Trade-off Between Accuracy and Fairness of Data-driven Building and Indoor Environment Models: A Comparative Study of Pre-processing Methods

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

[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

Official publication: https://doi.org/10.1016/j.energy.2021.122273

Abstract

Data-driven models have drawn extensive attention in the building domain in recent years, and their predictive accuracy depends on features or data distribution. Accuracy variation among users or periods creates a certain unfairness to some users. This paper addresses a new research problem called fairness-aware prediction of data-driven building and indoor environment models. First, three types of fairness definitions are introduced in building engineering. Next, *Type I* and *Type II* fairness are investigated. To achieve fairness *Type I*, we study the effect of suppressing the protected attribute (i.e., attribute whose value cannot be disclosed or be discriminated against) from inputs. To improve fairness *Type II* while preserving the predictive accuracy of data-driven building and indoor environment models, we propose three pre-processing methods for training dataset—sequential sampling, reversed preferential sampling, and sequential preferential sampling. The proposed methods are compared to two existing pre-processing methods in a case study for lighting status prediction in an apartment building. Overall, 576 study cases were used to study the effect of these pre-processing methods on the accuracy and fairness of 12 series of lighting status prediction based on 2 types of feature combinations and 4 types of classifiers. Predictive results show that suppressing the protected attribute slightly influences overall predictive accuracy, while all pre-processing methods decrease it. However, in general, sequential sampling would be a good option for improving fairness *Type II* with an acceptable accuracy decrease. Fairness improvement performance of other pre-processing methods varies depending on applied features and classifiers.

Keywords: Fairness; Accuracy; Machine learning; Data-driven model; Building and indoor environment; Privacy

Corresponding Author*: Fariborz.Haghighat@concordia.ca

1. Introduction

1.1. Motivation for investigating the accuracy and fairness of data-driven building and indoor environment models

As smart home energy management systems (HEMS) and sensors are rapidly gaining popularity, an abundance of information (indoor temperature, humidity, motion status, CO_2 concentration, and energy consumption, etc.) could be dynamically collected from buildings [1]. This has increased attention to developing data-driven prediction models for: indoor air temperature [2], building energy consumption [3–5], occupants' thermal comfort [6], occupancy status/numbers [7–9], indoor air quality [10], or heating ventilation and air-conditioning (HVAC) system performance [11]. These models could be further integrated into a model predictive controller (MPC) that can predefine optimal control signals based on users' demand, and control relevant devices via HEMS to achieve cost/energy saving, or peak shifting [12,13].

Existing building and indoor environment models were mostly evaluated or validated by accuracy measures to show the performance gap that infers the differences between predicted values and measured/simulated values [14–16]. For data-driven buildings models, commonly used accuracy measures could be classified for different types of data-driven models. On the one hand, for regression models that predict continuous quantity outputs, the following measures could be used: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Bias Error (MBE), Normalized MBE (NMBE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R Square (R^2). On the other hand, for classification models whose predictive outputs are discrete class labels, accuracy, precision, recall, sensitivity, specificity, and F1 score could be selected. Detailed reviews of data-driven building and indoor environment models and their accuracy measures can be found in [17–21].

Note that the predictive accuracy of data-driven models highly depends on a representative training dataset with properly selected features [22]. Commonly used input features for data-driven building and indoor environment models mainly include meteorological information, indoor environmental parameters, occupancy related data, time index, building characteristic data, socio-economic information, and historical data [21]. However, the available features could differ among

users due to differences in applied data collection devices or different opinions on information sharing. For example, newly constructed smart buildings are more likely to be equipped with additional data collection devices than old ones. In addition, some occupants may approve data-driven predictors to use their collected data for prediction, while others might deny it due to privacy concerns. Note that one common potential privacy issue in the building and indoor environment domain is the collection and use of occupancy-related data (e.g., number of occupants, motion status, types of occupant activities, etc.) as previous studies mention it might be used to infer occupants' individual location and behavior [23–25]. Thus, occupants should have the option to deny information disclosure. As a result, data-driven predictors often yield more accurate predictions for users who provide further information by installing more sensors or who are willing to provide extra private information. It is unfair for users who want to protect their privacy to have less accurate predictions.

Furthermore, the predictive performance of data-driven building and indoor environment models also relies on balanced data [26]. In reality, it may be challenging to collect a balanced training dataset. For instance, the collected training data from a HVAC system may contain a large number of normal data but a small portion of faulty data. In that case, the developed data-driven model would show an accurate prediction for a normal status but a poor result for faulty conditions [27]. Besides, when collecting data for energy prediction, data distribution varies for different users. If the user is retired and stays at home most of time, data would be collected mostly during occupied time. If the occupant is working during daytime, data would be distributed more evenly between occupied and unoccupied time. As a result, predictors could be more accurate during certain periods of time than others due to the higher volume of training data [28]. The relatively poor predictive performance during some periods may further reduce cost-saving potential when integrating the predictor into an MPC. It is unfair for users to lose cost-saving potential due to the intermittent poor performance.

However, the above-mentioned fairness problems (e.g., predictive performance diversity caused by the difference in available/authorized features and unbalanced data) in data-driven building and indoor environment models have rarely attracted attention. Therefore, the concept of fairness is introduced in Section 1.2. Then, accuracy and fairness of data-driven models are demonstrated via application to a detailed monitored apartment building.

1.2. Background

In general, fairness improvement studies could be categorized into three categories based on the type of fairness they achieved [29]:

Type I. The prediction provided by the developed model, or the decision made based on its output is independent of the protected attribute(s). Note that protected attributes are defined as attributes whose values cannot be disclosed [30] or attributes that cannot be discriminated against [31]. Commonly used protected attributes include gender, race, and sexual orientation, etc. For building models, occupancy-related data could be selected as protected attributes, because occupants may deny usage of this information as input features due to privacy concern. This type of fairness could be evaluated by two ways: 1) The protected attribute is discarded (e.g., occupancy-related data is excluded from inputs of building models by occupants); 2) The predictive outcomes/values are similar for instances that have different values for the protected attribute(s) but the same value(s) for the unprotected attribute(s). One example could be given for non-discriminatory hiring: If two people come from different racial groups (i.e., protected attribute for this example) but their unprotected features (e.g., educational background and technical abilities) are the same, they should have the same opportunity to get that job.

Type II. Some given measures of predictive performance (e.g., accuracy) are equal across groups/conditions defined by the protected attribute(s). For example, to achieve this type of fairness when predicting energy consumption using occupancy status as the protected attribute, accuracy should be similar in both cases (occupied/unoccupied). This definition of fairness is more applicable when predictive performance is the main concern and protected attributes are not expected to affect it.

Type III. Predictive outcomes should be independent of the predictive probability score of input data in different groups/conditions defined by the protected attribute(s). Note that probability score is generalized by probabilistic classifiers (such as logistic regression and Naïve Bayes) to present probability distribution over a set of classifiers [32]. Here is an example related to non-discriminatory hiring: if two people have different protected features and unprotected features but get the same probability score of being employed from the trained probabilistic classifier, their admission result should be the same.

Examples for *Type I* and *Type II* indicate the aspiration of considering fairness in the building and indoor environment domain. In fact, fairness-aware machine learning has drawn increasing attention in recent years and has been applied to non-discriminatory hiring [33–35], risk assessment for sentencing guidance [36,37], income prediction [37], loan allocation [38,39], and graph embedding [40]. However, it has not yet been used to investigate fairness problems among data-driven models in building and indoor environment given the gap between disciplines.

Fairness-aware data-driven building and indoor environment models are worth investigating for the following benefits:

1) Enabling authority management and privacy protection. Considering data privacy, some occupants do not want to contribute person-specific (or family-specific) information and occupancy-related information to predictors. Well-designed fairness-aware machine learning procedures in terms of *Type I* could avoid unauthorized private information usage.

2) Ensuring uniform predictive performance among different groups defined by protected attributes. A related example has been given above when defining *Type II* fairness: predicting energy consumption with occupancy status as the protected attribute. Having similar energy predictive accuracy when the building is occupied or unoccupied could ensure good predictor performance every time.

3) Preserving fairness for different users. For instance, users from different buildings usually show different habits and opinions for submitting sensory data to develop data-driven models. Thus, predictive performance could vary for different users as the available dataset for model training varies. Considering fairness among these developed models could ensure similar predictive performance; thus, all users could get similar service provided by these predictors.

Commonly used fairness improvement methods can be classified into three categories: pre-processing, in-processing, and post-processing. Among them, pre-processing removes discrimination from training data before model training. In-processing methods add fairness-related constraints or penalties to the optimization objective function during model training [41]. Post-processing is a method that modifies prediction results of a classifier to achieve fairness. Pre-processing is more applicable when more than one type of data-driven model is implemented.

Further, training data collected or processed by the fairness-aware procedure has been reported as the most important place to improve fairness in industrial product development [42].

The easiest pre-processing method to achieve fairness *Type I* is to suppress protected attributes from input features [43]. However, this method cannot ensure discrimination decrease in training dataset and predictive results, and might significantly decrease predictive accuracy when protected attributes and outputs are highly correlated. Feldman et al. [44] proposed a repair approach to change unprotected attributes to mask bias while preserving relevant data information. In their study, fairness is evaluated based on predictability of protected attributes versus unprotected attributes.

To improve fairness *Type II*, pre-processing approaches attempting to sample balanced data in different conditions defined by protected attributes and predictive outputs could be considered. For instance, to omit bias among the training dataset for a two-class classification problem with one binary protected attribute, Kamiran and Calders [43] proposed uniform sampling and preferential sampling methods that balance data amount in different conditions. When multi-variate non-binary protected variables are defined, an optimized data transformation procedure proposed by Calmon et al. [37] could be used to improve fairness with acceptable classification accuracy decrease. Furthermore, in building and indoor environment, some methods have been proposed in recent years to sample a balanced training dataset. For instance, Zhang et al. [28] proposed a clustering decision tree algorithm to identify building operation conditions and then randomly duplicate or remove data from conditions with more or less data than expected. Results show that increasing the training sample proportion of a condition would decrease the MAE of building energy load prediction under that condition. Yan et al. [45] applied the generative adversarial network algorithms to generate an additional artificial fault training dataset before training classifiers for fault detection and chiller diagnosis. The rebalanced training dataset significantly increases fault detection classification accuracy. However, these studies have never been applied to solve fairness problems.

1.3. Objective, contribution, and structure of this study

The main objective of this study is to introduce and investigate the fairness concept in the building and indoor environment domain with minimal accuracy decrease. To be more specific,

this research is intended to study accuracy and fairness (*Type I* or *Type II*) in data-driven building and indoor environment models by suppressing the protected attribute from inputs or by eliminating discrimination in training dataset via pre-processing methods. This study presents three pre-processing methods—sequential sampling, reversed preferential sampling, and sequential preferential sampling. These methods are meant to transform candidate training dataset into a balanced training dataset, thereby ensuring that developed data-driven models perform well in different situations defined by protected attributes (i.e., improve fairness *Type II*). These proposed methods are compared to two existing pre-processing methods—uniform sampling and preferential sampling—in a case study investigating their effect on accuracy and fairness of data-driven building and indoor environment models.

Therefore, the primary role of this study is to investigate the trade-off between fairness and accuracy of data-driven building and indoor environment models. It is the first work to introduce the fairness concept into the building and indoor environment domain. Additionally, this study proposes three pre-processing methods to improve fairness while preserving predictive accuracy of data-driven models. The proposed pre-processing methods are initially designed to process the training dataset of a two-class classification problem with a binary protected attribute. However, these pre-processing methods could extend to all predictive problems whose output and protected attributes could be converted into discrete class labels. Furthermore, these methods could be applied to solve fairness problems in data-driven models for any building type (e.g., commercial and residential).

This paper unfolds into several segments. Section 2 introduces the pre-processing methods studied along with accuracy and fairness measures. Section 3 illustrates the application of these pre-processing methods through a case study, along with its collected data and a description of 576 study cases. Section 4 presents results in terms of accuracy measures and fairness measures. To explain results, data distribution change under different pre-processing methods is discussed in Section 5. Finally, Section 6 concludes this study and lists future research avenues.

2. Methodology

The studied or proposed pre-processing methods are introduced in Section 2.1. Then, in Section 2.2, accuracy measures (to evaluate closeness of predicted values to measured values) and

fairness measures (to reflect the predictive accuracy difference between different situations grouped by the protected attribute) for two-class classification problems are explained.

2.1. Pre-processing methods

This section presents five types of pre-processing methods—uniform sampling, sequential sampling, preferential sampling, reversed preferential sampling, and sequential preferential sampling. These methods process the original training data (i.e., candidate training dataset denoted by $X_{candidate}$) and produce a designed training dataset ($X_{designed}$). Among them, uniform sampling and preferential sampling are proposed by Kamiran and Calders [43], while others are proposed by the authors. Note that all these methods are designed to obtain a more balanced training dataset for two-class classification problems with a binary protected attribute.

2.1.1. Uniform sampling

In uniform sampling, each data point in $X_{candidate}$ has the same chance to be duplicated or removed. The procedure is as follows:

First, partition the candidate training set into four groups: 1) PP (*P*ositive protected attribute and *P*ositive actual class label); 2) PN (*P*ositive protected attribute and *N*egative actual class label); 3) NP (*N*egative protected attribute and *P*ositive actual class label), and 4) NN (*N*egative protected attribute and *N*egative actual class label) based on Equations 1–4. The number of training points in these groups are represented by |PP|, |PN|, |NP|, and |NN|, respectively.

$$\mathrm{PP} := \{\mathrm{S} = \text{Positive and } \mathrm{Y} = \text{Positive}\} \quad (1)$$

$$\mathrm{PN} := \{\mathrm{S} = \text{Positive and } \mathrm{Y} = \text{Negative}\} \quad (2)$$

$$\mathrm{NP} := \{\mathrm{S} = \text{Negative and } \mathrm{Y} = \text{Positive}\} \quad (3)$$

$$\mathrm{NN} := \{\mathrm{S} = \text{Negative and } \mathrm{Y} = \text{Negative}\} \quad (4)$$

where S is the protected attribute, and Y is the class label of the training point.

