Fairness-Aware Data-Driven Building Models (DDBMs) and Their Application in Model Predictive Controller (MPC)

Ying Sun

A Thesis
In the Department
of
Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the
Requirements For the Degree of
Doctor of Philosophy (Building Engineering)
at Concordia University
Montreal, Québec, Canada

July 2022

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: **Ying Sun**

Entitled: **Fairness-Aware Data-Driven Building Models (DDBMs) and Their Application in Model Predictive Controller (MPC)**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Building Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____Chair
Dr. Yuhong Yan

_____Thesis Supervisor
Dr. Fariborz Haghighat

_____Thesis Supervisor
Dr. Benjamin C.M. Fung

_____Examiner
Dr. Anjali Awasthi

_____Examiner
Dr. Mazdak Nik-Bakht

_____Examiner
Dr. Liangzhu Wang

_____External Examiner
Dr. Yinping Zhang

Approved by _____
Dr. Mazdak Nik-Bakht, Graduate Program Director

_____
Dr. Mourad Debbabi, Dean of Faculty

Date _____

Abstract

Fairness-Aware Data-Driven Building Models and Their Application in Model Predictive Controller (MPC)

Ying Sun, Ph.D.
Concordia University, 2022

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, due to the variation in data volume of different conditions, existing DDBMs may present distinct accuracy for different users/occupants or periods/conditions. Accuracy variation among users or periods may creates unfairness problems (i.e., algorithmic biases created by data-driven models). This thesis explores and tackles this research problem called fairness-aware prediction of DDBMs.

This thesis first presents a comprehensive review of the entire process involved in developing a DDBM and emphasizes the research gap on achieving fairness in DDBMs. As the first research that introduces fairness concepts into the building engineering domain, this thesis summarizes three types of commonly used fairness definitions. Among these concepts, achieving *Type I* and *Type II* fairness in DDBMs shows the beneficial for enabling users to do authority management, achieving uniform predictive performance under different periods or situations, and preserving fairness for different users. In addition, this thesis reviews the commonly used fairness improvement methods for data-driven models.

Then, with the aim of improving fairness for DDBMs to have uniform predictive performance under different conditions and letting MPCs in buildings get optimal control signals based on fair prediction, this research proposes fairness improvement methods for both classification problems and regression problems in the building engineering domain and integrates fairness-aware DDBMs into model predictive controllers (MPCs). This work is separated into three tasks: 1) Task A: For classification problems, four kinds of pre-processing methods are proposed to balance the training dataset. 2) Task B: For regression problems, four in-processing methods, which incorporate fairness-related constraints or penalties into the optimization objective function during the training process of data-driven models, are studied. 3) Task C: The fairness improvement methods proposed in Task A and Task B will be integrated into MPCs.

Case studies are conducted to implement the proposed fairness improvement methods to DDBMs for apartments, develop and integrate the fairness-aware DDBMs into MPCs to get the optimal set-point temperature for controlling the electrically heated floor system (EHF, a heating system with energy storage ability) in a bungalow building. The results show that 1) The proposed pre-processing methods could improve the predictive accuracy of minority conditions and increase fairness in terms of the accuracy rate between different conditions. 2) The proposed in-processing methods could achieve user-defined trade-off between accuracy and fairness. The *Type II* fairness is achieved by increasing the predictive performance similarity between different conditions. 3) Although improving predictive fairness would decrease the overall predictive accuracy, fairness-aware data-driven based MPCs would not decrease the cost saving and peak shifting ability, compared to the traditional MPC without considering fairness.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisors, Dr. Fariborz Haghighat and Dr. Benjamin C. M. Fung. I am so lucky to have Dr. Haghighat as my supervisor for Master and PhD program. He gave me a lot of invaluable and professional help and support over these years. His kindness help inspires me to be a supportive supervisor like him in the future. I also want to thank my co-supervisor Dr. Fung. His advice and support throughout my study guide me the way to discover the interesting research topic for my PhD program.

I would like to appreciate my friend, Karthik Panchabikesan, for kindly helping me improve the English of this thesis. Thanks for giving me many knowledgeable comments. I want to show my deep appreciation to Dr. Yanping Yuan, Dr. Chao Zeng, and Li Yang from Southwest Jiaotong University. It was a precious and unforgettable experience to learn in their team. Also, I would like to appreciate my friend, Enyao, for his support and encouragement in the last year of my study. I would like to thank all my friends and colleagues in our research group, Concordia University, and Annex 37, for their kindness help and friendship.

Last but not least, I wish to express my warmest appreciation and love to my grandma, my parents and my younger brother. May the god bless my grandma, no more pain in heaven. In the last two years, my family suffered a lot. Hope everything will be fine in the nearly future.

If winter comes, can spring be far behind? ——*P. B. Shelley*

This thesis dedicated to my beloved family.

Contribution of Authors

Journal Papers:

Chapter	**2**
Title	**A Review of the-State-of-the-Art in Data-driven Approaches for Building Energy Prediction**
Authors	Ying Sun, Fariborz Haghighat, Benjamin C. M. Fung
Status	Published
Abstract	Building energy prediction plays a vital role in developing a model predictive controller for consumers and optimizing energy distribution plan for utilities. Common approaches for energy prediction include physical models, data-driven models and hybrid models. Among them, data-driven approaches have become a popular topic in recent years due to their ability to discover statistical patterns without expertise knowledge. To acquire the latest research trends, this study first summarizes the limitations of earlier reviews: seldom present comprehensive review for the entire data-driven process for building energy prediction and rarely summarize the input updating strategies when applying the trained data-driven model to multi-step energy prediction. To overcome these gaps, this paper provides a comprehensive review on building energy prediction, covering the entire data-driven process that includes feature engineering, potential data-driven models and expected outputs. The distribution of 105 papers, which focus on building energy prediction by data-driven approaches, are reviewed over data source, feature types, model utilization and prediction outputs. Then, in order to implement the trained data-driven models into multi-step prediction, input updating strategies are reviewed to deal with the time series property of energy related data. Finally, the review concludes with some potential future research directions based on discussion of existing research gaps.
Reference	Sun, Y., Haghighat, F., & Fung, B. C. (2020). A review of the-state-of-the-art in data-driven approaches for building energy prediction. *Energy and Buildings*, 221, 110022. https://doi.org/10.1016/j.enbuild.2020.110022.

Chapter	**2, 3, 4, 5**
Title	**Trade-off Between Accuracy and Fairness of Data-driven Building and Indoor Environment Models: A Comparative Study of Pre-processing Methods**
Authors	Ying Sun, Fariborz Haghighat, Benjamin C. M. Fung
Status	Published

Description	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.
Reference	Sun, Y., Haghighat, F., & Fung, B. C. (2022). Trade-off between accuracy and fairness of data-driven building and indoor environment models: A comparative study of pre-processing methods. *Energy*, 239, 122273. https://doi.org/10.1016/j.energy.2021.122273.

Chapter	**2, 3, 4, 5**
Title	**The generalizability of pre-processing techniques on the accuracy and fairness of data-driven building models: a case study**
Authors	Ying Sun, Benjamin C. M. Fung, Fariborz Haghighat
Status	Published
Abstract	In recent years, massive data collected from buildings made development and application of data-driven building models is a hot research topic. Due to the variation of data volume in different conditions, existing data-driven building models (DDBMs) would present distinct accuracy for different users or periods. This may create further fairness problems. To solve these issues, balancing training dataset between different conditions using pre-processing techniques could help. In this study, a sequentially balanced sampling (SBS) technique is proposed. Its generalizability to improve fairness and preserve accuracy of DDBMs is compared with four existing pre-processing techniques—random sampling (RS), sequential sampling (SS), reversed preferential sampling (RPS), and sequential preferential sampling (SPS).

	Totally, 4960 cases are carried out to apply these pre-processing techniques to process training dataset before developing 4 types of classifiers for one-week ahead lighting status prediction of 155 lights in 16 apartments through a year. Note that the collected data show 5 distribution modes. The newly proposed SBS shows comparable performance to RPS. They significantly improve predictive accuracy for minority classes but decrease the accuracy for majority classes. On the other hand, SS and SPS show a slight accuracy improvement for minority classes with an acceptable price of accuracy decrease on majority classes. In terms of fairness improvement, SBS, RS, and RPS could effectively increase the recall rate. However, RS and RPS show more negative effect on accuracy rate and specificity rate. The results of this study provide guidance for researchers to select proper pre-processing techniques to improve the preferred predictive performance under different data distribution.
Reference	Sun, Y., Fung, B. C., & Haghighat, F. (2022). The generalizability of pre-processing techniques on the accuracy and fairness of data-driven building models: a case study. *Energy and Buildings*, 112204. https://doi.org/10.1016/j.enbuild.2022.112204

Chapter	**2,3,4,5**
Title	**Fairness-aware Regression Models for Building Energy Prediction to Achieve Uniform Performance**
Authors	Ying Sun, Benjamin C. M. Fung, Fariborz Haghighat
Status	Submitted
Description	Developing data-driven building energy prediction models has become a challenging topic in recent years, because accurate predictive result could benefit both user and suppliers. However, a model with high overall accuracy would not ensure a good predictive performance on all conditions. The performance variation may cause fairness problems. Although pre-processing methods were proposed to improve predictive fairness 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. In this study, 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, are proposed to improve the predictive performance similarity between different conditions by adding 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. The aim of this study case is to improve predictive fairness to let the predictive accuracy be uniform no matter if there is personnel activity, while preserving the overall predictive accuracy. 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.
Reference	

Conference Papers:

Chapter	**3,4,5**
Title	**The fairness improvement abilities of pre-processing methods for data-driven building models: A parametric study**
Authors	Ying Sun, Fariborz Haghighat, Benjamin C. M. Fung
Status	Accepted
Description	As plenty of data are collected from buildings in recent years, data-driven building models becomes a hot topic in building engineering domain. Existing models suffer an issue that the predictive performance may vary among different conditions due to the different volume of training data. This problem is also called as unfairness problem. To solve this problem, three pre-processing methods, namely sequential sampling, reversed preferential sampling, and sequential preferential sampling, were proposed in authors' previous study. In this study, the applicability and robustness of these three pre-processing methods will be studied through applying them to two apartments. Also, the effect of rankers (logistic regression or Naïve Bayes) in reversed preferential sampling and sequential preferential sampling on the predictive accuracy and fairness will be studied. The predictive results show that all pre-processing methods decrease the overall predictive accuracy: Reversed preferential sampling presents the largest negative effect on the overall accuracy and specificity, followed by sequential preferential sampling and sequential sampling. By contrast, reversed preferential sampling has the best recall improvement potential than other methods. In terms of fairness improvement ability, in general, reversed preferential sampling shows more harm to the accuracy rate, but it effectively increases the recall rate. Note that the fairness improvement performance of pre-processing methods is different among classifiers. Users are recommended to select proper combinations of pre-processing methods and classifiers based on their objective.
Publisher	CEBE2021: The 2nd International Conference for Global Chinese Academia on Energy and Built Environment

Chapter	**3,4,5**
Title	**Fairness-aware Regression Models for Building Energy Prediction to Achieve Uniform Performance**
Authors	Ying Sun, Fariborz Haghighat, Benjamin C. M. Fung
Status	Accepted

Description	Data-driven energy prediction models have drawn extensive attention in building domain in recent years. Improving the predictive accuracy of energy prediction models has been the main concern for existing research. However, an accurate model could not ensure perfect performance under all situations and the performance variation may cause fairness problems. To improve the fairness in terms of having uniform predictive accuracy under different situations, this paper applied two in-processing methods for a regression model, named mean square error penalized (MSEP) regression and mean square error constrained (MSEC) regression, to predict hourly energy consumption for an apartment. The result show that MSEP and MSEC could effectively increase the accuracy similarity in terms of MSE rate to be higher than 75%. However, MSEC would decrease the NMBE from -3.2% (for reference case) to -14.7%, while MSEP did not show significant negative effect on predictive accuracy in terms of NMBE.
Publisher	COBEE2022: 5th International Conference on Building Energy and Environment

Table of Contents

List of Figures xiv
List of Tables xviii
List of Symbols and Abbreviations xix
1. Introduction 1
1.1. Background and motivation 1
1.2. Objectives and contributions 3
1.3. Thesis outline 4
2. Chapter Two: Literature Review 6
2.1. DDBMs and their development procedure 6
2.1.1. Feature engineering 6
2.1.1.1. Feature types 6
2.1.1.2. Feature extraction methods 9
2.1.2. Supervised data-driven algorithms 12
2.1.2.1. Linear regression 12
2.1.2.2. Logistic regression (LR) 13
2.1.2.3. Time series analysis 13
2.1.2.4. Naïve Bayes (NB) 14
2.1.2.5. Decision tree (DT)/Regression tree (RT) 14
2.1.2.6. Support vector machine (SVM) / Support vector regression (SVR) 14
2.1.2.7. Artificial neural network (ANN) 15
2.1.2.8. Deep learning 16
2.1.2.9. Ensemble methods 17
2.1.3. Outputs 19
2.1.3.1. Output type 19
2.1.3.2. Building type 19
2.1.3.1. Scale 20
2.1.3.2. Temporal granularity 20
2.1.4. Performance criteria 20
2.1.5. Summary 22
2.2. Fairness-aware data-driven models 25
2.2.1. Fairness concepts 25

2.2.2. Benefits of considering fairness in DDBMs 26
2.2.3. Fairness improvement methods 26
2.2.3.1. Pre-processing methods 27
2.2.3.2. In-processing methods 28
2.2.3.3. Post-processing methods 29
2.2.4. Summary 30
3. Chapter Three: Methodology 31
3.1. Task A: Pre-processing methods for classification problems 31
3.1.1. Sequentially balanced sampling (SBS) 32
3.1.2. Sequential sampling (SS) 34
3.1.3. Reversed preferential sampling (RPS) 35
3.1.4. Sequential preferential sampling (SPS) 36
3.1.5. Random sampling (RS) 37
3.1.6. Preferential sampling (PS) 38
3.2. Task B: In-processing methods for regression problems 38
3.2.1. In-processing methods 38
3.2.1.1. Mean residual difference penalized regression (MRDP) 39
3.2.1.2. Mean square error penalized regression (MSEP) 39
3.2.1.3. Mean residual difference constrained regression (MRDC) 40
3.2.1.4. Mean square error constrained regression (MSEC) 40
3.2.2. Optimization algorithm 40
3.3. Task C: Fairness-aware data-driven based MPC for buildings 41
4. Chapter Four: Description of Case Studies 43
4.1. Task A 43
4.1.1. Case study A-1: Comparison with existing methods 43
4.1.1.1. Data description 43
4.1.1.2. Case description 45
4.1.1.3. Performance evaluation criteria 47
4.1.2. Case study A-2: Investigation of generalizability 48
4.1.2.1. Data description 48
4.1.2.2. Case description 52
4.2. Task B 54
4.2.1. Data description and feature selection 54
4.2.2. Case description 55

4.3. Task C 56
4.3.1. Experimental building 56
4.3.2. Modified TRNSYS model 57
4.3.3. Fairness-aware MPC development 58
5. Chapter Five: Results and Discussion 60
5.1. Task A 60
5.1.1. Results for case study A-1 60
5.1.1.1. A-1 Results: accuracy measures 60
5.1.1.2. A-1 Results: fairness measures 65
5.1.1.3. Discussion 69
5.1.2. Results for case study A-2 71
5.1.2.1. A-2 Results: accuracy measures 71
5.1.2.2. A-2 Results: fairness measures 77
5.1.2.3. Discussion 81
5.2. Task B 85
5.2.1. Results: Accuracy in terms of MSE and fairness in terms of MSE rate 85
5.2.2. Results: Accuracy in terms of MAE and fairness in terms of MAE rate 88
5.2.3. Discussion 90
5.2.3.1. Effect of Loss_ori selection 90
5.2.3.2. Effect of optimization algorithms 91
5.3. Task C 92
5.3.1. Predictive performance of data-driven models 92
5.3.1.1. Predictive result of energy prediction models 92
5.3.1.2. Predictive result of air temperature prediction models 99
5.3.2. Control performance comparison 100
5.3.3. Discussion 102
5.3.3.1. Effect of increasing the lower bound for set-point temperature 102
5.3.3.2. Effect of maximizing peak shifting 104
5.3.3.3. Others 106
6. Chapter Six: Conclusion 108
6.1. Summary and conclusion 108
6.2. Future work and recommendations 111
References 113
Appendix A – Hyperparameters for classifiers developed in case study A-1 124

Appendix B – Correlation matrix between input features and output in case study A-2 126
Appendix C – Feature selection for data-driven models used by MPC in Task C 130

LIST OF FIGURES

Figure 2-1: Illustration of autoencoder model architecture [92] 11
Figure 2-2: Schematic of decision tree 14
Figure 2-3: Schematic of margin between different categories 15
Figure 2-4: Schematic of typical ANN 16
Figure 2-5: Schematic of (a) fully connected layer, and (b) convolutional layer (c) loop in RNN 17
Figure 2-6: Schematic of different ensemble methods 18
Figure 3-1: Procedure of SBS, SS, RPS, and SPS in a two-class classification problem with a binary protected attribute 33
Figure 3-2: General procedure of a DE 41
Figure 3-3: Simplified schematic of fairness-aware data-driven based MPC 42
Figure 4-1: Ratios of PP, PN, NP, and NN for Lighting Status No.1 to No.12 45
Figure 4-2: Training and validation procedure of study cases A-1 46
Figure 4-3: Lay-out of studied apartments 49
Figure 4-4: Number of samples for each apartment 49
Figure 4-5: Ratios of PP, PN, NP, and NN among 16 apartments 51
Figure 4-6: Training and validation procedure of study cases A-2 53
Figure 4-7: NHEC distribution 55
Figure 4-8: Experimental building used in the case study of Task 3 57
Figure 4-9: Assembly of EHFs [169] 57
Figure 4-10: Schematic of the TRNSYS model 58
Figure 5-1: Accuracy under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 61
Figure 5-2: Recall under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 63
Figure 5-3: Specificity under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 64
Figure 5-4: Accuracy rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 66
Figure 5-5: Recall rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 67
Figure 5-6: Specificity rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes 68
Figure 5-7: Ratios of PP, PN, NP, and NN among the training dataset under different pre-processing methods 70
Figure 5-8: Accuracy for different modes 72
Figure 5-9: Recall for different modes 74
Figure 5-10: Specificity for different modes 76
Figure 5-11: Accuracy rate for different modes 78
Figure 5-12: Recall rate for different modes 79
Figure 5-13: Specificity rate for different modes 80

Figure 5-14: Absolute difference between recall and specificity for different modes 82
Figure 5-15: Correlation matrix of input features and outputs 84
Figure 5-16: Effect of MRDP on the predictive accuracy in terms of MSE during (a) model training and (b) model validation 85
Figure 5-17: Mean measured NHEC and mean predicted NHEC for conditions S=Positive and S=Negative during model training 85
Figure 5-18: Effect of MSEP on the predictive accuracy in terms of MSE during (a) model training and (b) model validation 86
Figure 5-19: Effect of MRDC on the predictive accuracy in terms of MSE during (a) model training and (b) model validation 86
Figure 5-20: Effect of MRDC on the (a) abs(MRE) and (b) abs(MRE) rate during model training 87
Figure 5-21: Effect of MRDC on the (a) abs(MRE) and (b) abs(MRE) rate during model validation 87
Figure 5-22: Effect of MSEC on the predictive accuracy in terms of MSE during (a) model training and (b) model validation 87
Figure 5-23: Effect of in-processing methods on the predictive fairness in terms of MSE rate during (a) model training and (b) model validation 88
Figure 5-24: Effect of MRDP on the predictive accuracy in terms of MAE during (a) model training and (b) model validation 88
Figure 5-25: Effect of MSEP on the predictive accuracy in terms of MAE during (a) model training and (b) model validation 88
Figure 5-26: Effect of MRDC on the predictive accuracy in terms of MAE during (a) model training and (b) model validation 89
Figure 5-27: Effect of MSEC on the predictive accuracy in terms of MAE during (a) model training and (b) model validation 89
Figure 5-28: Effect of in-processing methods on the predictive fairness in terms of MAE rate during (a) model training and (b) model validation 90
Figure 5-29: Comparison between y and y when using (1) MSE or (2) MAE as the loss function 91
Figure 5-30: General procedure of a basic GA 92
Figure 5-31: $X_{candidate}$ distribution of energy prediction models 93
Figure 5-32: Validation data distribution of energy prediction models 93
Figure 5-33: Residual difference distribution during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 95
Figure 5-34: Residual difference distribution during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 95
Figure 5-35: Residual difference distribution for Y=0 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 96
Figure 5-36: Residual difference distribution for Y=0 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 96
Figure 5-37: Residual difference distribution for Y=24.77 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 97

Figure 5-38: Residual difference distribution for Y=24.77 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 97
Figure 5-39: Residual difference distribution for Y=41.28 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 98
Figure 5-40: Residual difference distribution for Y=41.28 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS 98
Figure 5-41: Predictive accuracy of air temperature during (a) model training and (b) model validation..... 99
Figure 5-42: Predictive recall of air temperature during (a) model training and (b) model validation 100
Figure 5-43: Predictive specificity of air temperature during (a) model training and (b) model validation..... 100
Figure 5-44: Daily heating cost of different controllers 100
Figure 5-45: Average energy consumption of different controllers 101
Figure 5-46: Number of hours that indoor air temperature is lower than 21 °C..... 101
Figure 5-47: Hourly Set-point temperature of MPC_ReferenceCase..... 102
Figure 5-48: Hourly Set-point temperature of MPC_RS..... 102
Figure 5-49: Hourly Set-point temperature of MPC_RPS..... 102
Figure 5-50: Number of hours that indoor air temperature is lower than 21 °C when setting 21°C as the lower bound in MPCs 103
Figure 5-51: Daily heating cost when setting 21°C as the lower bound in MPCs 103
Figure 5-52: Average energy consumption when setting 21°C as the lower bound in MPCs ... 104
Figure 5-53: Average energy consumption when maximizing peak shifting by MPCs 105
Figure 5-54: Daily heating cost when maximizing peak shifting by MPCs..... 105
Figure 5-55: Hourly Set-point temperature of MPC_ReferenceCase using maximizing peak shifting as the objective function 106
Figure 5-56: Number of hours that indoor air temperature is lower than 21 °C when maximizing peak shifting by MPCs..... 106
Figure 5-57: Energy consumption per period simulated by TRNSYS with (a) 1-hour time interval and (b) 5-min time interval 107
Figure C1: Energy predictive result of SVM with 'poly' kernel with inputs $Tset, i$, $Tset, i-1$, $Tambient, i$,$Tambient, i-1$, and $Qi-1$.....130
Figure C2: Energy predictive result of SVM with 'poly' kernel with inputs $Tset, i$ and $Tambient, i$ 130
Figure C3: Energy predictive result of SVM with 'poly' kernel with inputs $Tset, i$, $Tambient, i$, and $Tambient, i-1$..... 131
Figure C4: Energy predictive result of SVM with 'poly' kernel with inputs $Tset, i$, $Tset, i-1$, and $Tambient, i$ 131
Figure C5: Energy predictive result of SVM with 'linear' kernel with inputs $Tset, i$, and $Tambient, i$ 131
Figure C6: Energy predictive result of SVM with 'linear' kernel with inputs Tset, i, T$set, i-1$, and T$ambient, i$..... 131

Figure C7: Energy predictive result of SVM with 'linear' kernel with inputs $Tset, i$, $Tset, i-1$, $Tambient, i$, and $Tambient, i-1$ 132
Figure C8: Energy predictive result of SVM with 'linear' kernel with inputs Tset, i, T$ambient, i$, and T$ambient, i-1$..... 132

LIST OF TABLES

Table 2-1: Strengths and weaknesses of feature selection methods 12
Table 2-2: Confusion matrix 21
Table 2-3: Summary of existing studies focusing on developing data-driven building models... 24
Table 2-4: Summary of studies on in-processing methods 29
Table 3-1: PP, PN, NP and NN defined by the protected attribute and output labels……31
Table 3-2: Conditions defined by a i-class prediction problem with a j-class protected attribute 32
Table 4-1: Statistical distribution of the collected data…….44
Table 4-2: Description of sensors [19] in A-1 44
Table 4-3: Description of study cases in A-1 45
Table 4-4: Description of sensors in A-2 49
Table 4-5: Description of data modes 50
Table 4-6: Description of study cases in A-2 52
Table 4-7: Search space for hyperparameter optimization 53
Table 4-8: Selected input features and their correlation with the output attribute 55
Table 4-9: Description of study cases for Task B 56
Table 4-10: Hyperparameters of DE 56
Table 4-11: Electricity price during winter implemented in Montreal, Canada [167] 57
Table 4-12: Thermophysical properties of floor layers 57
Table A1: Hyperparameters for SVM classifiers…….124
Table A2: Hyperparameters for ANN classifiers 125
Table A3: Hyperparameters for Logistic Regression classifiers 125
Table A4: Hyperparameters for Gaussian Naive Bayes classifiers 125
Table B1: Detailed correlation matrix values……..126
Table C1: Air temperature predictive result of different SVM models……...132

LIST OF SYMBOLS AND ABBREVIATIONS

Symbols

c	Constant
D	Unprotected attributes
h	Output of the hidden layer in ANN
NN	The condition with Negative protected attribute and Negative actual class label
$\|NN\|$	The number of data points in NN of $X_{candidate}$
$\|NN\|_{design}$	The expected number of data points in NN of $X_{designed}$
NP	The condition with Negative protected attribute and Positive actual class label
$\|NP\|$	The number of data points in NP of $X_{candidate}$
$\|NP\|_{design}$	The expected number of data points in NP of $X_{designed}$
p	Order of AR
PN	The condition with Positive protected attribute and Negative actual class label
$\|PN\|$	The number of data points in PN of $X_{candidate}$
$\|PN\|_{design}$	The expected number of data points in PN of $X_{designed}$
PP	The condition with Positive protected attribute and Positive actual class label
$\|PP\|$	The number of data points in PP of $X_{candidate}$
$\|PP\|_{design}$	The expected number of data points in PP of $X_{designed}$
$p\_negative$	The possibility of classifying a data point as negative
$p\_positive$	The possibility of classifying a data point as positive
q	Order of MA
r_{xy}	Pearson correlation coefficient between input feature x and target output y
rho_{xy}	Spearman's ranking correlation between input feature x and target output y
S	Protected attributes
$X_{candidate}$	Candidate training dataset
$X_{designed}$	Designed training dataset
$\| X_{designed} \|$	The number of data points in $X_{designed}$
$\overline{x}$	Mean value of input feature
x_i'	Input rank of i-th individual sample points
$\overline{x'}$	Mean rank values of input feature
Y	Class label of the training point
$\hat{Y}$	Predicted class label
y_i	Target output of i-th sample point
$\overline{y}$	Mean value of target (measured) output
y_i'	Target output rank of i-th sample point
$\overline{y'}$	Mean rank values of target output
$\hat{y}$	Predicted output

w_0	Bias term
w	Weight matrix
$\varphi_1, \dots, \varphi_p$	Weights for AR
$\theta_1, \dots, \theta_q$	Weights for MA
ε	White noise
$\emptyset$	Activation function of output layer in ANN
σ	Activation function for the hidden layer in ANN

Abbreviations

AE	Autoencoder
ANN	Artificial Neural Network
AHUs	Air Handling Units
AR	AutoRegressive Model
ARIMA	AutoRegression Integrated Moving Average
ARMA	AutoRegression-Moving Average
ASHRAE	The American Society of Heating, Refrigerating and Air-Conditioning Engineers
BMS	Building Management Systems
CNN	Convolutional Neural Networks
CV(RMSE)	Coefficient of Variation of the Root Mean Square Error
DDBMs	Data-Driven Buildings Models
DT	Decision Tree
DNN	Deep Neural Networks
ELM	Extreme Learning Machine
FDD	Fault Detection and Diagnosis
FEMP	Federal Energy Management Program
GAN	Generative Adversarial Network
GBDT	Gradient Boosting Decision Trees
IPMVP	International Performance Measurement and Verification Protocol
HVAC	Heating Ventilation and Air-Conditioning
HEMS	Home Energy Management System
IoT	Internet of Things
kNN	k-Nearest Neighbor
LR	Logistic Regression
LSTM	Long Short-Term Memory
MA	Moving Average Model
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MBE	Mean Bias Error
MLP	Multi-Layer Perceptron
MPC	Model Predictive Controller
NMBE	Normalized Mean Bias Error
NB	Naïve Bayes
PCA	Principal Component Analysis

PLSR	Partial Least Squares Regression
R^2	R Square
RF	Random Forests
RICNN	Recurrent Inception Convolution Neural Network
RMSE	Root Mean Square Error
RNN	Recurrent Neural Networks
RT	Regression Tree
RPS	Reversed Preferential Sampling
RS	Random Sampling
SBS	Sequentially Balanced Sampling
SMOTE	Synthetic Minority Oversampling Technique
SPS	Sequential Preferential Sampling
SS	Sequential Sampling
SVR	Support Vector Regression
XGB	Extreme Gradient Boosting

INTRODUCTION

1.1. BACKGROUND AND MOTIVATION

In recent years, buildings and construction sectors have become data-rich, due to the rapid popularity of the Internet of Things (IoT) and building management systems (BMS) [1]. This means that plenty of data (such as indoor environment parameters, occupancy-related data, energy consumption, equipment and device operational data, etc.) has been dynamically collected from buildings [2]. These data could be the basis for developing novel and efficient data-driven building models (DDBMs) for several purposes, such as energy, indoor air temperature predictions [3–6], indoor air quality, thermal comfort, energy efficiency enhancement strategies [7], occupancy modelling [8–12], fire hazard [13], fault detection and diagnosis[14,15]. In general, DDBMs could be classified as classification algorithms and regression algorithms: classification algorithms predict discrete class labels, while regression algorithms predict continuous quantity outputs.

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 [16]. Among these features, occupancy-related data (e.g., number of occupants, motion status, types of occupant activities, etc.) may be denied from inputs, due to the concern of privacy issues and fairness problems. Previous studies mentioned that occupancy data might infer occupants' location and behavior [17–19]. Thus, occupants must be informed, and appropriate permission must be obtained before data logging and usage. Information/data that the occupants might not be willing to share is termed as protected or sensitive information and such attributes are defined as protected attributes. Using protected attributes as inputs, or providing different predictive accuracy for different groups among the protected attributes, may cause fairness problems. For example, while developing data-driven energy predictive models for users from different occupations, the predictor may be more accurate for occupants from some occupations as their energy usage patterns are more regular and easier to be discovered. However, it would be unfair to other occupants as the predictive accuracy is low.

Besides, existing studies on DDBMs mainly focus on improving the model predictive accuracy to ensure that the predictive result could reasonably represent indoor environment parameters, energy consumption patterns, or device operational status. Commonly used accuracy improvement methods mainly include improving the representativeness of the training dataset and strengthening the structure of data-driven models. For instance, González-Vidal et al. [20] proposed a feature selection structure to improve the mean absolute error (MAE) by 42.28% and root mean square error (RMSE) by 36.62% for energy prediction regression models. Esrafilian-Najafabadi and Haghighat [21] proposed a multi-objective genetic algorithm to maximize the occupancy predictive accuracy while selecting the most effective input features. The proposed method increased the median accuracy of long-term occupancy prediction by 4.81%.

However, a DDBM with high overall predictive accuracy could not ensure a good predictive performance for all users/occupants or scenarios/operation situations [19]. This is mainly caused by the problem of imbalanced data. For instance, available features could be different for different users because of the difference in installed data collection devices or different opinions for information sharing: newly constructed smart buildings are more likely to be equipped with more data collection devices, e.g., presence detector, temperature sensor, and CO_2 detector, etc., than

aged buildings; some occupants may approve of the authority for the data-driven predictors to utilize their collected data for prediction, while others might deny it due to the concern of privacy. Meanwhile, the amount of data collected during different periods or operation scenarios could be different for one user. For instance, if one occupant is retired and stays at home most of the time, data for that user would be mostly collected during the occupied time. On the other hand, for a working couple or user, the collected data would distribute more evenly between occupied and unoccupied hours.

In reality, different volumes of training dataset under various conditions may cause better predictive accuracy under majority conditions and poor accuracy under other conditions [20]. For instance, HVAC system operational data are collected mostly under normal conditions. Few are collected during faulty scenarios. As a result, even if a model trained on these data works well to predict the normal HVAC operation status, the problem remains that it wrongly predicts faulty scenarios as normal [1].

Some existing studies on DDBMs tried to improve the predictive accuracy for minority conditions through data-preprocessing methods or algorithm-based methods. For example, data pre-processing methods have been used to oversample or generate faulty data to improve the fault detection and diagnosis (FDD) accuracy [22–24]. Algorithm-based methods, also called cost-sensitive learning algorithms, are used to assign higher misclassification costs to data from minority classes, and thus, force the classifier concentrate on minority classes, such as faulty conditions [25–27].

However, above-mentioned existing studies only focus on minority classes among the outputs, instead of minority conditions defined by other attributes. Besides, they could not reveal the fairness problems caused by the different predictive accuracy between users or conditions. For example, if a DDBM yields a more accurate prediction for a certain group of users, it would be unfair to others who receive a less accurate prediction; if a predictor is more accurate during certain periods than other times 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 intermittent poor performance.

In fact, fairness-aware machine learning has been a hot topic in recent years. It has been applied to non-discriminatory hiring [29–31], risk assessment for sentencing guidance [32,33], income prediction [33], loan allocation [34,35], and graph embedding [36]. However, fairness problems among DDBMs have not yet been investigated, given the gap between disciplines.

Based on the literature, there are mainly three types of fairness concepts: *Type I:* The predicted output is independent of the protected attribute(s). *Type II:* Some performance measures (e.g., accuracy) are similar across classes/conditions defined by the protected attribute(s). *Type III*: Predictive outcomes should be independent of the predictive probability score of different classes/conditions defined by the protected attribute(s). Note that fairness concepts could be further categorized foreach category. However, it is to be mentioned that different types of fairness could not be achieved at the same time. Therefore, researchers are recommended to clearly define which type of fairness they plan to achieve, and accordingly design proper fairness improvement methods.

Achieving *Type I* or *Type II* fairness for DDBMs would be an interesting topic, as *Type I* fairness could allow the users to do authority management to protect their privacy information, while *Type II* fairness would be beneficial for ensuring uniform predictive performance for different users or conditions defined by the protected attribute(s). Among the different fairness concepts defined in the literature, the current study is mainly focused on investigating the possibility of achieving *Type II* fairness for DDBMs.

In general, widely applied fairness-improvement techniques can be grouped into 1) Pre-processing: preprocesses training data to remove discrimination before the training phase; 2) In-processing: adds fairness-related constraints or penalties to the model's optimization objective during the training phase [37]; and 3) Post-processing: changes a classifier's predictive results to achieve fairness [38]. Among these methods, pre-processing methods are more applicable and easier to implement, because these methods do not require the modification of the model structure. In-processing methods could achieve specific fairness measures chosen by the programmer while preserving high accuracy, while post-processing lacks the flexibility in achieving accuracy and fairness trade-off [38]. Therefore, this study will concentrate on pre-processing methods and in-processing methods for fairness improvement of DDBMs. To be more specific, pre-processing methods will be proposed to re-balance the training dataset for classification algorithms to improve *Type II* fairness, while in-processing methods will be implemented to regression problems to achieve the trade-off between *Type II* fairness and accuracy.

Furthermore, DDBMs could be integrated into model predictive controllers (MPCs) to optimally operate devices in buildings based on the predicted values to achieve peak shifting/cost-saving [39–41]. To investigate the applicability of fairness-aware DDBMs in MPCs, the developed fairness-aware data-driven models will be integrated into MPC to control a heating system in a residential building. The effect of fairness improvement methods on the control performance will be compared with traditional controllers.

1.2. OBJECTIVES AND CONTRIBUTIONS

To meet the research gap on fairness-aware DDBMs and their application in MPC, the major objective of this thesis is to improve the predictive fairness for DDBMs while preserving an acceptable predictive accuracy and make the fairness-aware MPC to optimize the control signal for building devices based on uniform prediction. To achieve this objective, the thesis could be separated into the following specific works:

1. Introduce and investigate the fairness concept in building engineering domain without significantly decreasing accuracy. To be specific, fairness improvement methods is proposed for both classification and regression problems in building engineering domain. The trade-off between fairness and accuracy of DDBMs is investigated. This part could be further separated into two tasks:
   Task A: Propose pre-processing fairness improvement methods for classification models and implement the proposed methods to process the training dataset before training classifiers to predict the discrete outputs.
   Task B: Propose in-processing fairness improvement methods for regression models, and subsequently, investigate effect of proposed methods on the predictive accuracy and fairness of linear regression models.

2. Integrate the fairness-aware DDBMs into MPC to control systems/devices in buildings in order to maximize cost-saving, and/or peak shaving while ensuring a uniform predictive performance. This could be conducted through the third task of this research:
   Task C: Develop fairness-aware data-driven based MPC for buildings. The control performance could be validated by simulating the proposed MPC on a TRNSYS building model.

This study is the first work introducing fairness concepts into the building domain. Considering fairness in DDBMs could either help users to achieve authority management or ensuring the users to receive fair service provided by the developed models. To improve fairness in terms of achieving uniform predictive performance in classification problems while preserving predictive accuracy, this study proposes four pre-processing methods, namely sequential sampling (SS), sequentially balanced sampling (SBS), reversed preferential sampling (RPS), and sequential preferential sampling (SPS), to sample the training dataset into a balanced dataset. To obtain fairness in regression problems, this study proposes four in-processing methods, including mean residual difference penalized regression (MRDP), mean square error penalized regression (MSEP), mean residual difference constrained regression (MRDC), and mean square error constrained regression (MSEC), to incorporate fairness-related constraints or penalty into the optimization objective of the model during model training. Finally, these fairness improvement methods are integrated into the data-driven models in MPCs to control systems/devices in buildings. Therefore, the primary contribution of this study is to propose new techniques for improving the predictive fairness of DDBMs to ensure uniform predictive performance. Besides, this study investigates the trade-off between fairness and accuracy of DDBMs, and integrates the proposed fairness-aware DDBMs into MPCs.

1.3. THESIS OUTLINE

Chapter 1 introduces the rapid development of DDBMs and motivation to improve these models' fairness while preserving the predictive accuracy and its potential application in MPCs. Then, the main objective and three tasks of this thesis are proposed, followed by the thesis outline.

Chapter 2 reviews existing DDBMs in terms of commonly used input features, feature extraction methods, supervised data-driven algorithms, factors reflected from DDBM outputs, and performance criteria for evaluating the predictive result. Through summarizing DDBMs in these aspects, the requirement for improving predictive fairness is proposed. Then, fairness definitions are summarized into three categories. Benefits of considering fairness for DDBMs are explained. Then, commonly used fairness improvement methods are reviewed.

Chapter 3 presents the proposed methodologies for the three tasks: In task A, four pre-processing methods are proposed to make the training dataset of classification problems to be balanced among conditions defined by the protected attribute and output labels. In task B, four in-processing algorithms are proposed to achieve user-defined trade-off between fairness and accuracy for regression problems. In task C, the concept of fairness-aware data-driven based MPC is proposed to integrate the fairness-aware data-driven models into the MPC, so that the optimal control signal could be calculated based on uniform prediction.

Chapter 4 describes the case studies designed to investigate the performance of proposed methodologies: For task A, two cases studies are presented to apply the proposed pre-processing

methods to predict lighting status with considering motion status as the protected attribute. Case study A-1 compares the effect of proposed methods on the predictive accuracy and fairness with existing methods, while case study A-2 investigates the generalizability of these methods. For task B, the proposed in-processing methods are applied to predict the energy consumption of an apartment by considering motion status as the protected attribute. The effect of these methods on predictive accuracy in terms of MSE and MAE and on predictive fairness in terms of MSE rate and MAE rate is analyzed. For task C, fairness-aware data-driven based MPCs are applied to control the heating system of a residential building located in Montreal, Canada, in order to save the heating cost by shifting heating load from peak periods to off-peak periods.

Chapter 5 analyzes the results for the three aforesaid tasks. For task A and task B, the results are present in terms of predictive accuracy and fairness, while for Task C the control performance of MPC is analyzed in terms of energy consumed during different periods and cost saving resulted from the shifted load.

Chapter 6 summarizes the major conclusions, outcomes and findings from the study. Also, the recommendations for future words are present.

CHAPTER TWO: LITERATURE REVIEW

In this chapter, a detailed literature review is first carried out for DDBMs. The review is organized by the order of the model development procedure. First, commonly used input features and feature selection methods are summarized. Next, supervised data-driven algorithms that discover statistical pattern from the training dataset and use input features to predict continuous or discrete outputs are reviewed. Then, aspects (such as output type, building type, scale, and temporal granularity, etc.) considered in outputs are illustrated. A summary is given to summarize the research interests in DDBMs and limitations of existing studies.

The second part of the literature review is aimed at presenting the concept of fairness-aware data-driven models and showing the benefits of considering fairness in DDBMs. Finally, existing fairness improvement methods are reviewed.

2.1. DDBMS AND THEIR DEVELOPMENT PROCEDURE

2.1.1. FEATURE ENGINEERING

Before a data-driven procedure, data is first collected from simulation, measurement/survey, or public database. Then, the data should be thoroughly processed to remove/correct the missing/incorrect data. This process is called data cleaning. Commonly utilized outlier/anomaly detection methods can be found in [42,43], while approaches to impute/replace missing data were presented [44].

After data collection and data cleaning, features contributing most to prediction results need to be constructed and extracted. Therefore, in this section, the most commonly used features for DDBMs and feature extraction methods are presented.

2.1.1.1. Feature types

Meteorological information

Meteorological information mainly includes ambient dry bulb temperature, wet bulb temperature, dew point temperature, humidity, wind speed, solar radiation, rainfall, air pressure, etc. [45].

The correlation between weather variables and building load (except heating load) has been studied by Cai et al. [46] for three buildings located in Alexandria VA, Shirley NY, and Uxbridge MA, respectively. Among these weather variables, outdoor temperature was found to be positively correlated with building load, while the relation between other variables and building load was insignificant. However, when the ambient temperature was lower than 24.4 °C, it was irrelevant to the electricity demand of residential buildings in Italy [47]. This is because the main heating fuel for homes in Italy is natural gas, while electricity is used for cooling systems. Besides ambient temperature, Solar radiation is also commonly utilized in building energy prediction, due to its significant effect on thermal demand and its accessible from weather forecasting [48].

Indoor environmental information

Indoor conditions that include set-point temperature of thermostats, indoor temperature, indoor humidity, indoor carbon dioxide concentration, etc. have been identified as a priority for residential cooling and heating load calculation [49]. Note that, unlike constant design values of indoor conditions during the design stage, these values are dynamic during reality operation. Therefore, to predict building conditions precisely, indoor environmental information needs to be considered as a potential feature.

Chammas et al. [50] considered indoor temperature and humidity when predicting energy consumption for a residential house. However, their study did not compare the importance of meteorological information, indoor conditions and time. Ding et al. [51] presented that interior variables would further improve heating load prediction accuracy. However, due to unpredictable interior temperature, the variables could not be utilized for 24 hour ahead heating demand prediction. Wei et al. [52] found that indoor relative humidity, dry-bulb temperature and carbon dioxide concentration are among the top 10 important variables for energy consumption prediction of an office building. These three features were also included for predicting desk fan usage preferences [53]. Furthermore, indoor temperature and humidity have been used as inputs in predicting air conditioning operation[54].

It is interesting to note that studies with considering set-point temperature as inputs for predicting building loads are generally aimed at developing demand response control strategy, such as in the research by Behl et al. [55]. Otherwise, studies tend to ignore the effect of set-point temperature on predictive result, even though residents in residential buildings have the ability to adjust the set-point temperature to meet their thermal comfort requirement and save energy [56].

Occupancy related data

Occupancy related data, such as number of occupants and types of occupant activities, would affect internal gain and then influence the pattern of energy usage [57,58]. Therefore, it would be a potential feature for building energy prediction.

The principal component analysis of Wei et al. [52] indicated that the number of occupants is even more important than meteorological information for energy prediction in an office building. Wang et al. [59] utilized linear regression to observe the strong linear relation between plug load power and occupant count for working days and then selected it as one of features for plug load prediction. Sala-Cardoso et al. [60] predicted the activity indicator through a recurrent neural network (RNN) and then integrated it with a power demand prediction model to improve the prediction accuracy of HVAC thermal power demand for a research building.

However, short leave of occupants would not affect the load consumption. Besides, if a public building is controlled without taking into account the occupancy status, its energy consumption might not be strongly related to occupancy patterns [61]. Furthermore, in most cases, the types of occupant activities are not flexible to be collected.

Time index

Time index means the stamps series for time, which mainly includes time of the day, day of the week, hour type (peak hour or off-peak hour), day type (weekday or weekends), calendar day, etc. The purpose of introducing time index into DDBMs is to indicate the occupancy related effect.

For instance, occupants tend to do similar activities at the same time on different days or at the same day on different weeks. Therefore, time index would be a good option when occupancy related data is unavailable.

Fan et al. [62] found that due to the similar energy consumption pattern on the same weekday, 7-days and 14-days ahead peak power demand and energy consumption were the four most important inputs for next-day energy consumption and peak power demand prediction of a commercial building. This indicates that day of the week could be selected as one of the input features able to represent similar energy consumption patterns during the same weekday. Similar justification could be used for selecting time of the day, holiday/workday, peak hour/off-peak hour as inputs.

Building characteristic data

Building characteristic features mainly include relative compactness, surface area, wall area, roof area, overall height, orientation, glazing area, heat transfer coefficient of building envelopes, absorption coefficient for solar radiation of exterior walls, window-wall ratio, shading coefficient etc. [63].

Once a building is constructed, these data would remain relatively constant. Therefore, it is meaningless to contain this information when using data-driven models to predict dynamic conditions for a specific building. However, when the study object is multiple buildings or when the objective is using the known load of an existing building to predict the load of a new building, building characteristic features would be beneficial. Seyedzadeh et al. [64] drew feature correlation maps for building characteristic and building heating/cooling loads, and utilized these features as inputs for DDBMs. Wei et al. [65] predicted annual heating, cooling and electricity intensity for different office buildings based on input factors relevant to building form, e.g. aspect ratio, window-wall ratio, number of floors, orientation and building scale. Talebi et al. [5] utilized thermal mass as one of the input features to predict the heating demand of a district. Similar studies could also be found in references [66–69].

Socio-economic information

Socio-economic information shows the socio-economic situation of the studied area [70]. It mainly includes income, electricity price, GDP, population, etc.

These features are commonly utilized to do long term (e.g. months or years) load prediction for large scale (e.g. district, region or country) [71]. For instance, He et al. [72] found that average electricity price and number of electricity customers/permanent residents could be important in forecasting annual electricity consumption of a city. He et al. [73] identified that historical energy consumption, average annual GDP growth rate and total GDP were the key factors for annual energy consumption prediction of the Anhui province, China. However, GDP was revealed by Beyca et al. [74] to be insignificant in natural gas consumption prediction of Istanbul, while the price of natural gas and population showed a high correlation to the prediction result.

Historical data

Due to the thermal mass of building envelopes, building loads could be affected by historical factors, such as historical values of exogenous features or historical energy consumption. For example, Wang et al. [75] found that the historical heating consumption is the leading factor for

heating demand prediction of district heated apartment buildings. Similarly, Ahmad et al. [76] concluded that previous hour's electricity consumption was more important than meteorological information, time index and occupancy related data for 1-hour-ahead HVAC energy consumption prediction of a hotel in Spain. Ding et al. [77] proved that historical cooling capacity data is the most important data for cooling load prediction of an office building. Huang et al. [78] proposed a historical energy comprehensive variable named EVMA to improve the energy demand prediction accuracy for residential buildings based on ensemble methods. Furthermore, He et al. [73] found the historical annual energy consumption of Anhui province in China significantly affected its future annual energy consumption. Due to the ability to increase the prediction accuracy of dynamic loads, the interests in applying historical data as features for data-driven models have been increasing in recent years.

2.1.1.2. Feature extraction methods

Properly constructed features could reduce the computation time of a DDBM without sacrificing prediction accuracy [79]. The commonly applied feature extraction methods with the ability to select useful features or reconstruct feature vectors are introduced as below:

Variable ranking

The idea of variable ranking is to choose the desired number of features most relevant to the output by a scoring function.

One popular quick and easy use function for variable ranking is the Pearson correlation coefficient (see Equation 2-1 [80]). This method determines the strength and direction of the linear relationship between two variables. To calculate the monotonic relationship between two continuous or ordinal variables, Spearman's rank correlation (see Equation 2-2 [81]) could be utilized. Note that Spearman's rank correlation between two variables equals to the Pearson correlation of rank values of these two variables.

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

where:

r_{xy} is the Pearson correlation coefficient between input feature *x* and target output *y*;

n is sample size;

x_i, y_i are the *i*-th individual sample points;

$\overline{x}$, $\overline{y}$ are the mean value of input feature and target output, respectively.

$$rho_{xy} = \frac{\sum_{i=1}^{n}(x_i'-\overline{x'})(y_i'-\overline{y'})}{\sqrt{\sum_{i=1}^{n}(x_i'-\overline{x'})^2}\sqrt{\sum_{i=1}^{n}(y_i'-\overline{y'})^2}} \tag{2-2}$$

where:

rho_{xy} is the Spearman's ranking correlation between input feature *x* and target output *y*;

n is sample size;

x_i', $y_i'$are the ranks of i-th individual sample points;

$\overline{x'}$, $\overline{y'}$are the mean rank values of input feature and target output, respectively.

One challenge for variable ranking is to determine the desired number of features, which could be considered as a hyperparameter (i.e. a pre-defined parameter which affects the running time of feature engineering process and prediction accuracy of the developed DDBMs [82]). Another drawback of variable ranking is that it could only calculate the relationship between individual variables and output, instead of between subsets of features and output. For instance, Aaron et al. [83] utilized standardized association factors to find out that dry bulb temperature, wet bulb temperature and enthalpy are most relevant to building electricity use. However, they failed to estimate the possible inter-relevance between temperatures and enthalpy. To solve this problem, filter and wrapper methods could be utilized to select the best subset.

Filter and Wrapper methods

Both filter and wrapper methods could be utilized for best-subset selection, which means they could consider the interrelationship between features. Among them, filter methods evaluate the importance of individual or subset of features through statistical measures. Filter methods have two different categories: Rank Based (i.e. variable ranking) and Subset Evaluation Based [84]. The filter methods mentioned here refer to the later types. Unlike filter methods, wrapper methods consider all possible subsets of features and measure their performance through supervised learning algorithms.

Filter methods are more efficient than wrapper techniques in terms of computational complexity, while wrapper methods are more stable [84]. Yuan et al. [85] applied partial least squares regression (PLSR) and random forests (RF) to rank the top 10 important input features for predicting weekly coal consumption for space heating. The reason for employing these two filter methods is that they can consider the inter-dependence between input variables. Then, they utilized a SVM based wrapper method to evaluate the proper number of features. The prediction accuracy based on the selected top 6 features met the requirement of ASHRAE Guidelines 14-2014 [86].

Embedded method

Unlike the wrapper method, which selects the best subset with the highest prediction performance in a specific learning algorithm, the embedded method integrates feature selection into the learning algorithm. For instance, regularization added to data-driven models could be considered as an embedded method. Jain et al. [87] employed Lasso, a linear regression model which adds an L1 penalty to the squared error loss, to forecast energy consumption of a multi-family residential building. Their results confirmed that in certain cases, Lasso could outperform a Support Vector Regression (SVR) model that did not consider feature selection.

One challenge of the embedded method is that the selected regularization method should adapt the optimization procedure to ensure the existence of optimum solution. Furthermore, this method could not present the importance of features.

Principal component analysis (PCA)

The idea of traditional PCA is to project features into a lower-dimensional sub-space with linearly uncorrelated variables [88], while kernel PCA utilizes a kernel function to map nonlinear related original inputs into a new feature space and then perform a linear PCA in this new space [89].

Li et al. [90] compared the building load prediction accuracy between SVR with PCA, SVR with kernel PCA, and SVR without any feature selection techniques. Their results illustrate that SVR with PCA increased the cooling load prediction accuracy compared to the SVR model, while kernel PCA could further improve prediction performance.

Furthermore, Yuldiz et al. [91] showed the way to apply PCA to tackle the multi-collinearity problem in original input variables, and gave a detailed description about how to determine the dimension of reduced feature space. A similar application of PCA has been introduced by Wei et al. [52]. From these studies, one limitation of PCA has been revealed: the dimension of final feature space needs to be manually selected. Besides, when applying kernel, the type of kernel function should be determined.

Autoencoder (AE)

AE is a type of unsupervised artificial neural network (ANN) that can learn a compressed nonlinear representation of the input data. As shown in Figure 2-1, an autoencoder generally consists of two networks: (1) Encoder: maps the original inputs to a compressed low dimension; (2) Decoder: recovers original inputs from the compressed representation.

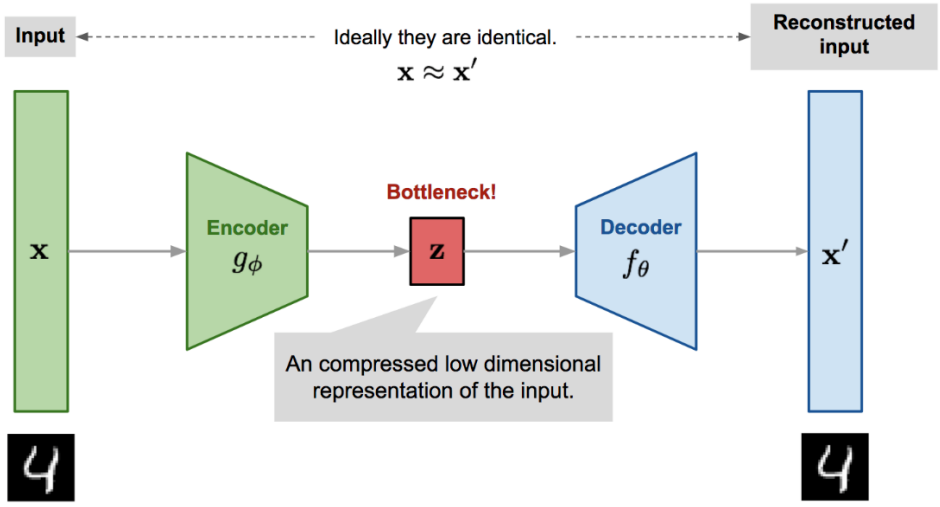


Figure 2-1: Illustration of autoencoder model architecture *[92]*

Fan et al. [93] compared three types of deep learning based feature selection methods (i.e. fully connected AE, convolutional AE and generative adversarial networks) to variable ranking and PCA. The result shows that the deep learning-based feature selection method enhanced the one-step-ahead cooling load prediction performance for an educational building. Furthermore, Mujeeb and Javaid [94] proposed an efficient sparse autoencoder as feature extraction method, and then utilized the compressed feature space as inputs for an non-linear autoregressive network. The proposed method decreased forecasting error of the non-linear autoregressive network for regional load forecasting.

Note that the application of autoencoder for feature extraction in the field of DDBMs is still uncommon. One reason is that the dimension of original input features is usually small, thus, AE would be computing intensively compared to other feature extraction methods. Following the explosive growth of collected data and implementation of deep learning, the interests in AE would increase.

Strengths and weaknesses of previous introduced feature selection methods are summarized in Table 2-1.

