MA 480

Master’s Thesis

Development of a Data-driven Fault Detection and Diagnosis Algorithm for a Residential Propane-based Heat Pump

Methodische Entwicklung einer datengetriebenen Fehlererkennung und -diagnose am Beispiel einer Propan-Warmepumpe im Wohnbereich

Aachen, May 2023

Francisco Santoro Delgado
Matriculation number: 449895

Supervisors:
Sebastian Borges, M.Sc.
Hannah Romberg, M.Sc.
Univ.-Prof. Dr.-Ing. Dirk Müller

The present work was submitted to the:
E.ON Energy Research Center | ERC
Institute for Energy Efficient Buildings and Indoor Climate | EBC
Mathieustrasse 10, 52074 Aachen, Germany

Abstract

In the context of the energy trilemma (equity, sustainability, and security) the reliance on natural gas is experiencing important drawbacks. Climate change pledges call for electrification, while the second largest consumer of this fossil fuel is the residential sector. An alternative to traditional gas boilers are heat pumps. However, the operation of heat pumps is threatened by faults, which can hinder performance, diminish efficiency, increase operation costs, and reduce equipment's lifespan. Still, literature shows small deployment of FDD methods in the residential sector, mostly due to the high costs of a comprehensive sensor array, absence of data, wide range of models, and variability of installations. Within FDD classification, data-driven techniques, based on machine learning algorithms, stand out for their versatility and ease of deployment. These methods do not require complex models, can handle noisy data, and a reduced set of inputs. This thesis compares three supervised machine learning algorithms, trained for the fault detection and diagnosis (FDD) of a residential-sized air-to-water heat pump operating in heating mode. The algorithms considered in this thesis are artificial neural networks, decision trees and ensembles, and supported vector machines. To address the obstacle of the sensor array, a set of inexpensive temperature measurements is proposed. The first objective of this work is to explore the performance attainable by a model with only these inputs. Then, considering that real household data is scarce and comprehensive experimental trials represent a major effort, a simulation model is developed. Data is generated using Modelica Language and Dymola software, taking an existing heat pump as reference. Across the spectre of common soft faults, evaporator fouling and refrigerant leakage are selected. The fault modeling strategies involve decreasing the refrigerant mass for leakage or undercharge, similarly, the fan speed and overall heat transfer coefficient are reduced to emulate fouling. The simulations only consider steady-state, however, variance is introduced through weather data in typical year simulations. Next, simulated datasets are split in training and test sets, extracting representative fractions of the winter period from the first and last trimester of the year. The aforementioned temperature measurements are interpreted as features by the machine learning algorithms. As a result, the detection of evaporator fouling was successfully achieved through classification-based and regression-based strategies, which are developed and compared. Across all stages, neural networks exhibit the best scores, with the regression algorithms outperforming the classifiers. The resulting algorithm showed in the final evaluation with a 71% correct rate, 29% false alarm, and 0% missed detection rate.

