Detection and diagnosis of multiple dependent faults in HVAC systems using machine learning techniques

Behrad Bezyan

A PhD research proposal

In the Department

Of

Building, Civil and Environmental Engineering

Presented in Partial Fulfilment of the Requirements

for the Degree of

Doctor of Philosophy (PhD) (Building Engineering) at

Concordia University

Montreal, Quebec, Canada

March 2022

© Behrad Bezyan, 2022

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Behrad Bezyan

Entitled: Detection and diagnosis of multiple dependent faults in HVAC systems using machine learning techniques

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. Anjali Awasthi

_____ Thesis Supervisor
Dr. Radu Grigore Zmeureanu

_____ Examiner
Dr. Ramin Sedaghati

_____ Examiner
Dr. Bruno Lee

_____ Examiner
Dr. Mohamed Ouf

_____ External Examiner
Dr. Alan S. Fung

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

«March 14th, 2022» _____
Dr. Mourad Debbabi , Dean

Abstract

Detection and diagnosis of multiple dependent faults in HVAC systems using machine learning techniques

Behrad Bezyan, Ph.D.
Concordia University, 2022

The building sector accounts for about 40% of the total annual energy consumption in the United States and 25% in Canada. Therefore, it is so essential to design and operate energy-efficient smart buildings. About 15 to 30% of the energy in the commercial buildings would be wasted if the heating, ventilation, and air conditioning (HVAC) systems are not maintained regularly, or they are inappropriately controlled, and if the system degradation has not been detected at early stages. Therefore, the building performance should be monitored in real-time using the Building Automation System (BAS), in order to detect any potential fault in the system and diagnose the sources of malfunction in the HVAC systems.

Machine learning (ML) models which are developed from BAS trend data without information about the physical system, proved in the past, good performance for analysis the linear and non-linear systems. Therefore, ML models due to their capacity for distinguishing the faulty from normal operation are applied in this thesis for multiple dependent fault detection and diagnosis (MDFDD). In this thesis, ML models are proposed for MDFDD of sensors in an air handling unit (AHU) of an institutional building located in the Concordia University campus.

Two approaches are proposed with application of experimental and synthetic data sets: 1) combination of machine learning models with rule-based technique, 2) classification machine learning models.

1) ML models (e.g., Support vector regression (SVR)), developed from building automation system (BAS) trend data, predict air temperature of two target sensors, under normal operation conditions without known problems. The fault symptom is detected when the residual of measured and predicted values exceed the threshold. The recurrent neural network (RNN) models predict the normal operation values of regressor sensors, which are compared with measurements, as the first step for the identification of fault symptoms. Rule-based models are used for fault diagnosis of sensors or equipment.

2) The optimized classification ML models (e.g., shallow artificial neural network (ANN), deep ANN, K-Nearest Neighbor (KNN), decision tree classification, random forest classification, support vector machine (SVM), Naïve Bayes, and principal component analysis) are developed for MDFDD. ML algorithms parameters are optimized over RandomizedSearch method by varying the length of training dataset, input time lags, and relevant parameters of each ML models. Results from the two data set types of an existing building show the high quality of proposed method for the detection and diagnosis of the multiple dependent faults.

Acknowledgements

At first, I would like to thank my supervisor, Dr. Radu Zmeureanu, for his precious supervision and advice during my PhD program that I could obtained so many worthful experiences through my research. His valuable supervision, comments, and advice led me to accomplish each step successfully and could improve my research skills.

I would like to appreciate my parents, Shahin, Alimohammad for their deep supports during my studies and their encouragements to carry on my studies through the best way in order to overcome the hard situations.

Most importantly, I would like to appreciate my wife, Elnaz who is the love of my life and made my life more beautiful than ever, and she always standing by my side with her deep support to give me this courage for starting and finishing each step peacefully and successfully.

Moreover, I would like to thank my brother, Yashar who consulted and supported me on every stage of my PhD thesis and encouraged me to keep learning and improving my skills.

Also, I would like to thank my colleagues in the research lab and my friends with their supports and all of the productive discussions we had.

I would like to thank Natural Sciences and Engineering Research Council of Canada (NSERC) and Faculty of Engineering and Computer Science of Concordia University for their financial supports.

Table of Contents

Table of Figures x

Table of Tables xii

List of Acronyms xv

Nomenclature xvii

1. Introduction 1

1.1. Problem statement 1

1.2. Motivation and applications 1

1.3. Expected contributions to the advancement of knowledge 2

2. Literature review 3

2.1. Quantitative models (physics-based models) 5

2.2. Qualitative model (Rule-based model) 7

2.3. Process-history based models (data-driven models) 8

2.3.1. Feature selection 11

2.3.2. Main variables for FDD and typical common faults 15

2.3.3. Threshold definition for FDD 17

2.3.4. Dataset selection techniques for model development 18

2.3.5. Machine learning models for FDD 19

2.3.6. Deep learning models 27

2.3.7. Research including the comparison of different models for MFDD 31

2.3.8. Genetic algorithm (GA) 32
2.3.9. Summary of research publications for single FDD 35
2.3.10. Summary of research for Multiple FDD 35
2.4. Summary of reviewed literature for FDD methods 36
2.5. Conclusions on findings and gaps 38
3. Research method 41
3.1. Method # 1: The combination of machine leaning model for prediction with rule-based technique 43
3.2. Machine learning models 50
3.2.1. Support vector regression (SVR) 50
3.2.2. Artificial neural network (ANN) 52
3.2.3. Recurrent neural network (RNN) 54
3.3. Rule-based model for fault diagnosis 56
3.3.1. Group A of rule-based models 56
3.3.2. Group B of rule-based models 57
3.4. Method # 2: The classification machine learning models 58
3.5. Artificial faults generation using grey-box models 60
3.5.1. LSM for obtaining grey-box model coefficients 61
3.5.2. Genetic algorithm method for grey-box model development 61
4. Case study 63

4.1. Data preprocessing and optimization of training data sets for model development 66

4.2. Results from the artificial faults generation 68

4.2.1. Results of LSM for finding grey-box model coefficients: 68

4.2.2. Results of GA for finding grey-box models coefficients 71

4.2.1. LSM versus GA methods for finding the models coefficients 71

5. Application of proposed methods 72

5.1. Application # 1: Method # 1 applied on experimental and synthetic data 72

5.1.1. Optimization of input data set 72

5.1.2. Development of machine learning models 74

5.1.3. Comparison of results between different ML models 79

5.1.4. Comparison of ML models results on experimental and synthetic data..... 81

5.1.5. Results and discussion on fault detection and diagnosis 82

5.1.6. Comparison with another method..... 90

5.2. Application # 2: Method # 2 applied on experimental and synthetic data 91

5.2.1. Development of classification machine learning models 91

5.2.2. Results and discussion on fault detection and diagnosis on experimental and synthetic data..... 94

6. Conclusions, contributions, and thesis limitations..... 103

6.1. Conclusions 103

6.2. Contributions 105

6.3. Thesis limitations 106

7. Future works 107

8. References 108

9. Appendices 123

9.1. Appendix A 123

9.2. Appendix B 130

9.3. Appendix C 133

9.3.1. Python codes for the SVR model 133

9.3.2. Python codes for the Random Forest model 138

9.3.3. Python codes for the RNN model 141

9.3.4. Python codes for the classification machine learning models model 144

Table of Figures

Figure 2.1. The proportion of FDD methods out of 197 publications [15] 5

Figure 2.2: The proportion of FDD methods out of 123 publications [15] 9

Figure 2.3: The proportion of FDD methods out of 110 publications [15] 9

Figure 3.1. Schematic of Air Handling Unit from experimental data 42

Figure 3.1. Schematic ML model for prediction of target variables T_ma and T_ahc at time t for fault symptom detection..... 44

Figure 3.2. MDFDD diagram..... 45

Figure 3.3. Schematic RNN model for prediction of regressor sensor X at time t for fault symptom detection..... 46

Figure 3.4. Recurrent neural network with LSTM algorithm [126] 54

Figure 3.5. ML model architecture for the MDFDD 59

Figure 3.6. Impact of Pc on RMSE of the developed model of T_ma using GA 62

Figure 3.7. Impact of Pc on RMSE of the developed model of T_ahc using GA 62

Figure 4.1. Genomic building [159] 63

Figure 4.2. Schematic of Air Handling Unit from experimental data 65

Figure 4.3. Measurements with artificial faults and predictions of T_ma. 69

Figure 4.4. Measurements and predictions of T_oa with artificial faulty data 70

Figure 4.5. Measurements and predictions of T_ra with artificial faulty data. 70

Figure 5.1. Training and testing data sets with sliding window technique from January 13 to January 21 for the prediction of T_ma and T_ahc. 74

Figure 5.2. Measurements versus predictions of T_ma over testing data set under normal operation conditions..... 76

Figure 5.3. Measurements and predictions of T_ahc over testing data set under normal operation conditions 77
Figure 5.4. Residual between measurements and predictions of T_ma including artificial faulty data. 83
Figure 5.5. Residual between measurements of T_ma including faults and predictions of T_ma 84
Figure 5.6. Measurements with artificial faults of T_ma and predictions of normal operation of T_ma 85
Figure 5.7. Residual between measurements of T_ma and predictions of T_ma 85
Figure 5.8. Residual between measurements and predictions of T_oa using RNN model..... 86
Figure 5.9. Residual between measurements and predictions of T_ra using RNN model 87
Figure 5.10. Residual between measurements of T_ma and predictions of T_ma using RNN model..... 91

Table of Tables

Table 2.1: Common faults in the AHU [21, 67-69] 15

Table 2.2: Common faults in the chillers 16

Table 2.3: Confusion matrix for fault detection 22

Table 2.4: Strengths and weaknesses of FDD methods 37

Table 3.1. List of AHU variables from the experimental data 43

Table 4.1. List of AHU variables from the experimental data 66

Table 4.2. Grey-box models for prediction T_ma and T_ahc using LSM 68

Table 4.3. Grey-box models development for prediction of T_ma and T_ahc using GA 71

Table 5.1. Prediction performance of SVR model of T_ma over training data set of three days and average testing results over the next six days. 75

Table 5.2. Prediction performance of SVR models of T_ahc over training data set of seven days and testing over the next day. 76

Table 5.3. Prediction performance of Recurrent Neural Network for selected regressors sensors using the experimental data 78

Table 5.4. Results for the performance of the other ML models for the prediction of T_ma and T_ahc, using 288 data points (3-days) of training data (experimental data) 79

Table 5.5: Results for the performance of the other ML models for the prediction of T_ma and T_ahc, using including 288 data points (3-days) of training data (synthetic data) 80

Table 5.6: Predictions of the Recurrent Neural Network using synthetic data 80

Table 5.7: Feedforward Neural Networks prediction results using synthetic data. 81

Table 5.8: Confusion matrix for fault detection of T_ma 83

Table 5.9. Accuracy, precision, and sensitivity of the SVR models for detecting artificial faults of T_ma. 83
Table 5.10: Confusion matrix for fault detection of T_oa 88
Table 5.11: Confusion matrix for fault detection of T_ra 88
Table 5.12. Accuracy, precision and sensitivity of the RNN models for MDFDD. 89
Table 5.13. Prediction performance of Recurrent Neural Network for selected regressors sensors 90
Table 5.14: Confusion matrix for fault detection of T_ma 94
Table 5.15: Confusion matrix for fault detection of T_oa 95
Table 5.16: Confusion matrix for fault detection of T_ra 95
Table 5.17. Accuracy, precision, and sensitivity of the SVM models for detecting artificial faults of T_ma. 95
Table 5.18.Accuracy, precision and sensitivity of the ML models over the testing data for T_ma on experimental data 96
Table 5.19: Confusion matrix for fault detection of T_oa 97
Table 5.20: Confusion matrix for fault detection of T_ra 97
Table 5.21: Confusion matrix for fault detection of T_ma 97
Table 5.22. Accuracy, precision and sensitivity of the SVM models for MDFDD. 98
Table 5.23. Accuracy, precision and sensitivity of the ML models over the testing data set (experimental data) 98
Table 5.24: Accuracy, precision and sensitivity of the ML models using synthetic data over the testing data set 100

Table 5.25. Accuracy, precision and sensitivity of the ML models using experimental data over the testing data set 101

Table 9.1. Data-driven models for single faults detection and diagnosis applications in HVAC systems 123

Table 9.2. Data-driven models for multiple faults detection and diagnosis applications in HVAC systems 130

List of Acronyms

Name	Definition
AHU	Air handling unit
ANN	Artificial neural network
BAS	Building automation system
BEMS	Building energy management system
CART	classification and regression tree
CNN	Convolutional neural network
DANN	Deep artificial neural network
DL	Deep learning
DT	Decision tree
FDD	Fault detection and diagnosis
FFNN	Feedforward neural networks
FN	False negative
FP	False positive
GA	Genetic algorithm
GAN	Generative adversarial network
HVAC	Heating, ventilation and air conditioning
KNN	K-Nearest Neighbor
LDA	Linear discriminant analysis
LR	Linear regression
LSM	Least square error method
LSTM	Long short-term memory
MAPE	Mean absolute percentage error
MBE	Mean Bias Error
MFDD	Multiple fault detection and diagnosis
MDFDD	Multiple dependent fault detection and diagnosis
ME_max	Maximum Absolute Error
ML	Machine learning

MLP	Multi-layer perceptron
MLR	Multiple linear regression
PCA	Principal component analysis
PSO	Particle swarm optimization
QDA	Quadratic discriminant analysis
R^2	Coefficient of determination
RF	Random forest
RMSE	Root mean squared error
RNN	Recurrent neural network
SFS	Sequential feature selection
SLR	Single linear regression
SVC	Support vector classification
SVDD	Support vector data description
SVM	Support vector machine
SVR	Support vector regression
TN	True negative
TP	True positive
VAV	Variable air volume

Nomenclature

Name	Definition
α	Ratio of the outdoor mass air flow rate to mixed mass air flow
N	Number of hidden layer neurons in ANN
σ	Standard deviation
μ	Average
$\Delta T\_s, fan$	Air temperature rise over supply fan
ΔT_ac	Air temperature difference over coil
Valve_HC	Heating coil valve opening position
V_ra	Return air volumetric flow rate
V_oa	Outdoor air volumetric flow rate
V_ma	Mixed air volumetric flow rate
T_sa	Supply air temperature
T_ra	Return air temperature
T_oa	Outdoor air temperature
T_ma	Mixed air temperature
T_HW_S	Supply hot water temperature
T_ahc	Air temperature after heating coil
Q_HC	Electric input for heating coil
ρ_{ra}	Return air density
ρ_{oa}	Outdoor air density
ρ_{ma}	Mixed air density
ρ_{ahc}	After heating coil air density
y_i	The measured value
$\bar{y}$	The average target value
$x_{j,m}^{t}$	The measured regressor value at time (t)
Y_p^t	The target value at time (t)
X_p^t	Predicted regressor value at time (t)
$X_{normalized}$	Normalized value

X_m^{t-i}	Measured value at previous time steps
N_{Output}	Number of outputs variables in ANN
N_{Input}	Number of inputs variables in ANN
$C_{p,ra}$	Return air specific heat
$C_{p,oa}$	Outdoor air specific heat
$C_{p,ma}$	Mixed air specific heat
$C_{p,ac}$	After heating coil air specific heat
$\hat{y}_i$	The predicted value
$\overline{x^t}$	The average values of regressors

1. Introduction

1.1. Problem statement

The building sector accounts for about 25% of the total annual energy consumption in Canada (Canada Energy Regulator - Canada`s Energy Future 2017) [1] and 40% in the United States (U.S. Renewable Energy Data Book 2016) [2]. Therefore, it is so essential to design and operate energy-efficient smart buildings. About 15 to 30% of the energy in the commercial buildings would be wasted if the heating, ventilation, and air conditioning (HVAC) systems are not maintained regularly, or they are inappropriately controlled, and if the system degradation has not been detected at early stages [3]. Therefore, the building performance should be monitored in real-time using the building energy management system (BEMS), in order to detect any potential fault in the system and diagnose the sources of malfunction in the HVAC systems. The development of an efficient artificial intelligence-based model for detecting and diagnosing the multiple dependent faults (MDFDD) in the HVAC system components in a commercial building is the main issue of this research project.

1.2. Motivation and applications

Evaluation of the monitored data from sensors in the HVAC systems of buildings will help to detect and diagnose the multiple dependent faults in the system, which result in energy waste and degradation of indoor thermal comfort before complete failure in the system and occupant discomfort. Early MDFDD increases the energy efficiency of the HVAC system, improve indoor air quality, comfort, reduce the energy loss, and reduce the greenhouse gas emission [4].

A method was proposed for MDFDD with the application of machine learning techniques in an HVAC system of an institutional building during heating season, located in the Concordia University campus, Montreal, Canada. Machine learning, a subfield of artificial intelligence (AI) domain, helps learning from data to generate data-driven models to detect and diagnose simultaneous multi-faults of AHU sensors. These models give the computer ability to learn patterns without being explicitly programmed [5].

Most researchers covered the fault detection and diagnosis (FDD) of a single faulty sensor or multiple non-dependent faults. This thesis focuses on one solution of a more general problem that is the detection and diagnosis of the multiple dependent sensors' faults using machine learning models. Machine learning models, which are developed from building automation system (BAS) trend data without information about the physical system, proved in the past good performance for linear and non-linear systems [3] and [6].

1.3. Expected contributions to the advancement of knowledge

1- Development of a novel efficient and practical sequential machine learning models which have not been studied so far according to the reviewed literatures for the prediction of a target sensors' variables in an AHU for the detection and diagnosis of the multiple dependent faults in the sensors of an AHU in a commercial building.

2- Development of rule-based technique including a set of rules for the diagnosis of the main sources of detected faults in the system.

3- Development of the novel classification-based machine learning, deep learning (DL) for detection and diagnosis of the multiple faulty sensors in the AHU of an institutional building.

4- Comparison of the developed model for the MDFDD using the experimental data collected by the BAS, and the synthetic data extracted by eQuest program.

2. Literature review

Approximately 25% and 40% of the total annual energy use in Canada and the United States belongs to the building sector [1-2]. The operation of the heating, ventilation, and air conditioning (HVAC) systems is normally monitored, but the potential offered by building automation systems (BAS) trend data for the fault detection and diagnosis (FDD) is still not fully exploited, in spite of the extensive research over last decades. If HVAC systems are not maintained regularly or they are inappropriately controlled, and if the system faults and degradation are not regularly detected, around 15 to 30% of the energy in the commercial buildings is wasted [7].

The literature review has not discovered publications about the automated detection and diagnosis of multiple faults (MFDD) in HVAC systems, where one fault can have an impact on one or more other faults. This is still a challenging problem, since the combination of several faults makes difficult the separation of individual faults [8]. Three examples are listed herein:

(i) One fault has a positive or negative impact on another fault.

(ii) Two faults occur, but their combined effect is not observed on the third sensor, which indicate normal operation, and

(iii) Two faults occur, but only the effect of one fault on the third sensor is observed.

The building automation systems (BAS) are deployed to monitor the energy performance of the buildings and record HVAC system operation in real-time, then the collected databases are used for evaluation the operation of system equipment.

Evaluation of the sensors values in the HVAC systems of buildings will help to detect and diagnose the multiple faults in the system, which results in energy waste and degradation of indoor thermal comfort before complete failure in the system and occupant discomfort. Early MFDD increases

the energy efficiency of the HVAC system, improve indoor air quality, comfort, reduce the energy loss, and reduce the greenhouse gas emission [4].

Faults in the HVAC systems are divided into two categories: 1) hard faults, which occur suddenly and result in system shut down, and also it will result in high economic losses; one example is the fan shut down because of belt cut; and 2) soft faults, which are the malfunction in the system and result in degradation in the system performance while the system still operates. Some examples for soft faults are sensors issues, dampers issues, and coils fouling. Some of the soft faults are not obvious to be detected and diagnosed easily without a well-developed model [9].

The fault detection and diagnosis methods are categorized into three groups [7, 10-11]: 1) quantitative model-based, 2) qualitative model-based, and 3) process history-based models.

Quantitative models are the physics-based models of system operation and known as the white-box model, the model is described in Section 2.1.

Qualitative models are the knowledge-based models for the operation of the system and follows a series of expert rules obtained from mass and energy balances in order to detect any potential fault, and this model is explained in Section 2.2. In addition, control signal is used to specify the system operation mode [12].

Process history-based models use historical measured data of the system operation which are monitored by the sensors in the HVAC system [13]; they are also called the data-driven models, which can be classified into supervised and unsupervised learning-based models, as described in Section 2.3. The process history-based models which are based on the monitored historical data of the system, are more popular due to less complexity in developments [14-15].

Out of 197 reviewed publications by [7, 10, 15] for fault detection and diagnosis (FDD), 62% were process history-based, 26% were qualitative-based models, and 12% were quantitative-based models (Figure 2.1). The process history-based models are more popular due to their robustness on FDD, simplicity in developments without knowing the physical system information, and applicability on non-linear systems.

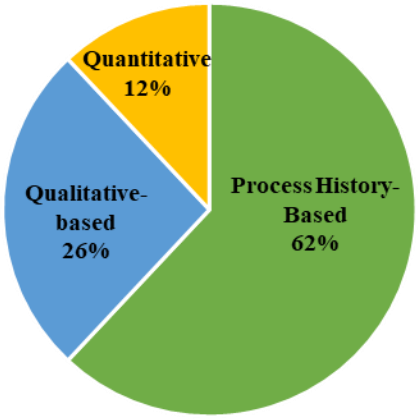

Figure 2.1. The proportion of FDD methods out of 197 publications [15]

2.1. Quantitative models (physics-based models)

Physics-based models are also named white box models, analytical, or first principal models. These are mathematical models of processes happening in the system which can represent the system operation [3, 16-17]. These models can predict the thermal environment in a space due to the building envelope, weather conditions (e.g., radiation, outdoor temperature, wind velocity), and HVAC operating conditions [3].

For instance, the supply air temperature in an air handling unit can be calculated using a physics-based model. The hourly variables that have an impact on the supply air temperature are monitored, then the hourly supply air temperature is calculated. The white-box model can be used for

providing a baseline value of supply air temperature. The new measured values are compared with the reference value, and if the difference between measurement and reference values exceed the defined threshold, then a potential fault is detected.

Researchers developed physically-based models using collected measured and synthetic data sets for detection and diagnosis of the multi-faults in the chiller, fan coil unit, and also the whole HVAC systems. The target faults for chiller are, condenser fouling, reduced condenser and evaporator water flow rate, non-condensable gas in the refrigerant, refrigerant leak/undercharge and overcharge [18]. For the fan coil unit, the common faults are recirculation damper stuck, cooling coil fouling, supply fan speed malfunction for the whole HVAC system [19], fan, heating/ cooling coils and control valve, filters, air damper, and temperature sensors [20].

Other researchers combined the physically-based models with process-history based models such as SVM [21], Bayesian network [22-24], and also the rule-based modes [25-26] to develop hybrid models for the MFDD in the air handling unit (AHU) and chillers. The target for MFDD in the AHU were fan failure, duct leakage, return/ outdoor/ mixing air damper leakage or stuck, heating/ cooling coil fouling, valve reverse, sensors (air temperature) errors, heating/cooling valve opening faults or leakage, heat recovery pump. For chillers the targets were, compressor error, valve fault, reduction in condenser and evaporator water flow rate.

The reviewed literature revealed the physically-based models have acceptable accuracy, flexibility to capture steady and transient operation, while these models have some drawbacks, such as; complexity in the models development, intensive computations, and if there is not enough information of the system operation, the estimation of the target variable may not be accurate. Therefore, these are some of the reasons that this method has the least popularity for the MFDD.

2.2. Qualitative model (Rule-based model)

Rule-based models use a series of rules derived from experts' knowledge, and from energy and mass balance equations. Researchers developed rule-based models for the fault detection and diagnosis in the HVAC system components [27-33]. The rule-based models were combined with other physically-based [25-26, 34] and process history-based models, such as decision tree [35], Bayesian network [36-38], principal component analysis (PCA) [39], to develop the hybrid models for the FDD of the mentioned targeted faults. Rules-based models can be developed without proper understanding and information about physical processes in HVAC systems. The literature review revealed the limitations of this method. Main limitations of these models are due the extraction of specific rules for a given system, the export to other systems, and the addition of new rules when situation requires.

Some of the target faults out of the literature review are: sensors (air temperature, air flow) error, cooling/ heating coil valve leakage, damper stuck, mechanical cooling fault, too high airflow setpoint, slipping supply fan drive belt, undersized supply duct, disconnected zone temperature setpoint, zone temperature control loop tuning error, undersized VAV box [27-33]. The rule-based models were combined with other physically-based [25-26, 34] and process history-based models, such as decision tree [35], Bayesian network [36-38], principal component analysis (PCA) [39], to develop the hybrid models for the MFDD of the mentioned targeted faults. Beside the strengths of the rule-based model, the literature review revealed the limitations of this method. The limitation of rule-based models is the development and application of the specific rule-based models on a specific system, which means the model cannot be assured to be used in other systems

for the MFDD purposes. The knowledge is required in order to change and set new rules if the new operation and faults are discovered.

2.3. Process-history based models (data-driven models)

The process history-based models are classified into two groups: 1) black-box, and 2) grey-box models. The black box models are developed only based on the historical data without any knowledge on the physical models and operation of the system. The grey-box model is developed with combination of physical-based models (first-principal) of the system and historical data in order to estimate the model parameters and predict the system performance. The grey-box model can be developed using least square error method or genetic algorithms (GA). Some studies used grey-box models for the MFDD using the measure and synthetic data sets in the sensors and actuators [40-41].

The black-box models can be classified into different models: 1) statistical models (e.g. polynomial regression, principal component analysis (PCA)), 2) artificial neural networks (ANNs), and 3) pattern recognition (e.g. support vector machine (SVM)).

Out of 110 studies on black-box models for FDD, 63% were based on statistical models, 25% case studies applied ANNs, and 12% used other pattern recognition techniques [15] (Figure 2.3).
Out of 123 publications using process history-based methods, 72% developed black-box models for FDD, 12% applied grey-box models, and the rest of 16% of studies applied hybrid methods which are combination of black-box or grey-box with quantitative or qualitative-based models [15] (Figure 2.2).

The grey-box models are developed with the combination of white-box (physics-based models) and measurements. The hybrid models are a combination of, for example, black-box + grey-box, black-box + rule-based, black-box + physics-based.

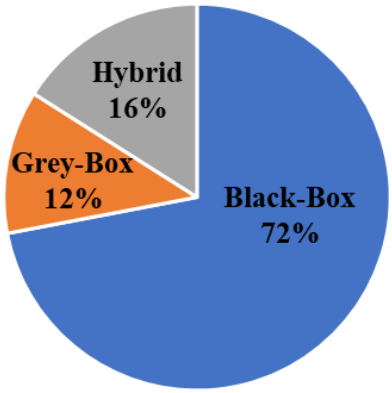

Figure 2.2: The proportion of FDD methods out of 123 publications [15]

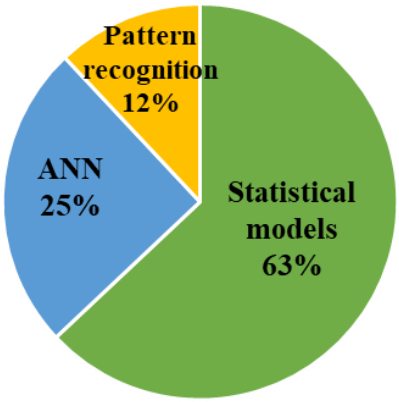

Figure 2.3: The proportion of FDD methods out of 110 publications [15]

Machine learning, which is a data-driven based method in the artificial intelligence field, gives the computer the ability to learn without being explicitly programmed [5] and learns the patterns of data to develop an algorithm for prediction purposes of the patterns [42].

The process-history based model is the other technique for MFDD in the HVAC systems. Nevertheless, for the development of high accurate models, a large and reliable pre-processed data set is required [43], but, these models have the positive points which most of the times will overcome the drawbacks. The models can be well developed with just input and output data set of the system without knowing the physical system information. Their simplicity for development, good performance, and their applicability for linear and non-linear systems are the other strengths. Based on these reasons and the results revealed from the reviewed literatures, the process history-based models are placed as the most popular models for the MFDD in the HVAC systems.

The process history-based models are separated into supervised and unsupervised learning algorithms. The supervised based models are trained with the labeled targets, and then the developed model is applied on the new data set for prediction purposes, which can be applied for MFDD purposes. If the difference between the measurement in an HVAC system equipment and the prediction of that value using the supervised-learning model (residual) exceeds the defined threshold, a fault is detected. Unsupervised learning-based models do not include the labeled targets; therefore, the model is trained based on the pattern of the data set and system operation, then different clusters are generated and labeled as the group of same performance. Then the developed model is applied on the new data set to predict the new data points are located into which cluster.

The literature review section mainly focuses on the application of data-driven models for detection and diagnosis of faults in the HVAC systems.

Development of a data-driven model for the fault detection and diagnosis in the HVAC system has various linked steps which need to be considered and implemented by a researcher to obtain an accurate and reliable model.

2.3.1. Feature selection

In the data-driven based models for supervised and unsupervised learning algorithms, the influencing features (variables) which impact the output of the target variable are selected as the inputs in the models in order to predict the target values. The selection of relevant features helps improving the prediction model performance and reduction in the computational time.

Principal component analysis (PCA) [44] is a technique for dimension reduction. This method transforms the observations (measurements) of j-variables a reduced set of k-variables ($k < j$), which are named the Principal Components (PC). In other words, the observations of j-variables are projected into a k-dimensional PC-based space. Previous studies [44-47] revealed the number of PCs should be selected in a way in order to explain at least 75% to 90% of minimum cumulative variance of the initial data set. In order to select the desired principle components (features), there are different techniques to apply as described in the following subsections.

There are three common feature selection methods which are used for the development of the process history-based models [48] such as, 1) Filter, 2) Wrapper, and Embedded methods.

2.3.1.1. Filter methods

This method applies the ranking of variables by the ordering of principal variables as the ranking methods are used due to their good performance and simplicity. A ranking criterion is deployed to evaluate the variables scores and a threshold is set to remove the variables which are less than the threshold, and the less relevant variables are removed, and the most important ones are kept [49]. There are two ranking criterions for filter method, 1) correlation criteria, 2) mutual information.

2.3.1.1.a. Correlation criteria

A method for feature selection is the cross-correlation analysis which the Pearson correlation coefficient (R(i)) is calculated [50-51]. Cross-correlation analysis is a linear method for evaluation the linear impact of one variable on another variable [52], which can be calculated also using a tool (function) in MATLAB [53] or Python [54]. Using cross-correlation analysis, the variables which have high impact on prediction of another variable are selected as the input regressors for development of the prediction models. The correlation coefficient is calculated with Equation (2.1).

$$R(i) = \frac{\sum_j (Y_p^t - \bar{y})(x_{j,m}^t - \overline{x^t})}{\sqrt{\sum_j (Y_p^t - \bar{y})^2 \sum_j (x_{j,m}^t - \overline{x^t})^2}} \tag{2.1}$$

where: Y_p^t = target value at time t

$x_{j,m}^t$ = the regressor value at time t

$\overline{y^t}$ = the average values of target and regressors within the specific time-set

$\overline{x^t}$ = the average values of target and regressors within the specific time-set

The meaning of correlation coefficients and the classification are defined as represented below [1]:

- R(i) = 1.0, completely linear correlation
- R(i) ≥ 0.8, high correlation
- 0.5 ≤ R(i) < 0.8, moderate correlation
- 0.3 ≤ R(i) < 0.5, low correlation
- R(i) < 0.3, no correlation

The correlation coefficients reveal positive or negative correlation between two variables. Positive correlation coefficient indicates impact of one variable on another variable. As an example, the correlation coefficient between mixed air temperature and outdoor air temperature has the value of 0.99, which means that when outdoor air temperature increases, the mixed air temperature increases and reverse.

2.3.1.1.b. Mutual information

In this information theoretical ranking method, the dependency between two variables of input and target are evaluated [49-51, 56-57].

As on of the strengths of ranking features is their less computationally intensive and preventing the overfitting and performs well for different data sets. The filter methods do not contingent upon the learning algorithms.

2.3.1.2. Wrapper methods

These methods rely on the classification for subset feature selection. The Wrapper methods are classified into two algorithms: 1) Sequential Feature Selection (SFS) Algorithms, and 2) Heuristic Search Algorithms.

Sequential Feature Selection (SFS) Algorithms:

In the forward SFS algorithms, the features are added in a subset heuristically in order to find the maximum accuracy of the objective function. A backward SFS can also be developed for feature selection which in this method, the total numbers of variables are set in the subset and the variables which has the less impact on the target prediction are removed from the subset. A criterion is set

for searching of the subsets in order to reach the maximum performance with the minimum number of features. In another word the heuristic approach evaluates different features selection to optimize the objective function [58-59].

Heuristic Search Algorithms:

The Genetic Algorithm (GA) [60] and Particle Swarm Optimization (PSO) [61] are the algorithms for the Heuristic Search Algorithms in order to find the optimum subsets, for reaching the maximum prediction performance of the objective function. In this approach, which GA and PSO can be found, an evolutionary algorithm is deployed on the data set to obtain the best search result and the consequently, the best model performance.

The main negative point of Wrapper method is the high number of computations which is required to obtain the subset of feature.

2.3.1.3. Embedded methods

In these methods the feature selection is conducted over the algorithm development, which means, these methods are embedded in the algorithm and their aims are reduction in computation time for the classification of the subsets which are conducted in the Wrapper methods [48, 50, 62-63].

As the most common embedded methods, different sorts of decision tree algorithms such as CART, random forest [64] can be mentioned. The embedded methods evaluate the feature weights on the basis of models' regularization using the objective functions which minimize the errors of fitting and meanwhile forces the feature coefficients to be small. These methods are based on Lasso [65] or Elastic Net [66] which deploys the linear classifiers such as SVM.

2.3.2. Main variables for FDD and typical common faults

The main variables which are required to be monitored for FDD in the AHU and chillers as the main components in the HVAC systems of all building types are represented in Table 2.1 and Table 2.2, respectively. Moreover, in these tables the lists of common typical faults in the AHU and chillers are reported, with reference to [21, 67-69].

Table 2.1: Common faults in the AHU [21, 67-69]

Component	Faults
Sensors	- (positive/ negative bias or frozen) - Air temperature sensor reading error - Air flow rate sensor error - Water flow rate sensor error - Air pressure sensor error
Heating and cooling coils	- Heating/ cooling coil valve stuck open - Low/ high hot water temperature or low/ high boiler performance - Heating/ cooling coil valve stuck closed - Low/ high chilled water temperature or low/ high chiller performance - Hot/ chilled water coil fouling - Heating/ cooling coil leakage
Dampers	- Outdoor air damper stuck open/ close or partially open
Fans	- Supply/ return fan stuck at constant rate - Supply/ return fan complete failure - Belt slippage
Duct	- Duct leakage
Pumps	- Water circulation pump fault

Table 2.2: Common faults in the chillers

Faults
- Condenser fouling - Excess oil - Reduced water flow rate in condenser - Reduced water flow rate in evaporator - Non-condensable in the refrigerant - Refrigerant leakage - Refrigerant undercharge - Refrigerant overcharge

The MFDD of sensors in the HVAC systems is a topic currently under development using the process history-based models which utilizes BAS trend data as one technique [70]. According to Annex 34 [43], the reliability of the MFDD models relies on the reliability of the measurements.

In order to detect and diagnose simultaneous multi-faults in the AHU sensors, the ML models as powerful techniques are applied.

Artificial Neural Network (ANN), and support vector machine models account for the supervised learning-based models. Unsupervised learning-based models do not include the labeled targets; therefore, the model is trained based on the pattern of the data set and system operation. Clustering, principal component analysis are the examples of unsupervised learning-based models.