Table 2-1: Strengths and weaknesses of feature selection methods

Type of feature selection	**Strengths**	**Weaknesses**
Variable ranking	1. Fastest and easiest to use 2. Quantitatively calculate the relevance between individual variables and outputs	1. Hard to determine the number of desired features 2. Unavailable for considering the effect of inter-relevance between features on the output 3. Could not select the best subset
Filter method	1. Fast and easy to use 2. Subset selection 3. Robust to overfitting	1. Less stable
Wrapper method	1. Subset selection that considers inter-relevant of input features 2. More stable	1. Computational expensiveness 2. High risk of overfitting
Embedded method	1. Easy to use 2. Unnecessary to eliminate features	1. Unable to quantitatively present the importance of features
PCA	1. Relatively easy to use 2. Effective when the original feature space dimension is not too large 3. Unnecessary to eliminate features	1. Hard to determine the number of desired features 2. For kernel PCA, kernel function needs to be properly selected
AE	1. Learn nonlinear representation of original input 2. More powerful for compressing the dimension of features with lower loss of information	1. Computational expensiveness

2.1.2. SUPERVISED DATA-DRIVEN ALGORITHMS

2.1.2.1. Linear regression

Linear regression is one of the traditional statistical approaches to study the relationship between a dependent continuous variable (i.e. response or output) and one or more independent variables (i.e. predictor or input features). Its general form is shown in Equation 2-3.

$$\hat{y} = w_0 + wx \tag{2-3}$$

where,

$\hat{y}$ is the predicted output,

w_0 is the bias term,

w is a weight matrix for features x.

Note that the general form could only discover the linear relationship between features and output. To extend the applicability of linear regression, the input variables could be converted to other forms through different active functions, such as polynomial (Equation 2-4) or natural logarithm function (Equation 2-5).

$$\hat{y} = w_0 + wx^m \tag{2-4}$$

where m means m-th polynomial.

$$\hat{y} = w_0 + wlog(x) \tag{2-5}$$

The main advantage of linear regression is that it is very easy to use and intuitive to understand. The contribution of individual variables to the prediction result could be directly found from the weight matrix. Besides, extended linear regression could be applied to solve nonlinear problems. However, its limitations should also be noted: (1) The general form of linear regression could not consider nonlinear relationships between inputs and outputs; (2) The prediction performance of extended linear regression is highly dependent on the proper selection of active function, which could be a significant challenge; (3) Multicollinearity of input features would hurt the prediction result of linear regression. Therefore, feature extraction methods are recommended to be applied before developing linear regression models.

2.1.2.2. Logistic regression (LR)

Logistic Regression (LR) is a kind of generalized linear regression model used to solve categorical problems. It is widely used in two-class classification problems. Its fundamental function is to predict the possibility of an object belonging to a positive class using a logistic function (see Equation 2-6 [95]).

$$p(\hat{y} = 1) = \frac{1}{1+e^{-(x-\mu)/s}} \tag{2-6}$$

where $p(\hat{y} = 1)$ is the probabilities that $\hat{y}$ belongs to class 1; μ is a location parameter, it equals to the midpoint where $p(\hat{y} = 1|\mu)$=0.5; s is a scale parameter.

2.1.2.3. Time series analysis

The most commonly used methods for time series analysis are AutoRegressive-Moving Average (ARMA) and AutoRegressive Integrated Moving Average (ARIMA) [10]. ARMA (see Equation 2-7) mainly includes two parts: an autoregressive model (AR) with order p and a moving average model (MA) with order q.

$$\widehat{y_t} = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i y_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{2-7}$$

where φ_1,…,φ_p are weights for AR, θ_1,…,θ_q are weights for MA, ε is white noise, c is a constant.

ARMA could only handle stationary time series. When predicting nonstationary time series, ARIMA would be a better choice since it integrated an initial differencing step to eliminate the non-stationary [71].

ARMA and ARIMA show the ability to consider the effect of historical data, thus, their prediction performance would be acceptable if the output is highly impacted by previous values.

However, determining the orders for AR and MA models and the times of initial difference would be a challenge.

2.1.2.4. Naïve Bayes (NB)

Naïve Bayes (NB) classifiers are a set of simple classifiers that predict discrete outputs by using Bayes' algorithm (see Equation 2-8) with the 'naive' assumption of conditional independence between attributes given the class label value [96]. Naïve Bayes classifiers include three main types: Gaussian NB, Multinomial NB, and Bernoulli NB. This study uses Gaussian NB.

$$p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)} \tag{2-8}$$

where C_k is the k-class outputs.

The main advantage of NB is that a small number of training points is enough to estimate the parameters for classification.

2.1.2.5. Decision tree (DT)/Regression tree (RT)

Decision tree (DT), see Figure 2-2, is a kind of classification model that starts with a root node where the input data is split into different internal nodes or leaf nodes. For internal nodes, the inputs are continuously split into subsets, while leaf nodes represent the output of the model. This implies that there is a chance that the DT makes predictions without involving the entire feature space. Furthermore, Regression tree (RT) is an extension of DT with continuous target variables.

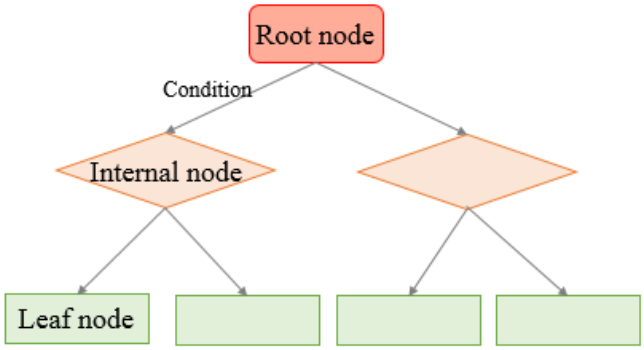


Figure 2-2: Schematic of decision tree

The main advantage of DT and RT is easy to understand and interpret due to the fact that they could be displayed graphically [40], [72]. Besides, they could outperform traditional statistical methods once proper features are selected [73]. The disadvantages of them are: (1) They could be sensitive to small changes of data; (2) Their structure fails to determine smooth and curvilinear boundaries. Furthermore, to enhance their prediction performance, groups of DT or RT could be combined as an ensemble model.

2.1.2.6. Support vector machine (SVM) / Support vector regression (SVR)

Support vector machine (SVM) maximizes the margin between different categories as shown in Figure 2-3, while support vector regression (SVR) is a regression application of SVM. For SVR,

the goal is to find a linear regression function that could predict the result with acceptable deviation from the actual target [97]. For nonlinear regression problems, a kernel function should first be selected to map the original inputs to a high-dimensional feature space, and then apply the SVR. Therefore, one challenge of SVR is the proper selection of kernel function.

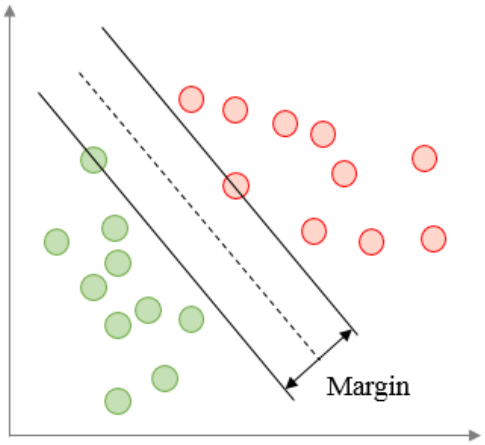


Figure 2-3: Schematic of margin between different categories

The advantages of SVM/SVR are: (1) It has the ability to solve global minima instead of local minima[98]; (2) Its computational complexity is not determined by the dimensionality of feature space[99]; and (3) Its prediction performance is not sensitive to the noisy data.

2.1.2.7. Artificial neural network (ANN)

ANN is a machine learning technique inspired by biological neural network [100]. As shown in Figure 2-4, a typical ANN usually consists of three layers: input layer, hidden layer and output layer. The training goal of an ANN model is to learn the weights and bias (as shown in Equation 2-9) with proper number of neurons and hidden layers as well as activation functions. Note that although ANN with a single hidden layer can present any Boolean function and ANN with two hidden layers shows the ability to train any function to arbitrary accuracy, the number of hidden layers should be carefully selected to achieve better accuracy with fewer neurons. Furthermore, once the number of hidden layers is increased, the ANN could be considered as deep learning.

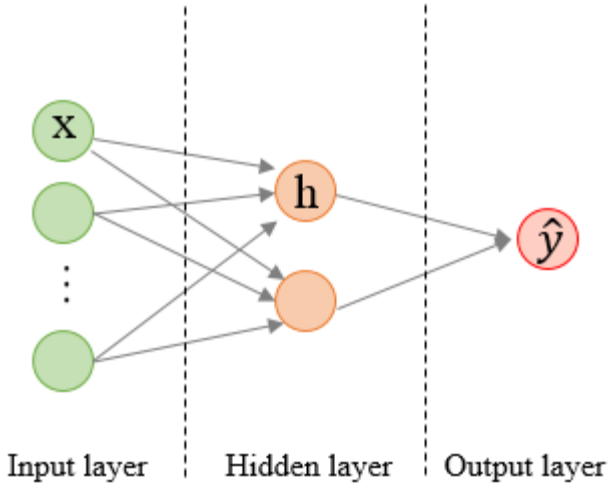


Figure 2-4: Schematic of typical ANN

$$\hat{y} = \emptyset(w_{out}h + b_{out}) = \emptyset[w_{out}\sigma(wx + b) + b_{out}] \quad (2\text{-}9)$$

where:

$\emptyset$ is the activation function of output layer;

h is the output of the hidden layer, $h = \sigma(wx + b)$;

σ is the activation function for the hidden layer;

w_{out} and w are the weight matrix;

b_{out} and b are the bias terms.

The advantages and disadvantages of ANN have been described in References [101] and [102]. The main advantage of ANN is the ability to deal with non-linear problems without expertise, while the main disadvantage is the long time required for training models with large number of networks.

2.1.2.8. Deep learning

Deep learning based on ANN includes three categories: deep neural networks (DNN), convolutional neural networks (CNN) and recurrent neural networks (RNN).

(1) DNN

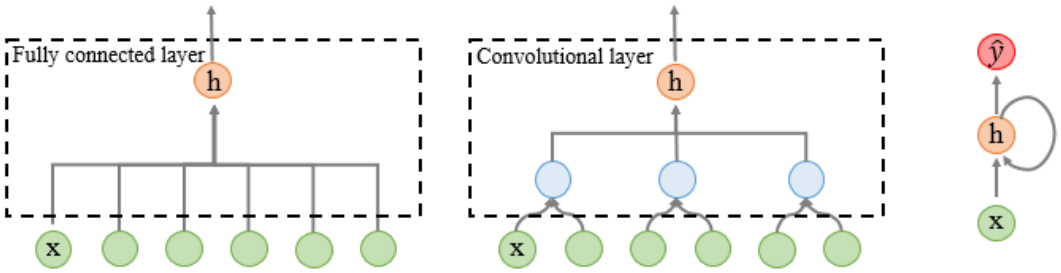
A DNN is a complex version of ANN containing multiple hidden layers between input and output layers [103]. Typical DNN is a feedforward network without lopping back [104]. Generally, DNN refers to fully connected networks (shown in Figure 2-5(a)), which means that each neuro in one layer receives information from all neuros from previous layer.

The motivations of utilizing DNN instead of simple ANN have been argued by Good Fellow et al.[105]: (1) DNN requires less neurons than simple ANN in representing complex tasks; (2) In practice, DNN generally presents higher prediction accuracy than ANN. However, implementing

DNN models should be done with careful attention to two common issues: overfitting and computing intensive.

(2) CNN

CNN is a special class of DNN, which adopts convolutional layers (shown in Figure 2-5(b)) to group input unites and apply the same function to gathered groups (i.e. parameter sharing). Compared with general DNN, CNN decreases the risk of overfitting by reducing the connectedness scale and structure complexity. Therefore, CNN could also be treated as a regularized version of typical DNN.



(a) (b) (c)

Figure 2-5: Schematic of (a) fully connected layer, and (b) convolutional layer (c) loop in RNN

CNN is well-known in the field of visual imagery analysis, such as image recognition [106], image classification[107], medical image analysis [108] and natural language processing [109]. To implement CNN into load prediction, Sadaei et al. [110] converted hourly load data, hourly temperature data and fuzzified version of load data into multi-channel images, and then fed it to a CNN model. The prediction performance of the developed CNN model was even better than Long Short-Term Memory (LSTM) models, a kind of RNN.

(3) RNN

The distinction between RNNs and other deep learning algorithms is that RNNs involve loops (shown as the cycle in Figure 2-5(c)) in their structure and makes it possible that information flow in any direction. These cycles introduce time delay in RNN and make RNN more suitable to exhibit temporal dynamic behavior. Therefore, the utilization of RNNs in energy prediction has attracted increasing research interests in recent years.

However, as the weight for the loop is the same for each time step, gradients in the traditional RNN tend to explode or vanish when the loop runs for many times. This problem is called long dependency. To solve this problem, one commonly utilized RNN model, called LSTM, could be applied to remember information for a long period.

2.1.2.9. Ensemble methods

An ensemble method combines the output of multiple learning algorithms in order to enhance the prediction performance of single data-driven models [111]. Commonly used ensemble methods could be classified into three categories: bagging, boosting and stacking models (also called parallel homogeneous, sequential homogeneous and heterogeneous ensemble methods [112]). Schematics of these three types of ensemble methods are shown in Figure 2-6.

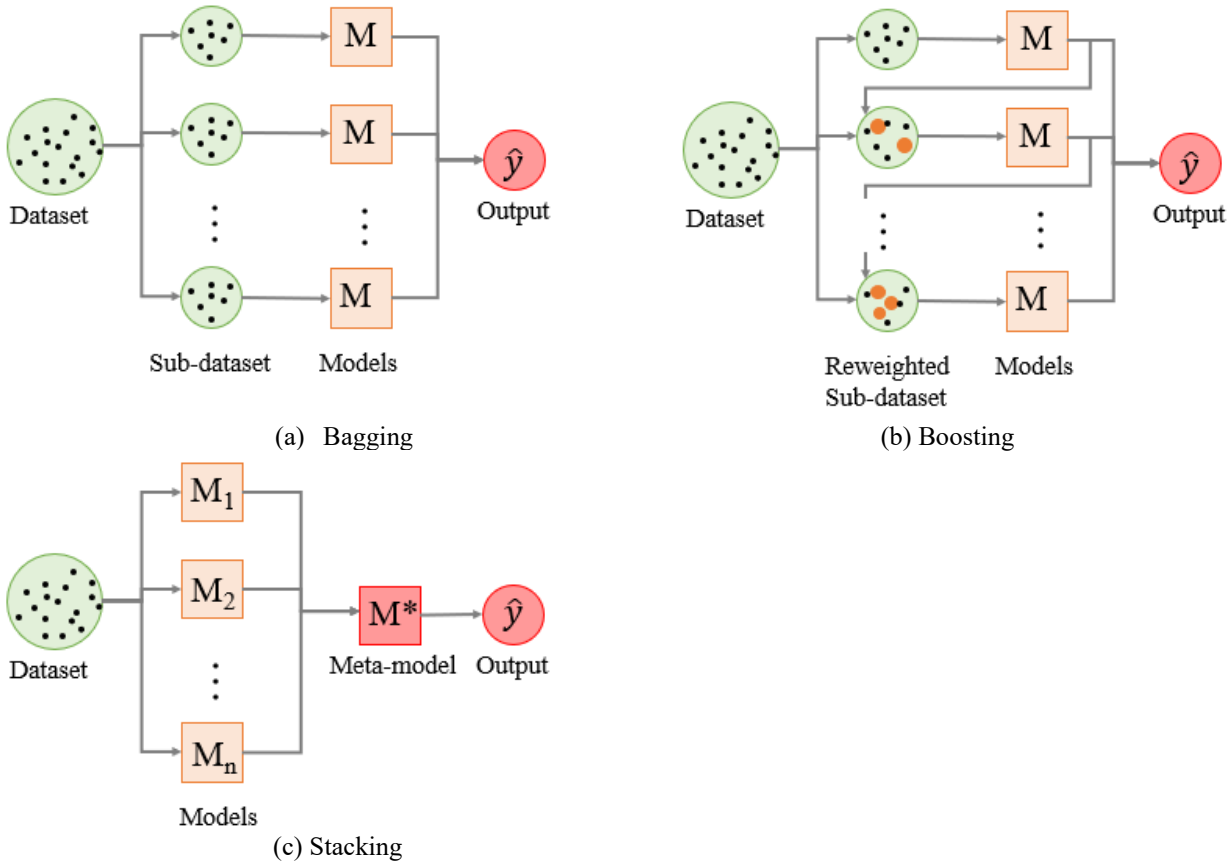


Figure 2-6: Schematic of different ensemble methods

(1) Bagging

Bagging, also called bootstrap aggregating, predicts the output by training the same baseline models parallel on different sub-datasets, which are sampled from original input datasets uniformly by replacement. This algorithm tends to decrease the variance when running the trained model on the validation set, due to the independence of each baseline model.

The most commonly utilized bagging method is random forests (RF), for which the baseline models are decision trees. Wang et al. [113] reported that RF is more accurate than RT and SVR in hourly electricity consumption prediction. Furthermore, Johannesen et al. [114] found that RF provides better 30 min-ahead electrical load prediction for urban area compared with kNN and linear regression. Wang et al. [79] proposed an ensemble bagging tree model to predict hourly educational building electricity demand. Their result shows that the proposed ensemble model is more accurate than RT. However, the larger training time of the bagging tree model than RT would be an issue. Besides, the required additional process for generating sub-dataset and the less interpretable than RT also limits the application of the proposed bagging method.

(2) Boosting

The difference between bagging and boosting is that boosting trains the baseline models incrementally, which means every successive model tries to fix the mistake made by previous

models. To achieve this goal, the basic solution is to increase the weight for misclassified data (i.e. orange points in Figure 2-6(b)). As a result of boosting, the training error would be decreased.

Robinson et al. [115] utilized a gradient boosting regression model to predict annual energy consumption for different types of commercial buildings located in different regions. Their results indicate that the gradient boosting regression model outperforms general linear models (e.g. linear regression and SVR) and even bagging models with limited number of features. Besides, Walter et al. [116] reported that the gradient boosting decision trees (GBDT) is flexible and accurate for very short term load forecasting for a factory.

Besides comparing the prediction accuracy between different models, interpretability, robustness and efficiency of different models should also be studied. Wang et al. [75] compared these four aspects of five models (i.e. extreme gradient boosting (XGB), GBDT, RF, ANN and SVM) based on a case study of 2-hour ahead heating load prediction for a residential quarter. They concluded that there is no best model when considering all performance. For instance, RF shows the highest accuracy, interpretability and robustness, while XGB presents better efficiency.

(3) Stacking

Unlike bagging and boosting, which utilize the same baseline models, stacking works on an arbitrary set of models. As shown in Figure 2-6(c), different models are trained on the available input dataset, and then a meta-model is trained based on the outputs of these models to make the final prediction.

Huang et al. [78] combined XGB, extreme learning machine (ELM), linear regression and SVR as an ensemble learning method, and then utilized it to do a 2-hour ahead heating load prediction for a ground source heat pump that supplies space heating for a residential area. Their result shows the proposed ensemble model is more accurate than XGB, ELM, linear regression and SVR. Fan et al. [62] developed an ensemble model integrated by eight learning algorithms to enhance the prediction accuracy for next day energy consumption and peak power demand.

2.1.3. OUTPUTS

2.1.3.1. Output type

As illustrated before, DDBMs could predict indoor air temperature [6], building energy consumption [3–5], occupants' thermal comfort [7], occupancy status/numbers [8–10], indoor air quality [11], lighting status [117],device operation status (normal/faulty) [15,118], or heating ventilation and air-conditioning (HVAC) system performance [14]. According to the type of output (discrete labels or continuous numbers), DDBMs could be classified as classification algorithms and regression algorithms: classification algorithms predict discrete class labels, while regression algorithms predict continuous quantity outputs.

2.1.3.2. Building type

When developing DDBMs, the type of buildings (i.e. residential or non-residential) should be distinguished, because the energy consumption habits and preferred indoor conditions would be different for different types of buildings. For instance, the load consumed by cooking could be a huge contribution for peak load in residential buildings, while official equipment would consume a considerable percentage of commercial building loads. Besides, the set-point temperature for air

conditioning is generally controllable for occupants in residential buildings, while it shows a large chance of being constant for nonresidential buildings.

Note that non-residential buildings further include commercial buildings, educational buildings, industrial buildings and hotels.

2.1.3.1. Scale

Scale can be classified into four classes: sub-building, building, district, region (city), country. Note that a sub-building refers to an individual room or component in a building.

DDBMs for larger scale (e.g. region and country) should not be considered as a simple aggregation of smaller scale (e.g. sub-building and building), since the effective and available features for different scales would vary [70]. For instance, socio-economic information tends to be collectable and useful for large scale energy prediction, while its effect declines in predicting building/sub-building level energy consumption. Besides, the application of DDBMs in reality varies for different scales. For instance, the energy prediction model developed for sub-building and building scale could be utilized for demand response control, while large scale energy prediction model is applicable in energy distribution.

2.1.3.2. Temporal granularity

Two types of temporal granularity need to be determined: horizon and resolution. Horizon means the length of time-ahead prediction, while resolution means the duration of a time step. When horizon is longer than resolution, the developed model makes an n-step ahead prediction, where n equals to horizon divided by resolution. For instance, if the prediction horizon is 1 hour and the resolution of data points is 15 min, then the model does a 4-step ahead prediction. One thing to note is that under the same resolution, longer horizon prediction takes more risk for higher error. For instance, Ding et al. [119] utilized historical data and meteorological parameters to predict one hour ahead and one day ahead cooling load with 30 min intervals. Their results indicate that a shorter horizon (i.e. 1 hour) prediction presents higher accuracy than a one day ahead prediction.

Time horizon of electrical load prediction models is usually classified as four categories: very short-term, short-term, medium-term and long-term [70], [120]. However, the cut-off horizon for these categories varies among different references. Generally, when the time horizon is less than one month, it belongs to short term prediction or even very short-term prediction. Very short term and short-term predictions help users to implement proper control strategy, while medium- and long-term prediction could be beneficial for utilities to upgrade their equipment and for governments to formulate building standards.

2.1.4. PERFORMANCE CRITERIA

For two-class classification problems, accuracy (Equation 2-10), recall (Equation 2-11), and specificity (Equation 2-12), precision ((Equation 2-13) are commonly used criteria for evaluating predictive accuracy.

The meanings of TP, TN, FP and FN in these equations are shown in Table 2-2.

Table 2-2: Confusion matrix

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

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

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \quad (2-10)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2-11)$$

$$\text{Specificity} = \frac{\text{TN}}{\text{FP} + \text{TN}} \quad (2-12)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2-13)$$

The importance of using these accuracy measures is: 1) Accuracy reflects the overall predictive accuracy of the data-driven model. It shows the closeness of the predicted values to the measured data; 2) Recall indicates the true positive rate, which is the probability of predicted as positive when the actual class label is positive. For instance, for lighting status prediction, recall evaluates the percentage of correct prediction when the actual lighting status is ON. Thus, a worse recall indicates that the lighting status is falsely predicted as OFF when it should be turned ON; 3) 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 the lighting when the actual lighting status is OFF. Thus, if specificity is too low, the predictor would wrongly predict to turn ON the lighting when it is not needed; and 4) Precision calculates the correct predictive proportion among predicted positives. A high precision enables the predictor to be very sure of its positive prediction.

On the other hand, for regression problems, common predictive performance criteria include [86,121–124]:

Mean Absolute Error (MAE) = $\frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$ (2-14)

Mean Absolute Percentage Error (MAPE) (%) = $\frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| * 100$ (2-15)

Mean Bias Error (MBE) = $\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)}{n}$ (2-16)

Normalized MBE (NMBE) (%) = $\frac{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)}{n}}{\bar{y}} * 100$ (2-17)

Mean Squared Error (MSE) = $\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$ (2-18)

Root Mean Square Error (RMSE) = $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$ (2-19)

$$\text{Coefficient of Variation of the Root Mean Square Error (CV(RMSE))} = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}}{\overline{y}} \quad (2\text{-}20)$$

$$\text{R Square } (R^2) = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad (2\text{-}21)$$

where $\overline{y}$ is the average value of measured outputs.

These criteria could also be utilized to evaluate the prediction accuracy during model training. Through comparing the prediction accuracy between model training and model validation, overfitting or underfitting could be detected [125]. For instance, if training accuracy is much higher than validation, it might indicate overfitting which means the trained data-driven model fits too closely to the training set with covering the noise and outlier. Besides, if both training and validation accuracy are not acceptable, underfitting occurs to show that the developed data-driven model cannot capture the structure of the studied problem. Both overfitting and underfitting undermine the developed models' generalization, which refers to the ability to predict unseen data.

Here, a short description is given in the following to help the criteria selection.

MAE is the mean value of the sum of absolute errors, while MBE is the average prediction error which could be understood as how far the average predicted values is above or below the average of measured output value. Both MAE and MBE have units that should be taken into consideration when utilizing them to compare the results of different works. Note that the under-predicted outputs would reduce the value of MBE, which means cancellation errors. Therefore, if choosing MBE, other criteria without cancellation errors should be considered.

MAPE is a commonly utilized measure of prediction accuracy because it calculates the mean relative prediction error without units. However, it cannot be utilized when there are zero values in the measured output. By contrast, zero values would not be a big concern when utilizing NMBE, which also shows the advantages of having no units. However, NMBE is limited by cancellation errors.

MSE has the ability to evaluate both variance and bias of the predicted value to the measured output. Note that the unit of MSE would be square of the unit of predicted outputs. To have the same unit as the predicted outputs, RMSE could be utilized. In terms of principle, CV(RMSE) is calculated by dividing RMSE by the mean value of measured outputs; therefore, it evaluates how much the predicted error varies with respect to the mean target value. It is not limited by cancellation errors. Furthermore, NMBE and CV(RMSE) have been recommended as evaluation criteria for building energy prediction models by several standards, such as ASHRAE [86], FEMP [126] and IPMVP [127].

R^2 indicates the goodness of fit. The bigger the value of R^2, the closer its predicted value will be to its target value.

2.1.5. SUMMARY

A simplified summary of existing studies on DDBMs is presented in Table 2-3. Detailed review for DDBMs and their accuracy measures could be found in our paper [16].

From Table 2-3, occupancy related data, such as the number of occupants, motion status, types of occupant activities, etc., has been widely utilized as inputs or outputs for DDBMs. This is because occupancy related information could be beneficial for decreasing the gap between predicted energy consumption and measured values [128,129], improving the IAQ predictive performance [11], and ensuring thermal comfort and energy saving when integrated into HVAC control systems [8]. To collect the occupancy related data for data-driven model development, survey [61,130] or monitoring devices (e.g. PIR sensors [19,131], camera [8,132], WIFI [131], and Bluetooth, etc.) could be utilized. However, utilizing the collected or predicted occupancy-related data might suffer privacy issues, as previous studies mention it might infer occupants' individual location and behavior [17–19]. Thus, occupants should have the option to deny disclosure of these information.

Besides, the available features may differ among users/buildings, due to the difference in installed data collection devices or users' opinions for information sharing. For instance, newly constructed smart buildings are more likely to be equipped with additional data collection devices than old ones. In addition, occupancy-related data may be available for some, while denied to use by other users due to privacy concerns. On the other hand, data volume may differ among different conditions. 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. Due to the variation of data volume in different users/conditions, existing DDBMs would present distinct accuracy among these users/conditions. The predictive performance variation among different situations may leads to fairness problems, such as using non-permitted features or providing better predictive performance for a certain groups/conditions than others.

However, most existing DDBMs were validated by accuracy measures. These measures just reflect the overall predictive performance, while the predictive performance under different conditions/periods could not be evaluated. In other words, the predictive fairness could not be evaluated. For instance, if a heating demand predictor is developed for the HVAC system of a house with an occupant who is working during daytime, the collected data may show that the heating demand is generally decreased during daytime. Then, the data-driven predictor trained based on the collected data would predict that the heating demand would decrease during the daytime and it would automatically recommend/control the HVAC system to set a lower set-point temperature. The overall accuracy measures would show that the predictor works perfect because its predictive values are close enough to the measured data. However, the predictor could still show a poor predictive result when the occupant actually requires a warmer indoor environment during daytime, because these phenomena are rare in the collected data. Inspired by this problem, evaluating predictive fairness to have uniform performance in different situations should be considered.

Table 2-3: Summary of existing studies focusing on developing data-driven building models

Reference	Classification/ Regression	Model(s)	Input features	Output	Validation criteria	Type of building
[59]	Regression	LSTM, ANN, linear regression, ARIMA	**Occupant count**, time index, historical data	Plug load	RMSE, CV(RMSE)	Office
[60]	Regression	Recurrent neural network (RNN)	**Occupant activity indicator**, meteorological information	HVAC power demand	RMSE, MAPE, MASE, R^2, the maximum error	Educational
[133]	Regression	MLR, random forest regression	**Occupancy count and behavior**, meteorological information, building and device characteristic data	Indoor PM2.5 concentrations	R^2, RMSE, index of agreement (IOA)	Residential
[10]	Classification	SVM, classification and regression trees (CART), inhomogeneous hidden Markov (IHMM), GcForest	Indoor environmental information	**Occupancy count**	Estimation accuracy (EA), MAE, detection accuracy (DA)	Office
[134]	Classification	CART, random forest (RF), gradient boosting machines (GBM), linear discriminant Analysis (LDA)	Indoor environmental information	**Occupancy status**	Accuracy	Office

2.2. FAIRNESS-AWARE DATA-DRIVEN MODELS

Above-mentioned problems in existing DDBMs, such as privacy issues caused by using sensitive information, and variation performance among different conditions, could be solved by fairness-aware data-driven procedures. In fact, fairness-aware machine learning has drawn increasing attention in recent years and has been applied to non-discriminatory hiring [29–31], risk assessment for sentencing guidance [32,33], income prediction [33], loan allocation [34,35] and graph embedding [36]. However, it is not yet used to investigate the fairness problems among data-driven models in the building domain, due to the gap between disciplines. Therefore, in this section, the commonly used fairness definitions will firstly be summarized in Section 2.2.1. The benefits of considering fairness in data-driven based building models will be explained in Section 2.2.2, while methods for improving fairness among the predictive results will be presented in Section 2.2.3. Finally, Section 2.2.4 summarizes findings from the review for fairness-aware data-driven models.

2.2.1. FAIRNESS CONCEPTS

Fairness-aware machine learning models are usually designed to tackle different fairness problems. In general, these studies could be categorized into three categories based on the type of fairness they achieved [135]:

Type I. The predictive result of the developed model or the decision made based on the output of the model is independent of some protected attributes (also called sensitive attributes). Note that protected attributes are defined as attributes whose values are not willing to be disclosed [136]. The commonly used protected attributes include gender, race, and sexual orientation, etc. For building models, occupancy-related data could be selected as protected attributes. This type of fairness could be evaluated by two forms: (1) The protected attribute is not utilized as one of the inputs. For instance, for building models, the occupancy-related data is excluded from the inputs by occupants due to the concern of privacy; (2) The predictive outcomes/values are similar among instances, which come from different groups defined by the protected attribute(s) but have the same value(s) for the unprotected attribute(s). The potential requirement for fairness in terms of this type is not found temporarily in building domain. However, an example could be given for non-discriminatory hiring: the possibility of getting a position should be equal for a white people or a people from a colored race (see Equation 2-22), if their other performances/inputs (such as work performances) are the same.

$$\Pr\left(\widehat{\mathrm{Y}} = \mathrm{Employed} \,\middle|\, \mathrm{S} = \mathrm{White}\right) = \Pr\left(\widehat{\mathrm{Y}} = \mathrm{Employed} \,\middle|\, \mathrm{S} = \mathrm{Colored}\right) \qquad (2\text{-}22)$$

where $\widehat{\mathrm{Y}}$ is the predicted result, S is the protected attribute.

Type II. Some given measures of predictive performance, which could reflect the difference between predicted values and measured values (such as accuracy measures), are similar across groups defined by the protected attributes. For instance, when predicting the lighting status (ON/OFF) with considering motion status (ON/OFF) as the protected attribute, the true positive

rate when motion status is ON is similar with the corresponding rate when the motion status is OFF (see Equation 2-23). Then, the developed lighting status predictive model is fair in terms of this type. Note that this definition of fairness is more applicable when the predictive performance is the main concern, and it is not expected to be affected by the protected attributes.

$$\Pr(\hat{Y} = \mathrm{ON} | Y = \mathrm{ON}, S = \mathrm{ON}) \approx \Pr(\hat{Y} = \mathrm{ON} \mid Y = \mathrm{ON}, S = \mathrm{OFF}) \quad (2\text{-}23)$$

where Y is the actual output.

Type III. The predictive outcomes should be independent of protected attributes conditional on the predicted probability score. This type of fairness is usually considered in classification problems. For instance, when developing models for non-discriminatory hiring, some kinds of predictors (such as logistic regression) could return a predicted probability score (denoted by Sc) for each instance, then, if two persons who come from different races get the same probability score, they should have the same opportunity to be employed (see Equation 2-24).

$$\Pr(\hat{Y} = \mathrm{Employed} \mid Sc = sc, S = \mathrm{White}) = \Pr(\hat{Y} = \mathrm{Employed} \mid Sc = sc, S = \mathrm{Colored}) \quad \forall sc \in Sc \quad (2\text{-}24)$$

Note that previous studies mention that fairness could not be achieved for all types at the same time in most cases [137], researches are recommended to select proper fairness measures based on their research objective.

2.2.2. BENEFITS OF CONSIDERING FAIRNESS IN DDBMS

Above listed examples for *Type I* and *Type II* indicate the aspiration of considering fairness in the building engineering domain. To be more specific, fairness-aware DDBMs are worth to be investigated due to the following benefits:

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

2) Ensuring uniform predictive performance among different conditions defined by the protected attributes. A related example has been given above: predicting lighting status with considering motion status as the protected attribute. Achieving fairness for this problem in terms of having similar predictive accuracy when the motion status is ON or OFF could ensure the predictor to work stable at any time.

3) Preserving fairness for different users. For instance, users of different buildings usually show different habits and different opinion for submitting their sensory data to develop data-driven models. Thus, the predictive performance could vary for different users because the available dataset for model training varies. Considering fairness among these developed models could make sure their predictive performance is similar, and thus, all users could obtain similar service provided by these predictors.

2.2.3. FAIRNESS IMPROVEMENT METHODS

To improve fairness, commonly used methods can be classified in to three categories: pre-processing, in-processing and post-processing. Detailed review for each kind of methods is presented in the following subsections.

2.2.3.1. Pre-processing methods

Pre-processing is a procedure to preprocess the training data with the goal of removing discrimination among the data before model training. It is more applicable and easier to implement when more than one type of data-driven model is developed. Further, training data collected or processed by the fairness-aware procedure has been reported as the most important place to improve fairness in the industrial products development [138].

The easiest method to achieve fairness under *Type I* is to suppress the protected attributes from the input features [139]. However, this method could not ensure the decline of discrimination among the dataset and predictive results, and it might destroy the predictive accuracy when the protected attributes and the outputs are highly correlated. Besides, Feldman et al. [140] proposed a repair approach to change unprotected attributes in order to mask bias while preserve relevant information in the data. Note that in their study, fairness is evaluated by the predictability of the protected attributes from the unprotected attributes.

To achieve similar predictive performance among different conditions, processing a balanced training dataset among these conditions could be a good option. To obtain a balanced training dataset, oversampling for minority conditions without sufficient data and/or undersampling for majority conditions could be considered.

Pre-processing methods that process the original training dataset to a balanced one have been widely used in fault detection and diagnosis (FDD). For instance, Fan et al. [22] applied an oversampling technique, called SMOTE, to oversample faulty samples before training the support vector machine (SVM) model for chiller fault diagnosis. The re-balanced training dataset improved diagnostic accuracy. However, if the oversampling size is larger than 100% of original data, the large volume of synthetic samples would increase the classification uncertainty. Yan et al. [141] increased data in faulty conditions among the training dataset by inserting confidently predicted samples by data-driven classifiers. It results in over 80% diagnostic accuracy for air handling units (AHUs) in summer and 89% in winter. Besides, generative adversarial network (GAN) has been used to enrich the training dataset by generating samples for conditions that initially do not have adequate respective data. For instance, Yan et al. [142] applied the GAN to generate faulty training samples for fault detection and diagnosis (FDD) of 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), SVM, multi-layer perceptron (MLP), k-nearest-neighbor (KNN) and decision tree (DT)) from nearly 50% to almost 100%. Yan et al. [24] have also applied the GAN to re-balance the training dataset for automatic FDD for chillers. Li et al. [1] proposed a modified GAN to improve the diagnostic accuracy for building HVAC systems by taking advantage of the re-balanced labeled and unlabeled data. However, the above-mentioned data generation techniques, such as GAN, are hard to update in order to capture changes in the training dataset [143].

Even if the above pre-processing methods have been applied to solve problems caused by imbalanced dataset, they have not been used to achieve *Type II* fairness of DDBMs, which requires similar predictive performance between conditions defined by the protected attributes. This is because that above pre-processing methods only balance the data among conditions defined by the output labels, instead of situations defined by the protected attribute.

To achieve *Type II* fairness, three data pre-processing approaches (i.e., massaging, reweighing and sampling) proposed by Kaviran and Calders [139] could be utilized to omit the bias among the training dataset, and then, ensure fairness in a two-class classification problem with respect to one binary protected attribute. When multi-variate non-binary protected variables are defined, an optimized data transformation procedure proposed by Calmon et al. [33] could be utilized to improve the fairness with a small cost of classification accuracy. Although these pre-processing methods have been applied to improve *Type II* fairness, they have never been applied to process data for DDBMs.

2.2.3.2. In-processing methods

In-processing means adding fairness-related constraints or penalty to the optimization objective of the model during model training [37]. It could achieve specific fairness measures chosen by the programmer while preserving high accuracy. However, due to the type of fairness measure could vary among different predictive tasks, the code may need to be modified accordingly. It would result in increasing difficulty when applying a developed method to a new task.

There are mainly three types of in-processing methods: fairness constraints, prejudice remover regularizer and adversarial debiasing. All of them could be applied to regression or classification problems. Among these methods, the fairness constraints method adds fairness constraints to the loss function of training process; the prejudice remover regularizer method applies a fairness regularizer to the loss function; while the adversarial debiasing method develops a predictor and an adversary at the same time to weak the power of predicting the protected attribute from the predictive outputs. A summary of existing studies focusing on in-processing methods is given in

Table *2-4*.

Table 2-4: Summary of studies on in-processing methods

Reference	Classification/ Regression	Type of in-processing methods	Model(s)	Type of fairness	Field of application
[144]	Classification	Fairness constraints	Logistic regression, SVM	*Type III*	Income prediction, loan allocation
[145]	Regression	Fairness constraints	Linear regression	*Type II*	Risk equalization for health plan payment
[146]	Regression	Fairness constraints	Gaussian process, support vector regression, neural network regression, regression tree	*Type III*	Violent recidivism
[147]	Regression	Prejudice remover regularizer	Linear regression, logistic regression	*Type II*	Risk assessment for sentencing guidance, income prediction, loan allocation, violent recidivism, grade prediction
[148]	Classification	Prejudice remover regularizer	Logistic regression	*Type I*	Income prediction
[149]	Classification	Prejudice remover regularizer	Deep learning	*Type I, Type II*	Income prediction, violent recidivism, wine quality prediction
[150]	Classification	Adversarial debiasing	Neural network	*Type I, Type II*	Violent recidivism
[151]	Classification, regression	Adversarial debiasing	Neural network	*Type I, Type II, Type III*	Word embedding, income prediction

Similar as in-processing fairness improvement methods, cost-sensitive learning assigns different misclassification costs to data from different classes, and thus, forces the classifier concentrate on minority classes or classes that desire higher predictive accuracy. For instance, cost-sensitive learning algorithms are widely used in building engineering to improve classification accuracy of minority classes. For instance, Li et al. [152] integrated cost-sensitive analysis into classification models, such as RF, SVM, and DT, to improve the identification performance of personalized occupancy behavior. Jufri et al. [153] improved the grid damage forecasting accuracy by applying a cost-sensitive learning algorithm on the regression model. Tang et al. [25] proposed a cost-sensitive extremely randomized trees algorithm for wind turbine generator fault detection. The proposed method outperforms the traditional randomized trees in terms of average missing detection rate. Furthermore, AdaBoost models are commonly used cost-sensitive models in FDD[26,27] and energy prediction [154]. However, these cost-sensitive learning algorithms have never been utilized to solve fairness problems in the building engineering domain.

2.2.3.3. Post-processing methods

Post-processing is a method that modifies the prediction results of a classifier to achieve fairness. It is generally applied to two-class classification algorithms, which return a score to indicate the possibility of having a positive or negative class label for each individual. A threshold

would be set, so that when the score is higher than this threshold, its class label would be positive, otherwise, it would be predicted as negative. The post-processing method would adjust the threshold, in order to have similar predictive results for different protected groups [155]. This method could be extended to any classifiers and it does not require the modification of classifiers. However, it lacks the flexibility of achieving accuracy and fairness trade-off [38].

2.2.4. SUMMARY

In Section 2.2, three types of fairness were introduced. Existing studies verified that achieving multi-kinds of fairness at the same time could be a challenging and almost impossible work, and thus, researchers are suggested to select a proper type of fairness based on their study objective.

Examples of considering fairness in DDBMs were given when introducing fairness concepts. Inspired by them, potential benefits of fairness-aware DDBMs were summarized: 1) Enabling authority management and privacy protection; 2) Ensuring uniform predictive performance among different conditions defined by the protected attributes; 3) Preserving fairness for different users. The study scope of this research is to achieve *Type II* fairness that ensures uniform predictive performance among conditions defined by the protected attribute.

Finally, fairness improvement methods were reviewed in terms of pre-processing methods, in-processing methods, and post-processing methods. Two limitations have been found: 1) Existing fairness improvement methods have never been applied to DDBMs; 2) Existing pre-processing methods and in-processing methods in the building engineering domain show similar fundamental as these fairness improvement methods, however, they are aimed at balancing predictive performance among conditions defined by output labels, instead of conditions defined by protected attributes. In other words, existing methodologies in the building engineering domain could solve problems caused by imbalanced data, however, they have not been applied to solve fairness problems.

Therefore, to meet above-mentioned limitations, this study will propose pre-processing methods to solve fairness problems in classification DDBMs, while apply in-processing methods to achieve *Type II* fairness for regression problems in the building engineering domain.

CHAPTER THREE: METHODOLOGY

This chapter explains methodologies proposed and investigated in each task: For task A, four pre-processing methods are proposed to make the training dataset for classification problems balance among conditions defined by the protected attribute. They are aimed at removing discriminations from the training dataset, and thus, letting the predictive performance similar among different conditions. For task B, four in-processing methods are designed to set fairness-related penalties or constraints in the loss function when training the regression models. They are aimed at achieving the trade-off between accuracy and fairness through commanding the loss function during model training directly. For task C, the fairness improvement methods proposed in task A and task C would be integrated into a MPC.

Detailed explanation for methodologies used in each task are presented as below:

3.1. TASK A: PRE-PROCESSING METHODS FOR CLASSIFICATION PROBLEMS

The proposed sequentially balanced sampling (SBS), sequential sampling (SS), reversed preferential sampling (RPS), and sequential preferential sampling (SPS) are present in this section. Besides, existing pre-processing methods, namely random sampling (RS) and preferential sampling (PS), are also introduced and used to compare with the proposed methods. These preprocessing methods are aimed at processing the original candidate training dataset ($X_{candidate}$) to generate a designed training dataset ($X_{designed}$) that distributes evenly among conditions defined by the protected attribute and output labels.

For two-class classification problems with a binary protected attribute, there would be four conditions, i.e., *PP, PN, NP,* and *NN*, as listed in Table 3-1. For example, in lighting status (ON/OFF) prediction considering motion status (ON/OFF) as the protected attribute, *PP* means the condition that lighting status is ON and motion status is ON; *PN* refers to the period during which lighting status is OFF and motion status is ON; *NP* is the situation that lighting status is ON and motion status is OFF; *NN* means that lighting status and motion status are OFF. In addition, the number of samples in these conditions are represented by $|PP|$, $|PN|$, $|NP|$, and $|NN|$, respectively. To eliminate bias among these conditions, the expected number of data points for each condition in $X_{designed}$ is calculated using Equation 3-1.

Table 3-1: *PP, PN, NP* and *NN* defined by the protected attribute and output labels

		Y	
		Positive	Negative
S	Positive	PP	PN
	Negative	NP	NN

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

$$|PP|_{design} = |PN|_{design} = |NP|_{design} = |NN|_{design} = 0.25 * \left| X_{designed} \right| \quad (3\text{-}1)$$

where $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, and $|NN|_{design}$ is the expected number of data points in *PP, PN, NP,* and *NN* of $X_{designed}$, respectively; $\left| X_{designed} \right|$ is the size of $X_{designed}$.

For multi-class classification problems with multi-class protected attributes, such as a *i*-class prediction problem with a *j*-class protected attribute, there would be *i*j* conditions (see Table 3-2) and the expected number of data points in each condition is $\frac{1}{i*j}|X_{designed}|$.

Table 3-2: Conditions defined by a i-class prediction problem with a j-class protected attribute

		Y			
		Y_1	Y_2	...	Y_i
S	S_1	S_1Y_1	S_1Y_2	...	S_1Y_i
	S_2	S_2Y_1	S_2Y_2	...	S_2Y_i
	...	...		...	...
	S_j	S_jY_1	S_jY_2	...	S_jY_i

Note S_jY_i is the condition in which data's protected attribute is S_j and output label is Y_i.

In the following subsections, the detailed procedure of these techniques is mainly explained for two-class classification problems with a binary protected attribute, while additional explanation for multi-class classification problems with multi-class protected attributes is provided when it is necessary. In general, the procedure could be simply modified for multi-class classification problems with multi-class protected attributes through considering more conditions in each step.

3.1.1. SEQUENTIALLY BALANCED SAMPLING (SBS)

Sequentially balanced sampling (SBS) is aimed at getting a balanced training dataset at the first time of processing $X_{candidate}$ with preserving the latest information from $X_{candidate}$. Its procedure is as follows:

Step 1. Partitioning samples in $X_{candidate}$ into *PP, PN, NP,* and *NN*.

Step 2. List the data in each condition in descending order of collection time.

Step 3. Sample the most recently collected $0.25*|X_{designed}|$ data points from each condition. If the number of points in one condition is less than $0.25*|X_{designed}|$, it would duplicate the most recently collected data until the data in that condition reaches $0.25*|X_{designed}|$.

These steps are summarized in Figure 3-1. Detailed coding algorithm for SBS is shown in Algorithm 3-1.

In addition, for a *i*-class prediction problem with a *j*-class protected attribute, Step 1 would divide samples into *i*j* conditions, while Step 3 would sample $\frac{1}{i*j}\left|X_{designed}\right|$ recently collected data for each condition.

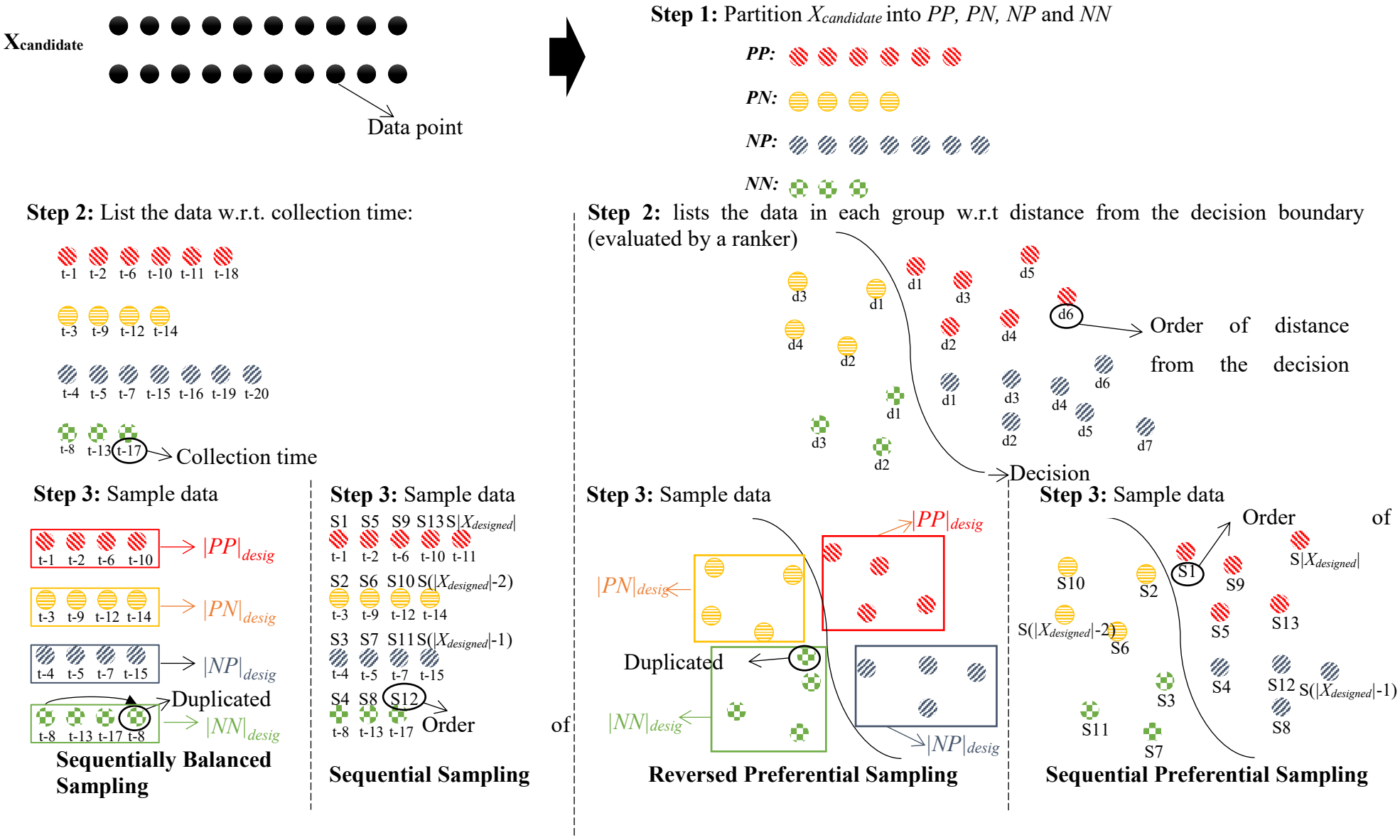


Figure 3-1: Procedure of SBS, SS, RPS, and SPS in a two-class classification problem with a binary protected attribute

Algorithm 3-1: Sequentially Balanced 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. Order training points in x descending w.r.t. collection time
3. Calculate $|PP|_{design}$, $|PN|_{design}$, $|NP|_{design}$, $|NN|_{design}$
4. [*i_total, i_PP, i_PN, i_NP,i_NN*] = 0
5. **for** *x* **in** [*PP, PN, NP, NN*]:
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. **else**:
13. Slice the first $|x|_{design}$ number of points in the group with max($|PP|$, $|PN|$, $|NP|$, $|NN|$) to $X_{designed}$<*D, S, Y*>
14. **return** Designed training set $X_{designed}$<*D, S, Y*>

Note: In this algorithm, <*D, S, Y*> means the elements of one training point, where *D* is the unprotected features.

3.1.2. SEQUENTIAL SAMPLING (SS)

Sequential sampling (SS) also captures the latest information from $X_{candidate}$, but it will make $X_{designed}$ balance as the time of updating $X_{candidate}$ increases and the number of data points in each condition of $X_{candidate}$ are more than the expected number.

As shown in Figure 3-1, its first two steps are the same as SBS, the difference is in Step 3: SS samples a data point from each condition to $X_{designed}$ each time by ascending order, until reaching $|X_{designed}|$. To be more specific, it samples the first point from *PP, PN, NP,* and *NN* in turns at the first time. Then, it gets the second data point and next round the third point. After each round, the sampling will move to the next data. If all data points in one condition are sampled but this condition in $X_{designed}$ still does not get 0.25*$|X_{designed}|$ points, SS will continue sample data from other conditions until data in $X_{designed}$ reaching $|X_{designed}|$. In other words, when newly observed data comes to update $X_{candidate}$, it will catch this data to $X_{designed}$, and then, delete one old data from the majority condition. Its coding algorithm is shown in Algorithm 3-2.

Algorithm 3-2: Sequential sampling

Algorithm 3-2: 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*>

1. Partition *PP, PN, NP, NN* from $X_{candidate}$<*D, S, Y*>
2. Order training points in *x* descending w.r.t. collection time
3. [*i_total, i_PP, i_PN, i_NP,i_NN*] = 0
4. **for** n **in** range (0, $|X_{designed}$<*D, S, Y*>|):
5. **for** *x* **in** [*PP, PN, NP, NN*]:
6. **if** *i_x* >= |*x*| or *i_total* >= $|X_{designed}$<*D, S, Y*>|:
7. **continue**
8. Slice the *i_x* th data in *x* to $X_{designed}$<*D, S, Y*>
9. *i_x* = *i_x*+1
10. *i_total* = *i_total*+1
11. **if** *i_total* >= $|X_{designed}$<*D, S, Y*>|:
12. **break**
13. **return** Designed training set $X_{designed}$<*D, S, Y*>

3.1.3. REVERSED PREFERENTIAL SAMPLING (RPS)

Reversed preferential sampling (RPS) was suggested following the hypothesis that duplicating data close to the decision boundary (a hypersurface separating the dataset into two classes) in minority conditions will help distinguish the decision boundary, while removing data furthest from the decision boundary in majority conditions could avoid making large changes to the decision boundary. Its procedure is explained as below.

Step 1. The same as SBS.

Step 2. List the data in each condition in ascending order of distance from the decision boundary. For conditions with *Y*=Positive, data's distance from the decision boundary is represented by its possibility of classifying as positive (*p_positive*), while evaluated by the probability of negative (*p_negative*) when *Y*=Negative. In this study, *p_positive* and *p_negative* are calculated by a ranker (coding algorithm shown in Algorithm 3-3), in which a probabilistic classifier, such as logistic regression (LR) or Naïve Bayes (NB), is trained using $X_{candidate}$. For multi-class classification problems, data is listed by the ascending order of correct prediction probability. For instance, in condition S_jY_i, data is listed by the ascending order of the possibility of predicting a data as class Y_i.

Algorithm 3-3: Ranker

Algorithm 3-3: 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 | *X*=*x*)
4. Calculate *p_negative* := $Pr(\hat{Y}$ = Negative | *X*=*x*)
5. **return** <*p_positive, p_negative*> for each training point

Step 3. If the number of actual samples in one condition is above 0.25*|$X_{designed}$|, slice the first 0.25*|$X_{designed}$| training points to $X_{designed}$. Otherwise, duplicate points in that condition by ascending order until reaching 0.25*|$X_{designed}$| and sample them to $X_{designed}$.

The entire procedure of RPS is coded as Algorithm 3-4.

Algorithm 3-4: Reversed preferential sampling

Algorithm 3-4: 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. Order training points in *PP* and *NP* ascending w.r.t. *p_positive*
5. Order training points in *PN* and *NN* ascending w.r.t. *p_negative*
6. **for** *x* **in** [*PP, PN, NP, NN*]:
7. **if** |*x*| >= $|x|_{design}$:
8. Slice the first $|x|_{design}$ number of points in *x* to $X_{designed}$<*D, S, Y*>
9. **else if** |*x*| != 0**:**
10. Slice |*x*| number of points in *x* to $X_{designed}$<*D, S, Y*>
11. Duplicate the first ($|x|_{design}$ - |*x*|) number of points from *x*
12. Transfer the duplicated points to $X_{designed}$<*D, S, Y*>
13. **else**:
14. Slice the first $|x|_{design}$ number of points in the group with max(|*PP*|, |*PN*|, |*NP*|, |*NN*|) to $X_{designed}$<*D, S, Y*>
15. **return** Designed training set $X_{designed}$<*D, S, Y*>

3.1.4. SEQUENTIAL PREFERENTIAL SAMPLING (SPS)

Sequential preferential sampling (SPS, coded as Algorithm 3-5) gradually gets a balanced training dataset, while maintaining the most representative data for distinguishing the decision boundary. Therefore, its difference from the RPS is in Step 3: SPS iteratively samples a training point each time from the four conditions in $X_{candidate}$ to $X_{designed}$, in ascending order of distance from the decision boundary. Like SS, it will get a balanced training dataset as the times of updating $X_{candidate}$ increases.