Next, calculate the expected number of training points for each group to sample a balanced training dataset. As the research objective is eliminating bias from protected attributes and target class, the expected data distribution in $X_{designed}$ should be: $Pr_{exp}(\mathrm{S} = \text{Positive}) = Pr_{exp}(\mathrm{S} = \text{Negative})$

$= 0.5$, and $Pr_{exp}(\text{Y} = \text{Positive}) = Pr_{exp}(\text{Y} = \text{Negative}) = 0.5$, where Pr_{exp} means the expected possibility of one point belonging to a specified group. S and Y should be statistically independent, which means $Pr_{exp}(\text{S} = \text{s}, \text{Y} = \text{y}) = Pr_{exp}(\text{S} = \text{s}) * Pr_{exp}(\text{Y} = \text{y}) = 0.25$, where $\text{s}, \text{y} \in [\text{Negative}, \text{Positive}]$. Therefore, the expected number of points in PP, PN, NP, and NN of the designed training set, respectively, is calculated using Equation 5.

$$|\text{PP}|_{\text{design}} = |\text{PN}|_{\text{design}} = |\text{NP}|_{\text{design}} = |\text{NN}|_{\text{design}} = 0.25 * \left| \text{X}_{\text{designed}} \right| \quad (5)$$

Finally, sample $|\text{PP}|_{\text{design}}$, $|\text{PN}|_{\text{design}}$, $|\text{NP}|_{\text{design}}$, and $|\text{NN}|_{\text{design}}$ training points randomly from groups PP, PN, NP, and NN in $\text{X}_{\text{candidate}}$ to $\text{X}_{\text{designed}}$, respectively. When the actual points in one group are higher than the designed number, randomly slice the designed number of training points from that group to $\text{X}_{\text{designed}}$. On the other hand, when the number of actual points in one group is positive but less than the designed number, randomly duplicate points from this group until the designed number is reached, and then slice to $\text{X}_{\text{designed}}$. Furthermore, if one group is empty, its designed number of training points will be randomly sliced from the group with most training points.

A formal description of the uniform sampling method is shown in Algorithm 1, where <D, S, Y> denotes the elements of one training point and D represents the unprotected features.

Algorithm 1: Uniform Sampling

Algorithm 1: Uniform Sampling

Input: Candidate training set $X_{candidate}$<D, S, Y> and length of designed training set $|X_{designed}$<D, S, Y>|

Output: Designed training set $X_{designed}$<D, S, Y>

1. Partition PP, PN, NP, NN from $X_{candidate}$<D, S, Y>
2. Calculate $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, $|NN|_{design}$
3. **for** x **in** [PP, PN, NP, NN]:
4. **if** $|x| >= |x|_{design}$:
5. Randomly slice $|x|_{design}$ number of points in x to $X_{designed}$<D, S, Y>
6. **else if** |x| != 0:
7. Slice |x| number of points in x to $X_{designed}$<D, S, Y>
8. Randomly duplicate $|x|_{design}$-|x| number of points from x
9. Transfer the duplicated points to $X_{designed}$<D, S, Y>
10. **else:**
11. Randomly slice $|x|_{design}$ number of points in the group with max(|PP|, |PN|, |NP|, |NN|) to $X_{designed}$<D, S, Y>
12. **return** Designed training set $X_{designed}$<D, S, Y>

2.1.2. Sequential sampling

The sequential sampling method algorithm is listed in Algorithm 2. Its first two steps (i.e., partition groups and calculate designed number of points) are the same as the uniform sampling. However, in the final step, sequential sampling slices most recent training data from four groups in turns. This indicates that after sequential sampling, the number of points in some groups might not reach the designed number. However, sequential sampling could capture the latest information from $X_{candidate}$.

Algorithm 2: Sequential sampling

Algorithm 2: Sequential Sampling

Input: Candidate training set $X_{candidate}$<D, S, Y> and length of designed training set $|X_{designed}$<D, S, Y>|

Output: Designed training set $X_{designed}$<D, S, Y>

```
Partition PP, PN, NP, NN from X_candidate<D, S, Y>
Calculate |PP|_design, |PN|_design, |NP|_design, |NN|_design
[i_total, i_PP, i_PN, i_NP,i_NN] = 0
for n in range (0, |X_designed<D, S, Y>|):
    for x in [PP, PN, NP, NN]:
        if i_x >= |x| or i_total >= |X_designed<D, S, Y>|:
            continue
        Slice the i_x th most recent in x to X_designed<D, S, Y>
        i_x = i_x+1
        i_total = i_total+1
    if i_total >= |X_designed<D, S, Y>|:
        break
return Designed training set X_designed<D, S, Y>
```

2.1.3. Preferential sampling

Preferential sampling and uniform sampling differ as preferential sampling duplicates or removes data close to the decision boundary (a hypersurface dividing dataset into two classes) for each group. Preferential sampling is proposed because data points close to the decision boundary are more likely to be discriminated or favored [43]. Closeness to decision boundary is determined by a ranker (see Algorithm 3). In the ranker, a probabilistic classifier is first trained using $X_{candidate}$. In this study, Naïve Bayes is selected as the classifier in the ranker. Then, the trained classifier could return the possibility of classifying each training point as positive (p_positive) or negative (p_negative).

In preferential sampling (see Algorithm 4), training points in group PP and NP are placed in ascending order with respect to p_positive, while those in group PN and NN are placed in ascending order with respect to p_negative. The more front the training point is in each group, the closer it is to the decision boundary. Then, if the actual points in one group exceed the designed number, slice the last designed number of training points from that group to $X_{designed}$. When the number of actual points in one group is less than the designed number but higher than zero, duplicate points closest to the decision boundary until reaching the designed number, and then

slice these points to $X_{designed}$. Furthermore, if one group is empty, the designed number of training points for that group will be sliced in descending order from the group with the most training points.

Algorithm 3: Ranker

Algorithm 3: Ranker

Input: Candidate training set $X_{candidate}$<D, S, Y>

Output: Probability of each training point to be classified as Positive or Negative <p_positive, p_negative>

1. Training a model by using $X_{candidate}$<D, S, Y>
2. **for** x **in** $X_{candidate}$<D, S, Y>:
3. Calculate p_positive := $Pr(\hat{Y} = Positive \mid X=x)$
4. Calculate p_negative := $Pr(\hat{Y} = Negative \mid X=x)$
5. **return** <p_positive, p_negative> for each training point

Algorithm 4: Preferential sampling

Algorithm 4: Preferential Sampling

Input: Candidate training set $X_{candidate}$<D, S, Y> and length of designed training set $|X_{designed}$<D, S, Y>|

Output: Designed training set $X_{designed}$<D, S, Y>

1. Partition PP, PN, NP, NN from $X_{candidate}$<D, S, Y>
2. Calculate $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, $|NN|_{design}$
3. Learn a ranker by using $X_{candidate}$<D, S, Y>
4. **for** x **in** [PP, NP]:
5. Order training points in x ascending w.r.t. p_positive
6. **if** $|x| >= |x|_{design}$:
7. Slice the last $|x|_{design}$ number of points in x to $X_{designed}$<D, S, Y>
8. **else if** |x| != 0:
9. Slice |x| number of points in x to $X_{designed}$<D, S, Y>
10. Duplicate the first ($|x|_{design}$-|x|) number of points from x
11. Transfer the duplicated points to $X_{designed}$<D, S, Y>
12. **for** x **in** [PN, NN]:
13. Order training points in x ascending w.r.t. p_negative
14. **if** $|x| >= |x|_{design}$:
15. Slice the last $|x|_{design}$ number of points in x to $X_{designed}$<D, S, Y>
16. **else if** |x| != 0:
17. Slice |x| number of points in x to $X_{designed}$<D, S, Y>
18. Duplicate the first ($|x|_{design}$-|x|) number of points from x
19. Transfer the duplicated points to $X_{designed}$<D, S, Y>
20. **for** x **in** [PP, PN, NP, NN]:
21. **if** |x| == 0:
22. Slice the last $|x|_{design}$ number of points in the group with max(|PP|, |PN|, |NP|, |NN|) to $X_{designed}$<D, S, Y>
23. **return** Designed training set $X_{designed}$<D, S, Y>

2.1.4. Reversed preferential sampling

The reversed preferential sampling method algorithm is presented in Algorithm 5. The difference between reversed preferential sampling and preferential sampling is that when the actual points in one group are higher than the designed number, points furthest from the decision boundary would be removed by the reversed preferential sampling method. This sampling method

is proposed following the hypothesis that removing data close to the decision boundary might change the original decision boundary.

Algorithm 5: Reversed preferential sampling

Algorithm 5: Reversed preferential Sampling

Input: Candidate training set $X_{candidate}$<D, S, Y> and length of designed training set $|X_{designed}$<D, S, Y>|

Output: Designed training set $X_{designed}$<D, S, Y>

1. Partition PP, PN, NP, NN from $X_{candidate}$<D, S, Y>
2. Calculate $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, $|NN|_{design}$
3. Learn a ranker by using $X_{candidate}$<D, S, Y>
4. **for** x **in** [PP, NP]:
5. Order training points in x ascending w.r.t. p_positive
6. **if** $|x| >= |x|_{design}$:
7. Slice the first $|x|_{design}$ number of points in x to $X_{designed}$<D, S, Y>
8. **else if** |x| != 0**:**
9. Slice |x| number of points in x to $X_{designed}$<D, S, Y>
10. Duplicate the first ($|x|_{design}$ - |x|) number of points from x
11. Transfer the duplicated points to $X_{designed}$<D, S, Y>
12. **for** x **in** [PN, NN]:
13. Order training points in x ascending w.r.t. p_negative
14. **if** $|x| >= |x|_{design}$:
15. Slice the first $|x|_{design}$ number of points in x to $X_{designed}$<D, S, Y>
16. **else if** |x| != 0:
17. Slice |x| number of points in x to $X_{designed}$<D, S, Y>
18. Duplicate the first ($|x|_{design}$ - |x|) number of points from x
19. Transfer the duplicated points to $X_{designed}$<D, S, Y>
20. **for** x **in** [PP, PN, NP, NN]:
21. **if** |x| == 0:
22. Slice the first $|x|_{design}$ number of points in the group with max(|PP|, |PN|, |NP|, |NN|) to $X_{designed}$<D, S, Y>
23. **return** Designed training set $X_{designed}$<D, S, Y>

2.1.5. Sequential preferential sampling

Sequential preferential sampling (see Algorithm 6) slices training points close to the decision boundary from the four groups of $X_{candidate}$, one by one to $X_{designed}$. This indicates sequential preferential sampling will not duplicate the training points, and thus all sampled training points are the actual observed data.

Algorithm 6: Sequential preferential sampling

Algorithm 6: Sequential preferential Sampling

Input: Candidate training set $X_{candidate}$<D, S, Y> and length of designed training set $|X_{designed}$<D, S, Y>|

Output: Designed training set $X_{designed}$<D, S, Y>

1. Partition PP, PN, NP, NN from $X_{candidate}$<D, S, Y>
2. Calculate $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, $|NN|_{design}$
3. Learn a ranker by using $X_{candidate}$<D, S, Y>
4. **for** x **in** [PP, NP]:
5. Order training points in x ascending w.r.t. p_positive
6. **for** x **in** [PN, NN]:
7. Order training points in x ascending w.r.t. p_negative
8. [i_total, i_PP, i_PN, i_NP,i_NN] = 0
9. **for** n **in** range (0, $|X_{designed}$<D, S, Y>|):
10. **for** x **in** [PP, PN, NP, NN]:
11. **if** i_x >= |x| or i_total >= $|X_{designed}$<D, S, Y>|:
12. **continue**
13. Slice the i_x th number of points in x to $X_{designed}$<D, S, Y>
14. i_x = i_x+1
15. i_total = i_total+1
16. **if** i_total >= $|X_{designed}$<D, S, Y>|:
17. **break**
18. **return** Designed training set $X_{designed}$<D, S, Y>

To compare these five pre-processing methods, their procedures are summarized in Figure 1. The first two steps (i.e., partition data points into four groups and calculate designed number of samples for each group) are the same for all methods. In Step 3, uniform sampling randomly duplicates or removes data for each group until samples in the corresponding group reach the designed number. Sequential sampling lists data in each group chronologically (the earlier the data, the closer to the current time) and samples it from four groups in turns. Preferential sampling duplicates or removes data closer to the decision boundary. Reversed preferential sampling duplicates data closer to the decision boundary for groups without enough data, and removes data furthest from the decision boundary from groups with more samples than the designed number. Sequential preferential sampling lists data in each group in order of closeness to decision boundary, and samples data from these four groups in turns.