Researchers have developed and applied ML models using Python [54] and MATLAB [53] for the MFDD in various buildings components and HVAC systems, such as air handling unit, chiller, fan coil unit, solar hot water system, variable refrigerant flow, packaged rooftop unit, ground source heat pump (GSHP), photovoltaic panels and etc. Also, the critical reviews regarding the

application of artificial intelligence-based methods for FDD in the building energy systems were presented by [13]. The proposed machine learning models were developed using Scikit learn [71], Keras [72] and Tensorflow [73] which are the open source libraries in Spyder Python 3.8 [54].

Some of the conducted research using ML models for MFDD in the HVAC components are critically reviewed in this thesis.

2.3.3. Threshold definition for FDD

Thresholds are defined as a measure for discriminating and detecting the faults from the normal performance of the system. For an instance, the fault will be detected in a sensor if the difference between measured and expected values exceed the defined threshold. This threshold may be set based on different approaches such as using the sensor uncertainty for defining a threshold. There are different methods for thresholds development as explained below:

I. **Heuristic methods:** the available data from the system operation are used to determine suitable steady-state detection thresholds. The values of the threshold are assigned based on expert knowledge out of historical data from the similar HVAC system or from the same system at different times.

~~II.~~ **Statistical methods:** these methods are based on confidence interval and hypothesis testing using estimates of the means and standard deviations of the parameters. Training data is used to estimate the means and standard deviations. The obtained values are combined in order to define a threshold including upper and lower boundaries, and the values beyond these boundaries are detected as the faults.

III. **User selection:** in this method the user can modify alarm generation threshold in order to gain suitable rates for faults and fault sensitivity.

The threshold can be set based on the sensor uncertainty. The sensor's overall uncertainty includes fixed and random (bias) errors, which indicates the sensor reading errors, and always the measurements are in effect by a level of uncertainty [74]. For an instance, the overall uncertainty of an air temperature sensor is ± 0.49°C, which includes 0.45°C as the fixed bias and 0.019°C for the random bias as reported in a study by [75].

2.3.4. Dataset selection techniques for model development

The training data set for the development of data-driven models such as supervised and unsupervised ML models, is an important factor to obtain an accurate and reliable model for the prediction purposes of a target variable over a desired testing set, and consequently, the fault detection scope. Therefore, for training the model, three methods for training data set selections have applied by researchers: 1) static, 2) accumulative, and 3) sliding window techniques.

1) **Static window:** Model is trained using a selected data set, then the developed model is applied for predicting new data sets without retraining the model.

2) **Accumulative window:** this technique, which is also known as augmented window technique, is a training method which uses the data sets for every single training day for prediction of the following test set, and then the training data set is increased with the previous elapsed days for training. The training data set is augmented by every desired data set, and the model is retrained with the new training set.

3) **Sliding window:** this technique uses a constant training data set size, then the training data set size is shifted forward by adding the new day data set and removing the first old day data set, then the model is retrained with the new data set.

2.3.5. Machine learning models for FDD

The ML models are categorized into two approaches: 1) regression machine learning models, and 2) classification machine learning models, which are summarized in the following sub-sections. These techniques are used for the prediction of target values of the variables, and consequently the fault detection and diagnosis purposes.

2.3.5.1. Regression machine learning models

In the regression machine learning models, the target variable is predicted in order to generate a reference value, and the monitored values are compared with this reference values at the same time step. A fault is detected when the difference between measured and the predicted values exceeds the pre-defined threshold. Some of the regression machine learning models which also applied in different case studies are simple/ multiple linear regression, polynomial regression, logistic regression, random forest regression, decision tree regression, support vector regression, K-Nearest Neighbor, shallow and deep artificial neural network, recurrent neural network.

2.3.5.2. Prediction performance of regression models

The performances of the prediction models in the regression machine learning models are evaluated with statistical indices Equations (2.2) to (2.5)): Root-Mean-Squared-Error (RMSE) [76], Mean Absolute Percentage Error (MAPE), Mean Bias Error (MBE), and Maximum Absolute Error (ME_{max}) [76] [77]. The lower these statistical indices are calculated; the higher accuracy of the model will be.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{2.2}$$

$$\mathrm{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| . 100 \tag{2.3}$$

$$\mathrm{MBE} = \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)}{n} \tag{2.4}$$

$$\mathrm{ME}_{max} = |\hat{y}_i - y_i| \tag{2.5}$$

where: $\hat{y}_i$ = the predicted value

y_i = the measured value

$\bar{y}$ = the average measured values of the variable within the selected period

n = the number of data points

2.3.5.3. Classification machine learning models

In classification models, the target is labeled as normal or faulty operation. The classification models of ML are developed using normal and faulty operation data sets over the training set, then, the developed model is applied for fault detection and diagnosis of simultaneous multi-faults in the testing data set. The correct sensor reading is labeled as 0, while the faults are labeled as 1. The proposed classification-based ML models which were developed and applied in different studies are shallow and deep ANN, K-Nearest Neighbor (KNN) [78], decision tree classification, random forest classification [79], support vector machine (SVM), kernel SVM, support vector regression (SVR), Naïve Bayes, and hybrid approaches of PCA with other ML models.

2.3.5.4. Classification models and fault detection and diagnosis performance

The performance of the MDFDD models are evaluated by using the accuracy, precision, sensitivity (recall), and F-score (F1, which is which is the harmonic average of precision and sensitivity) measures (Equations (2.6) to (2.9)) [80-82]. Accuracy measures the number of correct predictions of faulty and normal readings, respectively, over the total number of predictions. Precision measures the number of correct predictions of faults out of all predicted faulty readings. Sensitivity measures the number of faults predictions out of all actual faults.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100\% \tag{2.6}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\% \tag{2.7}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\% \tag{2.8}$$

$$\text{F1} = \frac{2.\text{TP}}{2.\text{TP} + \text{FP} + \text{FN}} \times 100\% \tag{2.9}$$

where: TP = number of true positives, e.g., the number of correctly predicted faults

FP = number of false positives, i.e., the incorrectly predicted faults

FN = number of false negatives, e.g., the incorrectly labelled data as no-fault

TN = number of true negatives, e.g., the correctly labelled readings as no-fault

The confusion matrix (Table 2.3) represents the definition of the TP, TN, FP, and FN.

Table 2.3: Confusion matrix for fault detection

		Actual faults	
		Negative (0)	Positive (1)
Predicted faults	Negative (0)	Correctly predicted no-fault (TN)	Incorrectly predicted no-fault (FP)
	Positive (1)	Incorrectly predicted fault (FN)	Correctly predicted fault (TP)

2.3.5.5. Supervised learning methods

An artificial neural network (ANN) is a machine learning technique that is used for prediction and forecasting purposes [83-84]. Many researchers used ANN models for the prediction of variables or indices of performance in building HVAC systems. Such models can be used to predict the reference or typical operating performance that is useful for fault detection. If the difference between the measurement in an HVAC system and ANN prediction exceeds the defined threshold, a fault is detected. Based on the reported results of the publications, ANN models perform very well for the prediction of variables and MDFDD purpose in the HVAC systems [85-87].

Some researchers have focused on the development of different ANN algorithms, such as single, auxiliary, and backpropagation ANN using MATLAB and Python for FDD. They studied optimization of the input time lags and the number of hidden neurons for prediction or forecasting sensor values of the HVAC system or building energy use. ANN was introduced by Lee et al. [88] for AHU FDD. For another instance, forecasting the electrical use in commercial buildings by [89] which can be considered as a reference line and detect a fault once the residual between the monitored and predicted values exceed the threshold. In order to select the important input features,

a sensitivity analysis was conducted in order to select the variables as the inputs which have the most impact on the accuracy of predictions [90].

Elnour et al. [91] developed auto-associative neural network for the detection and diagnosis of single and multiple faults in the air temperature sensors of and AHU. Their study revealed good performance of the proposed model for the single and multiple FDD, but the proposed model cannot be used for the simultaneous faults in the sensors and actuators.

Two hyperparameters, such as the number of hidden layers and number of hidden layer neurons, were considered in a deep neural network model for fault detection and diagnosis (FDD) by [92]. They concluded that increasing the network size does not improve the fault detection accuracy more than a specific range.

Geyer et al. [93] proposed a component-based approach using ANN for prediction of energy performance in heating and cooling modes. They used the physical properties of building components as the inputs in each ANN of components in order to predict the conduction heat transfer rate. Furthermore, they used the summation of the outputs along with other input parameters as the input into the next ANN for the prediction of heating and cooling loads. This work can be used for simultaneous multi-fault detection and diagnosis.

Support vector machine (SVM) is a supervised machine learning tool [94-96], through which, a hyperplane is developed to separate the classified features over the selected data set. Some of studies using SVM for MFDD in the HVAC systems are reported in Table 9.1. Some examples of the conducted research are provided concisely here. Liang et al. [97] generated some faults using simulation software for recirculation damper stuck, cooling coil fouling/block, and supply fan speed increasing, and then applied SVM for MFDD. They found SVM classification is effective

with high accuracy for MFDD, and revealed supply air temperature, mixed air temperature, outlet water temperature of cooling coil, and control signals are sensitive to the faults, therefore, these variables can be selected as the fault indicators.

Yan et al. [98] developed a multi-class SVM for detection of multiple faults in an AHU and applied to different faulty samples in the training set during winter and summer seasons. Their results revealed the maximum MFDD accuracy of 88.96% and 92.53% over the summer and winter seasons, respectively.

Han et al. developed SVM model for multi-fault detection in centrifugal chillers and revealed 95% accuracy for the MFDD [99]. In their other studies, they developed hybrid SVM models combined with genetic algorithm (GA) for the MFDD. They revealed 2% increase on the accuracy with application of the developed hybrid model compared with the regular SVM model [100-101]. Support vector regression (SVR) is another machined learning model which comes from the SVM for regression-based purposes [102] and has bee applied in some studies for FDD scopes.

Bayesian network (BN) is another type of ML model. This model makes the relation between faults and symptoms and works with the probabilities of the event happening with consideration of the effects and causes obtained by the sensor records and expert knowledge [103-104]. Some of the publications which used the BN for the single and multiple FDD are summarized in the appendix. For instance, a study was developed by Cai et al. [105] for MFDD on the ground source heat pump (GSHP) with deploying the sensors of temperature and pressure in the condenser, evaporator, and compressor.

Decision tree with classification and regression tree (CART) algorithm as another supervised machine learning technique have been used for MFDD purposes, as this method can be understood

and interpreted for prediction purposes and therefore, it`s reliability will be validated with testing data and expert knowledge. Using the CART algorithm, the decision tree is developed with portioning the data space recursively and fitting the prediction model in each partition, and CART can be used for both classification and regression approaches [106-107]. In an implemented study by [108], the decision tree (CART) model was developed for MFDD of the typical AHU faults such as heating coil valve leakage, cooling coil valve stuck, return fan fixed speed or complete failure, outdoor/ exhaust air dampers stuck at fully closed, and duct leakage of AHU before the supply fan. Results revealed the accuracy of 97% for the MFDD.

2.3.5.6. Unsupervised learning methods

The application of the unsupervised learning-based models such as clustering and PCA for MFDD of the HVAC system components are summarized in the appendix, in Table 9.1 and Table 9.2 in addition to other ML models. The PCA method is a commonly used tool for dimension reduction [44] and feature extraction [109] which is used widely for ML models development for the MFDD purposes in the AHU [110-111].

Subtractive clustering analysis is an unsupervised machine learning technique which the center of the data set density is found at the first stage, then a threshold is defined. The data points out of these clusters are diagnosed as the faults. The applied clustering method revealed its capability in fault diagnosis in the HVAC system. Through fault diagnosis, a library is developed, and the developed model diagnose that this it is faulty or normal [112].

Fault detection and variables identification for the ongoing commissioning of chillers in an institutional building was conducted using PCA method by Cotrufo and Zmeureanu [113] for MFDD. In the proposed model an ellipsoid is developed as the threshold in order to distinguish

normal operation (data set inside the threshold) from faulty operation which the data points are located outside of the ellipsoid threshold. The proposed PCA-based model was applied in another case study by Bezyan and Zmeureanu [114] for MFDD in space heating and domestic hot water systems of semi-detached houses and they revealed the proposed model has a well performance in MFDD of the HVAC systems.

Simple PCA model and hybrid PCA models combined with other ML techniques have been applied in some studies for MFDD. PCA based method combining with the wavelet transform method was used for MFDD in an AHU by [111]. The wavelet is a technique which is applied for pre-treatment of data sets in order to differentiate the measurement fluctuations which comes after the faults. The conclusions revealed wavelet-PCA method performs well in MFDD in an AHU.

Lian et al. [115] developed a hybrid model which is the combination of PCA with SVDD for MFDD in the centrifugal chiller, and compared with traditional PCA, and SVDD. The SVDD is a classification algorithm which can be used for fault detection and operation monitoring. The concept is the spherical shape as the boundary threshold for separation the normal and faulty data points [116-117].

The hybrid models which are the combination of different ML models have been applied for the MFDD purposes in the HVAC systems. Furthermore, deep learning models such as deep ANN, recurrent neural networks, and convolutional neural networks have been developed and applied for the multi-fault detection and diagnosis in the HVAC systems.

2.3.6. Deep learning models

Deep learning models are the other techniques which have been applied recently for FDD in the HVAC systems. Deep artificial neural network (DANN), recurrent neural network (RNN), and convolutional neural network (CNN) are the deep learning algorithms which have been applied recently for MDFDD in the HVAC systems.

2.3.6.1. Deep artificial neural network (DANN)

Deep artificial neural network (DANN) as a deep learning model is an ANN algorithm which consists of two and more hidden layers. Lee et al., [118] developed a DANN model including five hidden layers, and 200 hidden neurons in each layer using the generated data set from EnergyPlus and revealed 95.16% accuracy in MFDD in AHU. Two hyperparameters; hidden layer, and hidden neurons impact on the performance of the DANN model which was studied by [119] and revealed the increase on the network size does not enhance the prediction performance above a certain level. However, increasing the size of data set can be considered as a key point for improvement in fault detection accuracy.

A sensitivity analysis was conducted in order to select the variables as the inputs which have the most impact on the accuracy of predictions [90]. Two hyperparameters, such as the number of hidden layers and number of hidden layer neurons, were studied in a deep neural network model for fault detection and diagnosis (FDD) by [92]. They also concluded that increasing the network size does not improve the fault detection accuracy at more than a certain achieved level.

Guo et al. [120] developed an optimized DANN model with two hidden layers and conducted a grid search for finding the optimum number of hidden layer neurons proposed by [121-124] for

detection and diagnosis of the faults in a variable flow refrigerant system, and they revealed that the developed model has 97.7% accuracy for MFDD.

2.3.6.2. Recurrent neural network (RNN)

The recurrent neural network (RNN) as another deep learning method which includes long-short-term memory (LSTM) architecture is developed for MFDD in the HVAC system using simulated and real measured databases [125]. RNN uses the previous sequential information to learn and predict the present values based on the trained model. LSTM architecture has the chain like structure of the neural networks and able to learn the long-term dependencies. The LSTM is capable of adding, storing and removing the information which are helpful for the prediction [126].

In addition to the application of RNN for MFDD in the HVAC systems, Shen and Khorasani [127] developed a hybrid machine learning model consisting of the integration of recurrent neural network (RNN) as a supervised ML model for feature selection and self organizing map (SOM) as an unsupervised ML model for clustering purposes and distinguishing the abnormal and abnormal performances, in order to be developed and applied for MFDD in the aircraft gas turbine engines. The proposed hybrid model was compared with the widely developed supervised neural network models such as deep artificial neural network (DANN), RNN, and multi-layer perceptron (MLP) for FDD, and the results revealed, their developed model has high performance in MFDD.

2.3.6.3. Convolutional neural network (CNN)

The Convolutional Neural Network (CNN) [128-130] is another deep learning model which is used for MFDD. This model is a type of feedforward neural networks which contains the convolution layers, pooling layers, and fully connected layers. The cost function which is applied in CNN is cross-entropy function. Furthermore, it is noteworthy to mention that CNN can be used

as a classification ML model which is trained by the labeled faulty and normal operation in the data set.

Miyata et al. [131] developed a CNN model for MFDD in the water side of a HVAC system in an office building, and they evaluated the severity of the faults. They used the 1-min time interval data sets which were generated using the first principal models. The considered faults in the system components, with consideration of chiller performance, sewage pump set value, heat exchanger efficiency, temperature sensor. The trained CNN model had 98.4% MFDD accuracy. They calculated the diagnosis probability for selection the simultaneous multiple faulty variables with the severity of the faults. They considered from 0% to 100% for the probability of the fault, which indicates from no fault to completely faulty operation in the system, respectively.

Elnour and Meskin [132] developed an optimized CNN model for the MFDD of damper stuck and water valve faults of AHU in a multi-zone office building, with tuning the hyperparameters e.g., number of layers and convolution filter size. The data set was generated using the simulated model in the TRNSYS with 1-min time interval. The results revealed very well performance of the developed CNN model for MFDD.

2.3.6.4. Generative adversarial network (GAN)

The GAN is an unsupervised deep learning model which was proposed by Goodfellow et al. [133] The GAN consists of two neural networks, 1) generator, and 2) a discriminator. The generator produces the synthetic data points from random noises, and discriminator compares the produced data points with the real one for providing the feedback. Yan et al. [134] applied generative adversarial network (GAN) for balancing the rate of normal and faulty data points in the data set to imitate the real-worlds system operation. They revealed an improvement on the fault detection

and diagnosis accuracy rate with application of GAN. The GAN is able to generate the synthetic faulty data point samples with deploying a few available real-world faulty samples. They selected six typical faults, exhaust air damper stuck (fully open), return fan at fixed speed, cooling coil valve control unstable, cooling coil valve partially closed (15% open), outdoor air damper leak, and AHU duct leaking (after supply fan). They deployed data set from ASHRAE project No. RP-1312 with 1-min time interval. They selected unbalanced and balanced for the normal and faulty data sets, then averaging 30 times repetition was conducted to validate the final classification results. The SVM was applied for feature selection of the most important variables, 10 features out of 102. They developed single and ensemble machine learning models using SVM, random forest, decision tree, and multi-layer perceptron of ANN. The balanced number of normal and faulty labeled data points is an influential factor for MFDD using classification machine learning models.

In another study, Yan et al., [135] applied conditional Wasserstein GAN (CWGAN) algorithm using Python with Tensorflow and Keras libraries in order to generate artificial faults and balance the normal and faulty data sets with 2-min time intervals of a 90-tons chiller, for fault detection and diagnosis purpose. The investigated faults were reduced condenser and evaporator water flow rates, refrigerant leak and overcharge, excess oil, condenser fouling, and non-condensable in refrigerant. They developed and applied, SVM, decision tree, random forest, k-nearest-neighbor, logistic regression, and Bayesian network, as the classification machine learning models for FDD. They revealed SVM has higher fault detection accuracy, moreover, the faultier samples are trained in the model, the higher the accuracy result for MFDD is concluded.

2.3.7. Research including the comparison of different models for MFDD

Some researchers developed and compared different ML models for MFDD in the HVAC systems. For instance, Zhao [136] developed and compared single linear regression (SLR), multiple linear regression (MLR), and decoupling FDD method for simultaneous multiple fault detection and diagnosis of a centrifugal chiller. Five characteristics were considered, such as, water temperature difference of the evaporator, condenser water temperature difference, refrigerant condenser subcooling, condenser approach temperature, and overall condenser heat loss coefficient. They concluded, decoupling based FDD method has better performance for the MFDD in the centrifugal chillers.

House et al. [31] applied different machine learning classification models (ANN, KNN, rule-based classifier, Bayes classifier) for MFDD in the VAV AHU with deploying the simulated synthetic data set. The targeted faults were: Stuck cooling coil valve, cooling coil fouling, heating coil leakage, stuck VAV box damper, performance degradation of the supply fan, return fan controller failure, mixing box damper failure. They revealed the Bayes classifier and rule-based classifier have higher fault detection and diagnosis accuracy, respectively.

Montazeri and Kargar [137] developed machine learning models, such as SVM, radial basis function neural network, PCA, and kernel PCA for MFDD in the sensors and actuators of an AHU. The results revealed 60% and 62% MFDD accuracy of the developed PCA and kernel PCA models, respectively. Furthermore, using the developed RBF neural network model, the model had 98.7% accuracy. Therefore, according to their comparison with previous works, the developed model has higher accuracy compared with deep belief network (DBF) and decision tree.

Ebrahimifakhar et al. [138] developed and compared optimized classification ML models with tuned parameters for MFDD in the packaged roof top units. The ML models were, logistic regression (LR), linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), K-NN (1 nearest neighbor), bagging (500 trees), random forest (550 trees), AdaBoost (max_depth of each tree is 8 and 900 trees), XGBoost (0.5 learning rate, max_depth of each tree is 5), and SVM. With 10-fold cross validation, they revealed SVM and LR with 96.2% and 93.6% accuracy for MFDD have the highest, and K-NN and LDA with 83.6% and 76.2% accuracy rate, have the lowest classification models for MFDD, respectively. They studied seven faults, refrigerant undercharge and overcharge, compressor valve leakage, liquid-line restriction, condenser airflow reduction, evaporator flow reduction, and non-condensable gas.

2.3.8. Genetic algorithm (GA)

Genetic algorithm (GA) is a stochastic global search method in order to imitate the natural selection procedure by simulating the evolution of a population of solutions [139]. The GA is applied to optimize the problem by finding the optimal solution and it`s fitness function. More information about GA is presented in [60, 140-142] and the introduction of different GA modules are represented concisely in this section. The GA requires two principal ingredients: 1) the genetic representation of the solutions (e.g., variables), and 2) fitness function (cost function) for evaluation of the solutions.

The genetic algorithm is a stochastic approach which includes three main components, population, chromosomes, genes, and begins with selection of random populations. The population is a bunch of chromosomes which corresponds to the set of random candidate solutions. Chromosomes are made of genes and corresponds to the candidate solution. The genes are the values of each variable which can be considered in binary (0 or 1) or continuous (real) values of the problem.

GA can deploy the roulette wheel selection operator as the well-known approach in order to assign probabilities to the individuals and selection for creating the next generations corresponding to the fitness values. The scope of selection operator is the parents' selection each time to combine their genes, in order to be able to make new generations.

Another module which is considered in development of the GA is crossover and applied for exchanging the genes. Crossover refers to the process of exchanging the genes between chromosomes to generate new chromosomes (child). There is a stochastic behavior for the crossover operator using random points every time. There is a scale for the crossover which is called crossover probability (Pc) and it is between 0 to 1. If the Pc is 1, everything is transferred, and all children are transferred to the next generation. If the Pc is 0, the crossover operator does not have impact and it is useless, therefore, the parents get transferred between generations.

The other module for the GA is mutation which applies changes in the genes for the next generations. The genes are changed randomly, hence, there is a probability of mutation (Pm). If Pm is 1, all genes are mutated, but if Pm is 0, none of the genes are get mutated.

The Elitism is the other module in the GA, which copies a small portion of the various individuals to the next generations, and it is specified with elitism ratio (Er) parameter. Moreover, a portion of the best solutions will be involved in the process of selection and mutation, therefore, they help to improve the quality of individuals in the subsequent generation as well. The elitism ratio (Er) defines the percentage of the population that should be kept and transferred to the next generation without changes.

After setting all defined modules, the maximum number of generations are set in order to reach to the desired optimum accuracy.

Genetic algorithm (GA) has been used widely in developing and optimization of the models for prediction of the variables in HVAC system components. In a study done by Nassif et al. [143], the HVAC system components were tuned with application of Genetic Algorithm. The GA options were set as: crossover probability (Pc) = 0.8, mutation probability Pm = 0.05, the population size = 50, and the maximum number of generations G_{max} = 100. In another study by Wang et al. [139] the parameters in the HVAC system were tuned by GA heuristically. The population size (n_p) of 120, with crossover probability (Pc) = 0.7, the mutation probability (Pm) = 0.05, and the maximum number of generations (G_{max}) equal to 100. Padilla et al. [144] studied the application of GA for developing the supply air temperature and pressure models through finding the parameters using genetic algorithm model. For the GA algorithm, 100 points as the population size, 0.05 for the mutation rate, and the maximum generation number of 200 were set. In addition, they selected the crossover probability heuristically.

Genetic algorithm also is applied as a technique for developing the optimized set points, such as supply chilled water temperature [145], and for optimization of the set points for supply air temperature and ventilation air flow rates [146]. The building envelop parameters were optimized using GA by [147], and HVAC system cost and performance were optimized using GA in a study by [148].

In this thesis the LSM is used as method for the grey-box model development for the scope of artificial faults generation as the results will be discussed in Section 4.2.

In the Section 2.3.9 and 2.3.10, a summary of ML methods for the single and multiple fault detection for various components in the HVAC systems are represented, respectively.

2.3.9. Summary of research publications for single FDD

1) Hybrid models account for about 35% of total number of studies, PCA 28%, ANN for 24%, rule-based 20%, Bayesian network 11%, SVM has been applied in 6%, Decision tree 11%, K-NN 11%, clustering 7%, SVR 4%, Random Forest 2%, SVDD 2%, Deep ANN 2%, CNN 2%, and fractal correlation dimension 2%.
2) In the 68% of the publications, experimental data were used, in 28% of studies, the synthetic data were used, and in 4% of studies both experimental and synthetic data were used.
3) In the domain of HVAC systems, 65% of the publications focused on the FDD models for AHUs, 15% on chillers, 6% on the variable refrigerant flow (VRF) systems, 4% on the zone sensors and actuators, and 2% whole HVAC system. Papers for other systems/equipment account for 2% each.

The summary of details for the detection and diagnosis of the single fault in the HVAC systems are provided in the Table 9.1 of Appendix A.

2.3.10. Summary of research for Multiple FDD

The most important finding of the literature review for the multiple FDD is provided in this section, and the summary of details for the detection and diagnosis of the multiple faults in the HVAC systems are provided in the Table 9.2 of Appendix B.

1) Hybrid models account for about 22% of total number of studies, ANN for 22%, SVM has been applied in 22%, K-NN 17%, Bayesian network 11%, rule-based 5%, Decision tree 5%, Random Forest 5%, clustering 5%, SVDD 5%, Deep ANN 5%, CNN 5%, and linear regression and linear discriminant analysis each account for about 5% of total publications.

2) In the 67% of the publications, experimental data were used, and in 33% of studies, the synthetic data were used.
3) In the domain of HVAC systems, 44% of the publications focused on the FDD models for AHUs, 33% on chillers, and the other HVAC systems such the vapor compression refrigerant, whole HVAC system, Packaged rooftop unit, and GSHP each of them account for 5% of total studies of the literature review.

While most publications presented FDD methods for one faulty sensor, the detection and diagnosis of multi-faults (sequential or concurrent) in HVAC systems which occur often is still a challenge, since the symptoms from a combination of several faults make difficult the separation of individual faults. Fewer number of publications covered this topic, being limited to combinations of two or three faults as listed in Table 9.2.

According to the literature review, the publications regarding the detection and diagnosis of the multiple dependent faults in the air temperature sensors of an AHU were not founds, therefore, in this thesis, the MDFDD is studied as a novelty to fill the gaps.

2.4. Summary of reviewed literature for FDD methods

In conclusion of literature review section, the data-driven techniques have the most attention for FDD in HVAC system, because they can be used if there is limited information about the system operation model. The black-box model is the most common model of the process history-based technique due to its simplicity, performance, and accuracy. However, the second most common technique is rule-based model which is a qualitative-based model. The quantitative-based model is the least common popular method for FDD due to its complexity in development.

The summary of the strengths and weaknesses of the FDD models are represented in Table 2.4.

Table 2.4: Strengths and weaknesses of FDD methods

Model	Strengths	Weaknesses
Process history-based	- The models are well developed with just input and output of the system without knowing the physical system information (Black-Box). - Simplicity. - Good performance. - Applicable for non-linear systems.	- A large and reliable data set is required to develop an accurate model.
Qualitative model-based	- Simplicity for development and application. - No need to know the mathematical models of the system operation.	- Rules can be developed and applied for a specific system which the model cannot be applied for other systems.
Quantitative model-based	- Accuracy on the target value prediction. - Capture steady and transient system operation. - Flexibility. - Generalisation with enough information, etc.	- Complexity of models. - Computationally intensive. - If there is not enough information of the system operation, the estimation of the target variable may not be accurate.

The black-box models can be developed using only measurements from the training data set. Understanding of physical processes is not required. Once the accurate model is obtained, the prediction or classification would be with high accuracy.

The grey-box models are developed with the combination of the physics-based models and measurements. Black-box model could give good predictions, but the user cannot say why. The

grey-box model can explain the reason of some results, as understood from physics-based models. the grey-box models need to have the physics-based fundamentals for the development.

For development of the black-box models, selection of the correlated variables is so important to consider the effectiveness of the model, the accuracy, and stability. For the HVAC system components, the system operation variables (air temperature, supply/ return water temperature, flow rate, etc.), environmental variable (temperature, etc.), time indicators (wee-days, weekend), and operation conditions (operation schedule, on/off) are potential correlated variables.

The model hyperparameter selection is another important aspect for the machine learning model developments. If a few parameters are used as the input data set, short training data set and less hidden layer neurons are selected, the model may not be developed perfectly, in other words it will be underfitted, i.e., the model has not been trained very well, and cannot predict well the system performance. But, if many parameters and features are used as the input, and many hidden layer neurons in ANN model are selected, the model will be overfitted. The model is trained perfectly and predicts accurately in training set. But prediction in the new (testing) data set is not accurate.

2.5. Conclusions on findings and gaps

This literature review section critically reviewed the published articles on the single and multiple fault detection and diagnosis in the HVAC systems. The FDD techniques are categorized into three principal methods: 1) quantitative based model (physics-based model), 2) qualitative-based model (expert rule), and 3) process history-based model (data-driven model). The advantages and limitations of the studied models out of each category were studied and reported, and some of them are summarized here. This review section can be used as a guideline for development and

application of the models for detection and diagnosis of the multiple dependent faults in the components of the HVAC systems.

- The black-box model as a process history-based model has the most attention for FDD due to their performance and accuracy.
- The rule-based model as a qualitative-based model has the second attention as they can be developed simply and applied, with the limited knowledge of the mathematical models of the system operation.
- The physically based model as a quantitative-based model is the least common popular method for FDD due to their complexity in development and computationally intensive, and, if there is not enough information of the system operation, the estimation of the target variable may not be accurate.
- Hyperparameter selection for the development of an accurate data-driven predictive model is the key approach in MDFDD with data-driven based model.
- The results from the literature review revealed the acceptable and good performance of the machine learning and deep learning models for the fault detection and diagnosis of the single and multiple faults.

Some gaps from the literature have been found, and therefore, some of them will be targeted in this thesis, and the novelties are proposed for the scope of multiple dependent fault detection and diagnosis in the AHU. Some of the proposed novelties are described briefly below:

- Development of a novel method for the detection and diagnosis of the multiple dependent faults in the AHU sensors as new research, which this kind of research for the MDFDD has not been found in other publications from the literature review.

- Development of a novel efficient and practical sequential (compound) machine learning models for the prediction of a target variable in order to generate reference values for the scope of MDFDD in AHU sensors.

- Proposal of a novel technique which is the combination of the sensor uncertainty and the model inaccuracy for the definition of a threshold for the goal of MDFDD.

- Proposing a novel hybrid machine learning technique, which is the combination of a machine learning model and rule-based technique for the MDFDD of the sensors in an AHU.

- Development and comparison of various optimized machine learning, deep learning, and hybrid models for the MDFDD scope with application of the K-fold cross validation for validation the average performance accuracy of the models, and then justification of the most appropriate model to suggest.

- Development of the novel classification machine learning models for the MDFDD, and optimization of the models respect to the models hyperparameters and the input time-lags.

3. Research method

This thesis proposes a method for the detection and diagnosis of multiple dependent faults of sensors and equipment of an AHU, by using a combination of machine learning (ML) models, which proved in the past good performances for linear and non-linear systems [10, 149], rule-based models, and as another alternative method the classification approach of machine learning model is developed and used for the MDFDD scope.

The method # 1 is the combination of machine leaning model with rule-based technique and it is described with details of the results from the experimental and synthetic data in Section 5.1.

The method # 2 is the classification machine learning model for the MDFDD, and the results and discussion are described in Section 5.2.

The ML models are developed from BAS trend data, and implemented using Python (version 3.8.1) [54] with open-source libraries-such as Scikit-learn [71], Keras [72], Tensorflow [73].

One possible approach would consist of launching an exhaustive search, by using ML models, for detecting faults of all sensors in the AHU at time t.

The novelty of method #1 proposed in this thesis is the combination of ML model for prediction with the rule-based technique for the MDFDD in the air temperature sensors of an AHU.

In the method # 2 the classification ML models are developed as the novel approach for the MDFDD of the air temperature sensors in the AHU.

In the following sub-sections, the details are provided as following order:

Section 3.1: method # 1, the combination of machine leaning model for prediction with rule-based technique

Section 3.2: machine learning models description

Section 3.3: rule-based model for fault diagnosis

Section 3.4: method # 2, the classification machine learning models

Section 3.5: artificial faults generation using grey-box model.

The proposed method is applied on an air handling unit which the schematic design of that along with the available variables are provided below.

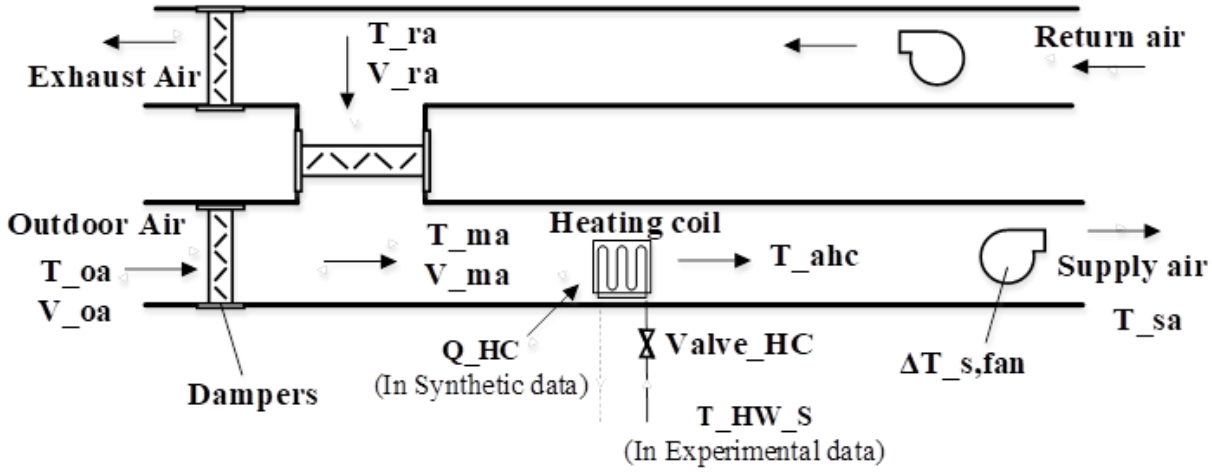

Figure 3.1. Schematic of Air Handling Unit from experimental data

Table 3.1. List of AHU variables from the experimental data.

No.	Variable	Description	Fixed (bias) (B_x)	Random error (R_x)	Total uncertainty	Unit
1	T_oa	Outdoor air dry-bulb temperature	0.45	0.19	0.49	°C
2	T_ra	Return air temperature				°C
3	T_ma	Mixed air temperature				°C
4	T_ahc	Air temperature after heating coil				°C
5	T_sa	Supply air temperature				°C
6	ΔT_s, fan	Air temperature rises over supply fan	-	-	-	°C
7	T_HW_S	Supply hot water temperature	0.31	2.20	2.22	°C
8	Q_HC	Electric input for heating coil	This is a variable from synthetic data			kW
9	V_ma	Mixed air volumetric flow rate	222.0	506.11	552.65	L/s
10	V_ra	Return air volumetric flow rate	222.0	229.84	319.55	L/s
11	V_oa = (V_ma – V_ra)	Outdoor air volumetric flow rate	222.0	355.98	419.53	L/s
12	Valve_HC	Heating coil valve position	-	-	2	%

3.1. Method # 1: The combination of machine leaning model for prediction with rule-based technique

In this method, the machine learning models (e.g., SVR) are developed for the prediction of the target air temperature sensors (T_ma and T_ahc).