Algorithm 3-5: Sequential preferential sampling

Algorithm 3-5: 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. Learn a ranker by using X_candidate<D, S, Y>
3. Order training points in PP and NP ascending w.r.t. p_positive
4. Order training points in PN and NN ascending w.r.t. p_negative
5. [i_total, i_PP, i_PN, i_NP,i_NN] = 0
6. for n in range (0, |X_designed<D, S, Y>|):
7.         for x in [PP, PN, NP, NN]:
8.                 if i_x >= |x| or i_total >= |X_designed<D, S, Y>|:
9.                         continue
10.                Slice the i_x th number of points in x to X_designed<D, S, Y>
11.                i_x = i_x+1
12.                i_total = i_total+1
13.                if i_total >= |X_designed<D, S, Y>|:
14.                        break
15. return Designed training set X_designed<D, S, Y>
```

3.1.5. RANDOM SAMPLING (RS)

Random sampling (RS, see

Algorithm *3-6*) randomly samples the expected number of data points from each condition in $X_{candidate}$ to $X_{designed}$. For instance, in two-class classification problems with a binary protected attribute, when the actual points in one condition are more than $0.25*|X_{designed}|$, randomly select $0.25*|X_{designed}|$ training points from that condition to $X_{designed}$. Otherwise, randomly duplicate points in that condition until $0.25*|X_{designed}|$ is reached, and then sample these data to $X_{designed}$.

Algorithm 3-6: Random sampling

Algorithm 3-6: Random 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>
```

3.1.6. PREFERENTIAL SAMPLING (PS)

Preferential sampling (PS) and RPS differ as preferential sampling duplicates or removes data close to the decision boundary for each group. Preferential sampling is proposed because data points close to the decision boundary are more likely to be discriminated or favored [139].

The Step 1 and Step 2 of PS are the same as RPS, while the difference is in Step 3: 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 3-7: Preferential sampling

Algorithm 3-7: 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 $X_{designed}<D, S, Y>$

3.2. TASK B: IN-PROCESSING METHODS FOR REGRESSION PROBLEMS

3.2.1. IN-PROCESSING 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* include mean square error (MSE, see Equation 3-2) and mean absolute error (MAE, see Equation 3-3). 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 converge 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 more robust to outliers than MSE.

$$\text{MSE} = \frac{1}{n}\sum_{i=1:n}(y_i - \hat{y}_i)^2 \tag{3-2}$$

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

$$\text{MAE} = \frac{1}{n}\sum_{i=1:n}|y_i - \hat{y}_i| \tag{3-3}$$

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 clear explanation, the original loss function (such as MSE or MAE) without considering fairness is denoted by *Loss_ori*, the protected attribute is assumed as a binary attribute. But these methods could be extended to problems with multi-class protected attribute through adding pairwise constraints/penalties.

3.2.1.1. Mean residual difference penalized regression (MRDP)

The loss function of this method, as shown in Equation 3-4, is consist of *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 the 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.

$$min\, Loss\_ori + \lambda[\frac{1}{s0}\sum_{h=1:s0,S=negative}(y_h - \hat{y}_h) - \frac{1}{s1}\sum_{k=1:s1,S=positive}(y_k - \hat{y}_k)]^2 \tag{3-4}$$

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 3-4 could be rewrite as Equation 3-5. It explains that the regularizer calculates the square difference between mean actual value difference and mean predicted value difference among conditions with *S* = Negative and *S* = Positive.

$$[\left(\frac{1}{s0}\sum_{h=1:s0,S=negative} y_h - \frac{1}{s1}\sum_{k=1:s1,S=positive} y_k\right) - \left(\frac{1}{s0}\sum_{h=1:s0,S=negative} \hat{y}_h - \frac{1}{s1}\sum_{k=1:s1,S=positive} \hat{y}_k\right)]^2 \tag{3-5}$$

3.2.1.2. Mean square error penalized regression (MSEP)

In this method, fairness significance is represented by the absolute difference between the MSE when S=Positive and S=Negative, see Equation 3-6. 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 as 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 make the MSE when S=Positive to be the same as the MSE when S=Negative.

$$min\ Loss\_ori + \lambda|\frac{1}{s0}\sum_{h=1:s0,S=negative}(y_h - \hat{y}_h)^2 - \frac{1}{s1}\sum_{k=1:s1,S=positive}(y_k - \hat{y}_k)^2| \quad (3\text{-}6)$$

3.2.1.3. Mean residual difference constrained regression (MRDC)

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 3-7, defines fairness by letting the absolute value of mean residual difference when *S* = Negative to be at least p or at most $\frac{1}{p}$ of absolute mean residual difference when *S* = Positive. When p =0.8, it infers that this method is trying to achieve fairness in terms of the "80 percent rule" [156]: 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.

$$min\ Loss\_ori \quad (3\text{-}7)$$

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

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

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

3.2.1.4. Mean square error constrained regression (MSEC)

The loss function of MSEC is present in Equation 3-8. 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.

$$min\ Loss\_ori \quad (3\text{-}8)$$

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

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

3.2.2. OPTIMIZATION ALGORITHM

Considering the increased complexity of loss function by in-processing methods, derivative-free optimization algorithms that do not use derivatives or finite differences [157] 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 algorithm (PSO), etc. Among these algorithms, DE has been proofed to be effective on solving constrained optimization problems [158]. Therefore, it will be selected as the solver for the optimization problems set by the in-processing methods.

DE, initially proposed by Storn and Price in the year of 1997 [159], is a heuristic approach that gets the global optimal solution by iteratively improving the candidate solution based on an evolutionary process. Its general procedure is presented in Figure 3-2. Detailed description of each step present in this figure 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. Crossover methods mainly include average and intuitive. Then, add the offspring to the population.

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

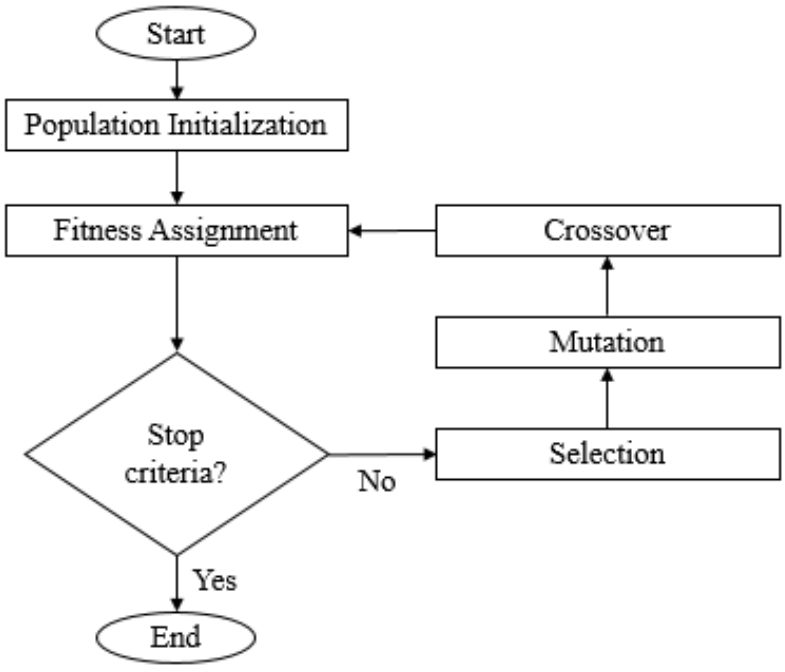


Figure 3-2: General procedure of a DE

3.3. TASK C: FAIRNESS-AWARE DATA-DRIVEN BASED MPC FOR BUILDINGS

The goal of this task is to integrate fairness-aware DDBMs into the MPC for buildings. The simplified schematic of the proposed fairness-aware MPC is shown in Figure 3-3. The proposed MPC is aimed at obtaining optimal future control actions for systems/devices in buildings based on a uniform prediction for future status. The general procedure of the proposed fairness-aware MPC is as below:

Step 1. Data collection. Weather information is collected from websites or weather stations, while building-related data (such as indoor air temperature, energy consumption, and device operation status, etc.) is measured by sensors or HEMS installed in the building (when the study is based on experimental study) or simulated by physical models (when the study is based on numerical study).

Step 2. Fairness-aware data-driven model training and prediction. In this step, users should define the protected attribute among which they wish the predictive performance is uniform. Besides, when the output is discrete labels, pre-processing methods would be used, while in-processing methods would be applied to achieve fairness when the output is continuous data. Users will be able to define p or λ value based on their preference on the fairness and accuracy.

Step 3. Construct the MPC and solve the optimal future control signal. The predicted values from Step 2 will be integrated into the objective function of MPC. The objective function could be aimed at getting the optimal control signals for devices in a building to achieve the minimum electricity cost or energy usage, or maximum peak shifting, while maintain the thermal comfort in a finite horizon of time.

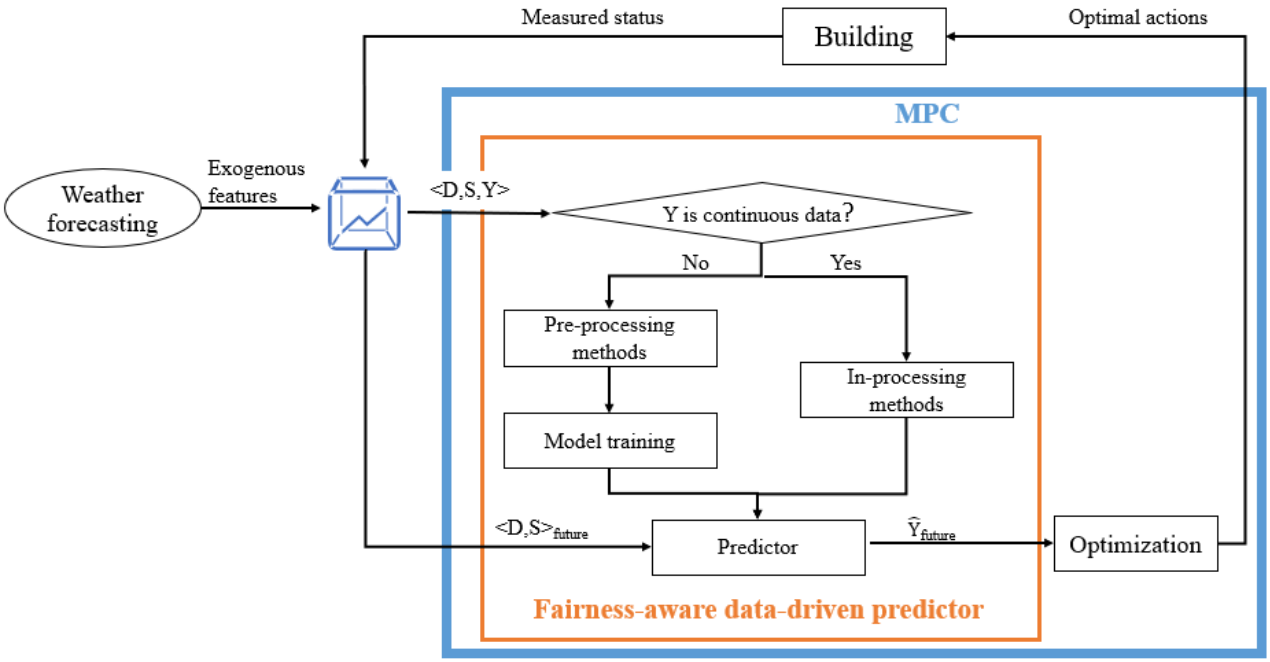


Figure 3-3: Simplified schematic of fairness-aware data-driven based MPC

One example that develops fairness-aware DDBMs for heating load prediction and indoor air temperature classification, and then, integrates these models into MPCs to get the optimal set-point temperature for heating a residential building will be explained in Section 4.3.

CHAPTER FOUR: DESCRIPTION OF CASE STUDIES

Case studies designed to investigate the proposed methodologies for each task are introduced in this chapter.

4.1. TASK A

In this section, two case studies (Case study A-1 and Case study A-2) are designed to compare the proposed pre-processing methods with existing pre-processing methods and to investigate the generalizability of these methods on different data distribution modes, respectively.

4.1.1. CASE STUDY A-1: COMPARISON WITH EXISTING METHODS

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.

4.1.1.1. Data description

Data in this task was collected from an apartment in a residential building in Lyon, France for the year 2016, with one-minute time intervals [19,160]. 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. [19].

Statistical distribution of the collected data is listed in Table 4-1. 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 4-2. Note that the attribute 'Motion Status_total' in Table 4-1 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 4-1: 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 4-2: Description of sensors *[19]* in A-1

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 4-1. For all lighting series, data is mainly distributed in groups NN (around 65% - 68%) and PN (around 21% - 32%).

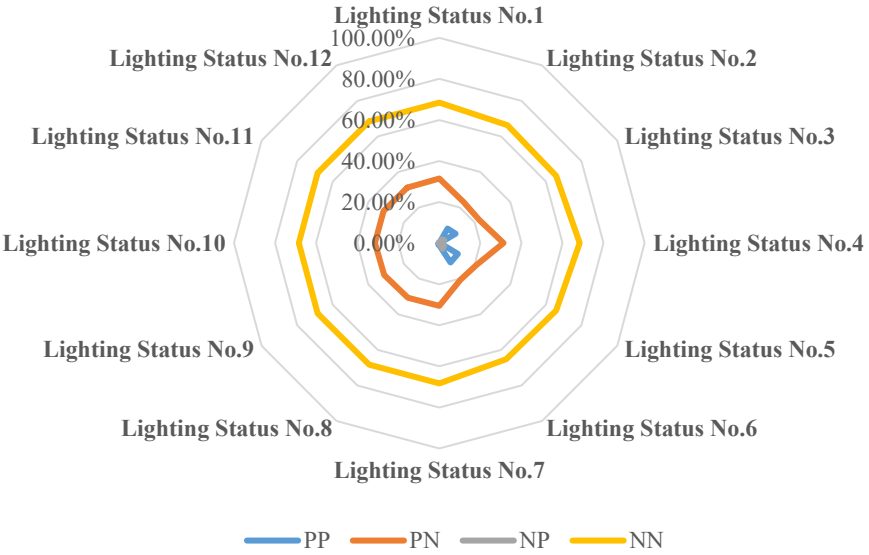


Figure 4-1: Ratios of PP, PN, NP, and NN for Lighting Status No.1 to No.12

4.1.1.2. Case description

As Table 4-3 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 4-3: Description of study cases in A-1

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 4-2 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).

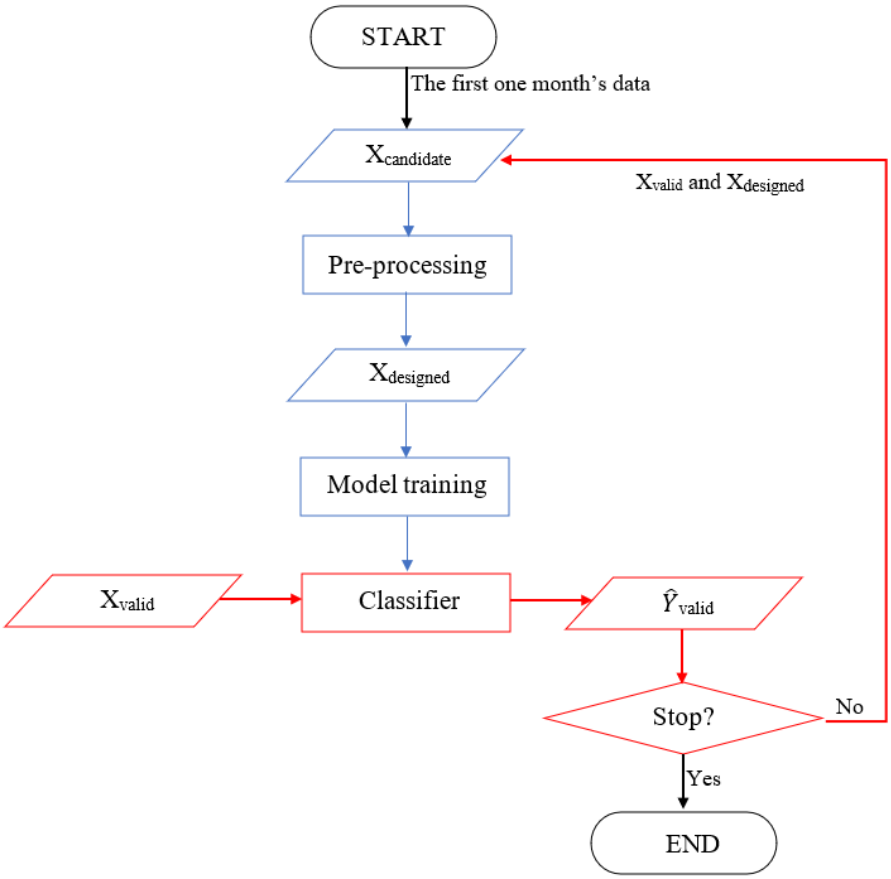


Figure 4-2: Training and validation procedure of study cases A-1

'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 [112]. SVM predicts the class label by maximizing the margin between different categories [161]; 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 [16]. 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 [95]. 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 [96]. 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 [82]. 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 Appendix A.

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

4.1.1.3. Performance evaluation criteria

This section will introduce accuracy measures for evaluating the difference between predicted values and measured values of two-class outputs, and fairness measures that could be helpful to indicate *Type II* fairness achievement through rating the difference of predictive performance between conditions defined by the protected attribute.

Accuracy measures

In this study, accuracy (Equation 2-10), recall (Equation 2-11), and specificity (Equations 2-12) are selected to evaluate the predictive accuracy for the studied two-class classification problems. Accuracy is the overall predictive accuracy, which means the rate of accurate predicted samples to the entire scope of samples. Recall stipulates the true positive rate, which is the rate of accurate prediction when *Y*=Positive. In other words, it reflects the predictive accuracy of *PP* and *NP* conditions. Specificity (also called true negative rate) is the portion of accurate prediction when *Y*=Negative. Thus, it shows the predictive accuracy for validation data in conditions *PN* and *NN*.

Fairness measures

The accuracy measures under different conditions defined by the protected attribute (denoted by S) are defined as the group conditional accuracy measures, i.e., c-Accuracy (Equation 4-1), c-Recall (Equation 4-2) and c-Specificity (Equation 4-3), etc. On the one hand, when *S* = Positive, its conditional accuracy measures are called 1-Accuracy, 1-Recall and 1-Specificity. Note that 1-Accuracy reflects the overall predictive accuracy of *PP* and *PN*, while 1-Recall is the predictive accuracy of *PP* and 1-Specificity shows the accuracy of *PN*. On the other hand, for *S* = Negative, its conditional accuracy measures are called 0-Accuracy, 0-Recall and 0-Specificity. 0-Accuracy is the predictive accuracy of *NP* and *NN*, while 0-Recall is the accuracy of *NP* and 0-Specificity presents the predictive accuracy of *NN*.

$$\mathrm{c-Accuracy} = P\left[\hat{Y} = y \,\middle|\, Y = y, S = s\right] \qquad (4-1)$$

$$\mathrm{c-Recall} = P\left[\hat{Y} = \mathrm{Positive} \,\middle|\, Y = \mathrm{Positive}, S = s\right] \qquad (4-2)$$

$$c - Specificity = P\left[\hat{Y} = Negative \,\middle|\, Y = Negative, S = s\right] \qquad (4-3)$$

where c shows the group conditional accuracy measures; $\mathrm{c} \in [0, 1]$.

To quantify the performance similarity between *S* = Positive and *S* = Negative, accuracy rate (Equation 4-4), recall rate (Equation 4-5) and specificity rate (Equation 4-6) are selected as

fairness measures. Furthermore, to determine whether *Type II* fairness exists (i.e., predictive performance between different conditions is similar enough), "80 percent rule" [156] could be utilized. This rule illustrated that the predictive result is fair when the selected fairness measure is higher than 80%.

$$\text{Accuracy rate} = \frac{\min(1-\text{Accuracy}, 0-\text{Accuracy})}{\max(1-\text{Accuracy}, 0-\text{Accuracy})} \quad (4-4)$$

$$\text{Recall rate} = \frac{\min(1-\text{Recall}, 0-\text{Recall})}{\max(1-\text{Recall}, 0-\text{Recall})} \quad (4-5)$$

$$\text{Specificity rate} = \frac{\min(1-\text{Specificity}, 0-\text{Specificity})}{\max(1-\text{Specificity}, 0-\text{Specificity})} \quad (4-6)$$

4.1.2. CASE STUDY A-2: INVESTIGATION OF GENERALIZABILITY

To investigate the generalizability of the proposed pre-processing techniques, this section designs a case study to apply them to process data collected from 16 apartments. These data could be classified into 5 modes based on their distribution on the output labels (ON/OFF lighting status) and protected attribute labels (ON/OFF motion status). Detailed description of the collected data and study cases is shown in Section 4.1.2.1 and Section 4.1.2.2, respectively.

4.1.2.1. Data description

The data used is collected from 16 apartments in a residential building located in Lyon, France. These 16 apartments present two types of lay-out: Lay-out Type I (Apt #1 to Apt #8) and Lay-out Type II (Apt #9 to Apt #16), as shown in Figure 4-3. Motion status and lighting status are collected from these apartments by presence sensors and lighting sensors, respectively. A detailed description of installed sensors is presented in Table 4-4. Besides, weather information, such as altitude of the sun, global horizontal illuminance, diffuse horizontal illuminance, global horizontal irradiance, and diffuse horizontal irradiance, etc., is collected from a local weather station.

The data was collected for the year 2016 with one-minute time intervals and processed to 5-minute intervals. Further analysis were carried out for missing data and outliers [19,162]. The remaining samples for each light in each apartment are presented in Figure 4-4.

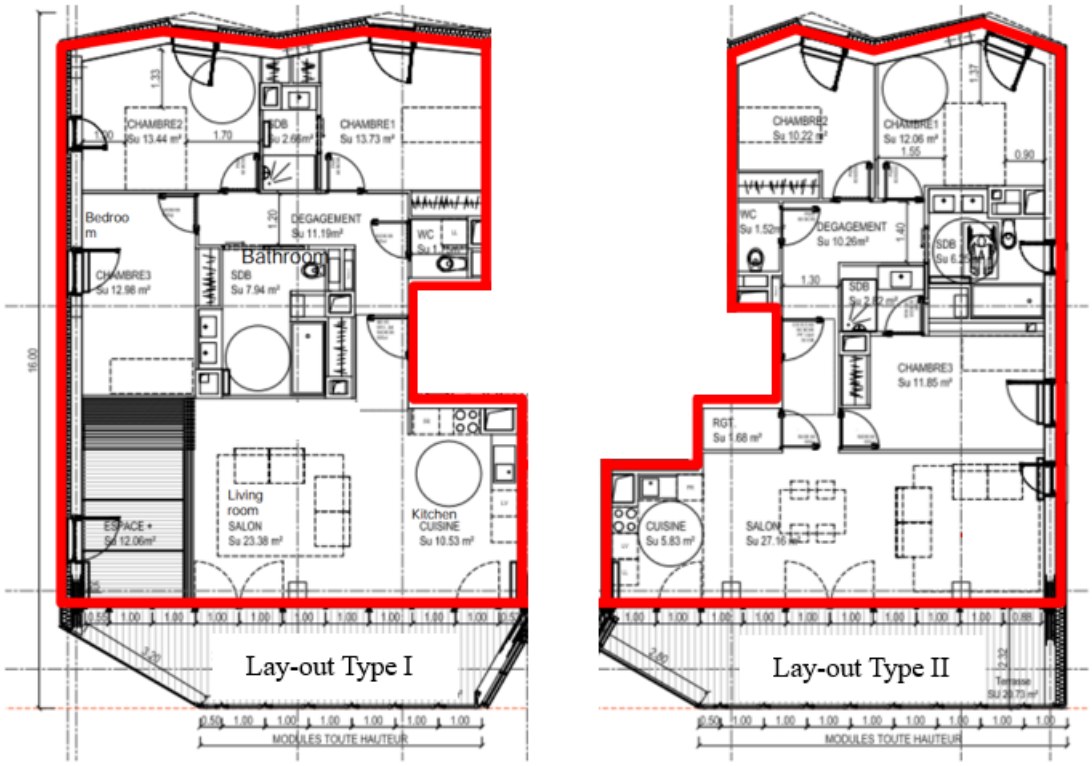


Figure 4-3: Lay-out of studied apartments

Table 4-4: Description of sensors in A-2

Sensor	Company name	Type	Accuracy	No. of sensors in Lay-out Type I	No. of sensors in Lay-out Type II
Presence Detector	Theben	PlanoCentro A-KNX	-(detection area 64 m^2 if seated)	14	14
Lighting Sensor	ABB	KNX Energy Module: EM/S 3.16.3	±2/3/6%	14	13

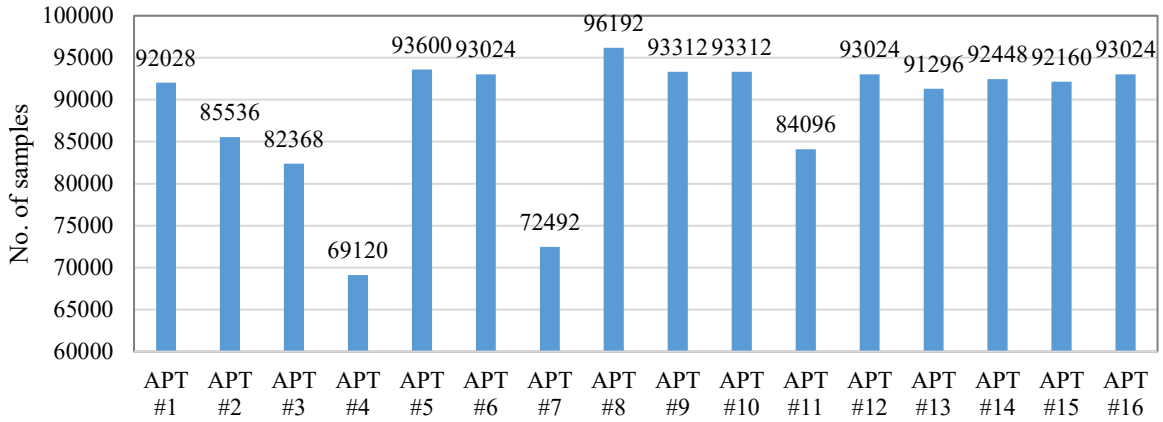


Figure 4-4: Number of samples for each apartment

Except the motion status data collected from presence sensors, one attribute called 'Motion Status_total' is added to each apartment to represent the overall motion status in the corresponding apartment. It is recorded as ON if at least one presence sensor in that apartment detected the ON signal. Besides, if a light is in one status (such as turn OFF) all the time, its corresponding lighting status attribute would be excluded from the collected dataset. As a result, data collected from 155 lights remain in the training dataset and work as data-driven models' outputs.

In this investigation, 'Motion Status_total' is the protected attribute, while lighting status the classifiers' outputs. The ratios of conditions PP, PN, NP, and NN for different lights in these 16 apartments are presented in Figure 4-5. Based on the distribution of PP, PN, NP and NN, the dataset could be separated into 5 modes. A description of each mode is in Table 4-5.

Table 4-5: Description of data modes

Mode No.	Description	Output of Data	Number of Lights
1	The light is OFF most of the time (>90%) and 'Motion Status_total' is OFF 60-70%. **Detailed distribution:** NN contains most samples (55-70%), followed by PN(20-40%), PP(0-8%) and NP(0-8%).	**APT #1:** Light_1, Light_4, Light_7, Light_8, Light_9, Light_10, Light_11, Light_12. **APT #2:** Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10, Light_11, Light_12. **APT #4:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10, Light_11, Light_12. **APT #6:** Light_1, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9. **APT #7:** Light_1, Light_2, Light_3, Light_4. **APT #8:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10, Light_11, Light_12, Light_13. **APT #11:** Light_1, Light_2, Light_3, Light_5, Light_6, Light_7, Light_8. **APT #12:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_7, Light_8 Light_9, Light_10, Light_11. **APT #14:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10. **APT #16:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9.	**92**
2	The light is OFF ~90% of the time, and 'Motion Status_total' is OFF ~70% of the time.	**APT #1:** Light_2, Light_3, Light_5, Light_6. **APT #2:** Light_1. **APT #6:** Light_2.	**6**
3	The light status and 'Motion Status_total' are OFF most of the time (>90%).	**APT #3:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8. **APT #10:** Light_1, Light_2, Light_3, Light_4.	**12**
4	The light is turned OFF most of the time (>90%), and the 'Motion Status_total' is evenly distributed with ~50% 'OFF' labels.	**APT #5:** Light_1, Light_2, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10, Light_11, Light_12. **APT #9:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10, Light_11. **APT #13:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10. **APT #15:** Light_1, Light_2, Light_3, Light_4, Light_5, Light_6, Light_7, Light_8, Light_9, Light_10.	**42**
5	The 'OFF' light status is <50%.	**APT #5:** Light_3. **APT #11:** Light_4. **APT #12:** Light_6.	**3**

		PP	PN	NP	NN
APT #1	Light_1	0.10%	31.46%	0.00%	68.44%
	Light_2	8.19%	23.37%	2.14%	66.30%
	Light_3	9.09%	22.47%	2.80%	65.64%
	Light_4	0.29%	31.27%	0.05%	68.39%
	Light_5	10.44%	21.12%	2.69%	65.75%
	Light_6	10.86%	20.70%	3.20%	65.24%
	Light_7	1.01%	30.55%	0.11%	68.33%
	Light_8	0.84%	30.72%	0.15%	68.29%
	Light_9	0.57%	30.99%	0.03%	68.41%
	Light_10	0.18%	31.38%	0.02%	68.42%
	Light_11	0.31%	31.25%	0.02%	68.42%
	Light_12	0.32%	31.24%	0.02%	68.42%
APT #2	Light_1	9.00%	24.47%	10.25%	56.28%
	Light_2	3.28%	30.20%	0.32%	66.20%
	Light_3	2.47%	31.00%	0.50%	66.02%
	Light_4	1.81%	31.67%	0.43%	66.09%
	Light_5	1.09%	32.39%	0.03%	66.50%
	Light_6	2.18%	31.30%	0.07%	66.46%
	Light_7	1.49%	31.99%	0.05%	66.48%
	Light_8	0.81%	32.67%	0.09%	66.44%
	Light_9	2.34%	31.14%	0.09%	66.43%
	Light_10	0.87%	32.61%	0.06%	66.46%
	Light_11	1.76%	31.72%	0.18%	66.35%
	Light_12	2.93%	30.55%	0.23%	66.29%
APT #3	Light_1	1.67%	8.33%	1.42%	88.57%
	Light_2	3.12%	6.89%	0.85%	89.15%
	Light_3	1.80%	8.21%	0.25%	89.75%
	Light_4	4.52%	5.48%	0.72%	89.27%
	Light_5	5.02%	4.98%	0.86%	89.13%
	Light_6	1.83%	8.18%	0.29%	89.70%
	Light_7	0.77%	9.24%	0.23%	89.76%
	Light_8	0.40%	9.61%	0.16%	89.84%
APT #4	Light_1	3.40%	32.73%	0.51%	63.37%
	Light_2	4.60%	31.52%	0.57%	63.30%
	Light_3	3.09%	33.03%	0.30%	63.58%
	Light_4	0.36%	35.76%	0.03%	63.85%
	Light_5	4.93%	31.19%	1.97%	61.90%
	Light_6	2.73%	33.40%	2.83%	61.05%
	Light_7	0.23%	35.89%	0.01%	63.87%
	Light_8	0.89%	35.23%	0.05%	63.83%
	Light_9	5.15%	30.97%	0.23%	63.64%
	Light_10	1.68%	34.45%	0.16%	63.72%
	Light_11	2.80%	33.32%	0.08%	63.80%
	Light_12	3.76%	32.36%	0.14%	63.74%

		PP	PN	NP	NN
APT #5	Light_1	5.15%	45.33%	0.06%	49.46%
	Light_2	9.79%	40.69%	0.17%	49.34%
	Light_3	32.15%	18.34%	30.23%	19.29%
	Light_4	5.90%	44.58%	0.23%	49.29%
	Light_5	3.84%	46.64%	0.27%	49.25%
	Light_6	14.99%	35.49%	0.48%	49.03%
	Light_7	1.27%	49.22%	0.59%	48.93%
	Light_8	0.85%	49.63%	0.04%	49.48%
	Light_9	9.83%	40.65%	0.02%	49.50%
	Light_10	4.27%	46.21%	0.18%	49.33%
	Light_11	4.42%	46.06%	0.01%	49.51%
	Light_12	6.43%	44.06%	0.02%	49.49%
APT #6	Light_1	4.81%	22.92%	0.19%	72.08%
	Light_2	8.51%	19.23%	0.66%	71.61%
	Light_3	2.04%	25.69%	0.07%	72.20%
	Light_4	6.00%	21.74%	0.35%	71.92%
	Light_5	1.18%	26.55%	0.06%	72.21%
	Light_6	1.26%	26.48%	0.15%	72.12%
	Light_7	1.50%	26.24%	0.13%	72.14%
	Light_8	1.47%	26.26%	0.09%	72.17%
	Light_9	2.03%	25.71%	0.10%	72.17%
APT #7	Light_1	0.54%	31.62%	0.36%	67.49%
	Light_2	1.50%	30.65%	0.40%	67.45%
	Light_3	2.88%	29.27%	0.19%	67.66%
	Light_4	0.48%	31.67%	0.22%	67.63%
APT #8	Light_1	3.32%	25.84%	1.28%	69.56%
	Light_2	3.20%	25.96%	1.54%	69.30%
	Light_3	3.98%	25.18%	2.81%	68.03%
	Light_4	3.12%	26.04%	1.70%	69.14%
	Light_5	3.44%	25.72%	1.17%	69.67%
	Light_6	4.02%	25.14%	1.62%	69.23%
	Light_7	2.30%	26.86%	0.85%	69.99%
	Light_8	1.93%	27.23%	0.45%	70.39%
	Light_9	0.80%	28.36%	0.35%	70.49%
	Light_10	2.98%	26.18%	0.92%	69.92%
	Light_11	2.29%	26.87%	0.87%	69.97%
	Light_12	2.17%	26.99%	0.82%	70.03%
	Light_13	3.34%	25.82%	3.11%	67.74%

		PP	PN	NP	NN
APT #9	Light_1	4.92%	43.64%	0.56%	50.87%
	Light_2	3.82%	44.74%	0.10%	51.34%
	Light_3	5.90%	42.67%	0.39%	51.04%
	Light_4	7.61%	40.95%	1.42%	50.02%
	Light_5	5.31%	43.26%	0.49%	50.95%
	Light_6	3.36%	45.20%	0.39%	51.04%
	Light_7	8.07%	40.50%	0.51%	50.93%
	Light_8	6.34%	42.23%	0.38%	51.06%
	Light_9	5.27%	43.30%	1.27%	50.17%
	Light_10	4.59%	43.97%	0.28%	51.16%
	Light_11	3.08%	45.48%	0.27%	51.16%
APT #10	Light_1	1.94%	3.62%	0.27%	94.17%
	Light_2	1.36%	4.20%	0.13%	94.32%
	Light_3	2.23%	3.33%	0.20%	94.25%
	Light_4	1.64%	3.92%	0.23%	94.21%
APT #11	Light_1	2.33%	30.47%	1.06%	66.14%
	Light_2	3.67%	29.13%	0.11%	67.09%
	Light_3	4.00%	28.80%	0.37%	66.82%
	Light_4	32.23%	0.58%	66.24%	0.96%
	Light_5	3.07%	29.73%	0.08%	67.12%
	Light_6	1.26%	31.54%	0.25%	66.94%
	Light_7	1.30%	31.51%	0.04%	67.16%
	Light_8	0.20%	32.60%	0.10%	67.09%
APT #12	Light_1	1.69%	39.37%	0.13%	58.81%
	Light_2	4.08%	36.98%	0.28%	58.67%
	Light_3	3.20%	37.86%	1.44%	57.51%
	Light_4	5.09%	35.97%	1.95%	57.00%
	Light_5	0.36%	40.69%	0.09%	58.85%
	Light_6	22.27%	18.78%	28.14%	30.81%
	Light_7	1.09%	39.97%	0.26%	58.69%
	Light_8	6.57%	34.48%	0.84%	58.11%
	Light_9	8.05%	33.01%	1.14%	57.80%
	Light_10	3.20%	37.86%	0.47%	58.47%
	Light_11	2.13%	38.93%	0.15%	58.80%

		PP	PN	NP	NN
APT#13	Light_1	1.20%	53.43%	0.04%	45.33%
	Light_2	4.43%	50.20%	0.41%	44.96%
	Light_3	3.22%	51.42%	0.28%	45.09%
	Light_4	7.72%	46.91%	0.91%	44.45%
	Light_5	2.34%	52.30%	0.07%	45.30%
	Light_6	1.57%	53.06%	0.04%	45.33%
	Light_7	4.25%	50.38%	0.10%	45.27%
	Light_8	3.26%	51.37%	0.11%	45.26%
	Light_9	3.64%	51.00%	0.10%	45.27%
	Light_10	1.14%	53.49%	0.09%	45.27%
APT#14	Light_1	3.76%	32.27%	1.14%	62.83%
	Light_2	2.83%	33.20%	0.09%	63.88%
	Light_3	0.77%	35.26%	0.08%	63.89%
	Light_4	1.61%	34.42%	0.18%	63.79%
	Light_5	2.17%	33.87%	0.27%	63.70%
	Light_6	1.80%	34.23%	0.28%	63.69%
	Light_7	6.81%	29.22%	3.32%	60.64%
	Light_8	3.75%	32.28%	1.16%	62.81%
	Light_9	3.26%	32.77%	1.20%	62.77%
	Light_10	1.26%	34.77%	0.10%	63.87%
APT#15	Light_1	5.89%	44.84%	0.23%	49.05%
	Light_2	3.33%	47.39%	0.33%	48.95%
	Light_3	2.20%	48.53%	0.08%	49.20%
	Light_4	4.41%	46.32%	0.16%	49.11%
	Light_5	3.50%	47.23%	0.58%	48.70%
	Light_6	7.70%	43.03%	0.49%	48.79%
	Light_7	2.29%	48.43%	0.37%	48.90%
	Light_8	1.07%	49.65%	0.02%	49.25%
	Light_9	8.10%	42.62%	0.19%	49.08%
	Light_10	2.39%	48.34%	0.05%	49.22%
APT#16	Light_1	0.74%	35.89%	0.06%	63.31%
	Light_2	3.20%	33.42%	1.08%	62.29%
	Light_3	1.56%	35.07%	0.08%	63.29%
	Light_4	2.47%	34.16%	0.44%	62.94%
	Light_5	2.95%	33.67%	1.16%	62.21%
	Light_6	2.75%	33.87%	0.74%	62.64%
	Light_7	1.90%	34.73%	0.06%	63.31%
	Light_8	1.63%	35.00%	0.23%	63.14%
	Light_9	1.08%	35.55%	0.09%	63.28%

Figure 4-5: Ratios of PP, PN, NP, and NN among 16 apartments

4.1.2.2. Case description

Eight kinds of cases are designed and named by the utilized pre-processing techniques, see Table 4-6. The suffix (LR or NB) for RPS and SPS indicates the ranker (LR or NB) utilized in the corresponding pre-processing technique. Note that these case studies are applied to 16 apartments with totally 155 lightings. For each kind of cases, 4 types of classifiers (i.e., SVM, ANN, LR, NB) are utilized. Therefore, a total of 4960 cases are investigated (the combination of 8 types of cases, 4 types of classifiers and 155 types of outputs (8*4*155=4960)). In other words, there are 2944 (8*4*92) cases for Mode 1, 192 (8*4*6) cases for Mode 2, 384 (8*4*12) cases for Mode 3, 1344 (8*4*42) cases for Mode 4, and 96 (8*4*3) cases for Mode 5, respectively.

Table 4-6: Description of study cases in A-2

Case Type	**Pre-processing techniques**	**Inputs**	**Classifiers**	**Output**
Reference Case		Hour of The Day, Day of The Week, Time of The Day, Altitude of The Sun, Global Horizontal Illuminance, Diffuse Horizontal Illuminance, Global Horizontal Irradiance, Diffuse Horizontal Irradiance, Motion Status	SVM, ANN, LR, NB	Lighting status (ON/OFF)
RS	Random Sampling			
SS	Sequential Sampling			
SBS	Sequentially Balanced Sampling			
RPS_LR	Reversed Preferential Sampling with using Logistic Regression as a ranker			
RPS_NB	Reversed Preferential Sampling with using Naïve Bayes as a ranker			
SPS_LR	Sequential Preferential Sampling using Logistic Regression as a ranker			
SPS_NB	Sequential Preferential Sampling with using Naïve Bayes as a ranker			

Training and validation procedure of these cases are present in Figure 4-6. For reference cases, classifiers are trained by previous four weeks' data to predict next week's lighting status. The training data is updated once a week by the newly observed data. The procedure for other cases is: Firstly, the $X_{candidate}$ is processed by the corresponding pre-processing technique to produce the $X_{designed}$ that contains four weeks' data. Next, $X_{designed}$ is used to train classifiers that will be used to predict one-week ahead lighting status. Then, the newly observed next week's data and $X_{designed}$ are updated as $X_{candidate}$ for next time's prediction.

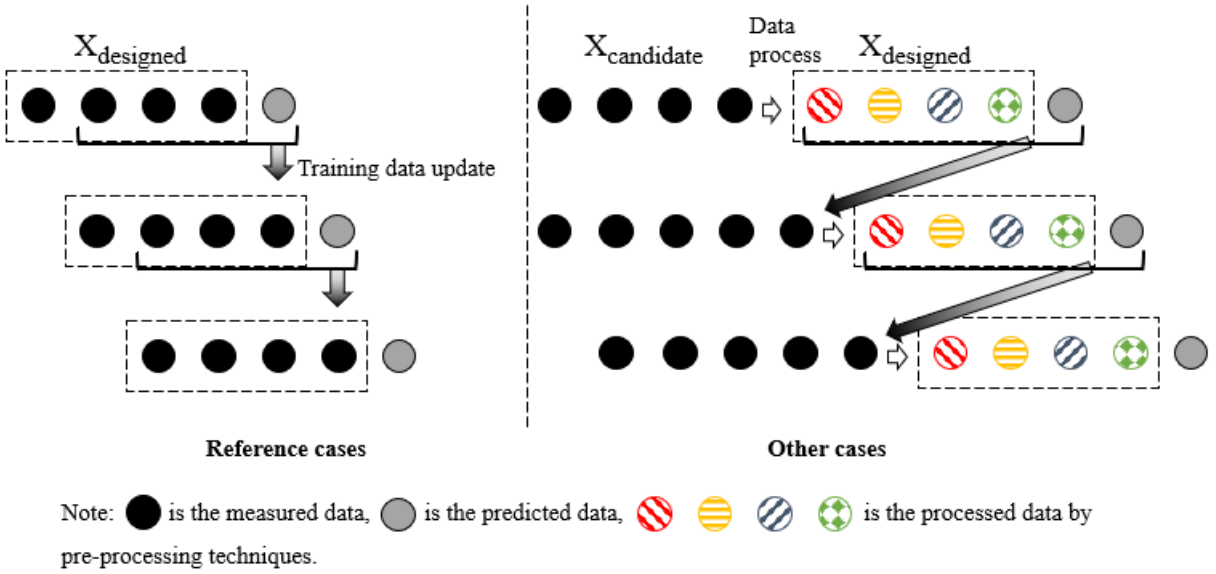


Figure 4-6: Training and validation procedure of study cases A-2

To test if these pre-processing techniques show consistent effect on different classifiers' predictive performance, four types of commonly used classifiers (SVM, ANN, LR, and NB) with different mathematical nature are developed. Among these classifiers, SVM predicts the class label by maximizing the margin between different categories [161]; ANN predicts the output by trained activation functions of neurons in hidden layers and output layer; LR predicts the possibility of an object belonging to a positive class using a logistic function [95]; and NB conducts a prediction by applying Bayes' algorithm with the 'naive' assumption of conditional independence between attributes given the class label value [96]. NB classifiers are generally classified as three types: Gaussian NB, Multinomial NB, and Bernoulli NB. This study uses Gaussian NB.

As the predictive performance of classifiers is affected by their hyperparameters [82], the hyperparameters of classifiers in reference case are optimized by a RandomizedSearchCV function in Python [163]. It optimizes hyperparameters by randomly selecting a chosen number of hyperparametric pairs from a given domain and testing only those. The search space of hypermeters for each kind of classifier is listed in Table 4-7. Moreover, this case study is aimed at investigating the effect of pre-processing techniques on predictive results instead of the effect of classifiers' hyperparameters, thus, the same kind of classifiers in other cases use the same hyperparameters as in reference cases.

Table 4-7: Search space for hyperparameter optimization

Classifier	Hyperparameter	Description	Search space
SVM	C	Regularization parameter. The strength of the regularization is inversely proportional to C.	loguniform(1e0, 1e3)
ANN	solver	The solver for weight optimization	'adam', 'lbfgs'
	hidden layer sizes	The i-th element represents the number of neurons in the i-th hidden layer.	(5,2), (3,3), (5,5), (4,4)
LR	solver	Algorithm to use in the optimization problem.	'newton-cg', 'lbfgs', 'liblinear'
	C	Regularization parameter. The strength of the regularization is inversely proportional to C.	loguniform(1e0, 1e3)
	Class weight	Weights associated with classes	'balanced', None
NB	var_smoothing	Portion of the largest variance of all features that is added to variances for calculation stability.	logspace(0,-9, num=100)

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

4.2. TASK B

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 when training regression 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 4.2.1, while the study cases are explained in Section 4.2.2.

4.2.1. 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 Lay-out Type 1 as shown in Figure 4-3. The data collection devices are explained in detail in [19,160,162,164]. 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. Besides, weather information, such as ambient temperature, ambient humidity, wind speed, wind 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 [21], 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 [165], 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.

The NHEC of lighting and plug-ins is the output of data-driven models. Its distribution is shown in Figure 4-7. 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 2-1. 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 4-8.

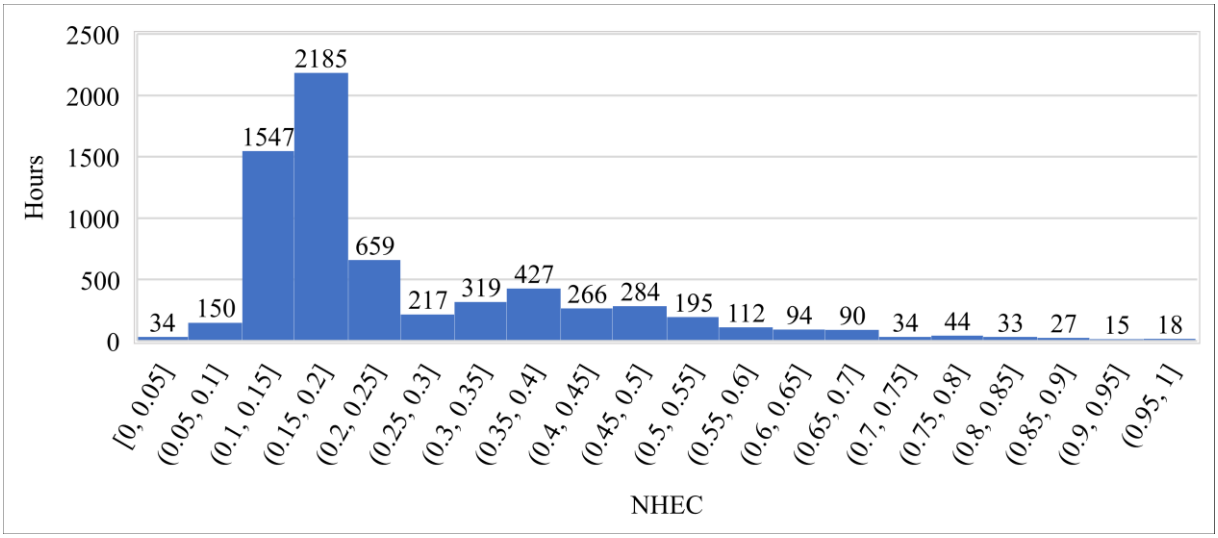


Figure 4-7: NHEC distribution

Besides, as energy consumption may belong to time series data that historical energy consumption would affect future values [165], previous 24 hours' consumption data and previous 168th hourly consumption are also added to the processed dataset as candidate features. Overall, there are 106 candidate features. To select the most representative features for prediction, correlation between the candidate features and the output is calculated by Equation 2-1. 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 4-8.

Table 4-8: 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.

4.2.2. CASE DESCRIPTION

As the start point of investigating in-processing fairness improvement methods in building engineering domain, a relatively simple regression model, i.e., linear regression (see Equation 2-3), is used in this study to predict the normalized hourly energy consumption (NHEC).

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 4-9. MSE is the *Loss_ori* for these cases.

Table 4-9: Description of study cases for Task B

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 by using the scikit-opt package [166], and its hyperparameters are shown in Table 4-10. 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 4-10: 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.

4.3. TASK C

To investigate the effect of integrating fairness-aware models into MPC, a case study could be done to compare the proposed fairness-aware MPC with the traditional MPC through applying these controllers to control the electrically heated floor (EHF) system in a residential building for the winter of 2021 (January to March).

4.3.1. EXPERIMENTAL BUILDING

The experimental building (as shown in Figure 4-8) used to implement controllers is a traditional residential building located in Montreal, Quebec, Canada. It was built in the year 1960, with a building area of 104 m^2. There are 6 rooms in the basement and 6 rooms in the ground floor.

Figure 4-8: Experimental building used in the case study of Task 3

In the city where the experimental building is located, distinct electricity prices for peak periods and off-peak periods (as shown in Table 4-11) are implemented to encourage consumers to shift their electricity demand and ease the press of the grid during winter. It was mentioned that peak periods often occur during the early morning and late afternoon. Therefore, in this study, to simplify the model complexity for MPC, the peak periods are assumed as 6 am to 12 pm and 18 pm to 0 am, while off-peak periods include 0 am to 6 am and 12 pm to 18 pm.

Table 4-11: Electricity price during winter implemented in Montreal, Canada *[167]*

		Periods	
	Condition	Peak period	Off-peak period
Electricity price (¢/kWh)	< 40 kWh/per day	50	3.98
Electricity price (¢/kWh)	> 40 kWh/per day	50	7.03
Subscription fee (¢/day)	-	40.64	

4.3.2. MODIFIED TRNSYS MODEL

The building was originally simulated as a multi-zone TRNSYS (TRaNsient SYStems simulation program) model created and validated by Aongya [168]. In this model, there are 16 zones, including hallways, the attic, and 12 rooms. Occupancy status, lighting energy consumption, and appliance energy consumption are modeled with a measured schedule. The building is heated by electric baseboard. Then, to investigate the applicability of electrically heated floor (EHF) on peak shifting, Thieblemong [169] replaced the heating system in the basement by a commonly used EHF system in Quebec, Canada. The assembly of EHFs is present in Figure 4-9, while the properties of its materials are listed in Table 4-12.

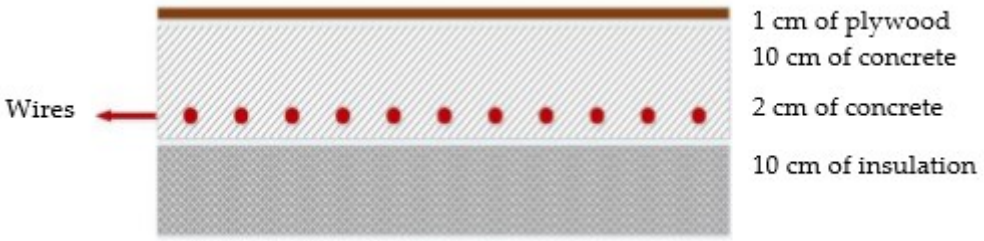


Figure 4-9: Assembly of EHFs *[169]*

Table 4-12: Thermophysical properties of floor layers

Material	**Conductivity (W/mk)**	**Specific Heat (kJ/kgK)**	**Density (kg/m^3)**
Concrete	2.25	0.99	2200
Insulation (XPS)	0.04	1.5	35
Floor Covering (plywood)	0.164	1.63	670

In this study, as the start point of investigation the applicability of fairness-aware data-driven based MPC, the building model is further simplified into a single-zone model that combines the ground floor as one zone heated by EHFs. The schematic of the modified TRNSYS model is presented in Figure 4-10. Detailed description of these components will not be presented in this thesis, as it is not the focus of this study and plenty of explanation for TRNSYS models have been given in previous studies [165,170,171].

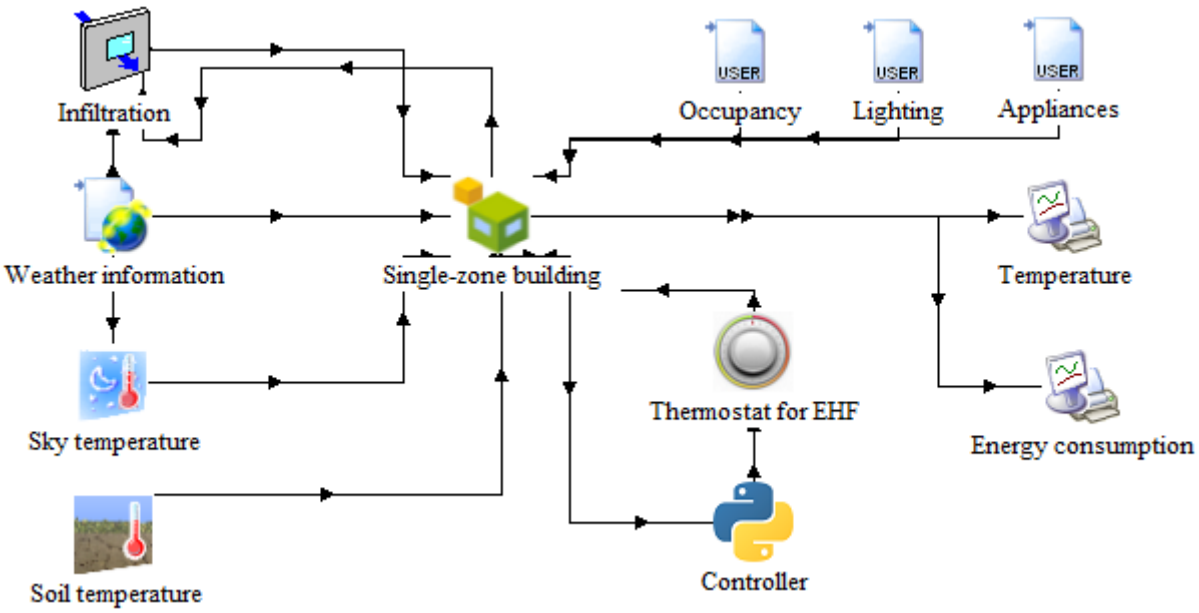


Figure 4-10: Schematic of the TRNSYS model

4.3.3. FAIRNESS-AWARE MPC DEVELOPMENT

In this section, the fairness-aware MPC will be developed following the procedure illustrated in Section 3.3. Detailed explanation for each step is present as below:

Step 1. Data collection.