Zusammenfassung

Im Kontext des Energie-Trilemmas (Gerechtigkeit, Nachhaltigkeit und Sicherheit) hat die Abhängigkeit von Erdgas entscheidene Nachteile. Daher, und um Klimaziele erreichen zu können, wird die Elektrifizierung erdgasbasierter Prozess angestrebt. Der Wohnungssekto ist derzeit der zweitgrößte Verbraucher von Erdgas. Eine Alternative zu herkömmlichen Gasheizungen im Wohnungssektor sind Wärmepumpen. Allerdings ist der Betrieb von Wärmepumpen durch Störungen bedroht, die die Leistung beeinträchtigen, die Effizienz verringern, Betriebskosten erhöhen und die Lebensdauer der Geräte verkürzen können. Dennoch zeigen Untersuchungen eine geringe Verbreitung von Methoden zur Störungserkennung und -diagnose (FDD) im Wohnungssektor, hauptsächlich aufgrund hoher Kosten für Sensorik, Datenmangel, einer Vielzahl von Modellen und der Varianz der Installationen. Im Bereich der FDD-Klassifikation zeichnen sich datengetriebene Methoden auf Basis von maschinellen Lernalgorithmen durch ihre Vielseitigkeit und einfache Implementierung aus. Diese Methoden erfordern keine komplexen Modelle, können mit störungsbehafteten Daten umgehen und benötigen nur einen reduzierten Satz von Eingangsgrößen. Diese Arbeit vergleicht drei überwachte maschinelle Lernalgorithmen, die für die Fehlererkennung und -diagnose einer luftgekühlten Wärmepumpe in Heizbetrieb ausgelegt sind. Die betrachteten Algorithmen sind künstliche neuronale Netze, Entscheidungsbäume und Ensemble-Methoden sowie unterstützte Vektor-Maschinen. Um das Hindernis der umfassenden Sensorik zu umgehen, wird ein Satz kostengünstiger Temperaturmessungen vorgeschlagen. Das erste Ziel dieser Arbeit besteht darin, die erreichbare Leistung eines Modells mit nur diesen Eingangsgrößen zu erforschen. Da jedoch reale Haushaltsdaten knapp sind und umfassende experimentelle Versuche einen erheblichen Aufwand darstellen, wird ein Simulationssmodell entwickelt. Die Daten werden unter Verwendung der Modelica-Sprache und der Dymola-Software generiert und eine vorhandene Wärmepumpe als Referenz herangezogen. Unter Berücksichtigung einer Vielzahl von gängigen Fehlern werden Verdampferverschmutzung und Kältemittelleckage ausgewählt. Die Strategien zur Fehlermodellierung umfassen die Verringerung der Kältemittelmasse für Leckagen oder Unterkühlung sowie die Reduzierung der Lüftergeschwindigkeit und des gesamten Wärmeübergangskoeffizienten zur Emulation der Verschmutzung. Die Simulationen berücksichtigen nur den stationären Zustand, jedoch wird die Varianz durch Wetterdaten in Simulationen eines typischen Jahres eingeführt. Anschließend werden die simulierten Datensätze in Trainings- und Testsets aufgeteilt, wobei repräsentative Anteile des Winterzeit raums aus dem ersten und letzten Trimester des Jahres entnommen werden. Die Temperaturmessungen gehen als Eingangsgrößen in die maschinellen Lernalgorithm ein. Als Ergebnis konnte die Erkennung von Verdampferverschmutzung erfolgreich durch Klassifikations- und Regressionsstrategien erreicht werden, die entwickelt und verglichen wurden. In allen Phasen erzielten neuronale Netzwerke die besten Ergebnisse, wobei die Regressionsalgorithmen die Klassifikatoren übertrafen. Der resultierende Algorithmus zeigte in der abschließenden Bewertung eine korrekte Rate von 71%, eine Fehlalarmrate von 29% und eine Rate verpasster Erkennungen von 0%.

Table of Contents

Nomenclature

Symbols and Units

Symbol	Meaning	Unit
R^2	Coefficient of correlation	-
COP	Coefficient of performance	-
h	Enthalpy	$\mathrm{J\,kg^{-1}}$
$\dot{Q}$	Heat flow	W
I	Integral parameter	-
$\dot{m}$	Mass flow	$\mathrm{kg\,s^{-1}}$
P	Proportional parameter	-
$\dot{W}$	Power	W
$RMSE$	Root mean squared error	-
v	Specific volume	$\mathrm{m^3/kg}$
S	Surface	$\mathrm{m^2}$
T	Temperature	°C
t	Time	s
$\dot{V}$	Volume flow	$\mathrm{m^3/s}$

Greek variables

Symbol	Meaning	Unit
α	Overall heat transfer coefficient	$\mathrm{W/(m^2 \cdot K)}$
η	Efficiency	—

Index and Abbreviations

Symbol	**Meaning**
a	Air
A/A	Air conditioning
ANN	Artificial neural network
AFDD	Automated fault detection and diagnosis
ADDMo	Automated data driven modeling tool
Bb	Black box
cond	Condenser
comp	Compressor
CM	Confusion matrix
Ddb	Data driven based
DT	Decision tree
dis	Discharge
ev	Evaporator
EEV	Electronic expansion valve
evap	Evaporating
EF	Evaporator fouling
fan	Fan
FDD	Fault detection and diagnosis
FN	False negative
FP	False positive
FXO	Fixed orifice
Gb	Grey box
h	heating
HP	Heat pump
HPSKL	Hyperopt - Scikit-learn
HVAC	Heating, ventilation, and air conditioning
HX	Heat exchanger
i	inlet
int	Integral
is	Isoentropic
mec	Mechanical
ML	Machine Learning
MLA	Machine Learning Algorithm

Continued on the next page

Index and Abbreviations

Symbol	**Meaning**
o	outlet
op	opening
Phb	Process history based
PhM	Physical model based
QlM	Qualitative model based
QtM	Quantitative model based
Rb	Rule based
ref	refrigerant
RF	Random Forest
RL	Refrigerant leakage
SC	Subcooling
SH	Superheating
SKL	Scikit-learn
suc	suction
Surf	Surface
SVM	Support vector machine
TN	True negative
TP	True positive
vol	volumetric
VS	Virtual sensor
w	water
XBG	Extreme boosting gradient

List of Figures

List of Tables

1 Introduction

In 2020, around 13000 pentajoules (PJ) of raw energy were consumed in heating applications. This energy was generated, mainly, by the combustion of fossil fuels, specifically, coal and natural gas. When focusing only on the natural gas consumption for heating, the residential and commercial sectors are second and third largest consumers [1]. Furthermore, in 2021 almost half of the energy demand for buildings was used for heat spaces and water [2].