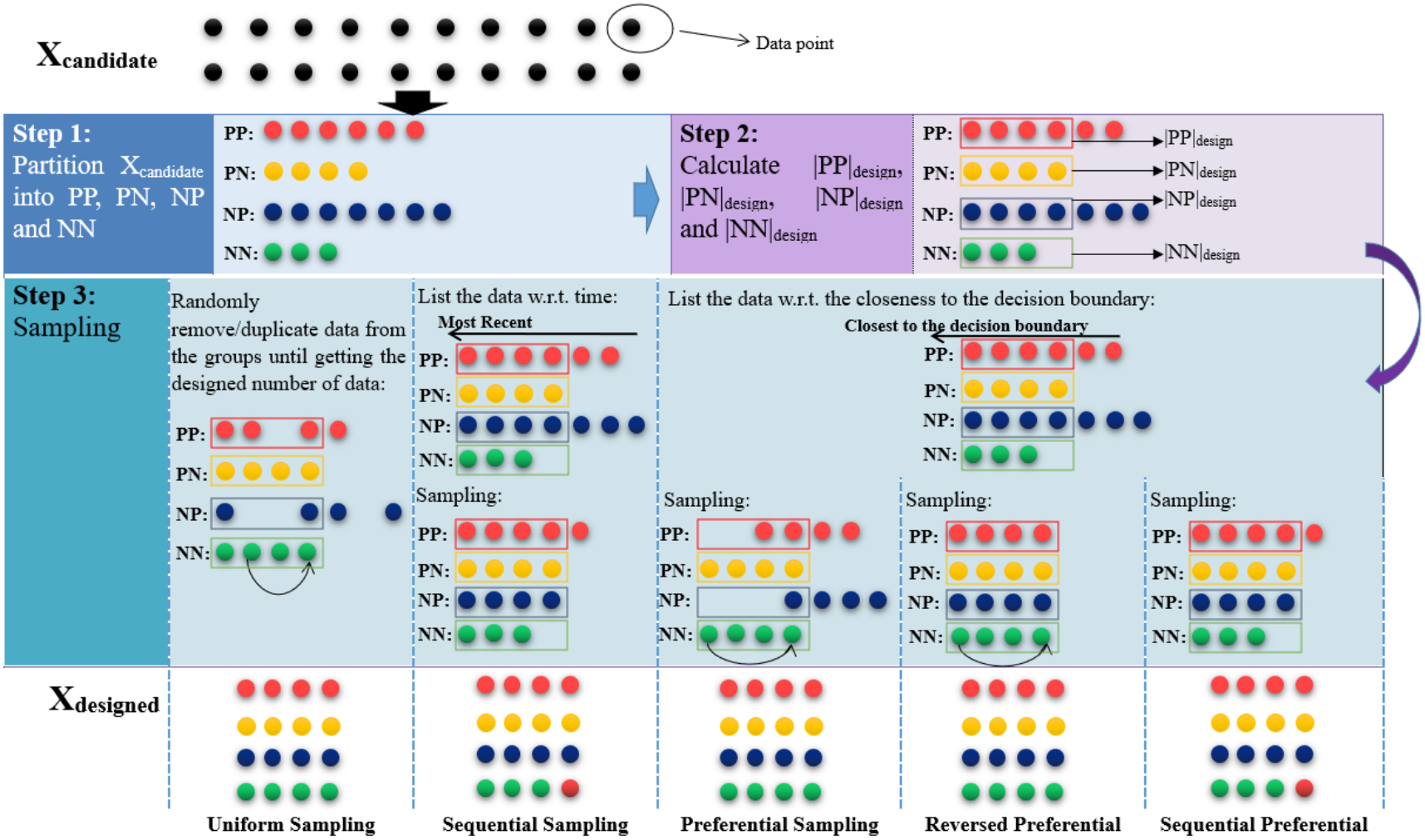


Figure 1: Pre-processing methods procedure

2.2. Accuracy and fairness measures

2.2.1. Accuracy measures

Accuracy, recall, and specificity (Equations 6–8) are commonly used accuracy measures for two-class classification problems. The meaning of TP, TN, FP, and FN in these equations are shown in Table 1.

Table 1: Confusion matrix

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Note: P = Positive; N = Negative; TP = True Positive; FP = False Positive; TN = True Negative; FN = False Negative

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \tag{6}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{FP} + \text{TN}} \tag{8}$$

Using these accuracy measures matters for three reasons. First, accuracy reflects the overall predictive accuracy of the data-driven model. Second, recall indicates the true positive rate, which is the proportion of correctly predicting the class label as positive when the actual class label is positive. For instance, in lighting status prediction, a small recall indicates lighting status is falsely predicted as OFF when it should be ON. Control actions (e.g., automatically turning OFF lighting or reducing electricity consumption for the user) based on this predictive signal would create poor user experience. And third, specificity (also called true negative rate) shows the ability of accurate prediction when the actual label is negative. For instance, in lighting status prediction, specificity calculates the rate of predicting as turning OFF lighting when the actual lighting status is OFF. Thus, if specificity is too low, the predictor would wrongly turn ON lighting when unnecessary. This would result in energy waste.

2.2.2. Fairness measures

Fairness measures could be classified into three broad categories according to the type of fairness to be evaluated [36]. Since not all fairness measures can be met simultaneously in most cases [46], researchers should select the ones relevant to research objective.

In this study, the highest concern is predictive accuracy in different situations defined by the protected attribute (denoted by S). It can be calculated by group conditional accuracy measures: c-Accuracy (Equation 9), c-Recall (Equation 10), and c-Specificity (Equation 11). Note that for S = Positive, its conditional accuracy measures are 1-Accuracy, 1-Recall, and 1-Specificity. These measures show predictive accuracy when the protected attribute label is Positive. On the other hand, for S = Negative, conditional accuracy measures are called 0-Accuracy, 0-Recall, and 0-Specificity. They reflect the predictive performance when the protected attribute is Negative.

$$\mathrm{c-Accuracy} = P\left[\hat{\mathrm{Y}} = \mathrm{y} \,\middle|\, \mathrm{Y} = \mathrm{y}, \mathrm{S} = \mathrm{s}\right] \tag{9}$$

$$\mathrm{c-Recall} = P\left[\hat{\mathrm{Y}} = \mathrm{Positive} \,\middle|\, \mathrm{Y} = \mathrm{Positive}, \mathrm{S} = \mathrm{s}\right] \tag{10}$$

$$\mathrm{c-Specificity} = P\left[\hat{\mathrm{Y}} = \mathrm{Negative} \,\middle|\, \mathrm{Y} = \mathrm{Negative}, \mathrm{S} = \mathrm{s}\right] \tag{11}$$

where c indicates the group conditional accuracy measures; $\mathrm{c} \in [0, 1]$, $\hat{\mathrm{Y}}$ means the predicted label, $\hat{\mathrm{Y}} \in [\mathrm{Negative}, \mathrm{Positive}]$.

Then, to evaluate fairness *Type II* by presenting the similarity of group conditional accuracy measures, accuracy rate (Equation 12), recall rate (Equation 13), and specificity rate (Equation 14) are selected fairness measures. They are the rates of minimum group conditional accuracy measures to the maximum group conditional accuracy measures. The higher the rates, the similar the predictive performance for Y in the situation when S = Positive and when S = Negative. When these rates exceed 80%, fairness *Type II* is achieved in terms of the "80 percent rule" [47]. This rule means the predictive result is fair when the predictive performance of any protected group is at least 80% of the highest predictive performance of these groups.

$$\mathrm{Accuracy\ rate} = \frac{\min(1 - \mathrm{Accuracy}, 0 - \mathrm{Accuracy})}{\max(1 - \mathrm{Accuracy}, 0 - \mathrm{Accuracy})} \tag{12}$$

$$\text{Recall rate} = \frac{\min(1 - \text{Recall}, 0 - \text{Recall})}{\max(1 - \text{Recall}, 0 - \text{Recall})} \quad (13)$$

$$\text{Specificity rate} = \frac{\min(1 - \text{Specificity}, 0 - \text{Specificity})}{\max(1 - \text{Specificity}, 0 - \text{Specificity})} \quad (14)$$

3. Case study

To demonstrate the proposed pre-processing methods and investigate their effect on the trade-off between accuracy and fairness for data-driven building models, study cases are designed to solve a two-class classification problem (i.e., lighting status prediction) with a binary protected attribute (i.e., motion status) for an apartment building. Data used for the case study is described in Section 3.1 and study cases are explained in Section 3.2.

3.1. Data description

Data in this study was collected from an apartment in a residential building in Lyon, France for the year 2016, with one-minute time intervals [25,48]. Weather information was processed from a local weather station in Vaulx-en-Velin, France. To increase the acceptable runtime for prediction (duration of a time step), and ensure representability of processed data, collected data was processed at 5-minute intervals. Missing data and outliers were processed by Li et al. [25].

Statistical distribution of the collected data is listed in Table 2. In this table, the numbering for motion status and lighting status represents the corresponding presence sensor and lighting sensor installed in the apartment. There are 14 presence sensors and 12 lighting sensors. Detailed information for the installed sensors is listed in

Table *3*. Note that the attribute 'Motion Status_total' in Table 2 represents the overall motion status in the studied apartment. It is recorded as ON if at least one presence sensor detected the motion status as ON at the same time.

Table 2: Statistical distribution of the collected data

Name	Type	Min/Least (No. points)	Max/Most (No. points)	Ave	Deviation
Motion Status No,1	Binominal	ON (4034)	OFF (87994)		
Motion Status No.2	Binominal	ON (8252)	OFF (83776)		
Motion Status No.3	Binominal	ON (10195)	OFF (81833)		
Motion Status No.4	Binominal	ON (715)	OFF (91313)		
Motion Status No.5	Binominal	ON (1299)	OFF (90729)		
Motion Status No.6	Binominal	ON (3061)	OFF (88967)		
Motion Status No.7	Binominal	ON (1487)	OFF (90541)		
Motion Status No.8	Binominal	ON (2406)	OFF (89622)		
Motion Status No.9	Binominal	ON (13650)	OFF (78378)		
Motion Status No.10	Binominal	ON (4750)	OFF (87278)		
Motion Status No.11	Binominal	ON (860)	OFF (91168)		
Motion Status No.12	Binominal	ON (4623)	OFF (87405)		
Motion Status No.13	Binominal	ON (233)	OFF (91795)		
Motion Status No.14	Binominal	ON (261)	OFF (91767)		
Motion Status_total	Binominal	ON (29041)	OFF (62986)		
Lighting Status No.1	Binominal	ON (95)	OFF (91933)		
Lighting Status No.2	Binominal	ON (9510)	OFF (82518)		
Lighting Status No.3	Binominal	ON (10946)	OFF (81082)		
Lighting Status No.4	Binominal	ON (315)	OFF (91713)		
Lighting Status No.5	Binominal	ON (12082)	OFF (79946)		
Lighting Status No.6	Binominal	ON (12942)	OFF (79086)		
Lighting Status No.7	Binominal	ON (1032)	OFF (90996)		
Lighting Status No.8	Binominal	ON (914)	OFF (91114)		
Lighting Status No.9	Binominal	ON (558)	OFF (91470)		
Lighting Status No.10	Binominal	ON (186)	OFF (91842)		
Lighting Status No.11	Binominal	ON (307)	OFF (91721)		
Lighting Status No.12	Binominal	ON (317)	OFF (91711)		
Global Horizontal Illuminance (lux)	Integer	-99	143800	13614	24420
Global Vertical North Illuminance (lux)	Integer	-99	28900	3075	4444
Global Vertical East Illuminance (lux)	Integer	-99	91900	7001	15587
Global Vertical South Illuminance (lux)	Integer	-99	105400	8774	17637
Global Vertical West Illuminance (lux)	Integer	-99	98700	5591	12259

Table 3: Description of sensors *[25]*

Sensor	Company name	Type	Accuracy
Presence Detector	Theben	PlanoCentro A-KNX	-(detection area 64 m^2 if seated)
Lighting Sensor	ABB	KNX Energy Module: EM/S 3.16.3	±2/3/6%

In this study, 'Motion Status_total' is the protected attribute, while Lighting Status No.1 to No.12 are classifier outputs. Therefore, Positive(ON)/Negative(OFF) protected attribute means motion status, while Positive(ON)/Negative(OFF) class label represents lighting status. The ratios of groups PP, PN, NP, and NN among the entire dataset for Lighting Status No.1 to No.12 are presented in Figure 2. For all lighting series, data is mainly distributed in groups NN (around 65% - 68%) and PN (around 21% - 32%).

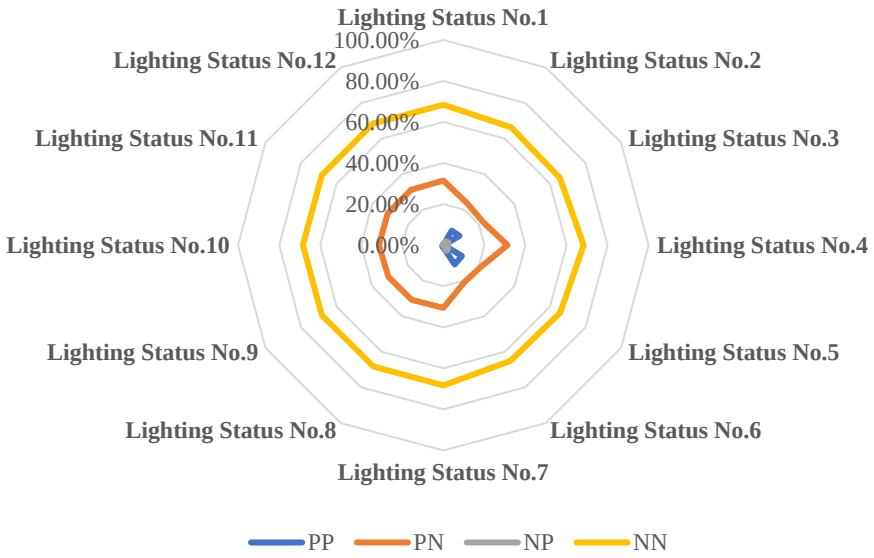


Figure 2: Ratios of PP, PN, NP, and NN for Lighting Status No.1 to No.12

3.2. Study cases

As Table 4 shows, 576 study cases were designed to compare the effects of 6 kinds of pre-processing methods (including the reference case) on the predictive accuracy and fairness of 12 series of lighting status (Lighting Status No.1 to No.12) under 2 types of input combinations (WithOccupancy or WithoutOccupancy) and 4 types of classifiers (i.e., Support Vector Machine (SVM), Artificial Neural Network (ANN), Logistic Regression, and Naïve Bayes). (6*12*2*4=576). Study cases are named by the utilized pre-processing method. Reference cases

refer to a situation with no pre-processing method implemented. Note these case studies consider 'Motion Status_total' as the protected attribute.

