberg K, Vicente LN. Introduction to derivative-free optimization. SIAM; 2009.
[35] Georgioudakis M, Plevris V. A Comparative Study of Differential Evolution Variants in Constrained Structural Optimization. Front Built Environ 2020;6.
[36] Storn R, Price K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. J Glob Optim 1997;11:341–59. https://doi.org/10.1023/A:1008202821328.
[37] Li J, Panchabikesan K, Yu Z, Haghighat F, Mankibi ME, Corgier D. Systematic data mining-based framework to discover potential energy waste patterns in residential buildings. Energy Build 2019;199:562–78. https://doi.org/10.1016/j.enbuild.2019.07.032.
[38] Khani SM, Haghighat F, Panchabikesan K, Ashouri M. Extracting energy-related knowledge from mining occupants' behavioral data in residential buildings. J Build Eng 2021;39:102319.

[39] Panchabikesan K, Haghighat F, Mankibi ME. Data driven occupancy information for energy simulation and energy use assessment in residential buildings. Energy 2021;218:119539. https://doi.org/10.1016/j.energy.2020.119539.
[40] Yang B, Haghighat F, Fung BCM, Panchabikesan K. Season-Based Occupancy Prediction in Residential Buildings Using Machine Learning Models. E-Prime - Adv Electr Eng Electron Energy 2021;1:100003. https://doi.org/10.1016/j.prime.2021.100003.
[41] Esrafilian-Najafabadi M, Haghighat F. Impact of predictor variables on the performance of future occupancy prediction: Feature selection using genetic algorithms and machine learning. Build Environ 2022;219:109152. https://doi.org/10.1016/j.buildenv.2022.109152.
[42] Fei Guo. scikit-opt. 2022.
[43] Tuhus-Dubrow D, Krarti M. Genetic-algorithm based approach to optimize building envelope design for residential buildings. Build Environ 2010;45:1574–81. https://doi.org/10.1016/j.buildenv.2010.01.005.
[44] Mossolly M, Ghali K, Ghaddar N. Optimal control strategy for a multi-zone air conditioning system using a genetic algorithm. Energy 2009;34:58–66.
[45] Wang S, Jin X. Model-based optimal control of VAV air-conditioning system using genetic algorithm. Build Environ 2000;35:471–87.
[46] Luo XJ, Oyedele LO, Ajayi AO, Akinade OO, Owolabi HA, Ahmed A. Feature extraction and genetic algorithm enhanced adaptive deep neural network for energy consumption prediction in buildings. Renew Sustain Energy Rev 2020;131:109980.
[47] Le LT, Nguyen H, Dou J, Zhou J. A comparative study of PSO-ANN, GA-ANN, ICA-ANN, and ABC-ANN in estimating the heating load of buildings’ energy efficiency for smart city planning. Appl Sci 2019;9:2630.
[48] Whitley D. A genetic algorithm tutorial. Stat Comput 1994;4:65–85. https://doi.org/10.1007/BF00175354.
[49] What Is the Genetic Algorithm? - MATLAB & Simulink n.d. https://www.mathworks.com/help/gads/what-is-the-genetic-algorithm.html (accessed June 3, 2022).

Supplementary Information:

Table S1: Candidate input features

Input feature	Input feature	Input feature	Input feature	Input feature	Input feature
Day of the week	Irradiance shadow band correction factor	Set-point temperature for heater 3	Blind position 1	Slat angle position 8	NHEC t-9
Time of the day	Direct horizontal irradiance	Set-point temperature for heater 4	Blind position 2	Slat angle position 9	NHEC t-10
Sun altitude	Global horizontal UVA irradiance	Set-point temperature for heater 5	Blind position 3	Slat angle position 10	NHEC t-11
Sun azimuth	Global horizontal UVB irradiance	Set-point temperature for heater 6	Blind position 4	Window opening status 1	NHEC t-12
Global horizontal illuminance	CO_2 concentration 1	Motion status 1	Blind position 5	Window opening status 2	NHEC t-13
Diffuse horizontal illuminance	CO_2 concentration 2	Motion status 2	Blind position 6	Window opening status 3	NHEC t-14
Global vertical north illuminance	CO_2 concentration 3	Motion status 3	Blind position 7	Window opening status 4	NHEC t-15
Global vertical east illuminance	CO_2 concentration 4	Motion status 4	Blind position 8	Window opening status 5	NHEC t-16
Global vertical south illuminance	Indoor temperature 1	Motion status 5	Blind position 9	Window opening status 6	NHEC t-17
Global vertical west illuminance	Indoor temperature 2	Motion status 6	Blind position 10	Window opening status 7	NHEC t-18
Global horizontal irradiance	Indoor temperature 3	Motion status 7	Slat angle position 1	NHEC t-1	NHEC t-19
Giffuse horizontal irradiance	Indoor temperature 4	Motion status 8	Slat angle position 2	NHEC t-2	NHEC t-20
Zenith luminance	Indoor relative humidity 1	Motion status 10	Slat angle position 3	NHEC t-3	NHEC t-21
Relative humidity	Indoor relative humidity 2	Motion status 11	Slat angle position 4	NHEC t-4	NHEC t-22
Wind direction	Indoor relative humidity 3	Motion status 12	Slat angle position 5	NHEC t-5	NHEC t-23
Wind speed	Indoor relative humidity 4	Motion status 13	Slat angle position 6	NHEC t-6	NHEC t-24
Dry bulb temperature	Set-point temperature for heater 1	Motion status 14	Slat angle position 7	NHEC t-7	NHEC t-168
Illuminance shadow band correction factor	Set-point temperature for heater 2	Motion status_Total		NHEC t-8	

Note that the number after the name of CO_2 concentration, indoor temperature, indoor relative humidity, set-point temperature for heaters, motion status, blind position, slat angle position, and window opening status means the corresponding measurement device, while the time index after NHEC illustrates the normalized hourly energy consumption at the corresponding time, t is the current time.