Currently, Europe faces several obstacles to secure heating energy. The principal reason is that heat is generated through the use of materials that are experimenting shortages in supply, exhibiting high costs, and have important climate change repercussions. Amidst this complex scenario, more attention is being drawn to the electrification of heating, which also represents an opportunity for the transition toward net-zero goals. This sets up the stage for the adoption of new technologies, where, right in the conjunction between heating technologies and renewable energies, heat pumps enter the scene. A device capable of providing heat with electricity as an energy source, well-suited for most of the residential and commercial sector needs. Thus, heat pumps are considered by International Energy Agency (IEA) a central pillar in improving the efficiency of energy consumption and reducing emissions. [2]

In Germany, the aggregate of heating and cooling from all sectors accounts for up to 50% of the final energy consumption. Moreover, the residential sector is third in consumption of energy and heat. The adoption from this sector of heat pumps will represent many benefits: to the environment (given the CO_2 emissions reduction), to the grid (reduction of demand due to increased efficiency), and the final user (because of energy security and affordability) [3]. This technology experienced during 2022 a record year in sales, growing by nearly 40% in Europe, with nearly 3 million units sold. The sales are particularly good for air-to-water heat pumps models, which jumped by almost 50% in Europe. One of the reasons is that these devices, in particular, are compatible with typical radiators and underfloor heating systems [4]. The previous trends are enhanced by the REPowerEU policies, their purpose is to accelerate and enlarge the goals for 2030. Amid their objectives, lays doubling the deployment of heat pumps [5]. This would imply that sales in heat pumps rise up to 7 million for 2030. IEA estimates that, in order to meet with the climate pledges worldwide, heat pumps will have to meet nearly 20% of global heating needs in buildings by 2030 [4].

Nevertheless, heat pumps have detractors, which claim slow and noisy operations, high cost, among other reasons. Although the veracity of these claims is arguable, there is evidence installed heat pumps performing with degraded efficiencies. Field surveys show significant losses in the efficiency performance of heat pumps installed in buildings. Research exhibits that 20 to 50% of heat pumps could be operating at 70–80% efficiency or lower than their design efficiency, with faulty operation contributing an additional 40% of energy consumption [6]. The reduction of efficiency derives in increased operation costs due to increased energy consumption for the same application. In addition, faults can imply the need for extra maintenance, which is another associated cost. If faults are not corrected early, the lifespan of the devices is threatened [7]. Therefore, there is a need to ensure efficient and reliable operation through time.

To address the faults in this type of devices, the field of Fault Detection and Diagnosis (FDD) has been applied to heating, ventilation and air conditioning (HVAC) for over 20 years, developing techniques for early detection and damage prevention. In this day and age, with Internet-of-Things, cloud services, affordable sensors, novel automated processes can be implemented to step further in the reliability of heat pumps. This technological advances are the base for a group of methods referred as *Data driven-methods* and are amidst the most popular for FDD research [8]. Current technology developments allow for broadening the implementations of FDD. However, authors such as Rogers et al. [9] state out that, in the residential sector, FDD has not being fully implemented. This sheds light on the importance to diversify the implementation of FDD and research each sectors needs.

In addition, Data-driven methods harness another important trend, such as machine learning algorithms. Leveraging from data that is usually being collected for historical trends or monitoring, algorithms can be trained to discern between normal operation and abnormal patterns. Considering the current availability of techniques for data collection in households and the possibility of implementing Data-driven FDD to analyze this information, the development of FDD residential-sized heat pumps seems attainable. This could translate into early detection of faults and clearer service by technicians, considering all the information derived from the implementation of FDD.

The combination of investment (motivated by heat pump's current momentum), technology deployment (exposed in the recent popularity of heat pumps), and need for reliability, brings the opportunity for the design of new heat pumps. For these heat pumps, FDD considerations must be brought to the design phase, where an array of sensors is placed for monitoring and data collection. This instrumentation shall not increase largely the costs of the equipment, it should be the minimum to aid the detection and diagnosis, preserving the life span of the device.