Table 4: Description of study cases

Case Name	Pre-processing methods	Inputs for training and prediction	Classifier
Reference Case		WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes
Uniform Sampling	Uniform Sampling	WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes
Sequential Sampling	Sequential Sampling	WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes
Preferential Sampling	Preferential Sampling	WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes
Reversed Preferential Sampling	Reversed Preferential Sampling	WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes
Sequential Preferential Sampling	Sequential Preferential Sampling	WithOccupancy (D', S')	SVM, ANN, Logistic Regression, Naïve Bayes
		WithoutOccpancy (D')	SVM, ANN, Logistic Regression, Naïve Bayes

Note: (1) D' = {Hour of The Day, Day of The Week, Global Horizontal Illuminance, Global Vertical North Illuminance, Global Vertical East Illuminance, Global Vertical South Illuminance, Global Vertical West Illuminance} (2) S' = {Motion Status No.1–No.14, Motion Status_total}

In reference cases, classifiers are trained by data from the previous four weeks, and then used for predicting next week's lighting status. Training data is updated weekly by newly observed data. Figure 3 shows the training and validation procedure for cases using pre-processing methods. First, the pre-processing strategy processes $X_{candidate}$ to produce the designed training set $X_{designed}$. In this study, $X_{designed}$ contains data for four weeks. Next, $X_{designed}$ is used to train the classification model. After, the trained classifier is used to predict lighting status ($\hat{Y}_{valid}$) one week ahead based on inputs extracted from next week's validation dataset X_{valid}. Then, X_{valid} and $X_{designed}$ are updated as $X_{candidate}$ every week, and the loop repeats until the procedure receives the 'stop' signal. In this study, the procedure stops after 41 prediction cycles (41 weeks).

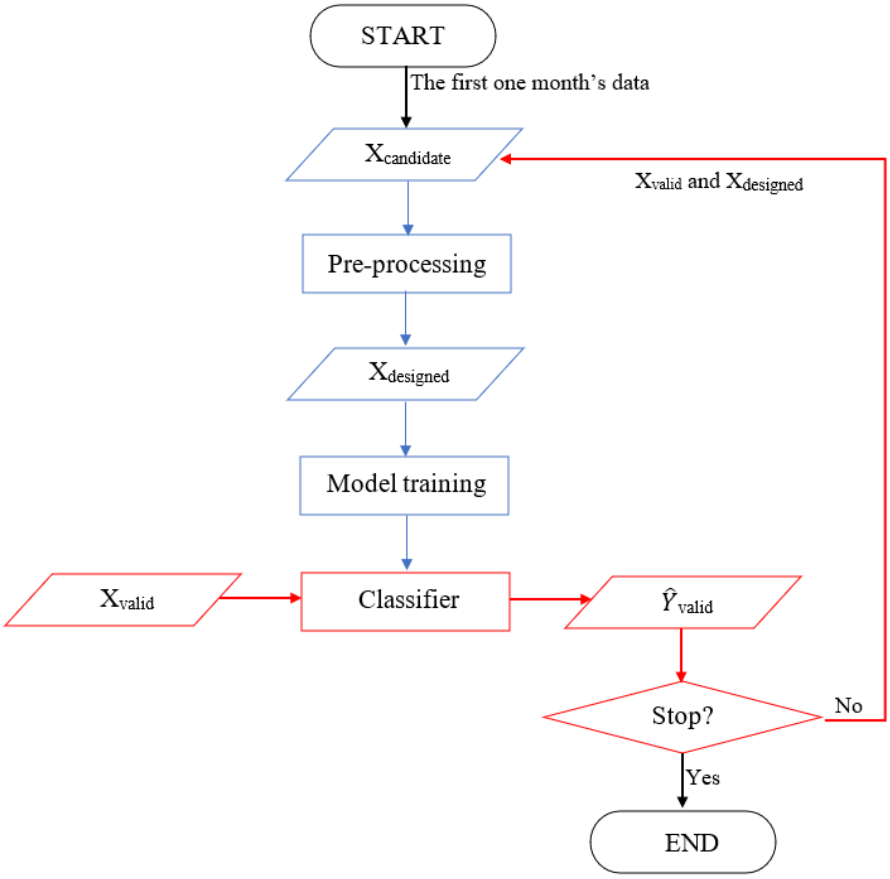


Figure 3: Training and validation procedure for cases using pre-processing methods

'WithOccupancy' means occupants give classifier permission to use occupancy-related data (motion status S') for prediction, while 'WithoutOccupancy' cases ban it. Comparing the effect of these two types of input combinations is done to investigate the possibility of achieving fairness *Type I*: The lighting status predictive result is independent of motion status.

Furthermore, four types of commonly used classifiers (SVM, ANN, Logistic Regression, and Naïve Bayes) are developed to study the robustness of pre-processing methods. These classifiers are of a different mathematical nature, but show good predictive performances when used for solving classification problems in building and indoor environment [19]. SVM predicts the class label by maximizing the margin between different categories [49]; it is not sensitive to noisy data. ANN usually consists of an input layer, several hidden layers, and an output layer. It predicts the output by learning the weight and bias of the activation functions in hidden layers and output layer. ANN is the basis for deep learning models [21]. Thus, studying the effect of pre-processing methods on ANN's predictive result could also reveal the potential applicability of pre-

processing methods in deep leaning models. Logistic Regression is popular for two-class classification problems. Its fundamental function is to predict the possibility of an object belonging to a positive class using a logistic function [50]. Naïve Bayes classifiers are a set of simple classifiers that apply Bayes' algorithm with the 'naive' assumption of conditional independence between attributes given the class label value [51]. Naïve Bayes classifiers include three main types: Gaussian Naïve Bayes, Multinomial Naïve Bayes, and Bernoulli Naïve Bayes. This study uses Gaussian Naïve Bayes.

Note that hyperparameters (i.e., pre-defined parameters before model training) in data-driven models could affect predictive performance [52]. Moreover, this case study is designed to compare the fairness improvement ability of pre-processing methods as well as their effect on predictive accuracy. Therefore, to avoid hyperparameter influence on results, they remain unchanged for the same classifier kind in different cases. A detailed description of hyperparameters of the four classifier types used here is in the supplementary information.

All cases are run by Python 3.7 on a laptop with Intel Core i7-7700HQ CPU @2.80GHz and 8GB of RAM.

4. Results

In this section, predictive results of cases are presented in terms of accuracy measures (Section 4.1) and fairness measures (Section 4.2).

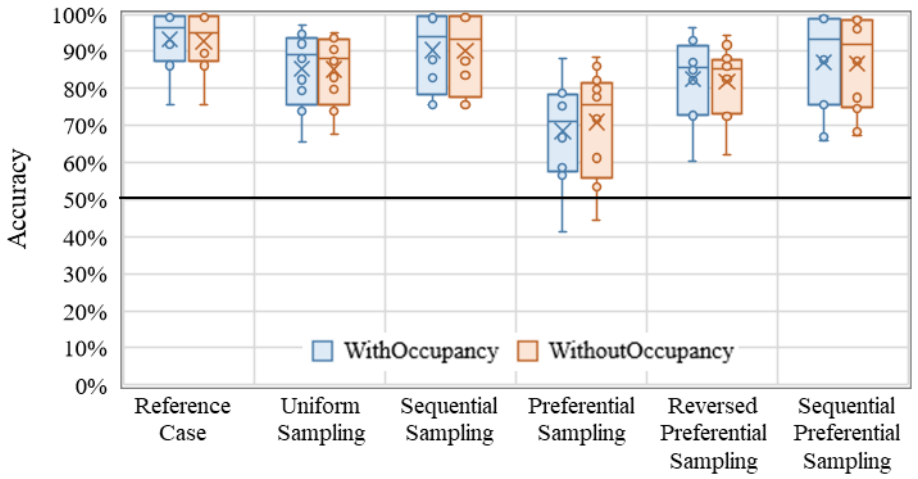
4.1. Results: accuracy measures

The overall predictive accuracy for Lighting Status No.1 to No.12 under different pre-processing strategies and classification models is statistically analyzed in Figure 4. Note that a circle point in this box and whisker plot represent overall predictive accuracy (y axis) for one type of lighting status after 41 weeks of prediction under the corresponding setting of input type (legend label), pre-processing method (x axis), and classification model (subfigure title). Thus, each box summarizes accuracy for 12 lighting status types. Figure 4 shows suppressing motion status (S') from input features influences predictive accuracy less than pre-processing strategies and classification methods. The difference of overall accuracy between cases using 'WithOccupancy' as inputs and cases simply using D' as inputs is negligible (less than 3% on average). This indicates

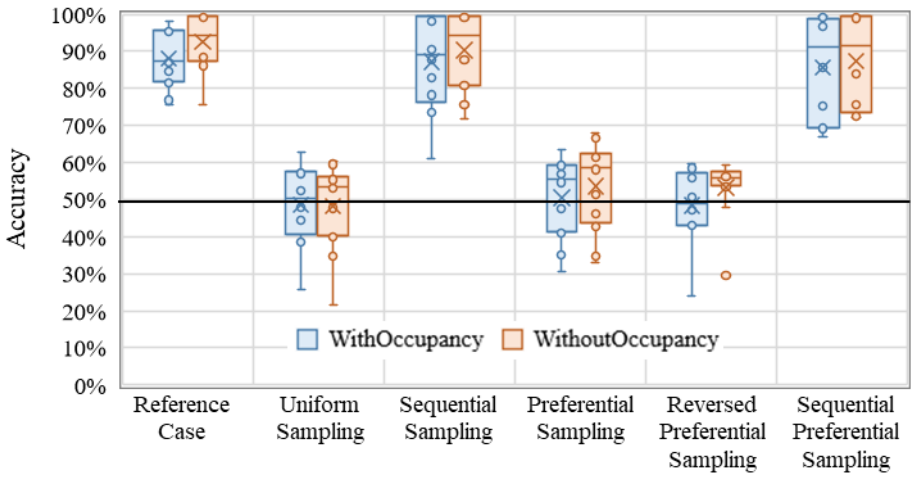
lighting status is independent from motion status for most lighting sensors in the apartment. Therefore, for these lighting sensors, lighting status prediction could be defined as fair in terms of *Type I*.

For all classification models, reference cases are more accurate. Sequential sampling and sequential preferential sampling strategies averagely decrease the overall accuracy by less than 5% for SVM, ANN, and Logistic Regression, and by around 10% for Naïve Bayes, while preferential sampling significantly decreases the average overall accuracy by over 35% for ANN and Logistic Regression, over 20% for SVM and 15% for Naïve Bayes on the average. The effect of uniform sampling and reversed preferential sampling on accuracy depends on classification methods. When using SVM, the mean accuracy is almost 85%. When using ANN, it drops to around 50%. As shown in Figure 4(a), reversed preferential sampling results in higher predictive accuracy for SVM than preferential sampling. Because SVM is meant to maximize the margin between different categories and is insensitive to data furthest from the decision boundary, hypothesis proposed in Section 2.1.4 could be verified: Sampling methods that remove data close to the decision boundary could change the original decision boundary more and cause poorer predictive accuracy. Moreover, the 50% accuracy line (expected accuracy of a random classifier) in each Figure 4 subfigure indicates the acceptable lower bound for all presented classifiers. Cases with accuracy lower than this line should be abandoned.

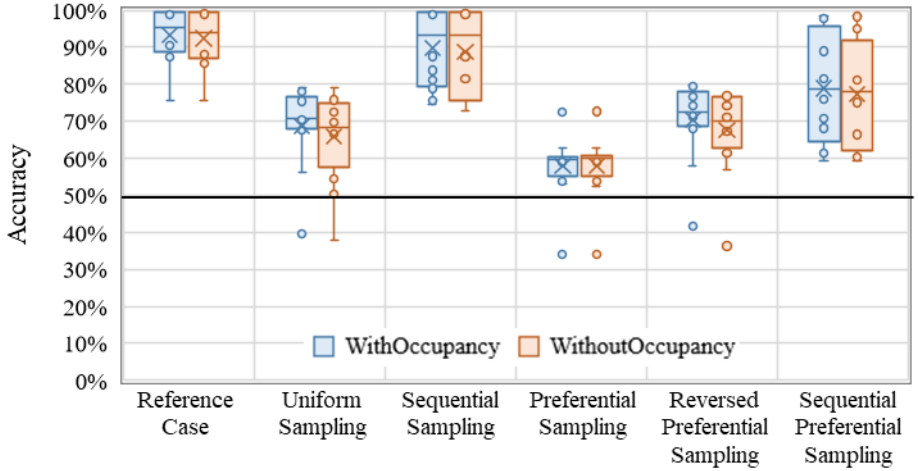
Further, in all cases the lowest point for each box (worst accuracy) in Figure 4 presents the accuracy for ‘Lighting Status No.1’. In Table 2, ‘Lighting Status No.1’ is OFF most of the time. This reveals that an unbalanced training dataset could cause worse predictive accuracy.