To collect the training data, the TRNSYS model is simulated based on weather data collected from January to March from the weather station located at Montréal-Pierre Elliott Trudeau International Airport for the year of 2011 to 2021. The time interval is 1 hour. The building is heated by the EHF with a random hourly integer set-point within the range of [18 °C, 24 °C]. Data got from the TRNSYS model include ambient temperature, energy consumed by the EHF, and indoor air temperature. Because DDBMs could be time series prediction, 12 hours' time lag of set-point temperature, energy consumption, and indoor air temperature are added as candidate features.

Step 2. Fairness-aware data-driven model training and prediction.

Two DDBMs would be developed to predict the energy consumption and indoor air temperature separately. The energy prediction would be used to calculate the electricity bill, while the indoor air temperature prediction would be used to constrain the set-point in order to meet thermal comfort.

The energy prediction model (see Equation 4-7) is a SVM model with a linear kernel that predicts energy consumption for a 6-hour period (off-peak or peak) based on the hourly setpoint during that period ($T_{set,i}$), and hourly ambient temperature during that period ($T_{ambient,i}$) and

previous period ($T_{ambient,i-1}$). The reason for model selection and feature selection is presented in Appendix C

$$\widehat{Q_i} = f(T_{set,i}, T_{ambient,i}, T_{ambient,i-1}) \quad (4\text{-}7)$$

where $\widehat{Q_i}$ is the predicted energy consumption during *i-th* period, kWh; i=1 means the period of 0am to 6am, i=2 means the period of 6am to 12pm, i=3 is the period of 12pm to 6pm, and i=4 is the period of 6pm to 0am; $T_{set,i}$ is a list of hourly set-point at *i-th* period, °C; $T_{ambient,i}$ is a list of hourly ambient temperature at *i-th* period, °C.

The indoor air temperature prediction model (see Equation 4-8) is actually a classification model that determines whether the indoor air temperature is lower than the threshold temperature for thermal comfort (such as 21 °C required in Montreal [172]). Here, SVM with a 'linear' kernel function is selected as the indoor air temperature prediction model, while $T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$, and $T_{ambient,i-1}$ are input features. The reason for model selection and feature selection is also presented in Appendix C

$$\widehat{T_{min,i}} = f(T_{set,i}, T_{set,i-1}, T_{ambient,i}, T_{ambient,i-1}) \quad (4\text{-}8)$$

where $\widehat{T_{min,i}}$ is the binary label that illustrates if the predicted minimum indoor air temperature during *i-th* period is lower than 21°C, $\widehat{T_{min,i}} \in [-1,1]$. $\widehat{T_{min,i}}$=-1 means that during the predicted period, indoor air temperature is higher than 21°C.

In this case study, data simulated from year 2011 to year 2020 is used as $X_{candidate}$, while the data for year 2021 is used as validation dataset. The effect of using fairness improvement pre-processing methods (RS and RPS) are investigated.

Step 3. Construct the MPC and solve the optimal future control signal.

The goal of this MPC is to get the hourly heating set-point temperature that could minimize the daily electricity bill. The indoor air temperature should be controlled to be higher than 21 °C in the future 24 hours, while the set-point temperature should be integer and within the range of [18 °C, 24 °C]. The objective function of MPC is presented in Equation 4-9.

$$\min \sum_{i=1:4} \widehat{Q_i} * Price_i \quad (4\text{-}9)$$

Subject to

$$\widehat{T_{min,i}} = \text{Negative}, \forall i \in [1,4],$$

$$18\ °C \leq T_{set,ji} \leq 24\ °C, \forall i \in [1,4], j \in [1,6]$$

In this case study, 'MPC_ReferenceCase', 'MPC_RS', and 'MPC_RPS' are developed to compare fairness-aware MPCs developed based on the training dataset processed by RS or RPS to the traditional MPC, i.e., 'MPC_ReferenceCase', that does not consider fairness.

CHAPTER FIVE: RESULTS AND DISCUSSION

In this chapter, the results obtained from the case studies for each task is analyzed.

5.1. TASK A

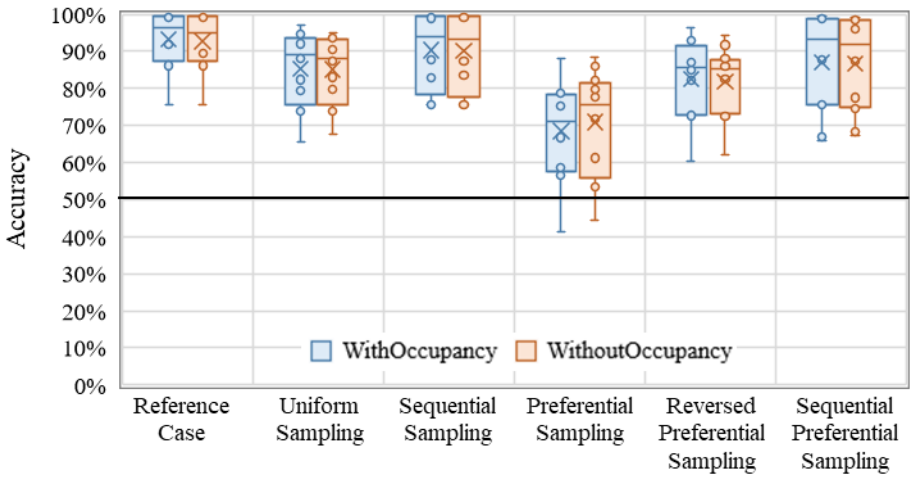
5.1.1. RESULTS FOR CASE STUDY A-1

5.1.1.1. A-1 Results: accuracy measures

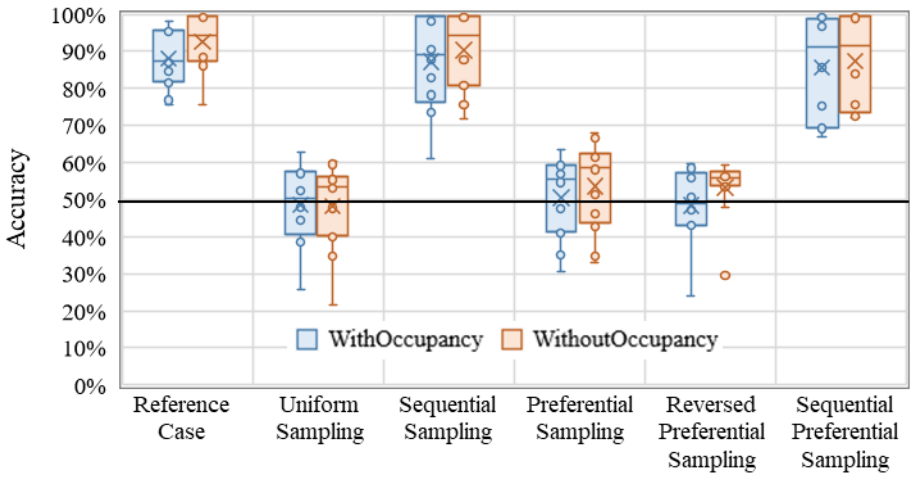
The overall predictive accuracy for Lighting Status No.1 to No.12 under different pre-processing strategies and classification models is statistically analyzed in Figure 5-1. Note that a circle point in this box and whisker plot represent overall predictive accuracy (y axis) for one type of lighting 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 5-1 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 5-1(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 3.1.3 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 5-1 subfigure indicates the acceptable lower bound for all presented classifiers. Cases with accuracy lower than this line should be abandoned.

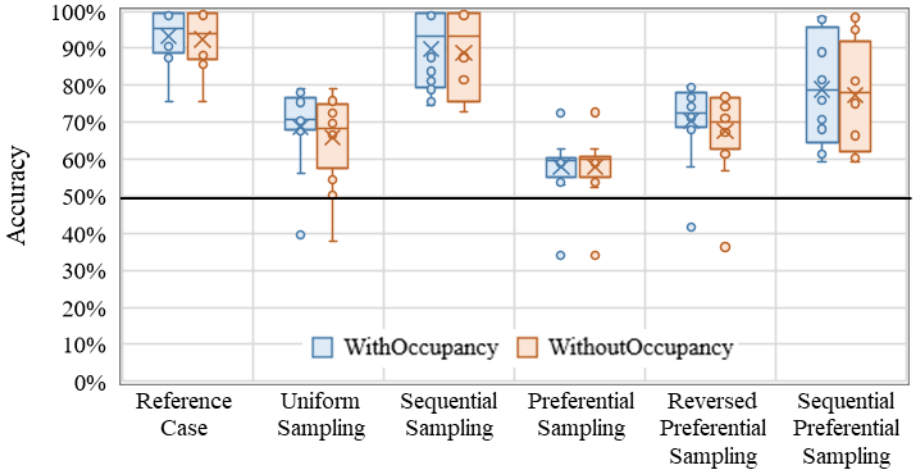
Further, in all cases the lowest point for each box (worst accuracy) in Figure 5-1 presents the accuracy for 'Lighting Status No.1'. In Table 4-1, 'Lighting Status No.1' is OFF most of the time. This reveals that an unbalanced training dataset could cause worse predictive accuracy.



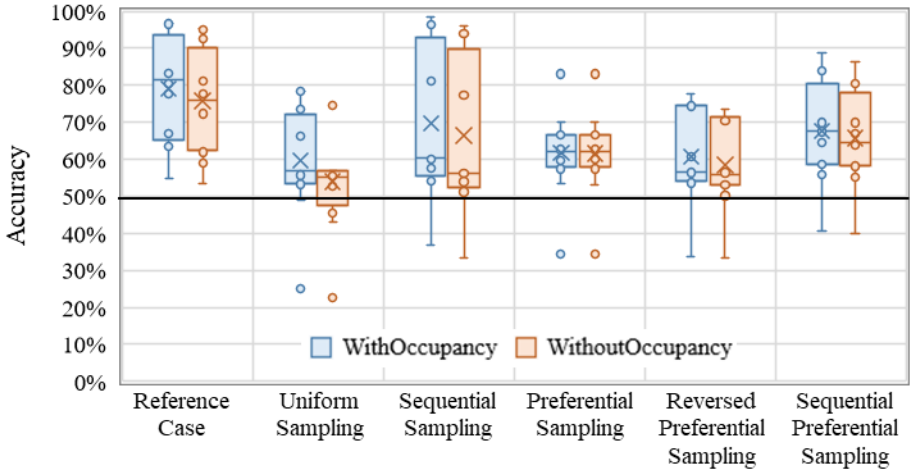
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

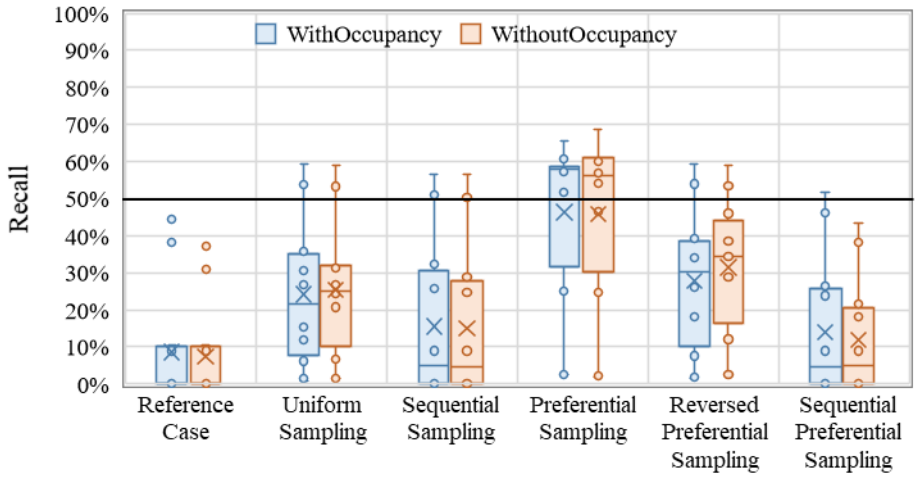
Figure 5-1: Accuracy under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

Figure 5-2 shows recall in different cases. Like Figure 5-1, recall differences between cases using 'WithOccupancy' or 'WithoutOccupancy' as inputs are ignorable. However, in contrast to Figure 5-1, 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.

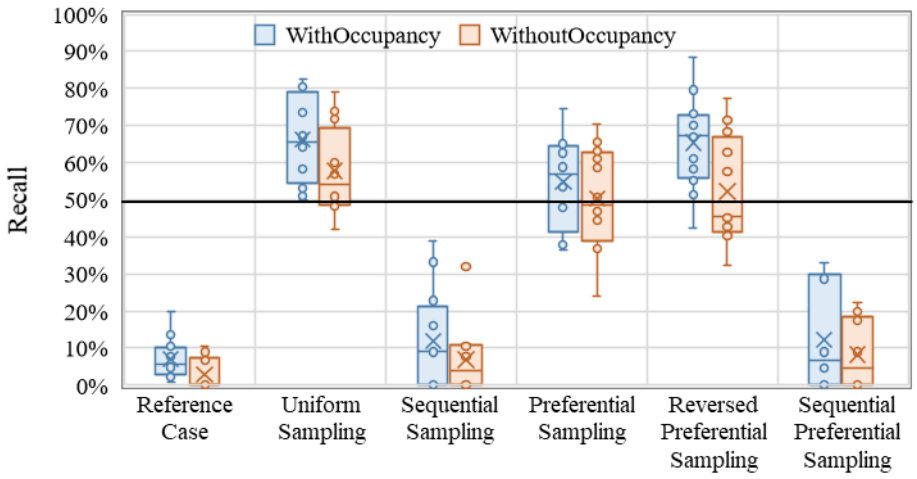
On the other hand, in Figure 5-2 (a)–(c), the lowest point for each box usually presents recall for lighting that is OFF most of the time (e.g., 'Lighting Status No.1', 'Lighting Status No.4', 'Lighting Status No.11', or 'Lighting Status No.12'). This result implies that recall (true positive rate) could be improved by increasing the number of training data with a positive observed class label.

Figure 5-3 shows specificity in different cases. For each case, specificity resembles accuracy as most observed target values are negative.

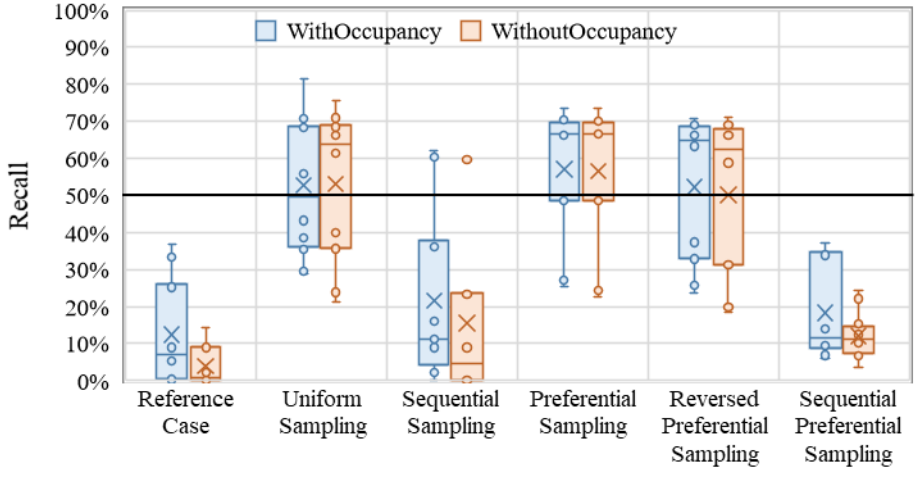
Overall, users should choose proper data pre-processing strategies and classifiers based on their demand. For instance, if better overall accuracy is prioritized, the reference case would be a good choice when the original training dataset is mainly negative class label data. However, if recall also 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.



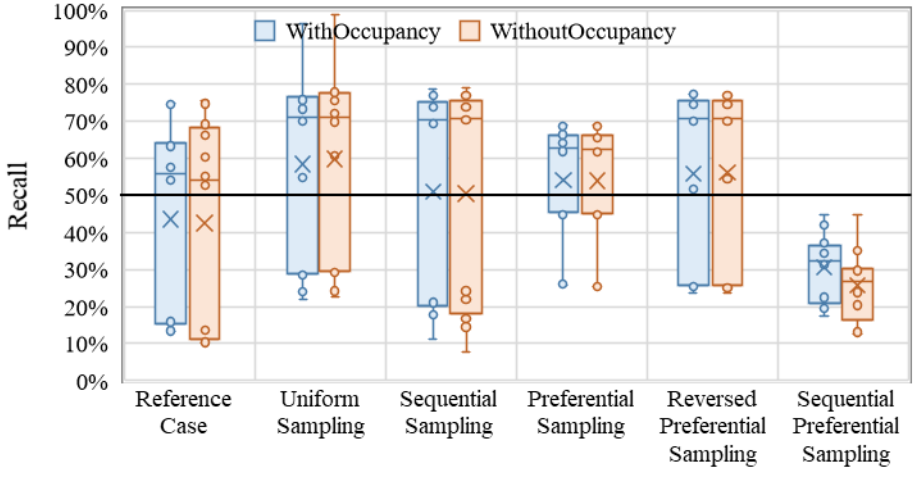
(a) SVM



(b) ANN

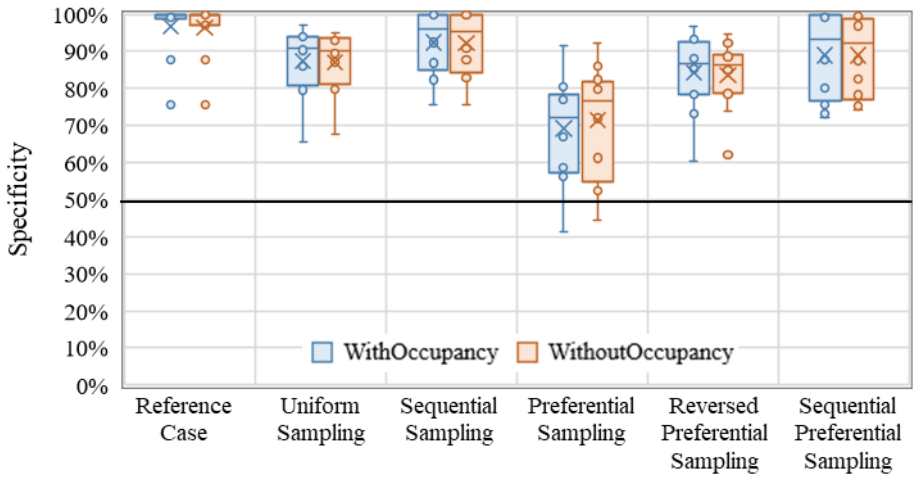


(c) Logistic Regression

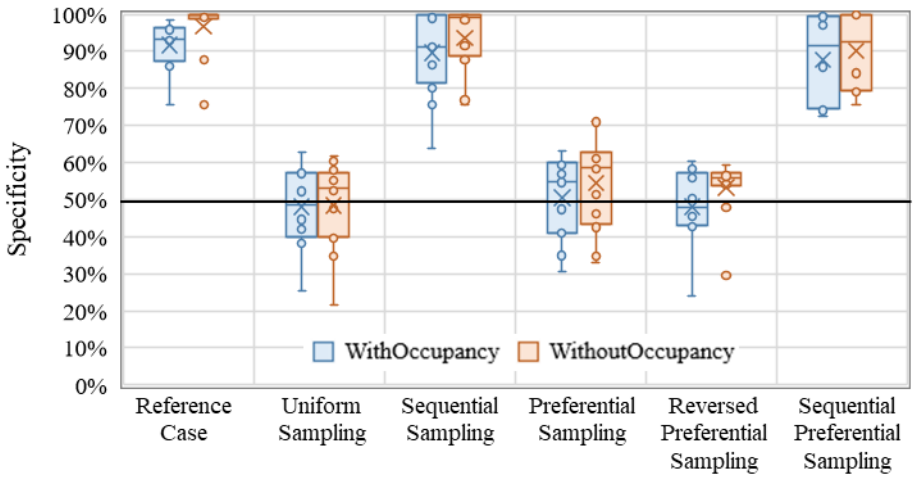


(d) Naïve Bayes

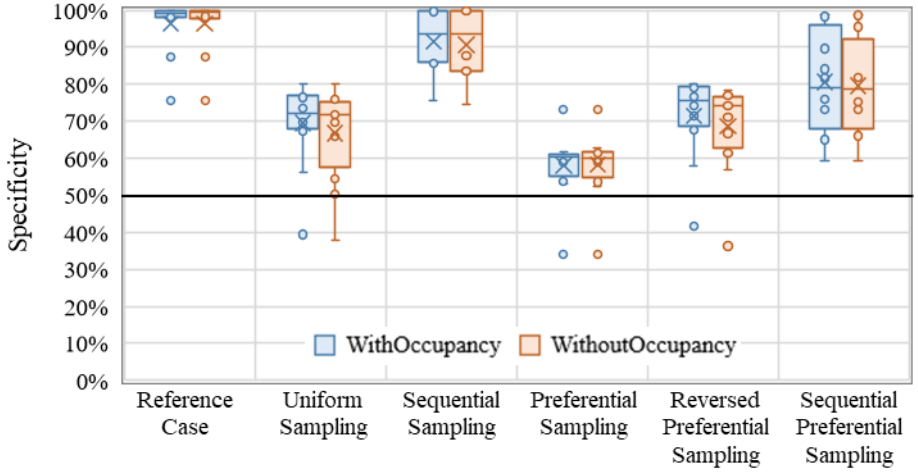
Figure 5-2: Recall under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



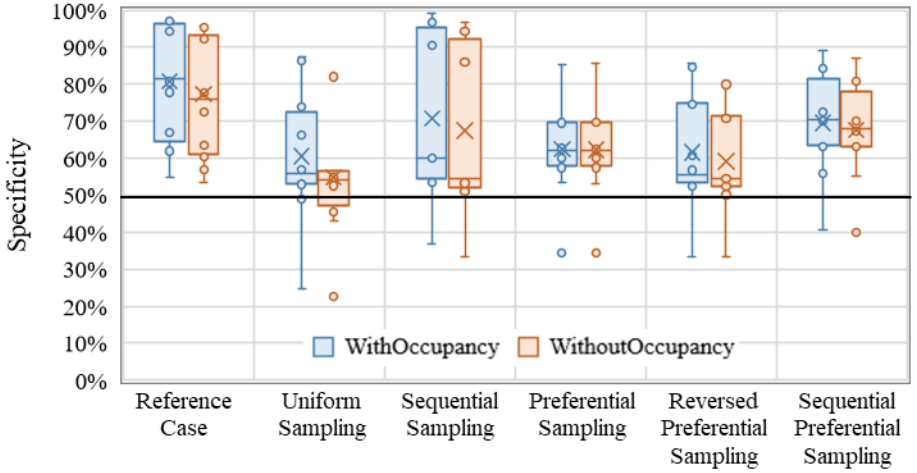
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

Figure 5-3: Specificity under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

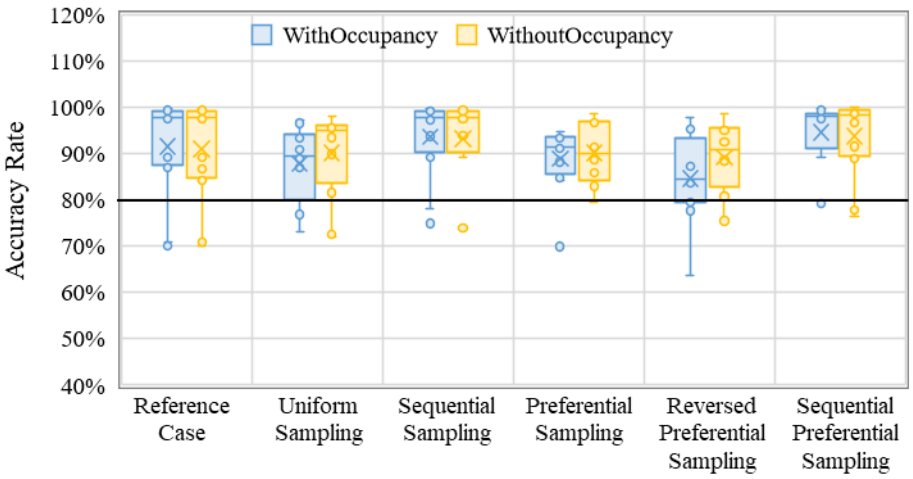
5.1.1.2. A-1 Results: fairness measures

Accuracy rate for different cases is shown in Figure 5-4. 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.

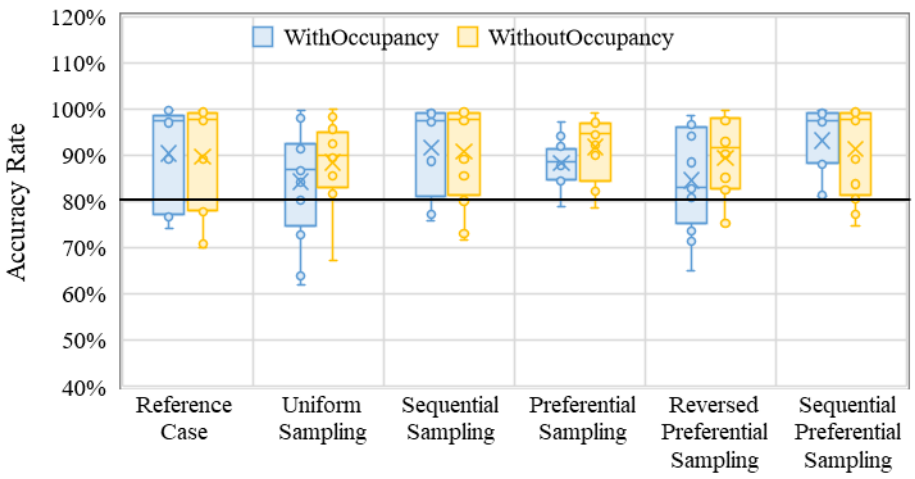
The recall rate of different cases is summarized in Figure 5-5. Reference cases using 'WithoutOccupancy' inputs have a higher recall rate (over 80% for most lighting status series) than reference cases with 'WithOccupancy' inputs. Besides, pre-processing strategies for improving recall rate should be 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 5-6 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.

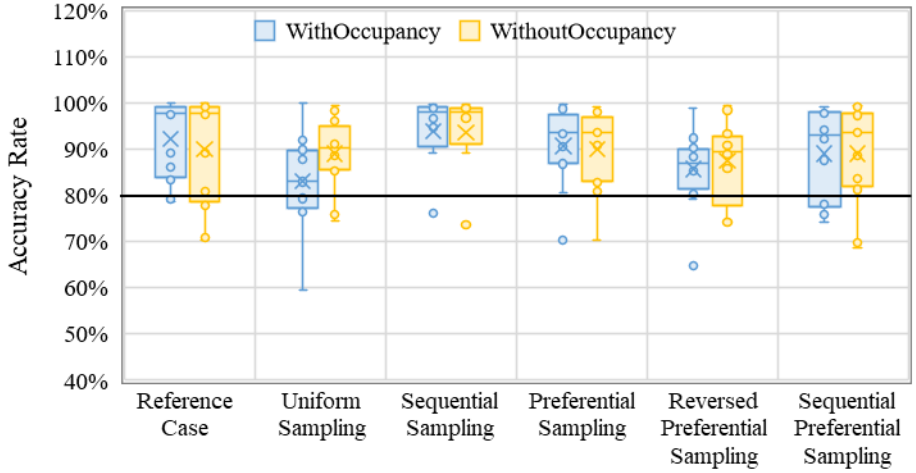
Therefore, the fairness improvement ability of sampling strategies varies among different features, classifiers, and fairness measures. In general, sequential sampling could be a useful strategy for increasing accuracy rate and recall rate while maintaining an acceptable specificity rate.



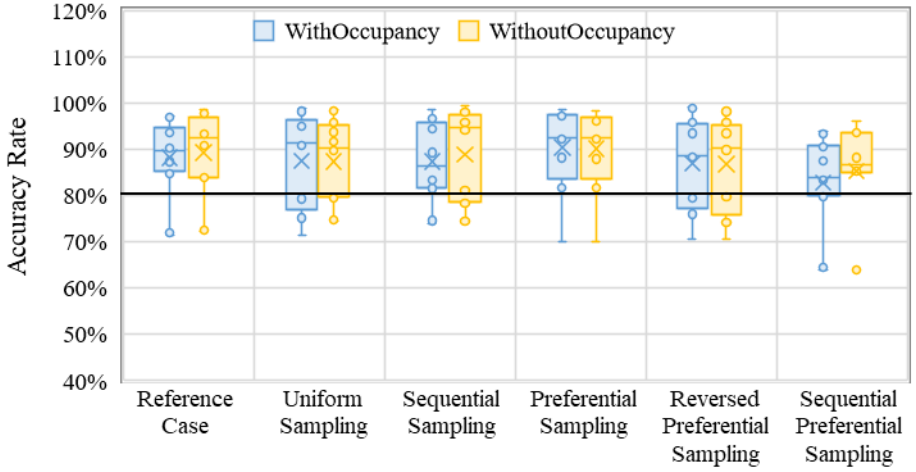
(a) SVM



(b) ANN

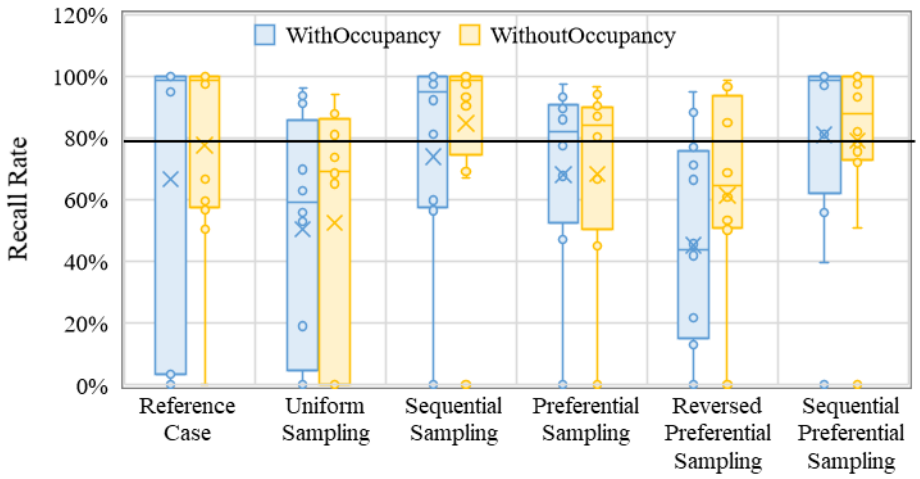


(c) Logistic Regression

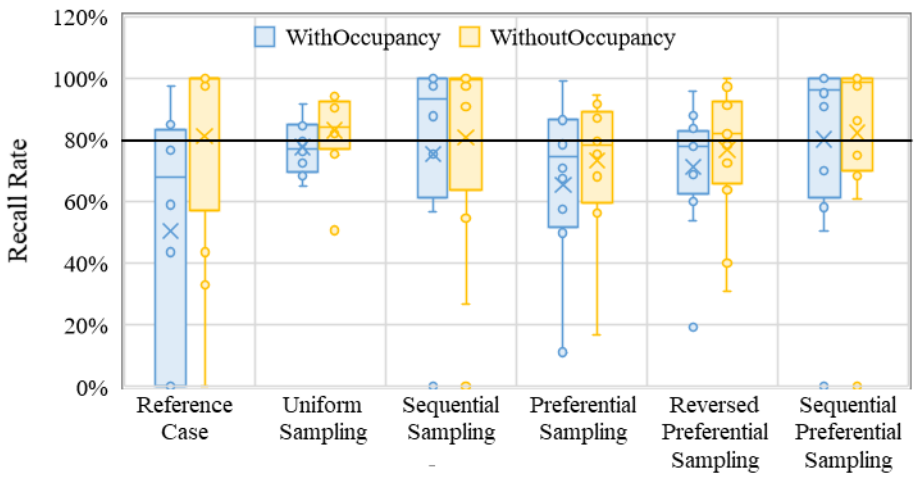


(d) Naïve Bayes

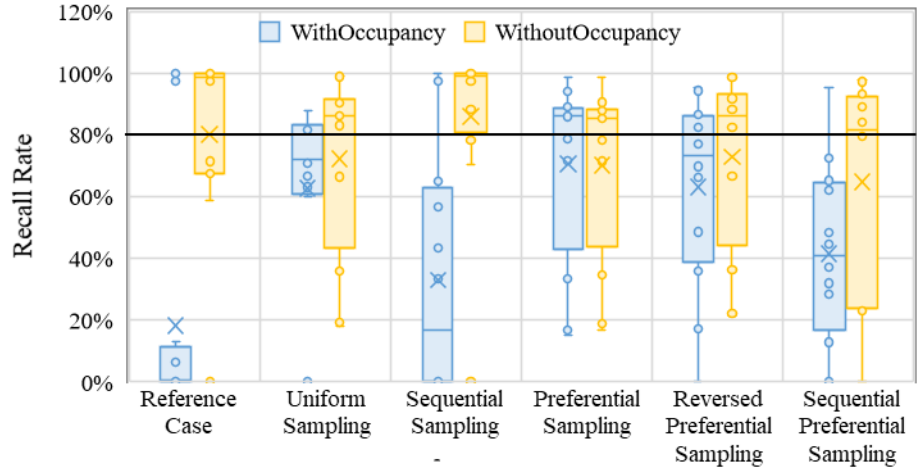
Figure 5-4: Accuracy rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



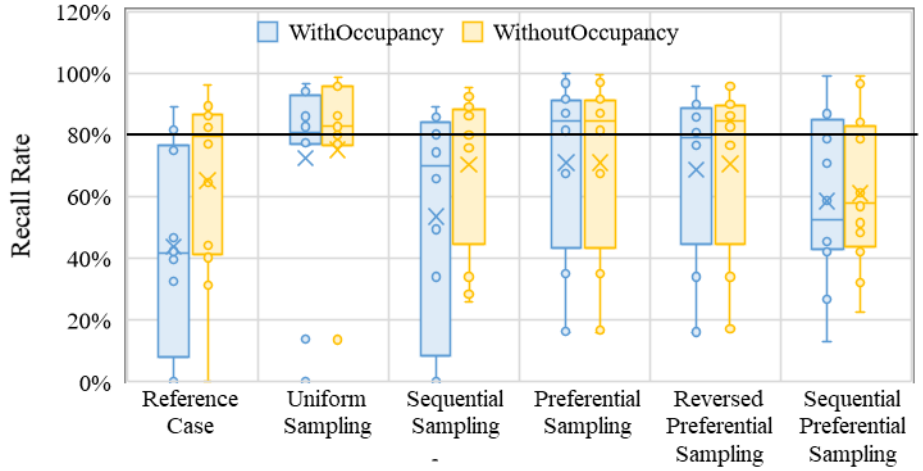
(a) SVM



(b) ANN

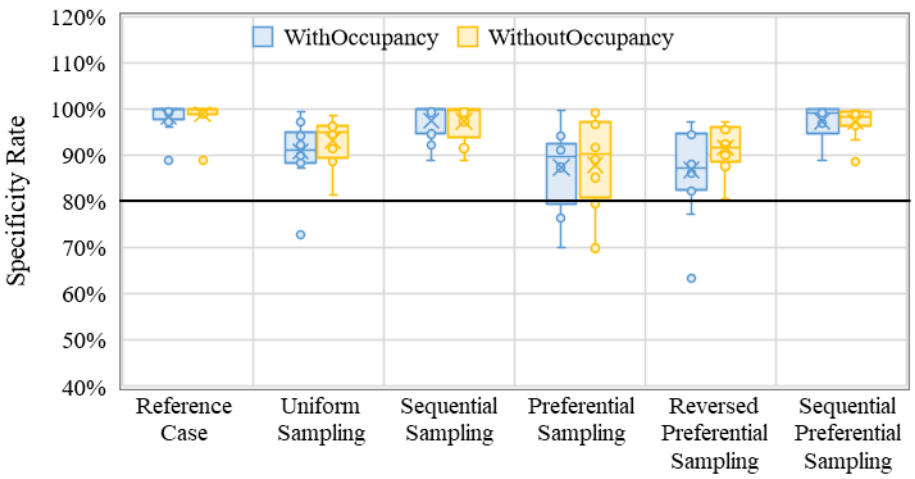


(c) Logistic Regression

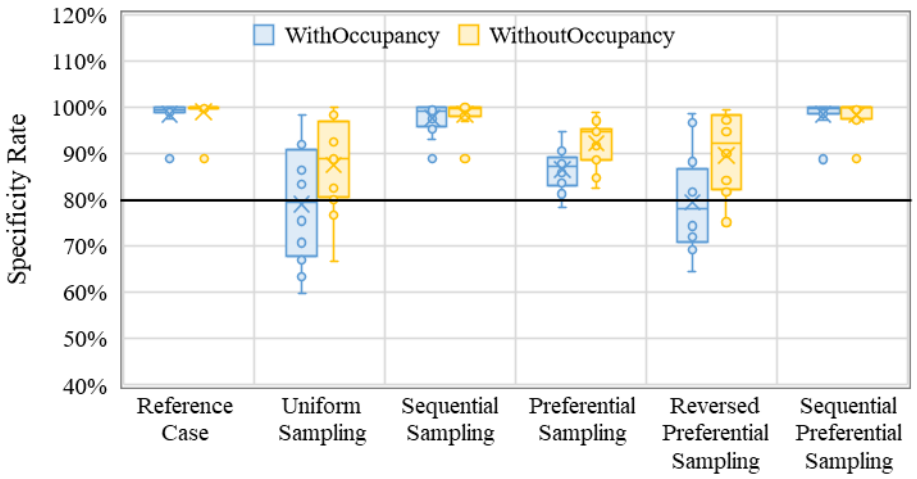


(d) Naïve Bayes

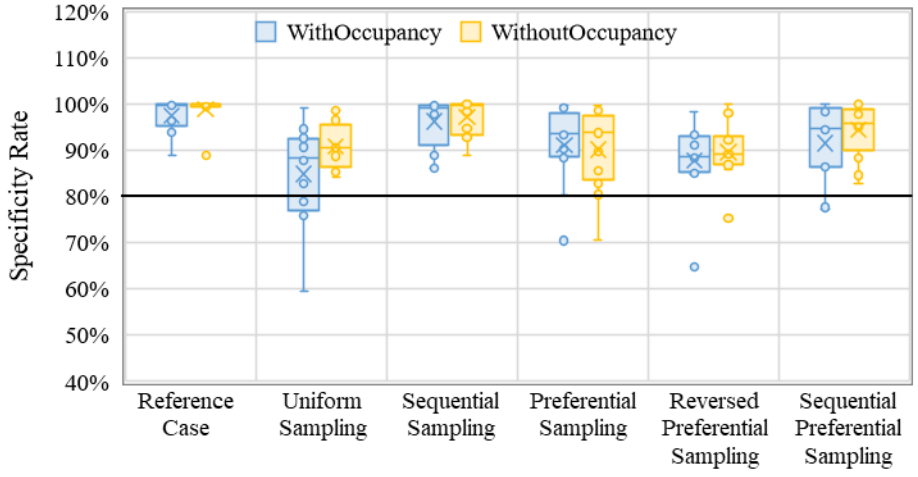
Figure 5-5: Recall rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes



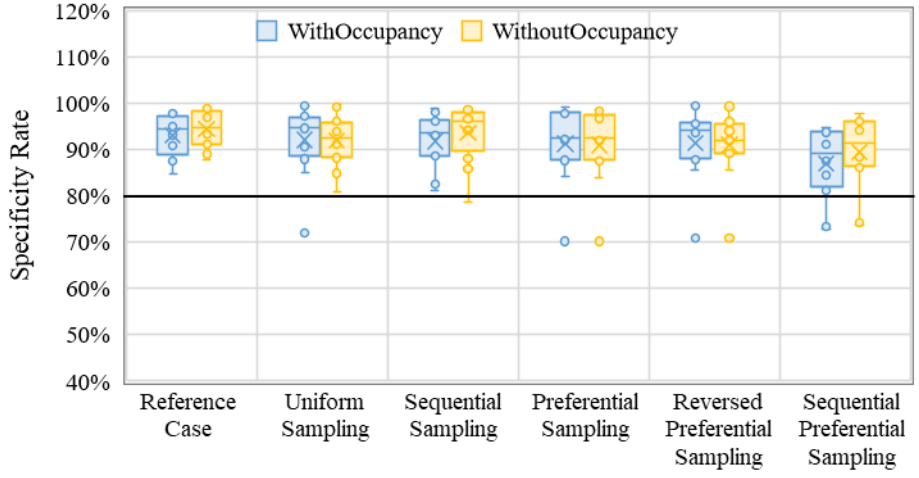
(a) SVM



(b) ANN



(c) Logistic Regression



(d) Naïve Bayes

Figure 5-6: Specificity rate under different cases using (a) SVM, (b) ANN, (c) Logistic Regression, and (d) Naïve Bayes

5.1.1.3. Discussion

To better understand data distribution change in each group as predictive cycle increases, Figure 5-7 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 5-7 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 5-7 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.

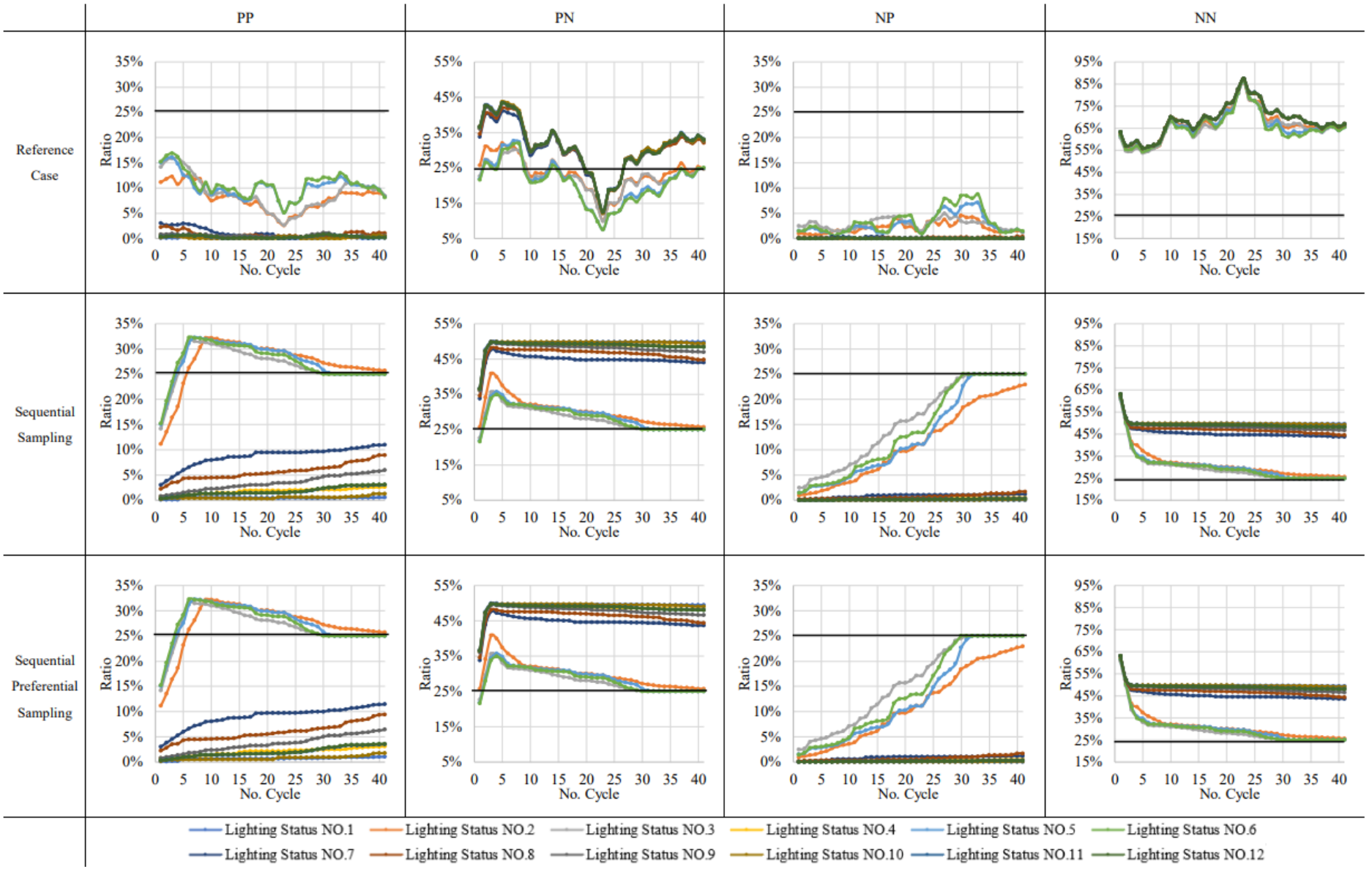


Figure 5-7: Ratios of *PP, PN, NP,* and *NN* among the training dataset under different pre-processing methods

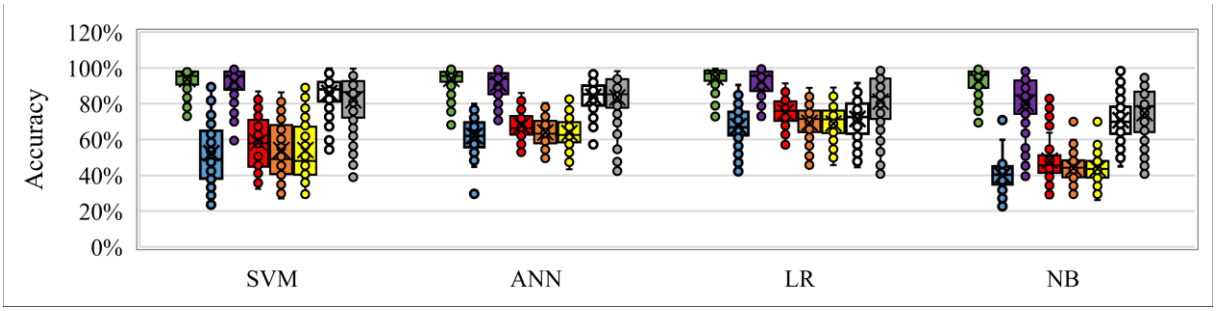
5.1.2. RESULTS FOR CASE STUDY A-2

5.1.2.1. A-2 Results: accuracy measures

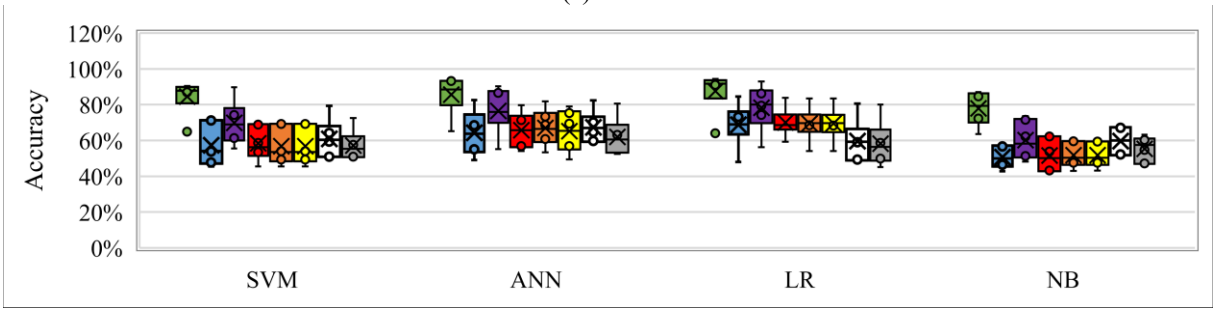
The overall accuracy (y axis) of different classifiers (x axis) trained based on data processed by different pre-processing techniques (legend), for lights under different modes (subfigure title) are summarized in Figure 5-8. It shows that reference cases only present a slight predictive accuracy variation between different classifiers in the same mode. However, when using the same classifier, reference cases under Mode 4 usually present the highest accuracy (higher than 90% mostly), followed by Mode 1 (higher than 85% mostly), Mode 2 (65% - 95%), Mode 3 (55% - 80%), and Mode 5 (40% - 55%). This indicates that even if lights classified in Mode 1 to Mode 4 show the same lighting status pattern (i.e. turned OFF most of the time), their predictive accuracy would be affected by the motion status distribution. The more evenly the motion status is distributed, the higher the lighting status predictive accuracy. Furthermore, the accuracy of all cases in Mode 5 is quite low and not acceptable. It might be because of the irrelevance between inputs features and the output. However, it is still analyzed to show the general effect of pre-processing techniques on the predictive output.

For most cases, the RS shows the most harmful influence on the accuracy, followed by RPS. For instance, in Figure 5-8(a), RS results in the worst accuracy than other pre-processing techniques. RPS decreases the accuracy to be lower than 80% for most lights in Mode 1, Mode 2 and Mode 4, while the accuracy is dropped to be less than 50% in Mode 3. The newly proposed SBS shows slightly better accuracy than RPS, but its reduction effect on the overall accuracy is more significant than SS and SPS. Besides, the effect of ranker on the predictive accuracy is ignorable for RPS, while using LR as the ranker for SPS shows higher accuracy than using NB under Mode 2 and lower accuracy than NB under Mode 3.

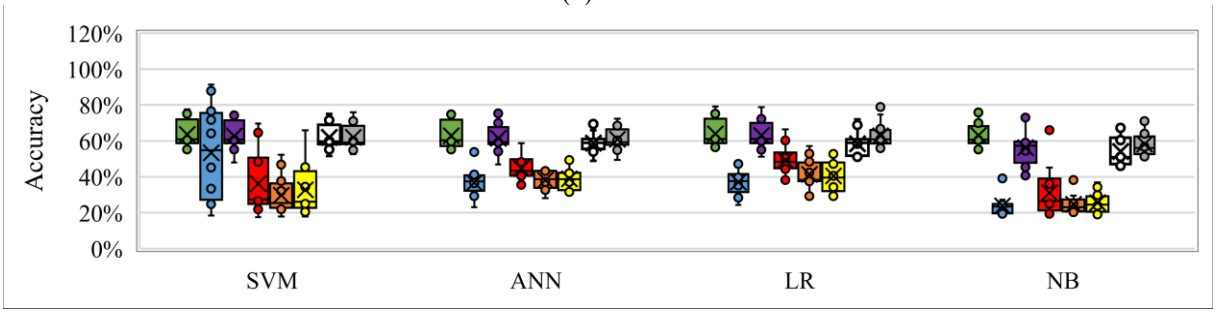
In Mode 5, RS significantly decreases the overall accuracy for Light_4 of APT #11 from ~40% to ~ 20% when using ANN and LR and ~10% when using SVM and NB. RPS decreases the accuracy for Light_4 of APT #11 to be lower than ~20%, while SBS could maintain the accuracy at ~25%. On the other hand, the effect of RS, RPS, and SBS on Light_3 of APT #5 and Light_6 of APT #12 is slight and usually depends on classifiers. Furthermore, SS and SPS usually slightly decrease the overall predictive accuracy for Mode 5.



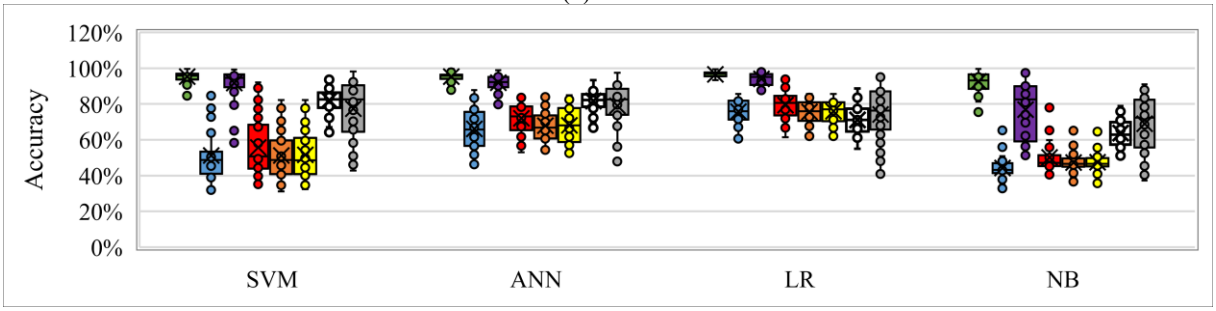
(a) Mode 1



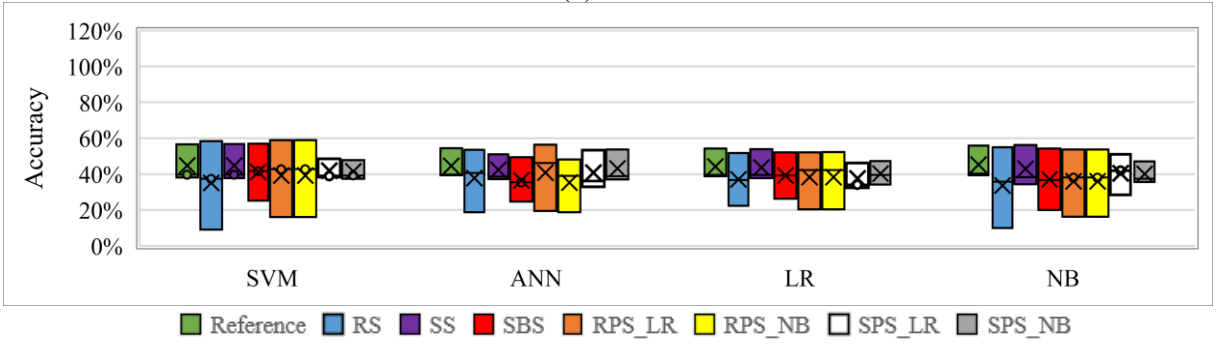
(b) Mode 2



(c) Mode 3



(d) Mode 4

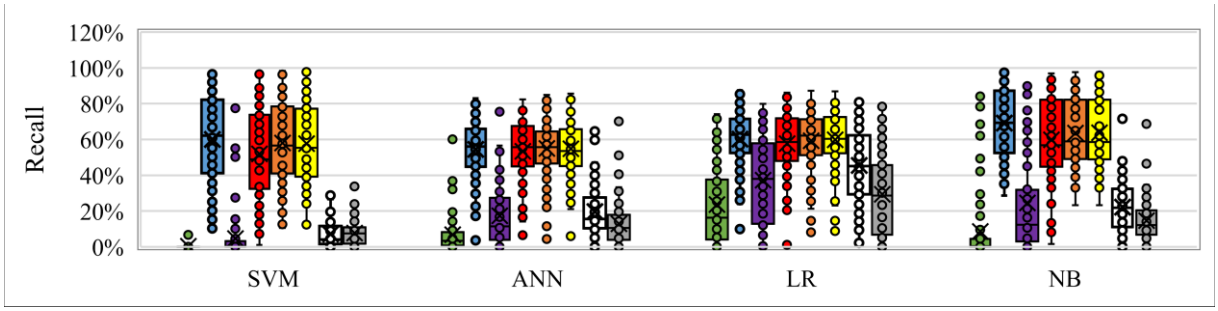


(e) Mode 5

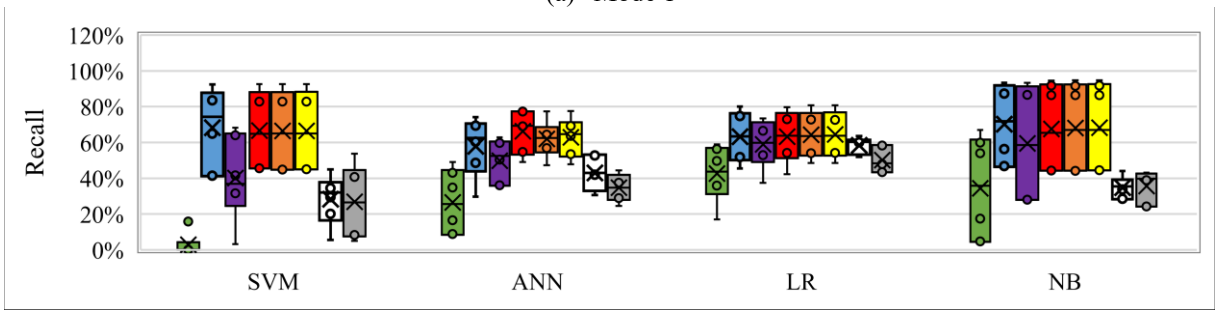
Figure 5-8: Accuracy for different modes

Besides, the effect of pre-processing techniques and classifiers on the recall are presented in Figure 5-9. For Mode 1 to Mode 4, the recall is zero for most of reference cases when using SVM. However, the recall of reference cases could be increased to ~5% - ~71% for Mode 1, ~17% - ~56% for Mode 2, ~5% - ~47% for Mode 3, and ~5% - ~86% for Mode 4, by using LR. Moreover, when using the same classifier for lights in Mode 1 to Mode 4, cases using RS, RPS, or SBS show significantly higher recall than cases using SS or SPS. This result is contrary to the overall accuracy. It is because that lights in Mode 1 to Mode 4 are turned OFF most of the time in $X_{candidate}$ and the validation dataset, thus, their overall accuracy is in line with the specificity (see Figure 5-10). When using these pre-processing techniques to process $X_{candidate}$, data with 'ON' lighting status would be increased in $X_{designed}$ and data with 'OFF' lighting status would be decreased. As a result, the recall would be increased, while specificity and accuracy would be decreased. Among these pre-processing techniques, RPS has the most powerful ability to process a balanced dataset while sampling the most representative data for distinguishing 'ON' class label, thus, it presents the highest recall improvement ability. Besides, SBS could get a balanced $X_{designed}$ at the first time of implementation and capture the people's most recent lighting usage habits, thus, it shows comparable recall improvement ability than RPS, especially in Mode 2. In addition, selecting different kinds of ranker would not affect the recall improvement ability of RPS. However, using LR as the ranker in SPS show higher recall than using NB.