Because of all the exposed reasons, this thesis aims to compare the performance of three machine algorithms, trained for fault detection and diagnosis, within the possibilities of a residential-sized propane heat pump. Faults will be imposed at diverse intensities in different operating conditions of common domestic heating through simulated data. The features are constrained to those inexpensive and coherent with household equipment. The alternatives of machine learning methods are: artificial neural networks, decision trees, and supported vector machines.

Section 2.1 introduces the basic concepts of heat pumps, next, Section 2.2 covers the state-of-the-art of FDD, strategies, and tools, Section 2.3 provides an overview of Machine Learning concepts and Algorithms. Regarding the methodological framework, Section 3.1 describes the use case, reference heat pump, and assumptions, Section 3.2 goes over the generation of the datasets to trains the algorithms and fault modeling strategies, Section 3.3 explains the testing and selecting of the estimators. At last, Section 4.1 reviews the simulation outputs, while 4.2 evaluates the algorithms' performance.

2 Theoretical Background

This chapter begins with an introduction to heat pumps, detailing classification and components, along with a brief summary of the vapor-compression cycle in Section 2.1. Section 2.2 presents the Fault Detection and Diagnosis research field with its categorizations, tools, and challenges. At last, Section 2.3 provides a comprehensive introduction to machine learning algorithms used for fault detection.

2.1 Basics of Heat Pump Technology

Heat pumps are conduits to transfer low-grade energy from a space (heat source) where it is available to a place where it is useful (heat sink). In general terms, heat pumps harness the latent heat of a work fluid to increase the temperature of a secondary fluid in direct contact with the heating demands; commonly, these devices are implemented for heating purposes, increasing the temperature of fluids such as air or water. Still, cycle reversal is also possible to provide cold conditions. Frequent heat sources are found in nature (e.g., ambient air, ground, lake or seawater) or in residuals of different kinds (e.g., exhaust air, sewage water). The most beneficial heat source is one with a high and stable temperature level. The type of heat source will have a strong influence on the heating capacity characteristics of the heat pump. Due to its availability, ambient air is a heat source of great potential for practical applications. However, ambient air has typically the most pronounced variations among heat sources, which leads to heating capacity (i.e., heating power) variations; meaning very low capacity on the coldest day given the evident small amount of available energy. [10]

Heat pumps can be classified based on the type of natural source/sink they use [11]:

- Air-to-air.
- Air-to-water.
- Water-to-water.
- Ground-to-water.
- Ground-to-air.

Most heat pumps operate under the vapor-compression cycle, where the working fluid is called a *refrigerant*. The refrigerant circulates in a closed loop and is subject to four processes:

evaporation, compression, condensation, and throttling. The basic configuration of these systems include an evaporator, a compressor, a condenser, and an expansion valve. [10, 12]

In vapor-compression cycles, the heating capacity is strongly affected by the temperatures in the evaporator and the condenser. From the second law of thermodynamics, it is known that operating energy must be supplied in order to accomplish this process. The operating energy in relation to the heat output is a strong function of the temperature levels of the heat source and the heat sink. [10]

The elemental principle of the vapor compression cycle is that a compressor, the work provider component, assures appropriate pressures at the two temperature levels. At the lower temperature side, a low pressure is maintained, allowing the liquid refrigerant to be vaporized or evaporated. At the higher temperature side, a high pressure is maintained, forcing the vapor to be liquefied, or condensed. [12]

During vaporization at the lower temperature, the refrigerant absorbs heat. In this process, the temperature of the refrigerant remains essentially constant, and the temperature is called the *evaporating temperature* (T_{evap}). During the stage at the higher temperature side, the refrigerant rejects heat. Here, the refrigerant vapor is brought back to liquid state through condensation, and the corresponding temperature is called the *condensing temperature* (T_{cond}). Pure fluids evaporate and condense at a saturation temperature, this temperature is characteristic of each fluid and depends on the pressure. [12]

2.1.1 Evaporators

The evaporator is a container wherein the refrigerant vaporizes at a low temperature. Under steady-state conditions, the evaporator is supplied with a continuous flow of refrigerant, which is vaporized successively by the heat transferred from the heat source. The pressure drop in the flow direction of the refrigerant is regularly small. At the inlet to the evaporator, the refrigerant consists of a mixture of saturated liquid and saturated vapor. To protect the compressor, it is preferred that the vapor continues to absorb heat from the refrigerated space and become slightly superheated before it leaves the evaporator. [12]

When the heat source is air, a typical heat exchanger configuration consists of a finned tube bundle with rectangular box headers on both ends of the tubes; this configuration is known as *fin-and-tube*. Refrigerant flows in the tubes, often made of copper, and air is blown by fans [11]. For temperatures below 0 °C on the evaporator surface, there will be frost deposits on the evaporator. This corresponds to ambient temperatures usually below 5°C. To maintain the evaporator's performance, it is necessary to arrange defrosting at certain intervals [10].