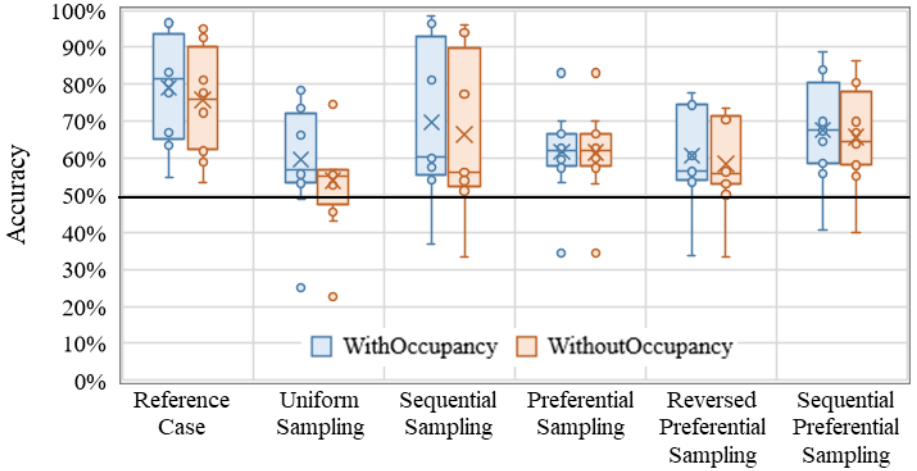
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

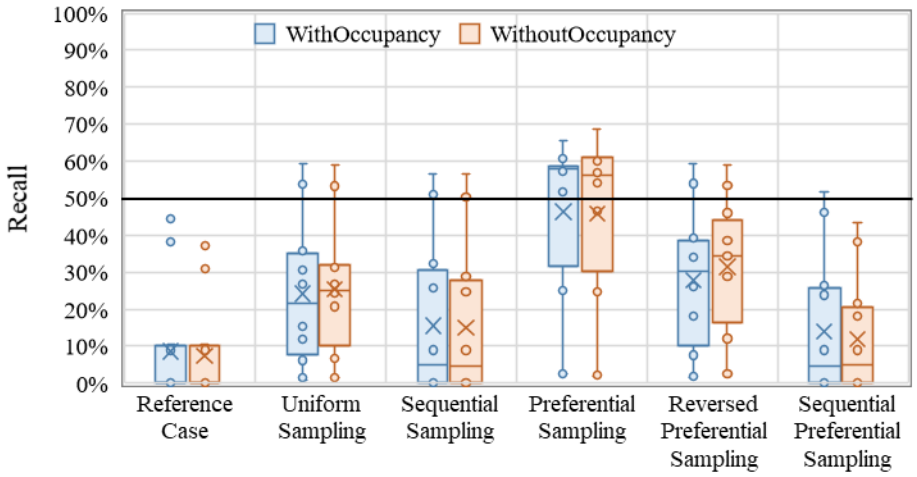
Figure 4: Accuracy under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

Figure 5 shows recall in different cases. Like Figure 4, recall differences between cases using ‘WithOccupancy’ or ‘WithoutOccupancy’ as inputs are ignorable. However, in contrast to Figure 4, the reference case presents the worst recall (less than 10% for most lighting status series) compared to other cases when SVM, ANN, and Logistic Regression are classifiers. When using these classifiers, sequential sampling and sequential preferential sampling show less recall improvement potential than uniform sampling, preferential sampling, and reversed preferential sampling. For the Naïve Bayes classifier, recall of sequential preferential sampling is even worse than the reference case. Overall, uniform sampling, preferential sampling, and reversed preferential sampling could effectively increase recall over 50% when using ANN, Logistic Regression, or Naïve Bayes as classifier.

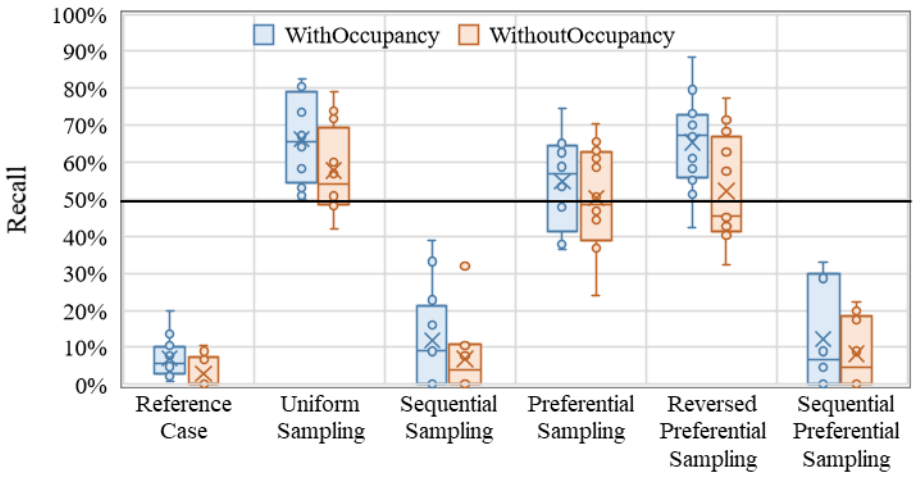
On the other hand, in Figure 5(a)–(c), the lowest point for each box usually presents recall for lighting that is OFF most of the time (e.g., ‘Lighting Status No.1’, ‘Lighting Status No.4’, ‘Lighting Status No.11’, or ‘Lighting Status No.12’). This result implies that recall (true positive rate) could be improved by increasing the number of training data with a positive observed class label.

Figure 6 shows specificity in different cases. For each case, specificity resembles accuracy as most observed target values are negative.

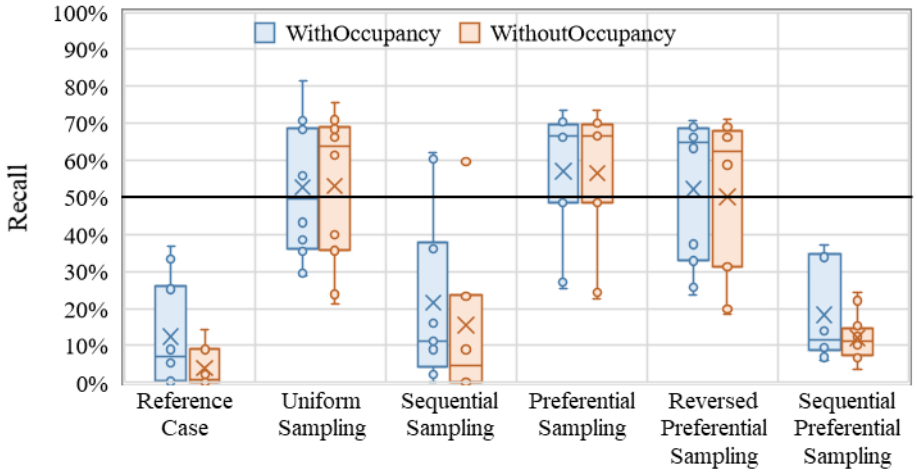
Overall, users should choose proper data pre-processing strategies and classifiers based on their demand. For instance, if better overall accuracy is prioritized, the reference case would be a good choice when the original training dataset is mainly negative class label data. However, if recall also matters, sequential sampling could be considered. If higher recall is the priority, uniform sampling, preferential sampling, and reversed preferential sampling can be the sampling strategy, while ANN could be the classifier.



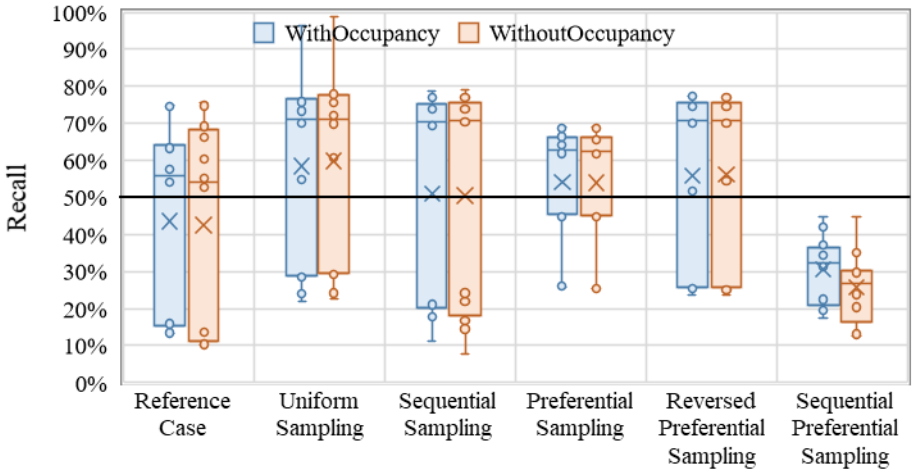
(a) SVM



(b) ANN

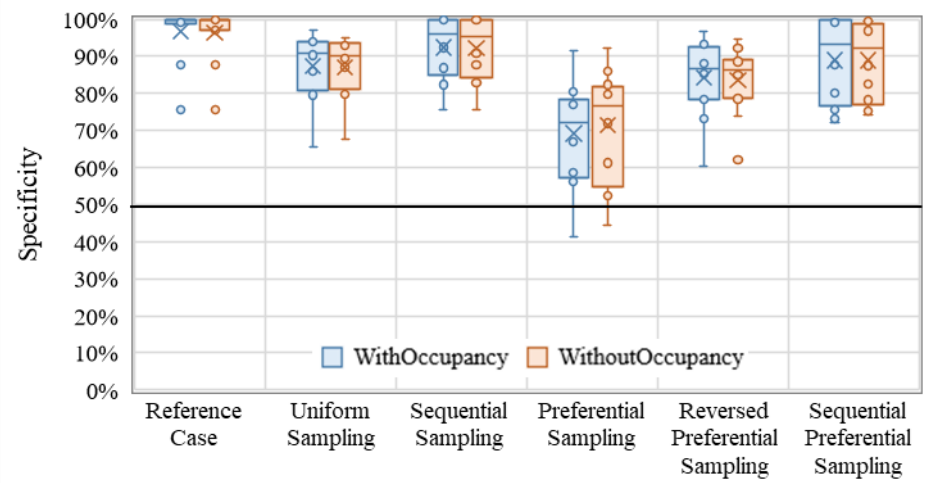


(c) Logistic Regression

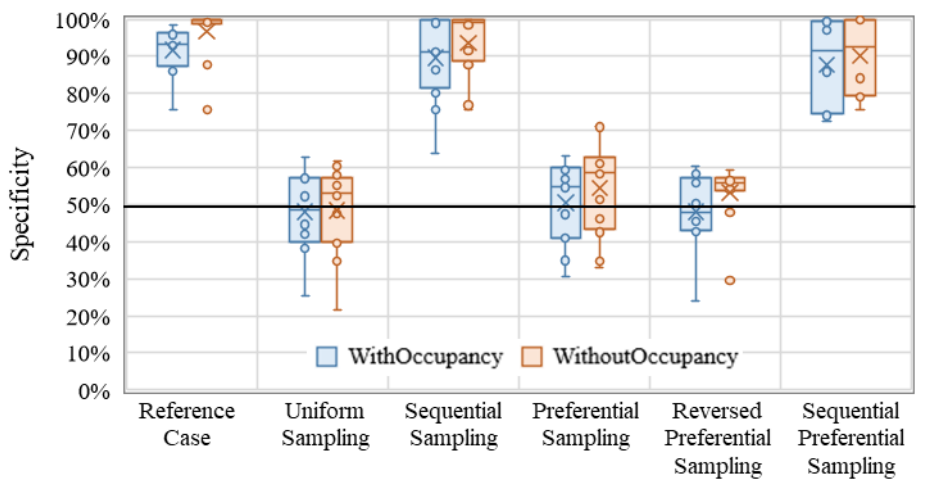


(d) Naïve Bayes

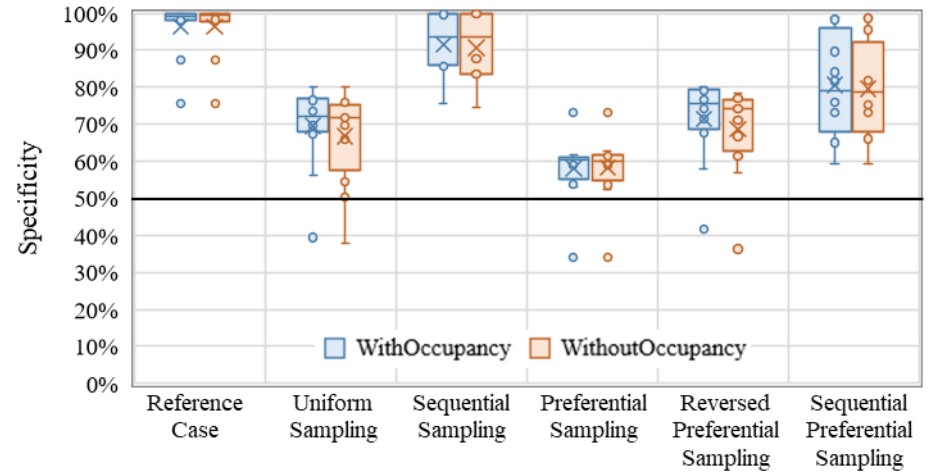
Figure 5: Recall under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



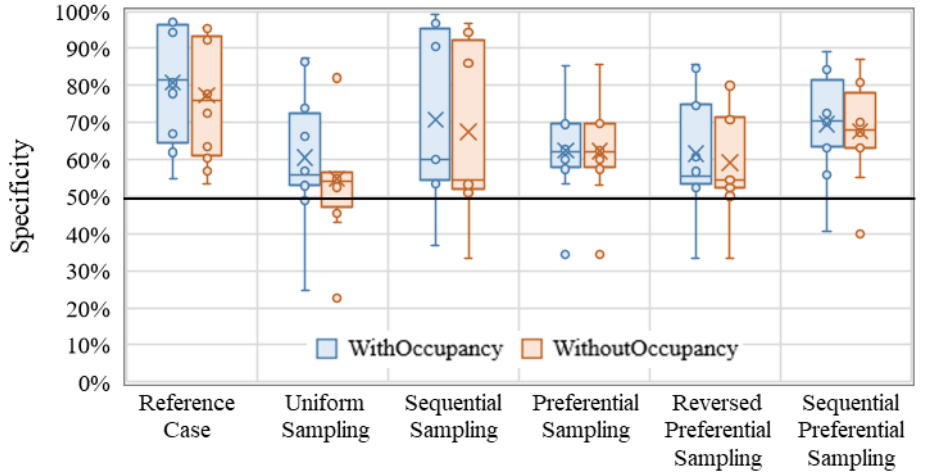
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