In this thesis, a novel approach is proposed which is the compound ML model. A compound ML model (Figure 3.2) uses two distinct but related models (ML no.1 and ML no.2), one for the prediction of T_ma sensor value, and another for the prediction of T_ahc sensor value, each one measuring the impact of several inputs. Sensors that measure those independent inputs are called regressor sensors, for instance the outdoor air temperature (T_oa) sensor.

The first ML model is used for the prediction of T_ma, then the predicted T_ma is used as an input for the second ML for the prediction of T_ahc. If measured T_ma is different from predicted T_ma, the predicted T_ma is used as input to ML#2. If measured T_ma is almost equal with predicted T_ma, either measured T_ma or predicted T_ma can be used as input to ML#2. However, for simplification and be consistent with the first option, predicted T_ma is also use as input to ML#2.

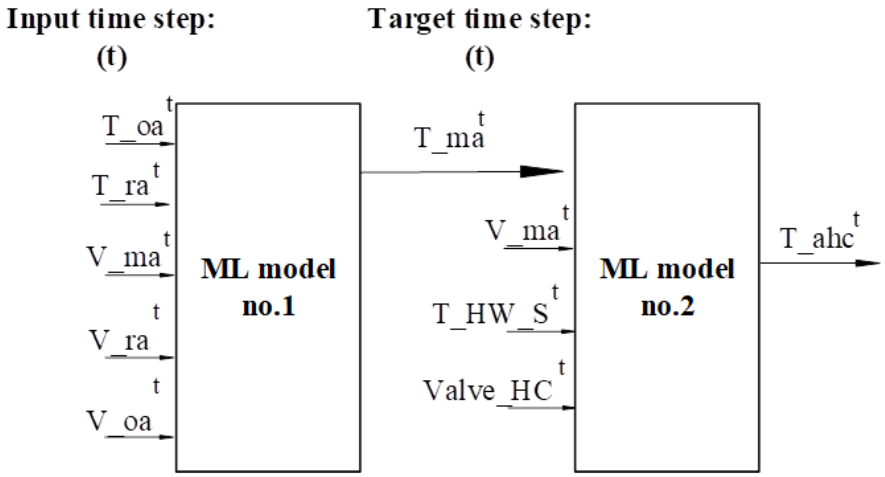

Figure 3.2. Schematic ML model for prediction of target variables T_ma and T_ahc at time t for fault symptom detection

The following steps are implemented for the detection and diagnosis of the dependent multiple faults in the air temperature sensors of the AHU (Figure 3.3):

1) Data collection from BAS.
2) Data preprocessing for the outliers' removal.
3) Training and testing data sets selection.
4) ML model development for the prediction of target sensor (T_ma).
5) ML model development for the prediction of target sensor (T_ahc).
6) Residuals between predicted and measured values of T_ma and T_ahc, respectively.

7) Application of rule-based technique for the fault diagnosis step.

The sensors should be calibrated regularly to provide measured data sets with reliability. Otherwise, the preprocessing step on the data set is required.

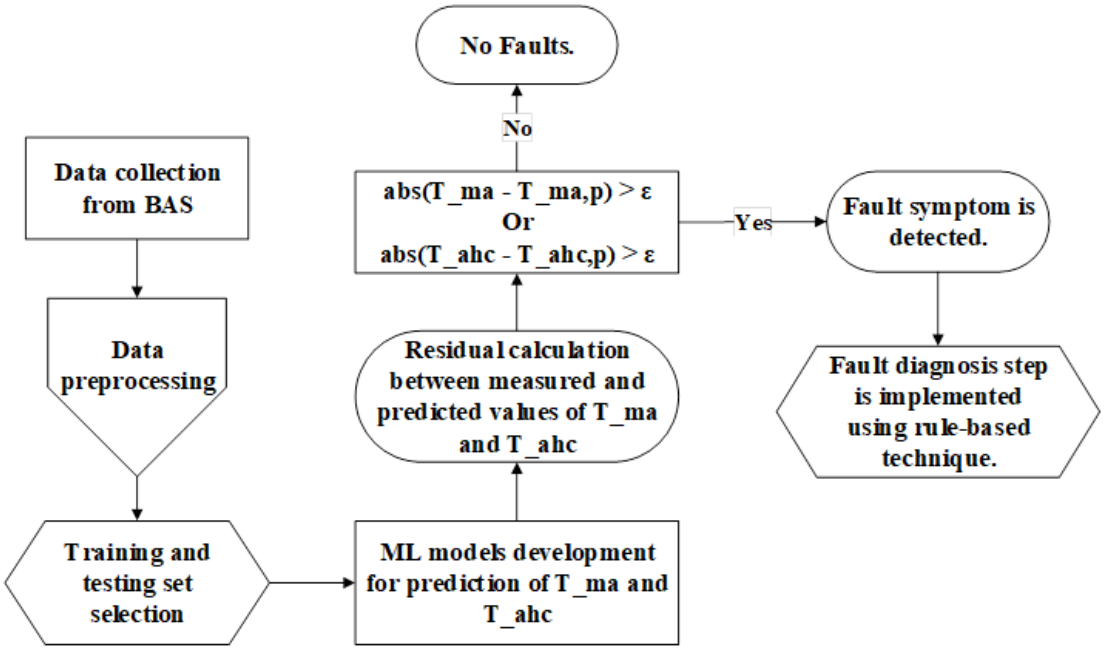

Figure 3.3. MDFDD diagram

In the data sets, always some outliers may exit which can be due to the transient operation in the system or some abnormal activities as the results of the systems operators. Therefore, it is vital to preprocess the data sets by removing the outliers in order to develop an accurate model.

There are two phases of fault symptom detection at each time step t:

Phase 1

The support vector regression (SVR) is used to predict the mixed air temperature T_ma,p that corresponds to normal operation, without known problems. If the residual (Res) of actual

measurements of T_ma and predicted values T_ma,p exceeds the threshold ε, with positive or negative measuring bias, i.e., Res_ma = abs(T_ma - T_ma,p) > ε, the fault symptom of T_ma sensor is detected. Then the fault diagnosis method is activated. Otherwise Phase 2 is activated for the prediction of following target sensor, i.e., T_ahc.

The fault symptom could be generated by abnormal measurements of faulty target sensor, or by abnormal operation of other sensors/devices due to the improper control or degradation of performance. Therefore, one can ask the question: is the T_ma sensor faulty, and/or the regressor sensors (T_oa, T_ra, V_ma, V_ra, V_oa) are faulty (Figure 3.2).

To respond to this question, the recurrent neural network (RNN) is used for the fault symptom detection of regressor sensors. The value of each regressor sensor X at time t is predicted by using the past measured values at t-1, t-2, …, t-n (Figure 3.4). If the residual between actual measurements and predicted values, corresponding to normal operation of regressor sensor X, exceeds the threshold ε, the fault symptom of regressor sensor X is detected.

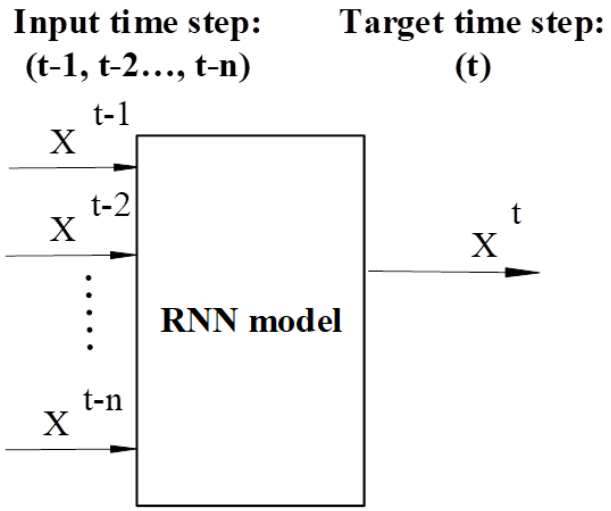

Figure 3.4. Schematic RNN model for prediction of regressor sensor X at time t for fault symptom detection

As an example, fault symptoms might be detected in two sensors, T_ma and T_oa. The diagnosis must clarify if both sensors are faulty, or only one sensor. If the fault symptom of regressor sensor T_oa is detected, the corrected sensor output T_ma,R under the influence of faulty T_oa is calculated by using a grey-box model (Equation (3.8)).

The steps for the development of the grey-box model are provided below:

$$T\_ma = \frac{\rho_{oa}.C_{p,oa}.V\_oa.T\_oa + \rho_{oa}.C_{p,ra}.V\_ra\ .T\_ra}{\rho_{ma}.C_{p,ma}.V\_ma} \quad (3.1)$$

$$\rho = \frac{P}{R\ T} \quad (3.2)$$

$$C_p = a + bT + cT^2 + dT^3 \quad (3.3)$$

where: V_oa, V_ma, V_ra, and V_ahc = the outdoor, mixed, return and after heating coil air volumetric flow rates (m^3/s)

ρ_{oa}, ρ_{ra}, ρ_{ma} and ρ_{ahc} = the outdoor, return, mixed and after heating coil air density (kg/m^3)

$C_{p,oa}$, $C_{p,ma}$, $C_{p,ra}$ and $C_{p,ac}$ = the specific heat capacity of air for outdoor, mixed, return and after heating coil at constant pressure (kJ/kg. K) (Equation (3.3))

The coefficients of Equation (3.3) are derived from [2]: a=28.11, b=0.1967×10-^{2}, c=0.4802×10^{-5}, d=-1.966×10^{-9}.

$$V\_oa = \frac{a * \rho_{ma}.C_{p,ma}.V\_ma}{\rho_{oa}.C_{p,oa}} \quad (3.4)$$

$$a = \frac{\rho_{oa}.C_{p,oa}.V\_oa}{\rho_{ma}.C_{p,ma}.V\_ma} \quad (3.5)$$

The outdoor air fraction (a) is the ratio of the outdoor mass air flow rate to mixed mass air flow rate (Equation (3.5)). The volumetric air flow rates at difference stages in the AHU are defined in Equations (3.6) and (3.7).

$$V\_ma = V\_ahc = V\_sa \quad (3.6)$$

$$V\_ma = V\_oa + V\_ra \quad (3.7)$$

$$T\_ma,R = a \cdot Toa + b \cdot T\_ra \quad (3.8)$$

where: T_ma,R = the expected value of T_ma as affected by the faulty T_oa and/or T_ra

coefficients a =Voa/Vma, and b = Vra/Vma = 1-a; coefficients a and b are identified by using the least square method (LSM) with training data set

For simplification, it is assumed that other regressor sensors are accurate.

If the residual of measurements of T_ma and corrected predicted values T_ma,R (Equation (3.8)) does not exceed the threshold ε, then T_ma is not a faulty sensor, but it signals the deviation due

to the faulty T_oa. Therefore, only T_oa sensor is faulty. Similar approach is used for other regressor sensors such as T_ra. A few examples of rule-based diagnosis models are presented in Section 3.3.

Phase 2

The support vector regression (SVR) is used to predict T_ahc,p that corresponds to normal operation, without known problems. If the residual (Res) of actual measurements of T_ahc and predicted values T_ahc,p exceeds the threshold ε, i.e., Res_ahc =(T_ahc - T_ahc,p) > ε, the fault symptom of T_ahc sensor is detected. Then the fault diagnosis step is activated.

If the regressor sensor T_ma is faulty (see Phase 1), the corrected sensor output T_ahc,R under the influence of faulty T_ma is calculated by using a grey-box model (Equation (3.9)):

$$\mathrm{T\_ahc,R} = \frac{\mathrm{c} \cdot \mathrm{T\_HW\_S}}{\mathrm{d} \cdot \mathrm{V\_ma}} + \mathrm{e} \cdot \mathrm{T\_ma} \tag{3.9}$$

where: T_ahc,R = the expected value when T_ahc is affected by the faulty T_ma

coefficients c, d, and e = identified by using the least square method with training data set

If the residual of measured T_ahc and predicted T_ahc,R (Equation (3.9)) does not exceed the threshold, we can conclude that T_ahc is not faulty, but it signals the deviation due to the faulty T_ma. Similar approach is used for other regressor sensors such as T_HW_S. A few examples of rule-based diagnosis models are presented in Section 3.3.

3.2. Machine learning models

In this thesis, since there are two types of data sets: experimental and synthetic data, in order to evaluate the proposed methodology, the ML models are applied on both data set types. Also, different machine learning models are developed in this thesis, such as: support vector regression (SVR), artificial neural networks (ANN), recurrent neural network (RNN), deep feedforward neural networks (FFNN), decision tree (DT), and random forest (RF), to compare their performance in order to suggest which ML model performs better for the scope of MDFDD. The summaries of the proposed ML models are provided in the following subsections.

3.2.1. Support vector regression (SVR)

Support vector regression (SVR) is a supervised machined learning model which comes from the support vector machine (SVM) for regression-based purposes [94-95, 102, 150]. The SVM model predicts the target value with the function of (f) by mapping using a nonlinear function ($\emptyset$), the data set of x into a higher dimension feature space.

$$f(x) = \langle w, \emptyset(x) \rangle + b \tag{3.10}$$

where: w = the matrix of regression coefficients

b = the intercept

x = the matrix of regressors

$\langle .,. \rangle$ = the dot product

The optimization model was proposed by Vapnik [151] to formulate function f which includes regression coefficient (w) and intercept (b) to predict the target vectors (y) with a precision of δ.

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C.\sum_{i=1}^{l}(\xi_i + \xi_i^*)$$
$$\langle w, \emptyset(x_i)\rangle + b - y_i \leq \delta + \xi_i \,, \tag{3.11}$$
$$y_i - \langle w, \emptyset(x_i)\rangle - b \leq \delta + \xi_i \,,$$
$$\xi_i, \xi_i^* \geq 0, i = 1, \ldots, l.$$

where: y_i = the target vector observation

ξ_i = a slack variable

δ and C = the parameters which needs to be selected through random search over a given range of values

The regression function f(x) and regression coefficients (w) are presented in Equations (3.12) and (3.13).

$$f(x) = (\alpha_i + \alpha_i^*)k(x_i, x) + b \tag{3.12}$$

$$w = \sum_{i=1}^{l}(\alpha_i + \alpha_i^*)x_i \tag{3.13}$$

where: α_i and α_i^* = the Lagrange multipliers

k = the kernel function

The kernel function is used for the distribution representation of input values of the training data set [152]. Radial basis function is used as the kernel function as represented in Equation (3.14).

$$k(x_i, x_j) = \exp\left(-\gamma.\left\|x_i - x_j\right\|^2\right) \tag{3.14}$$

where: γ = the width parameter which reflects the variation range of all regressors in the training data set.

The values of required parameters (δ, C, γ) are identified using the training data set, and thus the SVR model is developed. The predicted target values are obtained by using the testing data set. The compound SVR models for the prediction of the T_ma and T_ahc are summarized by Equations (3.15) and (3.16).

$$T\_ma^t = f(T\_oa^t, T\_ra^t, V\_oa^t, V\_ra^t, V\_ma^t) \quad (3.15)$$

$$T\_ahc^t = f(T\_ma^t, V\_ma^t, T\_HW\_S^t, Valve\_HC^t) \quad (3.16)$$

3.2.2. Artificial neural network (ANN)

Another ML model in this thesis is the artificial neural network (ANN). The development of the ANN model is discussed in this section for the scope of prediction of the target air temperature sensors (T_ma and T_ahc).

The following steps are taken:

Several ANN architectures are compared with the number of training days from 1 to 30, and the number of hidden layer neurons from 1 to 10; in all cases, there is only one hidden layer. The Sigmoid activation function is used for the hidden layer neurons, and the output node.

1. Each architecture is evaluated ten times as the 10-fold cross-validation; the average RMSE (between the predictions and measurements) of 10 times ANN architecture evaluations are recorded. The optimum ANN architecture, which requires the lowest number of training days with RMSE smaller than the overall uncertainty of the corresponding temperature sensor, is

selected as the optimum ANN#1 model. In this thesis, the overall uncertainty of the air temperature sensor [74] is ± 0.49°C [45] based on previous measurements of the same AHU. For the selection of the optimum number of hidden layer neurons (N), the approach proposed by [122] is applied (Equation (3.17)).

$$N = \frac{N_{Input} + N_{Output}}{2} \tag{3.17}$$

where: N_{Input} = the number of inputs

N_{output} = the number of outputs

2. The optimum ANN#1 architecture for the prediction of T_ma over the time horizon of training data has three hidden layer neurons.
3. The mixed air temperature T_ma is predicted at the time (t) by using the regressors that are measured at the same time (t). If the difference between the measurement of T_ma and the prediction of T_ma obtained with proposed ANN#1 does not exceed the threshold value, there is no fault with the T_ma sensor. Then, the measured T_ma is used as an input to ANN#2. While, if the difference exceeds the threshold value, there is a fault with the T_ma sensor, and the predicted T_ma is used as an input for ANN#2.
4. The same approach is used for the ANN#2, which is trained by using the measurements over the past training set. The optimum ANN#2 for the prediction of T_ahc at the current time (t) requires two hidden layer neurons (from Equation (3.17)).
5. The air temperature T_ahc after the heating coil is predicted at the time (t) by using selected regressors that are measured at the same time (t), and T_ma predicted or measured (as

discussed at item 2 above). If the difference between the measured T_ahc and the ANN#2 prediction of T_ahc does not exceed the threshold value, there is no fault of T_ahc sensor. The measured T_ahc is used as an input to predict the supply air temperature T_sa. If the difference exceeds the threshold value, the predicted T_ahc is used as an input to predict T_sa.

6. The supply air temperature (T_sa) is predicted at the same time (t) by adding the calculated air temperature rise in the supply fan (ΔT_s,fan) to the T_ahc value. In this study, ΔT_s,fan=1.8°C, based on previous measurements of the same AHU [17].

When a fault is detected with the sensors of T_ma and T_ahc, the fault diagnosis model is applied.

3.2.3. Recurrent neural network (RNN)

The RNN model predicts the values of the correlated regressor sensors at time t by using the previous values at times t-1, t-2, …, t-n. The RNN model is a deep learning model with long-short-term memory (LSTM) architecture that uses the previous sequential information to learn and predict the present values. LSTM architecture has the chain like structure of the neural networks and able to learn the long-term dependencies. The LSTM is capable of adding, storing and removing the information [126]. The internal schematic structure of the RNN with the LSTM algorithm is illustrated in Figure 3.5.

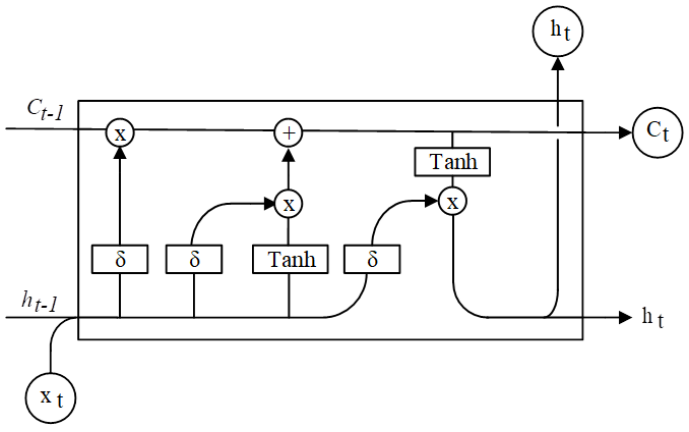

Figure 3.5. Recurrent neural network with LSTM algorithm [126]

where: x_t = the input vector

h_t = the hidden layer or output vector

tanh and δ = the activation functions

C_t is the state of cell

The RNN models for the prediction of regressor values at time t are summarized by Equations (3.18) to (3.22).

$$T\_oa^t = f(T\_oa^{t-1}, T\_oa^{t-2}, \dots, T\_oa^{t-n}) \quad (3.18)$$

$$T\_ra^t = f(T\_ra^{t-1}, T\_ra^{t-2}, \dots, T\_ra^{t-n}) \quad (3.19)$$

$$T\_ma^t = f(T\_ma^{t-1}, T\_ma^{t-2}, \dots, T\_ma^{t-n}) \quad (3.20)$$

$$T\_HW\_S^t = f(T\_HW\_S^{t-1}, T\_HW\_S^{t-2}, \dots, T\_HW\_S^{t-n}) \quad (3.21)$$

$$Valve\_HC^t = f(Valve\_HC^{t-1}, Valve\_HC^{t-2}, \dots, Valve\_HC^{t-n}) \quad (3.22)$$

The RNN models predict the regressor sensor output at time t and the following steps are implemented:

a. For target sensor T_ma, the regressor sensors are T_oa, T_ra, V_ma, V_oa, V_ra; and
b. For the target sensor T_ahc, the regressor sensors are T_ma, V_ma, T_HW_S and Valve_HC.

The optimum hyperparameters of the RNN models will be selected by using The RandomizedSearchCV method [71]. This is a tool in the Scikit-learn package of Python, which

randomly selects the hyperparameters out of the values and options assigned by the user, to obtain the optimum ML models for T_ma and T_ahc over k-fold cross validation:

For an example, in the optimized RNN model, the number of hidden layers = 4, number of hidden neurons in each hidden layer = 50, the dropout regularization ratio = 0.2, and the sigmoid activation function. Ten time-lags of measurement are selected by random search as inputs of regressors.

3.3. Rule-based model for fault diagnosis

After the fault symptoms are detected for target and regressor sensors, some rule-based models are used to diagnose the potential causes of faults. A few examples of such rules are presented below. When some sensors or devices are identified as possibly faulty, physical investigation by the maintenance staff is needed. In the meantime, the values of T_ma,p, T_ra,p, T_oa,p, and T_ahc,p, which are predicted by SVR or RNN models (used as virtual sensors), can be used for the continuation of correct operation of AHU.

3.3.1. Group A of rule-based models

When the fault symptom of sensor T_ma is detected (i.e., Res_ma = abs(T_ma - T_ma,p) > ε), the following rules apply (for the simplification of explanation, all other regressor sensors are assumed to be correct):

a. If Res_oa = abs(T_oa - T_oa,p) > ε, and Res_ma = abs(T_ma - T_ma,p) > ε, then both T_ma and T_oa sensors have fault symptoms.
b. If Res_ra = abs(T_ra – T_ra,p)> ε, and Res_ma = abs(T_ma - T_ma,p) > ε. then both T_ma and T_ra sensors have fault symptoms.
c. If Res_oa> ε, and/or Res_ra>ε, and Res_ma = abs(T_ma - T_ma,R) < ε, then T_ma sensor is not faulty; and T_oa and/or T_ra sensors are faulty.

d. If Res_oa < ε, and/or Res_ra < ε, and Res_ma = abs(T_ma - T_ma,p) > ε, then only T_ma sensor is faulty.

e. If Res_oa > ε, and Res_ra > ε, and Res_ma = abs(T_ma - T_ma,P) > ε , then all three sensors, T_oa, T_ra, and T_ma, are faulty.

Other rules can be used for the diagnosis of outdoor and return air flow dampers position.

3.3.2. Group B of rule-based models

When the fault symptom of sensor T_ahc is detected with positive measuring bias Res_ahc = (T_ahc – T_ahc,p) > ε, the corrected sensor output T_ahc,R under the influence of faulty T_ma is calculated, and the following rules apply:

a. If Res = (T_ahc - T_ahc,R) < ε, then T_ahc sensor is not faulty.

b. If Res = (T_ahc – T_ahc,R) > ε, then T_ahc sensor is faulty.

b. If the heating coil valve position (Valve_HC) is recorded open with (Valve_HC – Valve_HC,p) < ε, abs(T_ma – T_ma,p) < ε, and (T_HW_S - T_HW_S,p) > ε, then T_ahc sensor is possibly faulty, and/or the hot water temperature is too high.

c. If (Valve_HC) is recorded open with (Valve_HC – Valve_HC,p) > ε, abs(T_ma – T_ma,p) < ε, and (T_HW_S - T_HW_S,p) < ε, then T_ahc sensor is possibly faulty, and/or the heating coil valve might be stuck opened.

d. If (Valve_HC) is recorded open with abs(Valve_HC – Valve_HC,p) < ε, abs(T_ma – T_ma,p) < ε, and (T_HW_S - T_HW_S,p) < ε, the T_ahc sensor is possibly faulty with positive bias.

e. If (Valve_HC) is recorded as closed, and abs(T_ma – T_ma,p) < ε, then Valve_HC and/or heating coil leaks.

f. If (Valve_HC) is recorded open, and (T_ma – T_ma,p) > ε, then T_ma sensor and/or T_ahc sensor might be faulty.

3.4. Method # 2: The classification machine learning models

In this section the development of classification-based machine learning model is discussed as an alternative approach for the multiple dependent fault detection and diagnosis in the sensors of an AHU. The following machine learning models with classification approach are developed over optimization process for the models' hyperparameters selection.

Support vector machine (SVM), K-Nearest Neighbor (KNN), and Naïve Bayes, shallow artificial neural network (ANN), Deep ANN, decision tree classification, random forest classification, and principal component analysis are developed in this thesis to evaluate the performance of each model on the MDFDD. Then, the model with the highest accuracy for the MDFDD will be selected. The results discuss the best performance in terms of the accuracy, precision, and sensitivity for the single and MDFDD of air temperature sensors in an AHU of an institutional building.

Classifier model development:

In training and testing process of the model, for the multiple dependent fault detection and diagnosis in three air temperature sensors T_oa, T_ra, and T_ma, the data is selected from the following sensors: T_oa, T_ra, T_ma, V_oa, V_ra, V_ma.

The values of the regressors sensors are input into the ML models, and the condition of target sensors are binary (Normal or Faulty) to indicate whether the target sensor is faulty or not.

Once the data set is collected and includes the normal and faulty data points, the classification ML model is trained. The architecture of the classification ML model is shown in Figure 3.6 and the

model is presented in Equation (3.23). The scope of this ML model is the classification of the target sensor condition.

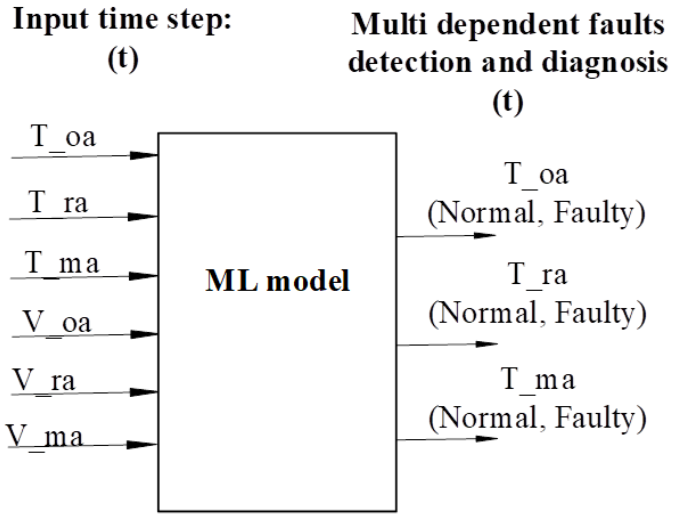

Figure 3.6. ML model architecture for the MDFDD

$$\text{T\_oa(condition: Normal/Faulty)}^{t}, \text{T\_oa(condition: Normal/Faulty)}^{t}, \text{T\_oa(condition: Normal/Faulty)}^{t} = f(\text{T\_oa}^{t}, \text{T\_ra}^{t}, \text{T\_ma}^{t}, \text{V\_oa}^{t}, \text{V\_ra}^{t}, \text{V\_ma}^{t}) \quad (3.23)$$

In this equation, the target is the classification of the conditions of three air temperature sensors; T_oa, T_ra, and T_ma at time t. The target conditions for each sensor can be normal or faulty, which are indicated by 0 or 1, respectively. The inputs of this equation are the values of the sensors (T_oa, T_ra, T_ma, V_oa, V_ra, V_ma) at time t.

The ML model from the provided list in Section 3.4 (e.g., SVM, ANN, KNN) is developed with the input of available variables for the MDFDD of the air temperature sensors in the heating season. A data set of over the heating season from January 13 to January 19 at every single 15-min time interval for the measured data including faulty and normal data is used for training (672 data

points). The training data set is composed of normal data subset (70% of training data, or 472 data points), and faulty data subset (30% of training data, or 200 data points).

3.5. Artificial faults generation using grey-box models

In some case studies, since the data set may only include the normal operation, artificial faults should be generated. There are different methods for generation of artificial faults in the data sets. For instance, if the data set is synthetic and generated by the software such as eQuest, EnergyPlus, etc., and some modifications can be implemented in the features of the developed model in the software to introduce faulty operation, and in results, the faulty data are generated. Also, there are other methods for the faults' generation such as data-driven based approaches for labeling the faults and physically based models. Some references are provided here for the artificial faults' generation [153-156].

Another approach for the faults' generation is introduction of positive/ negative bias in the sensors' readings. This method is combined with the grey-box models as a data-driven based method in this thesis for the artificial fault generation.

In this thesis, in most cases of measurements from BAS, the available data set does not contain enough abnormal operation data due to sensor faults. In the absence of faulty data, in order to evaluate the performance of the proposed MDFDD model, artificial faults of some sensors are inserted in the testing data set. For instance, if a fault is inserted in the file of T_oa, the relationship between the value of regressor sensor and target sensor (e.g., T_ma) must also be added. For this purpose, two grey-box models (Equations (3.8) and (3.9) are developed, that will predict the relationship between the regressor sensors and target sensors of T_ma and T_ahc, respectively.

For the development of the grey-box models, some approaches can be implemented, which in this thesis two approaches are studied: 1) least-square-error-method (LSM), 2) genetic algorithm (GA).

In this thesis, the same approach was used for the generation of artificial faults for both synthetic and experimental data over the same data set.

3.5.1. LSM for obtaining grey-box model coefficients

The LSM is used as an approach for finding the coefficients of the grey-box model with use of Microsoft Excel Spreadsheet Software. The results for the grey-box model development using LSM method are provided in Section 4.2.1.

3.5.2. Genetic algorithm method for grey-box model development

Another approach for the development of the optimized grey-box model is the GA, which is studied in this thesis for prediction of mixed air temperature, and air temperature after heating coil. Then, the performances of the proposed models are evaluated and compared with the prediction performance of the models developed by LSM.

The Genetic Algorithm solver in the optimization tool of MATLAB [53] is used in order to find the coefficients of the model to develop a grey-box model for the prediction of T_ma and T_ahc.

The program setup and options are selected heuristically through trial and error in order to minimize the average of squared difference between measured and predicted values over test set for T_ma and T_ahc (E_1 and E_2, Equations (3.24) and (3.25)). The RMSE is considered as a measure for evaluation the impact of Pc changes on the developed GA model, and the results are illustrated in Figure 3.7 and Figure 3.8. The results for finding the optimum values revealed the crossover probability of 0.9 developed models for T_ma and T_ahc, respectively.

These values have been selected based on trial-and-error approach.

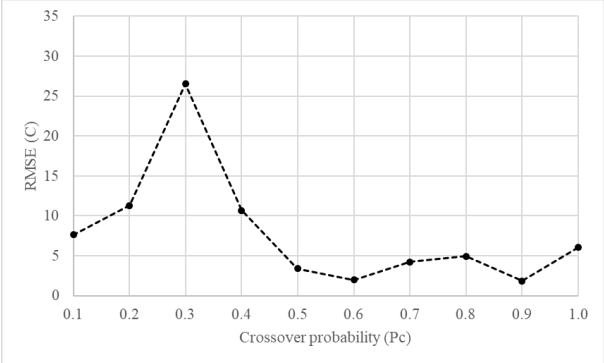

Figure 3.7. Impact of Pc on RMSE of the developed model of T_ma using GA

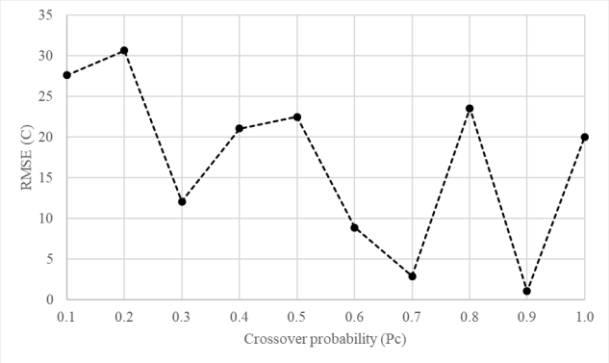

Figure 3.8. Impact of Pc on RMSE of the developed model of T_ahc using GA

The optimum setting of the GA algorithms is selected as follow; the population size (n_p) is set to 100, the crossover probability (Pc) is 0.9, the mutation probability (Pm) is 0.02, the elitism ratio is 0.05, with 250 maximum number of generations (G_{max}).

$$E\_1 = \sum_1^n |T\_ma\_measured - T\_ma\_predicted|^2/n \quad (3.24)$$

$$E\_2 = \sum_1^n |T\_ahc\_measured - T\_ahc\_predicted|^2/n \quad (3.25)$$

where, n is the number of values over the test set.

The results of the GA method for the development of the grey-box model are reported in Section 4.2.2.

4. Case study

In this thesis, the case study is an air handling unit of an institutional building which is the Genomic building with the total floor area of 5,400 m^2 including three floors (Figure 4.1). The building is located in Montreal, Canada, with the orientation of 60º NW and window-to-wall ratio of 33%. This building has 48 offices, three conference rooms and corridors which allocate for about 53% of the total floor area. The laboratories with the fume hoods accounts for 30% of the area, and the remaining areas are accounted for the kitchen (lounge) and restroom on each floor. The details for the HVAC system can be found in [157-158]. The building was certified LEED gold in 2013.

Figure 4.1. Genomic building [159]

The systems for the offices and laboratories of the building have variable air volume (VAV) flow rate, and constant temperature with continuous functionality.

The supply air temperature from AHU is controlled in terms of outdoor air temperature. The mixed air temperature is controlled in such an away to reduce the operation of heating/cooling coils. The heating coil leaving temperature is controlled by adjusting the heating coil water valve.

There are VAV boxes in room that control the supply flow rate to keep the room set point air temperature. Over the occupied period, the design supply air flow rate is 10 air changes per hour (ACH). Over the unoccupied periods during the day, the design supply air flow rate is 6 ACH. Over the unoccupied period over the night, the design supply air flow rate is 3 ACH.

Total fan characteristics are as follows: supply fans have total design capacity of 42,500 L/s flow rate, and 120 kW electric power input; the return fans have total flow rate of 14,200 L/s and 30 kW design power; the exhaust fans have total flow rate of 33,034 L/s and 60 kW design power.

The supply air temperature set point is between 12 and 16 °C.

In this thesis two data sets are used for evaluation of both methods # 1 and method # 2: 1) Experimental data, and 2) Synthetic data.

For the first application in method # 1, the experimental data recorded at 15-minute interval, over the heating season from December 26, 2016 to January 29, 2017, from the AHU (Figure 4.2) is used for the development and evaluation of the proposed methods for the MDFDD, and discussed in the Section 5.1.

For the same proposed method # 1, the synthetic data generated with eQuest program developed by [157] is used over the same period of heating season as deployed in the experimental data, and application of proposed method on the synthetic data is studied in Section 5.1. The model calibration was carried out through a bottom-up approach, starting with variables at the zone level, then with variables at the AHU level, and finished with variables at the building supply level. The synthetic data, which are error-free, are selected as normal operation data of the system.