2.1.2 Compressors

The compressor maintains a pressure difference allowing the refrigerant to vaporize in the evaporator and to condense in the condenser. The refrigerant enters the compressor as a slightly superheated vapor. The work input to the refrigerant in the compressor makes it possible to lift the heat absorbed in the evaporator to a higher temperature level. This process is often considered. An electric motor commonly supplies the work. In the theoretical cycle, the compression process is assumed isentropic. In reality, multiple factors intervene. As a consequence, a ratio between the theoretical and real process is defined, called *isentropic efficiency* (Equation 2.1), which often is in the range [0,6; 0,8] [12]. Moreover, further efficiencies are established, the main ones are the volumetric efficiency (Equation 2.2) to describe recirculations within the compressor's stroke, and the mechanical efficiency (Equation 2.3), to account for the losses due to mechanical friction, slips, or magnetic losses in the coil, etc, which diminishes the power input from the power consumed. Another parameter is used to assess the units of energy supplied per energy consumed, known as *coefficient of performance* or *COP* (Equation 2.4.

$$\eta_{\mathrm{is}} = \frac{W_{\mathrm{is}}}{W} \tag{2.1}$$

$$\eta_{\mathrm{vol}} = \frac{V_{\mathrm{ref,o}}}{V_{\mathrm{ref,i}}} \tag{2.2}$$

$$\eta_{\mathrm{mec}} = \frac{W_{\mathrm{o}}}{W_{\mathrm{i}}} \tag{2.3}$$

$$COP = \frac{Q_{\mathrm{h}}}{W} \tag{2.4}$$

Compressors must be enabled to work off their nominal load. The on-off control is the simplest way to reach this goal, but it is the most energy consuming. In this way, a reference signal is chosen to control its operation. Generally, a secondary fluid supply temperature is chosen as a set-point, The difference between the measurement and the set-point activates or deactivates the compressor according to upper and lower limits. A method to obtain this continuous modulation consists in changing the rotation speed of the driving motor. Heat pumps often use an inverter to control the electric motor. Such a device changes the feeding frequency from lower values than the nominal or to higher frequencies. [11]

2.1.3 Condensers

The condenser is a recipient wherein the refrigerant liquefies, in specific conditions of constant pressure and temperature. The discharge line delivers the high pressure/high temperature vapor from the compressor to the condenser. Due to the high pressure level, the vapor is brought to condensation [12]. As a safeguard to the expansion valve, the liquid's temperature is further decreased (subcooled). This is necessary to avoid vapor bubbles formation that would damage the following component [11].

If refrigerant exchanges heat with water (as in air-to-water heat pumps), plate and frame heat exchangers are used, also known as *plate heat exchangers*. They show a high heat transfer efficiency while being compact devices. Plates are compressed together in a rigid frame and form a set of parallel channels with alternating hot and cold fluids. They have corrugated metal plates to transfer heat between the fluids. [12]

2.1.4 Expansion Valves

The expansion device (also called refrigerant flow control element) has a twofold purpose: accomplish the throttling process and maintaining the pressure difference between the condenser and the evaporator. The expansion valves adjust the refrigerant flow rate to the evaporator, keeping the evaporator filled with refrigerant liquid, avoiding the liquid to be carried over to the compressor. In the throttling step the valve discharge area is reduced, hence the pressure drops and the liquid begins to vaporize at a constant enthalpy [12]. Controllable expansion valves can be used to establish a superheating control. To keep the vapor superheat at the compressor inlet to a fixed set-point, the valve opening is adjusted correspondingly. The usual valves are capillary tubes, fixed-orifice (FXO), thermostatic expansion (TXO), and electronic (EEV) [11].

2.2 Fault Detection and Diagnosis

Fault Detection and Diagnosis (FDD) is a research field concerned with automating the processes of unveiling defects and identifying their causes. The fundamental objective of an FDD system is early detection, thus enabling correction of the irregularities before additional damage to the system or loss of service takes place. This is achieved by: continuously monitoring operations, recognizing abnormal conditions, assessing the faults associated with those conditions, evaluating the significance of the detected faults, and deciding suitable responses. [13]

Faults are conditions within the system that can lead to failure or degradation in the performance. Therefore, faults are defined as any deviation from an acceptable range of an observed variable or estimated parameter related with the evaluated process. Faults can be classified as *hard faults* or abrupt faults, and *soft faults* or gradual faults. Hard faults lead to complete system failures, hence being easier to detect than soft faults [14]. Rogers et al. [9]

differentiates both faults through the way in which they can be detected: hard faults may be detected by analyzing the provided indoor conditions, while insight to the system operation is necessary for detecting soft faults. Soft faults can remain undetected for longer periods of time because their severity is progressive [14].