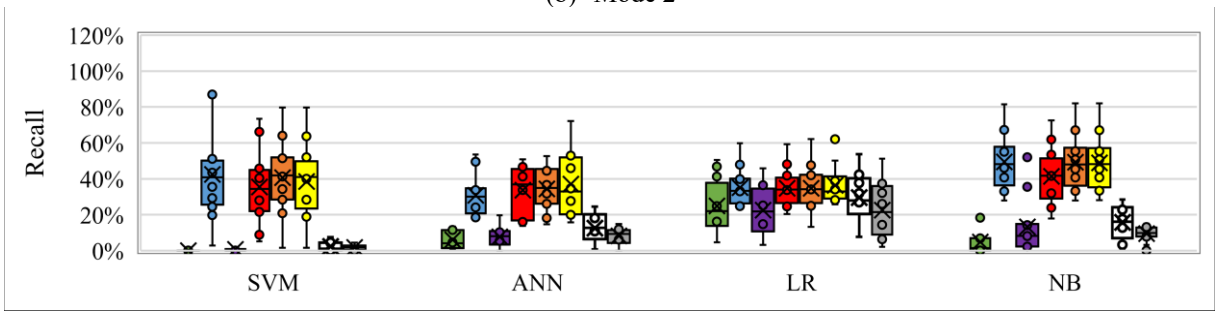
However, in Mode 5, all pre-processing techniques, especially RS, decrease the recall for Light_3 in APT #5 and Light_4 in APT #11, because these two lights are mostly turned ON in $X_{candidate}$ and applying pre-processing techniques would decrease this ratio. By contrast, these pre-processing techniques increase the recall for Light_6 of APT #12. Moreover, SS presents the highest recall for Light_6 of APT #12, when utilizing SVM or NB.



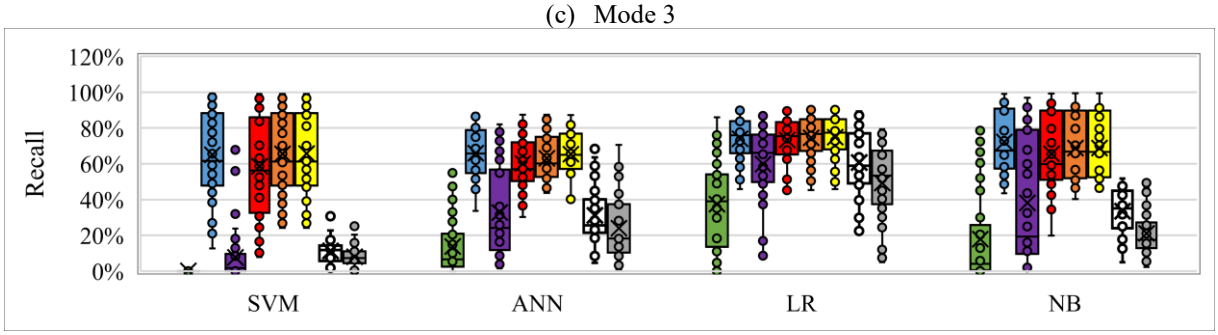
(a) Mode 1



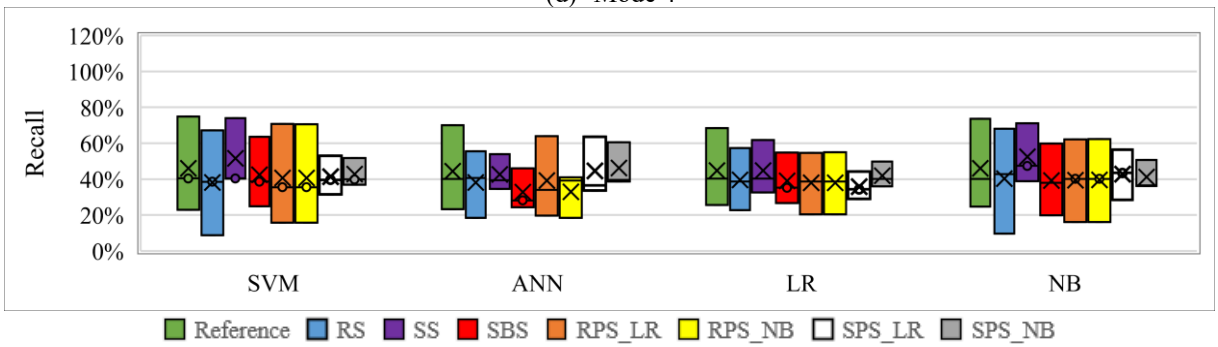
(b) Mode 2



(c) Mode 3



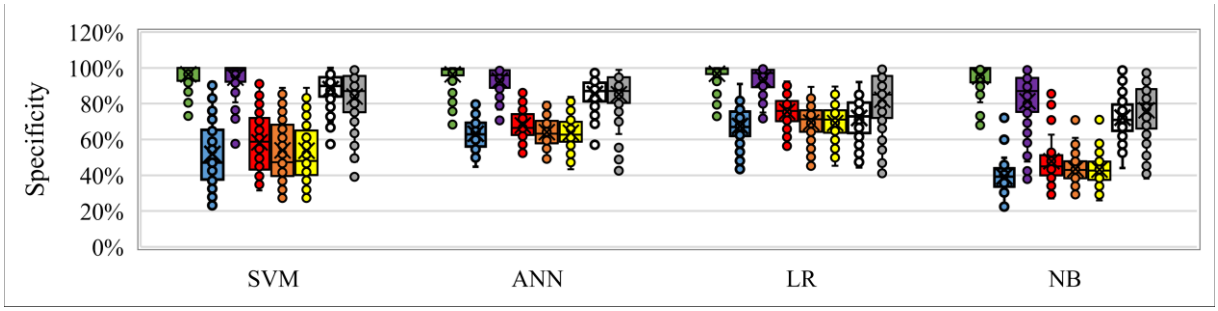
(d) Mode 4



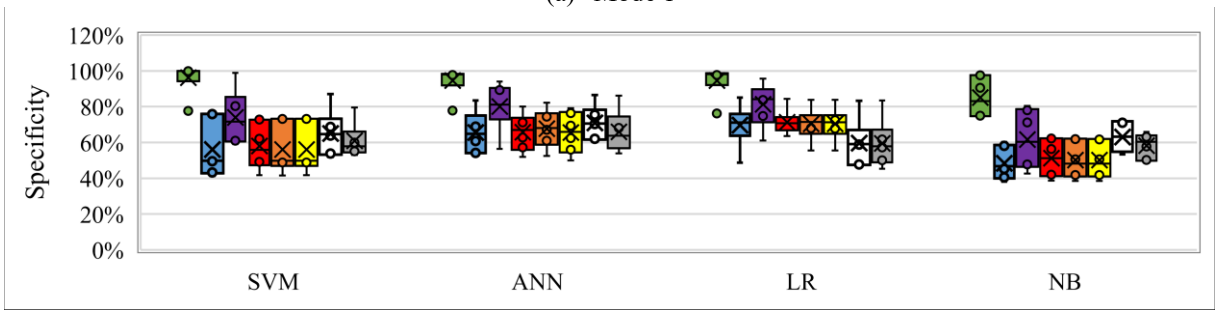
(e) Mode 5

Figure 5-9: Recall for different modes

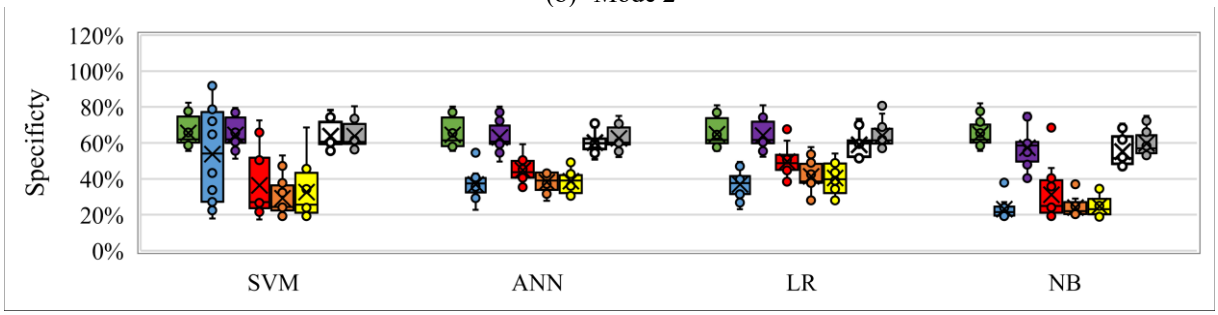
As illustrated in previous paragraphs, for all cases in Mode 1 to Mode 4, the overall accuracy is in line with the specificity. Therefore, detailed description and analysis of specificity for these modes will not be presented. However, the situation is different for Mode 5 (see Figure 5-10(e)). Although SBS and RPS increase the mean specificity, they show a specificity reduction for Light_6 of APT #12, whose status was evenly distributed among ON/OFF classes. Besides, the effect of RS is not consistent for different classifiers. Furthermore, the SS slightly decreases the mean specificity for Mode 5, because of the least specificity increasing ability for Light_3 of APT #5 and Light_4 of APT #11.



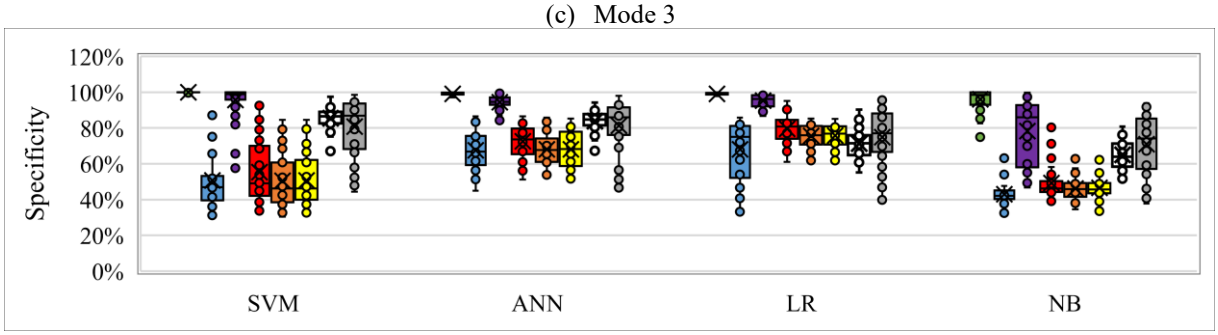
(a) Mode 1



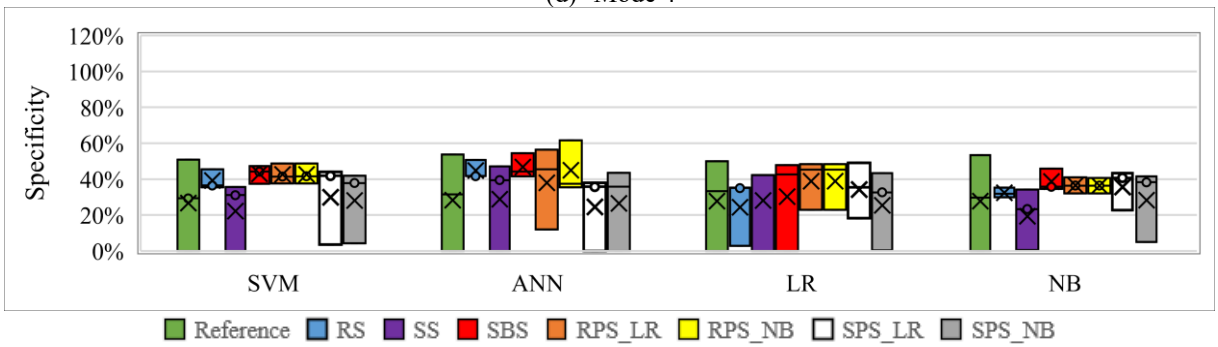
(b) Mode 2



(c) Mode 3



(d) Mode 4



(e) Mode 5

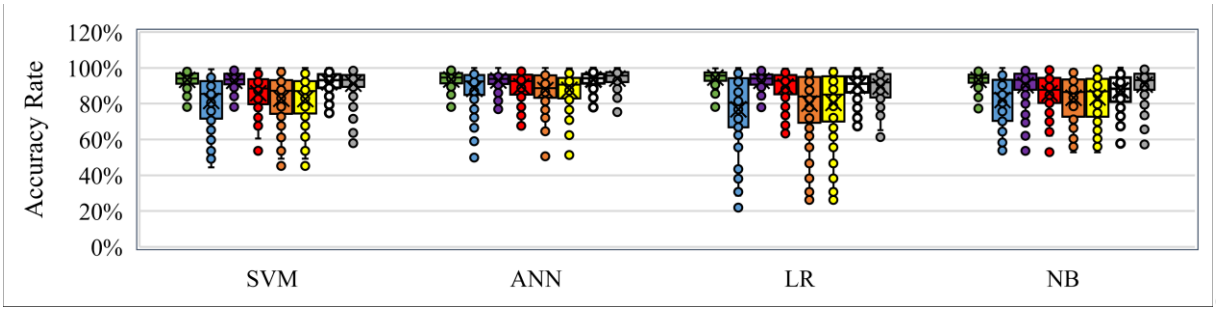
Figure 5-10: Specificity for different modes

5.1.2.2. A-2 Results: fairness measures

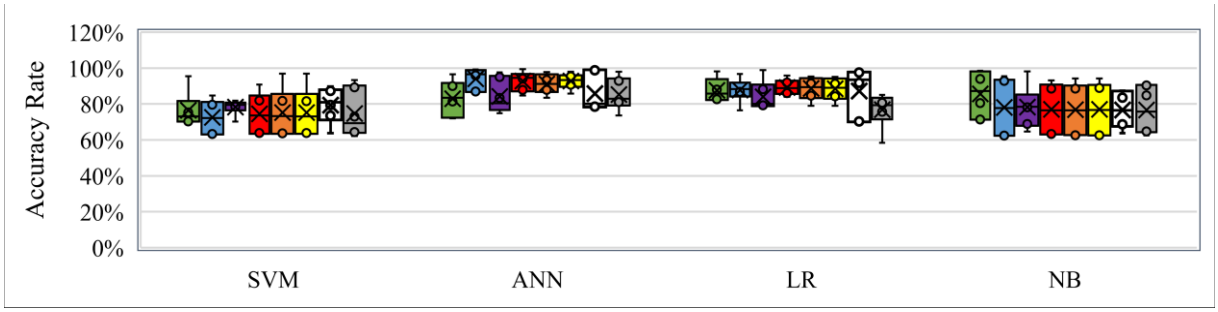
The accuracy rate for different modes is presented in Figure 5-11. Reference cases in Mode 1, Mode 4 and Mode 5 could ensure the accuracy rate to be higher than 80%. RS or RPS often results in the lowest accuracy rate compared to other pre-processing techniques in Mode 1, Mode 3 and Mode 4. For instance, in Figure 5-11(a), RS and RPS decrease the mean accuracy rate from 93% (reference case) to 82% when using SVM, and their lowest accuracy rate even drops to 45%. However, their effect on the accuracy rate of Mode 2 various among classifiers, while RS decreases the accuracy rate of Mode 5 but RPS increases it. More importantly, SBS results in higher accuracy rate for most lighting series than RPS. The effect of SS is comparable with SPS. For all modes, both slightly change the accuracy rate compared to reference cases. The effect of rankers on the accuracy rate is neglectable for both RPS and SPS. Moreover, most cases in Mode 5 present a higher than 80% accuracy rate.

The recall rate (see Figure 5-12) for Mode 1, Mode 3 and Mode 4 is almost zero in most reference cases, while the recall rate of reference cases varies between 0% and 80% for Mode 2. Applying RS, RPS and SBS could effectively improve the recall rate. For example, RPS increases the mean recall rate to be higher than 80% in Mode 1, Mode2 and Mode 4, and higher than 60% in Mode 3. In general, the recall rate improvement ability of SBS is ~7% lower than RPS in Mode 1, Mode 3 and 4 when using SVM, ANN, or LR, while it is almost equal to RPS when using NB or in Mode 2. RS also shows better recall rate than SBS in Mode 1 and Mode 4. In addition, SPS results in better recall rate than SS for Mode 1, Mode 3 and Mode 4. The influence of ranker on the recall rate is not significant for RPS, while using LR as the ranker in SPS usually presents higher recall rate than NB. Furthermore, for Mode 5, the recall rate of most cases is similar to their corresponding accuracy rate. RS presents the lowest mean recall rate in Mode 5.

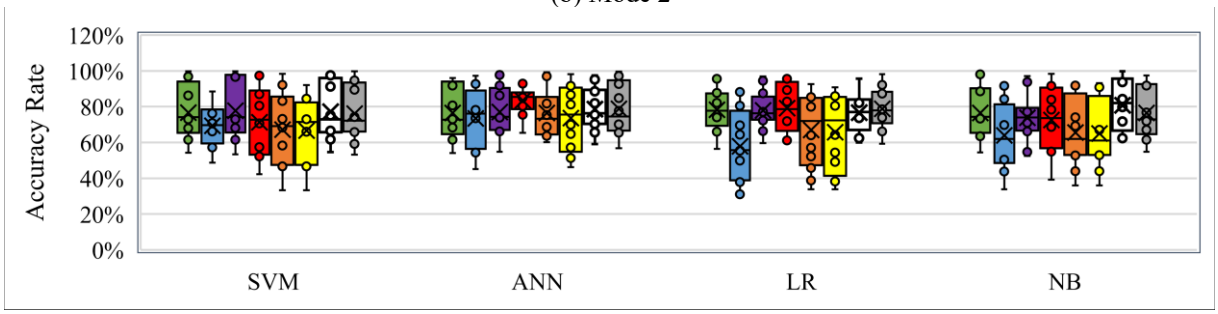
The specificity rate (see Figure 5-13) is similar to the accuracy rate for cases under Mode 1 and Mode 3. However, for Mode 2 and Mode 4, the specificity rate of reference cases is higher than their corresponding accuracy rate. This indicates that the poor recall rate shows negative effect on the accuracy rate. Furthermore, RS show lower specificity rate than SBS and RPS in Mode 1, Mode 3, and Mode 4. Moreover, for Mode 5 (see Figure 5-13(e)), the lowest specificity rate in each box represents the results for Light_4 of APT #11, which is turned ON most of the time. Its 0% specificity rate is caused by the 0% 1-Specificity and 0-Specificity; thus, it does not affect the accuracy rate.



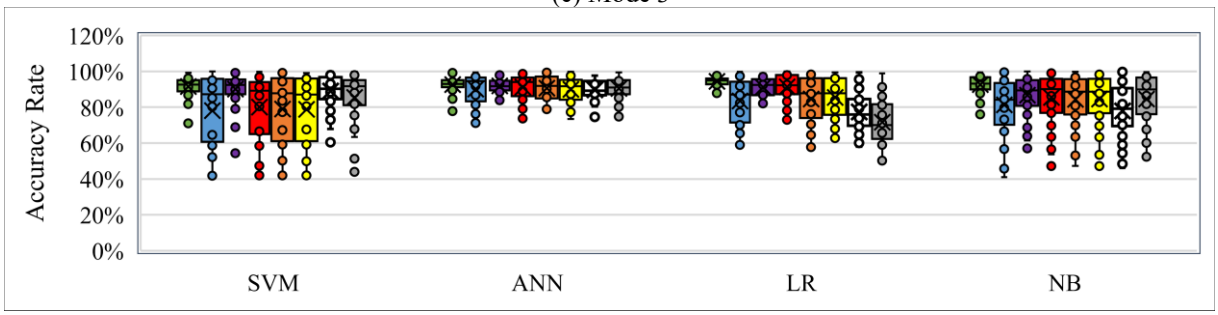
(a) Mode 1



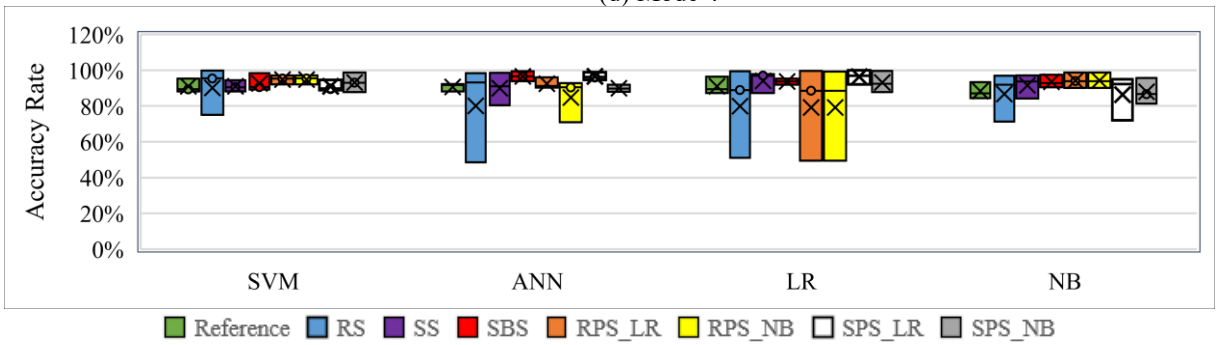
(b) Mode 2



(c) Mode 3

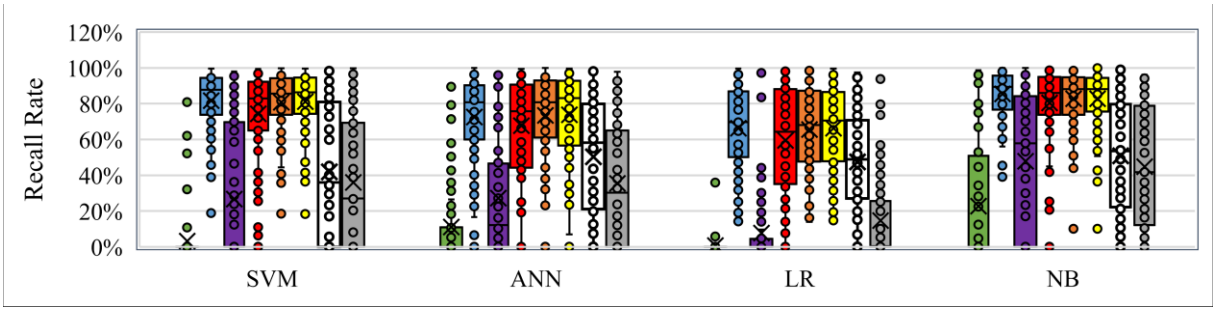


(d) Mode 4

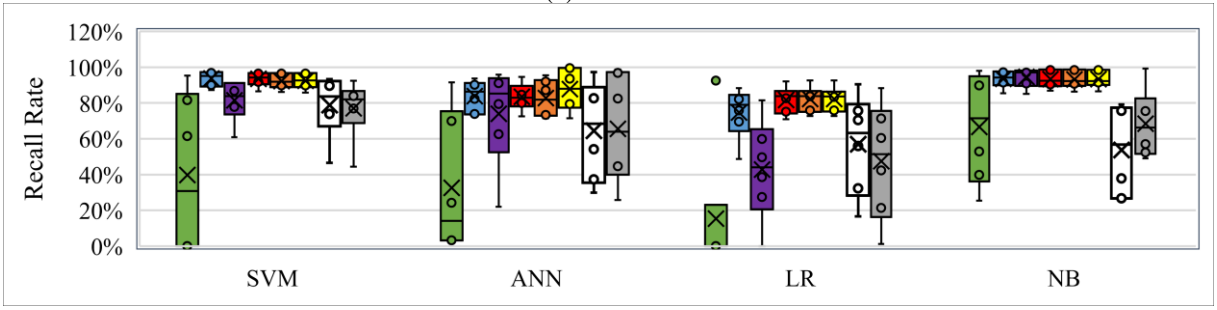


(e) Mode 5

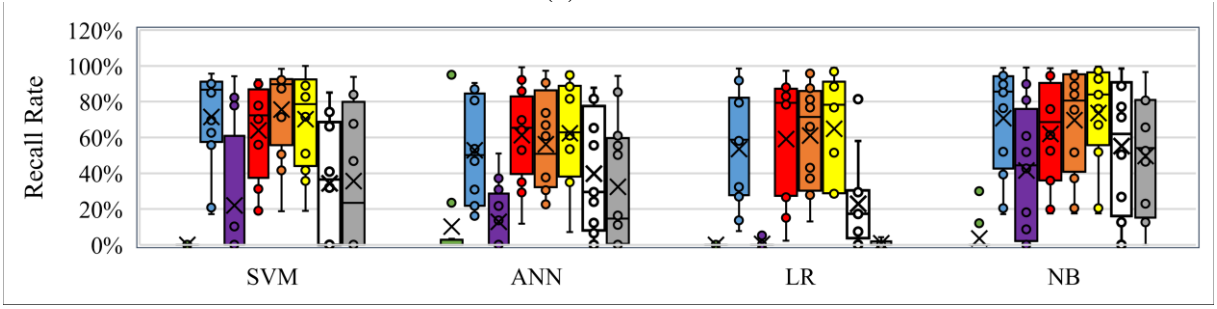
Figure 5-11: Accuracy rate for different modes



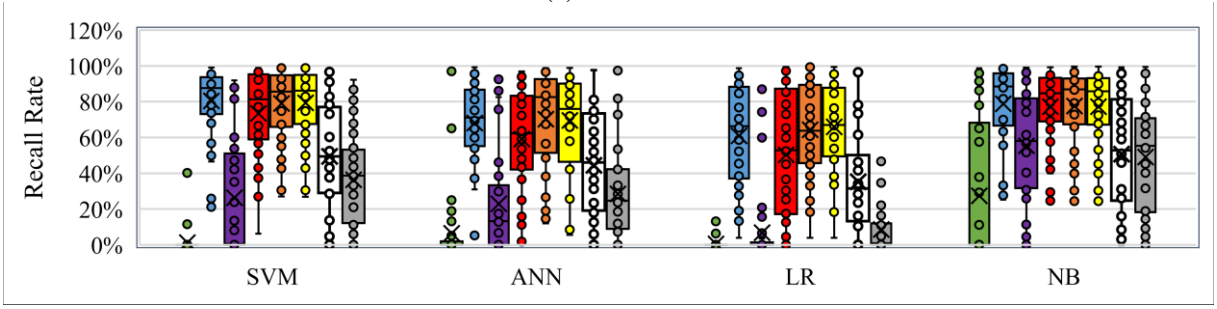
(a) Mode 1



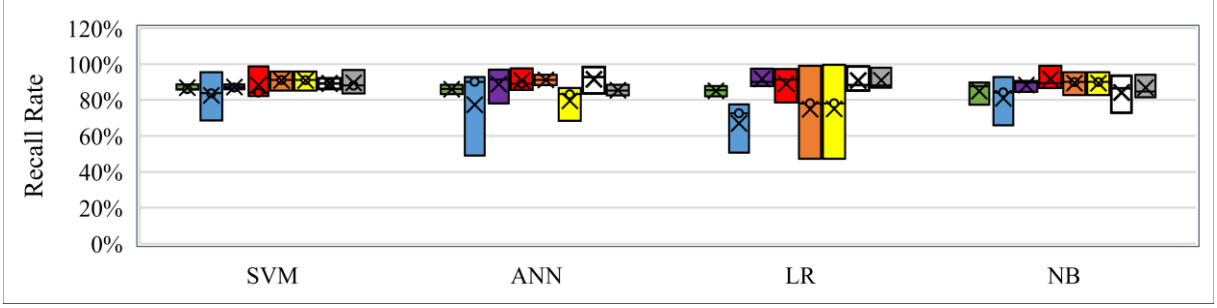
(b) Mode 2



(c) Mode 3

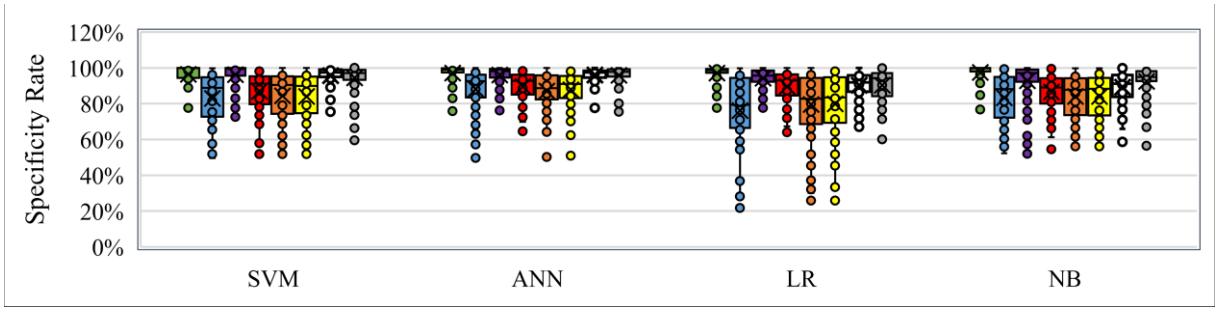


(d) Mode 4

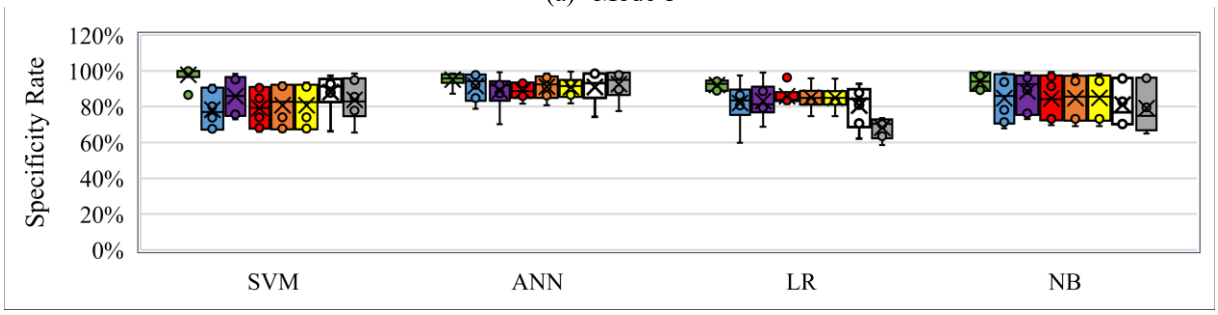


(e) Mode 5

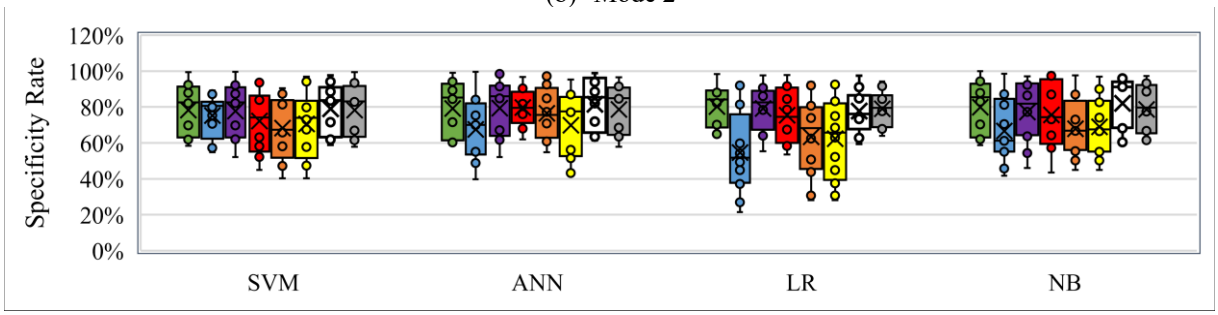
Figure 5-12: Recall rate for different modes



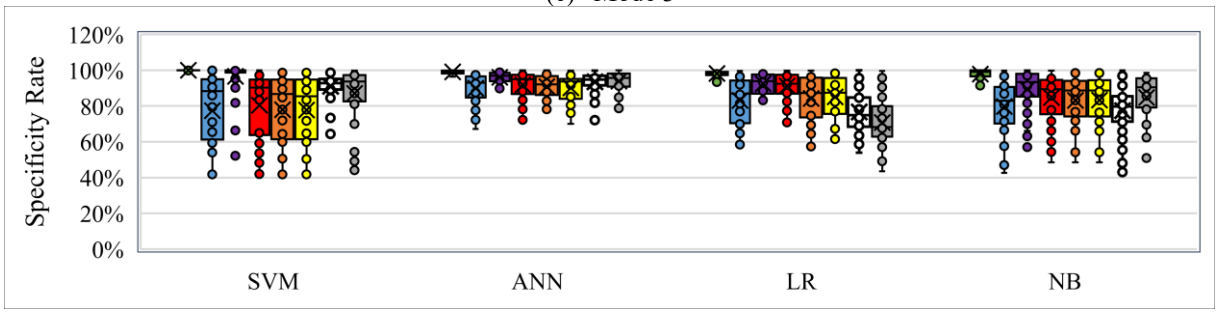
(a) Mode 1



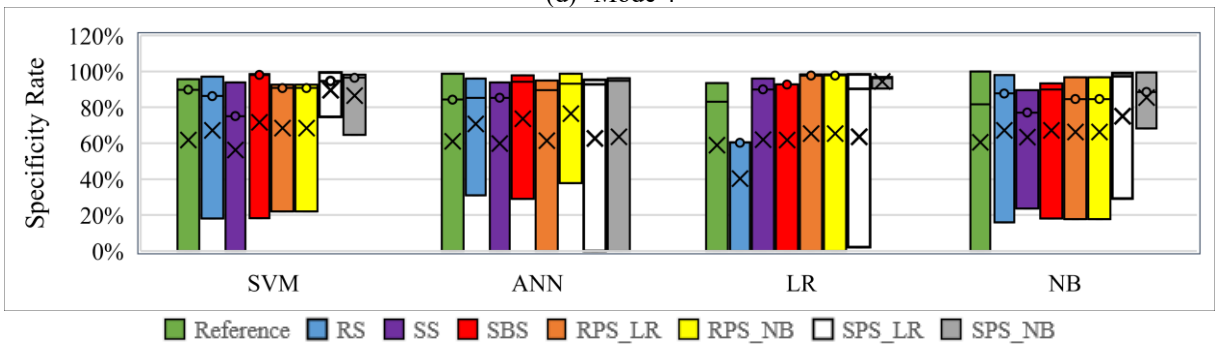
(b) Mode 2



(c) Mode 3



(d) Mode 4



(e) Mode 5

Figure 5-13: Specificity rate for different modes

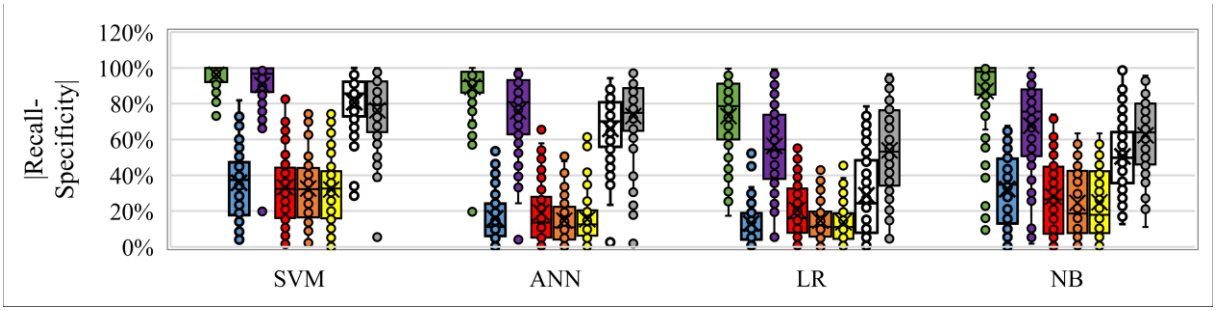
5.1.2.3. Discussion

Through analyzing results presented in Section 5.1.2.1, imbalanced training dataset would result in higher predictive accuracy for majority classes and lower for minority classes. For instance, lights in Mode 1 were turned OFF most of time in $X_{candidate}$, which means that lighting status 'OFF' is the majority class and 'ON' is the minority class. Reference cases for these lights usually present a higher than 90% specificity but almost 0% recall. To increase the predictive accuracy of the minority class, RPS would be the first choice. If a relatively simple algorithm is another requirement, RS and SBS could be good choices as they do not contain a ranker which contributes more hyperparameters to the pre-processing technique. The great predictive performance increasing ability of RS, RPS and SBS for minority class is because they produce a balanced training dataset at the first time of implementation. This finding reveals their potential application to fault detection, which requires higher accuracy for minority conditions.

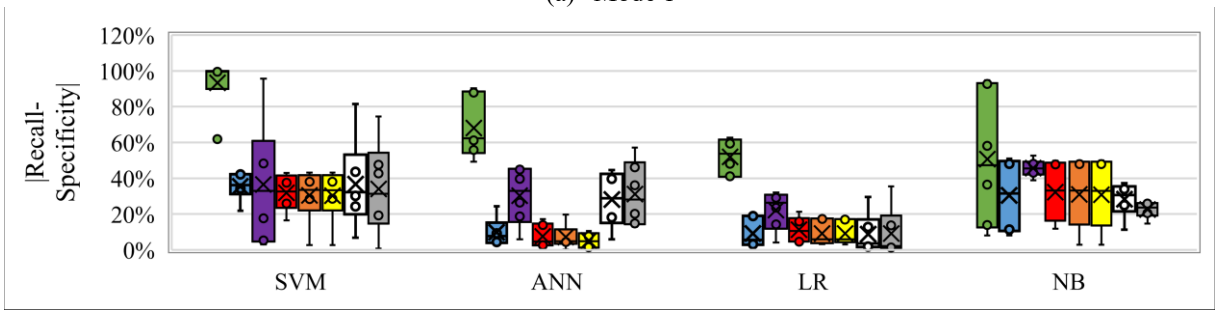
In some cases, SBS even shows comparable result than RPS. This is because occupancy habits, such as lighting usage and occupancy pattern, are seasonally changed [164]. Therefore, in these cases, newly collected data would be more representative than old ones. As a result, $X_{designed}$ of SBS would be similar as of RPS.

One the other hand, when utilizing SS or SPS, predictive accuracy for minority classes maybe increased gradually. This is because the amount of data in minority classes is increased as time goes on (discussed in our previous study [117]).

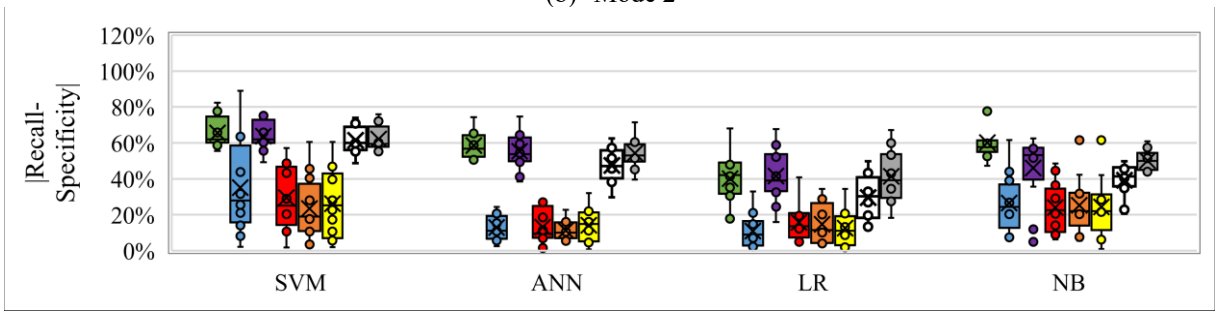
Furthermore, in some building and indoor environment cases, predictive performance for majority classes and minority classes all matters. Therefore, decreasing their difference also need to be considered. Absolute difference between recall and specificity are summarized in Figure 5-14. Reference cases present the highest absolute difference compared to other cases. It further indicates that imbalanced training dataset results in perfect predictive performance for majority classes but poor performance for minority classes. To narrow the absolute difference between recall and specificity, RS, SBS and RPS could be selected. In most cases, SBS shows comparable results with RPS. For instance, when using SVM in Mode 1, these two pre-processing techniques could decrease the mean absolute difference from ~96% (reference cases) to ~32%. However, SS shows the least difference decline. Therefore, it is not suitable for studies that pursue a model with uniform predictive performance for all classes.



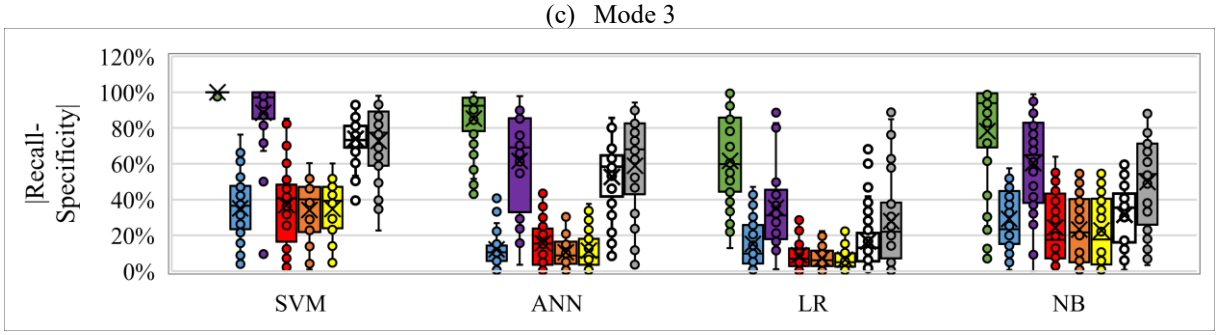
(a) Mode 1



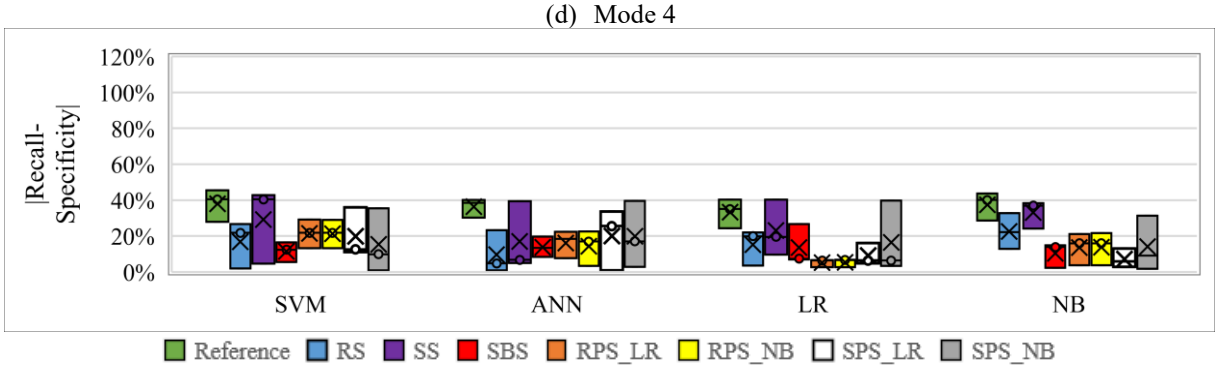
(b) Mode 2



(c) Mode 3



(d) Mode 4



(e) Mode 5

Figure 5-14: Absolute difference between recall and specificity for different modes

One problem in this study is the poor predictive accuracy for lights in Mode 5. It is resulted from the non-representative input features. From Figure 5-15, for Mode 5, all input features are not significantly related to the light status (output), while there are some strong relationships between input features and lights in Mode 1 to Mode 4. Note that in this study, input features are kept as the same for all cases to make the pre-processing techniques or classifiers as the control variables. However, in real-world application, proper features should be selected for specific problems. Another interesting finding from Figure 5-15 is that lighting status could be highly dependent on motion status in some cases, such as in Mode 3. In these cases, suppressing motion status from input features may destroy the predictive performance. This means that excluding protected attribute could not ensure the achievement of *Type I* fairness: The predictive result is independent of the protected attribute. Moreover, detailed correlation matrix values are present in the supplementary information.

Furthermore, this paper reveals that training dataset, in which the protected attribute is distributed imbalance, could results in worse fairness rate, when the protected attribute is related to the output. For instance, in Mode 1 and Mode 4, the 'Motion Status_total' is almost evenly distributed among ON/OFF labels, while it is turned OFF most of time in Mode 3 and 5. Thus, from Figure 5-11, reference cases in Mode 1 and Mode 4 present higher accuracy rate than reference cases in Mode 2 and Mode 3.

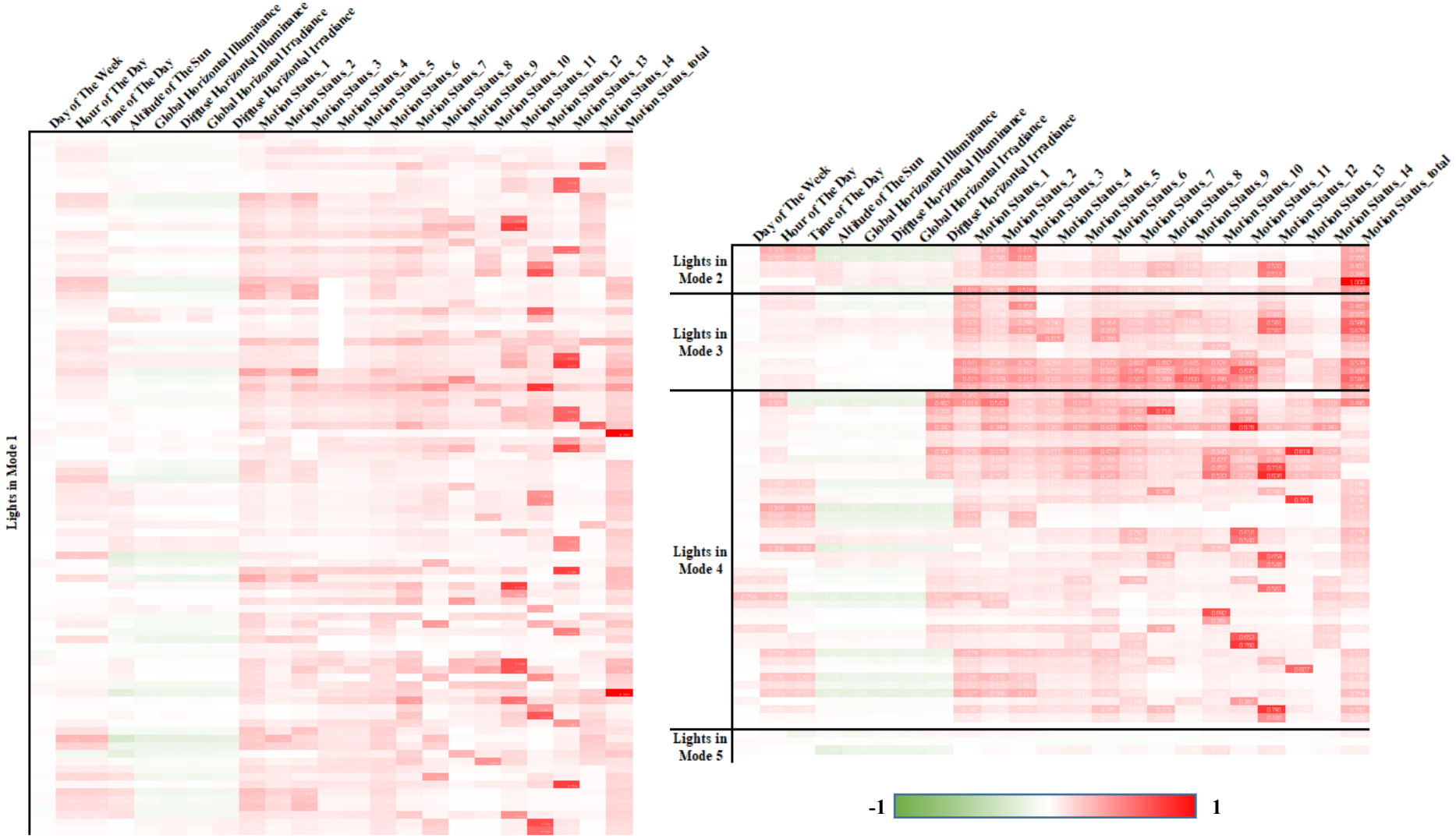
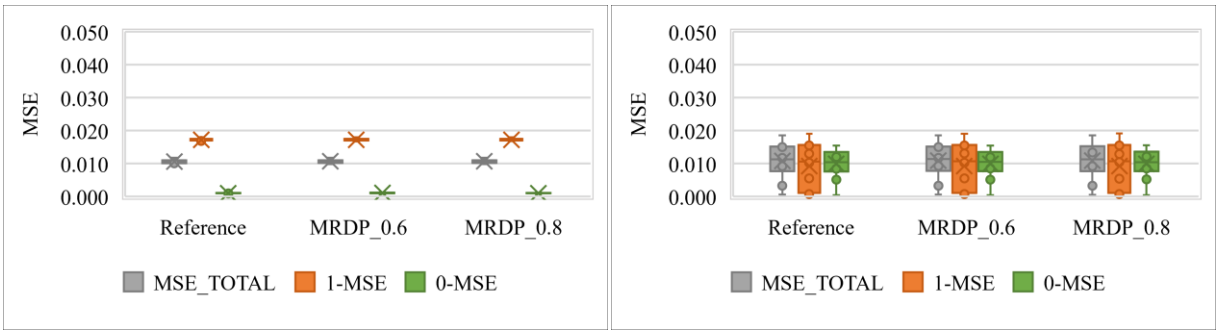


Figure 5-15: Correlation matrix of input features and outputs

5.2. TASK B

5.2.1. RESULTS: ACCURACY IN TERMS OF MSE AND FAIRNESS IN TERMS OF MSE RATE

The effect of MRDP on the predictive MSE, as shown in Figure 5-16, is negligible. As illustrated in section 3.2.1.1, the regularizer added by MRDP tries to make the predictive result fair for conditions with S=Positive and S=Negative in terms of having similar mean residual difference. In other words, the regularizer means to let the developed DDBMs 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 5-17， even in the reference cases, 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 5-17 is that the energy consumption when there are personnel 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 for residential buildings.



(a) (b)

Figure 5-16: Effect of MRDP on the predictive accuracy in terms of MSE during (a) model training and (b) model validation

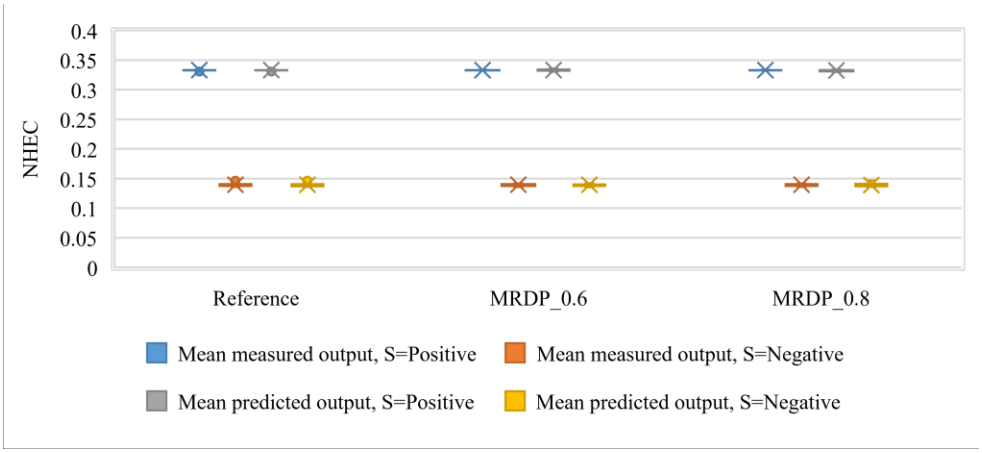
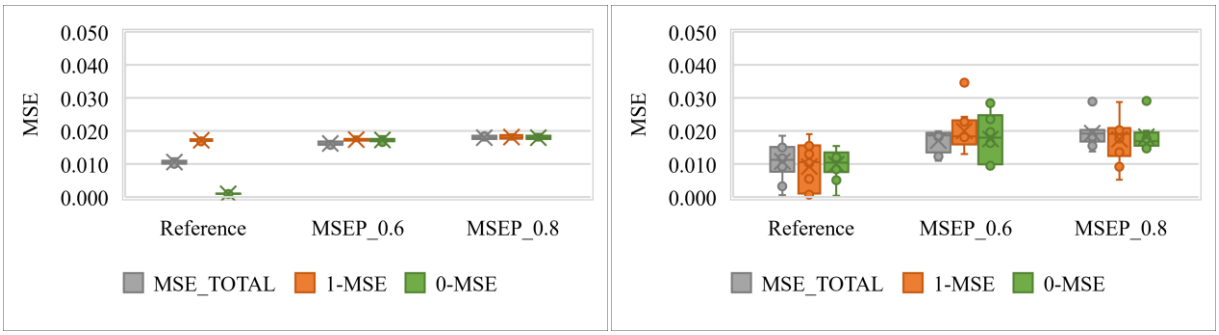


Figure 5-17: Mean measured NHEC and mean predicted NHEC for conditions S=Positive and S=Negative during model training

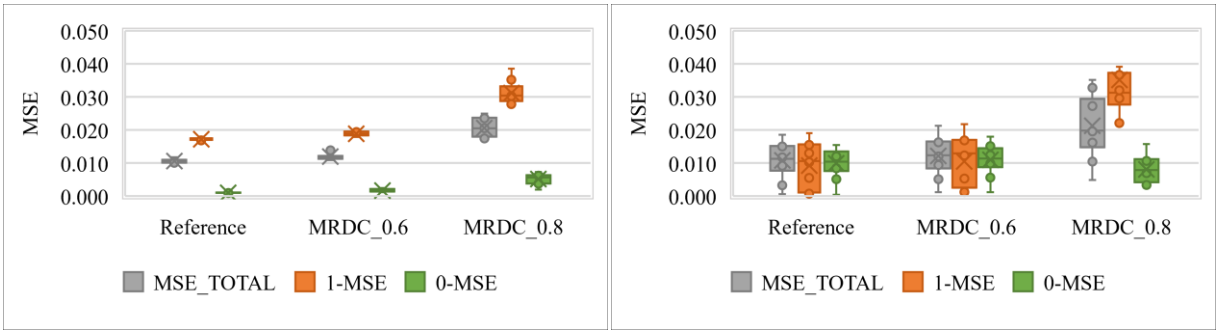
Figure 5-18 shows that, during model training, MSEP with λ=0.6 could effectively make the MSE when S= Positive to be similar to the MSE when S=Negative. However, the predictive accuracy would be decreased as the overall MSE is increased from 0.01 to 0.016. Increasing λ from 0.6 to 0.8 would not significantly contribute to the similarity of MSE between S= Positive and S=Negative. However, the overall predictive accuracy of MSEP with λ=0.8 is slightly worse than MSEP with λ=0.6.