The list of available variables along with the corresponding uncertainty of sensors (calculated for the case of measurements) are presented in Table 4.1. The sensors uncertainty, calculated from fixed (bias) and random errors, are used to generate the threshold to detect and diagnose the multiple faults. The fixed (bias) and random errors for the sensor's uncertainty were obtained from previous study of on the same AHU [113, 158]. In the synthetic data, instead of supply hot water temperature (T_HW_S), the electric input for heating (Q_HC) is available.

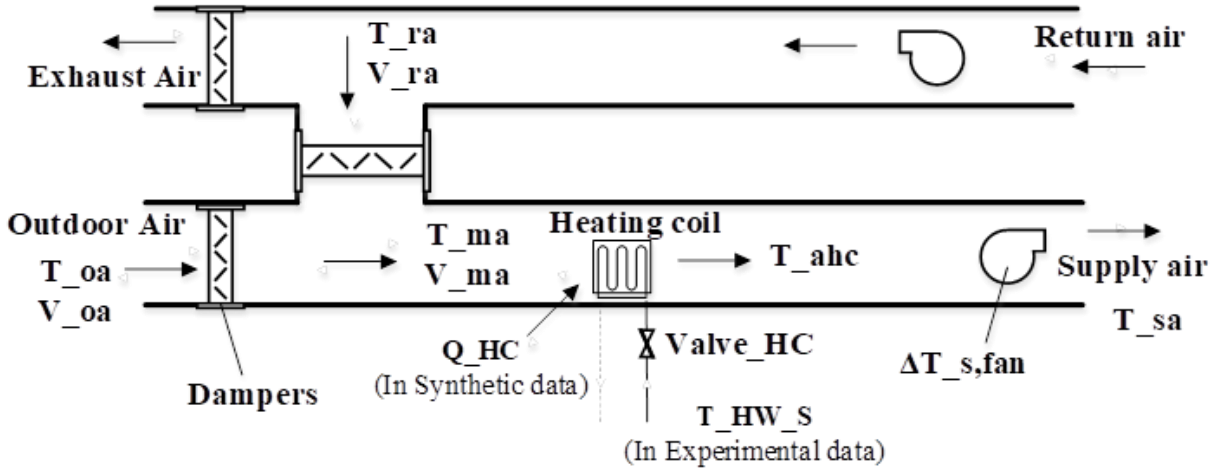

Figure 4.2. Schematic of Air Handling Unit from experimental data

Table 4.1. List of AHU variables from the experimental data.

No.	Variable	Description	Fixed (bias) (B_x)	Random error (R_x)	Total uncertainty	Unit
1	T_oa	Outdoor air dry-bulb temperature	0.45	0.19	0.49	°C
2	T_ra	Return air temperature				°C
3	T_ma	Mixed air temperature				°C
4	T_ahc	Air temperature after heating coil				°C
5	T_sa	Supply air temperature				°C
6	ΔT_s, fan	Air temperature rises over supply fan	-	-	-	°C
7	T_HW_S	Supply hot water temperature	0.31	2.20	2.22	°C
8	Q_HC	Electric input for heating coil	This is a variable from synthetic data			kW
9	V_ma	Mixed air volumetric flow rate	222.0	506.11	552.65	L/s
10	V_ra	Return air volumetric flow rate	222.0	229.84	319.55	L/s
11	V_oa = (V_ma – V_ra)	Outdoor air volumetric flow rate	222.0	355.98	419.53	L/s
12	Valve_HC	Heating coil valve position	-	-	2	%

This thesis presents the results of proposed MDFDD method # 1 and method # 2 with two experimental and synthetic data sets.

4.1. Data preprocessing and optimization of training data sets for model development

In the experimental data sets, always some outliers may exist which can be due to the transient operation in the system or some abnormal activities as the results of the systems operators. But, in the synthetic data, the data sets are fault free and can be used directly with less data preprocessing. Therefore, it is vital to preprocess the data sets in order to develop an optimized method for the scope of detection and diagnosis of the multiple dependent faults.

First, outliers in the measurements of all sensors are removed using statistical measures. Data sets are preprocessed for quality control, missing values, and normalization to mean and standard deviation of each variable (Equation (4.1)).

$$\mu_{(\text{dataset})} - 2 \times \sigma_{(\text{dataset})} < X < \mu_{(\text{dataset})} + 2 \times \sigma_{(\text{dataset})} \quad (4.1)$$

where, $\mu_{(\text{dataset})}$ is the average of the data set, $\sigma_{(\text{dataset})}$ is the standard deviation of the dataset, and X is the data point.

In addition, another process on the data set for deploying in the machine learning model development is the data normalizations using Equation (4.2). The normalization of the data set will increase the accuracy of the ML models results.

$$X_{\text{normalized}} = \frac{X - \min(X)}{\max(X) - \min(X)} \quad (4.2)$$

where: X is the value at time (t);

$\min(X)$ and $\max(X)$ = the minimum and maximum values of the selected data set

Once the data set is preprocessed, it is important to optimize the size of input data set in order to obtain results with less inaccuracies. In the following subsection the optimization of the length of input data set for the training of machine learning model is discussed.

4.2. Results from the artificial faults generation

4.2.1. Results of LSM for finding grey-box model coefficients:

The results of the LSM method for the development of grey-box models are reported in this section. In the Table 4.2 the statistical indices for the models' evaluation are provided.

The training data set is the first three days, and the test set is one following day for the training set over heating season from December 26, 2016 to January 29, 2017 including 15-min time interval. In the following Table 4.2, the grey-box models are developed using the experimental and synthetic data, separately, and the results are presented.

Table 4.2. Grey-box models for prediction T_ma and T_ahc using LSM

No.	Model	Training data set (288 data points (3-Days))					Test data set (96 data points (1-Day))	
		Parameter	Value	Unit	R^2 (%)	RMSE (°C)	R^2 (%)	RMSE (°C)
1	Equation (3.8) (Experimental data)	a	0.232	-	92.24	0.53	95.98	0.55
		b	0.747	-				
2	Equation (3.9) (Experimental data)	c	0.000227	-	45.84	1.07	86.95	1.78
		d	2.399075	s/L				
		e	0	-				
3	Equation (3.9) (Synthetic data)	c	0.038	C/kW	99.87	0.091	98.65	0.124
		d	0.345	s/m^3				
		e	1.037	-				

Two cases of artificial faults are presented as examples in this section for both experimental and synthetic data:

Case 1: only values (measurements) of T_ma are replaced with artificial faults, which are generated with a linear positive bias of 0.5°C starting on January 19 at 12:01 a.m. followed by a ramp of 0.02356 C/time step until January 21 at 12:00 a.m., for 48 hours. (Figure 4.3).

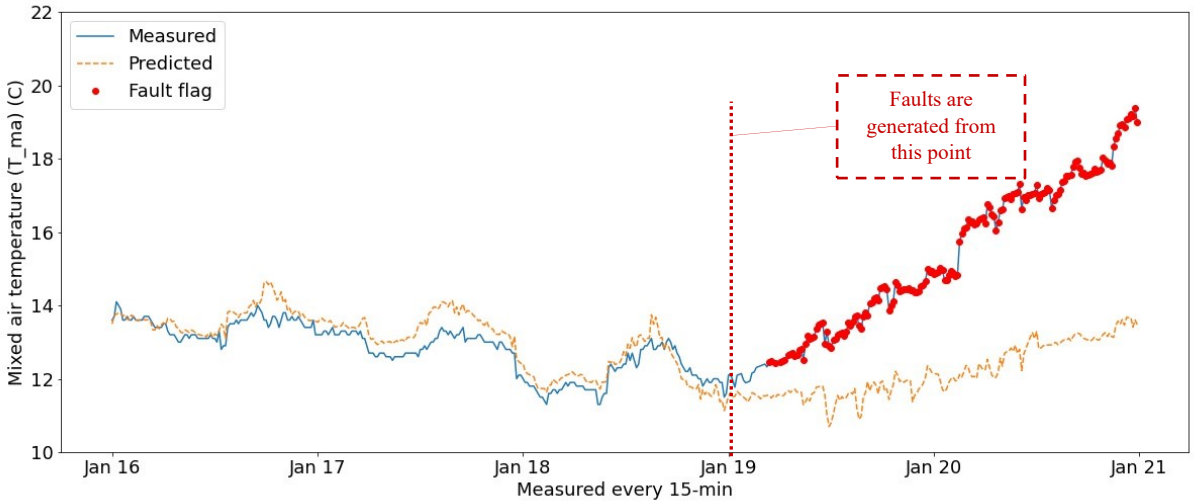

Figure 4.3. Measurements with artificial faults and predictions of T_ma.

Case 2: Only measurements of T_oa and T_ra are replaced with artificial faults, which are generated with linear positive bias of 0.5°C starting on January 19 at 12:01 a.m. followed by a ramp of 0.02356 C/time step until January 21 at 12:00 a.m., for 48 hours (Figure 4.4 and Figure 4.5).

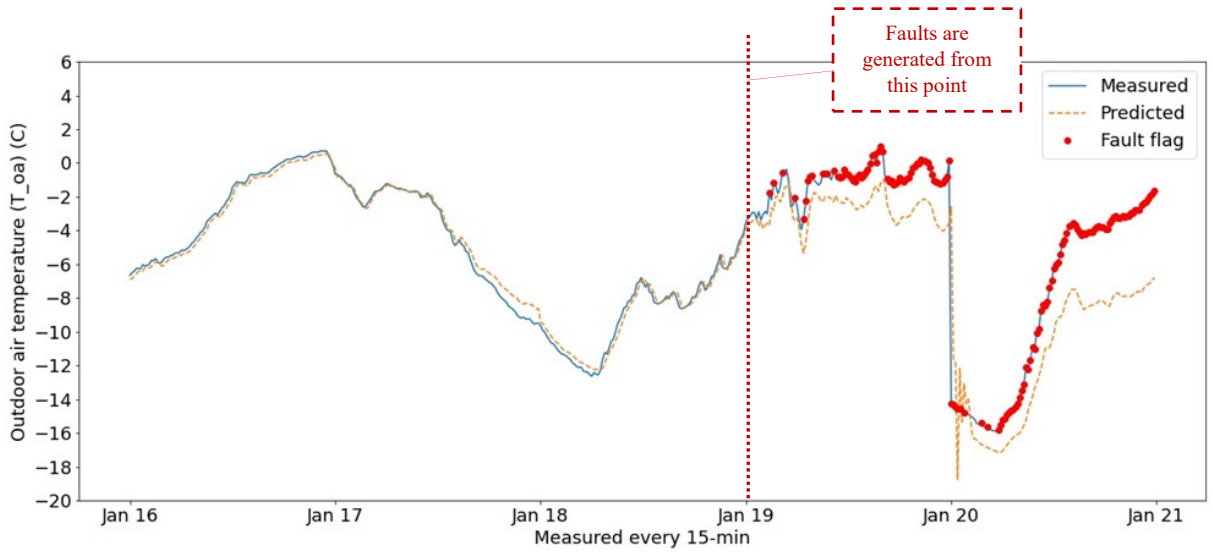

Figure 4.4. Measurements and predictions of T_oa with artificial faulty data

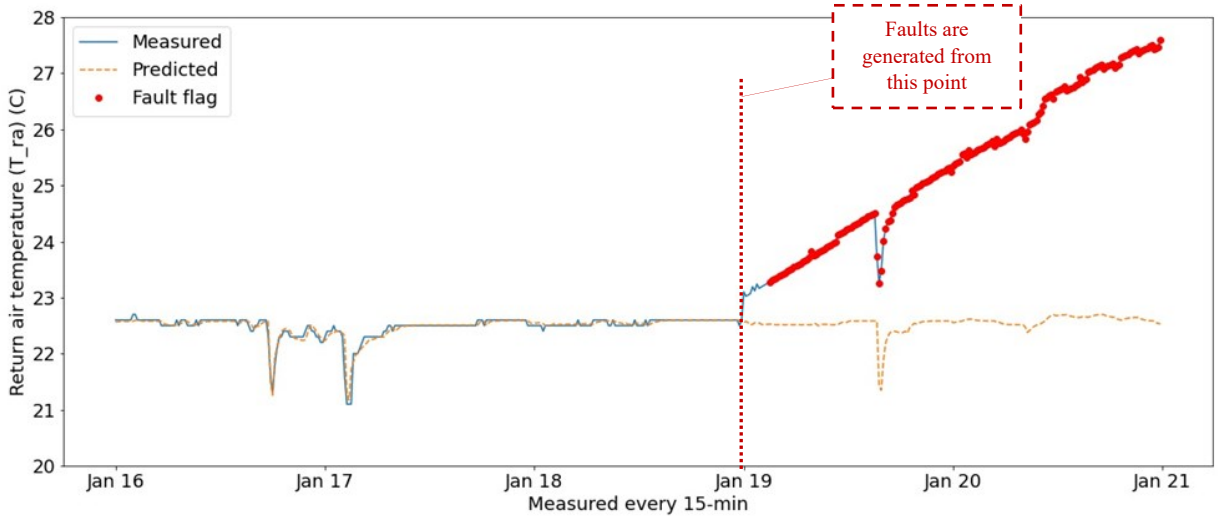

Figure 4.5. Measurements and predictions of T_ra with artificial faulty data.

In this thesis, the generated artificial faults of this section are used for the evaluation of the proposed MDFDD method in various applications.

4.2.2. Results of GA for finding grey-box models coefficients

The results of the GA method for the grey-box model development are provided in this section.

The optimum setting of the GA algorithms is selected as follow; the population size (n_p) is set to 100, the crossover probability (Pc) is 0.9, the mutation probability (Pm) is 0.02, the elitism ratio is 0.05, with 250 maximum number of generations (G_{max}). These values have found over trial-and-error approach.

The evaluation of the GA for the development of the grey-box model is reported in the Table 4.3.

Table 4.3. Grey-box models development for prediction of T_ma and T_ahc using GA

No.	Model	Training data set (288 data points (3-Days))					Test data set (96 data points (1-Day))	
		Parameter	Value	Unit	R^2 (%)	RMSE (°C)	R^2 (%)	RMSE (°C)
1	Equation (3.8) (Experimental data)	a	0.535	-	78.16	0.68	68.16	0.75
		b	0.465	-				
2	Equation (3.9) (Experimental data)	c	0.352	-	85.36	1.71	80.36	2.67
		d	4.177	s/L				
		e	1.201	-				

4.2.1. LSM versus GA methods for finding the models coefficients

According to the results of the LSM and the GA models for finding the parameters for the development of the grey-box model, it can be concluded that LSM is simpler in development and obtaining the parameters for the grey-box model. Moreover, the RMSE values for the grey-box models which have been developed by LSM for the prediction over test set is lower than the use of GA. Therefore, in this thesis, for the generation of the artificial faults using the grey-box model, the LSM method is applied for finding the grey-box models coefficients.

5. Application of proposed methods

The proposed methods for the detection and diagnosis of the multiple dependent faults are evaluated using different data sets (experimental and synthetic) as described in the above sections.

The first application deploys the method # 1 on the experimental and synthetic data, and the results are provided in Section 5.1.

The method # 2 is evaluated using the experimental data, and the results are reported in the Section 5.2.

5.1. Application # 1: Method # 1 applied on experimental and synthetic data

In this section, the experimental data over the heating season is used on the method # 1 for the MDFDD. The proposed ML models are developed on the experimental data and the fault detection and diagnosis steps are discussed in the following sections. Moreover, the optimization process for the ML models' development is discussed.

5.1.1. Optimization of input data set

In this thesis, the size of training set and the models hyperparameters are optimized to develop a more accurate model. The hyperparameters for the SVR model are δ, C, γ (Section 3.2.1). The hyperparameters for the development of optimized RNN models are the input time lags, size of training set, number of hidden layers, number of hidden neurons, and the dropout regularization ratio.

The developed ML models are optimized using RandomizedSearchCV tool [71]. This is a tool in the Scikit-learn package of Python, which randomly selects the hyperparameters out of the values and options assigned by the user, to obtain the optimum ML models for T_ma and T_ahc over k-fold cross validation.

The following steps are applied for the optimization of training data set size:

a) Let's assume, for the purpose of explanation, the length of training data set is of three days including 288 data points, from January 13 to January 15 (Figure 5.1), and is tested with data of January 16 (96 data points). A new training data set is selected, with same length of three days, by applying the sliding window technique, from January 14 to January 16, and tested with data of January 17. In all, the sliding window moves over six consecutive days. Hence, the 6-fold cross validation uses six different training data sets. The average RMSE value of predictions of T_ma over the corresponding testing data set is 0.41°C. By using a similar approach, the average RMSE value of predictions of T_ahc is 0.21°C.

~~b)~~ Results from different training data sets with lengths of 288, 480, 672, 864, 1,056, and 1,248 data points measured at 15-min interval are compared. The optimum length of training data set for the development of ML models of T_ma and T_ahc is composed of 288 data points. The comparison proved that by selecting data from three training days, reason for selecting three days, would be obtaining the good prediction performance over the test set with RMSE value smaller than sensor uncertainty. In addition, less computation time and storage memory would be used.

c) The RandomizedSearchCV tool, including 10 cross-validation and 30 times for the number of iterations, is applied to obtain the optimum values for the hyperparameters of the SVR model for the T_ma and T_ahc. The optimum values for T_ma are C = 15.32 and $\gamma = 0.015$; and for T_ahc are C = 12.87 and $\gamma = 0.057$; and the kernel is set to RBF for all SVR models.

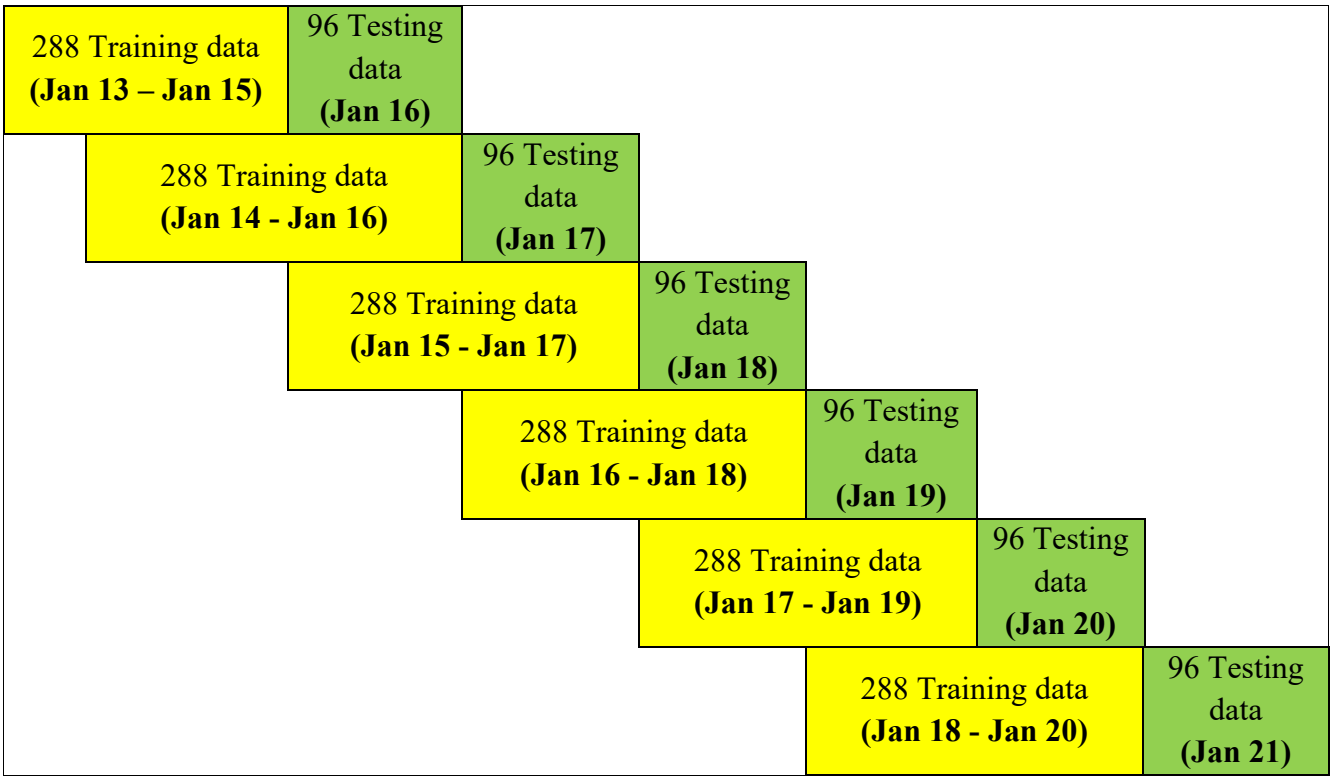

Figure 5.1. Training and testing data sets with sliding window technique from January 13 to January 21 for the prediction of T_ma and T_ahc.

5.1.2. Development of machine learning models

In this section the results of the developed ML models for the scope of MDFDD are reported in this and following sections.

Firstly, Table 5.1 reports the average statistical indices for the prediction of T_ma by using the SVR model over six consecutive testing days (Jan 16 to Jan 21).

Selection of threshold in this thesis is a novel approach which has not been done before. Since the ML models for the prediction of the target value have inaccuracies, therefore, the sensor uncertainty is combined with the ML model inaccuracy to define a threshold. The root-mean-squared-error (RMSE) in this thesis is considered as the inaccuracy of the ML model.

Therefore, in this section for the T_ma, the threshold of 0.90 °C for fault detection is set by considering the sensor uncertainty (0.49 ºC) and RMSE of model prediction (0.49 + 0.41 = 0.90 °C).

The comparison of measured and predicted values of T_ma over testing data set is illustrated in Figure 5.2. A few fault symptoms are detected on January 20, when the residual of measured and predicted values of T_ma exceeds the threshold of 0.90°C. However, the anomaly of measurements is detected only for about 75 minutes, and then the measurements return to normal values. Most likely this symptom was created by staff entering the AHU for maintenance purpose. Therefore, no faults of T_ma are detected under normal operation conditions.

Table 5.1. Prediction performance of SVR model of T_ma over training data set of three days and average testing results over the next six days.

Target	Input variables	Prediction performance of model over **training** data set (288 data points)					Average of 6-fold cross-validation of prediction performance over **testing** data set of one-day (96 data points)			
		R^2 (%)	RMSE (°C)	MAPE (%)	MBE (°C)	ME_{max} (°C)	RMSE (°C)	MAPE (%)	MBE (°C)	ME_{max} (°C)
T_ma	T_oa, T_ra, V_ma, V_ra, V_oa	98.33	0.31	0.43	0.04	1.51	0.41	2.63	0.34	0.97

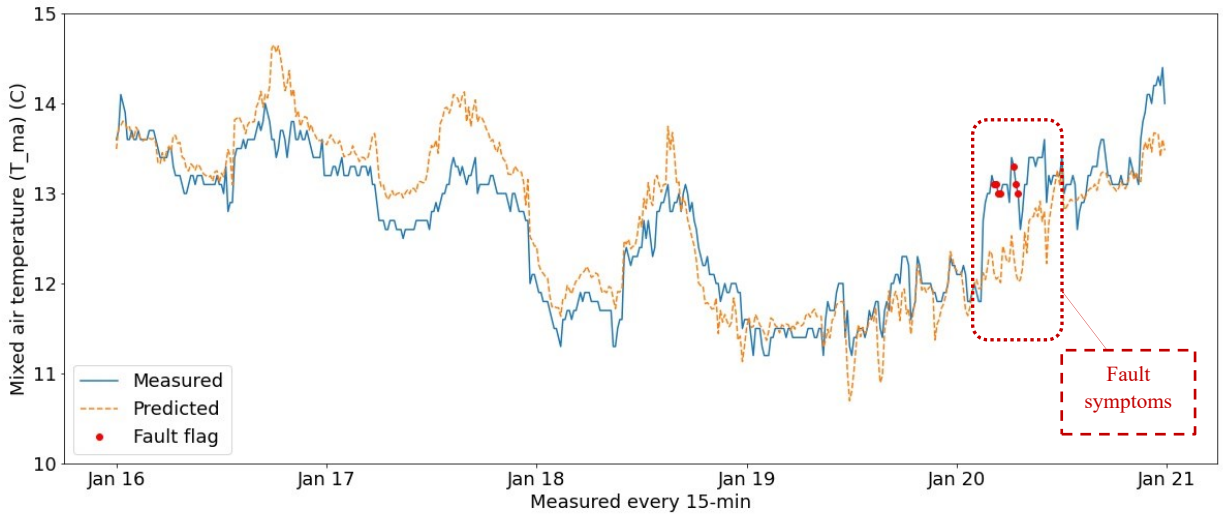

Figure 5.2. Measurements versus predictions of T_ma over testing data set under normal operation conditions

Table 5.2 reports the average statistical indices from the prediction of T_ahc over six consecutive testing days (Jan 16 to Jan 21). The comparison of measured and predicted values of T_ahc over testing data set is illustrated in Figure 5.3. Since the residual of measured and predicted values of T_ahc does not exceed the threshold of 0.70°C (0.49 + 0.21 = 0.70 °C), no faults of T_ahc are detected.

Table 5.2. Prediction performance of SVR models of T_ahc over training data set of seven days and testing over the next day.

Target	Input variables	Prediction performance of model over **training** data set (288 data points)					Average of 6-fold cross-validation of prediction performance over **testing** data set of one-day (96 data points)			
		R^2 (%)	RMSE (°C)	MAPE (%)	MBE (°C)	ME_{max} (°C)	RMSE (°C)	MAPE (%)	MBE (°C)	ME_{max} (°C)
T_ahc	T_ma, V_ma, T_HW_S, Valve_HC	98.83	0.16	0.14	0.02	0.57	0.21	1.08	0.17	0.60

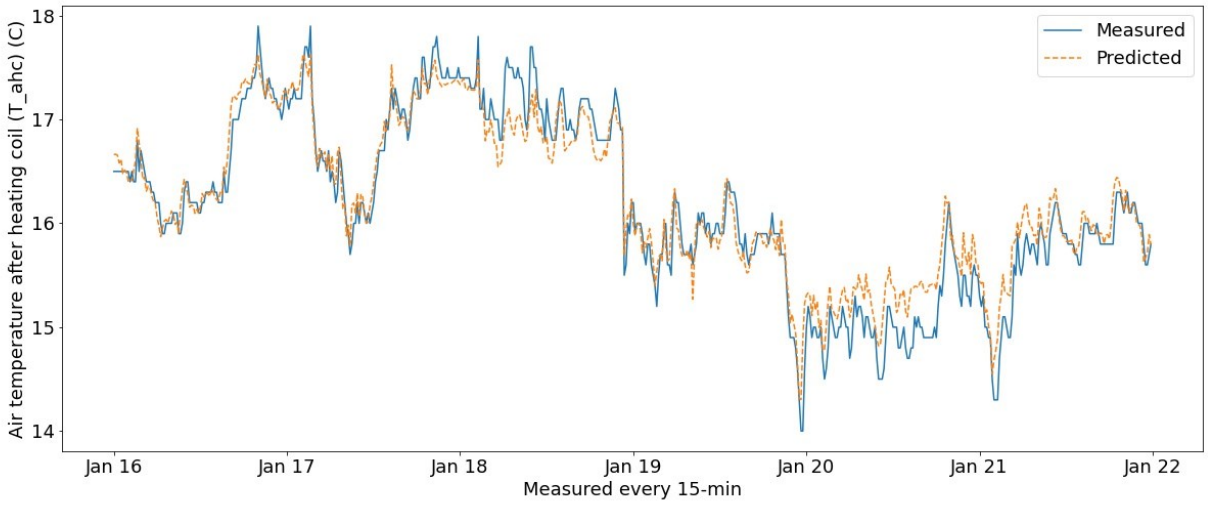

Figure 5.3. Measurements and predictions of T_ahc over testing data set under normal operation conditions

Once the ML models for the fault detection step are developed and evaluated successfully, the proposed ML models for the fault diagnosis step are developed and evaluated using the experimental data. These results will be discussed below.

The RNN is the ML model which is developed for the prediction of the regressor values for the fault diagnosis scope. Statistical indices of predictions of RNN models over normal operation conditions, and the threshold for each sensor are presented in Table 5.3. The results revealed well performance of the RNN model for the prediction purpose of the regressors. Therefore, the next steps will be implemented.

Table 5.3. Prediction performance of Recurrent Neural Network for selected regressors sensors using the experimental data

Regressor sensor at time 't'	Average of 6-fold cross-validation of prediction performance over testing data set of one day (96 data points)					Threshold for fault detection
Variable	Unit	RMSE	MAPE (%)	MBE	ME_{max}	ε = sensor uncertainty + RMSE
T_oa	°C	0.89	100.03	1.98	3.42	1.38
T_ra		0.08	0.91	0.20	0.38	0.57
T_ma		0.65	14.31	1.67	2.61	1.14
T_ahc		0.31	4.54	0.85	1.11	0.80
T_HW_S		0.70	4.75	1.99	2.74	1.00
Valve_HC	%	1.41	10.72	4.61	3.79	3.41

For the development of the models, different hyperparameters are optimized for each ML model as explained below:

a. **ANN:** Models include five and four input nodes for the T_ma and T_ahc, respectively, and the selected optimized hyperparameters for the ANN model are as follows: the single hidden layer neurons = 3, and the sigmoid activation function.

b. **Decision tree regression:** Using the RandomizedSearchCV including 30 iterations and 10 cross-validation the optimized max_depth of the tree for the T_ma and T_ahc were obtained to be 2.65 and 7.79, respectively.

c. **Homogeneous random forest:** With the use of application of RandomizedSearchCV, the optimized hyperparameters are 20 trees and max_dept=2.64 for T_ma, and one tree and max_dept=12.11 for T_ahc.

5.1.3. Comparison of results between different ML models

In order to evaluate the ML models, various models are developed and compared. The results are discussed in this section, with experimental data (Table 6.4) and synthetic data (Table 6.5), respectively.

The results of developed SVR model are compared with the other three ML models: the artificial neural network (ANN), decision tree regression, and random forest regression for the fault detection purpose.

The results presented in Table 5.4 reveal the very good prediction performance of SVR with the consideration of the RMSE values for the MDFDD scope in the AHU.

Table 5.4. Results for the performance of the other ML models for the prediction of T_ma and T_ahc, using 288 data points (3-days) of training data (experimental data)

Target	Input Variables	Machine learning model Developed with 3-days (288 data points) training set	Average of 6-fold cross-validation of prediction performance over test data set of 24 hours with (96 data collected every 15-min)			
		Model	RMSE (C)	MAPE (%)	MBE (C)	ME_{max} (C)
T_ma	T_oa_db, T_ra, V_ma, V_ra, V_oa	SVR	0.41	2.63	0.34	0.97
		ANN	0.40	2.72	0.35	0.86
		Decision Tree	0.60	4.04	0.53	1.17
		Random forest	0.58	3.69	0.48	1.03
T_ahc	T_ma, V_ma, HW_T_S, Valve_HC	SVR	0.21	1.08	0.17	0.60
		ANN	0.12	0.45	0.07	0.49
		Decision tree	0.13	0.30	0.05	0.62
		Random forest	0.18	0.59	0.09	0.74

In order to evaluate the proposed method on another data set, the synthetic data generated by eQuest are applied and the results are presented in the following tables.

Table 5.5: Results for the performance of the other ML models for the prediction of T_ma and T_ahc, using including 288 data points (3-days) of training data (synthetic data)

Target	Input Variables	Machine learning model Developed with 3-days (288 data points) training set	Average of 6-fold cross-validation of prediction performance over test data set of 24 hours with (96 data collected every 15-min)			
		Model	RMSE (C)	MAPE (%)	MBE (C)	ME_{max} (C)
T_ma	T_oa_db, T_ra, V_ma, V_ra, V_oa	SVR	0.075	0.41	0.06	0.13
		ANN	0.053	0.28	0.04	0.10
		Decision tree	0.077	0.32	0.05	0.16
		Random forest	0.135	0.60	0.09	0.39
T_ahc	T_ma, V_ma, Q_HC, Valve_HC	SVR	0.151	0.51	0.13	0.28
		ANN	0.130	0.43	0.11	0.24
		Random forest	0.416	1.26	0.31	1.04
		Decision tree	0.473	1.50	0.37	1.07

The RNN and the FFNN models are developed using the synthetic data and the results are provided in Table 5.6 and Table 5.7.

Table 5.6: Predictions of the Recurrent Neural Network using synthetic data

Target		RNN architecture with 1 hidden layer		Prediction performance over validation data set for 24 hours					
Variable	Time step	Input	Time lag	Unit	R^2 (%)	RMSE	MAPE (%)	MBE	ME_{max}
T_oa	t	T_oa	t-1, t-2…, t-10	C	49.42	1.39944	65.4873	0.9666	4.2991
T_ra		T_ra		C	84.90	0.02076	0.0692	0.0151	0.0557
V_ma		V_ma		m^3/s	100.00	0.00025	0.0091	0.0002	0.0002
V_ra		V_ra		m^3/s	100.00	0.00129	0.1913	0.0013	0.0013
V_oa		V_oa		m^3/s	100.00	0.00190	0.0935	0.0019	0.0019
Q_HC		Q_HC		kW	80.84	1.68295	3.6108	1.1756	4.7031
T_sa		T_sa		C	71.97	0.56523	1.5654	0.4175	1.2236

Table 5.7: Feedforward Neural Networks prediction results using synthetic data.

Target		FFNN architecture with 1 hidden layer		Prediction performance over validation data set for 24 hours					
Variable	Time step	Input	Time lag	Unit	R^2 (%)	RMSE	MAPE (%)	MBE	ME_{max}
T_oa	t	T_oa	t-1, t-2…, t-5	C	67.80	1.36083	52.1779	0.9448	4.9319
T_ra		T_ra		C	85.75	0.020239	0.0742	0.0162	0.0436
V_ma		V_ma		m^3/s	100	0.000527	0.0195	0.0005	0.0005
V_ra		V_ra		m^3/s	100	0.001476	0.2194	0.0015	0.0015
V_oa		V_oa		m^3/s	100	0.000908	0.0447	0.0009	0.0009
Q_HC		Q_HC		kW	84.13	1.700485	3.8161	1.2275	4.8029
T_sa		T_sa		C	83.76	0.307907	0.8015	0.2178	0.7697

Both FFNN and RNN models give acceptable prediction performance in order to be deployed for the fault diagnosis approach. For example, for prediction of T_ra the RMSE and MAPE are approximately 0.0202°C, 0.074% using FFNN model, and 0.0207°C, 0.069% using the RNN model.

The ML models using the hourly synthetic data have better prediction performance compared with the experimental data of higher frequency (e.g., 15-min time step), as the synthetic data extracted from the eQuest simulation program are fault free, and no short-term variation.

5.1.4. Comparison of ML models results on experimental and synthetic data

The results of the developed ML models on the experimental and synthetic data are discussed here. The accuracy of the ML models on the synthetic data is higher than the experimental and comes after the existence of noises in the experimental data compared with the synthetic data. Synthetic data is fault free and the ML model can predict the target values with higher accuracy.

In both data sets, the SVR model performs better with respect to the RMSE values for the prediction of the T_ma values, while in the prediction of the T_ahc, the ANN will perform slightly

better compared with the SVR. In addition, the computation time for the development and optimization of SVR is three times less than ANN.

All in all, the developed ML models perform well for the prediction of target values in the sensors of the T_ma and T_ahc with use of both experimental and synthetic data.

5.1.5. Results and discussion on fault detection and diagnosis

Since the collected data set is fault free, therefore, the artificial faults are inserted into the data set using the approach described in Section 3.5. The proposed SVR model is applied for the prediction of T_ma. The residuals of measured (artificial faults) values and predicted values of T_ma are calculated (Figure 5.4), and if the residuals exceed the threshold of 0.90°C, the fault is detected. The confusion matrix (Table 5.8) shows that the fault symptom was detected correctly for 253 data points, and only nine fault symptoms were wrongly detected as normal operation, out of total 262 faulty data points. Out of 314 normal data points, 308 data points were detected correctly as normal, and six points were wrongly detected as fault symptoms. The accuracy, precision, and sensitivity of the SVR model have values greater than 96% (Table 5.9). The results revealed, the SVR model detects well the fault symptoms of T_ma sensor.