According to Kim and Katipamula [15], over the past three decades, close to 200 automated fault detection and diagnosis (AFDD) studies related to building heating, ventilation and air-conditioning systems (HVAC), have been published. The third most popular FDD research topic in HVAC is the electrically driven vapor-compression air-conditioner or heat pump systems (31 studies of 197 until 2016). Matetic et al. [16] confirm that heat pumps are on the firsts trending research subjects in the FDD field.

Air-cooled vapor compression air-conditioning equipment are excellent candidates for FDD methods, considering that the deployment of this technology is abundant, also these systems typically receive less intensive maintenance than larger systems. Another notable attribute to applying FDD in HVAC is that most devices are manufactured at relatively low-cost, specially at residential and small to medium commercial levels. Consequently, they tend to have a high incidence of faults [17]. Finally, cooling and heating applications account for large portions of final consumption energy, more efficient systems decrease operational costs and carbon footprint [18].

FDD tools are applied to air-conditioning systems primarily to avoid degradation of capacity and loss of efficiency, thus having a distinct accent on soft faults. These faults could pass unnoticed by equipment operators, even between applications of routine maintenance. Moreover, they may remain unnoticed by maintenance technicians during more intensive maintenance tasks. [17]

Early detection has many benefits: fundamentally, it prevents damages to the system that would shorten the equipment's life span. On a residential scale, detecting faults before the home occupant notices the effects allows him to address those issues during shoulder seasons. This reduces the strain on air conditioning during peak season, furthermore, it would reduce repair costs for the homeowners as fault will not have worsened [9]. Additionally, energy consumption and associated costs will not increase. However, major emphasis is placed on the role of faults as threats to system's life under the light of the studies of Hu et al. [19]. The authors stated that: "the impact of faults on equipment life represent a larger contribution to operating cost than the impact of increased energy usage". Rogers et al. [9] listed the main benefits of FDD implementation:

- Reduced maintenance costs.
- Reduced electricity costs.
- Improved commissioning.

- Reduced peak demand.
- Reduced carbon emissions and electricity costs.

2.2.1 FDD categories

The most adopted classification for FDD in HVAC was established by Katipamula and Brambley in 2005 [13], categorizing AFDD methods into three main categories: *qualitative model-based* (QlM), *quantitative model-based methods* (QtM), and *process history-based* (Phb) [13]. In a later investigation, Kim and Katipamula (2017) [15] stated that the previous taxonomy

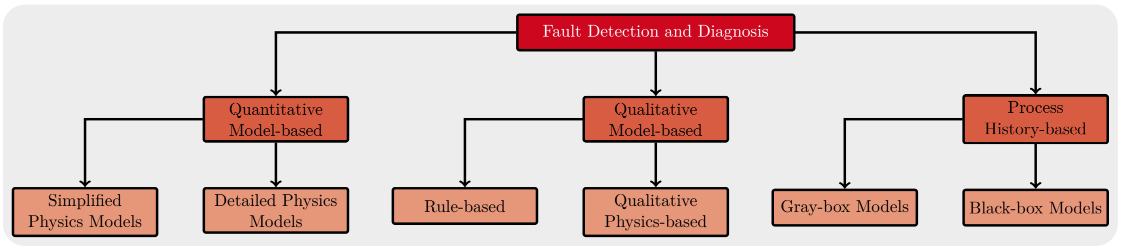


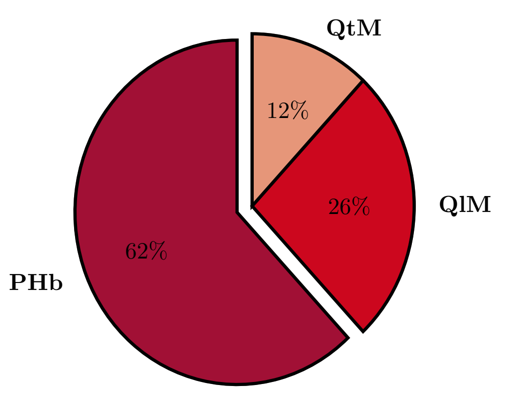
Figure 2.1: FDD Classification according to Katipamula and Brambley [13].

was still applicable. The main classes of AFDD can be further sub-classified. Out of the 197 articles reviewed by the latter authors, 74% of the studies on AFDD utilized black box (55%). A summary of their research is provided by Figure 2.2. The upcoming subsections explain the main FDD classes in detail.