Figure 6: Specificity under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

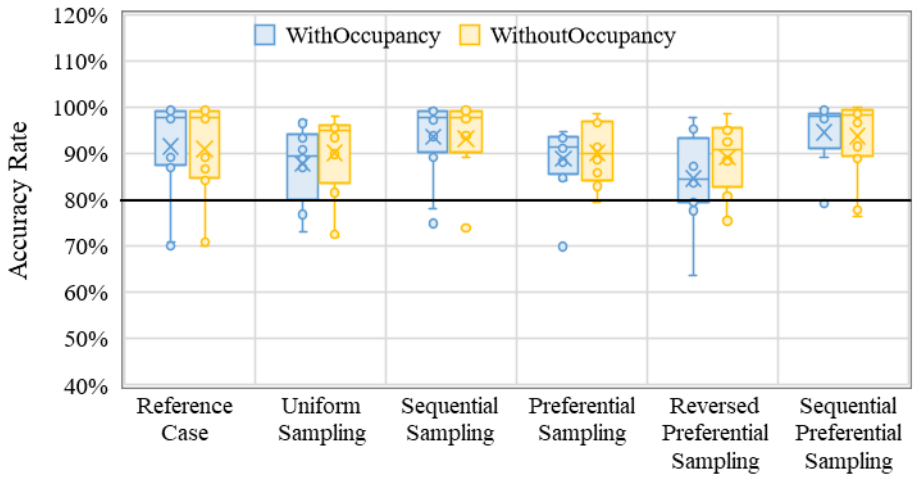
4.2. Results: fairness measures

Accuracy rate for different cases is shown in Figure 7. In this figure, accuracy rates of cases using ‘WithOccupancy’ inputs are similar to those using ‘WithoutOccupancy’. When using SVM as classifier, compared to the reference case, sequential sampling and sequential preferential sampling show the ability to increase accuracy rate to over 90%. For the ANN classifier, sequential sampling, preferential sampling, and sequential preferential sampling could increase accuracy rate to be higher than 80% for most lighting status series. Sequential sampling also increases accuracy rate when Logistic Regression is used as classifier. No sampling strategy could significantly improve fairness in terms of accuracy rate when using the Naïve Bayes classifier. Furthermore, most cases predicted by SVM present an accuracy rate over 80%, which is better than cases using other classifiers. Additionally, the fairness improvement ability of each pre-processing method varies among different lighting status series. However, no specific pattern between training data quality and accuracy rate improvement potentiality has been discovered.

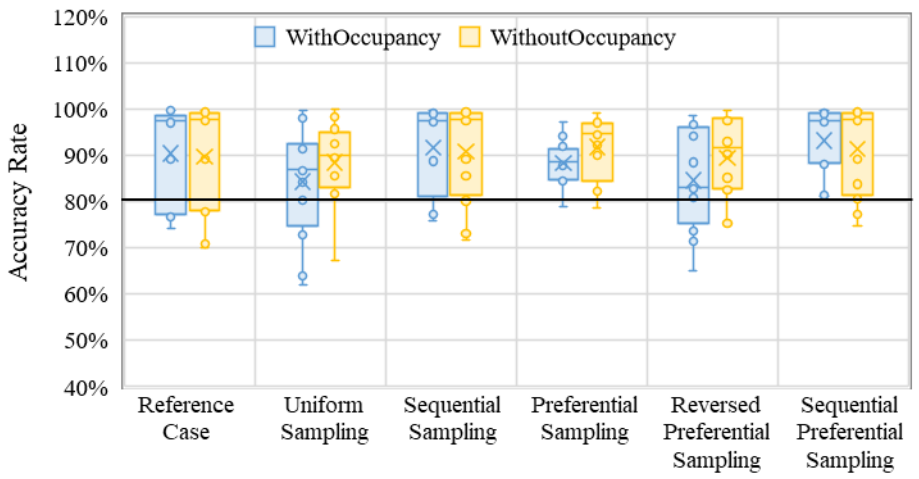
The recall rate of different cases is summarized in Figure 8. Reference cases using ‘WithoutOccupancy’ inputs have a higher recall rate (over 80% for most lighting status series) than reference cases with ‘WithOccupancy’ inputs. Besides, pre-processing strategies for improving recall rate should be selected based on classifiers. For SVM, sequential sampling would be the best option when motion status is not one of the features, and sequential preferential shows the best mean recall rate improvement ability when motion status is included. For ANN, uniform sampling could improve the mean and minimum recall rate, while sequential sampling and sequential preferential sampling could increase the median recall rate. For Logistic Regression, sequential sampling increases recall rate for cases using ‘WithOccupancy’ inputs and cases using ‘WithoutOccupancy’ inputs, while other sampling methods could significantly increase recall rate when motion status is one of the features. Finally, for Naïve Bayes, uniform sampling would be the best choice.

Figure 9 presents specificity rate for different cases and shows sampling strategies could not improve fairness in terms of specificity rate. However, sequential sampling and sequential preferential sampling could keep specificity rate meeting the “80 percent rule” for most lighting status series.

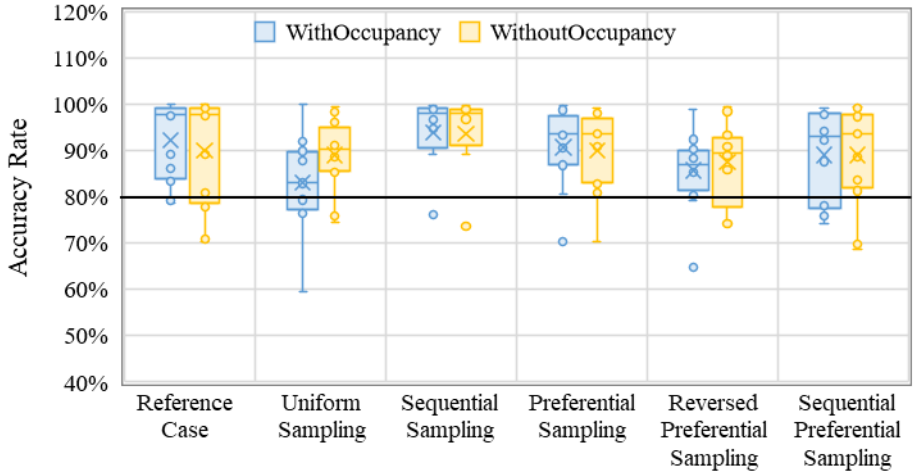
Therefore, the fairness improvement ability of sampling strategies varies among different features, classifiers, and fairness measures. In general, sequential sampling could be a useful strategy for increasing accuracy rate and recall rate while maintaining an acceptable specificity rate.



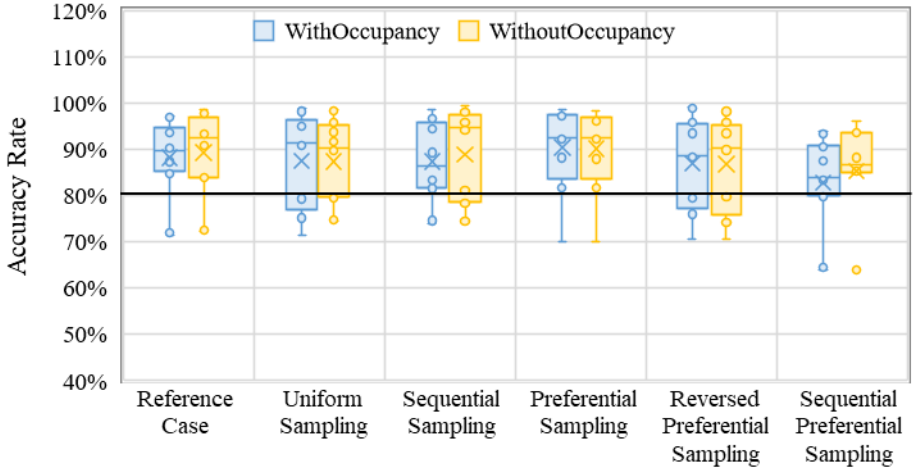
(a) SVM



(b) ANN

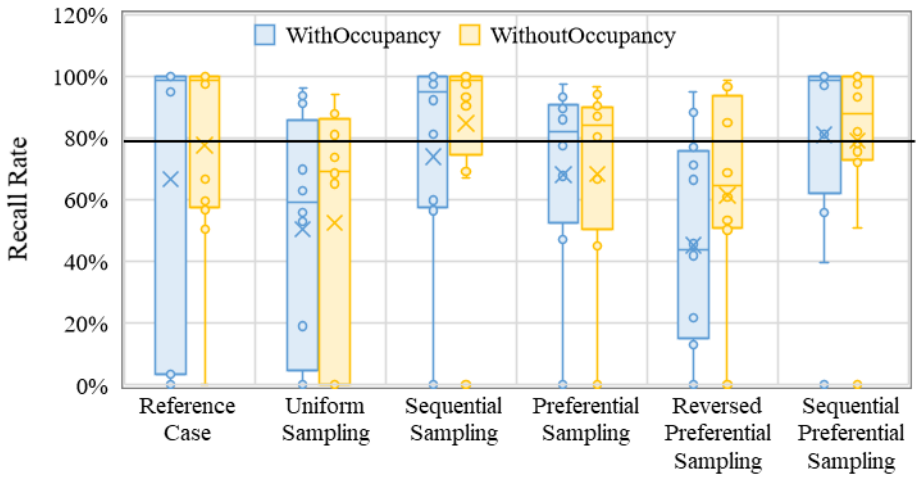


(c) Logistic Regression

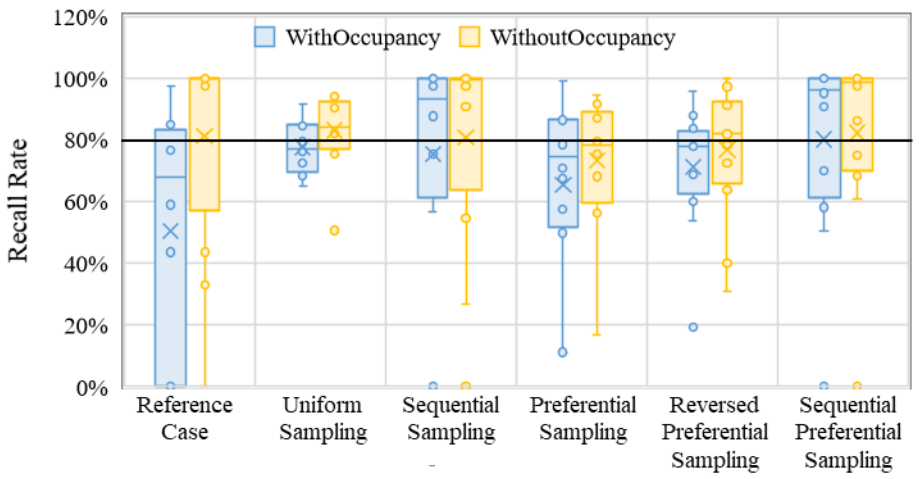


(d) Naïve Bayes

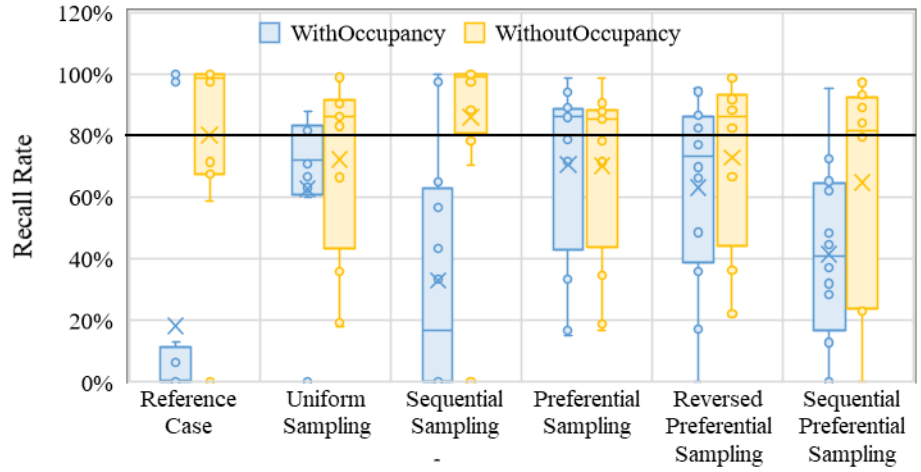
Figure 7: Accuracy rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