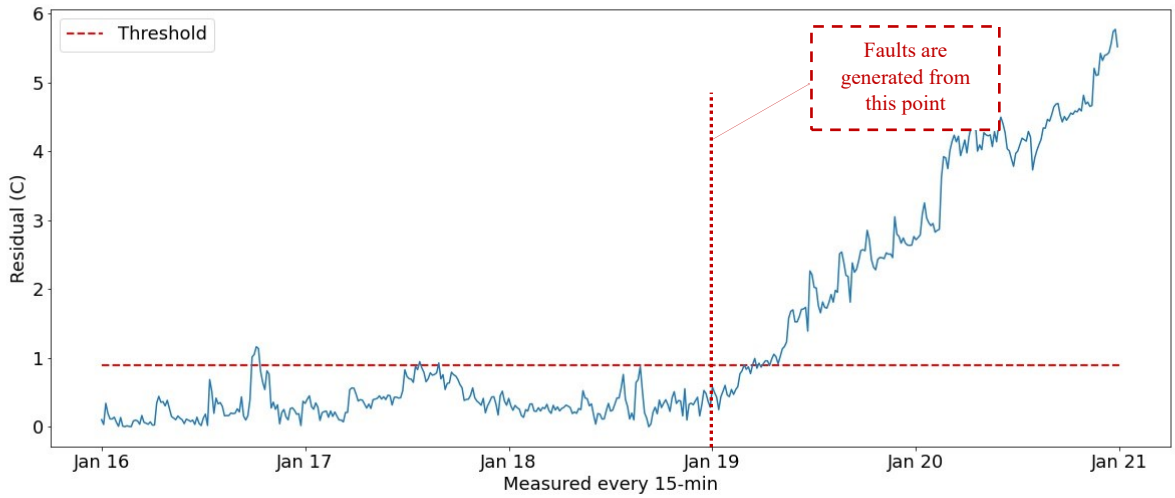

Figure 5.4. Residual between measurements and predictions of T_ma including artificial faulty data.

Table 5.8: Confusion matrix for fault detection of T_ma

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	308	9
	Faulty (1)	6	253

Table 5.9. Accuracy, precision, and sensitivity of the SVR models for detecting artificial faults of T_ma.

Target	Prediction performance of model over testing data set (6-days, 576 data points)		
	Accuracy (%)	Precision (%)	Sensitivity (%)
T_ma	97.40	96.56	97.68

The residuals of measured (artificial faults generate with grey-box model) values of T_ma, and predicted values T_ma,p are computed; when the residuals exceed the threshold ε=0.90°C, the fault symptom is detected (Figure 5.5).

If another case is considered, when artificial errors are inserted for T_ma, T_oa, and T_ra, the residual of measurements T_ma (with artificial faults) and predicted T_ma,R will not be zero. In that case, all three sensors (T_ma, T_ra, T_ma have fault symptoms).

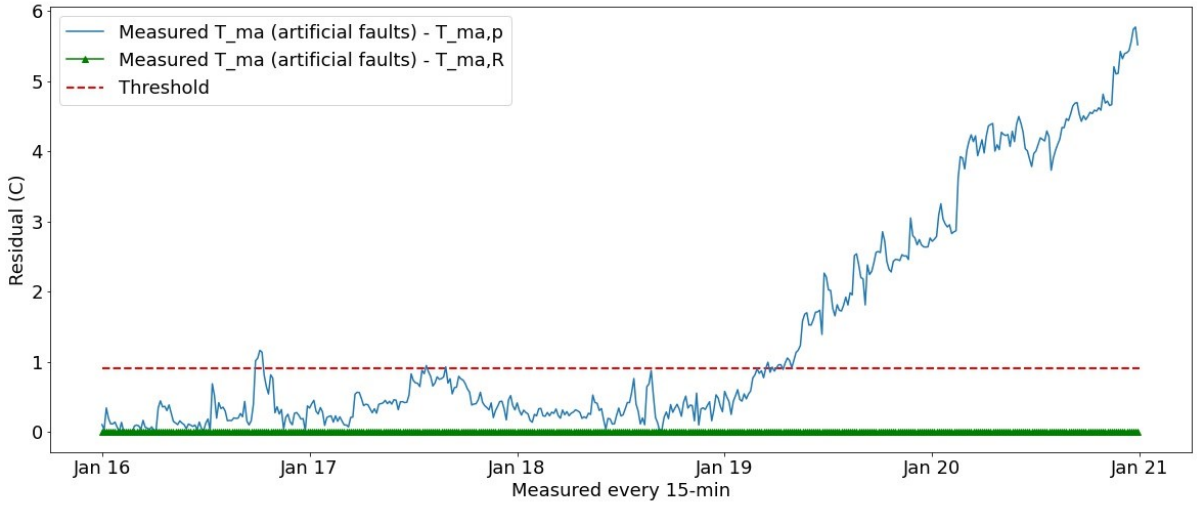

Figure 5.5. Residual between measurements of T_ma including faults and predictions of T_ma

Detection of fault symptoms

The T_ma,p value expected to be measured under normal operation (Figure 5.6) is predicted by the SVR model. Since the residual of measured values (i.e., due to artificial faults) of T_ma and predicted values T_ma,p exceed the threshold ε=0.90°C, a fault symptom of T_ma sensor is detected (Figure 5.7). Is the sensor T_ma faulty, or the T_ma sensor is correct under the influence of regressor sensors T_oa and T_ra?

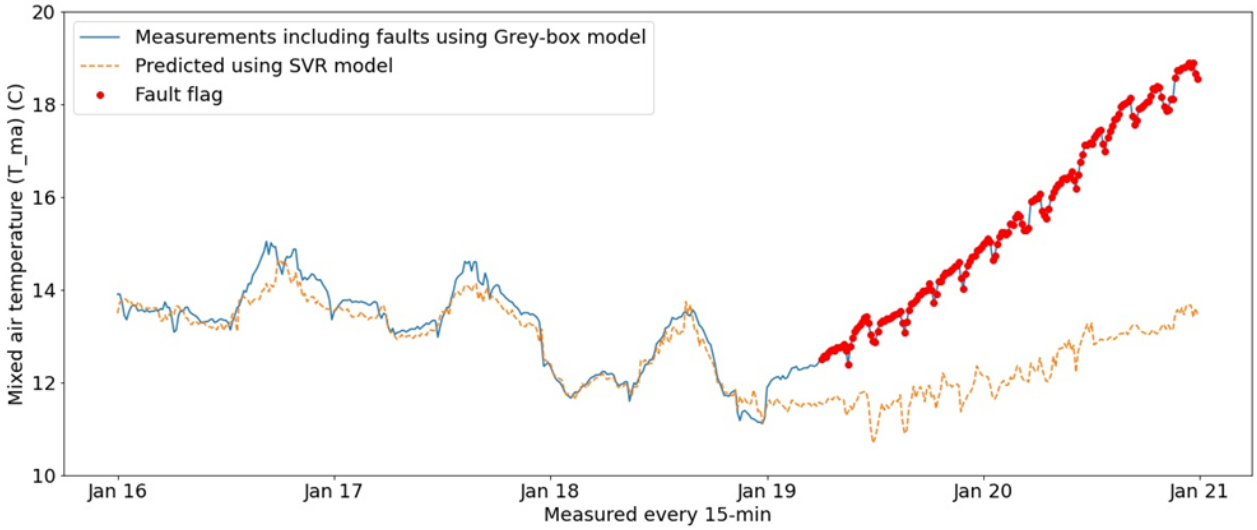

Figure 5.6. Measurements with artificial faults of T_ma and predictions of normal operation of T_ma

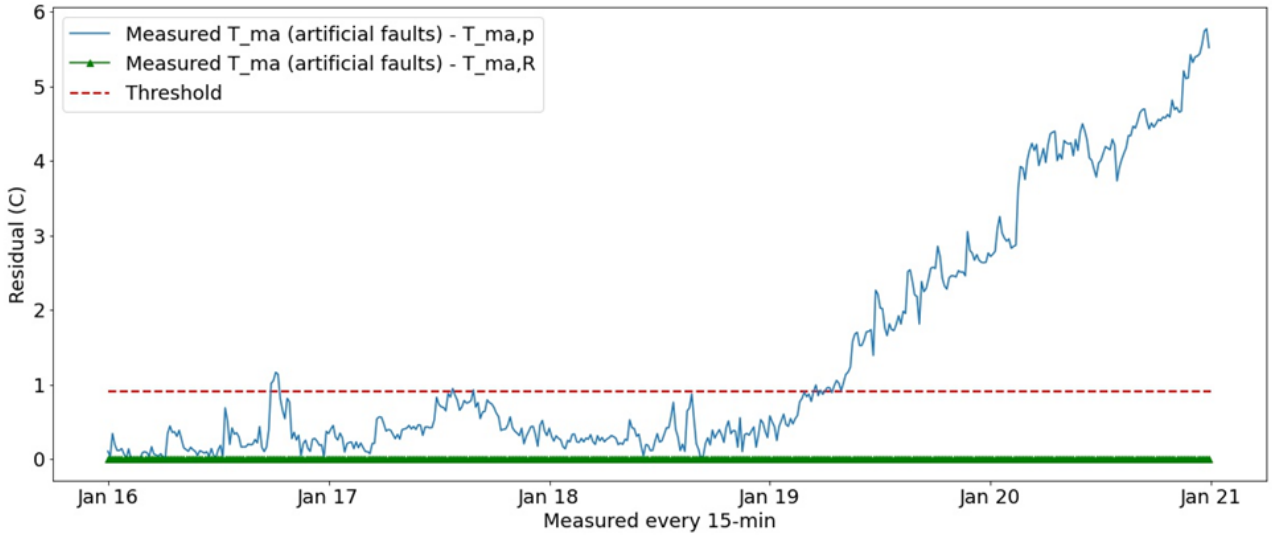

Figure 5.7. Residual between measurements of T_ma and predictions of T_ma

Diagnosis of faults

1) When the fault symptom of T_ma sensor is detected, the next step consists in the analysis of regressor sensors T_oa and T_ra. RNN models predict the values of T_oa,p and T_ra,p under normal operation conditions. When the residuals exceed the threshold (i.e., Res_oa = abs(T_oa - T_oa,p) > ε, and Res_ra = abs(T_ra - T_ra,p) > ε), fault symptoms of T_oa and T_ra are detected (Figure 5.8 and Figure 5.9) and in the ruled-based technique these detected fault symptoms are considered for the fault diagnosis step.

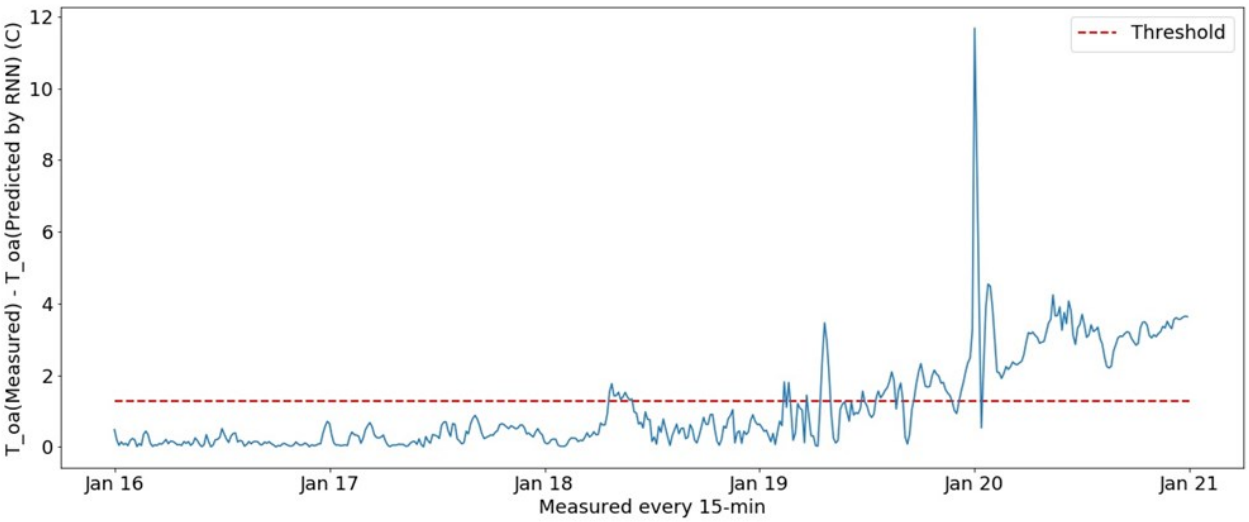

Figure 5.8. Residual between measurements and predictions of T_oa using RNN model

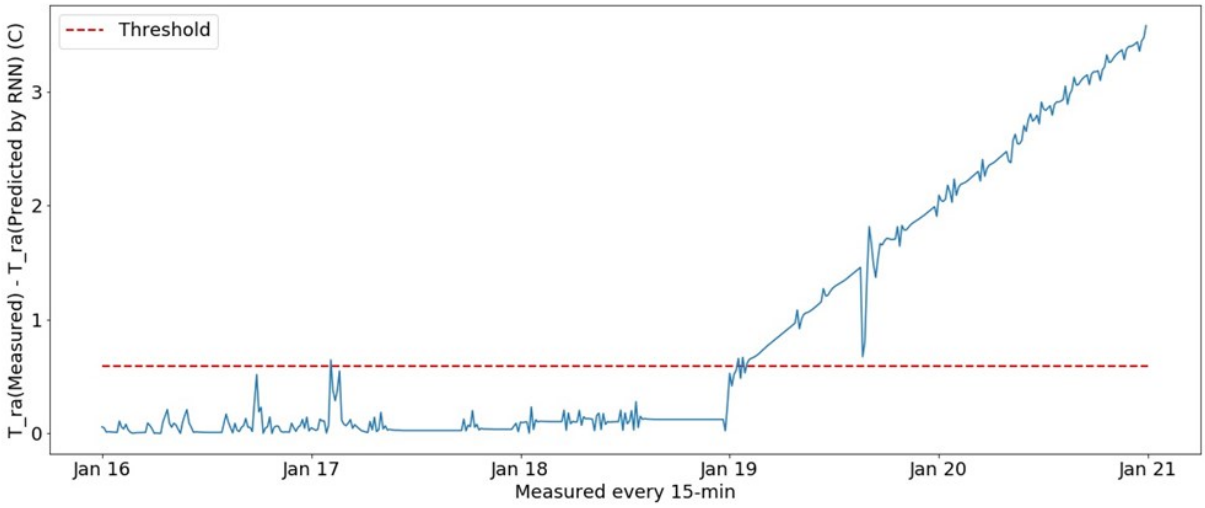

Figure 5.9. Residual between measurements and predictions of T_ra using RNN model

Out of 160 artificial faults of T_oa, 138 faults are detected correctly (Table 5.10). Out of 189 artificial faults of T_ra, 187 faults are detected correctly (Table 5.11). The accuracy, precision, and sensitivity of the RNN models have values greater than 86% for T_oa, and around 99% for T_ra (Table 5.12).

Therefore, T_oa and T_ra sensors are detected as faulty, which corresponds to artificial faults inserted in the data set.

2) The expected output of T_ma,R under the influence of faulty T_oa and T_ra sensors is calculated by using a grey-box model (Equation (3.8), with coefficients a =0.232 and b = 0.747 (Table 4.2).

3) Since the residual of measured values (i.e., due to artificial faults) of T_ma and predicted values T_ma,R does not exceed the threshold ε, the T_ma sensor is not faulty (Figure 5.7).

4) According to rule A.c. (Section 4.3), if T_oa and T_ra sensors are faulty, but the residual Res_ma = abs(T_ma - T_ma,R) < ε, then T_ma sensor is not faulty; only T_oa and/or T_ra sensors are faulty.

In conclusion of this example, without this approach, all three sensors T_ma, T_oa, and T_ra, would be wrongly considered as faulty.

If in another case, the residual of measurements and predictions of T_ma, T_oa, and T_ra, respectively, exceed the threshold, we can conclude that all three sensors (T_ma, T_ra, T_ma) have fault symptoms.

Table 5.10: Confusion matrix for fault detection of T_oa

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	311	22
	Faulty (1)	9	138

Table 5.11: Confusion matrix for fault detection of T_ra

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	289	2
	Faulty (1)	2	187

Table 5.12. Accuracy, precision and sensitivity of the RNN models for MDFDD.

Target	Prediction performance of model over testing data set (6-days, 576 data points)		
	Accuracy (%)	Precision (%)	Sensitivity (%)
T_oa	93.54	86.25	93.88
T_ra	99.16	98.95	98.95

In this section of the thesis, tested the initial hypothesis, and concluded that a combination of machine learning (ML) models using BAS trend data, and rule-based models is successful for the multiple dependent faults detection and diagnosis (MDFDD) in an AHU. A multiple dependent fault can have an impact on one or more other faults. This thesis presented a novel method that guides for faults search, by using the information and operation flow between sensors, and between sensors and devices. This approach was not found in any other publication.

SVR models along with other ML models (ANN, DT, RF) were used for the prediction of two target variables, the mixed air temperature (T_ma) and air temperature after heating coil (T_ahc). If the residual of measured values and predicted values exceeds the threshold, the fault symptom is detected.

The RNN models were used for the prediction of regressor sensors values (e.g. T_oa, T_ra, HW_T_S, Valve_HC). Rules-based models were used for the diagnosis of faults.

The proposed method was tested with measurements from BAS under normal operation, and with artificial faults inserted in the measurements data file. The results reveal the good performance of the proposed method for the multiple dependent faults of an AHU.

5.1.6. Comparison with another method

For comparison, this section presents the detection of faults by RNN models applied to all three sensors T_ma, T_ra, T_ma, but without any information about the relationship between sensors (Table 5.13).

Table 5.13. Prediction performance of Recurrent Neural Network for selected regressors sensors

Regressor sensor at time 't'	Average of 6-fold cross-validation of prediction performance over testing data set of one day (96 data points)					Threshold for fault detection
Variable	Unit	*RMSE*	*MAPE* (%)	*MBE*	ME_{max}	ε = sensor uncertainty + RMSE
T_oa	°C	0.89	100.03	1.98	3.42	1.38
T_ra		0.08	0.91	0.20	0.38	0.57
T_ma		0.55	14.04	2.82	1.71	1.04

In addition of residuals obtained from the use of RNN models for T_oa and T_ra sensors (Figure 5.8 and Figure 5.9). Figure 5.10 shows the residual of T_ma obtained from RNN model applied to artificial faults. It can conclude that by using RNN models without any information between sensors, the results show that all three sensors are faulty, which is not true.

In conclusion, the proposed MDFDD method detects the faulty sensors, while the broad application of ML models to all sensors, without any information between the dependent sensors does not detect correctly the faulty sensors.

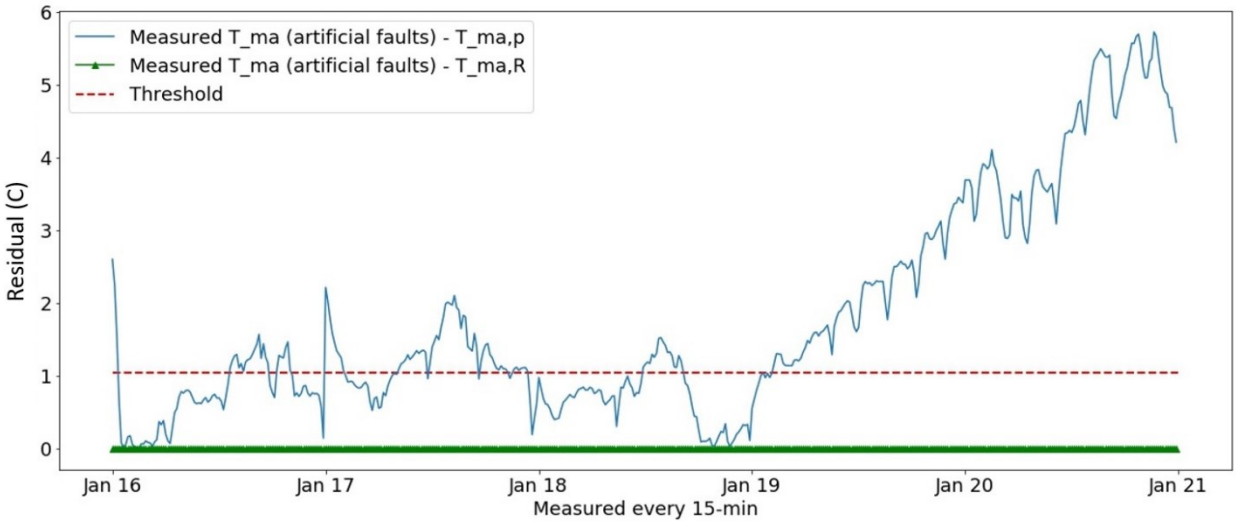

Figure 5.10. Residual between measurements of T_ma and predictions of T_ma using RNN model

5.2. Application # 2: Method # 2 applied on experimental and synthetic data

In this section the classification-based machine learning model is developed for the multiple dependent fault detection and diagnosis in the sensors of an AHU. The following machine learning models with classification approach are developed over optimization process for the models' hyperparameters selection. Support vector machine (SVM), K-Nearest Neighbor (KNN), and Naïve Bayes are developed in this thesis to evaluate their performances and then select the best model respect to the accuracy in detection and diagnosis of multiple dependent faults.

5.2.1. Development of classification machine learning models

The classification machine learning models are optimized through grid or random hyperparameter search approach. In this section, the hyperparameters search for the models' development and optimization are described. It is noteworthy to mention that over optimization process, the training and testing data sets are kept constants for all models to keep equal condition for each model.

The proposed ML models are developed using supervised methods which can be used for classification and regression purposes.

In this section, the hyperparameters selection for the optimization of the models is described. The parameter which is studied for all models is the input time-lag hours (maximum of ten time-lags).

The proposed ML models are developed using supervised methods which can be used for classification and regression purposes.

- **ANN:** The ANN [83-84] is developed using Tensorflow [73] and keras [72] libraries in Python 3.8 [54], with rectified linear unit (Relu) and sigmoid activation functions in the hidden and output layers, respectively, including 1000 epochs. For the shallow ANN, from 5 up to 30 hidden neurons are implemented, and according to the results for the highest accuracy of the MDFDD, 10 hidden neurons are selected as the optimum number of neurons.

- **Deep ANN:** In the Deep ANN, the same procedure as the development of ANN model is conducted, however more than one hidden layer is defined in the algorithm. The suggested 10 hidden neurons out of the shallow ANN are selected and deployed in two up to seven hidden layers. The results revealed, two hidden layers with 10 hidden neurons in each layer has the optimum MDFDD accuracy.

- **SVM:** The support vector classification (SVC) algorithm [94-96] out of libsvm library from the scikit-learn library [71] is applied to classify the target outputs. The SVC has the capability in conducting multi-class and binary classifications on the data set.

- **K-NN:** The K-neighbors classifier algorithm [78] from the nearest-neighbor library from the scikit-learn is applied for the K-NN classification model development. For the K-NN, out of 1 up to 30 nearest neighbours, selection of 13 neighbors in the K-NN model will reveal the highest MDFDD.

- **Decision Tree:** the decision tree (DT) is a supervised ML model which is used for prediction and classification purpose [106-107], using the decision rules emanated from the data features. In this paper, the decision tree classifier algorithm with the entropy criterion from the scikit-learn library is applied.

- **Random Forest:** The Random Forest model is an estimator which select and fit a number of decision tree classifier on different samples of data set, and making the average in order to improve the accuracy of classification [79]. The random forest in this thesis is developed with 50 up to 600 trees using the scikit-learn library, and the optimum Random Forest model is implemented with 400 number of trees.

- **PCA:** The PCA is used for the dimension reduction of the data sets [160], then these data sets are applied using scikit-learn in the ANN and Random Forest for the MDFDD. The optimum number of principal components for combination with the ANN and Random Forest will be 8 based on the revealed results on the MDFDD accuracy.

- **Naïve Bayes:** The Naïve Bayes uses the Bayes theorem with the naive assumption [103-104]. The relationship between the target and the features is developed using the Bayes theorem. The Gaussian Naïve Bayes using scikit-learn is implemented in this ML model for classification of the faulty and normal targets for MDFDD.

5.2.2. Results and discussion on fault detection and diagnosis on experimental and synthetic data

In this method, the generated artificial faults are used. Once the fault is inserted in the file of T_oa, the relationship between the value of regressor sensor and target sensor (e.g., T_ma) must also be added.

The support vector classification (SVC) algorithm [94-95, 150] out of libsvm library from the scikit-learn library is applied to classify the target outputs. The SVC has the capability in conducting multi-class and binary classifications on the data set.

The confusion matrix (Table 5.14) shows that the fault symptom was detected correctly for 179 data points, and only 13 fault symptoms were wrongly detected as normal operation, out of total 192 faulty data points. Out of 288 normal data points, 288 (100%) data points were detected correctly as normal. The accuracy, precision, and sensitivity of the SVM model have values greater than 93% (Table 5.17). The results revealed, the SVM model detects well the fault symptoms of T_ma sensor.

Table 5.14: Confusion matrix for fault detection of T_ma

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	288	13
	Faulty (1)	0	179

Table 5.15: Confusion matrix for fault detection of T_oa

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	480	0
	Faulty (1)	0	0

Table 5.16: Confusion matrix for fault detection of T_ra

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	480	0
	Faulty (1)	0	0

Table 5.17. Accuracy, precision, and sensitivity of the SVM models for detecting artificial faults of T_ma.

Target	Prediction performance of model over testing data set (5-days, 480 data points)		
	Accuracy (%)	Precision (%)	Sensitivity (%)
T_ma	97.30	93.23	100

The proposed MDFDD approach is evaluated with comparison of the SVM model with two other ML models, KNN, and Naïve Bayes, and the results are presented in Table 5.18. The results reveal SVM performs better compared with other ML models for the purpose of single FDD.

Table 5.18.Accuracy, precision and sensitivity of the ML models over the testing data for T_ma on experimental data

No.	Machine learning model	Prediction performance of model over testing data set (5-days, 480 data points)		
		Accuracy (%)	Precision (%)	Sensitivity (%)
1	SVM	97.30	93.23	100
2	K-NN (30 Neighbors)	79.37	48.43	100
3	Naïve Bayes	73.75	48.96	77.05

The artificial faults are generated for the T_oa and T_ra sensors as described in Section 3.5. Then, the developed ML model is applied on the test data set to detect and diagnose the multiple dependent faults.

Out of 192 artificial faults of T_oa, the SVM model correctly detected 153 fault symptoms (Table 5.19). Out of 192 artificial faults of T_ra, the SVM model correctly detected 191 fault symptoms (Table 5.20).

Out of 190 artificial faults in T_oa and T_ra, the SVM model correctly detected 106 fault symptoms in T_ma (Table 5.21).

The accuracy, precision, and sensitivity of the SVM models have values greater than 70% for T_oa, around 69% for T_ra, and around 56% for T_ma (Table 5.22). Therefore, T_oa and T_ra sensors are detected and diagnosed as faulty sensors, which influence T_ma sensor as the faults' symptoms are observed.

Table 5.19: Confusion matrix for fault detection of T_oa

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	222	39
	Faulty (1)	66	153

Table 5.20: Confusion matrix for fault detection of T_ra

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	204	1
	Faulty (1)	84	191

Table 5.21: Confusion matrix for fault detection of T_ma

		Actual faults	
		Normal (0)	Faulty (1)
Predicted faults	Normal (0)	230	84
	Faulty (1)	60	106

Table 5.22. Accuracy, precision and sensitivity of the SVM models for MDFDD.

Target	Prediction performance of model over testing data set (5-days, 480 data points)		
	Accuracy (%)	Precision (%)	Sensitivity (%)
T_oa	78.12	79.69	69.86
T_ra	82.30	99.45	69.45
T_ma	70.00	55.79	63.85

The proposed MDFDD method is evaluated with comparison of the SVM model with two other ML models, KNN, and Naïve Bayes, and the results are presented in Table 5.23. In these results, the Naïve Bayes performs better for the scope of MDFDD.

Table 5.23. Accuracy, precision and sensitivity of the ML models over the testing data set (experimental data)

No.	Machine learning model	Target	Prediction performance of model over testing data set (5-days, 480 data points)		
			Accuracy (%)	Precision (%)	Sensitivity (%)
1	SVM	T_oa	78.12	79.69	69.86
		T_ra	82.30	99.45	69.45
		T_ma	70.00	55.79	63.85
2	K-NN (30 Neighbors)	T_oa	83.71	64.58	89.21
		T_ra	57.41	4.17	28.57
		T_ma	71.46	30.00	93.44
3	Naïve Bayes	T_oa	83.33	58.33	100
		T_ra	95.21	98.43	90.43
		T_ma	92.92	96.84	86.79

Once the analysis of the method # 2 using the experimental data is accomplished, the synthetic data is deployed to evaluate the proposed method fort he MDFDD. The proposed ML models are re-trained and developed again using the new synthetic data, and the details for the hyperparameters of the models and the results are provided below.

ML models are developed using Scikit learn [71], Keras [72] and Tensorflow [73] which are the open-source libraries in Python 3.8 [54].

If the absolute air temperature difference between the predicted normal temperature value, and sensor measurement is greater than sensor uncertainty (threshold) of 0.49°C, then the sensor is faulty. Otherwise, the sensor is normal.

The proposed developed ML models are compared with respect to the accuracy, precision, sensitivity (Equations (2.6) to (2.8)) [80], over the testing data set.

ML models with the highest accuracy of detecting faulty sensors based on results of Table 5.18 are:

1) SVM with two-hour time-lags (i.e., at time t, t-1and to t-2);
2) Deep ANN with two hidden layers and 10 hidden neurons in each layer, and with two-hour time-lags;
3) PCA with eight principal components, combined with Random Forest of 400 trees, and with one-hour time-lag of inputs; and,
4) Kernel SVM with one-hour time-lag.

These ML models have also the highest precision over testing data set.

Table 5.24: Accuracy, precision and sensitivity of the ML models using synthetic data over the testing data set

No.	Machine learning model	Optimum input time-lag (t-n)	Testing Set		
		n (Hour)	Accuracy (%)	Precision (%)	Sensitivity (%)
1	SVM	2	93.97	97.58	93.95
2	Deep ANN 2 Hidden layers, 10 Neurons	2	91.96	95.66	92.97
3	PCA with random forest	1	90.14	95.17	90.99
4	Kernel SVM	1	88.91	93.53	91.02
5	SVR	1	88.80	92.79	91.41
6	Random forest classification (400 trees)	1	88.36	92.24	91.60
7	PCA (8 components) with shallow ANN	0	88.00	92.20	91.06
8	Decision Tree Classification	1	87.35	92.58	89.48
9	Shallow ANN (10 Hidden Neurons)	4	86.46	88.93	92.75
10	K-NN (13 Neighbors)	0	85.39	90.97	88.51
11	Naïve Bayes	0	78.06	92.60	75.77

Table 5.25. Accuracy, precision and sensitivity of the ML models using experimental data over the testing data set

No.	Machine learning model	Optimum input time-lag (t-n)	Testing Set		
		n (Hour)	Accuracy (%)	Precision (%)	Sensitivity (%)
1	SVM	0	89.48	90.29	90.61
2	SVR	0	88.82	89.87	89.87
3	Random forest classification (400 trees)	0	87.78	87.43	90.70
4	Kernel SVM	0	87.51	88.32	88.94
5	PCA with random forest	0	86.51	87.54	87.66
6	PCA (8 components) with shallow ANN	0	86.17	87.25	87.56
7	Decision Tree Classification	0	82.70	84.03	84.56
8	K-NN (20 Neighbors)	0	80.85	84.19	79.97
9	Naïve Bayes	0	67.48	70.38	69.98
10	Shallow ANN (10 Hidden Neurons)	10	55.44	62.95	46.03
11	Deep ANN 2 Hidden layers L, 10 Neurons	0	54.87	61.92	45.38

In this thesis, a novel approach using the classification machine learning technique for the multi dependent fault detection and diagnosis (MDFDD) of air temperature sensors of an AHU in an institutional building was studied. Support vector machine (SVM) was applied, and compared with two other ML models, KNN and Naïve Bayes. The models were optimized over grid-search analysis for the tuning of hyperparameters.

The results on the experimental data revealed the ML models with the highest MDFDD accuracy was SVM with accuracy of higher than 70% for the MDFDD and 97.30% for the single fault detection and diagnosis. The Naïve Bayes and KNN have accuracy of higher than 83% and 57% for the MDFDD, and 73% and 79% for the single fault detection and diagnosis, respectively.

Moreover, the results on the synthetic data revealed the ML models with the highest MDFDD accuracy were SVM and Deep ANN with two-hour input time lag, with accuracy of 93.97% and 91.96%, respectively. The PCA-Random Forest and kernel SVM models with one-hour time lag are placed on next positions for the MDFDD with accuracy of 90.14% and 88.91%, respectively.

The capability of proposed ML models which were developed using the experimental and synthetic data set (from the eQuest program) was validated in this thesis. In future work these models will be applied on the other HVAC components.

6. Conclusions, contributions, and thesis limitations

6.1. Conclusions

In this thesis the application of hybrid model with the combination of machine learning/ deep learning models with the rule-based techniques for the detection and diagnosis of multiple depended faults in an AHU of an institutional building was proposed and developed using different case studies with use of experimental and synthetic data.

Moreover, the classification machine learning techniques for the MDFDD were proposed and applied on different cases studies and the results revealed promising approach for the contribution of this method for the MDFDD in the sensors of an AHU.

Out of the proposed models and case studies, the following results are concluded:

- The combination of machine learning (ML) models using BAS trend data, and rule-based models is successful for the multiple dependent faults detection and diagnosis in an AHU.
- This thesis presented a novel method that guides for faults search, by using the information and operation flow between sensors, and between sensors and devices. This approach was not found in any other publication.
- ML models were used for the prediction of two target variables, the mixed air temperature (T_ma) and air temperature after heating coil (T_ahc). The RNN models were used for the prediction of regressor sensors values (T_oa, T_ra, HW_T_S, Valve_HC). Rules-based models were used for the diagnosis of faults. These results revealed well performance of these models for the fault detection and diagnosis purposes.

- The proposed method was tested with measurements from BAS under normal operation, and with artificial faults inserted in the measurements data file. The results revealed the good performance of the proposed method for the multiple dependent faults of an AHU.
- For the fault detection, the compound ML architecture was proposed for the prediction of mixed air temperature (T_ma) and air temperature after heating coils (T_ahc) at current time (t).
- The results revealed a good compound prediction model that can be used as a tool for fault detection of air temperature sensors of AHU in heating mode. The performance of the different ML models, e.g. SVR, ANN, decision tree, and random forest were compared for the MDFDD scope.
- For the fault diagnosis, the recurrent neural network (RNN) using the LSTM model was proposed for the prediction of every single regressor at the current time (t), using previous time steps observations. In addition to the RNN, the feedforward neural network (FFNN) model was used for the prediction of regressors at the time (t).
- The results proved that both proposed fault diagnosis models (RNN and FFNN) have good prediction performance for the scope of this study.
- For the fault diagnosis step, the results of the ML models were combined with the rule-based technique and the results revealed well performance of the proposed model for the MDFDD.
- As an alternative approach for the fault detection step, a novel approach using the classification machine learning technique for the multiple dependent fault detection and diagnosis (MDFDD) of air temperature sensors of an AHU in an institutional building was studied.