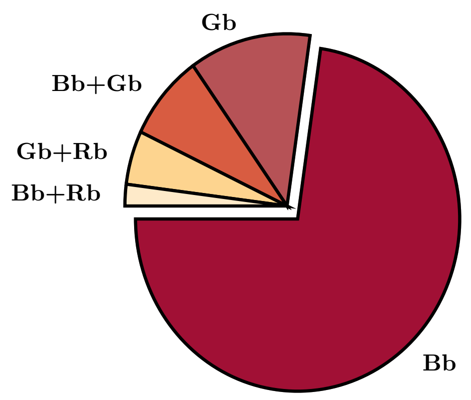
2.2.1.1 Qualitative model-based methods

Qualitative model-based methods (QlM) rely on deductive knowledge to draw conclusions about the state of a system. They can be subdivided in: rule-based and qualitative physics-based AFDD techniques. The most commonly used qualitative model-based technique is the rule-based technique (Rb), which employs a large set of *if-then-else* rules and an internal inference logic to identify the process condition from a previously defined set of potential states. The rule-based method relies on expert analysis of specific building systems and the setting of thresholds or alarms, which are derived from analysis of the historical sensor data [15]. The qualitative physics-based models contain equations derived from qualitative descriptions of relationships among the process variables or knowledge about the fundamental behavior of the system. Fault detection is performed by comparing the predicted qualitative behavior of a system based on a model with the actual observation [13].

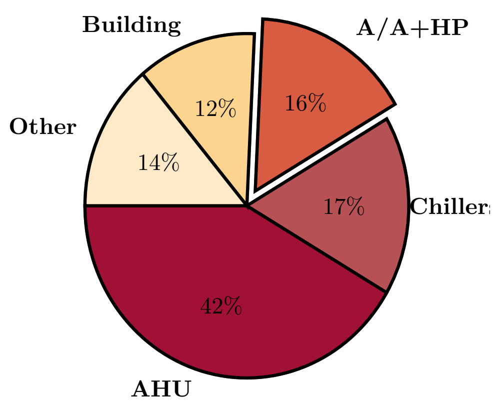
One of the biggest strengths of qualitative models is that they are simple to develop and apply. These models are ideal for data-rich environments and noncritical processes because



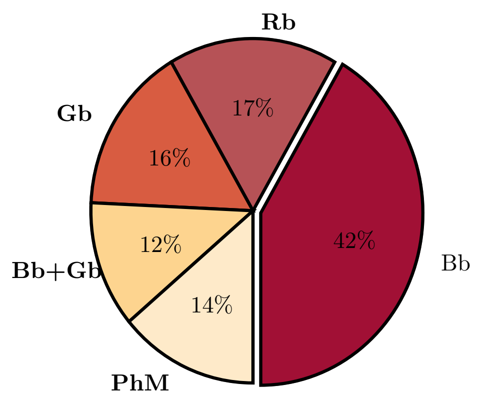
(a) FDD methods classification of publications



(b) FDD PHb methods distribution of publications



(c) HVAC systems focus of FDD publications



(d) Distribution of FDD methods in Heat pumps publications

Figure 2.2: FDD methods trends [15].

the models can assess a process without precise numerical inputs and exact expressions that govern the process. Moreover, the logic of the model is transparent and easy to interpret [15].

2.2.1.2 Quantitative model-based methods

Quantitative model-based (QtM) methods use an explicit mathematical model of the monitored plant. The mathematical equations to represent each component of the system are developed and solved to simulate the steady and transient behavior of the system. The quantitative model-based methods need to be properly validated with experimental data for fault-free and/or "faulty" operations before any credibility can be placed on their prediction accuracy and usefulness. These models can be further sub-classified into detailed physical and simplified physical models [15], both referred as *PhM* in Figure 2.2d.

Simplified physical models are based on the very fundamental knowledge governing the behavior of the system. In contrast, detailed physical models frequently consist in sets of intricate mathematical equations based on the mass, momentum, and energy balances as well as heat and mass transfer relations [14]. The simplified physical models calculate a physical quantity using a lumped parameter approach with limited assumptions. This approach is computationally simpler because coupled space partial differential equations are transformed into ordinary differential and algebraic equations [15].