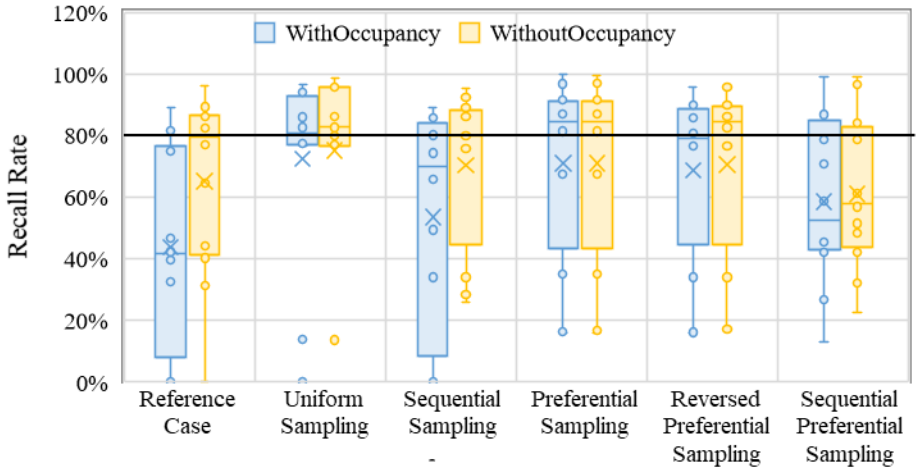
(a) SVM



(b) ANN

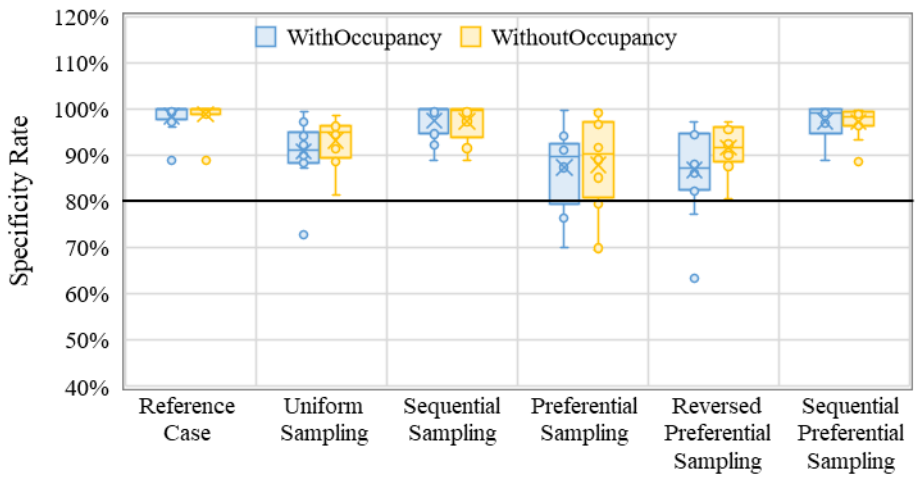


(c) Logistic Regression

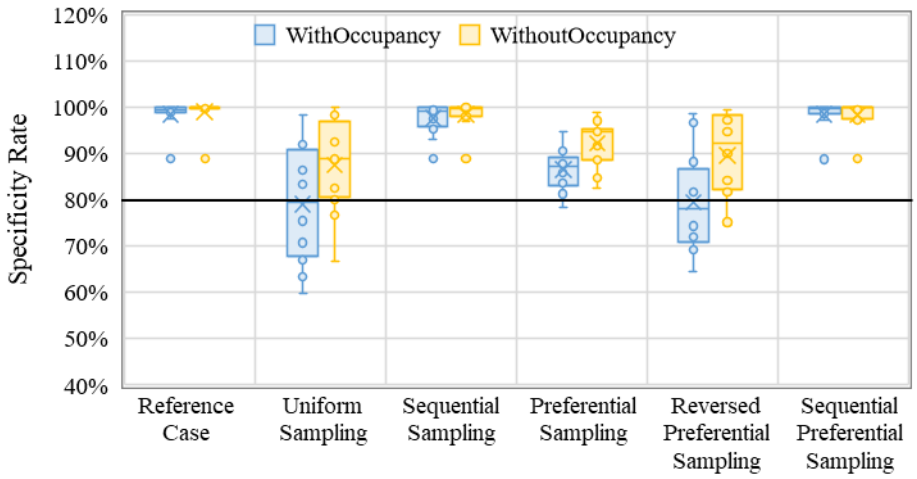


(d) Naïve Bayes

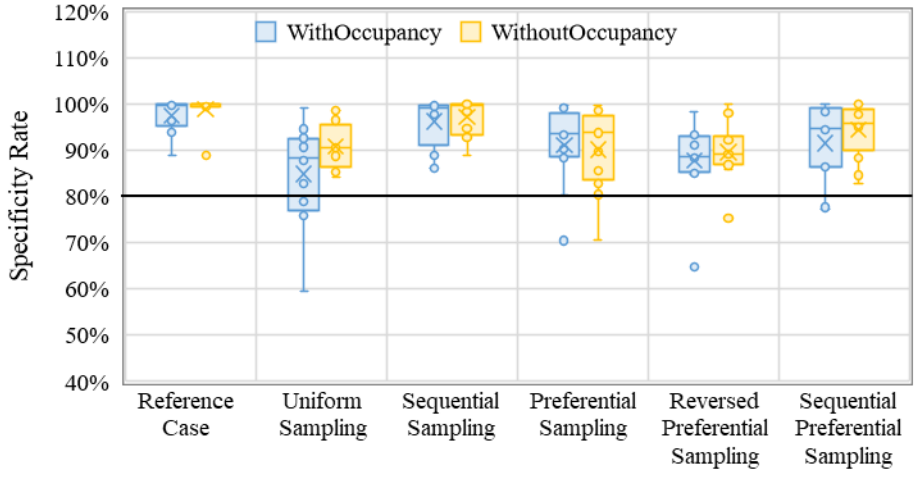
Figure 8: Recall rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



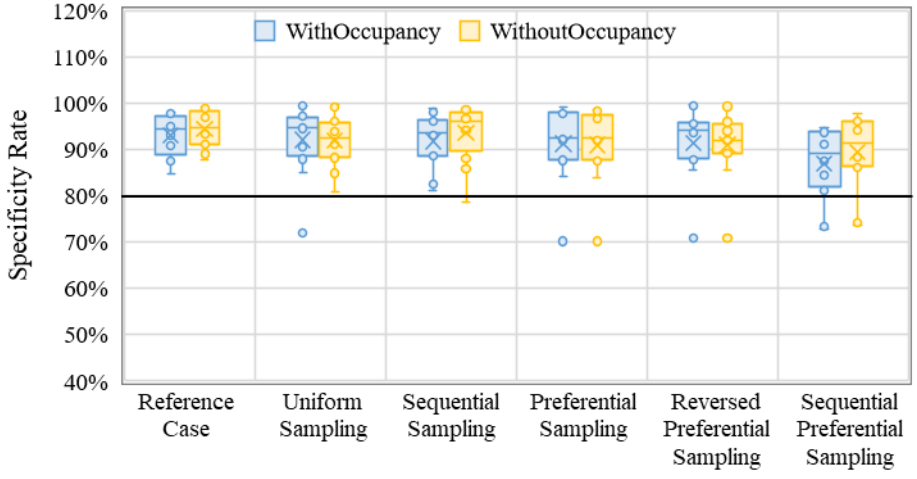
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

Figure 9: Specificity rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

5. Discussion

To better understand data distribution change in each group as predictive cycle increases, Figure 10 shows PP, PN, NP, and NN ratios among the training dataset processed by reference case, sequential sampling, and sequential preferential sampling using 'WithOccupancy' inputs. The 25% line for ratio means the amount of data in its corresponding group reaches the designed number. Note that data distribution for uniform sampling, preferential sampling, and reversed preferential sampling is not presented in this figure because |PP|, |PN|, |NP|, and |NN| are kept at the designed number all the time.

Figure 10 shows there is no specific pattern of ratio change for groups PP, PN, NP, and NN in the reference case. Among these four groups, NN accounts for the largest ratio (55% - 90%), followed by PN (5% - 45%), PP (0% - 18%), and NP (0% - 10%). This indicates lighting is OFF most of the time in the training dataset, and data is insufficient for representing the situation when lighting is ON. Therefore, it makes sense that recall and recall rate for reference cases with 'WithOccupancy' inputs are worse.

For sequential sampling and sequential preferential sampling, the ratios of these four groups try to reach 25% as prediction cycle increases. At the beginning of the prediction cycles, the ratios of PP and NP are even less than 5% for most lighting status series. When these ratios do not attain 25%, their recall improvement ability is worse than other pre-processing methods due to insufficient data when lighting is ON. However, as the data distribution is gradually balanced, sequential sampling and sequential preferential sampling could improve the recall.

Furthermore, unlike other pre-processing methods, sequential sampling and sequential preferential sampling do not duplicate data. Thus, they may harm the original data distribution pattern less. As a result, they present a better accuracy rate and recall rate improvement ability while maintaining specificity rate.

Moreover, Figure 10 shows the data ratio is almost the same between sequential sampling and sequential preferential sampling. However, as illustrated in Section 4, the predictive accuracy of sequential sampling is usually better than sequential preferential sampling. This is because

sequential sampling could capture the most recent pattern in the training dataset, while the predictive performance of sequential preferential sampling depends on its ranker.

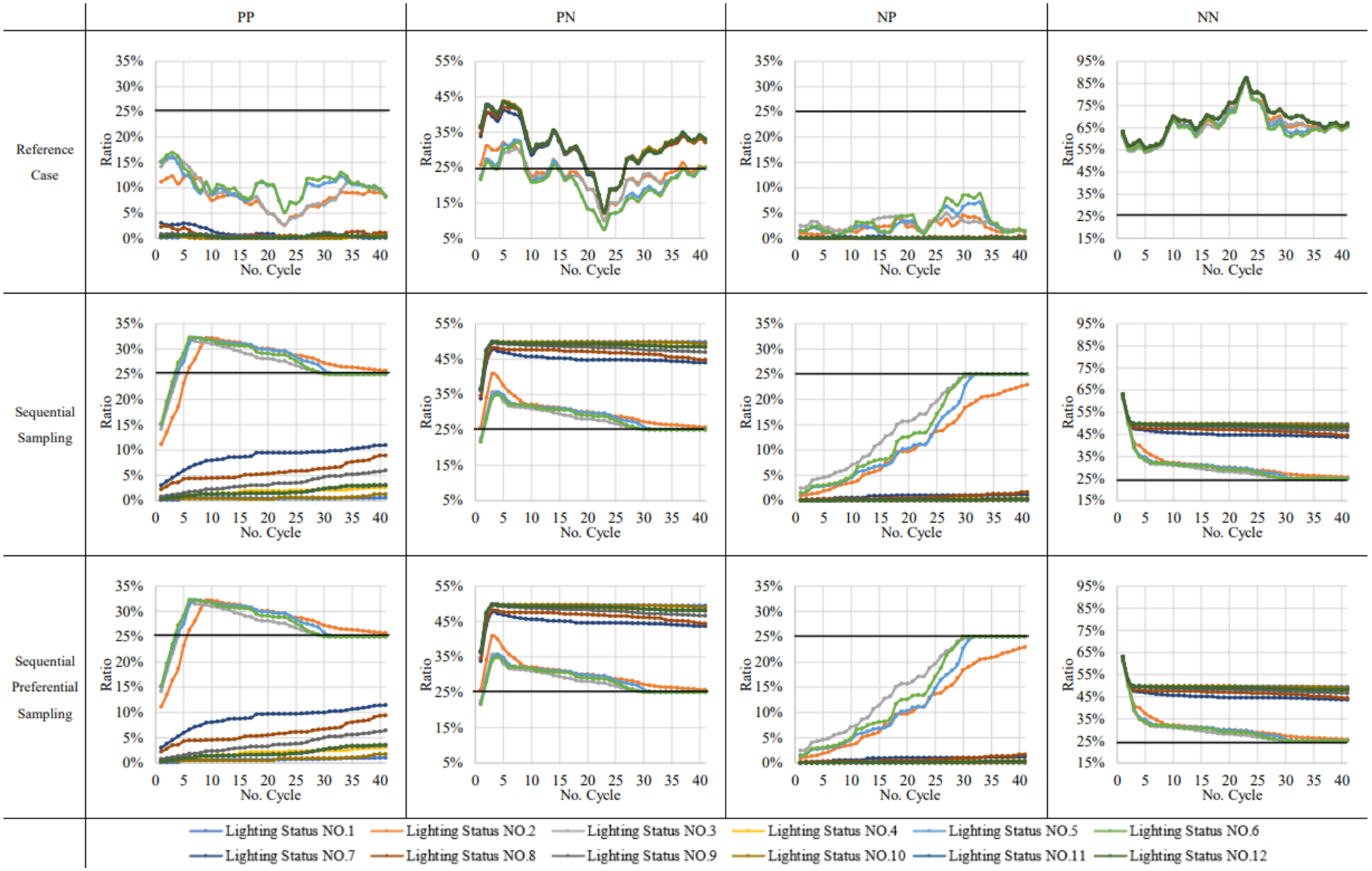


Figure 10: Ratios of PP, PN, NP, and NN among the training dataset under different pre-processing methods

6. Conclusion

In this study, the concept of fairness and the requirement to improve fairness in data-driven building and indoor environment models without harming their predictive accuracy were first explained. Then, fairness improvement methods were proposed and investigated: (1) To improve the fairness in terms of *Type II*, three pre-processing methods—sequential sampling, reversed preferential sampling, and sequential preferential sampling—were proposed to pre-process the training dataset of a two-class classification problem with a binary protected attribute. These proposed methods were compared to two existing pre-processing methods—uniform sampling and preferential sampling—regarding their impact on predictive accuracy and fairness. (2) To achieve fairness *Type I*, protected attribute suppression was implemented and its effect on the lighting status predictive performance was investigated.

This study used one-year data collected from one apartment building. Overall, 576 study cases were investigated to draw the comparison between 6 pre-processing methods under 12 series of lighting status, 2 combinations of features, and 4 classifiers. The predictive results of these cases were analyzed in terms of accuracy and fairness measures:

(1) Concerning the effect on predictive accuracy, suppressing the protected attribute would not destroy predictive accuracy. However, using pre-processing methods would decrease accuracy and specificity compared to cases that did not use them. Among these methods, sequential sampling and sequential preferential sampling worked best for preserving overall accuracy. On the other hand, pre-processing methods could effectively improve recall.

(2) For fairness improvement, sequential sampling could be a good option to increase accuracy and recall rates while maintaining an acceptable specificity rate. The fairness improvement performance of other strategies, however, varies among different features and classifiers.