- Support vector machine (SVM) was applied, and compared with other ML models, e.g. KNN and Naïve Bayes for the MDFDD. The models were optimized over grid-search analysis for the tuning of hyperparameters.
- The results revealed the ML models with the highest MDFDD accuracy was SVM for the MDFDD. The Naïve Bayes and KNN are placed on the second and third orders respect to accuracy for the MDFDD.

6.2. Contributions

In this thesis the following contributions and novelties are listed:

- A novel sequential (compound) machine learning model for the prediction of the target variable (T_ma and T_ahc) in the AHU for the scope of MDFDD was proposed.

- A novel technique for the threshold definition for the scope of FDD was proposed. This technique combines the sensor uncertainty and the ML model inaccuracy (RMSE).

- The hybrid technique, which is the combination of a machine learning model and rule-based technique was proposed for the MDFDD of the air temperature sensors in an AHU.

- Different machine learning, deep learning, and hybrid models for the MDFDD scope were developed and compared with the application of the K-fold cross validation for validation the average performance accuracy of the models.

- The novel classification machine learning models for the MDFDD were proposed, and different ML algorithms were compared to evaluate the performance of the model with the highest accuracy.

- The optimization of the models was studied respect to the models hyperparameters using RandomizedSearchCV tool, and the input time-lags using the grid search analysis.

6.3. Thesis limitations

The limitations of this thesis are provided in this section. Since in this thesis, the proposed method includes the data driven based approach, therefore, one of the crucial concerns regarding this method is the selection of the appropriate set of data. Two data sets have been deployed in this thesis, experimental and synthetic. The experimental data may not be accurate all the time, and some reading errors could exist. Therefore, some data preprocessing for the outliers' removal was implemented, however this process has also some minimal errors. Hence, the preprocessed data assumed to be fault free and normal for the implementation in the proposed MDFDD method.

Another concern is the synthetic data, which may not be much accurate to represent the real-world performance of the system operation, however, it was assumed that the synthetic data is the normal operation of the system.

Generation of artificial faults is one of the crucial points which was reported in this thesis. It was trying to introduce the artificial faults as the real-world faults, however, it can also be improved by introduction of the fault on real systems which are applicable to simulate the real world performance on the real data.

7. Future works

1- Since the proposed model for the MDFDD of the air temperature sensors of an AHU was validated in this thesis on the experimental and synthetic data, therefore, this method can be applied on the other sensors of the AHU (e.g. volumetric air flow rate, water flow rate) in future works.

2- Moreover, in the future works, the proposed model can be applied on the other HVAC systems with changing the type of variables to fit the model respect to the available sensors for that equipment for the scope of the MDFDD.

3- The recommended developed models will be added as a feature into the building automation system (BAS) of the HVAC system to control the system operation and alarm for any potential abnormal operation in the sensors/ equipment and diagnose the location of the fault, then report the severity of the faults.

4- In a promising future work of this research, the proposed models and be used in a portable online software for smartphones in order to control remotely the operation of the HVAC system in the commercial buildings, and then the user will be able to detect and diagnose the multiple faults.

8. References

[1] National Research Council of Canada, "National Energy Code of Canada for Buildings," National Research Council of Canada, Ottawa, 2017.

[2] P. Beiter, M. Elchinger, T. Tian, 2016 Renewable Energy Data Book, United States: National Renewable Energy Lab. (NREL), Golden, CO (United States), 2017.

[3] Kim, W., S. Katipamula, "A review of fault detection and diagnostics methods for building systems," *Science and Technology for the Built Environment,* vol. 24, pp. 3-21, 2018.

[4] Gunay, B., W. Shen, B. Huchuk, C. Yang, S. Bucking, W. O'Brien, "Energy and comfort performance benefits of early detection of building sensor and actuator faults," *Building Services Engineering Research and Technology,* pp. 1-15, 2018.

[5] Samuel, A. L., "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development,* pp. 210-229, 1959.

[6] Y. Zhao, T. Li, X. Zhang, C. Zhang, "Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future," *Renewable and Sustainable Energy Reviews,* vol. 109, pp. 85-101, 2019.

[7] Katipamula, S., M. R. Brambley, "Review Article: Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems—A Review, Part I," *HVAC&R Research,* vol. 11, no. 1, pp. 3-25, 2005.

[8] Yua, Y., D. Woradechjumroena, D. Yub, "A review of fault detection and diagnosis methodologies on air-handling units," *Energy and Buildings,* vol. 82, p. 550–562, 2014.

[9] Comstock, M., J. Braun, E. Groll, "The sensitivity of chiller performance to common faults," *HVAC & R Research,* vol. 7, no. 3, pp. 263-279, 2001.

[10] Kim, W., S. Katipamula, "A review of fault detection and diagnostics methods for building systems," *Science and Technology for the Built Environment ,* vol. 24, pp. 3-21, 2018.

[11] S. Katipamula, M.R. Brambley, "Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems—A Review, Part I," *HVAC & R Research,* vol. 11, no. 1, pp. 3-26, 2005.

[12] J. Schein, S. T. Bushby, N. S. Castro, J. M. House, "A rule-based fault detection method for air handling units," p. 1485–1492, 2006.

[13] Zhao, Y., T. Li, X. Zhang, C. Zhang, "Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future," *Renewable and Sustainable Energy Reviews,* vol. 109, pp. 85-101, 2019.

[14] Mirnaghi, M. S., F. Haghighat, "Fault detection and diagnosis of large-scale HVAC systems in buildings using data-driven methods: A comprehensive review," *Energy and Buildings,* vol. 229, p. 110492, 2020.

[15] Katipamula, S., M. R. Brambley, "Review Article: Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems—A Review, Part II," vol. 11, no. 2, pp. 169-187, 2005.

[16] Cengel, Y., M.A. Boles, Thermodynamics, An Engineering Approach, 6th edition, New York: McGraw Hill, 2001.

[17] N. Zibin, A Bottom-Up Method to Calibrate Building Energy Models Using Building Automation System (BAS) Trend Data, Montreal: MASc thesis - Concordia Unversity , 2014.

[18] Zhao, Z., S. Wang, F. Xiao, Z. Ma, "A simplified physical model-based fault detection and diagnosis strategy and its customized tool for centrifugal chillers," *HVAC & R Research,* vol. 19, pp. 283-294, 2013.

[19] J. Liang, R. Du, "Model-based Fault Detection and Diagnosis of HVAC systemsusing Support Vector Machine method," *International Journal of Refrigeration,* vol. 30, pp. 1104-1114, 2007.

[20] Pourariana, S., J. Wen, D. Veronica, A. Pertzborn, X. Zhouc, R. Liu, "A tool for evaluating fault detection and diagnostic methods for fan coil units," *Energy and Buildings,* vol. 136, pp. 151-160, 2017.

[21] Mulumba, T., A. Afshari, K. Yana, W. Shena, L. K. Norford, "Robust model-based fault diagnosis for air handling units," *Energy and Buildings,* vol. 86, pp. 698-707, 2015.

[22] Bonvini, M., M. D. Sohn, J. Granderson, M. Wetter, M. Ann Piette, "Robust on-line fault detection diagnosis for HVAC components based on nonlinear state estimation techniques," *Applied Energy,* vol. 124, pp. 156-166, 2014.

[23] Najafi, M., "Modeling and Measurement Constraints in Fault Diagnostics for HVAC Systems," Lawrence Berkeley National Laboratory, University of California, Berkeley, 2010.

[24] Najafi, M., D. M. Auslander, P. L. Bartlett, P. Haves, M. D. Sohn, "Application of machine learning in the fault diagnostics of air handling units," *Applied Energy,* vol. 96, pp. 347-358, 2012.

[25] Wang, H., Y. Chen, C. W. H. Chan, J. Qin, J. Wang, "Online model-based fault detection and diagnosis strategy for VAV air handling," *Energy and Buildings,* vol. 55, p. 252–263, 2012.

[26] Deshmukh, S., L. Glicksman, L. Norford, "Case study results: fault detection in air-handling units in buildings," *Advances in Building Energy Research,* 2018.

[27] Schein, J., S. T. Bushby, N. S. Castro, J. M. House, "A rule-based fault detection method for air handling units," *Energy and Buildings,* vol. 38, pp. 1485-1492, 2006.

[28] Schein, J., Results from Field Testing of Embedded Air Handling Unit and Variable Air Volume Box Fault Detection Tools, California : U.S. Department of Commerce, National Institute of Standards and Technology, 2006.

[29] Katipamula, S., M. R. Brambley, L. Luskay, "Automated Proactive Techniques for Commissioning Air handling unit," *Solar Energy Engineering - Transactions to ASME,* vol. 125, pp. 282-291, 2003.

[30] House, J. M., H. Vaezi-Nejad, J. M. Whitcomb, "An expert rule set for fault detection in air handling units," *ASHRAE Transactions,* vol. 107, pp. 858-871, 2001.

[31] House, J. M., W. Y. Lee, D. R. S., "Classification techniques for fault detection and diagnosis of an air handling unit," *ASHRAE Transactions: Symposia,* pp. 1087-1097, 1999.

[32] Katipamula, S., M. R. Brambley, N. N. Bauman, R. G. Pratt, "Enhancing Building Operations through Automated Diagnostics: Field Test Results," in *Proceedings of the Third International Conference for Enhanced Building Operations, October 13-15*, Berkeley, California, 2003.

[33] Yang, H., S. Cho, C.S. Tae, M. Zaheeruddin, "Sequential rule based algorithms for temperature sensor fault detection in air handling units," *Energy COnversion and Management ,* vol. 49, pp. 2291-2306, 2008.

[34] Wang, H., Y. Chen, "A robust fault detection and diagnosis strategy for multiple faults ofVAV air handling units," *Energy and Buildings,* vol. 127, pp. 442-451, 2016.

[35] Katipamula, S., R. G. Pratt, D. P. Chassin, Z. T. Taylor, "Automated Fault Detection and Diagnostics for Outdoor-Air Ventilation Systems and Economizers: Methodology and Results from Field Testing," *ASHRAE Transactions ,* vol. 105, pp. 1-13, 1999.

[36] Zhao, Y., J. Wen, F. Xiao, X. Yang, S. Wang, "Diagnostic Bayesian networks for diagnosing air handling units faults – part I: Faults in dampers, fans, filters and sensors," *Applied Thermal Engineering,* vol. 111, p. 1272–1286, 2017.

[37] Zhao, Y., J. Wen, S. Wang, "Diagnostic Bayesian networks for diagnosing air handling units faults - Part II: Faults in coils and sensors," *Applied Thermal Engineering,* vol. 90, pp. 145-157, 2015.

[38] Dey, D., B. Dong, "A probabilistic approach to diagnose faults of air handling units inbuildings," *Energy and Buildings,* vol. 130, pp. 177-187, 2016.

[39] Qin, J., S. Wang, "A fault detection and diagnosis strategy of VAV air-conditioning systems for improved energy and control performances," *Energy and Buildings,* vol. 37, pp. 1035-1048, 2005.

[40] Gunay, H. B, Z. Shi, G. Newsham, R. Moromisato, "Detection of zone sensor and actuator faults through inverse greybox modelling," *Building and Environment,* vol. 171, p. 106659, 2020.

[41] Shi, Z., W. O'Brien, H. B. Gunay, "Building zone fault detection with kalman filter based method," in *eSim*, Hamilton, 2016.

[42] Bishop, C., Pattern Recognition and Machine Learning. New York, NY: Springer, New York, NY: Springer, 2006.

[43] Annex 34, "Technical Synthetic Report Computer Aided Evaluation of HVAC System Performance," International Energy Agency, Birmingham, 2006.

[44] Jolliffe, I.T. , Principal Component Analysis, New York: Springer, 1986.

[45] Cotrufo, N., R. Zmeureanu, "PCA-based method of soft fault detection and identification for theongoing commissioning of chillers," *Energy and Buildings,* vol. 130, pp. 443-452, 2016.

[46] Morrison, P., F. Donald, Multivariate statistical method, 2nd ed., New York, USA: McGraw-Hill, 1976.

[47] Ladd, J. W., D. M. Driscoll, " A comparison of objective and subjective means of weather typing: an example from west Texas," *American Meteorological Society,* vol. 19, pp. 691-704, 1980.

[48] Chandrashekar, G., F. Sahin, "A survey on feature selection methods," *Computers and Electrical Engineering,* vol. 40, pp. 16-28, 2014.

[49] Kohavi, R., G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence,* vol. 97, pp. 273-324, 1997.

[50] Guyon, I., A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research ,* vol. 3, pp. 1157-1182, 2003.

[51] Battiti, R., "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions on Neural Networks,* vol. 5, pp. 537 - 550, 1994 .

[52] J. Lee Rodgers, W. A. Nicewander, "Thirteen Ways to Look at the Correlation Coefficient," *The American Statistician,* vol. 42, no. 1, pp. 59-66, 1988.

[53] MATLAB, 2018. [Online]. Available: https://www.mathworks.com/?s_tid=gn_logo. [Accessed 2020].

[54] Python , 3.8.1. [Online]. Available: https://www.python.org/.

[55] Guo, Y., G. Li, H. Chen, J. Wanga, M. Guo, S. Sun, W. Hub, "Optimized neural network-based fault diagnosis strategy for VRF system in heating mode using data mining," *Applied Thermal Engineering,* vol. 125, pp. 1402-1413, 2017.

[56] Lazar, C., J. Taminau, S. Meganck, D. Steenhoff, A. Coletta, C. Molter, V. de Schaetzen, R. Duque, H. Bersini, A. Nowe, "A Survey on Filter Techniques for Feature Selection in Gene Expression Microarray Analysis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* vol. 9, pp. 1106 - 1119, 2012.

[57] Kwak, N., C. H. Choi, "Input feature selection for classification problems. IEEE Trans Neural Networks 2002;13:143–59.," *IEEE Transactions on Neural Networks ,* vol. 13, pp. 143-59, 2002.

[58] Pudil, P, J. Novovicova, j. Kittler , "Floating search methods in feature selection," *Pattern Recognition Letters,* vol. 15, pp. 1119-1125, 1994.

[59] Reunanen, J., "Overfitting in Making Comparisons Between Variable Selection Methods," *Journal of Machine Learning Research ,* vol. 3, pp. 1371-1382, 2003.

[60] Goldberg, D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Boston: Addison-Wesley, 1988.

[61] Kennedy J., R. C. Eberhart , "Particle swarm optimization," *IEEE international conference on neural networks, IV,* p. 1942–1948, 1995.

[62] Blum, A. L., P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence ,* vol. 97, pp. 245-271, 1997.

[63] Lagley, P., "Selection of Relevant Features in Machine Learning," in *AAAI Fall symposium on relevance*, 1994.

[64] Sandri, M., P. Zuccolotto , "Variable Selection Using Random Forests. In: Zani S., Cerioli A., Riani M., Vichi M. (eds) Data Analysis, Classification and the Forward Search. Studies in Classification, Data Analysis, and Knowledge Organization," *Springer,* pp. 263-270, 2006.

[65] Ma, S. J. Huang, "Penalized feature selection and classification in bioinformatics," *Briefings in Bioinformatics,* vol. 9, pp. 392-403, 2008.

[66] Zou, H., T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* vol. 67, p. 301–320, 2005.

[67] Wang, H., Y. Chen, "A robust fault detection and diagnosis strategy for multiple faults ofVAV air handling units," *Energy and Buildings,* p. 442–451, 2016.

[68] Yu, Y., D. Woradechjumroena, D. Yu, "A review of fault detection and diagnosis methodologies on air-handling units," *Energy and Buildings,* p. 550–562, 2014.

[69] Beghi, A., R. Brignoli, L. Cecchinato, G. Menegazzo, M. Rampazzo, F. Simmini, "Data-driven fault detection and diagnosis for HVAC water chillers," *Control Engineering Practice,* vol. 53, pp. 79-91, 2016.

[70] Shia, Z., W. O'Brien, "Development and implementation of automated fault detection and diagnostics for building systems : A review," *Automation in Construction,* vol. 104, pp. 215-229, 2019.

[71] Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 12 , pp. 2825-2830, 2011.

[72] Chollet, F., 2015. [Online]. Available: https://github.com/fchollet/keras.

[73] Abadi, M., P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, "Large-Scale Machine Learning on Heterogeneous Systems," 2015. [Online]. Available: http://tensorflow.org/.

[74] ASHRAE GUIDELINE 2, "Engineering Analysis of Experimental Data," American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, 2005.

[75] Cotrufo, N., R. Zmeureanu, A. Athienitis, "Virtual measurement of the air properties in airhandling units (AHUs) or virtual re-calibration of sensors," *Science and Technology for the Built Environment,* vol. 25, no. 1, pp. 21-33, 2019.

[76] Barnston, A. G., "Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score," *Notes and Correspondence, Climate Analysis Center,* pp. 699-709, 1992.

[77] Kenney, J. F., Mathematics of Statistics, Part 1; 3rd edition, Van Nostrand, 1962.

[78] J. S. R. G. H. R. S. Goldberger, "Neighbourhood Components Analysis," *Advances in Neural Information Processing Systems,* vol. 17, pp. 513-520, 2005.

[79] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, pp. 5-32, 2001.

[80] Olson, L.D., D. Delen, Advanced Data Mining Techniques, Heidelberg: Springer, 2008.

[81] C. J. Van Rijsbergen, Information Retrieval, Butterworth- Heinemann, 1979.

[82] Y. Sasaki, "The truth of the F-measure," School of Computer Science, University of Manchester, Manchester, 2007.

[83] Braspenning, P. J., F. Thuijsman, A. J. M. M. Weijters, Artificial Neural Networks, Heidelberg: Springer, 1995.

[84] LeCun, Y., L. Bottou, G. B. Orr, K. R. Muller, "Efficient BackProb," *Springer,* 1998.

[85] W.Y. Lee, J.M. House, C. Park, G.E. Kelly, "Fault diagnosis of an air-handling unit using artificial neural networks," *ASHRAE Transact,* vol. 102, pp. 540-549, 1996.

[86] Chae, Y. T., R. Horesh, Y. Hwang, Y. M. Lee, "Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings," *Energy and Buildings,* vol. 111, pp. 184-194, 2016.

[87] Z. Hou, Z. Lian, Y. Yao, X. Yuan, "Data mining based sensor fault diagnosis and validation for building air conditioning system," *Energy Conversion and Management,* vol. 47, no. 15-16, pp. 2479-2490, 2006.

[88] Lee, W. Y., J. M. House, C. Parc, G. E. Kelly, "Fault diagnosis of an air-handling unit using artificial neural networks," in *Winter meeting of American Society of Heating, Refrigeration and Air Conditioning Engineers (ASHRAE transactions)*, Atlanta, GA, 1996.

[89] Y. T. Chae, R. Horesh, Y. Hwang, Y. M. Lee, "Artificial neural network model for forecasting sub-hourly electricity usage in commercial buildings," *Energy and Buildings,* vol. 111, pp. 184-194, 2016.

[90] B. Fan, Z. Du, X. Jin, X. Yang, Y. Guo, "A hybrid FDD strategy for local system of AHU based on artificial neural network and wavelet analysis," *Building and Environment,* vol. 45, pp. 2698-2708, 2010.

[91] Elnour, M., N. Meskin, M. Al-Naemi, "Sensor data validation and fault diagnosis using Auto-Associative Neural Network for HVAC systems," *Journal of Building Engineering,* vol. 27, 2020.

[92] S. Heo, J. H. Lee, "Fault detection and classification using artificial neural networks," *IFAC PapersOnLine,* vol. 51, no. 18, pp. 470-475, 2018.

[93] P. Geyer, S. Singaravel, "Component-based machine learning for performance prediction in building design," *Applied Energy,* vol. 228, pp. 1439-1453, 2018.

[94] Cristianini, N., J. Shawe-Taylor, An Introduction to Support Vector Machines, Cambridge : Cambridge University Press,, 2000.

[95] Burges, C. J.C. , "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery,* vol. 2, pp. 121-167, 1998.

[96] Ben-Hur, A., J. Weston, "A users guide to support vector machines, in: Data min- ing techniques for the life sciences," *Springer,* pp. 223-239, 2010.

[97] J. Liang, R. Du, "Model-based Fault Detection and Diagnosis of HVAC systems using Support Vector Machine method," *International Journal of Refrigeration,* vol. 30, pp. 1104-1114, 2007.

[98] Yan, K., C. Zhong, Z. Ji, J. Huang, "Semi-supervised learning for early detection and diagnosis of various air handling unit faults," *Energy and Buildings ,* vol. 181, pp. 75-83, 2018.

[99] H. Han, B. Gu, T. Wang, Z.R. Li, "Important sensors for chiller fault detection and diagnosis (FDD) from the perspective of feature selection and machine learning," *International journal of refrigeration ,* pp. 586-599, 2011.

[100] Han, H., B. Gua, J. Kang a, Z.R. Li, "Study on a hybrid SVM model for chiller FDD applications," *Applied Thermal Engineering ,* vol. 31, pp. 582-592, 2011.

[101] Han, H., B. Gu, Y. Hong, J. Kang, "Automated FDD of multiple-simultaneous faults (MSF) and the application to building chillers," *Energy and Buildings,* vol. 43, p. 2524–2532, 2011.

[102] Drucker, H., CJC. Burges, L. Kaufman, A. Smola, V. Vapnik, "Support vector regression machines," *MIT press,* pp. 155-161, 1997.

[103] Lewis, D. D., "Naive (Bayes) at forty: The independence assumption in information retrieval," *Lecture Notes in Computer Science ,* vol. 1398, 1998.

[104] Zhang, H., "The Optimality of Naive Bayes," *American Association for Artificial Intelligence,* 2004.

[105] Cai, B., Y. Liu, Q. Fan, Y. Zhang, Z. Liu, S. Yu, R. Ji, "Multi-source information fusion based fault diagnosis of ground-source heat pump using Bayesian network," *Applied Energy,* vol. 114, pp. 1-9, 2014.

[106] Loh, W. Y. , "Classification and regression trees," *John Wiley & Sons, Inc,* vol. 1, pp. 14-23, 2011.

[107] Rokach, L., O. Maimon, Data mining with decision trees: Theory and Applications, Singapore: World Scientific, 2008.

[108] Yan, R., Z. Ma, Y. Zhao, G. Kokogiannakis, "A decision tree based data-driven diagnostic strategy for air handling units," *Energy and Buildings,* vol. 133, pp. 37-45, 2016.

[109] Jackson, J.E., G.S. Mdholkar, "Control Procedures for Residuals Associated With Principal Component Analysis," *Thechnometrics,* pp. 341-349, 1979.

[110] R. Yan, Z. Ma, G. Kokogiannakis, Y. Zhao, "A sensor fault detection strategy for air handling units using cluster analysis," *Automation in Construction,* pp. 77-88, 2016.

[111] Li, S., J. Wen, "A model-based fault detection and diagnostic methodology based onPCA method and wavelet transform," *Energy and Buildings,* pp. 63-71, 2014.

[112] Z. Du, B. Fan, X. Jin, J. Chi, "Fault detection and diagnosis for buildings and HVAC systems using combined neural networks and subtractive clustering analysis," *Building and Environment,* vol. 73, pp. 1-11, 2014.

[113] Cotrufo, N., R. Zmeureanu, "Virtual outdoor air flow meter for an existing HVAC system in heating mode," *Automation in Construction,* vol. 92, p. 166–172, 2018.

[114] Bezyan, B., R. Zmeureanu, "Principal Component Analysis for the ongoing commissioning of northern houses," in *Proceedings of eSim 2018, the 10th conference of IBPSA-Canada*, Montreal, May 9-10, 2018.

[115] Lia, G., Y. Hu, H. Chen, L. Shena, H. Li, M. Hu, J. Liua, K. Sun, "An improved fault detection method for incipient centrifugal chillerfaults using the PCA-R-SVDD algorithm," *Energy and Buildings,* pp. 104-113, 2016.

[116] Tax, D.M.J., R.P.W. Duin, "Support Vector Data Description," *Machine Learning,* pp. 45-66, 2004.

[117] Tax, D.M.J, R.P.W. Duin, "Support vector domain description," *Pattern Recognition Letters,* pp. 1191-1199, 1999.

[118] Lee, K.P., B.H. Wu, S.L. Peng, "Deep-learning-based fault detection and diagnosis of air-handling units," *Building and Environment,* pp. 24-33, 2019.

[119] Heo, S., J. H. Lee, "Fault detection and classification using artificial neural networks," *IFAC PapersOnLine,* vol. 51, no. 18, pp. 470-475, 2018.

[120] Guo, Y., Z. Tanb, H. Chen, G. Lic, J. Wanga, R. Huanga, J. Liua, T. Ahmada, "Deep learning-based fault diagnosis of variable refrigerant flow air-conditioning system for building energy saving," *Applied Energy,* p. 732–745, 2018.

[121] Hecht-Nielsen, R., "Kolmogorov's mapping neural network existence theorem," *IEEE,* pp. 11-13, 1987.

[122] Heaton, J. , Introduction to neural networks with Java. Heaton Research, 2008.

[123] Blum, A., Neural Networks in C++, New York: Wiley, 1992.

[124] Berry, MJ., GS. Linoff, Data mining techniques. JSTOR; 2006, Wiley, 2006.

[125] Shahnazari, H., P. Mhaskar, J. M. House, T. I. Salsbury, "Modeling and fault diagnosis design for HVAC systems using recurrent neural networks," *Computers and Chemical Engineering,* pp. 189-203, 2019.

[126] Hochreiter, S., J. Schmidhuber, "Long Short-Term Memory," *Neural Computation ,* pp. 1735-1780, 1997.

[127] Shen, Y., K. Khorasani, "Hybrid multi-mode machine learning-based fault diagnosis strategies with application to aircraft gas turbine engines," *Neural Networks,* vol. 130, pp. 126-142, 2020.

[128] Nebauer, C., "Evaluation of convolutional neural networks for visual recognition," *IEEE,* vol. 9, pp. 685 - 696, 1998.

[129] LeCun, Y., Y. Bengio, G. Hinton, "Deep learning," *Nature,* vol. 521, pp. 436-444, 2015.

[130] Wu, J., "Introduction to Convolutional Neural Networks," Nanjing University, Nanjing, 2017.

[131] Miyata, S., Y. Akashi, J. Lim, Y. Kuwahara, K. Tanaka, "Model-based fault detection and diagnosis for HVAC systems using convolutional neural network," in *International Building Performance Simulation Association*, Rome, 2019.

[132] Elnour, M., N. Meskin, "Actuator Fault Diagnosis in Multi-Zone HVAC Systems using 2D Convolutional Neural Networks," in *IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, Doha, 2020.

[133] Goodfellow, I.J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative Adversarial Nets," *International Conference on Neural Information Processing Systems,* pp. 2672-2680, 2014.

[134] Yang K., J. Huang, W. Shen, Z. Ji, "Unsupervised learning for fault detection and diagnosis of air handling units," *Energy and Buildings,* vol. 210, 2020.

[135] Yan, K., A. Chong, Y. Mo, "Generative adversarial network for fault detection diagnosis of chillers," *Building and Environment,* vol. 172, 2020.

[136] Zhao, X, "Lab test of three fault detection and diagnostic methods' capability of diagnosing multiple simultaneous faults in chillers," *Energy and Buildings,* vol. 94, pp. 43-51, 2015.

[137] Montazeri, A., S.M. Kargar, "Fault detection and diagnosis in air handling using data-driven methods," *Journal of Building Engineering ,* vol. 31, 2020.

[138] Ebrahimifakhar, A., A. Kabirikopaei, D. Yuill, "Data-driven fault detection and diagnosis for packaged rooftop units using statistical machine learning classification methods," *Energy and Buildings,* vol. 225, p. 110318, 2020.

[139] Wang, H., Y. Chena, C. W.H. Chan, J. Qin, J. Wang, "Online model-based fault detection and diagnosis strategy for VAV air handling units," *Energy and Buildings,* vol. 55, pp. 252-263, 2012.

[140] R.L. Haupt, S.E. Haupt, Practical Genetic Algorithms, New York: John Wiley & Sons, 2004.

[141] Holland, J.H., "Adaptation in Natural and Artificial Systems," *The University of Michigan Press,* 1975.

[142] Golberg, D.E., "Genetic Algorithms in search, optimization, and machine learning," *Addison-Wesley,* 1989.

[143] Nassif, N., S. Moujaes, M. Zaheeruddin, "Self-tuning dynamic models of HVAC system components," *Energy and Buildings,* vol. 40, p. 1709–1720, 2008.

[144] Padilla, M., D. Choiniere, J. Candanedo, "A model-based strategy for self-correction of sensor faults in variable air volume air handling units," *Science and Technology for the Built Environment,* vol. 21, p. 1018–1032, 2015.

[145] Lu, L., W. Cai, L. Xie, S. Li, Y. C. Soh, "HVAC system optimization—in-building section," *Energy and Buildings q,* vol. 37, pp. 11-22, 2005.

[146] Nassif, N., S. Kajl, R. Sabourin, "Optimization of HVAC Control System Strategy Using Two-Objective Genetic Algorithm," *HVAC&R Research,* vol. 11, no. 3, pp. 459-486, 2005.

[147] Palonen, M., A. Hasan, K. Siren, "A GENETIC ALGORITHM FOR OPTIMIZATION OF BUILDING ENVELOPE AND HVAC SYSTEM PARAMETERS," in *Building Simulation*, Glasgow, Scotland, July 27-30, 2009.

[148] Wang, S., X. Jin, "Model-based optimal control of VAV air-conditioning system using genetic algorithm," *Building and Environment ,* vol. 35, pp. 471-487, 2000.

[149] Zhao, Y., T. Li, X. Zhang, C. Zhang, "Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future," *Renewable and Sustainable Energy Reviews,* vol. 109, pp. 85-101, 2019.

[150] Ben-Hur, A., J. Weston, "A users guide to support vector machines, in: Data mining techniques for the life sciences," *Springer,* pp. 223-239, 2010.

[151] Vapnik, V., "The nature of statistical learning theory.," *Springer Science & Business Media,* 1995.

[152] Cherkassky, V., Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural Networks,* vol. 17, no. 1, pp. 113-126, 2004.

[153] Yan K., C. Zhong, Z. Ji, J. Huang, "Semi-supervised learning for early detection and diagnosis of variousair handling unit faults," *Energy & Buildings,* vol. 181, pp. 75-83, 2018.

[154] Kahlen, J. N., M. Andres, A. Moser, "Improving Machine-Learning Diagnostics with Model-Based Data Augmentation Showcased for a Transformer Fault," *Energies,* vol. 14, 2021.

[155] Li, H., J. E. Braun, "A Methodology for Diagnosing Multiple simultaneous faults in vapor-compression air conditioners," *HVAC&R Research,* vol. 13, pp. 369-395, 2011.

[156] Elnour, M., N. Meskin, M. Al-Naemi, "Sensor data validation and fault diagnosis using Auto-Associative Neural Network for HVAC systems," *Journal of Building Engineering,* vol. 27, 2020.

[157] Mihai, A., R. Zmeureanu, "Bottom-up evidence-based calibration of the HVAC air-side loop of a building energy model," *Journal of Building Performance Simulation,* vol. 10, no. 1, pp. 105-123, 2017.

[158] Zibin, N., R. Zmeureanu, J. Love, "Bottom-up simulation calibration of zone and system level models using building automation system (BAS) trend data," in *eSim*, Ottawa, Canada, 2014.

[159] Photo by Nicolas McComber – JODOIN LAMARRE PRATTE ARCHITECTES, [Online]. Available: https://www.jlp.ca/en/project/centre-for-structural-and-functional-genomics-at-concordia-university-leed-gold/.

[160] Jolliffe, I.T. , Principal Component Analysis, New York: Springer, 1986.

[161] H. He , T. P. Caudell, D. Menicucci, A. A. Mammoli, "Application of adaptive resonance theory neural networks to monitor solar hot water systems and detect existing or developing faults," *Solar Energy,* vol. 86, pp. 2318-2333, 2012.

[162] H. He, D. Menicucci, T. Caudell, A. Mammoli, "Real-time fault detection for solar hot water system using adaptive resonance theory neural network," *ME 2011 5th International Conference on Energy Sustainability & 9th Fuel Cell Science, Wahington, DC, USA,* pp. 1059-1065, 2011.

[163] Tran, D. A. T., Y. Chen, C. Jiang, "Comparative investigations on reference models for fault detectionand diagnosis in centrifugal chiller systems," *Energy and Buildings,* Vols. 246-256, p. 133, 2016.

[164] Ding, X., Y. Guo, T. Liu, Q. Liu, H. Chen, "New fault diagnostic strategies for refrigerant charge fault in a VRF system using hybrid machine learning method," *Journal of Building Engineering,* vol. 33, p. 101577, 2021.

[165] Wang, S., Y. Chen, "Fault-tolerant control for outdoor ventilation air fow rate in buildings based on neural network," *Building and Environment,* vol. 37, pp. 691-704, 2002.

[166] Du, Z., X. Jin, Y. Yang, "Fault diagnosis for temperature, flow rate and pressure sensors in VAV systems using wavelet neural network," *Applied Energy,* vol. 86, p. 1624–1631, 2009.

[167] Du Z., X. Jin, Y. Yang, "Wavelet Neural Network-Based Fault Diagnosis in Air-Handling Units," *HVAC & R Research,* vol. 14, pp. 959-973, 2011.

[168] Lee, W. Y., J. M. House, N. H. Kyong, "Subsystem level fault diagnosis of a building's air-handling unit using general regression neural networks," *Applied Energy,* vol. 77, pp. 153-170, 2004.

[169] Allen W. H., A. Rubaai, R. Chawla, "Fuzzy Neural Network-Based Health Monitoring for HVAC System Variable-Air-Volume Unit," *IEEE,* vol. 52, pp. 2513 - 2524, 2015.

[170] Stephen, F., M. Heaney, X. Jin, J. Robertson, H. Cheung, R. Elmore, G. Henze, "Hybrid Model-based and Data-driven Fault Detection and Diagnostics for Commercial Buildings," in *National Renewable Energy Laboratory (NREL) August 21–26, 2016*, Pacific Grove, California .

[171] Holcomb, D., W. Li, S. A. Seshia, "Algorithms for Green Buildings: Learning-Based Techniques for Energy Prediction and Fault Diagnosis," Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA, 2009.

[172] Yang, X. B., X. Q. Jin, Z. M. Du, Y. H. Zhu, Y. B. Guo, "A hybrid model-based fault detection strategy for air handling unit sensors," *Energy and Buildings,* vol. 57, pp. 132-143, 2013.

[173] Wang, Z., Z. Wang, S. He, X. Gu, Z. F. Yan, "Fault detection and diagnosis of chillers using Bayesian network merged distance rejection and multi-source non-sensor information," *Applied Energy,* vol. 188, pp. 200-214, 2017.

[174] Zhao, Y., F. Xiao, S. Wang, "An intelligent chiller fault detection and diagnosis methodology using Bayesian belief network," *Energy and Buildings,* vol. 57, pp. 278-288, 2013.

[175] Wang, Z., L. Wang, K. Liang, Y. Tan, "Enhanced chiller fault detection using Bayesian network and principal component analysis," *Applied Thermal Engineering,* vol. 141, pp. 898-905, 2018.

[176] Dey, M., S. P. Rana, S. Dudley, "A case study based approach for remote fault detection using multi-level machine learning in a smart building," *Smart Cities,* vol. 3, pp. 401-419, 2020.

[177] Yan, R., Z. Ma, G. Kokogiannakis, Y. Zhao, "A sensor fault detection strategy for air handling units using cluster analysis," *Automation in Construction,* vol. 70, pp. 77-88, 2016.

[178] Li, G., Y. Hu, "Improved sensor fault detection, diagnosis and estimation for screw chillers using density-based clustering and principal component analysis," *Energy & Buildings,* vol. 173, p. 502–515, 2018.

[179] Chen, Y., L. Lan, "A fault detection technique for air-source heat pump water chiller/heaters," *Energy and Buildings,* vol. 41, pp. 881-887, 2009.