(a) (b)

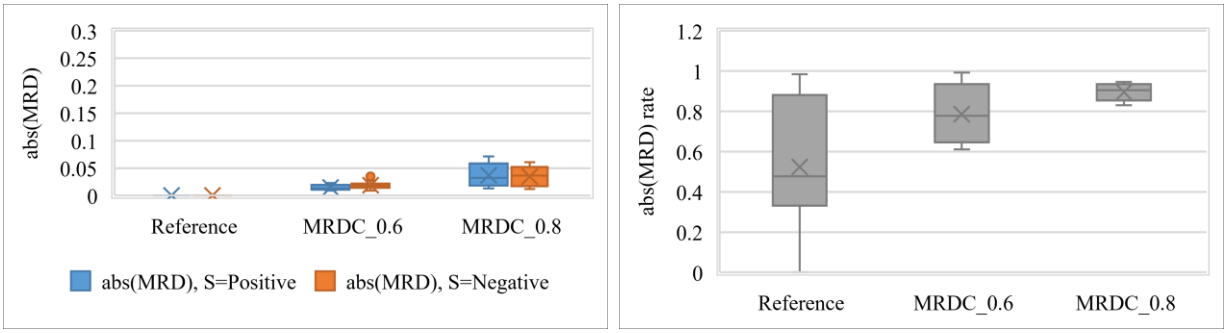
Figure 5-18: Effect of MSEP on the predictive accuracy in terms of MSE during (a) model training and (b) model validation

As shown in Figure 5-19, 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, it 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) similarity during model training (see Figure 5-20 (b)), however, the predictive accuracy would be decreased as the abs(MRE) would be increased (see Figure 5-20 (a)). Moreover, even if increasing p value could increase the abs(MRE) rate, this pattern is not generalizable during model validation (see Figure 5-21). This problem may be caused by the nonconvergence of the optimization algorithm when p=0.8. It could be solved by increasing the maximum iteration number of DE.



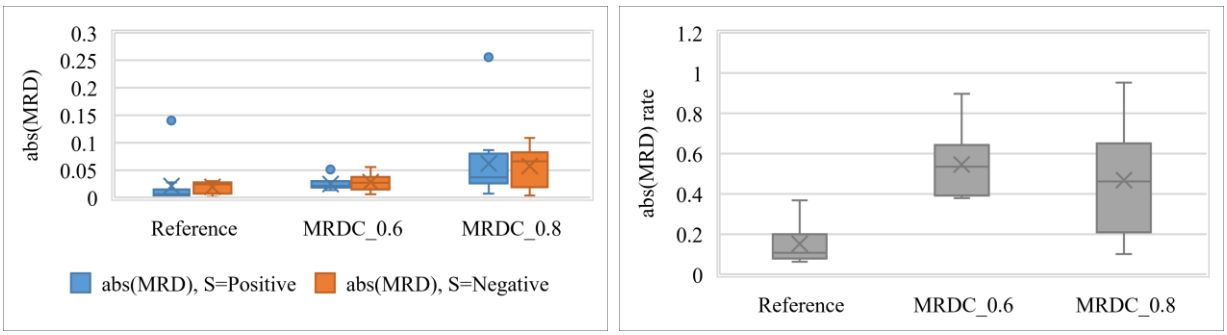
(a) (b)

Figure 5-19: Effect of MRDC on the predictive accuracy in terms of MSE during (a) model training and (b) model validation



(a) (b)

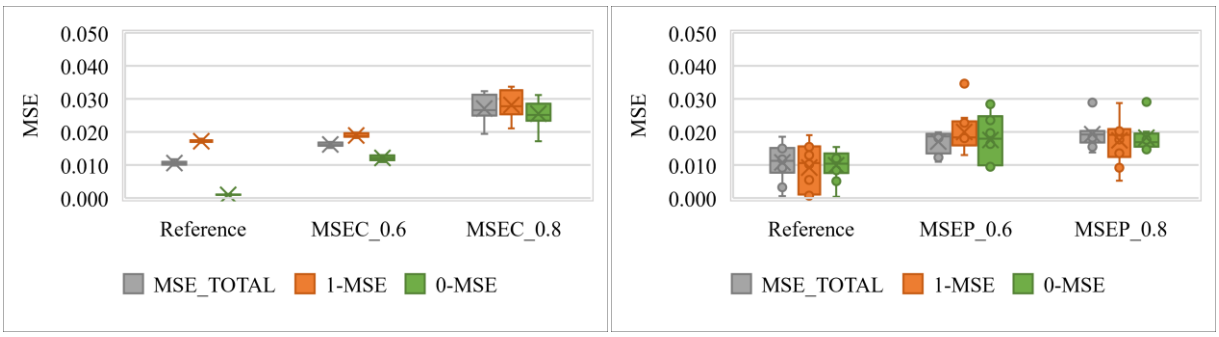
Figure 5-20: Effect of MRDC on the (a) abs(MRE) and (b) abs(MRE) rate during model training



(a) (b)

Figure 5-21: Effect of MRDC on the (a) abs(MRE) and (b) abs(MRE) rate during model validation

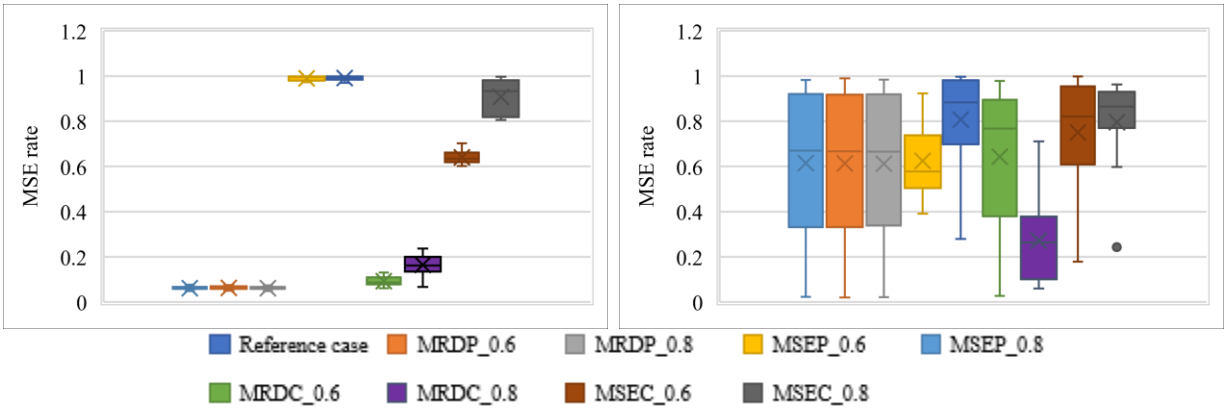
Figure 5-22 shows that increasing p value of MSEC would increase the MSE, however, the difference between 1-MSE and 0-MSE would be decreased. Figure 5-23 shows the effect of MSEC on the Type II fairness improvement: MSEC with p=0.6 could increase the MSE rate to be higher than 0.6 no matter during model training or model validation, while MSEC with p=0.8 could ensure the MSE rate to be higher than 0.8.



(a) (b)

Figure 5-22: Effect of MSEC on the predictive accuracy in terms of MSE during (a) model training and (b) model validation

To improve *Type II* fairness in terms of having a high MSE rate, MSEP and MSEC would be good solutions. However, from Figure 5-23, MSEC shows better generalizability on the validation dataset. As Figure 5-23 shows MRDP does not affect the MSE rate. Moreover, 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.

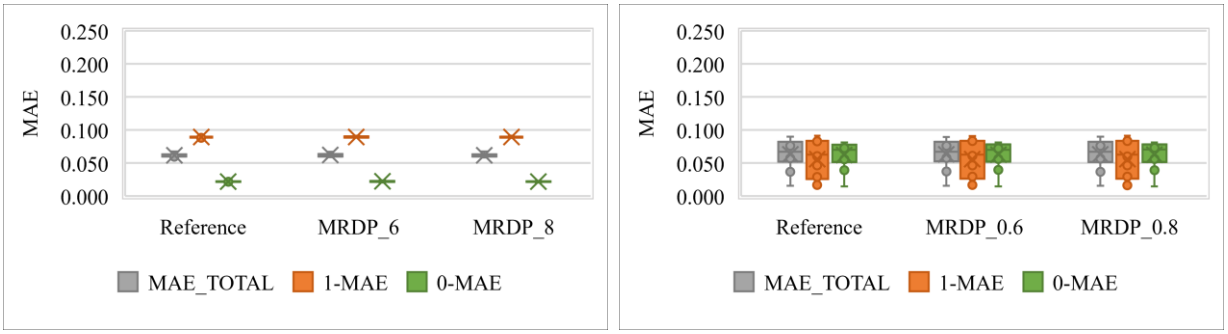


(a) (b)

Figure 5-23: Effect of in-processing methods on the predictive fairness in terms of MSE rate during (a) model training and (b) model validation

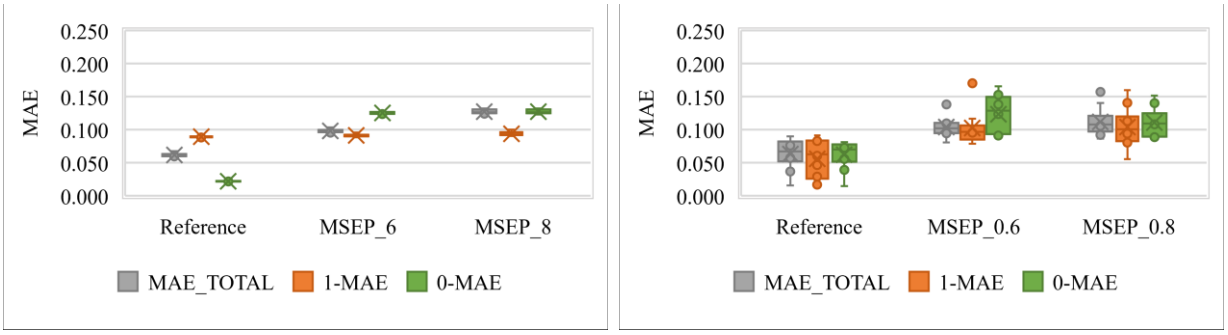
5.2.2. RESULTS: ACCURACY IN TERMS OF MAE AND FAIRNESS IN TERMS OF MAE RATE

As illustrated in Section 5.2.1, 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 5-24. Besides, Figure 5-25 shows that similar to the effect on MSE, MSEP with λ=0.6 could effectively decrease the difference between 1-MAE and 0-MAE. 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) (b)

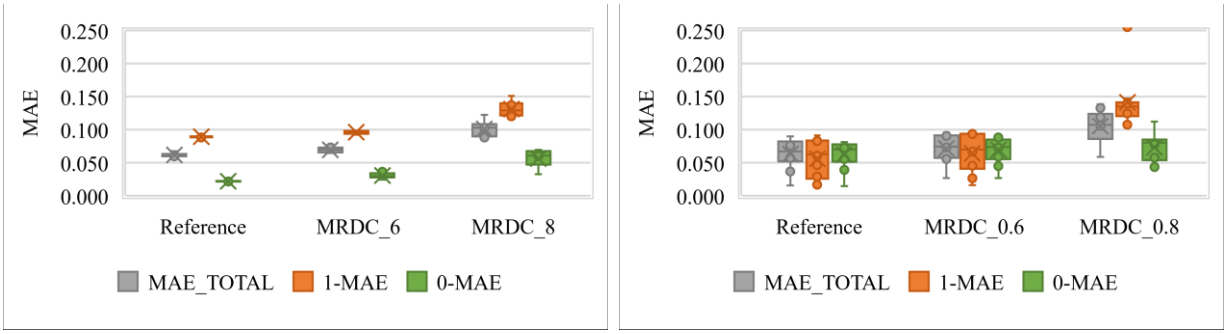
Figure 5-24: Effect of MRDP on the predictive accuracy in terms of MAE during (a) model training and (b) model validation



(a) (b)

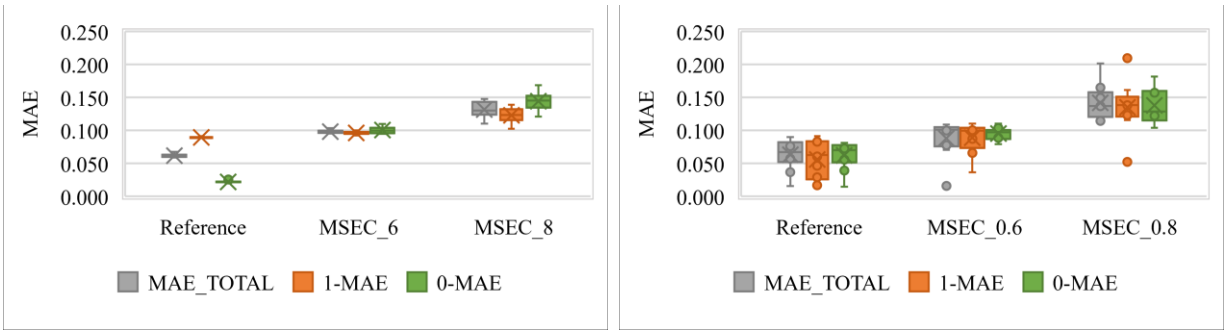
Figure 5-25: Effect of MSEP on the predictive accuracy in terms of MAE during (a) model training and (b) model validation

The effect of MRDC on the predictive MAE is presented in Figure 5-26. It shows that MRDC could not decrease the difference between 1-MAE and 0-MAE, 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. Increase the p value would not contribute more to the improve the similarity between 1-MAE and 0-MAE, but would show further harm to the overall predictive accuracy in terms of MAE.



(a) (b)

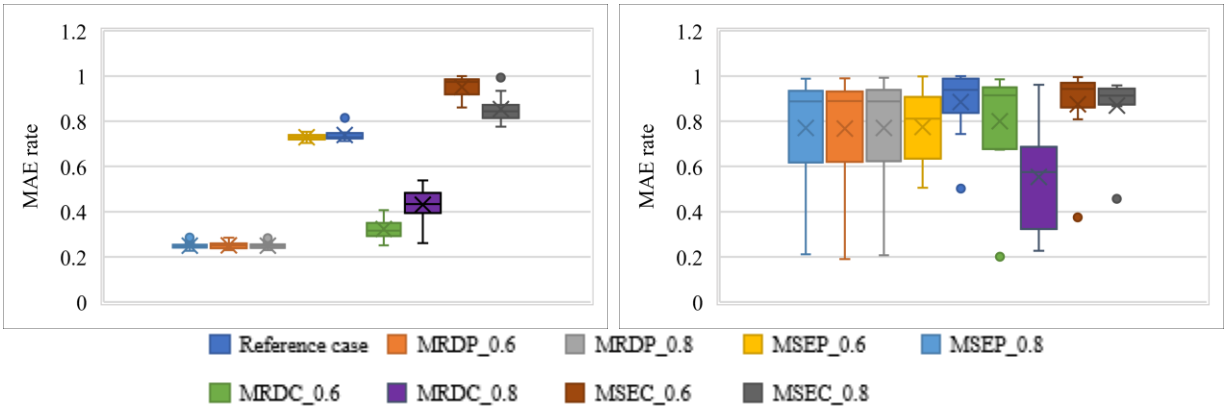
Figure 5-26: Effect of MRDC on the predictive accuracy in terms of MAE during (a) model training and (b) model validation



(a) (b)

Figure 5-27: Effect of MSEC on the predictive accuracy in terms of MAE during (a) model training and (b) model validation

The fairness improvement ability in terms of MAE rate is compared between these in-processing methods and is presented in Figure 5-28. 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 contrast pattern with MSE rate: increasing p value from 0.6 to 0.8 would not further improve the MAE rate.



(a) (b)

Figure 5-28: Effect of in-processing methods on the predictive fairness in terms of MAE rate during (a) model training and (b) model validation

5.2.3. DISCUSSION

5.2.3.1. Effect of Loss_ori selection

MSE was selected as *Loss_ori* for in-processing methods in study cases, 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 among which the value of loss function decreases. This reason is proofed through comparing the training time between using MSE or 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 3.2, MSE is more sensitive to outliers but MAE is more robust to outliers, because a same predictive value for an outlier would more significantly increase MSE (because of the square part of the function) than MAE. 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 4-7), 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.

However, when comparing the predicted NHEC and measured NHEC for linear regression models using MSE or MAE as the loss function in Figure 5-29, 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 accurately predict high NHEC. Further predictive performance comparison between MSE and MAE loss functions could be found in Table 5-1. 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.

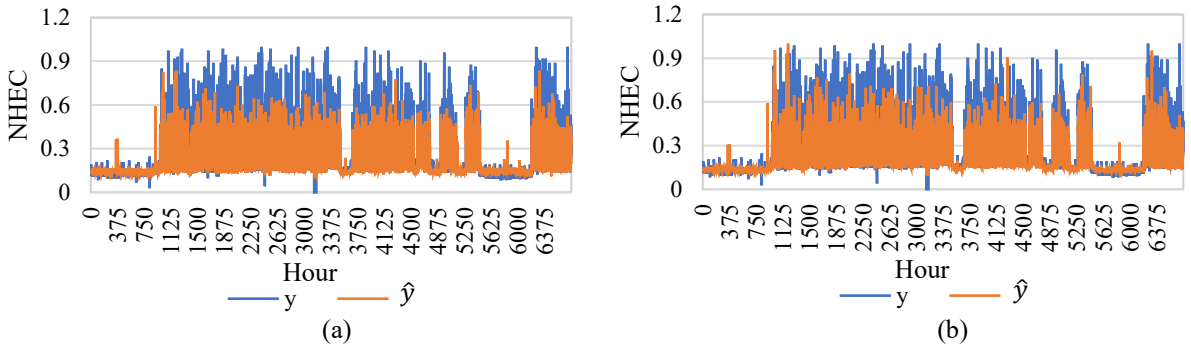


Figure 5-29: Comparison between y and $\hat{y}$ when using (1) MSE or (2) MAE as the loss function

Table 5-1: 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

5.2.3.2. 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 also work well on these constrained optimization problems. For example, genetic algorithm (GA) is a commonly used derivative-free optimization algorithm in the building engineering domain. It has been used to do optimal design [173], optimal control [174,175], and predictive model training [176,177], etc., for buildings. 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.

Genetic algorithm (GA)

GA is a metaheuristic inspired by the process of natural selection that selects the fittest individual to produce the next generation [178]. It could solve both constrained and unconstrained optimization problems, even if their objective function is discontinuous, nondifferentiable, stochastic, or nonlinear [179]. The general procedure of a basic GA is presented in Figure 5-30. 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.

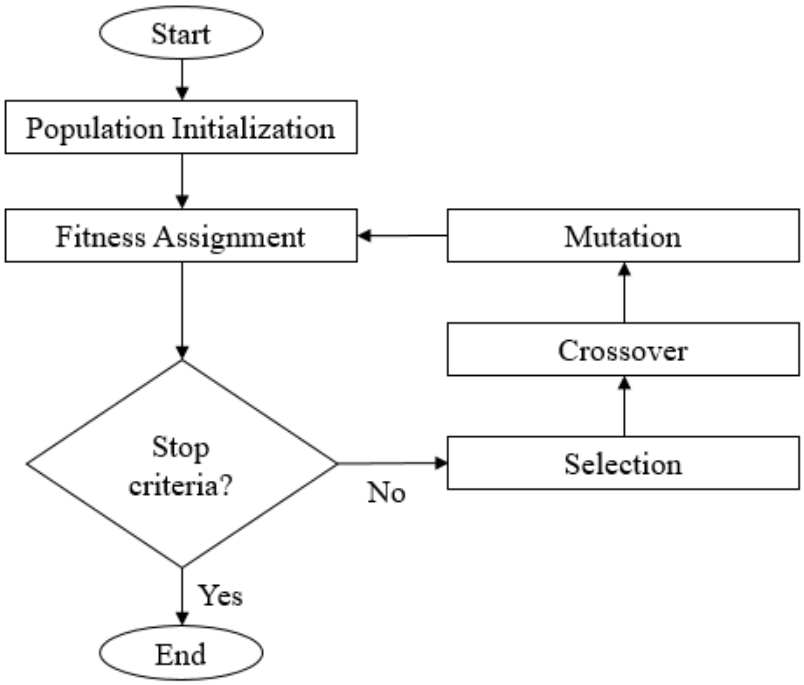


Figure 5-30: General procedure of a basic GA

In this section, DE and GA have the same population size and maximum iteration time. Their predictive accuracy for reference cases is compared in Table 5-2. It shows that DE always show a better accuracy than GE 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 concern.

Table 5-2: 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

5.3. TASK C

5.3.1. PREDICTIVE PERFORMANCE OF DATA-DRIVEN MODELS

5.3.1.1. Predictive result of energy prediction models

Before analyzing the predictive performance, data distribution should be first summarized. $X_{candidate}$ distribution of energy prediction models is shown in Figure 5-31. Most of the time, the energy consumption during a period is between 16.51 kWh to 33.02 kWh. For energy consumption at 0 kWh and 8.26 kWh, the off-peak period shows more samples than the peak period, while for energy consumption at 16.51 kWh to 33.02 kWh, peak periods have more samples. The number of samples is similar between peak periods and off-peak periods when the energy consumption is at 41.28 kWh or 49.53 kWh. Besides, the validation data distribution among energy consumption categories are present in Figure 5-32. Its pattern is similar to $X_{candidate}$. The difference is that energy consumption at 16.51 kWh during the off-peak period has more data than the peak period in the validation dataset.

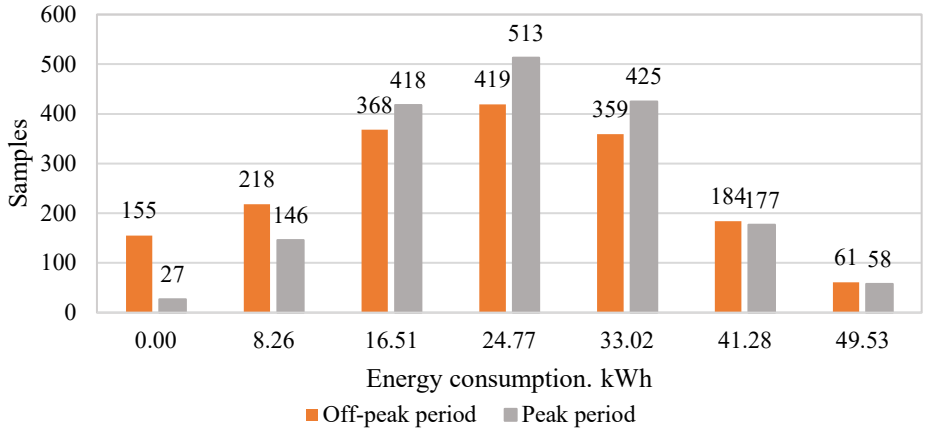


Figure 5-31: $X_{candidate}$ distribution of energy prediction models

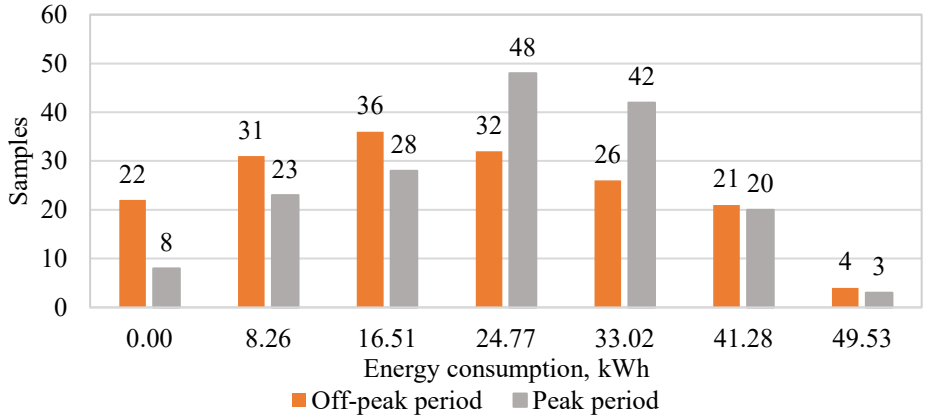


Figure 5-32: Validation data distribution of energy prediction models

The predictive result in terms of residual difference distribution during model training and model validation is presented in Figure 5-33 and Figure 5-34, respectively. Both RPS and RS would decrease the number of accurately predicted samples with 0 residual difference. RPS shows more negative effect on the overall predictive accuracy than RS. This result is inline so got from case study A-1 and A-2. However, RPS could still effectively ensure that the residual difference is within the range [-8.26, 8.26] most of the time. Moreover, during model validation, RPS shows a slight improvement effect on the accuracy during the off-peak period, however, it significantly decreases the predictive accuracy during the peak period.

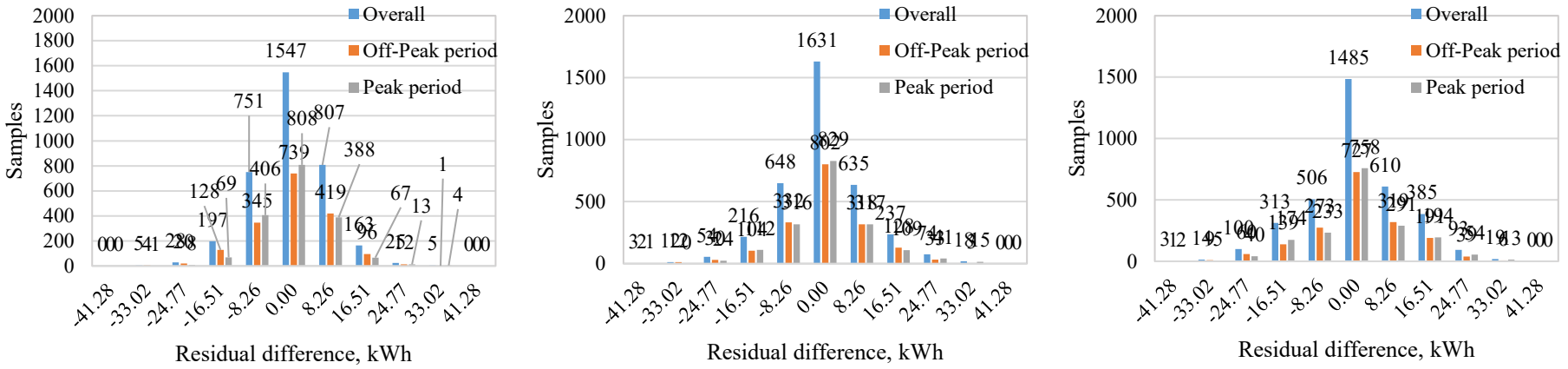
To analyze the effect of pre-processing methods on predictive result of conditions with different data volume, residual difference distribution when the measured energy consumption is 0 kWh (Figure 5-35 for model training and Figure 5-36 for model validation), 24.77 kWh (Figure 5-37 for model training and Figure 5-38 for model validation), and 41.28 kWh (Figure 5-39 for model training and Figure 5-40 for model validation) are presented.

From Figure 5-35, increasing samples in minority conditions could effectively increase the predictive accuracy. In the reference case, energy consumption is overestimated by 16.77 kWh for over half of samples when their measured energy consumption is 0. Using RPS or RS to increase the number of data samples in this condition would increase the number of accurately predicted

samples. RPS presents higher accuracy increasing effect than RS. This pattern could also be proofed during model validation in Figure 5-36.

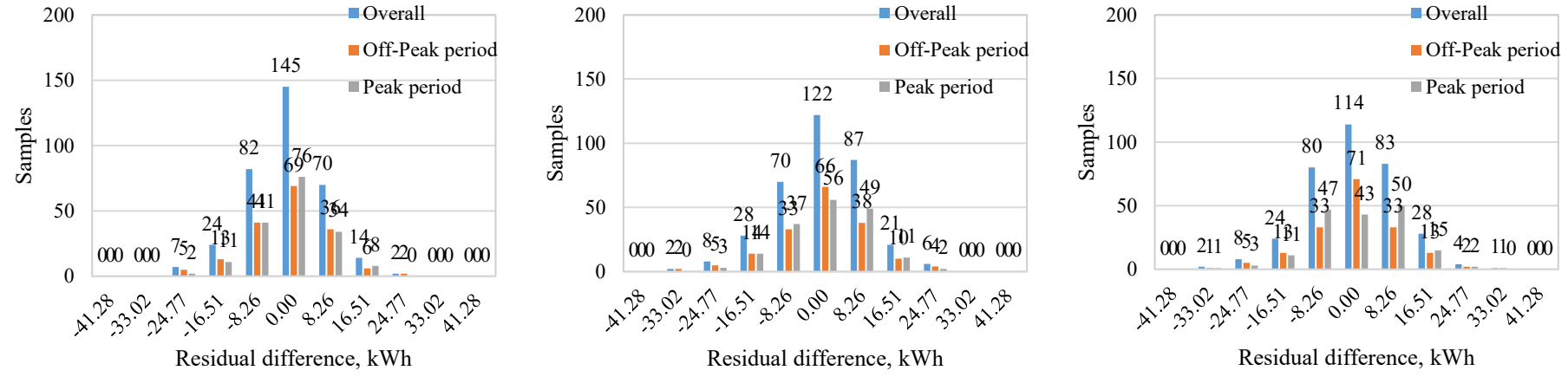
However, RPS and RS would decrease the predictive accuracy of majority conditions. From Figure 5-37, the predictive accuracy during model training would be destroyed when using RPS and RS to undersample data when Y= 24.77 kWh, however, Figure 5-38 shows that RPS and RS still present an acceptable residual difference range between [-8.26, 8.26]. Besides, RPS increases the similarity of accurate prediction between off-peak periods and peak periods.

RPS improves the predictive accuracy when Y= 41.28 kWh no matter during model training (Figure 5-39) or model validation (Figure 5-40). On the other hand, RS increases the number of accurately predicted samples training, but decreases it during model validation.



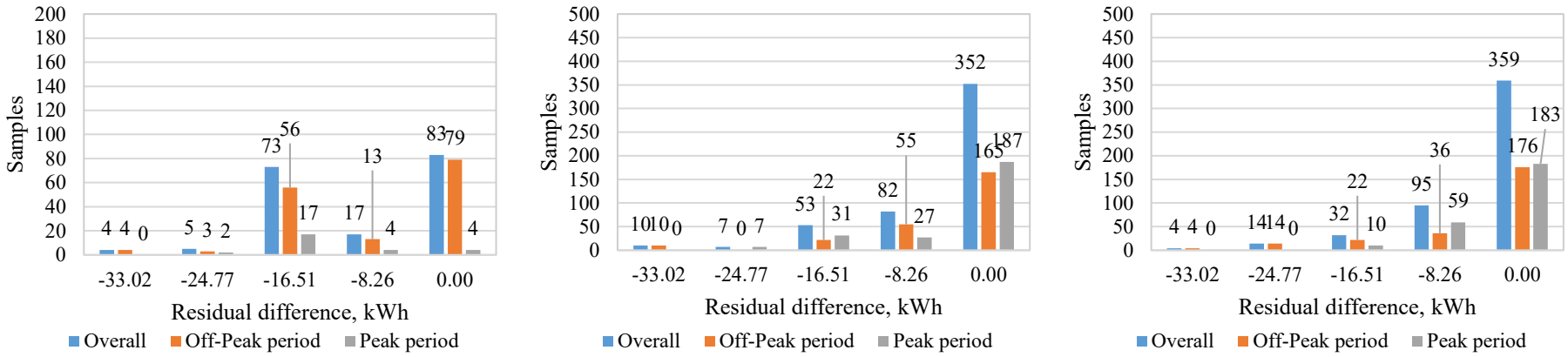
(a) Reference case (b) RS (c) RPS

Figure 5-33: Residual difference distribution during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



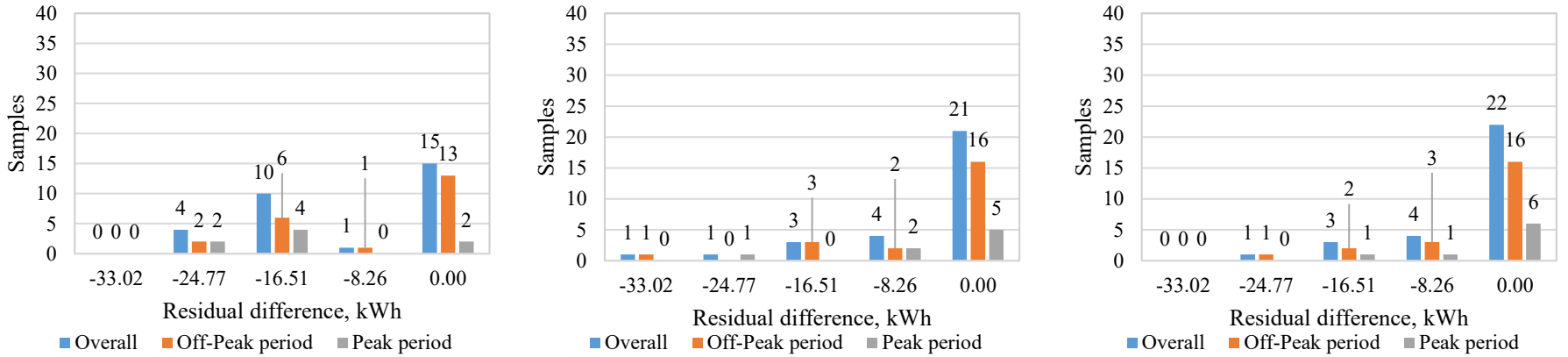
(a) Reference case (b) RS (c) RPS

Figure 5-34: Residual difference distribution during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



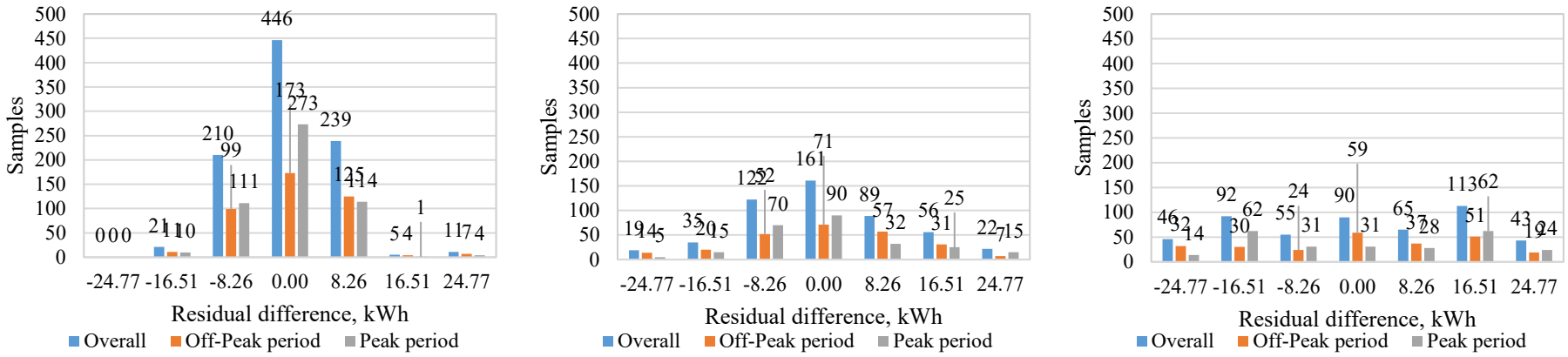
(a) Reference case (b) RS (c) RPS

Figure 5-35: Residual difference distribution for *Y*=0 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



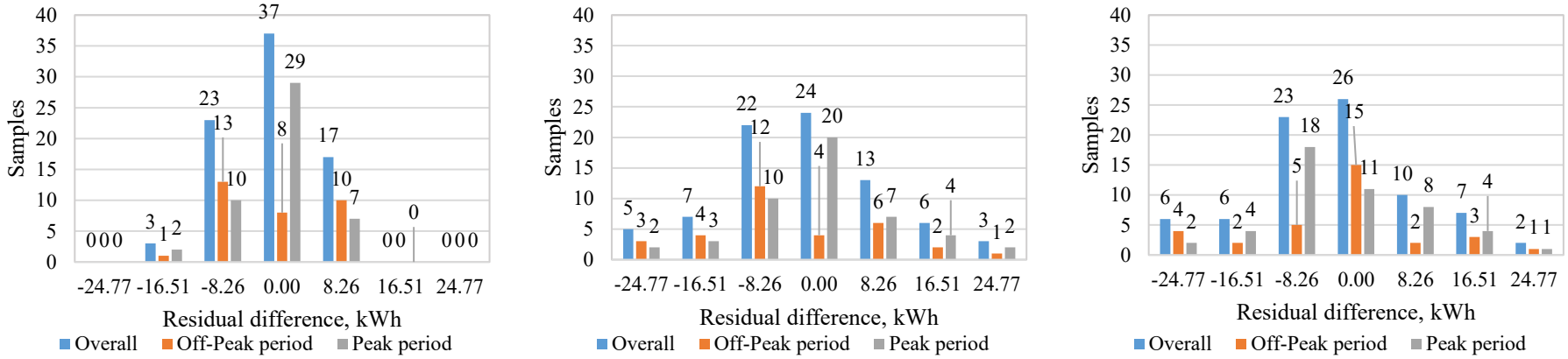
(a) Reference case (b) RS (c) RPS

Figure 5-36: Residual difference distribution for *Y*=0 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



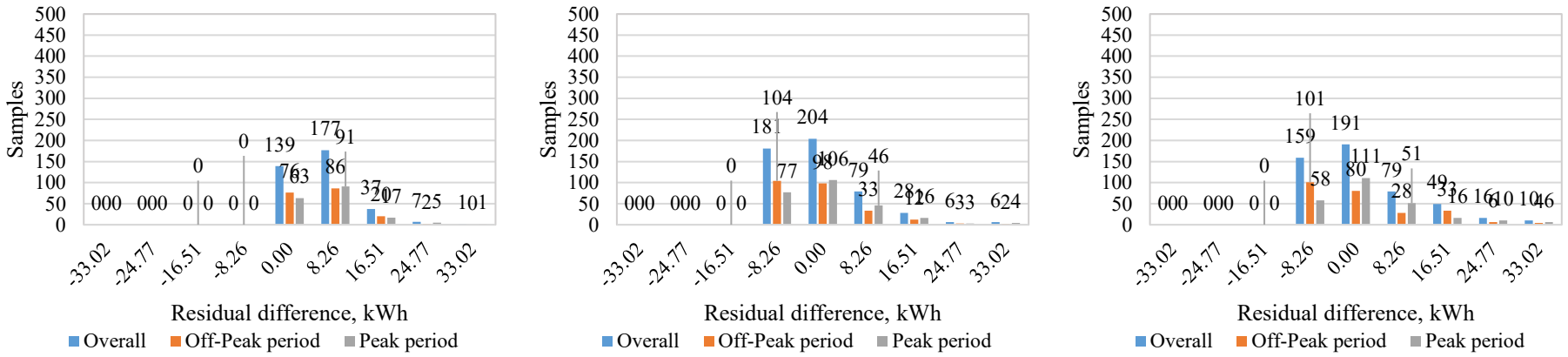
(a) Reference case (b) RS (c) RPS

Figure 5-37: Residual difference distribution for *Y*=24.77 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



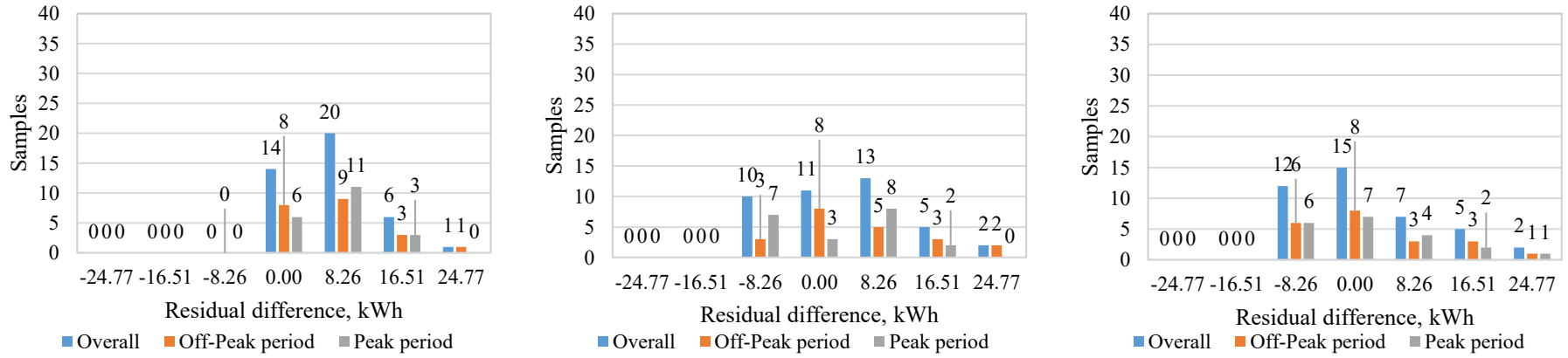
(a) Reference case (b) RS (c) RPS

Figure 5-38: Residual difference distribution for *Y*=24.77 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



(a) Reference case (b) RS (c) RPS

Figure 5-39: Residual difference distribution for *Y*=41.28 kWh during model training based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS



(a) Reference case (b) RS (c) RPS

Figure 5-40: Residual difference distribution for *Y*=41.28 kWh during model validation based on training data processed by (a) reference case (without pre-processing methods), (b) RS, and (c) RPS

5.3.1.2. Predictive result of air temperature prediction models

$X_{candidate}$ and validation dataset data distribution is presented in Table 5-3. In all datasets, indoor air temperature is lower than 21 °C most of time. Besides, off-peak periods have more data with negative air temperature category than peak periods, while peak periods have more data with an air temperature that is lower than 21 °C.

Table 5-3: $X_{candidate}$ and validation dataset distribution of air temperature prediction models

	Air temperature category	**Off-peak period**	**Peak period**
Model training	**-1**	706	491
	1	1058	1273
Model validation	**-1**	79	48
	1	93	124

Figure 5-41 shows that RPS decreases the overall predictive accuracy from 82.6% to 75.4% during model training, while RS could preserve it at 81.8%. However, RPS could ensure the overall predictive accuracy is higher than 79% during model validation, but RS decreases the accuracy to 77.3%. RS and RPS do not show a specific effect on the accuracy difference between the off-peak period and the peak period. This is because the number of samples collected during the peak period is the same as the off-peak period.

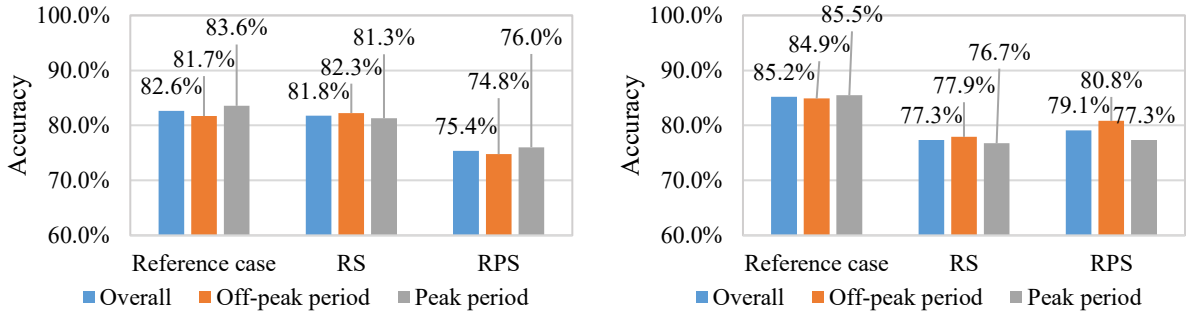


Figure 5-41: Predictive accuracy of air temperature during (a) model training and (b) model validation

From Figure 5-42, RPS and RS also decrease the predictive recall. The pattern of predictive recall is like the overall accuracy, because positive air temperature category is the majority condition during model training and validation. By contrast, as shown in Figure 5-43, RPS and RS could effectively increase the predictive accuracy of the minority condition by oversampling. The difference between recall and specificity is 17.6% during model training and 12.7% during model validation in the reference case. RPS could effectively decrease the difference to 2.5% during model training and 7.0% during model validation, while RS shows lower difference decreasing ability which is 5.5% during model training and 11% during model validation.

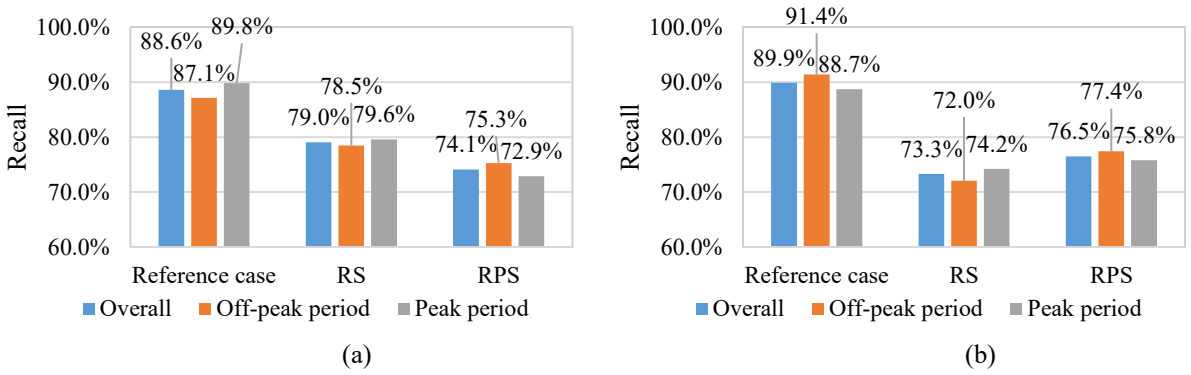


Figure 5-42: Predictive recall of air temperature during (a) model training and (b) model validation

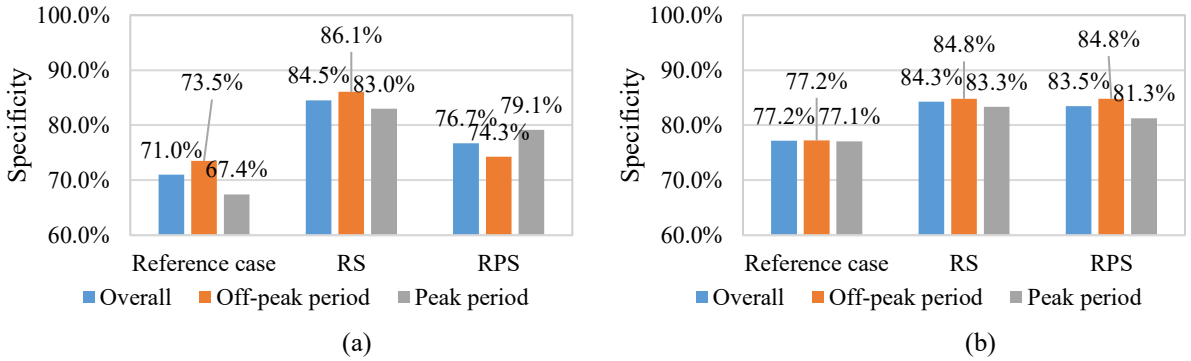


Figure 5-43: Predictive specificity of air temperature during (a) model training and (b) model validation

5.3.2. CONTROL PERFORMANCE COMPARISON

In Figure 5-44, the effect of MPCs on daily heating cost is compared with setting the set-point at a constant value, i.e., 21 °C. Using MPC could decrease the daily heating cost by ~17.8% - ~21.8%. Integrating pre-processing methods into the MPC would not reduce its cost saving ability. In fact, MPC_RS and MPC_RPS save more heating cost than the case that no pre-processing method is applied.

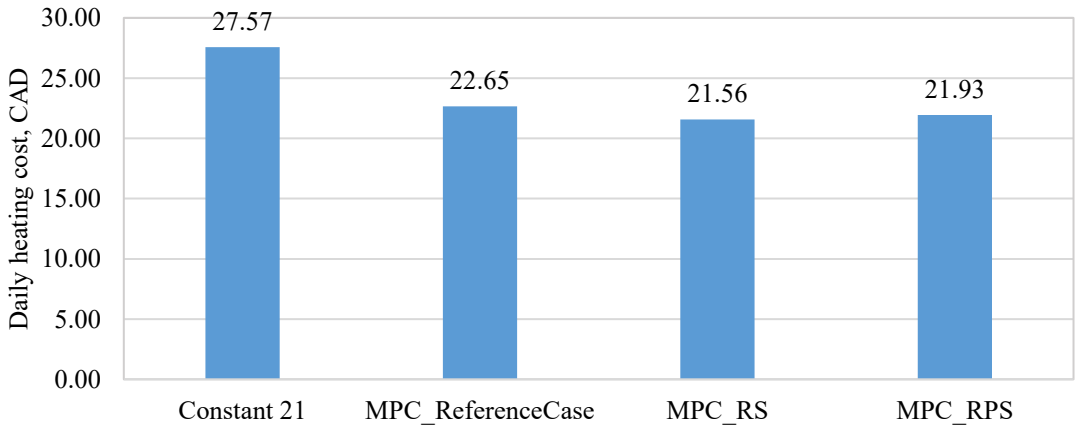


Figure 5-44: Daily heating cost of different controllers

The heating cost saving of MPCs is caused by shifting the energy consumption from peak periods to off-peak periods. From Figure 5-45, the effect of MPCs on daily energy consumption is

negligible. However, they are effective in increasing the average energy consumption during off-peak periods and decreasing the energy consumption during peak periods. Integrating fairness-aware data-driven models would not decrease the peak shifting ability of MPCs.

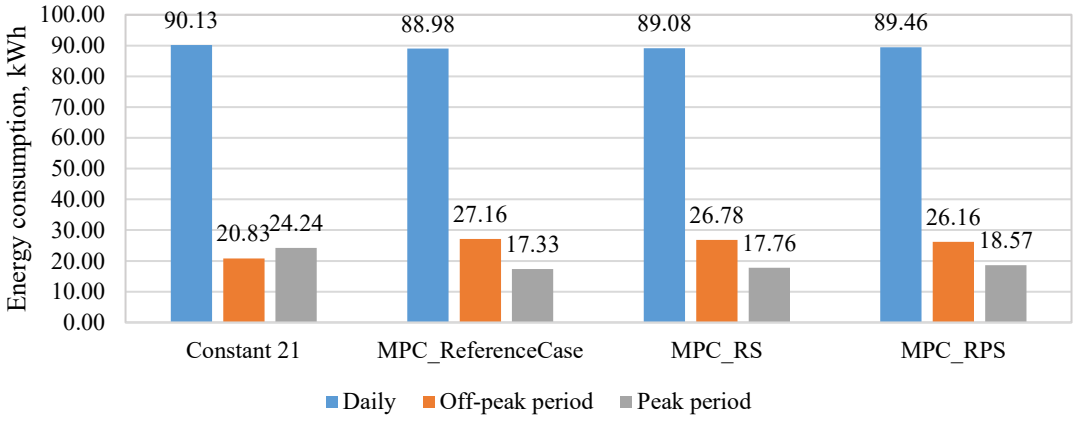


Figure 5-45: Average energy consumption of different controllers

Moreover, as shown in Figure 5-46, although MPC_RS shows a negative effect on the thermal comfort in terms of durations with indoor air temperature lower than 21 °C, MPC_ReferenceCase and MPC_RPS would improve thermal comfort. However, the thermal comfort still needs to be improved, as the indoor air temperature is lower than 21 °C for over 41.6% of simulation times. Potential thermal comfort improvement strategies are discussed in the next section.

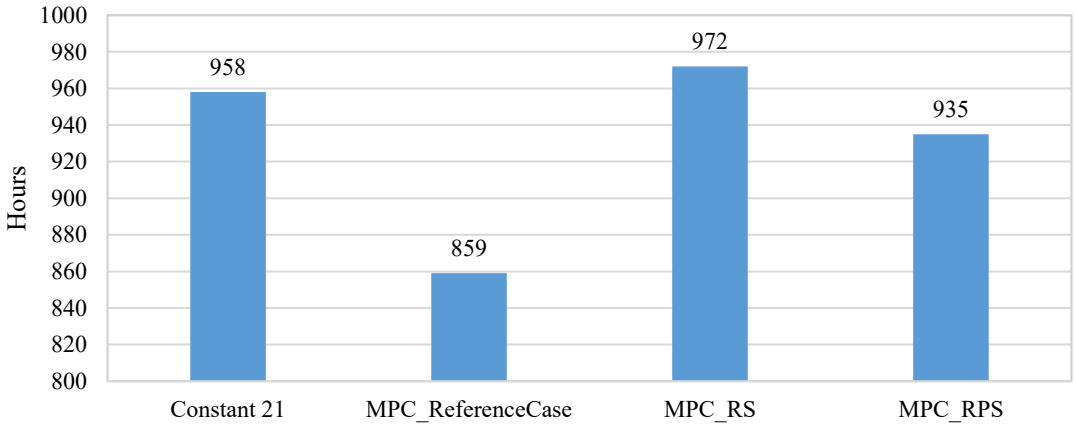


Figure 5-46: Number of hours that indoor air temperature is lower than 21 °C

Hourly set-point temperature of MPC_ReferenceCase, MPC_RS, and MPC_RPS is presented in Figure 5-47, Figure 5-48, and Figure 5-49, respectively. From these figures, MPCs, especially MPC_RS, usually set higher set-point temperature for off-peak periods and lower for peak periods. They also give a higher set-point temperature at the beginning of each peak period. It may be because MPCs try to maintain the indoor air temperature to be higher than 21 °C during the remaining time of the peak period. However, this phenomenon would limit their peak shifting ability. If the set-point temperature at the end of each off-peak period could be increased, more peak load could be shifted to the off-peak period. Reason behind this limitation include 1) the predictive accuracy of energy predictors still needs to be improved to accurately predict the future

energy consumption, 2) the optimization algorithm should be improved to get the global optimal solution.