This indicates that the proposed pre-processing methods could be used to improve fairness *Type II* for a classification problem with acceptable accuracy decrease. However, this study presents some limitations: (1) Pre-processing methods were studied based solely on data collected from one apartment. It cannot represent the applicability and generalizability of these methods to

other apartments/buildings. (2) Applicability of these pre-processing methods to multi-class classification problems with multi-class protected attributes was not studied. (3) Hyperparameters of pre-processing methods were not optimized. Therefore, future studies focusing on solving these drawbacks could be interesting.

Acknowledgments

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

Abbreviations

ANN	Artificial Neural Network
HEMS	Home Energy Management System
HVAC	Heating Ventilation and Air-Conditioning
MPC	Model Predictive Controller
SVM	Support Vector Machine

Nomenclature

D	Unprotected attributes
D’	Attributes without occupancy-related information
FN	False Negative
FP	False Positive
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MBE	Mean Bias Error
MSE	Mean Squared Error
NN	The group with *N*egative protected attribute and *N*egative actual class label
NMBE	Normalized Mean Bias Error [%]
NP	The group with *N*egative protected attribute and *P*ositive actual class label

PN	The group with *P*ositive protected attribute and *N*egative actual class label
PP	The group with *P*ositive protected attribute and *P*ositive actual class label
R^2	R Square
RMSE	Root Mean Square Error
S	Protected attributes
S’	Attributes with occupancy-related information
TN	True Negative
TP	True Positive
$X_{candidate}$	Candidate training dataset
$X_{designed}$	Designed training dataset
X_{valid}	Validation dataset
Y	Class label of the training point
$\widehat{\boldsymbol{Y}}$	Predicted class label
$\widehat{\boldsymbol{Y}}_{\boldsymbol{valid}}$	Predicted class label based on X_{valid}

Supplementary information

SVM

In this study, SVM classifiers are modelled by using sklearn.svm.SVC [53] in python. Hyperparameters for these classifiers are listed in Table S1. Detailed explanation for the meaning of each hyperparameter could be found in [53].

Table S1: Hyperparameters for SVM classifiers

Hyperparameters	**Value**
Regularization parameter	Squared l2 penalty
Kernel	Radial basis function
Gamma	Scale
Shrinking	True
Probability	False
Tolerance for stop criterion	$1e^{-3}$
Kernel cache size	200
Class weight	None

Verbose	False
Max iterations within solver	Unlimited
Decision function shape	One-vs-rest
Break ties	False
Random state	None

ANN

ANN classifiers are developed by using sklearn.neural_network.MLPClassifier [54] in python. Hyperparameters for these classifiers are listed in Table S2. Detailed explanation for the meaning of each hyperparameter could be found in [54].

Table S2: Hyperparameters for ANN classifiers

Hyperparameters	**Value**
Hidden layer numbers	2
Hidden later No.1 size	5
Hidden later No.2 size	2
Activation function	Rectified linear unit function
Solver	Lbfgs
Alpha (l2 penalty parameter)	0.0001
Batch size	Auto
Learning rate	Constant
Initial learning rate	0.001
Maximum number of iterations	200
Random state	None
Tolerance for optimization	$1e^{-4}$
Verbose	False
Warm start	False
Maximum number of loss function calls	15000

Logistic Regression

Logistic Regression classifiers are modelled by sklearn.linear_model.LogisticRegression [55] in python. Their hyperparameters are listed in Table S3.

Table S3: Hyperparameters for Logistic Regression classifiers

Hyperparameters	**Value**
Regularization parameter	Squared l2 penalty
Tolerance for stopping criteria	$1e^{-4}$
Inverse of regularization strength	1
Fit intercept	True
Class weight	None
Solver	Lbfgs
Maximum number of iterations	100
Random state	None
Multi class	Auto
Verbose	0
Warm start	False

Naïve Bayes

In this study, Gaussian Naïve Bayes is utilized and modelled by sklearn.naive_bayes.GaussianNB [56] in python. The hyperparameters are listed in Table S4.

Table S4: Hyperparameters for Gaussian Naive Bayes classifiers

Hyperparameters	**Value**
Prior probabilities of the classes	None
Variance smoothing	$1e^{-9}$

References

[1] Han D, Lim J. Smart home energy management system using IEEE 802.15.4 and zigbee. IEEE Trans Consum Electron 2010;56:1403–10. https://doi.org/10.1109/TCE.2010.5606276.

[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] Naji S, Keivani A, Shamshirband S, Alengaram UJ, Jumaat MZ, Mansor Z, et al. Estimating building energy consumption using extreme learning machine method. Energy 2016;97:506–16. https://doi.org/10.1016/j.energy.2015.11.037.

[4] Mocanu E, Nguyen PH, Gibescu M, Kling WL. Deep learning for estimating building energy consumption. Sustain Energy Grids Netw 2016;6:91–9. https://doi.org/10.1016/j.segan.2016.02.005.

[5] Talebi B, Haghighat F, Mirzaei PA. Simplified model to predict the thermal demand profile of districts. Energy Build 2017;145:213–25. https://doi.org/10.1016/j.enbuild.2017.03.062.

[6] Ascione F, Bianco N, De Stasio C, Mauro GM, Vanoli GP. Artificial neural networks to predict energy performance and retrofit scenarios for any member of a building category: A novel approach. Energy 2017;118:999–1017.

[7] Peng Y, Rysanek A, Nagy Z, Schlüter A. Using machine learning techniques for occupancy-prediction-based cooling control in office buildings. Appl Energy 2018;211:1343–58.

[8] Gariba D, Pipaliya B. Modelling human behaviour in smart home energy management systems via machine learning techniques, IEEE; 2016, p. 53–8.

[9] Zhou Y, Chen J, Yu ZJ, Li J, Huang G, Haghighat F, et al. A novel model based on multi-grained cascade forests with wavelet denoising for indoor occupancy estimation. Build Environ 2020;167:106461. https://doi.org/10.1016/j.buildenv.2019.106461.

[10] 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.

[11] 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.

[12] Tabares-Velasco PC, Speake A, Harris M, Newman A, Vincent T, Lanahan M. A modeling framework for optimization-based control of a residential building thermostat for time-of-use pricing. Appl Energy 2019;242:1346–57. https://doi.org/10.1016/j.apenergy.2019.01.241.

[13] Reynolds J, Rezgui Y, Kwan A, Piriou S. A zone-level, building energy optimisation combining an artificial neural network, a genetic algorithm, and model predictive control. Energy 2018;151:729–39.

[14] Boghetti R, Fantozzi F, Kämpf JH, Salvadori G. Understanding the performance gap: a machine learning approach on residential buildings in Turin, Italy. J Phys Conf Ser 2019;1343:012042. https://doi.org/10.1088/1742-6596/1343/1/012042.

[15] Figueiredo A, Kämpf J, Vicente R, Oliveira R, Silva T. Comparison between monitored and simulated data using evolutionary algorithms: Reducing the performance gap in dynamic building simulation. J Build Eng 2018;17:96–106. https://doi.org/10.1016/j.jobe.2018.02.003.

[16] Olsthoorn D, Haghighat F, Moreau A, Joybari MM, Robichaud M. Integration of electrically activated concrete slab for peak shifting in a light-weight residential building—Determining key parameters. J Energy Storage 2019;23:329–43. https://doi.org/10.1016/j.est.2019.03.023.

[17] Amasyali K, El-Gohary NM. A review of data-driven building energy consumption prediction studies. Renew Sustain Energy Rev 2018;81:1192–205. https://doi.org/10.1016/j.rser.2017.04.095.
[18] Mohandes SR, Zhang X, Mahdiyar A. A comprehensive review on the application of artificial neural networks in building energy analysis. Neurocomputing 2019;340:55–75. https://doi.org/10.1016/j.neucom.2019.02.040.
[19] Bourdeau M, Zhai X qiang, Nefzaoui E, Guo X, Chatellier P. Modeling and forecasting building energy consumption: A review of data-driven techniques. Sustain Cities Soc 2019;48:101533. https://doi.org/10.1016/j.scs.2019.101533.
[20] Wei N, Li C, Peng X, Zeng F, Lu X. Conventional models and artificial intelligence-based models for energy consumption forecasting: A review. J Pet Sci Eng 2019;181:106187. https://doi.org/10.1016/j.petrol.2019.106187.
[21] Sun Y, Haghighat F, Fung BCM. A review of the-state-of-the-art in data-driven approaches for building energy prediction. Energy Build 2020;221:110022. https://doi.org/10.1016/j.enbuild.2020.110022.
[22] Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res 2003;3:1157–82.
[23] Wang X, Tague P. Non-invasive user tracking via passive sensing: Privacy risks of time-series occupancy measurement. Proc. 2014 Workshop Artif. Intell. Secur. Workshop, 2014, p. 113–24.
[24] Jia R, Dong R, Sastry SS, Sapnos CJ. Privacy-Enhanced Architecture for Occupancy-Based HVAC Control. 2017 ACMIEEE 8th Int. Conf. Cyber-Phys. Syst. ICCPS, 2017, p. 177–86.
[25] 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.
[26] Mehrabi N, Morstatter F, Saxena N, Lerman K, Galstyan A. A survey on bias and fairness in machine learning. ArXiv Prepr ArXiv190809635 2019.
[27] Yan K, Huang J, Shen W, Ji Z. Unsupervised learning for fault detection and diagnosis of air handling units. Energy Build 2020;210:109689.
[28] Zhang C, Li J, Zhao Y, Li T, Chen Q, Zhang X, et al. Problem of data imbalance in building energy load prediction: Concept, influence, and solution. Appl Energy 2021;297:117139.
[29] Corbett-Davies S, Goel S. The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning. ArXiv180800023 Cs 2018.
[30] Tripathy BK. Database anonymization techniques with focus on uncertainty and multi-sensitive attributes. Handb. Res. Comput. Intell. Eng. Sci. Bus., IGI Global; 2013, p. 364–83.
[31] joan. What is a protected attribute? | Employment Law Australia n.d. https://employmentlawhandbook.com.au/chapters/discrimination/what-is-a-protected-attribute/ (accessed August 11, 2021).
[32] Niculescu-Mizil A, Caruana R. Predicting good probabilities with supervised learning. Proc. 22nd Int. Conf. Mach. Learn., 2005, p. 625–32.
[33] Bogen M, Rieke A. Help wanted: an examination of hiring algorithms. Equity Bias Upturn Dec 2018 2018.
[34] Cohen L, Lipton ZC, Mansour Y. Efficient candidate screening under multiple tests and implications for fairness. ArXiv190511361 Cs Stat 2019.
[35] Langenkamp M, Costa A, Cheung C. Hiring Fairly in the Age of Algorithms. ArXiv Prepr ArXiv200407132 2020.
[36] Friedler SA, Scheidegger C, Venkatasubramanian S, Choudhary S, Hamilton EP, Roth D. A comparative study of fairness-enhancing interventions in machine learning. ArXiv180204422 Cs Stat 2018.

[37] Calmon F, Wei D, Vinzamuri B, Natesan Ramamurthy K, Varshney KR. Optimized Pre-Processing for Discrimination Prevention. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Adv. Neural Inf. Process. Syst. 30, Curran Associates, Inc.; 2017, p. 3992–4001.
[38] Elzayn H, Jabbari S, Jung C, Kearns M, Neel S, Roth A, et al. Fair Algorithms for Learning in Allocation Problems. ArXiv180810549 Cs Stat 2018.
[39] Mukerjee A, Biswas R, Deb K, Mathur AP. Multi–objective Evolutionary Algorithms for the Risk–return Trade–off in Bank Loan Management. Int Trans Oper Res 2002;9:583–97. https://doi.org/10.1111/1475-3995.00375.
[40] Bose AJ, Hamilton W. Compositional fairness constraints for graph embeddings. ArXiv Prepr ArXiv190510674 2019.
[41] Zafar MB, Valera I, Gomez Rodriguez M, Gummadi KP. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. Proc. 26th Int. Conf. World Wide Web, 2017, p. 1171–80.
[42] Holstein K, Vaughan JW, Daumé III H, Dudík M, Wallach H. Improving fairness in machine learning systems: What do industry practitioners need? Proc 2019 CHI Conf Hum Factors Comput Syst - CHI 19 2019:1–16. https://doi.org/10.1145/3290605.3300830.
[43] 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.
[44] Feldman M, Friedler S, Moeller J, Scheidegger C, Venkatasubramanian S. Certifying and removing disparate impact. ArXiv14123756 Cs Stat 2015.
[45] 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.
[46] Kleinberg J, Mullainathan S, Raghavan M. Inherent Trade-Offs in the Fair Determination of Risk Scores. ArXiv160905807 Cs Stat 2016.
[47] 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.
[48] 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.
[49] Cortes C, Vapnik V. Support-vector networks. Mach Learn 1995;20:273–97. https://doi.org/10.1007/BF00994018.
[50] Walker SH, Duncan DB. Estimation of the Probability of an Event as a Function of Several Independent Variables. Biometrika 1967;54:167–79. https://doi.org/10.2307/2333860.
[51] 1.9. Naive Bayes — scikit-learn 0.23.2 documentation n.d. https://scikit-learn.org/stable/modules/naive_bayes.html (accessed September 16, 2020).
[52] Claesen M, De Moor B. Hyperparameter search in machine learning. ArXiv Prepr ArXiv150202127 2015.
[53] sklearn.svm.SVC — scikit-learn 0.23.1 documentation n.d. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed July 13, 2020).
[54] sklearn.neural_network.MLPClassifier — scikit-learn 0.23.1 documentation n.d. https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html (accessed July 13, 2020).
[55] sklearn.linear_model.LogisticRegression — scikit-learn 0.23.1 documentation n.d. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (accessed July 13, 2020).
[56] sklearn.naive_bayes.GaussianNB — scikit-learn 0.23.1 documentation n.d. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html (accessed July 13, 2020).