[180] Du, Z., X. Jin, X. Yang, "A robot fault diagnostic tool for flow rate sensors in air dampers and VAV terminals," *Energy and Buildings,* vol. 41, pp. 279-286, 2009.

[181] Jianying, Q., "A Fault Detection and Diagnosis Strategy for VAV Air Distribution," The Hong Kong Polytechnic University, 2006.

[182] Wang, S., F. Xiao, "Wavelet Neural Network-Based Fault Diagnosis in Air-Handling Units," *HVAC & R RESEARCH ,* vol. 12, pp. 959-973, 2005.

[183] Li, G., Y. Hu, H. Chen, H. Li, M. Hu, Y. Guo, S. Shi, W. Hu, "A sensor fault detection and diagnosis strategy for screw chillersystem using support vector data description-based D-statistic and DV-contribution plots," *Energy and Buildings,* vol. 133, pp. 230-245, 2016.

[184] Shi, S., G. Li, H. Chen, Y. Hua, X. Wanga, Y. Guo, S. Sun, "An efficient VRF system fault diagnosis strategy for refrigerant charge amount based on PCA and dual neural network model," *Applied Thermal Engineering,* vol. 129, p. 1252–1262, 2018.

[185] Yang, X. B., X. Q. Jin, Z. M. Du, Y. H. Zhu, "A novel model-based fault detection method for temperature sensor using fractal correlation dimension," *Building and Environment,* vol. 46, pp. 970-979, 2011.

[186] N. Koçyigit, "Fault and sensor error diagnostic strategies for a vapor compression refrigeration system by using fuzzy inference systems and artificial neural network," *International Journal of Refrigeration,* vol. 50, pp. 69-79, 2015.

[187] Magoules, F., H. Z. Zhao, D. Elizondo, "Development of an RDP neural network for building energyconsumption fault detection and diagnosis," *Energy and Buildings,* vol. 62, pp. 133-138, 2013.

[188] Yan K., J. Huang, W. Shen, Z. Ji, "Unsupervised learning for fault detection and diagnosis of air handling units," *Energy and Buildings,* vol. 210, 2020.

[189] Ebrahimifakhar, A., A. Kabirikopaei, D. Yuill, "Data-driven fault detection and diagnosis for packaged rooftop units using statistical machine learning classification methods," *Energy and Buildings,* vol. 225, p. 110318, 2020.

[190] Zhao, Y., F. Xiao, J. Wen, Y. Lu, S. Wang, "A robust pattern recognition-based fault detection and diagnosis (FDD) method for chillers," *HVAC & R Research,* vol. 20, pp. 798-809, 2014.

[191] Li, D., G. Hu, C. J. Spanos, "A data-driven strategy for detection and diagnosis of building chillerfaults using linear discriminant analysis," *Energy and Buildings,* vol. 128, pp. 519-529, 2016.

[192] Miyata, S., J. Lim, Y. Akashi, Y. Kuwahara, K. Tanaka, "Fault detection and diagnosis for heat source system using convolutional neural network with imaged faulty behavior data," *Science and Technology for the Built Environment ,* vol. 26, pp. 52-60, 2019.

9. Appendices

9.1. Appendix A

Summary of literature review for the single fault detection and diagnosis in HVAC systems are provided in this section.

Table 9.1. Data-driven models for single faults detection and diagnosis applications in HVAC systems

No.	FDD model	HVAC system	Data type	Available parameters	FDD targets	References
1	ANN (Auto-associative neural network)	AHU	Synthetic	Zone air temperature, chilled water tank temperature, supply/ outdoor air temperature, return water temperature, zone VAV control signal, water tank valve signal	Sensors of air temperature	[91]
2	ANN	Solar hot water system	Synthetic	Collector length/ width, absorber plate thickness, absorber conductivity, tubes No., inner/ outer tube diameter, fluid specific heat, cover details, glass emissivity and details.	Pump fault, impeller	[161-162]
3	Multiple linear regression, Kriging, radial basis function) + exponentially weighted moving average)	Chiller	Measured	Evaporator/ condenser water temperature difference, condenser approach temperature, lubricant oil temperature	Condenser fouling, excess oil, reduced condenser and evaporator water flow rate, non-condensable gas in the refrigerant, refrigerant leak/undercharge and overcharge, excess oil	[163]
4	ANN + Decision Tree	VRF	Measured	Outdoor air temperature, compressor discharge temperature, compressor shell/ module temperature, compressor frequency, compressor electric, sub-cooler outlet pipe temperature (gas, liquid), gas-liquid separator inlet/ outlet pipe temperature	Refrigerant overcharge and undercharge	[164]
5	ANN, KNN, Bayes classifier, rule-based	AHU	Synthetic	Supply/ mixed/ return air temperature, supply air pressure, volumetric air flow rate	Stuck cooling coil valve, cooling coil fouling, heating coil leakage, stuck VAV box damper,	[31]

					performance degradation of the supply fan, return fan controller failure, mixing box damper failure,	
6	ANN	AHU	Synthetic	Outdoor air flow rate, damper position control signal, supply/ return volumetric air flow rates	Stuck damper, air flow rate sensors (bias, drift or complete failure), CO2 sensors errors.	[165]
7	Adaptive fuzzy neural network	AHU	Measured	Supply air temperature set point, supply air pressure set point, water pump control signal	Sensors fault (air temperature, air flow rate, and pressure) Equipment fault (supply fan, cooling coil- water pump)	[166]
8	Wavelet ANN	AHU	Measured	Supply/ return air temperature, supply/ return air flow rate, supply air static pressure, supply/ return chilled water temperature, chilled water flow rate	Sensors faults (air/ water temperature, air flow rate)	[167]
9	ANN	AHU	Synthetic	Supply/ outdoor/ return/ mixed air temperature, supply/ return air flow rate, supply air static pressure, mixed air humidity ration, control signal of cooling coil valve/ air dampers	Sensors error (air temperature, air flow rate, and pressure)	[168]
10	Fuzzy logic (rule-based) + ANN	AHU (VAV)	Measured and synthetic	Damper position, air flow rate, air temperature	Sensor fault (air temperature, air flow rate), damper stuck	[169]
11	Ordinary least squares, quantile regression, KNN, Naïve bayes, decision tree, random forest, SVM	AHU	Synthetic	Air pressure, return/ economizer air temperature, damper control signal, air flow rate	Duct fouling, thermostat bias, Economizer outsider/ return air temperature sensor bias, Economizer damper stuck fault, Blower fan motor efficiency degradation, Excessive infiltration	[170]
12	Decision tree	AHU	Measured	Supply air flow rate, supply fan speed, supply air flow rate, return fan power, return fan speed control signal, cooling coil valve control signal, supply/ mixed/ outdoor air temperature	Heating coil valve leakage, cooling coil valve stuck, return fan fixed speed or complete failure, outdoor/ exhaust air dampers stuck at fully	[108]

					closed, duct leakage of AHU before the supply fan	
13	ANN, SVR, MLR + Decision tree	AHU	Synthetic	Damper control signal, air flow rate, hot/ chilled water flow rate, indoor/ outdoor air temperature	VAV damper stuck which cannot open/ close fully, Decrease in VAV fan efficiency, flow rate reduction on hot/ chilled water and condenser pumps	[171]
14	Rule-based + decision tree	AHU (Outdoor air ventilation system and economizer)	Measured	Mixed/ return/ outdoor air temperature, supply fan on/off statues, heating/ cooling coil on/off status.	Sensor faults, air damper stuck open/ closed or a position	[35]
15	SVM + RBF ANN, PCA	AHU	Synthetic	Supply/ return fan speed, system static pressure, supply/ return air flow rate, supply air temperature	Sensors and actuator	[137]
16	SVM + model-based Naïve Bayes, RBF	AHU	Measured	Supply/ mixed air temperature, supply air humidity	air damper, fan failure, cooling coil valve, duct leakage	[21]
17	SVR	AHU	Synthetic	Supply/ outdoor air temperature, outdoor air humidity ratio, supply air flow rate, supply water temperature, control signal for water valve	Air temperature sensors	[172]
18	Bayesian network	Chiller	Measured	Cooling load, chilled water temperature, entering condenser water temperature, saturation temperature difference, refrigerant suction superheat temperature, refrigerant discharge superheat temperature, condensing temperature, sub-cooling temperature, condenser water temperature difference, evaporator water temperature,	Condenser fouling, excess oil, reduced condenser and evaporator water flow rate, non-condensable in the refrigerant, refrigerant leak/undercharge and overcharge	[173-174]
19	Bayesian network+ model-based	AHU	Measured	Supply/ return/ mixed air temperature, supply air flow rate, supply hot/ chilled water temperature of coil, coil control signal	Return/ outdoor/ mixing air damper leakage or stuck, heating/ cooling coil fouling, valve reverse,	[23-24]
20	Bayesian network + PCA	Chiller	Measured	Supply/ return evaporator/ condenser water temperature, sub-cooling temperature, refrigerant	Condenser fouling, reduced condenser and evaporator	[175]

				discharge temperature, temperature of oil in stump	water flow rate, non-condensable in the refrigerant, refrigerant leak/undercharge and overcharge	
21	Bayesian network + rule based	AHU	Measured	Supply air pressure/ air flow rate, differential pressure sensor, air damper signal, supply/ return air fan	Sensors, fans, dampers, ducts, filters	[36]
22			Measured	Supply/ mixed air temperature, supply air pressure, supply hot water temperature, supply air flow rate, supply air humidity ratio	Sensors (temperature), heating/cooling coil fouling, chilled water circulation pump	[37]
23	Rule-based + Bayesian Belief Network	AHU	Measured	Supply/ mixed/ return/ outdoor air temperature, cooling/ heating coil control signal,	Heating/ cooling coil leakage or fouling, air dampers stuck open/ closed/ partially, air temperature sensor error, low hot water temperature, high chilled water temperature, controller logic/ signal error	[38]
24	Clustering	Fan coil	Measured	Supply air temperature, heating/ cooling coil valve control signal, hot/ chilled water temperature	Energy	[176]
25	Clustering	AHU	Synthetic	Supply/ return chilled water temperature, chilled water flow rate, supply air temperature, air flow rate	Sensors, chilled water temperature	[177]
26	Clustering + PCA	Chiller	Measured	Supply/ return chilled water temperature, chilled water pump mass flow rate, condenser water entering/ leaving condenser, condenser water pump flow rate, oil temperature/ pressure, supply/ return chilled water pump	Sensors (supply and return chilled water temperature/ flow rate)	[178]
27	PCA	AHU	Measured	Supply/ outdoor/ return/ mixed air temperature, supply air flow rate/ static pressure, mixing box damper position, heating/ cooling vale position signal, supply/ return fan axial meter, supply/ return fan speed signal	Heating and cooling coil valve leakage, damper	[111]
28		Chiller	Measured	Outdoor air temperature, supply/ return chilled water temperature, supply condenser water temperature, supply cooling tower	Sensors (air, chilled water, condenser water temperatures),	[45]

				water temperature, electrical power input, cooling tower fan signal	electrical power, fan power	
29		Heating and hot water system	Measured	Supply/ return hot water temperature, supply hot water temperature	Sensors (heating water temperature)	[114]
30		Chiller	Measured	Evaporator/ condenser water supply/ return temperature, Compressor supply/ return refrigerant temperature, Compressor supply/ return specific volume, Compressor supply/ return enthalpy, Condenser/ refrigerant water mass flow rate, Compressor input power, Cooling capacity, Specific heat of water	Reduced evaporator water flow, refrigerant leakage, reduced condenser air flow, reduced compressor efficiency	[69]
31		Air source heat pump	Measured	Evaporator water supply/ return temperature, Evaporator refrigerant supply/ return temperature, Condenser refrigerant supply/ return temperature, Condenser air supply/ return temperature	Condenser fouling	[179]
32		AHU	Measured	Air flow rate for supply/ return/ exhaust/ outdoor/ recycle, outdoor air control signal	Air flow rate sensors errors (fixed and drifting bias, or complete failure)	[180]
33		AHU (VAV system)	Measured	Zone temperature, zone temperature set-point,	Control pressure error, sensors error, controller hard failure, terminal under/ over capacity, damper stuck	[181]
34		AHU	Measured & synthetic	Outdoor/ mixed/ return/ supply air temperature/ humidity ration/ flow rate	Sensors errors (air humidity, temperature, flow rate, pressure)	[182]
35	PCA + expert-rule	AHU (VAV)	Measured & synthetic	Zone temperature, air flow rate,	Sensor error (air temperature, air flow rate), VAV terminal under/ over capacity, damper stuck, poor tuning of static pressure control loop, controller failure,	[39]
36	PCA + SVDD	Chiller	Measured	Chilled water supply temperature, Entering condenser water temperature, Leaving condenser water temperature, Evaporator/ condenser approach temperature, Refrigerant sub-cooling temperature, Refrigerant discharge	condenser fouling, refrigerant overcharge and leakage, reduced condenser and evaporator water flow rate	[115, 183]

				temperature, Temperature of oil in sump		
37	PCA + ANN	VRF	Measured	Outdoor air temperature, compressor frequency, condensing/ evaporating saturation temperature, compressor discharge temperature, valve opening signal, compressor electric/ voltage, outdoor fan electric/ voltage,	Refrigeration under and over charge,	[184]
38	Deep ANN	VRF	Measured (Experimental)	Compressor/ fan operational frequency/ electric current, evaporating/ condensing saturation pressure, compressor discharge temperature, defrost temperature, etc.	Valve fault, refrigerant undercharge and overcharge, condenser fouling	[120]
39	CNN	Actuator	Synthetic (TRNSYS)	Zone air temperature, chilled water tank temperature, supply air temperature, supply/ return chilled water temperature, VAV damper control signal, coil valve signal,	Stuck damper, water valve	[132]
40	Rule-based	AHU	Measured (Experimental)	Supply/ mixed/ return/ outdoor air temperature / set point, cooling/ heating coil valve control signal, air damper control signal,	Sensors (air temperature, air flow), cooling/ heating coil valve leakage, damper stuck, mechanical cooling fault, too high airflow setpoint, slipping supply fan drive belt, undersized supply duct, disconnected zone temperature setpoint, zone temperature control loop tuning error, undersized VAV box	[27-29]
41			Measured (Experimental)	Supply/ mixed/ return/ outdoor air temperature, cooling/ heating coil valve control signal, occupancy status, mixing box damper control signal, return/ outdoor air relative humidity	Sensors (Air temperature), heating/ cooling coil valve leakage, heating/ cooling coil valve stuck, air damper stuck,	[30]
42			Measured (Lab)	Supply/ mixed/ return/ outdoor air temperature, cooling/ heating coil valve signal,	Sensors errors (air temperature)	[33]
43	model-based + Rule-based	AHU	Measured	Supply/ mixed air temperature/ pressure, supply/ return chilled water temperature, coil valve control signal	Sensors (air temperature), cooling coil valve	[25]

44			Measured	Supply/ return/ outdoor/ mixed air temperature, heating/ cooling coil valve position, supply air flow rate, electric input of boiler/ fan	Sensors (air temperature), heating/cooling valve opening faults or leakage, heat recovery pump	[26]
45	Grey-box models	Zone sensors and actuator	Measured (Experimental)	Supply/ outdoor/ room air temperature, differential pressure, heating/ cooling valve position, occupancy, VAV damper position signal/ pressure	Radiator vale stuck, reheat valve stuck, VAV damper stuck, VAV pressure sensor fault	[40]
46	FCD	AHU	Synthetic (TRNSYS)	Supply/ outdoor air temperature/ humidity ratio, cooling coil load, supply air humidity ratio, chilled water temperature, supply air damper position	Sensor error (supply air temperature)	[185]

9.2. Appendix B

Summary of literature review for the multiple fault detection and diagnosis in HVAC systems are provided in this section.

Table 9.2. Data-driven models for multiple faults detection and diagnosis applications in HVAC systems

No.	FDD model	HVAC system	Data type	Available parameters	FDD targets	References
1	ANN + clustering	AHU	Measured	Supply air temperature, supply/ return chilled water temperature, chiller water flow rate, chilled water valve position	Sensors (air temperature), chilled water valve	[112]
2	ANN		Synthetic	Water/ air control signal, supply/ return water temperature, supply/ return water flow rate, supply/return/outdoor air flow rate	Sensors (air and water temperatures)	[90]
3	ANN (Auto-associative neural network)	AHU	Synthetic	Zone air temperature, chilled water tank temperature, supply/ outdoor air temperature, return water temperature, zone VAV control signal, water tank valve signal	Sensors of air temperature	[91]
4	ANN + Fuzzy inference system	Vapor compression refrigerant	Measured	Low/ high side pressure, compressor suction temperature, input temperature of expansion device, refrigerant mass flow rate, compressor surface temperature	Sensors errors, refrigeration system faults	[186]
5	Recursive deterministic perceptron neural network	Whole HVAC	Synthetic	Fan efficiency, motor efficiency, water flow rate, motor efficiency, coefficient of performance for chiller	Energy consumption, Fan coil, pump, chiller	[187]
6	SVM KNN, Decision tree, Random forest, extreme learning machine	AHU	Measured	Supply/ return/ outdoor air temperature, supply/ mixed air humidity, cooling/ heating coil load, chilled/ hot water coil discharge air temperature	Damper, fan speed, cooling valve, duct leakage and fouling, coil fouling	[98]
7	SVM-RF-MLP SVM+DT/KNN/RF+RF/KNN/ MLP DT+KNN/RF+MLP/RF	AHU	Measured	Supply/ return/ mixed/ outdoor air temperature/ humidity, chilled water coil discharge air temperature, supply/ return air flow rate, cooling coil energy consumption, supply fan energy consumption	Air damper, fan, cooling coil valve, duct leakage	[188]

	KNN+RF+ MLP					
8	SVM	Chiller	Measured	Water temperature inlet/ outlet the condenser/ evaporator, oil feed pressure, condenser flow rate, evaporator valve position	Sensors, refrigerant leak and overcharge, condenser and evaporator flow rate	[99-101]
9	SVM, LR, LDA, QDA, KNN, Bagging, RF, AdaBoost, XGBoost	Packaged rooftop unit (RTU)	Synthetic	Supply/ return/ outdoor dry/wet bulb air temperature, liquid-line temperature/ pressure, suction temperature/ pressure, compressor discharge temperature, condenser out air temperature, refrigerant saturation temperature in the evaporator/ condenser, compressor power.	Refrigerant undercharge and overcharge, compressor valve leakage, condenser and evaporator flow reduction	[189]
10	Bayesian network	GSHP	Measured	Evaporator/ condenser temperature, evaporator/ condenser pressure, compressor suction temperature, compressor discharge temperature, evaporator/ condenser water temperature difference	Evaporator fouling, refrigerant discharge	[105]
11	Bayesian network+ model-based	Chiller	Measured	Supply/ return water temperature of evaporator/ condenser, condenser power, cooling power, control signal of water valve, water flow rate of condenser/ evaporator,	Compressor error, valve fault, reduction in condenser and evaporator water flow rate	[22]
12	SVDD	Chiller	Measured	Chilled water supply temperature, supply/ return evaporator water temperature, supply/ return condenser water temperature, evaporator/ condenser approach temperature, sub-cooling temperature, refrigerant discharge temperature, temperature of oil in sump, evaporating/ condenser temperature	Condenser fouling, excess oil, reduced condenser and evaporator water flow rate, non-condensable in the refrigerant, refrigerant leak/undercharge and overcharge	[190]
13	Linear discriminant analysis (LDA)	Chiller	Measured	Condenser/ evaporator water flow rate, oil range, refrigerant flow rate,	Condenser fouling, excess oil, reduced condenser and evaporator water flow rate, non-condensable in the refrigerant, refrigerant leak/undercharge and overcharge	[191]
14	Linear regressions	Chiller	Measured (Lab)	Supply/ return evaporator/ condenser water temperature, refrigerant condensing/ evaporating/ discharge suction line temperature, compressor power input, evaporator/ condenser water flow rate	Condenser and evaporator fouling, refrigerant low charge and overcharge,	[136]
15	Deep ANN	AHU	Synthetic (EnergyPlus)	Supply/ room air temperature, room set point temperature, cooling coil water flow rate, supply/ return chilled water temperature	Sensors (air temperature and flow rate), chilled water flow rate, VAV box dampers	[118]

16	RNN	AHU	Measure (Lab)	Supply/ return chilled water temperature, coil valve opening signal, damper position, supply air temperature	Sensors and actuator	[125]
17	CNN	Chiller and hot water storage	Measured (Experimental)	Supply water temperature/ flow rate, energy load, sewage temperature, chiller performance, heat exchanger area	Temperature sensors, condenser fouling, heat exchanger efficiency	[131, 192]
18	Rule-based + residual-based exponentially weighted moving average	AHU	Measured (Experimental)	Supply air temperature/ set point, return/ mixed/ outdoor air temperature, cooling coil control valve, supply air pressure/ setpoint,	chilled water pump, cooling coil fouling and valve leakage, filter, air temperature sensors, pressure sensors	[34]

9.3. Appendix C

9.3.1. Python codes for the SVR model

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
import scipy.stats
import sklearn.preprocessing
from joblib import dump, load
from sklearn.model_selection import RandomizedSearchCV
import time
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.metrics import r2_score
import tensorflow as tf
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
import time


data = np.loadtxt('Data_AHU_1.csv',delimiter=',',skiprows=1) #Loading the test set

#Data Preprocessing

T_oa_1 = data[:,0]
T_ra_1 = data[:,1]
V_ma_1 = data[:,2]
V_oa_1 = data[:,3]
V_ra_1 = data[:,4]
T_ma_1 = data[:,5]
HW_T_S_1 = data[:,6]
Valve_HC_1 = data[:,7]
T_ahc_1 = data[:,8]
T_sa_1 = data[:,9]

T_oa = []
T_ra = []
T_ma = []
T_ahc = []
T_sa = []
V_ma = []
V_oa = []
V_ra = []
HW_T_S = []
Valve_HC = []
for i in range (len(T_oa_1)):

    # T_oa
    if T_oa_1[i] > ((np.mean(T_oa_1)+(2*(np.std(T_oa_1))))):
        T_oa.append('nan')
    elif T_oa_1[i] < ((np.mean(T_oa_1)-(2*(np.std(T_oa_1))))):
        T_oa.append('nan')
    else:
        T_oa.append(T_oa_1[i])

    # T_ra
    if T_ra_1[i] > ((np.mean(T_ra_1)+(2*(np.std(T_ra_1))))):
        T_ra.append('nan')
    elif T_ra_1[i] < ((np.mean(T_ra_1)-(2*(np.std(T_ra_1))))):
        T_ra.append('nan')
    else:
        T_ra.append(T_ra_1[i])

    # T_ma
    if T_ma_1[i] > ((np.mean(T_ma_1)+(2*(np.std(T_ma_1))))):
        T_ma.append('nan')
    elif T_ma_1[i] < ((np.mean(T_ma_1)-(2*(np.std(T_ma_1))))):
        T_ma.append('nan')
    else:
        T_ma.append(T_ma_1[i])

    # T_ahc
    if T_ahc_1[i] > ((np.mean(T_ahc_1)+(2*(np.std(T_ahc_1))))):
        T_ahc.append('nan')
    elif T_ahc_1[i] < ((np.mean(T_ahc_1)-(2*(np.std(T_ahc_1))))):
        T_ahc.append('nan')
    else:
        T_ahc.append(T_ahc_1[i])

    # T_sa
    if T_sa_1[i] > ((np.mean(T_sa_1)+(2*(np.std(T_sa_1))))):
        T_sa.append('nan')
    elif T_sa_1[i] < ((np.mean(T_sa_1)-(2*(np.std(T_sa_1))))):
        T_sa.append('nan')
    else:
        T_sa.append(T_sa_1[i])

    # V_ma
    if V_ma_1[i] > ((np.mean(V_ma_1)+(2*(np.std(V_ma_1))))):
        V_ma.append('nan')
    elif V_ma_1[i] < ((np.mean(V_ma_1)-(2*(np.std(V_ma_1))))):
        V_ma.append('nan')
    else:
        V_ma.append(V_ma_1[i])

```

```
93
94      # V_oa
95      if V_oa_1[i] > ((np.mean(V_oa_1)+(2*(np.std(V_oa_1))))):
96          V_oa.append('nan')
97      elif V_oa_1[i] < ((np.mean(V_oa_1)-(2*(np.std(V_oa_1))))):
98          V_oa.append('nan')
99      else:
100         V_oa.append(V_oa_1[i])
101
102     # V_ra
103     if V_ra_1[i] > ((np.mean(V_ra_1)+(2*(np.std(V_ra_1))))):
104         V_ra.append('nan')
105     elif V_ra_1[i] < ((np.mean(V_ra_1)-(2*(np.std(V_ra_1))))):
106         V_ra.append('nan')
107     else:
108         V_ra.append(V_ra_1[i])
109
110     # HW_T_S_1
111     if HW_T_S_1[i] > ((np.mean(HW_T_S_1)+(2*(np.std(HW_T_S_1))))):
112         HW_T_S.append('nan')
113     elif HW_T_S_1[i] < ((np.mean(HW_T_S_1)-(2*(np.std(HW_T_S_1))))):
114         HW_T_S.append('nan')
115     else:
116         HW_T_S.append(HW_T_S_1[i])
117
118     # Valve_HC
119     if Valve_HC_1[i] > ((np.mean(Valve_HC_1)+(2*(np.std(Valve_HC_1))))):
120         Valve_HC.append('nan')
121     elif Valve_HC_1[i] < ((np.mean(Valve_HC_1)-(2*(np.std(Valve_HC_1))))):
122         Valve_HC.append('nan')
123     else:
124         Valve_HC.append(Valve_HC_1[i])
125
126
127 T_oa = np.array(T_oa).astype('float')
128 T_ra = np.array(T_ra).astype('float')
129 T_ma = np.array(T_ma).astype('float')
130 T_sa = np.array(T_sa).astype('float')
131 T_ahc = np.array(T_ahc).astype('float')
132 V_ma = np.array(V_ma).astype('float')
133 V_oa = np.array(V_oa).astype('float')
134 V_ra = np.array(V_ra).astype('float')
135 HW_T_S = np.array(HW_T_S).astype('float')
136 Valve_HC = np.array(Valve_HC).astype('float')
137
138
139 X_1 = np.column_stack((T_oa, T_ra, V_oa, V_ra, V_ma, T_ma))
140 X_1 = X_1[~(np.isnan(X_1)).any(axis=1)] #Remove NAN rows
141 X = X_1[:,0:5]
142
143
144
145 #y = T_ma
146 y = X_1[:,-1]
147 y = y.reshape((len(y)),1)
148
149
150 ypred_Tot = []
151 ytst_Tot = []
152 signal_Tot = []
153 RMSE_Tot = []
154 R2_Tot = []
155 Me_max_Tot = []
156 MAPe_Tot = []
157 MBE_Tot = []
158
159 #for d in range (3,15,2):
160 for d in range (3,4):
161
162     days_trn = d #Number of days for training
163     days_tst = 1 #Number of days for Testing
164
165
166     ypred = []
167     ytst = []
168     signal = []
169     RMSE = []
170     R2 = []
171     Me_max = []
172     MAPe = []
173     MBE = []
174
175     for i in range (6): # number of sliding windows
176
177         X_trn = X[(15*24*4)-(d*24*4):(15*24*4)]
178         y_trn = y[(15*24*4)-(d*24*4):(15*24*4)]
179
180         X_tst = X[(15*24*4) + ((i)*24*4) : (15*4*24) + ((i+1)*24*4)]
181         y_tst = y[(15*24*4) + ((i)*24*4) : (15*4*24) + ((i+1)*24*4)]
182
183         #Scaling the Training set
```

```
        #Scaling the Training set

        sc_x = StandardScaler()
        sc_y = StandardScaler()
        X_trn = sc_x.fit_transform(X_trn)
        y_trn = sc_y.fit_transform(y_trn)

        X_tst = sc_x.transform(X_tst)

        ## ANN model
        startTime = time.time()

        SVR = sklearn.svm.SVR(kernel='rbf')

        RSVR = sklearn.model_selection.RandomizedSearchCV(SVR, param_distributions={'C':scipy.stats.reciprocal(1, 50),
                                        'gamma':scipy.stats.reciprocal(0.01, 10)},
                                        n_iter=30, random_state=0,cv=10)
        RSVR.fit(X_trn, y_trn.ravel())

        y_pred = sc_y.inverse_transform(RSVR.predict(X_tst))

        y_pred = y_pred.reshape(len(y_pred),1)
        #y_pred = ann.predict(X_tst)

        signal_T_ma = (abs(y_tst- y_pred) > 1)*1

        sig = signal_T_ma.astype('float')
        sig[sig == 0] = 'nan'
        signal_plot_T_ma = y_tst * sig


        ytst.append(y_tst)
        ypred.append(y_pred)
        signal.append(signal_plot_T_ma)

        RMSE.append(sqrt(mean_squared_error(y_pred, y_tst)))
        R2.append(r2_score(y_pred, y_tst)*100)
        Me_max.append(max(abs(y_pred-y_tst)))
        MAPe.append((sum(abs((y_pred-y_tst)/y_tst)) * 100)/96)
        MBE.append((sum(abs(y_pred-y_tst)))/96)


    ypred_Tot.append(ypred)
    ytst_Tot.append(ytst)
    signal_Tot.append(signal)
    RMSE_Tot.append(RMSE)
    R2_Tot.append(R2)
    Me_max_Tot.append(Me_max)
    MAPe_Tot.append(MAPe)
    MBE_Tot.append(MBE)

avg_RMSE = np.mean(RMSE_Tot)
avg_R2 = np.mean(R2_Tot)
avg_Me_max = np.mean(Me_max_Tot)
avg_MAPe = np.mean(MAPe_Tot)
avg_MBE = np.mean(MBE_Tot)



p = 0

#Plotting
ypred_7ddays = np.vstack(((ypred_Tot[p][0]),(ypred_Tot[p][1]),(ypred_Tot[p][2]),(ypred_Tot[p][3]),(ypred_Tot[p][4])))#,(ypred_Tot[p]
[5])))
ytst_7ddays = np.vstack(((ytst_Tot[p][0]),(ytst_Tot[p][1]),(ytst_Tot[p][2]),(ytst_Tot[p][3]),(ytst_Tot[p][4])))#,(ytst_Tot[p][5])))


Avg_RMSE = np.mean(RMSE)

sign = (abs(ytst_7ddays- ypred_7ddays) > (0.49 + Avg_RMSE) )*1
sig = sign.astype('float')
sig[sig == 0] = 'nan'
signal_plot_T_ma_1 = np.array(ytst_7ddays) * sig

signal_plot_T_ma_3 = np.empty((300,1))
signal_plot_T_ma_3.fill(np.nan)

signal_plot_T_ma = np.vstack((signal_plot_T_ma_3 ,signal_plot_T_ma_1[300:]))

fig=plt.figure(figsize=(20,8))
plt.plot(np.array(ytst_7ddays).reshape(-1,1), '-', label='Measured')
plt.plot(np.array(ypred_7ddays).reshape(-1,1), '--', label='Predicted')
plt.plot(np.array(signal_plot_T_ma).reshape(-1,1),'o',color='r',label='Fault flag')
plt.xlabel('Measured every 15-min',fontsize=18)
plt.ylabel('Mixed air temperature (T_ma) (C)',fontsize=18)
#plt.xticks(np.arange(0, (len(ytst_15ddays))+40, 40),fontsize=18)
plt.xticks(np.arange(0, (len(ytst_7ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
plt.yticks(np.arange(10, 16, 1),fontsize=18)
plt.legend(fontsize=18)
fig.savefig('SVR_T_ma.jpg')
```

```
############## Artificial Faults generation ###############

T_ma_Test = ytst_7ddays
T_ma_Predict = ypred_7ddays

## T_ma faulty
T_ma_Faulty = T_ma_Test[:,0] + np.hstack(([0]*288,np.linspace(0.5,5,192)))
T_ma_Faulty  = np.reshape(T_ma_Faulty, (-1,1))

sign_Test = (abs(T_ma_Test - T_ma_Faulty) > (0.49 + Avg_RMSE) )*1
sign = (abs(T_ma_Faulty - T_ma_Predict) > (0.49 + Avg_RMSE) )*1
sig = sign.astype('float')
sig[sig == 0] = 'nan'
signal_plot_T_ma_1 = np.array(T_ma_Faulty) * sig


signal_plot_T_ma_3 = np.empty((280,1))
signal_plot_T_ma_3.fill(np.nan)

signal_plot_T_ma = np.vstack((signal_plot_T_ma_3 ,signal_plot_T_ma_1[280:]))


fig=plt.figure(figsize=(20,8))
plt.plot(np.array(T_ma_Faulty).reshape(-1,1), '-', label='Measured')
plt.plot(np.array(T_ma_Predict).reshape(-1,1), '--', label='Predicted')
plt.plot(np.array(signal_plot_T_ma).reshape(-1,1),'o',color='r',label='Fault flag')
plt.xlabel('Measured every 15-min',fontsize=18)
plt.ylabel('Mixed air temperature (T_ma) (C)',fontsize=18)
#plt.xticks(np.arange(0, (len(ytst_15ddays))+40, 40),fontsize=18)
plt.xticks(np.arange(0, (len(ytst_7ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
plt.yticks(np.arange(10, 23, 2),fontsize=18)
plt.legend(fontsize=18)
fig.savefig('SVR_T_ma_Faulty.jpg')



fig=plt.figure(figsize=(20,8))
plt.plot(abs(np.array(T_ma_Faulty).reshape(-1,1) - np.array(T_ma_Predict).reshape(-1,1)), '-')#, label='Residual')
plt.xlabel('Measured every 15-min',fontsize=18)
plt.ylabel('T_ma(Measured) - T_ma(Predicted by SVR) (C)',fontsize=18)
plt.xticks(np.arange(0, (len(ytst_7ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
plt.yticks(np.arange(0, 7, 1),fontsize=18)
plt.hlines(y=(0.49 + Avg_RMSE), xmin=0, xmax=480, linewidth=2, linestyles='--', color='r', label='Threshold');
plt.legend(fontsize=18)
fig.savefig('SVR_T_ma_Residual.jpg')



# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score

cm_3 = confusion_matrix(sign, sign_Test)

Accuracy = accuracy_score(sign, sign_Test) *100
Precision = precision_score(sign, sign_Test)*100
Sensitivity = recall_score(sign, sign_Test)*100


################


## LSM fault generation for T_ma

### T_oa and T_ra Faulty


T_oa = X[(15*24*4) + ((0)*24*4) : (15*4*24) + ((4+1)*24*4),0]
T_oa = np.reshape(T_oa, (-1,1))
T_ra = X[(15*24*4) + ((0)*24*4) : (15*4*24) + ((4+1)*24*4),1]
T_ra = np.reshape(T_ra, (-1,1))
T_ma = ypred_7ddays



T_oa_Faulty = T_oa + np.reshape(np.hstack(([0]*288, np.linspace(0.5,5,192))), (-1,1))

T_ra_Faulty = T_ra + np.reshape(np.hstack(([0]*288, np.linspace(0.5,5,192))), (-1,1))

T_ma_Faulty = 0.536*(T_oa_Faulty) + 0.562*(T_ra_Faulty)


#Avg_RMSE = 0.536

sign = (abs(T_ma_Faulty- T_ma) > (0.49 + Avg_RMSE) )*1
sig = sign.astype('float')
sig[sig == 0] = 'nan'
signal_plot_T_ma_1 = np.array(T_ma_Faulty) * sig


```