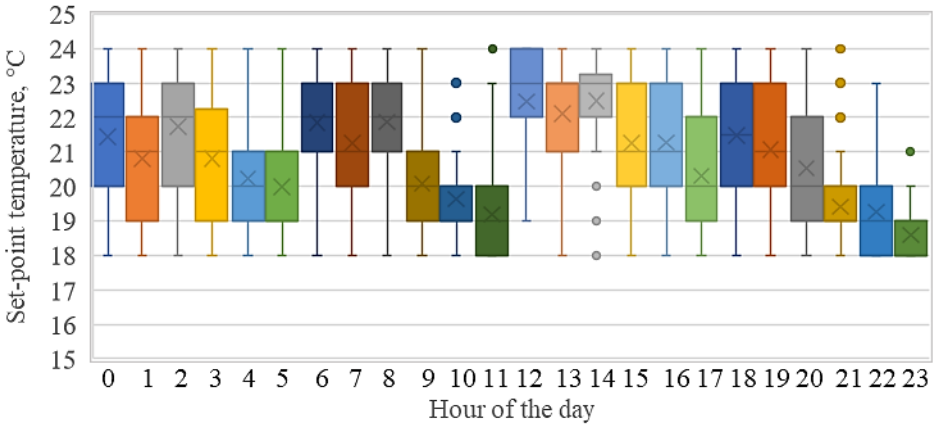


Figure 5-47: Hourly Set-point temperature of MPC_ReferenceCase

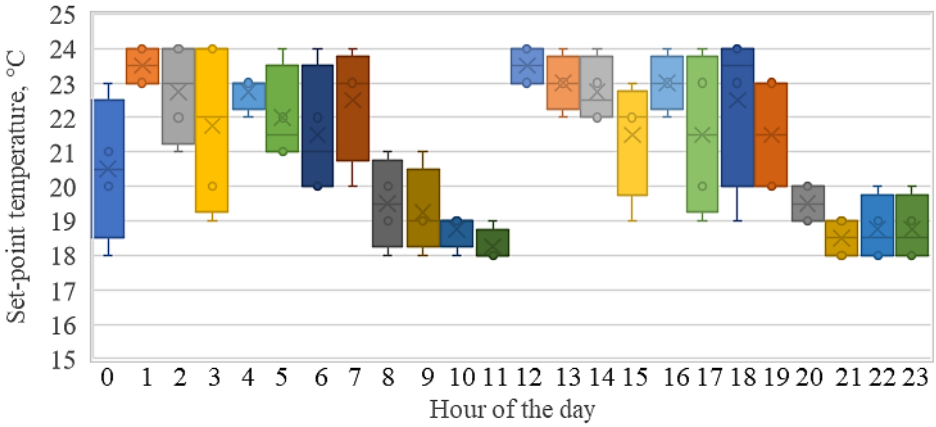


Figure 5-48: Hourly Set-point temperature of MPC_RS

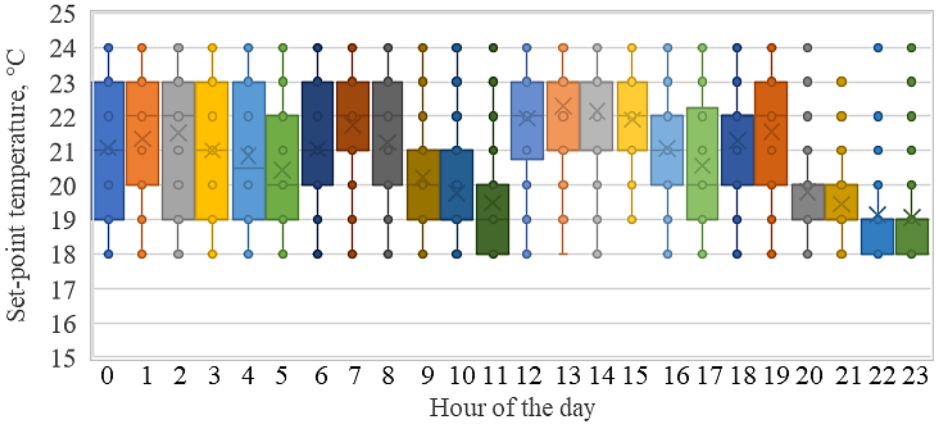


Figure 5-49: Hourly Set-point temperature of MPC_RPS

5.3.3. DISCUSSION

5.3.3.1. Effect of increasing the lower bound for set-point temperature

As illustrated before, even if constraints are added to MPCs to maintain the predicted air temperature to be higher than 21 °C, the indoor air temperature simulated based on the optimized set-point temperature could still be lower than 21 °C for over 850 hours. To solve this problem, increasing the lower bound for set-point temperature from 18 °C to 21 °C would be a good solution. As shown in Figure 5-50, this solution would effectively reduce the hours with an uncomfortable indoor air temperature to be less than 300. Fairness-aware MPCs (MPC_RS and MPC_RPS) show a slightly worse effect on improving thermal comfort than MPC_ReferenceCase. However, they could save a little more heating cost than MPC_ReferenceCase (see Figure 5-51). From Figure 5-52, the peak shifting ability of MPCs with 21 °C lower bound set-point temperature is negligible.

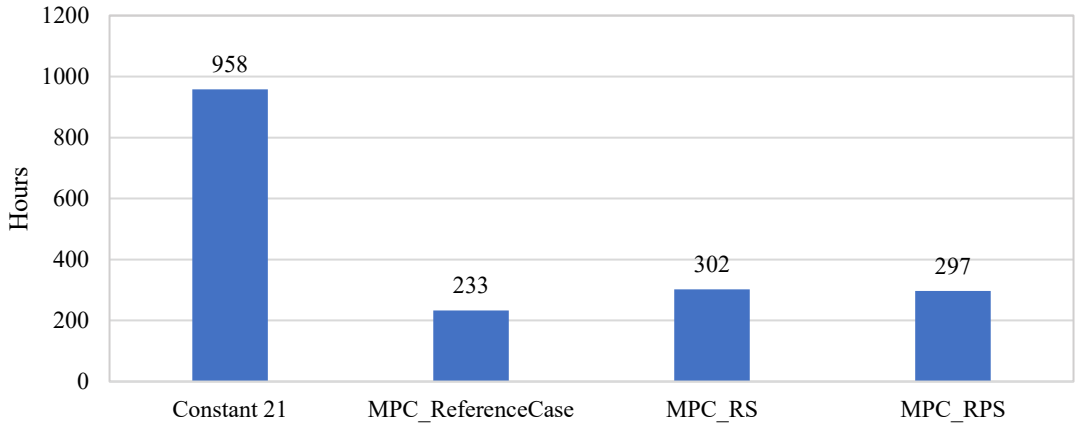


Figure 5-50: Number of hours that indoor air temperature is lower than 21 °C when setting 21°C as the lower bound in MPCs

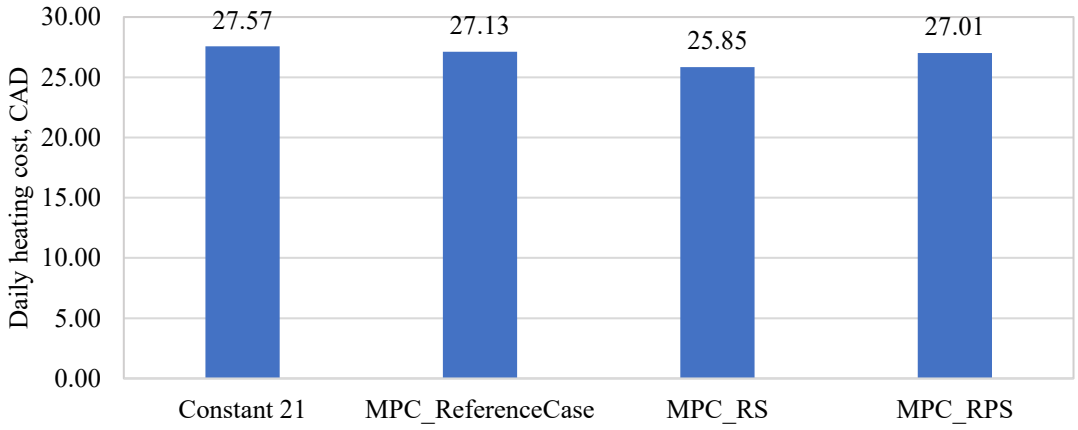


Figure 5-51: Daily heating cost when setting 21°C as the lower bound in MPCs

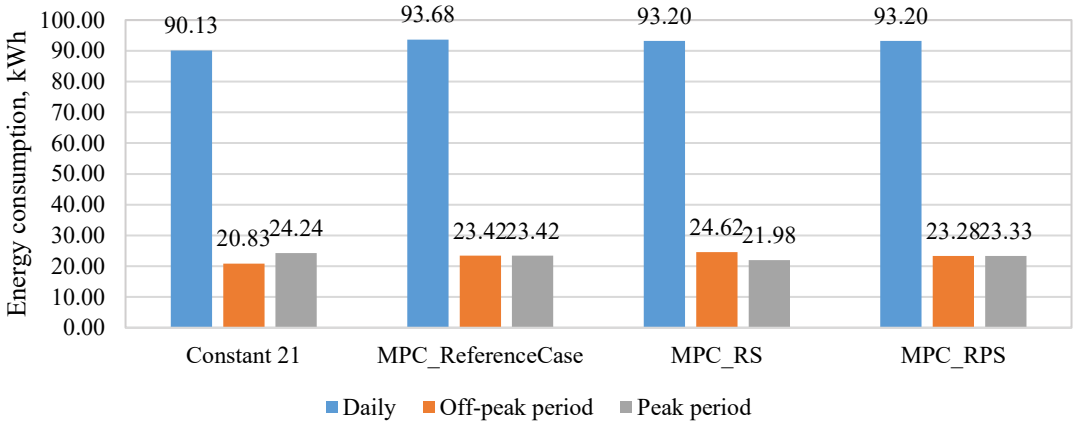


Figure 5-52: Average energy consumption when setting 21°C as the lower bound in MPCs

5.3.3.2. Effect of maximizing peak shifting

To maximize the amount of energy shifted to off-peak periods, the objective function of MPCs could be written as Equation 5-1. It maximizes the difference between energy consumed during off-peak periods and during peak periods. It could be rewritten as minimizing the energy consumption difference between peak period and off-peak period, as shown in Equation 5-2.

$$\max \widehat{Q_1} + \widehat{Q_3} - \widehat{Q_2} - \widehat{Q_4} \tag{5-1}$$

Subject to

$$\widehat{T_{min,i}} = \text{Negative}, \forall i \in [1,4],$$

$$18\ °C \leq T_{set,ji} \leq 24\ °C, \forall i \in [1,4], j \in [1,6]$$

$$\min \widehat{Q_2} + \widehat{Q_4} - \widehat{Q_1} - \widehat{Q_3} \tag{5-2}$$

Subject to

$$\widehat{T_{min,i}} = \text{Negative}, \forall i \in [1,4],$$

$$18\ °C \leq T_{set,ji} \leq 24\ °C, \forall i \in [1,4], j \in [1,6]$$

Figure 5-53 shows that using maximizing peak shifting as the objective function of MPCs would shift ~65% peak load from peak periods to off-peak periods, compared to the case with a constant set-point temperature at 21 °C. This objective function is more powerful on shifting peak load than the original one that is aimed at minimizing the heating cost.

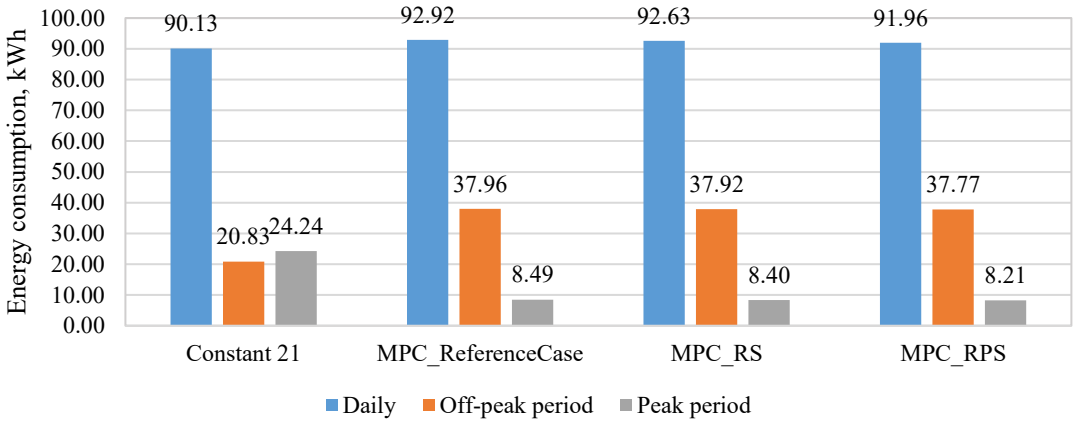


Figure 5-53: Average energy consumption when maximizing peak shifting by MPCs

One interesting finding is that although the modified objective function is aimed at maximizing peak shifting, it also works better than the original one on cost saving. From Figure 5-54, the daily heating cost of modified MPCs in this section is ~14 CAD, while the daily heating cost of the original MPCs is ~22 CAD, as shown in Figure 5-44. The cost saving of modified MPCs is resulted from setting higher set-point temperature for off-peak periods than peak periods, as shown in Figure 5-55. Furthermore, the worse result of original MPCs reveals that the optimization progress of original MPCs is not converged. This may be caused by their more complicated objective function when considering segmented electricity price. Selecting proper optimization algorithm or improving hyperparameters used in the original optimization algorithm (DE) would be potential solutions.

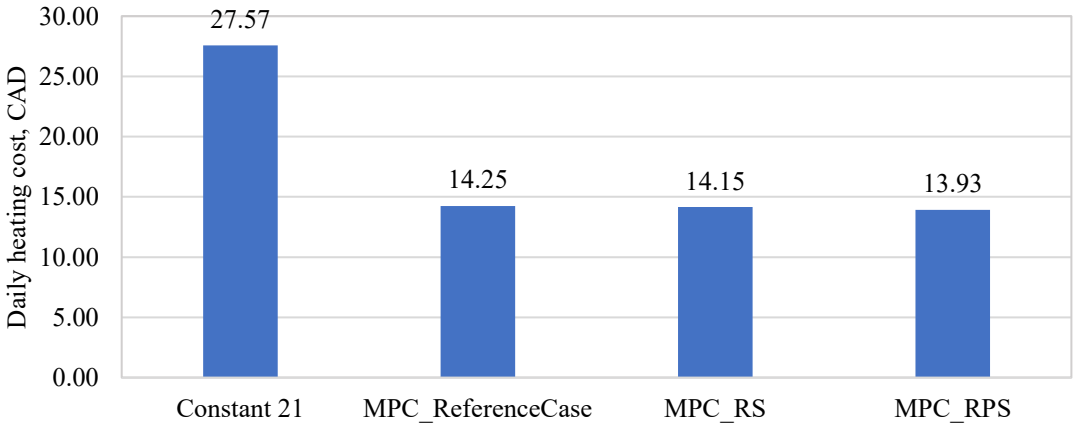


Figure 5-54: Daily heating cost when maximizing peak shifting by MPCs

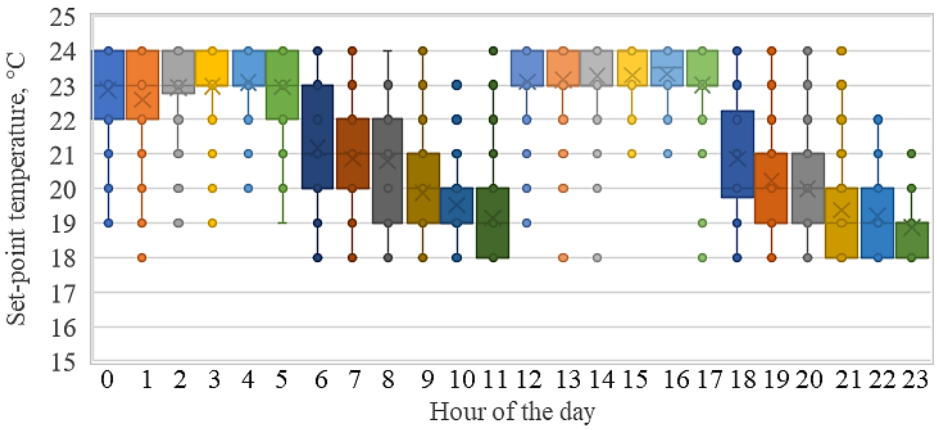


Figure 5-55: Hourly Set-point temperature of MPC_ReferenceCase using maximizing peak shifting as the objective function

The effect of the modified MPCs on thermal comfort is presented in Figure 5-56. The trend is inline with MPCs presented in Section 5.3.2 and Section 5.3.3.1: MPC_ReferenceCase works better on preserving the indoor air temperature over 21°C than MPC_RS and MPC_RPS. This is because the overall predictive accuracy of the air temperature prediction model used in MPC_ReferenceCase is higher than MPC_RS and MPC_RPS.

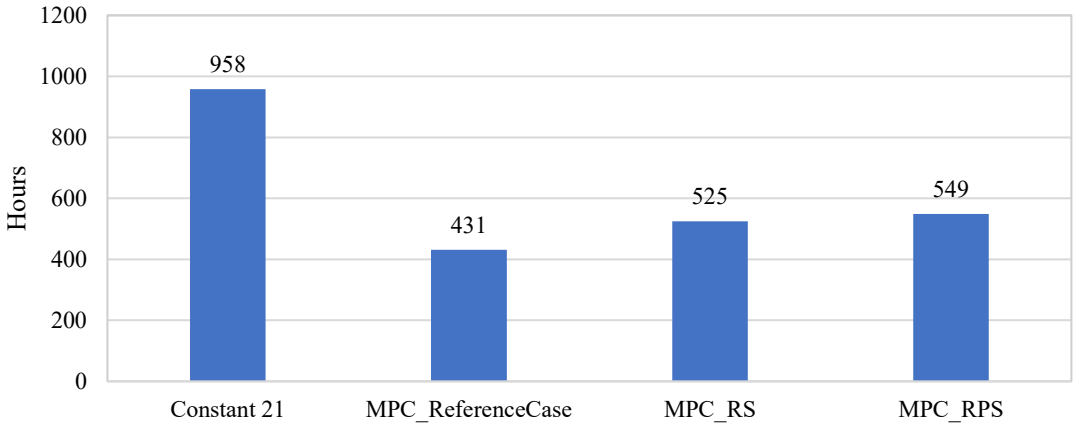
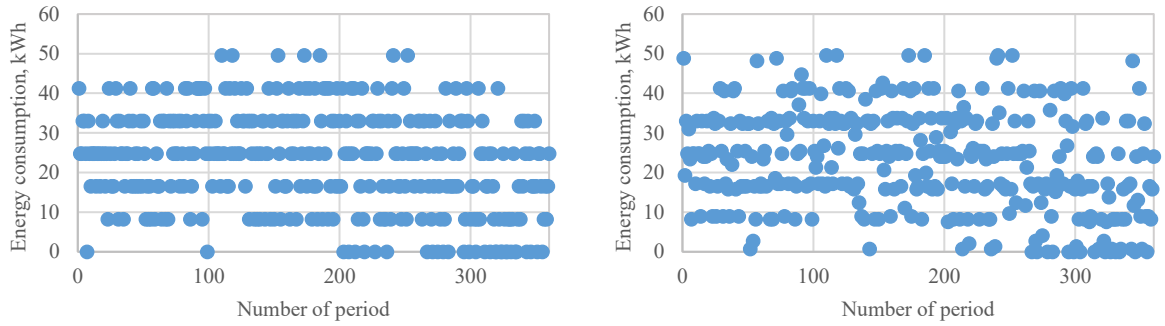


Figure 5-56: Number of hours that indoor air temperature is lower than 21 °C when maximizing peak shifting by MPCs

5.3.3.3. Others

To simplify the investigated problem, this study collected the training dataset by simulating TRNSYS model with 1-hour time interval. As a result, the energy consumption per period (duration of every 6 hours) became multi-class labels, see Figure 5-57(a). Therefore, energy predicting was a multi-class classification problem, and pre-processing methods were applied to improve the predictive performance of minority conditions. However, pre-processing methods could not achieve user-defined quantitively fairness improvement. Besides, in reality, energy demand is continuous data, which means predicting energy demand would require a regression model. Besides, when the simulation time interval is reduced to 5-min, the classes of energy consumption per period would be increased and the collected energy consumption data could be considered as continuous numbers (see Figure 5-57(b)). Therefore, integrating in-processing fairness improvement methods into MPCs to achieve user defined trade-off between predictive fairness and accuracy could be an interesting future work.



(a) (b)

Figure 5-57: Energy consumption per period simulated by TRNSYS with (a) 1-hour time interval and (b) 5-min time interval

Besides, one potential way to improve the control performance of MPCs is to improve the predictive performance of predictors. For instance, even if constraints that require the indoor air temperature to be higher than 21°C are added to the objective function of MPC, the actual indoor air temperature when applying the optimized set-point could still be lower than the comfort bound. This is mainly because the predictive recall of air temperature category models is not high enough, and the predictor would think the air temperature could be heated to be higher than 21°C when it is actually not the case. Therefore, improving the predictive accuracy of predictors integrated in MPCs is worth to be studied.

Furthermore, this study assumed that the duration of peak periods is the same as off-peak periods, in order to develop data-driven models to predict energy consumption and indoor air temperature for each period. However, in reality, the revolution of peak periods and off-peak periods could be different. Therefore, developing specific data-driven models for different periods could be considered. Moreover, decreasing the time interval of prediction to 1-hour is also a potential solution.

CHAPTER SIX: CONCLUSION

6.1. SUMMARY AND CONCLUSION

In this study, a detailed review of existing studies on DDBMs was first presented to find issues from existing studies and emphasize the requirement for achieving fairness among the predictive result of DDBMs. Three fairness concepts were summarized, and the main objective of this study is to improve *Type II* fairness to ensure a uniform predictive performance for DDBMs and investigate the applicability of fairness-aware DDBMs in MPCs. Overall, this research proposed four pre-processing methods for classification problems (such as lighting status prediction) and four in-processing methods for regression problems (such as energy prediction) to improve the fairness to have similar predictive performance between different conditions, while preserving the predictive accuracy. The effect of these proposed techniques on the predictive fairness and accuracy were investigated by implementing them to develop DDBMs for apartments. Although the proposed techniques could improve the predictive performance similarity among different conditions, their fairness improvement ability was usually evaluated by specific criteria and researchers are recommended to selected proper techniques based on their target performance. Furthermore, the proposed fairness-aware DDBMs were integrated into MPCs. A case study was conducted to apply the fairness-aware MPCs to control the EHFs in a residential building to save heating cost and shift peak load by taking advantage of thermal storage capacity of EHFs. The fairness-aware MPCs showed competitive effect on cost-saving and peak shifting, compared to the traditional data-driven based MPCs. The detailed conclusions for each part of this study are summarized as below:

The review of existing DDBMs summarized the commonly used input features for DDBMs, feature selection and feature engineering methods, supervised data-driven algorithms, factors reflected from outputs, and performance criteria for evaluating the predictive result. The potential feature types for DDBMs include meteorological information, indoor environmental information, occupancy-related data, time index, building characteristic data, socio-economic information, and historical data. Besides, feature extraction methods, such as variable ranking, filter and wrapper methods, embedded method, PCA, and AE, were reviewed. The strengths and weaknesses of these methods were summarized. For data-driven algorithms, a variety of models (such as linear regression, logistic regression, time series analysis, Naïve Bayes, DT/RT, SVM/SVR, ANN, deep learning, and ensemble methods) were introduced and their advantages, disadvantages were discussed. Factors reflected from the expected outputs (i.e., building type, output type, scale, temporal granularity) were also reviewed. Furthermore, the widely used performance criteria for classification problems and regression problems were summarized.

Through summarizing existing studies in terms of input features, data-driven algorithms, output features and output types, and performance indicators, two issues that may lead to fairness problems have been concluded:

1) The utilization of occupancy-related data may cause privacy issues and make the predictive result biased among different users.
2) Although most existing studies could provide a DDBM with high and acceptable overall predictive accuracy, the models could still suffer from the fairness problem that provides better predictive performance for certain groups/conditions than others.

To solve the above-mentioned research problems identified in DDBMs, fairness-aware data-driven models could be considered. As the first study that focuses on solving fairness issues in building engineering domain, three types of commonly used fairness definitions were firstly introduced in this study. Through giving examples in the building engineering domain, investigating Type I and Type II fairness among DDBMs shows the benefit of enabling users to do authority management, achieving uniform predictive performance under different periods or situations, and preserving fairness for different users. Furthermore, commonly used fairness improvement methods were reviewed and summarized as pre-processing methods, in-processing methods, and post-preprocessing methods. Because post-processing methods change the predictive result and lack the flexibility of achieving a trade-off between fairness and accuracy, this study only concentrated on pre-processing methods and in-processing methods. After improving the predictive fairness for DDBMs, the applicability of fairness-aware DDBMs in MPCs was also investigated.

Therefore, this study was mainly aimed at developing fairness-aware data-driven based MPCs for buildings. To be more specific, the developed DDBMs would not only provide accurate prediction but also be able to solve fairness problems in terms of achieving similar predictive performance between different periods/conditions. The fairness-aware DDBM would then be integrated into MPCs to ensure that the optimal control signal is calculated based on competitive accurate prediction of all conditions.

Accordingly, this study was separated into three tasks. The works done in each task and conclusion obtained from each task are summarized as below:

1) In task A, four pre-processing techniques namely SS, SBS, RPS, and SPS were proposed to balance the training dataset for classification problems by eventually sampling the same number of data points for conditions defined by the protected attribute(s) and output labels. Two case studies (A-1 and A-2) were conducted to compare the proposed methods with existing methods and investigate the generalizability of these pre-processing methods.
   In the case study A-1, the effect of SS, RPS and SPS on the predictive accuracy and *Type II* fairness was compared with existing methods (RS and PS). Besides, the possibility of achieving *Type I* fairness through suppressing protected attributes from inputs was also investigated. This case study used one-year data collected from one apartment building. Overall, 576 study cases were investigated to compare 6 pre-processing methods under 12 series of lighting status, 2 combinations of features, and 4 classifiers. The conclusion from this case study includes:
   - The proposed pre-processing methods could improve fairness *Type II* for a classification problem with an acceptable accuracy decrease.
   - Suppressing the protected attribute did not destroy predictive accuracy. This indicates the possibility of achieving *Type I* fairness by avoiding using the protected attribute as inputs. However, using pre-processing methods would decrease accuracy and specificity compared to cases that did not use them. Among these methods, SS and SPS worked best to preserve overall accuracy. On the other hand, pre-processing methods effectively improved the recall.
   - SS is a good option for increasing accuracy and recall rates while maintaining an acceptable specificity rate. The fairness improvement performance of other strategies, however, varies among different features and classifiers.

In the case study A-2, the generalizability of these four proposed pre-processing techniques—SS, SBS, RPS, and SPS—were studied and compared with RS by applying them to process 5 modes of Xcandidate before predicting the lighting status in 16 apartments. In total, 4,960 cases were investigated. The following conclusions are drawn from the case study:

- The imbalanced training dataset not only results in poor predictive accuracy for minority classes but also bad fairness rates.
- In terms of the effect on predictive accuracy, SBS, RS, and RPS show the most significant accuracy improvement ability for minority classes, but RS shows the most harmful influence on the accuracy of majority classes. When $X_{candidate}$ is quite imbalanced, its negative effect on the overall predictive accuracy could be unacceptable. Besides, SBS shows a comparable effect on predictive accuracy as RPS, but it is simpler as it does not require a ranker. On the other hand, SS and SPS show a slight accuracy improvement for minority classes with an acceptable price of accuracy decrease for majority classes. Rankers in pre-processing techniques could affect the predictive performance; however, no consistent pattern has been found.
- From the aspect of fairness improvement, all pre-processing techniques, especially SBS, RS, and RPS, could effectively increase the recall rate. However, RS would result in the greatest decrease in the specificity rate for Mode 1, Mode 3, and Mode 4. SS and SPS could remain the specificity rate for most cases in Mode 1 and Mode 4 to be higher than 80%. Moreover, RPS or RS often results in the lowest accuracy rate in Mode 1, Mode 3 and Mode 4.

Overall, the proposed pre-processing methods could improve Type II fairness with a price of overall accuracy decrease. However, the effect significance is varied among these methods. Researchers are recommended to select the proper pre-processing techniques based on their research objective and training data distribution.

2) In task B, four in-processing methods—MRDP, MSEP, MRDC, and MSEC—were proposed to achieve the trade-off between predictive *Type II* fairness and accuracy for regression problems. The fundamental of these methods is to set fairness-related penalties or constraints in the objective function of model training.

These in-processing methods were applied to develop linear regression models for the energy prediction of an apartment. Here, motion status was selected as the protected attribute. The effect of p/λ values of these methods on the predictive accuracy and fairness were investigated. Conclusions drawn 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 well ability to decrease the accuracy (MAE or MSE) difference between different conditions defined by the protected attribute. However, it works well in increasing the similarity of abs(MRE) between S=Positive and S=Negative.
- MSEC could decrease the difference between 1-MSE and 0-MSE by increasing the p value. 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 also a good option to increase MSE rate and MAE rate. 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 effect on the accuracy and fairness, researchers are recommended to select proper methods based on their research objective.

3) In task C, the fairness improvement methods proposed in Task A and Task B would be utilized by the DDBMs integrated into MPCs to optimize the control signal for building devices with the aim of cost-saving or peak shifting. Adding fairness-aware DDBMs into MPCs could ensure the optimal control signal is obtained based on uniform predictive performance under different conditions. It would also benefit other participants, such as utilities, to receive predicted values with uniform predictive performance.

   One case study was designed to implement pre-processing methods to process the training dataset for multi-class energy prediction models and two-class air temperature classification models. Then, these fairness-aware models were integrated into MPCs to get the next day's optimal hourly set-point temperature for EHFs, an active thermal mass storage system, in a bungalow building. Different objective functions were investigated to achieve cost-saving or peak shifting while preserving thermal comfort. Conclusions of this case study include:

   - Using RS or RPS to balance the training dataset would decrease the overall predictive accuracy for both multi-class classification problems and two-class classification problems. However, they could increase the predictive accuracy of minority classes.
   - Integrating fairness-aware DDBMs into MPCs would not show a negative effect on peak shifting or cost saving, but thermal comfort. This is because using pre-processing methods shows more negative effect on the two-class air temperature prediction models than the multi-class energy prediction models.
   - Selecting different objective functions shows a more significant effect on the control performance, such as peak shifting, cost-saving, thermal comfort, etc., than integrating fairness improvement methods. Maximizing the peak shifting ability was the most powerful objective function for improving heating cost-saving, peak shifting, and thermal comfort.

In conclusion, the proposed pre-processing methods and in-processing methods could improve *Type II* fairness for classification problems and regression problems, respectively. However, improving fairness resulted in a decreasing of overall accuracy. Researchers are recommended to select proper fairness improvement methods based on their prefer on the trade-off between fairness and accuracy. Moreover, integrating fairness-aware DDBMs into MPCs ensures that the optimal control signal is obtained based on fair prediction on indoor parameters. Fairness-aware MPCs did not show a negative impact on the control performance, compared to the traditional MPC. In other words, it is feasible to achieve fairness without compromising control performance.

6.2. FUTURE WORK AND RECOMMENDATIONS

Recommendations for future works on fairness-aware data-driven based MPCs would be given based on the limitations of this study:

1) As mentioned in Section 3.1, the proposed pre-processing methods could be applied to multi-class classification problems with multi-class protected attribute(s). However, in the case studies conducted in the current study, the proposed methods were mainly applied to specific problems, such as lighting status prediction, energy prediction, and indoor air temperature prediction. Future studies may apply the proposed pre-processing methods in other scenarios, such as fault detection and diagnosis.
2) In-processing methods were used for training linear regression models. However, 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 the objective function when training more complex and powerful data-driven models would be an interesting topic. Besides, finding fast and effective optimization algorithms to solve these objective functions would be a potential research direction.
3) Although pre-processing fairness improvement methods have been implemented to MPCs, the applicability of integrating in-processing fairness improvement methods to fairness-aware MPCs was still not studied. Besides, improving the predictive accuracy of fairness-aware DDBMs in MPCs would be vital to ensure control performance.

REFERENCES

[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] 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.

[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] 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.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] Dabirian S, Panchabikesan K, Eicker U. Occupant-centric Modeling for Urban Building Energy Simulation: Approaches, Inputs, and Data Sources-A Review. Energy Build 2021:111809.

[13] Tam WC, Fu EY, Peacock R, Reneke P, Wang J, Li J, et al. Generating Synthetic Sensor Data to Facilitate Machine Learning Paradigm for Prediction of Building Fire Hazard. Fire Technol 2020. https://doi.org/10.1007/s10694-020-01022-9.

[14] 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.

[15] 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.

[16] 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.
[17] 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.
[18] 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.
[19] 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.
[20] González-Vidal A, Jiménez F, Gómez-Skarmeta AF. A methodology for energy multivariate time series forecasting in smart buildings based on feature selection. Energy Build 2019;196:71–82. https://doi.org/10.1016/j.enbuild.2019.05.021.
[21] 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.
[22] 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.
[23] 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.
[24] 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.
[25] 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.
[26] 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.
[27] 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.
[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] Bogen M, Rieke A. Help wanted: an examination of hiring algorithms. Equity Bias Upturn Dec 2018 2018.
[30] Cohen L, Lipton ZC, Mansour Y. Efficient candidate screening under multiple tests and implications for fairness. ArXiv190511361 Cs Stat 2019.
[31] Langenkamp M, Costa A, Cheung C. Hiring Fairly in the Age of Algorithms. ArXiv Prepr ArXiv200407132 2020.
[32] 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.
[33] 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.

[34] 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.
[35] 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.
[36] Bose AJ, Hamilton W. Compositional fairness constraints for graph embeddings. ArXiv Prepr ArXiv190510674 2019.
[37] 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.
[38] Zhong Z. A Tutorial on Fairness in Machine Learning. Medium 2020. https://towardsdatascience.com/a-tutorial-on-fairness-in-machine-learning-3ff8ba1040cb (accessed January 25, 2021).
[39] Jin Y, Yan D, Zhang X, An J, Han M. A data-driven model predictive control for lighting system based on historical occupancy in an office building: Methodology development. Build Simul 2021;14:219–35. https://doi.org/10.1007/s12273-020-0638-x.
[40] Smarra F, Jain A, De Rubeis T, Ambrosini D, D'Innocenzo A, Mangharam R. Data-driven model predictive control using random forests for building energy optimization and climate control. Appl Energy 2018;226:1252–72.
[41] Wang J, Li S, Chen H, Yuan Y, Huang Y. Data-driven model predictive control for building climate control: Three case studies on different buildings. Build Environ 2019;160:106204. https://doi.org/10.1016/j.buildenv.2019.106204.
[42] Ben-Gal I. Outlier detection. Data Min. Knowl. Discov. Handb., Springer; 2005, p. 131–46.
[43] Hodge V, Austin J. A survey of outlier detection methodologies. Artif Intell Rev 2004;22:85–126.
[44] Badr W. 6 Different Ways to Compensate for Missing Values In a Dataset (Data Imputation with examples). Medium 2019. https://towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779 (accessed October 17, 2019).
[45] Wang Z, Srinivasan RS. A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. Renew Sustain Energy Rev 2017;75:796–808. https://doi.org/10.1016/j.rser.2016.10.079.
[46] Cai M, Pipattanasomporn M, Rahman S. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. Appl Energy 2019;236:1078–88. https://doi.org/10.1016/j.apenergy.2018.12.042.
[47] Alberini A, Prettico G, Shen C, Torriti J. Hot weather and residential hourly electricity demand in Italy. Energy 2019;177:44–56. https://doi.org/10.1016/j.energy.2019.04.051.
[48] Thieblemont H, Haghighat F, Moreau A, Lacroix G. Control of electrically heated floor for building load management: A simplified self-learning predictive control approach. Energy Build 2018;172:442–58. https://doi.org/10.1016/j.enbuild.2018.04.042.
[49] 2013 ASHRAE Handbook - Fundamentals (I-P Edition) - 17. Residential Cooling and Heating Load Calculations - Knovel n.d. https://app.knovel.com/web/view/pdf/show.v/rcid:kpASHRAEB2/cid:kt00TYG4J5/viewerType:pdf//root_slug:17-residential-cooling-and-heating-load-calculations/url_slug:residential-cooling-heating?cid=kt00TYG4J5&b-toc-cid=kpASHRAEB2&b-toc-root-slug=&b-toc-url-slug=residential-cooling-heating&b-toc-title=2013%20ASHRAE%20Handbook%20-%20Fundamentals%20(I-P%20Edition) (accessed July 22, 2019).

[50] Chammas M, Makhoul A, Demerjian J. An efficient data model for energy prediction using wireless sensors. Comput Electr Eng 2019;76:249–57. https://doi.org/10.1016/j.compeleceng.2019.04.002.
[51] Ding Y, Zhang Q, Yuan T, Yang K. Model input selection for building heating load prediction: A case study for an office building in Tianjin. Energy Build 2018;159:254–70. https://doi.org/10.1016/j.enbuild.2017.11.002.
[52] Wei Y, Xia L, Pan S, Wu J, Zhang X, Han M, et al. Prediction of occupancy level and energy consumption in office building using blind system identification and neural networks. Appl Energy 2019;240:276–94. https://doi.org/10.1016/j.apenergy.2019.02.056.
[53] Shetty SS, Hoang DC, Gupta M, Panda SK. Learning desk fan usage preferences for personalised thermal comfort in shared offices using tree-based methods. Build Environ 2019;149:546–60. https://doi.org/10.1016/j.buildenv.2018.12.040.
[54] Mun S-H, Kwak Y, Huh J-H. A case-centered behavior analysis and operation prediction of AC use in residential buildings. Energy Build 2019;188–189:137–48. https://doi.org/10.1016/j.enbuild.2019.02.012.
[55] Behl M, Smarra F, Mangharam R. DR-Advisor: A data-driven demand response recommender system. Appl Energy 2016;170:30–46. https://doi.org/10.1016/j.apenergy.2016.02.090.
[56] Edwards RE, New J, Parker LE. Predicting future hourly residential electrical consumption: A machine learning case study. Energy Build 2012;49:591–603. https://doi.org/10.1016/j.enbuild.2012.03.010.
[57] Yu Z, Fung BC, Haghighat F, Yoshino H, Morofsky E. A systematic procedure to study the influence of occupant behavior on building energy consumption. Energy Build 2011;43:1409–17.
[58] Ashouri M, Haghighat F, Fung BCM, Yoshino H. Development of a ranking procedure for energy performance evaluation of buildings based on occupant behavior. Energy Build 2019;183:659–71. https://doi.org/10.1016/j.enbuild.2018.11.050.
[59] Wang Z, Hong T, Piette MA. Predicting plug loads with occupant count data through a deep learning approach. Energy 2019;181:29–42. https://doi.org/10.1016/j.energy.2019.05.138.
[60] Sala-Cardoso E, Delgado-Prieto M, Kampouropoulos K, Romeral L. Activity-aware HVAC power demand forecasting. Energy Build 2018;170:15–24. https://doi.org/10.1016/j.enbuild.2018.03.087.
[61] Gul MS, Patidar S. Understanding the energy consumption and occupancy of a multi-purpose academic building. Energy Build 2015;87:155–65. https://doi.org/10.1016/j.enbuild.2014.11.027.
[62] Fan C, Xiao F, Wang S. Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. Appl Energy 2014;127:1–10. https://doi.org/10.1016/j.apenergy.2014.04.016.
[63] 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.
[64] Seyedzadeh S, Pour Rahimian F, Rastogi P, Glesk I. Tuning machine learning models for prediction of building energy loads. Sustain Cities Soc 2019;47:101484. https://doi.org/10.1016/j.scs.2019.101484.
[65] Wei L, Tian W, Zuo J, Yang Z-Y, Liu Y, Yang S. Effects of Building Form on Energy Use for Buildings in Cold Climate Regions. 8th Int Cold Clim HVAC Conf 2016;146:182–9. https://doi.org/10.1016/j.proeng.2016.06.370.

[66] Tsanas A, Xifara A. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. Energy Build 2012;49:560–7. https://doi.org/10.1016/j.enbuild.2012.03.003.
[67] Sekhar Roy S, Roy R, Balas VE. Estimating heating load in buildings using multivariate adaptive regression splines, extreme learning machine, a hybrid model of MARS and ELM. Renew Sustain Energy Rev 2018;82:4256–68. https://doi.org/10.1016/j.rser.2017.05.249.
[68] Kumar S, Pal SK, Singh RP. A novel method based on extreme learning machine to predict heating and cooling load through design and structural attributes. Energy Build 2018;176:275–86. https://doi.org/10.1016/j.enbuild.2018.06.056.
[69] Kumar S, Pal SK, Singh RP. Intra ELM variants ensemble based model to predict energy performance in residential buildings. Sustain Energy Grids Netw 2018;16:177–87. https://doi.org/10.1016/j.segan.2018.07.001.
[70] Kuster C, Rezgui Y, Mourshed M. Electrical load forecasting models: A critical systematic review. Sustain Cities Soc 2017;35:257–70. https://doi.org/10.1016/j.scs.2017.08.009.
[71] Lindberg KB, Seljom P, Madsen H, Fischer D, Korpås M. Long-term electricity load forecasting: Current and future trends. Util Policy 2019;58:102–19. https://doi.org/10.1016/j.jup.2019.04.001.
[72] He Y, Qin Y, Wang S, Wang X, Wang C. Electricity consumption probability density forecasting method based on LASSO-Quantile Regression Neural Network. Appl Energy 2019;233–234:565–75. https://doi.org/10.1016/j.apenergy.2018.10.061.
[73] He Y, Zheng Y, Xu Q. Forecasting energy consumption in Anhui province of China through two Box-Cox transformation quantile regression probability density methods. Measurement 2019;136:579–93. https://doi.org/10.1016/j.measurement.2019.01.008.
[74] Beyca OF, Ervural BC, Tatoglu E, Ozuyar PG, Zaim S. Using machine learning tools for forecasting natural gas consumption in the province of Istanbul. Energy Econ 2019;80:937–49. https://doi.org/10.1016/j.eneco.2019.03.006.
[75] Wang R, Lu S, Li Q. Multi-criteria comprehensive study on predictive algorithm of hourly heating energy consumption for residential buildings. Sustain Cities Soc 2019;49:101623. https://doi.org/10.1016/j.scs.2019.101623.
[76] Ahmad MW, Mourshed M, Rezgui Y. Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. Energy Build 2017;147:77–89. https://doi.org/10.1016/j.enbuild.2017.04.038.
[77] Ding Y, Zhang Q, Yuan T, Yang F. Effect of input variables on cooling load prediction accuracy of an office building. Appl Therm Eng 2018;128:225–34. https://doi.org/10.1016/j.applthermaleng.2017.09.007.
[78] Huang Y, Yuan Y, Chen H, Wang J, Guo Y, Ahmad T. A novel energy demand prediction strategy for residential buildings based on ensemble learning. Energy Procedia 2019;158:3411–6. https://doi.org/10.1016/j.egypro.2019.01.935.
[79] Wang Z, Wang Y, Srinivasan RS. A novel ensemble learning approach to support building energy use prediction. Energy Build 2018;159:109–22. https://doi.org/10.1016/j.enbuild.2017.10.085.
[80] Pearson K. VII. Note on regression and inheritance in the case of two parents. Proc R Soc Lond 1895;58:240–2.
[81] Spearman C. The proof and measurement of association between two things. 1961.
[82] Claesen M, De Moor B. Hyperparameter search in machine learning. ArXiv Prepr ArXiv150202127 2015.

[83] Zeng A, Liu S, Yu Y. Comparative study of data driven methods in building electricity use prediction. Energy Build 2019;194:289–300. https://doi.org/10.1016/j.enbuild.2019.04.029.
[84] Khaire UM, Dhanalakshmi R. Stability of feature selection algorithm: A review. J King Saud Univ - Comput Inf Sci 2019. https://doi.org/10.1016/j.jksuci.2019.06.012.
[85] Yuan P, Duanmu L, Wang Z. Coal consumption prediction model of space heating with feature selection for rural residences in severe cold area in China. Sustain Cities Soc 2019;50:101643. https://doi.org/10.1016/j.scs.2019.101643.
[86] ASHRAE. ASHRAE Guideline 14–2014, Measurement of Energy, Demand, and Water Savings. ASHRAE Atlanta; 2014.
[87] Jain R. K., Damoulas T., Kontokosta C. E. Towards Data-Driven Energy Consumption Forecasting of Multi-Family Residential Buildings: Feature Selection via The Lasso. Comput Civ Build Eng 2014 n.d.:1675–82. https://doi.org/10.1061/9780784413616.208.
[88] Pearson K. LIII. On lines and planes of closest fit to systems of points in space. Lond Edinb Dublin Philos Mag J Sci 1901;2:559–72.
[89] Schölkopf B, Smola A, Müller K-R. Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 1998;10:1299–319.
[90] Xuemei L, Lixing D, Jinhu L, Gang X, Jibin L. A Novel Hybrid Approach of KPCA and SVM for Building Cooling Load Prediction. 2010 Third Int. Conf. Knowl. Discov. Data Min., 2010, p. 522–6. https://doi.org/10.1109/WKDD.2010.137.
[91] Yildiz B, Bilbao JI, Sproul AB. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. Renew Sustain Energy Rev 2017;73:1104–22. https://doi.org/10.1016/j.rser.2017.02.023.
[92] From Autoencoder to Beta-VAE. Lil'Log 2018. https://lilianweng.github.io/2018/08/12/from-autoencoder-to-beta-vae.html (accessed July 24, 2019).
[93] Fan C, Sun Y, Zhao Y, Song M, Wang J. Deep learning-based feature engineering methods for improved building energy prediction. Appl Energy 2019;240:35–45. https://doi.org/10.1016/j.apenergy.2019.02.052.
[94] Mujeeb S, Javaid N. ESAENARX and DE-RELM: Novel Schemes for Big Data Predictive Analytics of Electricity Load and Price. Sustain Cities Soc 2019. https://doi.org/10.1016/j.scs.2019.101642.
[95] 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.
[96] 1.9. Naive Bayes — scikit-learn 0.23.2 documentation n.d. https://scikit-learn.org/stable/modules/naive_bayes.html (accessed September 15, 2020).
[97] Basak D. Support Vector Regression. Int J Neural Inf Process – Lett Rev 2007;11:203–24.
[98] Clarke SM, Griebsch JH, Simpson TW. Analysis of Support Vector Regression for Approximation of Complex Engineering Analyses. J Mech Des 2004;127:1077–87. https://doi.org/10.1115/1.1897403.
[99] Awad M, Khanna R. Support Vector Regression. In: Awad M, Khanna R, editors. Effic. Learn. Mach. Theor. Concepts Appl. Eng. Syst. Des., Berkeley, CA: Apress; 2015, p. 67–80. https://doi.org/10.1007/978-1-4302-5990-9_4.
[100] Kleene SC. Representation of events in nerve nets and finite automata. RAND PROJECT AIR FORCE SANTA MONICA CA; 1951.
[101] 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.

[102] Mat Daut MA, Hassan MY, Abdullah H, Rahman HA, Abdullah MP, Hussin F. Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: A review. Renew Sustain Energy Rev 2017;70:1108–18. https://doi.org/10.1016/j.rser.2016.12.015.
[103] Bengio Y. Learning Deep Architectures for AI. Found Trends Mach Learn 2009;2:1–127. https://doi.org/10.1561/2200000006.
[104] Zell A, Mache N, Huebner R, Mamier G, Vogt M, Schmalzl M, et al. SNNS (stuttgart neural network simulator). Neural Netw. Simul. Environ., Springer; 1994, p. 165–86.
[105] Goodfellow I, Bengio Y, Courville A. Deep Learning. Cambridge, Massachusetts: The MIT Press; 2016.
[106] [1409.1556] Very Deep Convolutional Networks for Large-Scale Image Recognition n.d. https://arxiv.org/abs/1409.1556 (accessed November 6, 2019).
[107] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst., 2012, p. 1097–105.
[108] Li Q, Cai W, Wang X, Zhou Y, Feng DD, Chen M. Medical image classification with convolutional neural network. 2014 13th Int. Conf. Control Autom. Robot. Vis. ICARCV, IEEE; 2014, p. 844–8.
[109] Kim Y. Convolutional Neural Networks for Sentence Classification. ArXiv14085882 Cs 2014.
[110] Sadaei HJ, de Lima e Silva PC, Guimarães FG, Lee MH. Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series. Energy 2019;175:365–77. https://doi.org/10.1016/j.energy.2019.03.081.
[111] Opitz D, Maclin R. Popular Ensemble Methods: An Empirical Study. J Artif Intell Res 1999;11:169–98. https://doi.org/10.1613/jair.614.
[112] 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.
[113] Wang Z, Wang Y, Zeng R, Srinivasan RS, Ahrentzen S. Random Forest based hourly building energy prediction. Energy Build 2018;171:11–25. https://doi.org/10.1016/j.enbuild.2018.04.008.
[114] Johannesen NJ, Kolhe M, Goodwin M. Relative evaluation of regression tools for urban area electrical energy demand forecasting. J Clean Prod 2019;218:555–64. https://doi.org/10.1016/j.jclepro.2019.01.108.
[115] Robinson C, Dilkina B, Hubbs J, Zhang W, Guhathakurta S, Brown MA, et al. Machine learning approaches for estimating commercial building energy consumption. Appl Energy 2017;208:889–904. https://doi.org/10.1016/j.apenergy.2017.09.060.
[116] Walther J, Spanier D, Panten N, Abele E. Very short-term load forecasting on factory level – A machine learning approach. Procedia CIRP 2019;80:705–10. https://doi.org/10.1016/j.procir.2019.01.060.
[117] 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.
[118] Beghi A, Brignoli R, Cecchinato L, Menegazzo G, Rampazzo M, Simmini F. Data-driven fault detection and diagnosis for HVAC water chillers. Control Eng Pract 2016;53:79–91.
[119] Ding Y, Zhang Q, Yuan T. Research on short-term and ultra-short-term cooling load prediction models for office buildings. Energy Build 2017;154:254–67. https://doi.org/10.1016/j.enbuild.2017.08.077.

[120] Hong T, Fan S. Probabilistic electric load forecasting: A tutorial review. Int J Forecast 2016;32:914–38. https://doi.org/10.1016/j.ijforecast.2015.11.011.
[121] Willmott CJ, Matsuura K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Clim Res 2005;30:79–82.
[122] de Myttenaere A, Golden B, Le Grand B, Rossi F. Mean Absolute Percentage Error for regression models. Neurocomputing 2016;192:38–48. https://doi.org/10.1016/j.neucom.2015.12.114.
[123] Lehmann EL, Casella G. Theory of point estimation. Springer Science & Business Media; 2006.
[124] Cameron AC, Windmeijer FA. An R-squared measure of goodness of fit for some common nonlinear regression models. J Econom 1997;77:329–42.
[125] Tetko IV, Livingstone DJ, Luik AI. Neural network studies. 1. Comparison of overfitting and overtraining. J Chem Inf Comput Sci 1995;35:826–33.
[126] Webster L, Bradford J, Sartor D, Shonder J, Atkin E, Dunnivant S, et al. M&V Guidelines: Measurement and Verification for Performance-Based Contracts. Version 4.0, Technical Report; 2015.
[127] Committee I. International Performance Measurement and Verification Protocol: Concepts and options for determining energy and water savings, Volume I. National Renewable Energy Lab., Golden, CO (US); 2001.
[128] Happle G, Fonseca JA, Schlueter A. A review on occupant behavior in urban building energy models. Energy Build 2018;174:276–92. https://doi.org/10.1016/j.enbuild.2018.06.030.
[129] Menezes AC, Cripps A, Bouchlaghem D, Buswell R. Predicted vs. actual energy performance of non-domestic buildings: Using post-occupancy evaluation data to reduce the performance gap. Appl Energy 2012;97:355–64.
[130] Yun GY, Kong HJ, Kim JT. A field survey of occupancy and air-conditioner use patterns in open plan offices. Indoor Built Environ 2011;20:137–47.
[131] Naghiyev E, Gillott M, Wilson R. Three unobtrusive domestic occupancy measurement technologies under qualitative review. Energy Build 2014;69:507–14.
[132] Tien PW, Wei S, Calautit JK, Darkwa J, Wood C. A vision-based deep learning approach for the detection and prediction of occupancy heat emissions for demand-driven control solutions. Energy Build 2020;226:110386.
[133] Yuchi W, Gombojav E, Boldbaatar B, Galsuren J, Enkhmaa S, Beejin B, et al. Evaluation of random forest regression and multiple linear regression for predicting indoor fine particulate matter concentrations in a highly polluted city. Environ Pollut 2019;245:746–53.
[134] Candanedo LM, Feldheim V. Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models. Energy Build 2016;112:28–39.
[135] Corbett-Davies S, Goel S. The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning. ArXiv180800023 Cs 2018.
[136] 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.
[137] Kleinberg J, Mullainathan S, Raghavan M. Inherent Trade-Offs in the Fair Determination of Risk Scores. ArXiv160905807 Cs Stat 2016.

[138] 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.
[139] 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.
[140] Feldman M, Friedler S, Moeller J, Scheidegger C, Venkatasubramanian S. Certifying and removing disparate impact. ArXiv14123756 Cs Stat 2015.
[141] Yan K, Zhong C, Ji Z, Huang J. Semi-supervised learning for early detection and diagnosis of various air handling unit faults. Energy Build 2018;181:75–83. https://doi.org/10.1016/j.enbuild.2018.10.016.
[142] Yan K, Huang J, Shen W, Ji Z. Unsupervised learning for fault detection and diagnosis of air handling units. Energy Build 2020;210:109689.
[143] 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).
[144] Zafar MB, Valera I, Rogriguez MG, Gummadi KP. Fairness constraints: Mechanisms for fair classification. Artif. Intell. Stat., PMLR; 2017, p. 962–70.
[145] van Kleef RC, McGuire TG, van Vliet RC, van de Ven WP. Improving risk equalization with constrained regression. Eur J Health Econ 2017;18:1137–56.
[146] Fitzsimons J, Al Ali A, Osborne M, Roberts S. A general framework for fair regression. Entropy 2019;21:741.
[147] Berk R, Heidari H, Jabbari S, Joseph M, Kearns M, Morgenstern J, et al. A convex framework for fair regression. ArXiv Prepr ArXiv170602409 2017.
[148] 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.
[149] Olfat M, Mintz Y. Flexible Regularization Approaches for Fairness in Deep Learning. 2020 59th IEEE Conf. Decis. Control CDC, IEEE; 2020, p. 3389–94.
[150] Wadsworth C, Vera F, Piech C. Achieving fairness through adversarial learning: an application to recidivism prediction. ArXiv Prepr ArXiv180700199 2018.
[151] Zhang BH, Lemoine B, Mitchell M. Mitigating unwanted biases with adversarial learning. Proc. 2018 AAAIACM Conf. AI Ethics Soc., 2018, p. 335–40.
[152] 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.
[153] Jufri FH, Oh S, Jung J, Choi M-H. A Method to Forecast Storm-Caused Distribution Grid Damages Using Cost-Sensitive Regression Algorithm. 2019 IEEE Innov. Smart Grid Technol. - Asia ISGT Asia, 2019, p. 3986–90. https://doi.org/10.1109/ISGT-Asia.2019.8880929.
[154] Pinto T, Praça I, Vale Z, Silva J. Ensemble learning for electricity consumption forecasting in office buildings. Neurocomputing 2021;423:747–55. https://doi.org/10.1016/j.neucom.2020.02.124.
[155] Hardt M, Price E, Srebro N. Equality of opportunity in supervised learning. ArXiv Prepr ArXiv161002413 2016.
[156] 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.

[157] Conn AR, Scheinberg K, Vicente LN. Introduction to derivative-free optimization. SIAM; 2009.
[158] Georgioudakis M, Plevris V. A Comparative Study of Differential Evolution Variants in Constrained Structural Optimization. Front Built Environ 2020;6.
[159] 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.
[160] 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.
[161] Cortes C, Vapnik V. Support-vector networks. Mach Learn 1995;20:273–97. https://doi.org/10.1007/BF00994018.
[162] 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.
[163] sklearn.model_selection.RandomizedSearchCV. Scikit-Learn n.d. https://scikit-learn/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html (accessed November 5, 2021).
[164] 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.
[165] 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.
[166] Fei Guo. scikit-opt. 2022.
[167] Rate Flex D | Hydro-Québec n.d. http://www.hydroquebec.com/residentiel/espace-clients/tarifs/tarif-flex-d.html (accessed March 12, 2020).
[168] Aongya S. Contrôle du Chauffage Pour la Gestion de la Demande Résidentielle—Rapport Technique sur la Création d'un Modèle Résidentiel Fonctionnel. Hydro-Quebec Shawinigan QC Can 2010.
[169] Thieblemont H. Simplified Predictive Control for Load Management: A Self-Learning Approach Applied to Electrically Heated Floor. PhD Thesis. Concordia University, 2017.
[170] Sun Y, Panchabikesan K, Haghighat F, Luo J (Tom), Moreau A, Robichaud M. Development of advanced controllers to extend the peak shifting possibilities in the residential buildings. J Build Eng 2021;43:103026. https://doi.org/10.1016/j.jobe.2021.103026.
[171] Sun Y, Panchabikesan K, Joybari MM, Olsthoorn D, Moreau A, Robichaud M, et al. Enhancement in peak shifting and shaving potential of electrically heated floor residential buildings using heat extraction system. J Energy Storage 2018;18:435–46.
[172] Montréal V de. Heating dwelling units n.d. https://montreal.ca/en/topics/heating-dwelling-units (accessed June 16, 2022).
[173] 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.
[174] 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.
[175] Wang S, Jin X. Model-based optimal control of VAV air-conditioning system using genetic algorithm. Build Environ 2000;35:471–87.