```
365
366 signal_plot_T_ma_3 = np.empty((288,1))
367 signal_plot_T_ma_3.fill(np.nan)
368
369 signal_plot_T_ma = np.vstack((signal_plot_T_ma_3 ,signal_plot_T_ma_1[288:480]))
370
371
372 fig=plt.figure(figsize=(20,8))
373 plt.plot(T_ma_Faulty[:480], '-', label='Measurements including faults using Grey-box model')
374 plt.plot(T_ma[:480], '--', label='Predicted using SVR model')
375 plt.plot(np.array(signal_plot_T_ma).reshape(-1,1),'o',color='r',label='Fault flag')
376 plt.xlabel('Measured every 15-min',fontsize=18)
377 plt.ylabel('Mixed air temperature (T_ma) (C)',fontsize=18)
378 #plt.xticks(np.arange(0, (len(ytst_15ddays))+40, 40),fontsize=18)
379 plt.xticks(np.arange(0, (len(T_ma[:480]))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
380 plt.yticks(np.arange(10, 22, 2),fontsize=18)
381 plt.legend(fontsize=18)
382 fig.savefig('SVR_T_ma_Due_to_T_oa_T_ra.jpg')
383
384
385 T_ma_Faulty_2 = T_ma_Test[:,0] + np.hstack(([0]*288,np.linspace(0.5,5,192)))
386 T_ma_Faulty_2 = np.reshape(T_ma_Faulty, (-1,1))
387
388 fig=plt.figure(figsize=(20,8))
389 plt.plot(abs(T_ma_Faulty[:480] - T_ma_Predict[:480]), '-', label='Measured T_ma (artificial faults) - T_ma,p')#, Label='Residual')
390 plt.plot(abs(T_ma_Faulty[:480] - T_ma_Faulty_2[:480]), '^-',color='green', label='Measured T_ma (artificial faults) - T_ma,R')#,
    Label='Residual')
391 plt.xlabel('Measured every 15-min',fontsize=18)
392 plt.ylabel('Measured T_ma (artificial faults)',fontsize=18)
393 plt.xticks(np.arange(0, (len(ytst_7ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
394 plt.yticks(np.arange(0, 7, 1),fontsize=18)
395 plt.hlines(y=(0.49 + Avg_RMSE), xmin=0, xmax=480, linewidth=2, linestyles='--', color='r', label='Threshold');
396 plt.legend(fontsize=18)
397 fig.savefig('SVR_T_ma_Due_to_T_oa_T_ra_Residual.jpg')
398
399
400
401 ## Normal T_ma and Predicted T_ma using Grey-box
402
403 T_ma_grey_box = 0.536*(T_oa) + 0.562*(T_ra)
404
405 T_ma_Faulty_grey_box = 0.536*(T_oa_Faulty) + 0.562*(T_ra_Faulty)
406
407 T_ma_Faulty_2 = T_ma_Test[:,0] + np.hstack(([0]*288,np.linspace(0.5,5,192)))
408 T_ma_Faulty_2 = np.reshape(T_ma_Faulty, (-1,1))
409
410 fig=plt.figure(figsize=(20,8))
411
412 plt.plot(abs(T_ma_Faulty_grey_box[:480] - T_ma_Faulty_2[:480]), '-')#, Label='Residual')
413
414 #plt.plot(abs(T_ma_grey_box[:480] - T_ma_Test[:480]), '-')#, Label='Residual')
415 plt.xlabel('Measured every 15-min',fontsize=18)
416 plt.ylabel('T_ma(Measured) - T_ma(Predicted by Grey-box model) (C)',fontsize=18)
417 plt.xticks(np.arange(0, (len(ytst_7ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21'),fontsize=18)
418 plt.yticks(np.arange(0, 7, 1),fontsize=18)
419 plt.hlines(y=(0.49 + Avg_RMSE), xmin=0, xmax=480, linewidth=2, linestyles='--', color='r', label='Threshold');
420 plt.legend(fontsize=18)
421 fig.savefig('SVR_T_ma_grey_box_and_faulty_Residual.jpg')
422
423
424
425 from sklearn.metrics import confusion_matrix, accuracy_score
426 from sklearn.metrics import precision_score
427 from sklearn.metrics import recall_score
428
429 cm_3 = confusion_matrix(sign, sign_Test)
430
431 Accuracy = accuracy_score(sign, sign_Test) *100
432 Precision = precision_score(sign, sign_Test)*100
433 Sensitivity = recall_score(sign, sign_Test)*100
434
```

9.3.2. Python codes for the Random Forest model

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
import scipy.stats
import sklearn.preprocessing
from joblib import dump, load
from sklearn.model_selection import RandomizedSearchCV
import time
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.metrics import r2_score
import tensorflow as tf
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
import time
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor


data = np.loadtxt('Data_AHU_1.csv',delimiter=',',skiprows=1) #Loading the test set

#Data Preprocessing

T_oa_1 = data[:,0]
T_ra_1 = data[:,1]
V_ma_1 = data[:,2]
V_oa_1 = data[:,3]
V_ra_1 = data[:,4]
T_ma_1 = data[:,5]
HW_T_S_1 = data[:,6]
Valve_HC_1 = data[:,7]
T_ahc_1 = data[:,8]
T_sa_1 = data[:,9]

T_oa = []
T_ra = []
T_ma = []
T_ahc = []
T_sa = []
V_ma = []
V_oa = []
V_ra = []
HW_T_S = []
Valve_HC = []
for i in range (len(T_oa_1)):

    # T_oa
    if T_oa_1[i] > ((np.mean(T_oa_1)+(2*(np.std(T_oa_1))))):
        T_oa.append('nan')
    elif T_oa_1[i] < ((np.mean(T_oa_1)-(2*(np.std(T_oa_1))))):
        T_oa.append('nan')
    else:
        T_oa.append(T_oa_1[i])

    # T_ra
    if T_ra_1[i] > ((np.mean(T_ra_1)+(2*(np.std(T_ra_1))))):
        T_ra.append('nan')
    elif T_ra_1[i] < ((np.mean(T_ra_1)-(2*(np.std(T_ra_1))))):
        T_ra.append('nan')
    else:
        T_ra.append(T_ra_1[i])

    # T_ma
    if T_ma_1[i] > ((np.mean(T_ma_1)+(2*(np.std(T_ma_1))))):
        T_ma.append('nan')
    elif T_ma_1[i] < ((np.mean(T_ma_1)-(2*(np.std(T_ma_1))))):
        T_ma.append('nan')
    else:
        T_ma.append(T_ma_1[i])

    # T_ahc
    if T_ahc_1[i] > ((np.mean(T_ahc_1)+(2*(np.std(T_ahc_1))))):
        T_ahc.append('nan')
    elif T_ahc_1[i] < ((np.mean(T_ahc_1)-(2*(np.std(T_ahc_1))))):
        T_ahc.append('nan')
    else:
        T_ahc.append(T_ahc_1[i])

    # T_sa
    if T_sa_1[i] > ((np.mean(T_sa_1)+(2*(np.std(T_sa_1))))):
        T_sa.append('nan')
    elif T_sa_1[i] < ((np.mean(T_sa_1)-(2*(np.std(T_sa_1))))):
        T_sa.append('nan')
    else:
        T_sa.append(T_sa_1[i])

```

```
    # V_ma
    if V_ma_1[i] > ((np.mean(V_ma_1)+(2*(np.std(V_ma_1))))):
        V_ma.append('nan')
    elif V_ma_1[i] < ((np.mean(V_ma_1)-(2*(np.std(V_ma_1))))):
        V_ma.append('nan')
    else:
        V_ma.append(V_ma_1[i])

    # V_oa
    if V_oa_1[i] > ((np.mean(V_oa_1)+(2*(np.std(V_oa_1))))):
        V_oa.append('nan')
    elif V_oa_1[i] < ((np.mean(V_oa_1)-(2*(np.std(V_oa_1))))):
        V_oa.append('nan')
    else:
        V_oa.append(V_oa_1[i])

    # V_ra
    if V_ra_1[i] > ((np.mean(V_ra_1)+(2*(np.std(V_ra_1))))):
        V_ra.append('nan')
    elif V_ra_1[i] < ((np.mean(V_ra_1)-(2*(np.std(V_ra_1))))):
        V_ra.append('nan')
    else:
        V_ra.append(V_ra_1[i])

    # HW_T_S_1
    if HW_T_S_1[i] > ((np.mean(HW_T_S_1)+(2*(np.std(HW_T_S_1))))):
        HW_T_S.append('nan')
    elif V_ra_1[i] < ((np.mean(HW_T_S_1)-(2*(np.std(HW_T_S_1))))):
        HW_T_S.append('nan')
    else:
        HW_T_S.append(V_ra_1[i])

    # Valve_HC
    if Valve_HC_1[i] > ((np.mean(Valve_HC_1)+(2*(np.std(Valve_HC_1))))):
        Valve_HC.append('nan')
    elif Valve_HC_1[i] < ((np.mean(Valve_HC_1)-(2*(np.std(Valve_HC_1))))):
        Valve_HC.append('nan')
    else:
        Valve_HC.append(Valve_HC_1[i])


T_oa = np.array(T_oa).astype('float')
T_ra = np.array(T_ra).astype('float')
T_ma = np.array(T_ma).astype('float')
T_sa = np.array(T_sa).astype('float')
T_ahc = np.array(T_ahc).astype('float')
V_ma = np.array(V_ma).astype('float')
V_oa = np.array(V_oa).astype('float')
V_ra = np.array(V_ra).astype('float')
HW_T_S = np.array(HW_T_S).astype('float')
Valve_HC = np.array(Valve_HC).astype('float')



X_1 = np.column_stack((T_ma, V_ma, HW_T_S, Valve_HC, T_ahc))
X_1 = X_1[~(np.isnan(X_1)).any(axis=1)] #Remove NAN rows
X = X_1[:,0:5]

#y = T_ma
y = X_1[:,-1]
y = y.reshape((len(y)),1)



startTime = time.time()

ypred_Tot = []
ytst_Tot = []
signal_Tot = []
RMSE_Tot = []
R2_Tot = []
Me_max_Tot = []
MAPe_Tot = []
MBE_Tot = []

#for d in range (3,15,2):
for d in range (3,4):

    days_trn = d #Number of days for training
    days_tst = 1 #Number of days for Testing


    ypred = []
    ytst = []
    signal = []
    RMSE = []
    R2 = []
    Me_max = []
    MAPe = []
    MBE = []
```

```
    for i in range (6): # number of sliding windows



        X_trn = X[(15*24*4)-(d*24*4):(15*24*4)]
        y_trn = y[(15*24*4)-(d*24*4):(15*24*4)]

        X_tst = X[(15*24*4) + ((i)*24*4) : (15*4*24) + ((i+1)*24*4)]
        y_tst = y[(15*24*4) + ((i)*24*4) : (15*4*24) + ((i+1)*24*4)]


        #Scaling the Training set

        sc_x = StandardScaler()
        sc_y = StandardScaler()
        X_trn = sc_x.fit_transform(X_trn)
        y_trn = sc_y.fit_transform(y_trn)

        X_tst = sc_x.transform(X_tst)

        ## ANN model


        val = ([1,20,40,60,80,100,120,140,160,180,200,300,400])

        RF = sklearn.ensemble.RandomForestRegressor()

        RRF = sklearn.model_selection.RandomizedSearchCV(RF, param_distributions={'n_estimators':val,
                                                          'max_depth':scipy.stats.reciprocal(1, 20)},
                                                           n_iter=30, random_state=0,cv=10)
        RRF.fit(X_trn, y_trn.ravel())

        y_pred = sc_y.inverse_transform(RRF.predict(X_tst))

        y_pred = y_pred.reshape(len(y_pred),1)
        #y_pred = ann.predict(X_tst)

        signal_T_ma = (abs(y_tst- y_pred) > 1)*1

        sig = signal_T_ma.astype('float')
        sig[sig == 0] = 'nan'
        signal_plot_T_ma = y_tst * sig


        ytst.append(y_tst)
        ypred.append(y_pred)
        signal.append(signal_plot_T_ma)

        RMSE.append(sqrt(mean_squared_error(y_pred, y_tst)))
        R2.append(r2_score(y_pred, y_tst)*100)
        Me_max.append(max(abs(y_pred-y_tst)))
        MAPe.append((sum(abs((y_pred-y_tst)/y_tst)) * 100)/96)
        MBE.append((sum(abs(y_pred-y_tst)))/96)


    ypred_Tot.append(ypred)
    ytst_Tot.append(ytst)
    signal_Tot.append(signal)
    RMSE_Tot.append(RMSE)
    R2_Tot.append(R2)
    Me_max_Tot.append(Me_max)
    MAPe_Tot.append(MAPe)
    MBE_Tot.append(MBE)

avg_RMSE = np.mean(RMSE_Tot)
avg_R2 = np.mean(R2_Tot)
avg_Me_max = np.mean(Me_max_Tot)
avg_MAPe = np.mean(MAPe_Tot)
avg_MBE = np.mean(MBE_Tot)


```

9.3.3. Python codes for the RNN model

```
# Recurrent Neural Network

# Part 1 - Data Preprocessing

# Import the libraries
import numpy as np # numpy makes able to creat arrays
import matplotlib.pyplot as plt # in order to creat graphs
import pandas as pd # in order to import data set
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.metrics import r2_score

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout

#T_oa = X[:,0]
#T_ra = X[:,1]
#V_ma = X[:,2]
#V_oa = X[:,3]
#V_ra = X[:,4]
#T_ma = X[:,5]
#HW_T_S = X[:,6]
#Valve_HC_AHU = X[:,7]
#T_ahc = X[:,8]
#T_sa = X[:,9]

# Importing the training set
dataset_train = pd.read_csv('Data_AHU_1.csv')



ypred_Tot = []
ytst_Tot = []
signal_Tot = []
RMSE_Tot = []
R2_Tot = []
MAPe_Tot = []
MBe_Tot = []
Me_max_Tot = []
for d in range (6):
#d = 0
    D = d * 24 * 4 # Time_steps of Day number
    V =  # Which variable respect to the column number
    H = 3*24*4  #Training hours

    T = 0 #testing hour

    training_set = dataset_train.iloc[0+D:H+D, V:V+1].values # we are only getting the first index from 1 to 2 we only get the first index and 2 is excluded - and with .values with get numpy array

    # Feature Scaling (Normalisation)
    from sklearn.preprocessing import MinMaxScaler
    sc = MinMaxScaler(feature_range = (0, 1))
    training_set_scaled = sc.fit_transform(training_set)

    # Creating a data structure with 10 timesteps and 1 output
    X_train = []
    y_train = []
    for i in range (10, H): #(time_lag, following time_steps)
        X_train.append(training_set_scaled[i-10:i, 0]) # 10 time-lags
        y_train.append(training_set_scaled[i, 0])
    X_train, y_train = np.array(X_train), np.array(y_train)

    # Reshaping
    X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))


```

```
    # Part 2 - Building the RNN

    # Importing the Keras libraries and packages available


    # Initialising the RNN
    regressor = Sequential()

    # Adding the first LSTM layer and some Dropout regularisation
    regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
    regressor.add(Dropout(0.2))

    # Adding the second LSTM layer and some Dropout regularisation
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))

    # Adding the third LSTM layer and some Dropout regularisation
    regressor.add(LSTM(units = 50, return_sequences = True))
    regressor.add(Dropout(0.2))

    # Adding the fourth LSTM layer and some Dropout regularisation
    regressor.add(LSTM(units = 50))
    regressor.add(Dropout(0.2))

    # Adding the output layer
    regressor.add(Dense(units = 1))

    # Compiling the RNN
    regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

    # Fitting the RNN to the training set
    regressor.fit(X_train, y_train, epochs = 1000, batch_size = 32)


    # Part 3 - Making the predictions and visualising the results

    # Getting the real stock price of 2017
    dataset_test = pd.read_csv('Data_AHU_1.csv')
    dataset_test = dataset_test.iloc[H+D:H+D+96, V:V+1].values

    real_stock_price = dataset_train.iloc[H+D:H+D+96, V:V+1].values


    # Getting the predicted stock price of 2017
    dataset_total  =(np.concatenate((training_set.reshape(len(training_set),1), dataset_test .reshape(len(dataset_test),1)),0))
    #dataset_total = pd.concat((dataset_train['(T_ra)'], dataset_test['(T_ra)']), axis = 0)
    inputs = dataset_total[len(dataset_total) - len(dataset_test) - 10:]
    inputs = inputs.reshape(-1,1)
    inputs = sc.transform(inputs)

    X_test = []
    for i in range(10, 106):
        X_test.append(inputs[i-10:i, 0])
    X_test = np.array(X_test)
    X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))

    predicted_stock_price = regressor.predict(X_test)
    predicted_stock_price = sc.inverse_transform(predicted_stock_price)

    rmse = sqrt(mean_squared_error(real_stock_price, predicted_stock_price))
    R2 = r2_score(predicted_stock_price, real_stock_price)*100
    MAPe = (sum(abs((real_stock_price-predicted_stock_price)/real_stock_price )) * 100)/24
    MBe = (sum(abs(real_stock_price-predicted_stock_price)))/24
    Me_max = max(abs(real_stock_price- predicted_stock_price))

    ypred_Tot.append(predicted_stock_price)
    ytst_Tot.append(dataset_test)
    RMSE_Tot.append(rmse)
    R2_Tot.append(R2)
    MAPe_Tot.append(MAPe)
    MBe_Tot.append(MBe)
    Me_max_Tot.append(Me_max)


RMSE_Final = np.mean(RMSE_Tot)
R2_Final = np.mean(R2_Tot)
MAPe_Final = np.mean(MAPe_Tot)
MBe_Final = np.mean(MBe_Tot)
Me_max_Final = np.mean(Me_max_Tot)

# Visualising the results


fig=plt.figure(figsize=(20,8))
plt.plot(real_stock_price, color = 'red',label = 'Measured')
plt.plot(predicted_stock_price, color = 'blue',label = 'Predicted')
plt.xlabel('6 days (15-minutes)')
plt.ylabel('Mixed air temperature (V_ma) (L/s)')
plt.xticks(np.arange(0, (len(real_stock_price)), 5))
plt.legend()
plt.show()
```

```
ypred_15ddays = np.vstack(((ypred_Tot[0]),(ypred_Tot[1]),(ypred_Tot[2]),(ypred_Tot[3]),(ypred_Tot[4]),(ypred_Tot[5])))
ytst_15ddays = np.vstack(((ytst_Tot[0]),(ytst_Tot[1]),(ytst_Tot[2]),(ytst_Tot[3]),(ytst_Tot[4]),(ytst_Tot[5])))


sign = (abs(ytst_15ddays- ypred_15ddays) > (2+RMSE_Final) )*1
sig = sign.astype('float')
sig[sig == 0] = 'nan'
signal_plot_T_ma = np.array(ytst_15ddays) * sig

fig=plt.figure(figsize=(20,8))
plt.plot(np.array(ytst_15ddays).reshape(-1,1), '-', label='Measured')
plt.plot(np.array(ypred_15ddays).reshape(-1,1), '--', label='Predicted')
plt.plot(np.array(signal_plot_T_ma).reshape(-1,1),'o',color='r',label='Fault Flag')
plt.xlabel('Measured every 15-min',fontsize=18)
plt.ylabel('Supply hot water temperature (HW_T_S) (C)',fontsize=18)
plt.xticks(np.arange(0, (len(ytst_15ddays))+40, 40),fontsize=18)
plt.yticks(np.arange(37, 48, 2),fontsize=18)
plt.xticks(np.arange(0, (len(ytst_15ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21','Jan 22'),fontsize=18)
plt.legend(fontsize=18)
fig.savefig('RNN_T_ma.jpg')




## T_oa Faulty

T_oa_Test = ytst_15ddays
T_oa_Predict = ypred_15ddays

T_oa = np.reshape(T_oa_Test, (-1,1))
T_oa_Faulty = T_oa_Test[:,0] + np.hstack(([0]*288,np.linspace(0.5,5,288)))
T_oa_Faulty  = np.reshape(T_oa_Faulty, (-1,1))

sign_Test = (abs(T_oa_Test - T_oa_Faulty) > (0.49 + RMSE_Final) )*1
sign = (abs(T_oa_Faulty - T_oa_Predict) > (0.49 + RMSE_Final) )*1
sig = sign.astype('float')
sig[sig == 0] = 'nan'
signal_plot_T_oa = np.array(T_oa_Faulty) * sig

fig=plt.figure(figsize=(20,8))
plt.plot(np.array(T_oa_Faulty).reshape(-1,1), '-', label='Measured')
plt.plot(np.array(T_oa_Predict).reshape(-1,1), '--', label='Predicted')
plt.plot(np.array(signal_plot_T_oa).reshape(-1,1),'o',color='r',label='Fault flag')
plt.xlabel('Measured every 15-min',fontsize=18)
plt.ylabel('Outdoor air temperature (T_oa) (C)',fontsize=18)
#plt.xticks(np.arange(0, (len(ytst_15ddays))+40, 40),fontsize=18)
plt.xticks(np.arange(0, (len(ytst_15ddays))+96, 96), ('Jan 16','Jan 17','Jan 18','Jan 19','Jan 20','Jan 21','Jan 22'),fontsize=18)
plt.yticks(np.arange(-16, 8, 2),fontsize=18)
plt.legend(fontsize=18)
fig.savefig('SVR_T_oa_Faulty.jpg')


# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score

cm_3 = confusion_matrix(sign, sign_Test)

Accuracy = accuracy_score(sign, sign_Test) *100
Precision = precision_score(sign, sign_Test)*100
Sensitivity = recall_score(sign, sign_Test)*100

```

9.3.4. Python codes for the classification machine learning models model

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
tf.__version__

# Importing the dataset
dataset = pd.read_csv('Data1.csv')
#dataset = pd.read_excel("Data-periodically.xlsx", "Sheet1")

x = dataset.iloc[:1440, 3:12].values
y = dataset.iloc[:, 19:24].values


## Creating Random values for Faults and shuffling
Random_faults = np.random.uniform(0.5,3,720) # creat random values between 0 and 3
z = np.zeros(720)
Faults = np.concatenate([Random_faults,z])
df = pd.DataFrame(Faults)

## T_oa Faults
Faults_T_oa = df.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_oa = Faults_T_oa.to_numpy().flatten() # Convert DataFrame to Numpy

## T_oa Faults
Faults_T_ra = df.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ra = Faults_T_ra.to_numpy().flatten() # Convert DataFrame to Numpy

## T_ma Faults
Faults_T_ma = df.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ma = Faults_T_ma.to_numpy().flatten() # Convert DataFrame to Numpy

## T_ahc Faults
Faults_T_ahc = df.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ahc = Faults_T_ahc.to_numpy().flatten() # Convert DataFrame to Numpy

## T_sa Faults
Faults_T_sa = df.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_sa = Faults_T_sa.to_numpy().flatten() # Convert DataFrame to Numpy

#Faults = dataset.iloc[:720,13].values

n = 10 # number of total days
H = n*24 # number of total hours

T_oa = x[:,0]
T_ra = x[:,1]
T_ma = x[:,5]
T_ahc = x[:,7]
T_sa = x[:,8]

Q = x[:,6]
V_ma = x[:,2]
V_oa = x[:,3]
V_ra = x[:,4]


## T_oa
T_oa_F = T_oa+Faults_T_oa
T_oa_signal = abs(T_oa_F-T_oa)

## T_ra
T_ra_F = T_ra+Faults_T_ra
T_ra_signal = abs(T_ra_F-T_ra)

## T_ma
T_ma_F= ( (0.2114)*T_oa_F + (0.7428)*T_ra_F ) + Faults_T_ma
T_ma_signal = abs(T_ma_F-T_ma)

## T_ahc
Q_F = Q

V_ma_F = V_ma

T_ahc_F = ( ((0.1871*Q_F)/(0.2316*V_ma_F))+(1*T_ma_F) ) + Faults_T_ahc
T_ahc_signal = abs(T_ahc_F-T_ahc)


## T_ra
T_sa_F= (T_ahc_F+1.8) + Faults_T_sa
T_sa_signal = abs(T_sa_F-T_sa)

## Combination of Temperature with fault
# Faulty Data set
X_Fault =(np.concatenate((T_oa_F.reshape(len(T_oa_F),1),
T_ra_F.reshape(len(T_ra_F),1),V_ma.reshape(len(V_ma),1),V_oa.reshape(len(V_oa),1),V_ra.reshape(len(V_ra),1),
T_ma_F.reshape(len(T_ma_F),1),Q.reshape(len(Q),1), T_ahc_F.reshape(len(T_ahc_F),1), T_sa_F.reshape(len(T_sa_F),1)),1))

```

```
Input = X_Fault
X_train = Input

Output = (np.concatenate((T_oa_signal.reshape(len(T_oa_signal ),1), T_ra_signal .reshape(len(T_ra_signal ),1),
T_ma_signal.reshape(len(T_ma_signal),1), T_ahc_signal.reshape(len(T_ahc_signal),1), T_sa_signal.reshape(len(T_sa_signal),1)),1))
Output = (Output > 0.5)*1
y_train = Output


plt.plot(X_Fault[:,0])
plt.plot(x[:,0])



#### Testing Data Set
# Importing the dataset
dataset = pd.read_csv('Data1.csv')
#dataset = pd.read_excel("Data-periodically.xlsx", "Sheet1")

x_test = dataset.iloc[1440:1800, 3:12].values


## Creating Random values for Faults and shuffling
Random_faults_test = np.random.uniform(0.5,3,180) # creat random values between 0 and 3
z_test = np.zeros(180)
Faults_test = np.concatenate([Random_faults_test,z_test])
df_test = pd.DataFrame(Faults_test)

## T_oa Faults
Faults_T_oa_test = df_test.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_oa_test = Faults_T_oa_test.to_numpy().flatten() # Convert DataFrame to Numpy

## T_oa Faults
Faults_T_ra_test = df_test.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ra_test = Faults_T_ra_test.to_numpy().flatten() # Convert DataFrame to Numpy

## T_ma Faults
Faults_T_ma_test = df_test.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ma_test = Faults_T_ma_test.to_numpy().flatten() # Convert DataFrame to Numpy

## T_ahc Faults
Faults_T_ahc_test = df_test.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_ahc_test = Faults_T_ahc_test.to_numpy().flatten() # Convert DataFrame to Numpy

## T_sa Faults
Faults_T_sa_test = df_test.sample(frac = 1) # Shuffle the data sets from row by row
Faults_T_sa_test = Faults_T_sa_test.to_numpy().flatten() # Convert DataFrame to Numpy

#Faults_test = dataset.iloc[:180,13].values

T_oa_test = x_test[:,0]
T_ra_test = x_test[:,1]
T_ma_test = x_test[:,5]
T_ahc_test = x_test[:,7]
T_sa_test = x_test[:,8]

Q_test = x_test[:,6]
V_ma_test = x_test[:,2]
V_oa_test = x_test[:,3]
V_ra_test = x_test[:,4]


## T_oa
T_oa_F_test = T_oa_test+Faults_T_oa_test
T_oa_signal_test = abs(T_oa_F_test-T_oa_test)


## T_ra
T_ra_F_test = T_ra_test+Faults_T_ra_test
T_ra_signal_test = abs(T_ra_F_test-T_ra_test)


## T_ma

T_ma_F_test = ((0.2114)*T_oa_F_test + (0.7428)*T_ra_F_test)+Faults_T_ma_test
T_ma_signal_test = abs(T_ma_F_test-T_ma_test)

## T_ahc

Q_F_test = Q_test

V_ma_F_test = V_ma_test

T_ahc_F_test = (((0.1871*Q_F_test)/(0.2316*V_ma_F_test))+(1*T_ma_F_test))+Faults_T_ahc_test
T_ahc_signal_test = abs(T_ahc_F_test-T_ahc_test)


## T_ra
T_sa_F_test = (T_ahc_F_test+1.8) + Faults_T_sa_test
T_sa_signal_test = abs(T_sa_F_test-T_sa_test)
```

```
## Combination of Temperature with fault
# Faulty Data set
X_Fault_test =(np.concatenate((T_oa_F_test.reshape(len(T_oa_F_test),1),
T_ra_F_test.reshape(len(T_ra_F_test),1),V_ma_test.reshape(len(V_ma_test),1),V_oa_test.reshape(len(V_oa_test),1),V_ra_test.reshape(len(V_
ra_test),1), T_ma_F_test.reshape(len(T_ma_F_test),1),Q_test.reshape(len(Q_test),1), T_ahc_F_test.reshape(len(T_ahc_F_test),1),
T_sa_F_test.reshape(len(T_sa_F_test),1)),1))

Input_test = X_Fault_test
X_test = Input_test

Output_test = (np.concatenate((T_oa_signal_test.reshape(len(T_oa_signal_test),1), T_ra_signal_test.reshape(len(T_ra_signal_test),1),
T_ma_signal_test.reshape(len(T_ma_signal_test),1), T_ahc_signal_test.reshape(len(T_ahc_signal_test),1),
T_sa_signal_test.reshape(len(T_sa_signal_test),1)),1))
Output_test = (Output_test > 0.5)*1
y_test = Output_test



## ### Time Lags
# Training

Results_final = pd.DataFrame(columns=['Accuracy','Precision','Sensitivity'])


for N in range (0,11): # Number of Time-Lag Hours t-n
    L = 1440-N-1 # starting the row as the input

    I_T_oa = []
    I_T_ra = []
    I_T_ma = []
    I_T_ahc = []
    I_T_sa = []
    I_V_oa = []
    I_V_ra = []
    I_V_ma = []
    I_Q = []


    for i in range (L, 1440):
        I_T_oa.append(Input[i-L:i+1, 0])
        I_T_ra.append(Input[i-L:i+1, 1])
        I_V_ma.append(Input[i-L:i+1, 2])
        I_V_oa.append(Input[i-L:i+1, 3])
        I_V_ra.append(Input[i-L:i+1, 4])
        I_T_ma.append(Input[i-L:i+1, 5])
        I_Q.append(Input[i-L:i+1, 6])
        I_T_ahc.append(Input[i-L:i+1, 7])
        I_T_sa.append(Input[i-L:i+1, 8])

    I_T_oa = np.array(I_T_oa)
    I_T_ra = np.array(I_T_ra)
    I_V_ma = np.array(I_V_ma)
    I_V_oa = np.array(I_V_oa)
    I_V_ra = np.array(I_V_ra)
    I_T_ma = np.array(I_T_ma)
    I_Q = np.array(I_Q)
    I_T_ahc = np.array(I_T_ahc)
    I_T_sa = np.array(I_T_sa)

    O = []
    for i in range (N, 1440):
        O.append(Output[i,0:5])
    O = np.array(O)

    Inputs_lag = np.concatenate([I_T_oa,I_T_ra,I_V_ma,I_V_oa,I_V_ra,I_T_ma,I_Q,I_T_ahc,I_T_sa]).T
    Output_lag = O




    # Testing

    I_T_oa_test = []
    I_T_ra_test = []
    I_T_ma_test = []
    I_T_ahc_test = []
    I_T_sa_test = []
    I_V_oa_test = []
    I_V_ra_test = []
    I_V_ma_test = []
    I_Q_test = []


    l = 360-N-1 # starting the row as the input

```

```
    l = 360-N-1 # starting the row as the input

    for i in range (l, 360):
        I_T_oa_test.append(Input_test[i-l:i+1, 0])
        I_T_ra_test.append(Input_test[i-l:i+1, 1])
        I_V_ma_test.append(Input_test[i-l:i+1, 2])
        I_V_oa_test.append(Input_test[i-l:i+1, 3])
        I_V_ra_test.append(Input_test[i-l:i+1, 4])
        I_T_ma_test.append(Input_test[i-l:i+1, 5])
        I_Q_test.append(Input_test[i-l:i+1, 6])
        I_T_ahc_test.append(Input_test[i-l:i+1, 7])
        I_T_sa_test.append(Input_test[i-l:i+1, 8])

    I_T_oa_test = np.array(I_T_oa_test)
    I_T_ra_test = np.array(I_T_ra_test)
    I_V_ma_test = np.array(I_V_ma_test)
    I_V_oa_test = np.array(I_V_oa_test)
    I_V_ra_test = np.array(I_V_ra_test)
    I_T_ma_test = np.array(I_T_ma_test)
    I_Q_test = np.array(I_Q_test)
    I_T_ahc_test = np.array(I_T_ahc_test)
    I_T_sa_test = np.array(I_T_sa_test)

    O_test = []
    for r in range (N, 360):
        O_test.append(Output_test[r,0:5])
    O_test = np.array(O_test)

    Inputs_lag_test =
np.concatenate([I_T_oa_test,I_T_ra_test,I_V_ma_test,I_V_oa_test,I_V_ra_test,I_T_ma_test,I_Q_test,I_T_ahc_test,I_T_sa_test]).T
    Output_lag_test = O_test


    ## Training and Testing Data sets

    X_train = Inputs_lag
    y_train = Output_lag

    X_test = Inputs_lag_test
    y_test = Output_lag_test


    # Decision Tree

    # Feature Scaling
    from sklearn.preprocessing import StandardScaler
    sc = StandardScaler()
    X_train = sc.fit_transform(X_train)
    X_test = sc.transform(X_test)

    # Training the Naive Bayes model on the Training set
    from sklearn.tree import DecisionTreeClassifier
    classifier = DecisionTreeClassifier(criterion='entropy', random_state = 0)
    classifier.fit(X_train, y_train)

    # Predicting the Test set results
    y_pred = classifier.predict(X_test)


    y_pred = (y_pred > 0.5)
    y_pred = y_pred*1
    y_pred_T_oa = y_pred[:,0]
    y_pred_T_ra = y_pred[:,1]
    y_pred_T_ma = y_pred[:,2]
    y_pred_T_ahc = y_pred[:,3]
    y_pred_T_sa = y_pred[:,4]

    y_test = (y_test > 0.5)
    y_test = y_test*1
    y_test_T_oa = y_test[:,0]
    y_test_T_ra = y_test[:,1]
    y_test_T_ma = y_test[:,2]
    y_test_T_ahc = y_test[:,3]
    y_test_T_sa = y_test[:,4]


    y_compare =(np.concatenate((y_pred_T_oa.reshape(len(y_pred_T_oa),1),
y_pred_T_ra.reshape(len(y_pred_T_ra),1),y_pred_T_ma.reshape(len(y_pred_T_ma),1),y_pred_T_ahc.reshape(len(y_pred_T_ahc),1),y_pred_T_sa.re
shape(len(y_pred_T_sa),1)),1))


    y_pred_conf_1 =(np.concatenate((y_pred_T_oa.reshape(len(y_pred_T_oa),1),
y_pred_T_ra.reshape(len(y_pred_T_ra),1),y_pred_T_ma.reshape(len(y_pred_T_ma),1),y_pred_T_ahc.reshape(len(y_pred_T_ahc),1),y_pred_T_sa.re
shape(len(y_pred_T_sa),1))))
    y_test_conf_1 =(np.concatenate((y_test_T_oa.reshape(len(y_test_T_oa),1),
y_test_T_ra.reshape(len(y_test_T_ra),1),y_test_T_ma.reshape(len(y_test_T_ma),1),y_test_T_ahc.reshape(len(y_test_T_ahc),1),y_test_T_sa.re
shape(len(y_test_T_sa),1))))


    y_pred_conf =(np.concatenate((y_pred_conf_1.reshape(len(y_pred_conf_1),1), y_test_conf_1.reshape(len(y_test_conf_1),1)),1))

```

```
342
343     # Making the Confusion Matrix
344     from sklearn.metrics import confusion_matrix, accuracy_score
345     from sklearn.metrics import precision_score
346     from sklearn.metrics import recall_score
347
348     cm_3 = confusion_matrix(y_test_conf_1, y_pred_conf_1)
349
350     Accuracy = accuracy_score(y_test_conf_1, y_pred_conf_1) *100
351     Precision = precision_score(y_test_conf_1, y_pred_conf_1)*100
352     Sensitivity = recall_score(y_test_conf_1, y_pred_conf_1)*100
353
354     Results_final.loc[N, ['Accuracy']] = Accuracy
355     Results_final.loc[N, ['Precision']] = Precision
356     Results_final.loc[N, ['Sensitivity']] = Sensitivity
357
```