[176] 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.
[177] 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.
[178] Whitley D. A genetic algorithm tutorial. Stat Comput 1994;4:65–85. https://doi.org/10.1007/BF00175354.
[179] 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).
[180] sklearn.svm.SVC — scikit-learn 0.23.1 documentation n.d. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed July 12, 2020).
[181] 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 12, 2020).
[182] 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 12, 2020).
[183] 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 12, 2020).

APPENDIX A – HYPERPARAMETERS FOR CLASSIFIERS DEVELOPED IN CASE STUDY A-1

SVM

In this study, SVM classifiers are modelled by using sklearn.svm.SVC [180] in python. Hyperparameters for these classifiers are listed in Table A1. Detailed explanation for the meaning of each hyperparameter could be found in [180].

Table A1: 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 [181] in python. Hyperparameters for these classifiers are listed in Table A2. Detailed explanation for the meaning of each hyperparameter could be found in [181].

Table A2: 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 [182] in python. Their hyperparameters are listed in Table A3.

Table A3: 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 [183] in python. The hyperparameters are listed in Table A4.

Table A4: Hyperparameters for Gaussian Naive Bayes classifiers

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

APPENDIX B – CORRELATION MATRIX BETWEEN INPUT FEATURES AND OUTPUT IN CASE STUDY A-2

Table B1: Detailed correlation matrix values

Mode	APT	Light	Day of The Week	Hour of The Day	Time of The Day	Altitude of The Sun	Global Horizontal Illuminance	Diffuse Horizontal Illuminance	Global Horizontal Irradiance	Diffuse Horizontal Irradiance	Motion Status_1	Motion Status_2	Motion Status_3	Motion Status_4	Motion Status_5	Motion Status_6	Motion Status_7	Motion Status_8	Motion Status_9	Motion Status_10	Motion Status_11	Motion Status_12	Motion Status_13	Motion Status_14
Mode 1	APT #1	Light_1	-0.01	0.02	0.02	-0.01	-0.02	-0.02	-0.02	-0.02	0.11	0.04	0.03	0.02	0.02	0.04	0.02	0.00	0.00	-0.00	0.00	0.01	0.01	0.01
		Light_4	0.02	0.05	0.05	-0.02	-0.03	-0.03	-0.03	-0.02	0.03	0.05	0.08	0.02	0.01	0.02	0.02	0.02	0.03	0.01	0.01	0.02	0.02	0.02
		Light_7	0.01	0.08	0.08	-0.05	-0.05	-0.05	-0.05	-0.04	0.05	0.13	0.13	0.11	0.09	0.13	0.10	0.07	0.08	0.02	0.06	0.06	0.02	0.04
		Light_8	-0.02	0.07	0.07	-0.07	-0.05	-0.05	-0.05	-0.04	0.05	0.06	0.05	0.04	0.03	0.05	0.04	0.10	0.09	0.11	0.04	0.08	0.03	0.05
		Light_9	0.01	0.03	0.03	0.01	-0.00	-0.01	-0.00	-0.01	0.05	0.10	0.08	0.08	0.09	0.09	0.22	0.12	0.03	0.06	0.14	0.12	0.09	0.53
		Light_10	-0.01	0.04	0.04	-0.04	-0.02	-0.03	-0.02	-0.02	0.02	0.04	0.04	0.02	0.02	0.03	0.07	0.04	0.03	0.03	0.09	0.04	0.04	0.02
		Light_11	-0.01	-0.01	-0.01	0.03	0.02	0.01	0.02	0.01	0.01	0.03	0.03	0.04	0.04	0.03	0.10	0.09	0.02	0.05	0.16	0.11	0.60	0.09
		Light_12	-0.00	-0.01	-0.01	0.03	0.02	0.02	0.02	0.01	0.01	0.03	0.04	0.04	0.04	0.04	0.10	0.08	0.03	0.05	0.16	0.10	0.58	0.09
	APT #2	Light_2	-0.02	0.15	0.15	-0.04	-0.10	-0.09	-0.10	-0.09	0.25	0.16	0.27	0.07	0.07	0.12	0.09	0.08	0.04	0.06	0.04	0.09	0.04	0.24
		Light_3	-0.00	0.14	0.14	-0.07	-0.09	-0.09	-0.09	-0.09	0.18	0.14	0.19	0.05	0.05	0.09	0.07	0.05	0.03	0.04	0.04	0.06	0.02	0.18
		Light_4	-0.01	0.06	0.06	0.02	-0.03	-0.02	-0.03	-0.02	0.13	0.09	0.12	0.08	0.09	0.11	0.12	0.16	0.04	0.08	0.10	0.08	0.05	0.15
		Light_5	-0.00	0.01	0.01	0.04	0.02	0.03	0.02	0.02	0.10	0.06	0.10	0.04	0.05	0.07	0.07	0.15	0.20	0.06	0.57	0.06	0.03	0.14
		Light_6	-0.02	0.03	0.03	0.07	0.05	0.06	0.06	0.05	0.13	0.08	0.12	0.07	0.09	0.11	0.12	0.27	0.28	0.09	0.76	0.08	0.05	0.20
		Light_7	-0.01	0.10	0.10	-0.00	-0.03	-0.03	-0.03	-0.03	0.16	0.12	0.14	0.13	0.14	0.18	0.19	0.22	0.09	0.10	0.09	0.11	0.06	0.17
		Light_8	-0.03	0.07	0.07	0.01	0.01	0.00	0.01	0.00	0.07	0.06	0.06	0.04	0.04	0.05	0.06	0.11	0.25	0.05	0.14	0.04	0.02	0.11
		Light_9	-0.01	0.04	0.04	0.07	0.05	0.05	0.05	0.05	0.17	0.11	0.16	0.11	0.13	0.17	0.24	0.13	0.05	0.14	0.07	0.17	0.57	0.21
		Light_10	-0.01	0.09	0.09	-0.03	-0.04	-0.05	-0.04	-0.05	0.10	0.09	0.09	0.06	0.07	0.09	0.12	0.06	0.02	0.19	0.02	0.16	0.05	0.12
		Light_11	0.02	0.04	0.04	0.02	0.00	0.01	0.00	0.01	0.11	0.08	0.10	0.08	0.09	0.11	0.15	0.09	0.05	0.20	0.05	0.47	0.09	0.17
		Light_12	-0.01	0.08	0.08	0.06	0.03	0.03	0.03	0.03	0.14	0.10	0.13	0.11	0.12	0.16	0.20	0.12	0.06	0.27	0.07	0.66	0.11	0.23
	APT #4	Light_1	0.02	0.22	0.22	-0.10	-0.10	-0.10	-0.10	-0.09	0.25	0.22	0.20	0.00	0.05	0.14	0.07	0.06	0.09	-0.00	0.04	0.05	0.04	0.01
		Light_2	0.02	0.19	0.19	-0.10	-0.11	-0.11	-0.11	-0.10	0.32	0.20	0.30	0.00	0.08	0.18	0.07	0.07	0.08	0.00	0.03	0.08	0.03	-0.00
		Light_3	0.02	0.12	0.12	0.00	-0.04	-0.03	-0.04	-0.04	0.28	0.14	0.31	0.00	0.07	0.16	0.08	0.06	0.07	0.04	0.06	0.07	0.05	0.02
		Light_4	-0.00	0.07	0.07	-0.02	-0.02	-0.03	-0.03	-0.02	0.04	0.05	0.03	0.00	0.03	0.04	0.05	0.08	0.18	0.04	0.04	0.06	0.04	0.02
		Light_5	-0.02	0.05	0.05	0.12	0.07	0.04	0.07	0.03	0.13	0.07	0.13	0.00	0.10	0.13	0.17	0.26	0.11	0.14	0.10	0.61	0.09	0.08
		Light_6	-0.02	0.02	0.02	0.12	0.11	0.02	0.10	0.01	0.04	0.01	0.04	0.00	0.05	0.06	0.07	0.11	0.05	0.07	0.03	0.28	0.04	0.05
		Light_7	-0.01	0.04	0.04	-0.00	-0.01	-0.00	-0.01	0.00	0.07	0.03	0.06	0.00	0.07	0.08	0.06	0.05	0.03	0.01	0.03	0.03	0.05	0.02
		Light_8	-0.01	0.11	0.11	-0.06	-0.04	-0.05	-0.04	-0.04	0.06	0.06	0.06	0.00	0.05	0.08	0.12	0.20	0.08	0.23	0.10	0.20	0.11	0.04
		Light_9	-0.02	0.10	0.10	0.08	0.05	0.05	0.05	0.05	0.24	0.18	0.22	0.00	0.17	0.24	0.29	0.20	0.15	0.11	0.19	0.22	0.16	0.30
		Light_10	0.00	0.11	0.11	-0.05	-0.06	-0.05	-0.05	-0.04	0.10	0.11	0.09	0.00	0.06	0.09	0.12	0.10	0.07	0.07	0.22	0.12	0.21	0.02
		Light_11	0.02	0.08	0.08	0.07	0.06	0.07	0.06	0.06	0.10	0.10	0.09	0.00	0.06	0.09	0.12	0.09	0.08	0.08	0.28	0.12	0.68	0.04
		Light_12	0.02	0.10	0.10	0.07	0.07	0.07	0.06	0.07	0.12	0.11	0.11	0.00	0.07	0.11	0.14	0.11	0.09	0.09	0.33	0.14	0.78	0.04
	APT #6	Light_1	-0.01	0.14	0.14	-0.07	-0.11	-0.11	-0.11	-0.10	0.36	0.23	0.43	0.12	0.13	0.24	0.16	0.14	0.13	0.06	0.09	0.11	0.08	0.09
		Light_3	0.01	0.09	0.09	-0.03	-0.06	-0.05	-0.06	-0.05	0.16	0.11	0.16	0.11	0.13	0.16	0.18	0.22	0.45	0.10	0.10	0.16	0.07	0.08
		Light_4	0.01	0.06	0.06	0.02	-0.03	-0.01	-0.03	-0.01	0.23	0.13	0.25	0.18	0.21	0.25	0.32	0.40	0.19	0.27	0.21	0.79	0.22	0.16
		Light_5	0.02	0.06	0.06	-0.02	-0.05	-0.04	-0.05	-0.04	0.15	0.09	0.13	0.14	0.16	0.18	0.17	0.16	0.11	0.11	0.09	0.14	0.07	0.07
		Light_6	-0.01	0.03	0.03	-0.04	-0.05	-0.05	-0.05	-0.05	0.13	0.10	0.13	0.12	0.11	0.13	0.16	0.22	0.10	0.25	0.08	0.18	0.07	0.06
		Light_7	0.02	0.01	0.01	0.04	0.01	0.01	0.01	0.02	0.09	0.05	0.10	0.07	0.08	0.10	0.14	0.13	0.07	0.07	0.24	0.20	0.60	0.06
		Light_8	0.02	0.01	0.01	0.04	0.01	0.00	0.01	0.01	0.10	0.06	0.10	0.08	0.09	0.12	0.16	0.14	0.08	0.08	0.25	0.20	0.62	0.07
		Light_9	0.00	0.05	0.05	0.02	-0.02	-0.00	-0.02	-0.00	0.16	0.09	0.18	0.15	0.15	0.17	0.31	0.18	0.12	0.10	0.13	0.18	0.09	0.62
	APT #7	Light_1	0.04	0.04	0.04	0.03	0.00	0.03	0.00	0.03	0.09	0.05	0.01	0.05	0.02	0.02	0.03	0.03	0.02	0.03	0.00	0.04	0.00	0.06
		Light_2	0.03	0.01	0.01	0.05	0.02	0.03	0.02	0.03	0.02	0.11	0.02	0.12	0.08	0.09	0.12	0.14	0.15	0.04	0.06	0.06	0.35	0.14
		Light_3	-0.01	0.01	0.01	0.05	0.01	0.04	0.01	0.04	0.09	0.11	0.01	0.10	0.11	0.11	0.13	0.20	0.28	0.06	0.16	0.13	0.69	0.23
		Light_4	-0.02	0.00	0.00	-0.00	-0.01	-0.01	-0.01	-0.01	0.08	0.09	0.02	0.07	0.07	0.07	0.09	0.13	0.06	0.03	0.02	0.13	0.05	0.07
	APT #8	Light_1	0.00	0.08	0.08	0.02	-0.05	-0.07	-0.05	-0.06	0.17	0.09	0.12	0.06	0.06	0.10	0.07	0.06	0.03	0.06	0.06	0.07	0.02	0.02
		Light_2	0.00	0.14	0.14	-0.03	-0.07	-0.08	-0.07	-0.08	0.16	0.07	0.14	0.04	0.05	0.09	0.06	0.06	0.03	0.05	0.05	0.06	0.01	0.02

Mode	APT	Light	Day of The Week	Hour of The Day	Time of The Day	Altitude of The Sun	Global Horizontal Illuminance	Diffuse Horizontal Illuminance	Global Horizontal Irradiance	Diffuse Horizontal Irradiance	Motion Status_1	Motion Status_2	Motion Status_3	Motion Status_4	Motion Status_5	Motion Status_6	Motion Status_7	Motion Status_8	Motion Status_9	Motion Status_10	Motion Status_11	Motion Status_12	Motion Status_13	Motion Status_14
		Light_3	0.01	0.19	0.19	-0.10	-0.11	-0.12	-0.11	-0.11	0.14	0.09	0.13	0.04	0.03	0.07	0.05	0.04	0.02	0.04	0.06	0.05	0.01	0.02
		Light_4	0.00	0.10	0.10	0.07	0.03	0.02	0.03	0.01	0.06	0.04	0.06	0.03	0.03	0.05	0.07	0.10	0.21	0.05	0.05	0.09	0.03	0.01
		Light_5	0.01	0.09	0.09	0.10	0.03	0.05	0.03	0.03	0.08	0.04	0.07	0.05	0.05	0.07	0.09	0.13	0.03	0.09	0.07	0.43	0.03	0.03
		Light_6	-0.00	0.09	0.09	0.11	0.05	0.05	0.04	0.04	0.07	0.03	0.07	0.03	0.04	0.06	0.08	0.13	0.03	0.09	0.06	0.45	0.04	0.03
		Light_7	-0.01	0.06	0.06	0.05	0.00	0.00	0.00	0.00	0.07	0.05	0.06	0.05	0.05	0.08	0.13	0.12	0.06	0.09	0.08	0.10	0.02	0.02
		Light_8	0.00	0.08	0.08	0.07	0.02	0.01	0.02	0.00	0.07	0.03	0.06	0.03	0.04	0.05	0.07	0.11	0.04	0.24	0.07	0.12	0.02	0.01
		Light_9	0.01	0.02	0.02	0.08	0.06	0.06	0.06	0.05	0.05	0.03	0.05	0.02	0.03	0.04	0.08	0.03	0.02	0.02	0.04	0.03	0.01	0.24
		Light_10	0.01	0.08	0.08	0.07	-0.00	-0.01	-0.01	-0.02	0.09	0.05	0.09	0.05	0.05	0.08	0.10	0.10	0.04	0.08	0.24	0.11	0.02	0.02
		Light_11	0.00	0.03	0.03	0.10	0.07	0.08	0.07	0.08	0.03	0.03	0.03	0.02	0.02	0.03	0.06	0.08	0.05	0.03	0.03	0.07	0.47	0.02
		Light_12	0.00	0.03	0.03	0.10	0.06	0.08	0.06	0.07	0.03	0.03	0.03	0.02	0.02	0.03	0.06	0.08	0.05	0.03	0.03	0.07	0.45	0.02
		Light_13	0.01	0.21	0.21	-0.19	-0.14	-0.14	-0.13	-0.13	0.09	0.09	0.07	0.02	0.02	0.04	0.03	0.03	0.02	0.03	0.04	0.03	0.00	0.01
	APT #11	Light_1	0.04	0.01	0.01	-0.15	-0.09	-0.10	-0.09	-0.08	0.04	0.07	0.02	0.06	0.06	0.09	0.06	0.29	0.05	0.05	0.03	0.04	0.04	0.05
		Light_2	-0.02	0.02	0.02	0.04	0.01	0.03	0.01	0.02	0.20	0.16	0.18	0.17	0.19	0.20	0.20	0.08	0.15	0.07	0.17	0.02	0.75	0.02
		Light_3	0.00	0.13	0.13	-0.08	-0.11	-0.12	-0.11	-0.11	0.34	0.19	0.28	0.07	0.06	0.07	0.05	0.02	0.01	0.03	0.05	0.01	0.05	0.02
		Light_5	-0.02	0.00	0.00	0.06	0.03	0.03	0.03	0.03	0.15	0.09	0.12	0.11	0.12	0.15	0.27	0.06	0.19	0.08	0.76	0.06	0.16	0.05
		Light_6	-0.01	-0.00	-0.00	0.01	-0.01	-0.00	-0.01	-0.00	0.05	0.00	0.05	0.06	0.07	0.08	0.20	0.03	0.09	0.10	0.38	0.07	0.06	0.02
		Light_7	0.02	0.02	0.02	0.04	-0.00	0.02	-0.00	0.01	0.13	0.09	0.11	0.14	0.15	0.15	0.27	0.05	0.39	0.10	0.18	0.03	0.07	0.03
		Light_8	-0.02	0.02	0.02	0.05	0.06	0.02	0.06	0.02	0.03	0.03	0.02	0.00	0.01	0.01	0.02	0.02	0.00	0.01	0.03	0.34	0.01	0.00
	APT #12	Light_1	0.03	0.04	0.04	-0.04	-0.06	-0.05	-0.06	-0.05	0.17	0.06	0.10	0.14	0.10	0.20	0.19	0.01	0.13	0.16	0.16	0.01	0.02	0.18
		Light_2	-0.01	0.08	0.08	-0.01	-0.06	-0.04	-0.06	-0.04	0.17	0.08	0.16	0.04	0.02	0.16	0.02	0.40	0.04	-0.00	0.06	0.30	0.14	0.05
		Light_3	0.01	-0.01	-0.01	-0.08	-0.07	-0.06	-0.07	-0.05	0.11	0.03	0.07	0.10	0.08	0.12	0.05	0.06	0.07	0.08	0.08	0.01	0.50	0.11
		Light_4	-0.03	0.14	0.14	-0.04	-0.12	-0.12	-0.12	-0.12	0.15	0.20	0.17	0.01	-0.00	0.06	-0.00	0.04	0.00	-0.01	0.03	0.02	0.05	0.03
		Light_5	0.00	0.02	0.02	-0.02	-0.03	-0.02	-0.03	-0.02	0.05	0.05	0.06	-0.00	-0.00	0.03	0.00	0.02	0.01	-0.00	0.02	0.03	0.02	0.02
		Light_7	-0.07	0.01	0.01	-0.05	-0.04	-0.04	-0.04	-0.03	0.06	0.06	0.10	0.13	0.06	0.08	0.18	-0.01	0.07	0.16	0.04	-0.02	-0.00	0.10
		Light_8	0.04	0.02	0.02	0.07	-0.00	0.01	-0.01	0.01	0.16	0.08	0.15	0.18	0.09	0.16	0.22	0.02	0.28	0.22	0.69	0.03	0.03	0.08
		Light_9	0.01	0.02	0.02	0.06	-0.01	0.01	-0.01	0.01	0.14	0.04	0.21	0.09	0.16	0.13	0.24	0.15	0.26	0.37	0.67	0.17	0.04	0.06
		Light_10	0.01	0.02	0.02	0.05	0.00	0.01	0.00	0.00	0.14	0.04	0.12	0.03	0.01	0.14	0.01	0.27	0.01	-0.01	0.04	0.44	0.12	0.04
		Light_11	0.04	0.04	0.04	-0.07	-0.07	-0.07	-0.07	-0.06	0.13	0.07	0.13	0.08	0.13	0.13	0.21	-0.02	0.20	0.22	0.22	0.00	0.08	0.01
	APT #14	Light_1	0.02	0.06	0.06	-0.19	-0.11	-0.11	-0.11	-0.09	0.16	0.05	0.05	0.11	0.12	0.15	0.18	0.08	0.07	0.11	0.14	0.05	0.13	0.19
		Light_2	0.00	0.03	0.03	-0.01	-0.03	-0.01	-0.03	-0.01	0.12	0.06	0.09	0.15	0.16	0.17	0.39	0.08	0.11	0.18	0.56	0.13	0.16	0.22
		Light_3	0.01	0.01	0.01	0.02	-0.00	0.01	-0.01	0.00	0.04	0.01	0.02	0.05	0.06	0.04	0.17	0.02	0.05	0.07	0.13	0.41	0.06	0.11
		Light_4	0.01	0.01	0.01	0.05	0.01	0.02	0.01	0.01	0.04	0.00	0.02	0.05	0.05	0.06	0.23	0.01	0.06	0.07	0.19	0.66	0.04	0.15
		Light_5	0.00	0.06	0.06	-0.01	-0.01	-0.00	-0.01	-0.01	0.13	0.07	0.09	0.13	0.13	0.16	0.17	0.08	0.05	0.07	0.16	0.05	0.41	0.17
		Light_6	0.02	0.12	0.12	-0.06	-0.06	-0.06	-0.06	-0.06	0.22	0.10	0.23	0.10	0.10	0.18	0.07	0.09	-0.01	0.06	0.04	0.01	0.05	0.15
		Light_7	-0.01	0.29	0.29	-0.27	-0.18	-0.18	-0.17	-0.17	0.21	0.28	0.15	0.10	0.09	0.17	0.04	0.25	-0.03	0.03	0.02	-0.00	0.06	0.22
		Light_8	-0.01	0.18	0.18	-0.12	-0.12	-0.12	-0.11	-0.11	0.22	0.18	0.17	0.11	0.11	0.18	0.08	0.18	-0.01	0.06	0.06	0.01	0.06	0.19
		Light_9	0.01	-0.09	-0.09	-0.18	-0.09	-0.08	-0.08	-0.07	0.03	-0.00	0.00	0.08	0.09	0.07	0.19	0.02	0.29	0.08	0.15	0.05	0.11	0.17
		Light_10	-0.01	0.04	0.04	-0.03	-0.04	-0.03	-0.04	-0.02	0.10	0.05	0.06	0.12	0.12	0.14	0.21	0.07	0.05	0.43	0.16	0.05	0.07	0.14
	APT #16	Light_1	0.01	0.08	0.08	0.07	-0.05	-0.05	-0.05	-0.04	0.14	0.09	0.11	0.13	0.12	0.14	0.10	0.07	0.02	0.02	0.06	0.04	0.05	0.02
		Light_2	-0.02	0.16	0.16	0.13	-0.09	-0.07	-0.09	-0.06	0.09	0.08	0.05	0.06	0.06	0.11	0.06	0.37	0.00	-0.00	0.03	0.12	0.06	0.03
		Light_3	-0.01	0.04	0.04	0.03	0.00	0.02	0.00	0.02	0.13	0.09	0.11	0.10	0.10	0.13	0.11	0.10	0.03	0.07	0.07	0.15	0.75	0.01
		Light_4	0.01	0.18	0.18	0.15	-0.09	-0.10	-0.09	-0.09	0.25	0.18	0.21	0.06	0.07	0.15	0.06	0.05	0.01	-0.00	0.05	0.03	0.02	0.03
		Light_5	0.03	0.14	0.14	0.11	-0.04	-0.07	-0.04	-0.07	0.22	0.14	0.24	0.04	0.04	0.11	0.04	0.02	0.01	-0.01	0.04	0.02	0.02	0.02
		Light_6	-0.04	0.14	0.14	0.11	-0.07	-0.08	-0.07	-0.08	0.24	0.16	0.26	0.04	0.04	0.12	0.05	0.03	0.01	-0.00	0.04	0.02	0.03	0.02
		Light_7	-0.02	0.10	0.10	0.13	-0.07	-0.07	-0.07	-0.07	0.12	0.11	0.10	0.09	0.10	0.12	0.21	0.04	0.09	0.07	0.43	0.07	0.05	0.03
		Light_8	0.01	0.03	0.03	0.00	0.05	0.05	0.05	0.06	0.10	0.05	0.07	0.05	0.05	0.12	0.06	0.21	0.01	0.01	0.06	0.70	0.11	0.01
		Light_9	0.01	0.02	0.02	-0.01	0.03	0.05	0.03	0.06	0.09	0.03	0.07	0.05	0.04	0.10	0.04	0.18	0.01	0.01	0.06	0.59	0.09	0.01
Mode 2	APT #1	Light_2	-0.02	0.33	0.33	-0.22	-0.18	-0.18	-0.17	-0.17	0.10	0.30	0.48	0.04	0.03	0.11	0.03	0.05	0.09	0.00	0.02	0.04	0.01	0.03
		Light_3	0.00	0.26	0.26	-0.20	-0.18	-0.18	-0.17	-0.16	0.10	0.29	0.44	0.05	0.04	0.12	0.04	0.06	0.13	0.01	0.03	0.06	0.01	0.03
		Light_5	0.02	0.10	0.10	0.12	0.05	0.06	0.05	0.05	0.13	0.22	0.22	0.09	0.11	0.15	0.16	0.28	0.20	0.15	0.12	0.53	0.07	0.07
		Light_6	0.02	0.12	0.12	0.13	0.06	0.07	0.05	0.06	0.14	0.22	0.22	0.09	0.11	0.15	0.16	0.28	0.20	0.15	0.12	0.51	0.07	0.07
	APT #2	Light_1	-0.05	0.00	0.00	0.12	0.07	0.10	0.06	0.09	0.07	0.09	0.06	0.02	0.02	0.05	0.03	0.02	0.02	0.07	0.00	0.05	0.03	0.14

Mode	APT	Light	Day of The Week	Hour of The Day	Time of The Day	Altitude of The Sun	Global Horizontal Illuminance	Diffuse Horizontal Illuminance	Global Horizontal Irradiance	Diffuse Horizontal Irradiance	Motion Status_1	Motion Status_2	Motion Status_3	Motion Status_4	Motion Status_5	Motion Status_6	Motion Status_7	Motion Status_8	Motion Status_9	Motion Status_10	Motion Status_11	Motion Status_12	Motion Status_13	Motion Status_14
	APT #6	Light_2	-0.01	0.12	0.12	-0.06	-0.14	-0.13	-0.14	-0.12	0.42	0.28	0.52	0.15	0.16	0.27	0.20	0.20	0.17	0.11	0.13	0.21	0.11	0.12
Mode 3	APT #3	Light_1	-0.01	0.10	0.10	-0.03	-0.05	-0.03	-0.05	-0.03	0.26	0.13	0.19	0.01	0.06	0.11	0.08	0.08	0.04	0.06	0.08	0.13	0.05	0.02
		Light_2	-0.01	0.12	0.12	-0.08	-0.09	-0.07	-0.09	-0.07	0.28	0.12	0.46	0.02	0.09	0.15	0.12	0.11	0.06	0.09	0.06	0.21	0.05	0.02
		Light_3	0.02	0.05	0.05	0.04	0.01	0.04	0.01	0.04	0.24	0.14	0.19	0.03	0.09	0.14	0.15	0.20	0.29	0.15	0.12	0.30	0.09	0.03
		Light_4	-0.01	0.07	0.07	0.11	0.07	0.08	0.06	0.07	0.32	0.18	0.30	0.29	0.15	0.36	0.24	0.24	0.20	0.19	0.13	0.58	0.14	0.08
		Light_5	-0.02	0.06	0.06	0.08	0.06	0.06	0.06	0.05	0.33	0.18	0.37	0.27	0.16	0.36	0.24	0.25	0.17	0.18	0.14	0.59	0.15	0.09
		Light_6	-0.02	0.08	0.08	0.05	0.03	0.05	0.02	0.04	0.24	0.15	0.19	0.42	0.11	0.39	0.16	0.13	0.06	0.10	0.11	0.15	0.07	0.04
		Light_7	0.05	0.04	0.04	0.02	-0.01	0.04	-0.01	0.04	0.14	0.08	0.12	0.03	0.09	0.09	0.12	0.16	0.14	0.26	0.09	0.13	0.07	0.02
		Light_8	0.02	0.02	0.02	0.01	-0.01	0.00	-0.01	0.00	0.10	0.06	0.09	0.02	0.07	0.08	0.11	0.08	0.08	0.04	0.25	0.08	0.15	0.04
	APT #10	Light_1	-0.03	0.06	0.06	0.00	-0.02	-0.01	-0.02	0.00	0.44	0.36	0.38	0.25	0.25	0.37	0.41	0.48	0.44	0.43	0.39	0.31	0.05	0.20
		Light_2	-0.04	0.04	0.04	0.03	0.01	0.00	0.01	0.01	0.37	0.30	0.31	0.27	0.26	0.33	0.46	0.32	0.41	0.37	0.63	0.21	0.06	0.21
		Light_3	-0.02	0.09	0.09	0.01	-0.01	-0.01	-0.02	0.00	0.47	0.38	0.41	0.32	0.31	0.41	0.51	0.40	0.60	0.50	0.47	0.23	0.10	0.20
		Light_4	-0.04	0.09	0.09	-0.06	-0.05	-0.04	-0.05	-0.03	0.34	0.26	0.29	0.24	0.24	0.30	0.39	0.32	0.42	0.49	0.36	0.14	0.01	0.16
Mode 4	APT #5	Light_1	0.00	0.24	-0.12	-0.12	-0.12	-0.11	-0.11	0.36	0.36	0.33	0.15	0.17	0.30	0.18	0.12	0.10	-0.01	0.09	0.07	0.02	0.10	0.10
		Light_2	0.01	0.30	-0.14	-0.15	-0.16	-0.15	-0.15	0.48	0.41	0.54	0.21	0.24	0.37	0.27	0.23	0.15	0.08	0.15	0.21	0.08	0.20	0.32
		Light_4	-0.01	0.09	0.08	0.00	0.03	0.00	0.03	0.31	0.16	0.28	0.20	0.23	0.29	0.29	0.37	0.72	0.11	0.18	0.31	0.11	0.15	0.23
		Light_5	0.00	0.07	0.02	-0.01	0.00	0.00	0.00	0.22	0.12	0.24	0.14	0.17	0.21	0.22	0.29	0.20	0.13	0.14	0.39	0.12	0.16	0.18
		Light_6	0.00	0.06	0.04	-0.04	-0.03	-0.04	-0.03	0.39	0.20	0.40	0.25	0.30	0.38	0.43	0.52	0.33	0.35	0.36	0.88	0.29	0.27	0.40
		Light_7	0.02	0.06	-0.02	-0.03	-0.04	-0.03	-0.05	0.12	0.08	0.10	0.13	0.14	0.14	0.14	0.12	0.08	0.01	0.08	0.08	0.02	0.05	0.05
		Light_8	0.02	0.03	0.02	-0.01	0.00	-0.02	-0.01	0.13	0.07	0.13	0.08	0.11	0.13	0.14	0.19	0.14	0.12	0.10	0.13	0.05	0.07	0.09
		Light_9	-0.03	0.11	0.01	-0.02	-0.01	-0.02	-0.01	0.38	0.27	0.37	0.24	0.28	0.35	0.42	0.28	0.20	0.16	0.34	0.31	0.29	0.82	0.33
		Light_10	0.00	0.10	-0.10	-0.09	-0.08	-0.08	-0.07	0.21	0.15	0.20	0.18	0.21	0.22	0.29	0.19	0.12	0.13	0.43	0.21	0.35	0.18	0.20
		Light_11	-0.03	0.03	0.05	0.02	0.03	0.02	0.03	0.21	0.12	0.20	0.14	0.17	0.23	0.28	0.20	0.12	0.12	0.45	0.29	0.72	0.21	0.21
		Light_12	-0.04	0.04	0.04	0.02	0.02	0.02	0.02	0.26	0.15	0.25	0.14	0.19	0.27	0.34	0.22	0.13	0.15	0.53	0.31	0.84	0.28	0.26
	APT #9	Light_1	0.02	0.18	0.18	-0.09	-0.10	-0.11	-0.10	-0.10	0.13	0.07	0.08	0.12	0.12	0.15	0.13	0.11	0.08	0.10	0.11	0.07	0.06	0.03
		Light_2	0.02	0.13	0.13	-0.04	-0.05	-0.05	-0.05	-0.04	0.08	0.05	0.05	0.10	0.09	0.16	0.11	0.39	0.08	0.12	0.10	0.29	0.06	0.03
		Light_3	0.02	0.08	0.08	0.05	0.03	0.02	0.02	0.02	0.13	0.07	0.11	0.15	0.14	0.17	0.16	0.12	0.11	0.08	0.08	0.06	0.76	0.04
		Light_4	0.01	0.34	0.34	-0.20	-0.17	-0.18	-0.17	-0.17	0.21	0.14	0.16	0.04	0.03	0.06	0.02	0.04	-0.02	0.08	-0.01	0.00	0.03	-0.02
		Light_5	0.02	0.24	0.24	-0.13	-0.14	-0.14	-0.14	-0.14	0.27	0.06	0.28	0.02	0.02	0.04	0.01	0.02	-0.03	0.01	-0.01	-0.01	0.03	-0.01
		Light_6	0.01	0.19	0.19	-0.10	-0.11	-0.11	-0.10	-0.11	0.21	0.04	0.23	0.01	0.00	0.02	0.00	0.00	-0.03	0.00	-0.02	0.00	0.02	-0.01
		Light_7	-0.01	0.07	0.07	0.06	0.04	0.05	0.04	0.05	0.06	0.06	0.05	0.10	0.10	0.14	0.29	0.14	0.16	0.10	0.62	0.14	0.06	0.08
		Light_8	0.01	0.05	0.05	0.10	0.07	0.09	0.07	0.08	0.06	0.06	0.05	0.10	0.09	0.13	0.26	0.13	0.12	0.08	0.54	0.12	0.04	0.07
		Light_9	0.02	0.31	0.31	-0.19	-0.13	-0.14	-0.13	-0.12	0.00	0.07	-0.02	0.05	0.04	0.05	0.09	0.10	0.15	0.26	0.06	0.06	0.04	-0.01
		Light_10	-0.01	0.06	0.06	0.08	0.06	0.08	0.06	0.08	0.08	0.03	0.07	0.10	0.08	0.16	0.13	0.33	0.07	0.08	0.16	0.66	0.06	0.08
		Light_11	-0.02	0.00	0.00	0.09	0.04	0.07	0.04	0.07	0.05	0.02	0.06	0.07	0.06	0.11	0.10	0.26	0.06	0.06	0.14	0.55	0.03	0.07
	APT #13	Light_1	0.08	0.08	-0.04	-0.06	-0.06	-0.06	-0.06	0.11	0.09	0.08	0.10	0.11	0.12	0.09	0.03	0.07	0.03	0.04	0.03	0.05	0.02	0.02
		Light_2	0.14	0.14	0.01	-0.03	-0.02	-0.03	-0.02	0.14	0.11	0.09	0.07	0.08	0.18	0.07	0.27	0.07	0.05	0.05	0.23	0.05	0.04	0.17
		Light_3	0.06	0.06	0.02	-0.01	0.00	-0.01	-0.01	0.11	0.09	0.08	0.10	0.11	0.13	0.10	0.07	0.07	0.06	0.07	0.07	0.56	0.02	0.14
		Light_4	0.26	0.26	-0.15	-0.16	-0.17	-0.16	-0.16	0.23	0.24	0.19	0.09	0.09	0.14	0.07	0.01	0.08	0.03	0.04	0.06	0.05	0.03	0.21
		Light_5	0.13	0.13	-0.06	-0.08	-0.09	-0.08	-0.09	0.19	0.11	0.21	0.04	0.05	0.10	0.03	0.00	0.04	0.01	0.03	0.03	0.03	0.02	0.13
		Light_6	0.05	0.05	0.03	0.01	0.01	0.00	0.01	0.07	0.06	0.05	0.08	0.09	0.09	0.17	0.03	0.08	0.06	0.69	0.05	0.05	0.04	0.11
		Light_7	0.02	0.02	0.02	0.00	0.00	0.00	0.00	0.02	0.02	0.01	0.03	0.03	0.03	0.08	0.01	0.03	0.03	0.38	0.03	0.02	0.00	0.06
		Light_8	0.15	0.15	0.00	-0.03	-0.03	-0.03	-0.03	0.17	0.15	0.13	0.14	0.16	0.16	0.20	0.04	0.34	0.04	0.09	0.06	0.05	0.03	0.18
		Light_9	0.05	0.05	0.08	0.04	0.04	0.04	0.03	0.11	0.09	0.07	0.05	0.06	0.14	0.06	0.18	0.04	0.04	0.06	0.65	0.05	0.05	0.16
		Light_10	0.05	0.05	0.07	0.02	0.02	0.01	0.01	0.11	0.08	0.06	0.05	0.06	0.15	0.06	0.20	0.05	0.06	0.05	0.78	0.06	0.05	0.17
	APT #15	Light_1	-0.01	0.22	0.22	-0.15	-0.14	-0.14	-0.13	-0.13	0.28	0.23	0.25	0.20	0.20	0.24	0.16	0.05	0.05	0.12	0.07	0.10	0.06	0.04
		Light_2	-0.01	0.13	0.13	-0.03	-0.05	-0.07	-0.05	-0.07	0.16	0.11	0.14	0.13	0.12	0.20	0.10	0.21	0.06	0.06	0.08	0.25	0.06	0.02
		Light_3	0.00	0.06	0.06	0.01	-0.01	-0.01	-0.01	-0.01	0.13	0.08	0.12	0.12	0.12	0.16	0.13	0.09	0.03	0.07	0.06	0.12	0.61	0.01
		Light_4	0.00	0.23	0.23	-0.15	-0.12	-0.13	-0.12	-0.12	0.27	0.27	0.22	0.09	0.09	0.15	0.08	0.01	0.02	0.07	0.03	0.04	0.02	0.02
		Light_5	0.07	0.13	0.13	-0.08	-0.09	-0.07	-0.09	-0.06	0.21	0.18	0.19	0.07	0.07	0.12	0.07	0.01	0.01	0.05	0.04	0.05	0.02	0.02
		Light_6	0.00	0.21	0.21	-0.18	-0.17	-0.18	-0.16	-0.17	0.33	0.27	0.32	0.12	0.11	0.18	0.11	0.04	0.03	0.10	0.06	0.12	0.06	0.02
		Light_7	0.06	0.00	0.00	-0.03	-0.03	-0.02	-0.03	-0.02	0.07	0.04	0.06	0.09	0.09	0.08	0.16	0.04	0.14	0.07	0.39	0.06	0.05	0.02
		Light_8	0.00	0.10	0.10	0.02	-0.03	-0.03	-0.03	-0.02	0.19	0.10	0.16	0.13	0.12	0.24	0.12	0.27	0.05	0.10	0.12	0.79	0.10	0.03

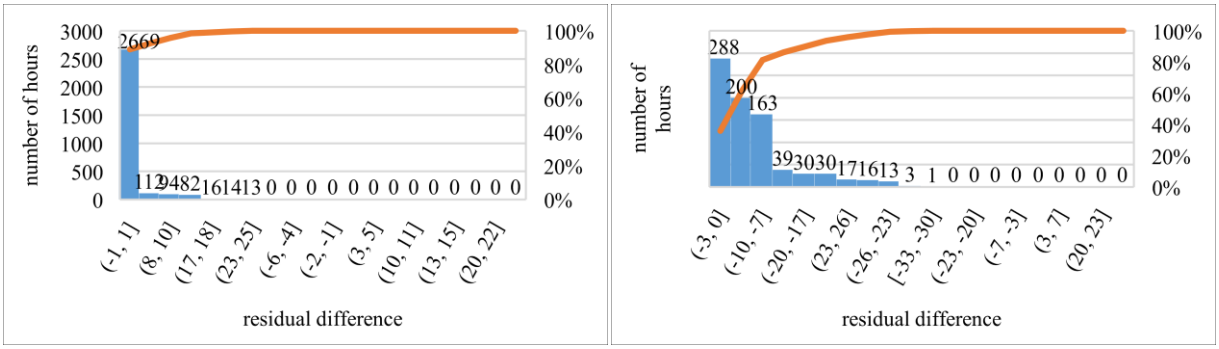
Mode	APT	Light	Day of The Week	Hour of The Day	Time of The Day	Altitude of The Sun	Global Horizontal Illuminance	Diffuse Horizontal Illuminance	Global Horizontal Irradiance	Diffuse Horizontal Irradiance	Motion Status_1	Motion Status_2	Motion Status_3	Motion Status_4	Motion Status_5	Motion Status_6	Motion Status_7	Motion Status_8	Motion Status_9	Motion Status_10	Motion Status_11	Motion Status_12	Motion Status_13	Motion Status_14
		Light_9	-0.01	0.02	0.02	0.01	-0.01	-0.01	-0.01	0.00	0.09	0.03	0.07	0.06	0.05	0.12	0.06	0.13	0.03	0.06	0.07	0.44	0.03	0.03
Mode 5	APT #5	Light_3	0.00	0.02	-0.10	-0.06	-0.08	-0.05	-0.06	0.04	0.05	0.04	0.03	0.02	0.03	0.02	0.01	0.01	0.00	0.02	0.01	0.03	0.01	0.03
	APT #11	Light_4	0.03	0.02	0.02	-0.01	0.01	-0.03	0.01	-0.03	-0.02	-0.02	-0.01	-0.04	-0.03	-0.03	-0.03	-0.01	-0.02	-0.02	0.00	-0.02	0.00	-0.02
	APT #12	Light_6	0.02	0.01	0.01	-0.19	-0.13	-0.11	-0.12	-0.10	0.00	0.02	0.01	0.05	0.06	0.02	0.05	-0.02	0.06	0.12	0.03	0.10	-0.01	0.03

APPENDIX C – FEATURE SELECTION FOR DATA-DRIVEN MODELS USED BY MPC IN TASK C

To select the proper kernel function and input features for the SVM energy prediction model, we trained the candidate kernel functions and input features based on the same training dataset and compared the predictive performance on a validation dataset. The training dataset contains 3000 samples and the validation dataset includes 800 samples. Candidate kenrnel functions include 'linear', 'poly', 'rbf'. Candidate input features include $T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$,$T_{ambient,i-1}$, and $\widehat{Q_{i-1}}$.

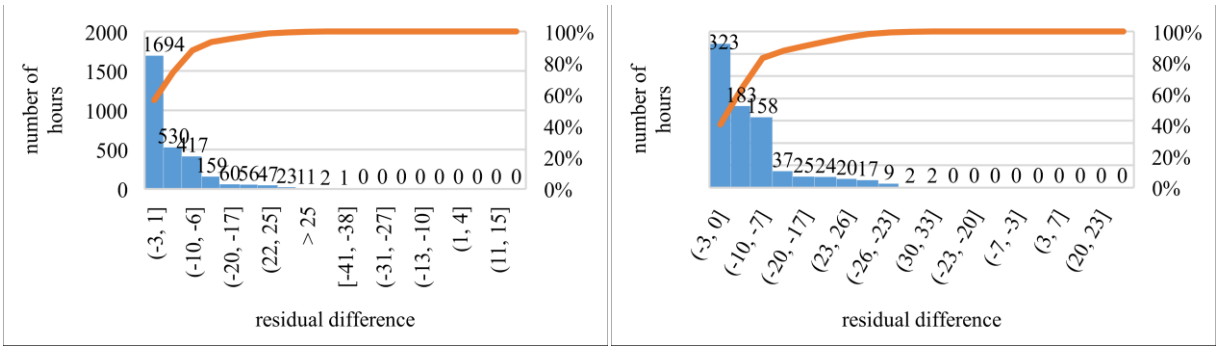
Through comparing the energy predictive result among model training and validation (see figures given in this section), 'linear' is selected as the kernel function of SVM, while $T_{set,i}$, $T_{ambient,i}$,and $T_{ambient,i-1}$are selected as the input features. Note that the result of SVM with 'rbf' function is not given in this section, because its predicted result on validation dataset is a constant and there is no need to present the meaningless result.

Similary, for the model that predicts whether or not the indoor air temperature during one period is lower than 21°C, we tested the same candidate features and models. The predictive result is given in Table C1. From this table, SVM with 'linear' kernel function is selected as the data-driven model for indoor air temperature prediction, while $T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$, and $T_{ambient,i-1}$ are input features.



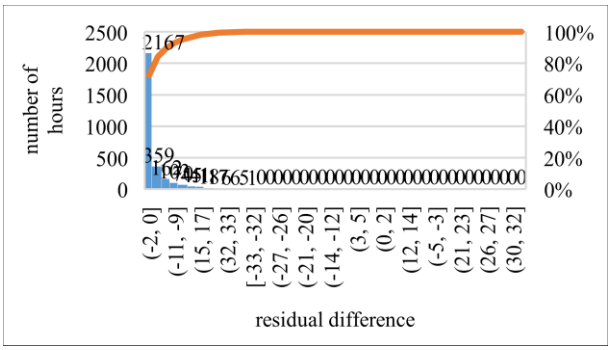
(a) Model training (b)Model validation

Figure C1: Energy predictive result of SVM with 'poly' kernel with inputs $T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$,$T_{ambient,i-1}$, and $\widehat{Q_{i-1}}$

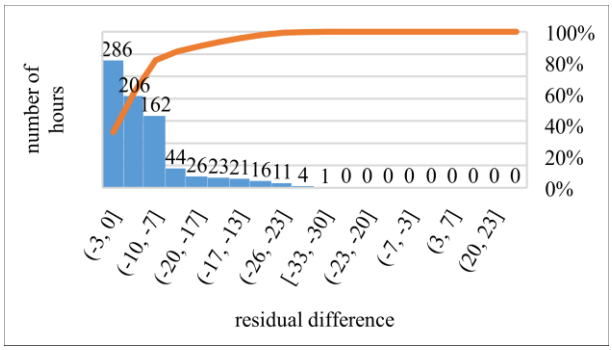


(a) Model training (b)Model validation

Figure C2: Energy predictive result of SVM with 'poly' kernel with inputs $T_{set,i}$ and $T_{ambient,i}$

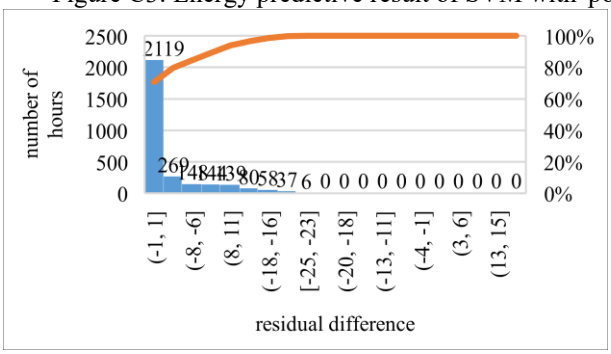


(a) Model training

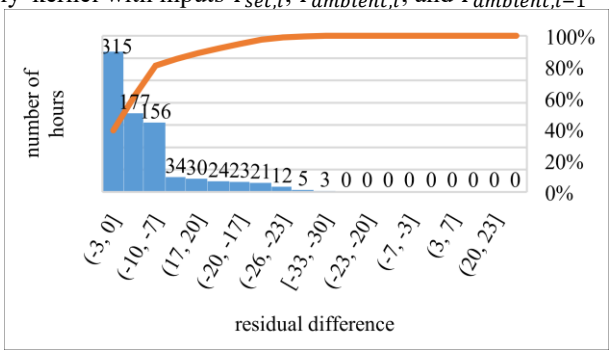


(b)Model validation

Figure C3: Energy predictive result of SVM with 'poly' kernel with inputs $T_{set,i}$, $T_{ambient,i}$, and $T_{ambient,i-1}$

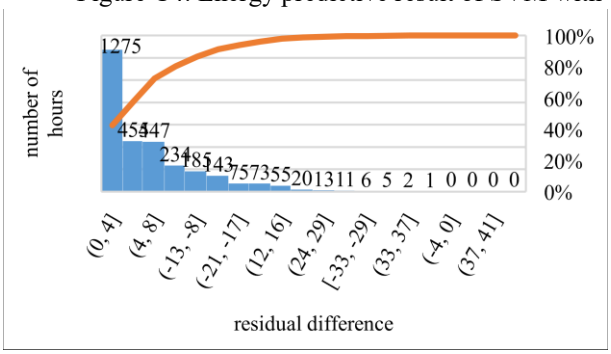


(a) Model training

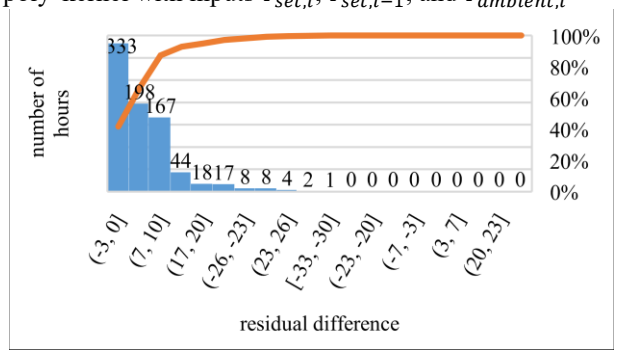


(b)Model validation

Figure C4: Energy predictive result of SVM with 'poly' kernel with inputs $T_{set,i}$, $T_{set,i-1}$, and $T_{ambient,i}$

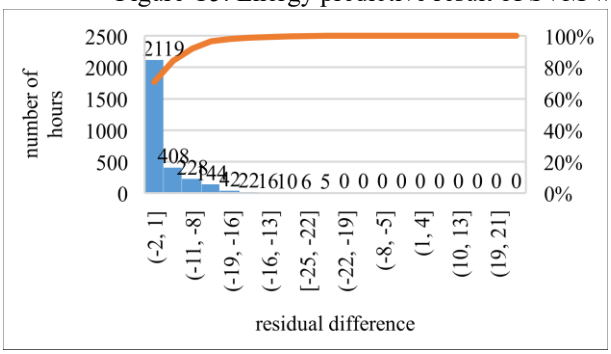


(a) Model training

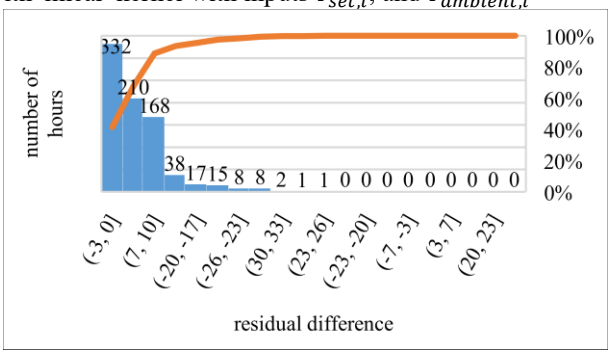


(b)Model validation

Figure C5: Energy predictive result of SVM with 'linear' kernel with inputs $T_{set,i}$, and $T_{ambient,i}$

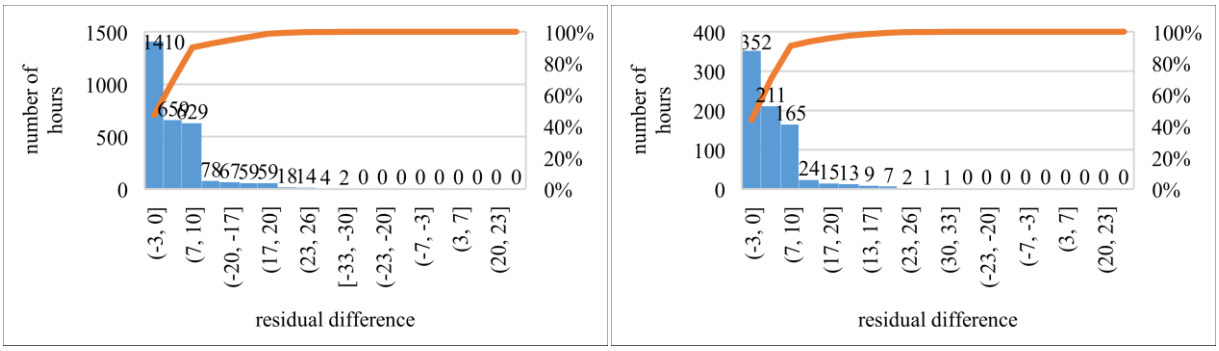


(a) Model training



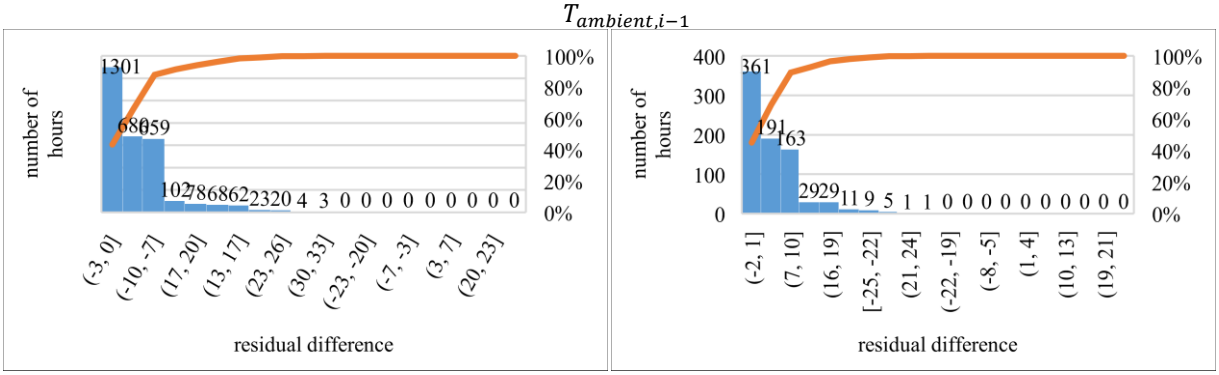
(b)Model validation

Figure C6: Energy predictive result of SVM with 'linear' kernel with inputs $T_{set,i}$, $T_{set,i-1}$, and $T_{ambient,i}$



(a) Model training (b)Model validation

Figure C7: Energy predictive result of SVM with 'linear' kernel with inputs $T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$, and $T_{ambient,i-1}$



(a) Model training (b)Model validation

Figure C8: Energy predictive result of SVM with 'linear' kernel with inputs $T_{set,i}$, $T_{ambient,i}$, and $T_{ambient,i-1}$

Table C1: Air temperature predictive result of different SVM models

Kernel	**Input features**	**Model training**			**Model validation**		
		Accuracy	**Recall**	**Specificity**	**Accuracy**	**Recall**	**Specificity**
Linear	$T_{set,i}$ $T_{ambient,i}$	0.79	0.87	0.64	0.76	0.83	0.62
	$T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$	0.82	0.88	0.71	0.79	0.85	0.68
	$T_{set,i}$, $T_{ambient,i}$, $T_{ambient,i-1}$	0.80	0.87	0.66	0.78	0.84	0.65
	$T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$, $T_{ambient,i-1}$	0.83	0.89	0.72	**0.82**	**0.88**	**0.70**
poly	$T_{set,i}$, $T_{ambient,i}$	0.81	0.88	0.68	0.76	0.83	0.63
	$T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$	0.88	**0.92**	0.81	0.75	0.81	0.64
	$T_{set,i}$, $T_{ambient,i}$, $T_{ambient,i-1}$	0.85	0.90	0.74	0.75	0.82	0.60
	$T_{set,i}$, $T_{set,i-1}$, $T_{ambient,i}$, $T_{ambient,i-1}$	**0.90**	**0.92**	**0.85**	0.79	0.84	0.69