A main advantage of quantitative models is that they are based on sound physical or engineering principles, which provide the most accurate estimators. A substantial effort is required to develop such models. Often the data required for modelling is not available in the field and these models cannot be generalized easily. In addition, the method needs adequate and reliable sensors for data acquisition, which compromises cost-effectiveness. Therefore, these models are more suited for critical industrial processes than for commercial or residential systems; still, simplified quantitative models can play an important role in building AFDD application. [15]

2.2.1.3 Process history-based methods

Process history-based (Phb) models derive in complex relationships established directly from measurement data obtained over time. These AFDD methods have been the most popular because their reliance on historical information to train the models. The models are automatically formulated through learned patterns from the data. Process history-based approaches can be split into grey-box (Gb) and black-box (Bb) models [15]. The black box model relies on parameter estimation to identify faults in the system, although in many cases the physical meaning of the parameter deviation is not known. The gray box model is formulated such that the parameter estimates used for AFDD can be traced to actual physical parameters that govern the system or the component.

Process history-based models are well suited to problems for which theoretical models of behavior are poorly developed or inadequate. The models are ideal when training data is substantial and/or inexpensive to create and collect; since large amount of data is required to make accurate conclusions. Additionally, an important shortcoming of these models is their limited ability to extrapolate beyond the range of the training data. They are suited to the system for which they are trained and rarely applicable to other systems. [15]

Gray-box models

Gray box models (Gb) use physical knowledge or first principles to specify the mathematical form of terms in the model, and measured data are used to empirically determine the model parameters. The parameters of the gray box models are estimated using a training data set. The training data can be obtained from the equipment manufacturer, laboratory tests, or the field when the unit is operating normally. Gray box models that are based on first principles also require a thorough understanding of the system and expertise in statistics. As a result, the use of a gray box model requires a high level of user expertise both in formulating the appropriate form for the model and in estimating model parameters. Consequently, gray box models are more robust than black box models for AFDD and online control applications, and they can also provide insight into and understanding of faults for fault diagnosis. In addition, when using a gray box model fewer data is required to obtain an acceptable fit and there is better confidence that the model extrapolates well for operating conditions outside of the range used to obtain the parameters. Another advantage is that they can be used for limited extrapolation outside of the training data range [15].

Black-box models

A black box (Bb) model is formulated based on a relationship between the inputs and outputs of a process or a system, but does not consider any physical significance. A black box model consisting of behavioral models is derived from process history data. Most utilized black-box approaches are: statistical models, artificial neural networks, and pattern recognition [15].

In spite of Kim and Katimapula's [15] former validation of the FDD classification, Zhao et al. [8], proposed a modern taxonomy acknowledging more the role of artificial intelligence (AI) on FDD. According to these authors, FDD methods can be classified into two subcategories: *Data driven-based* (Ddb) and *Knowledge driven-based* (Kdb).

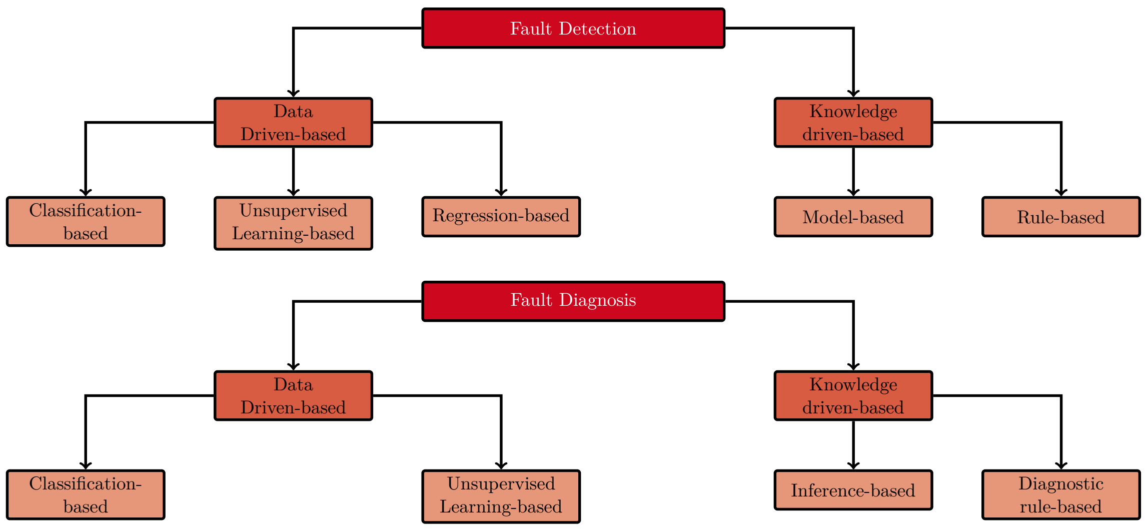
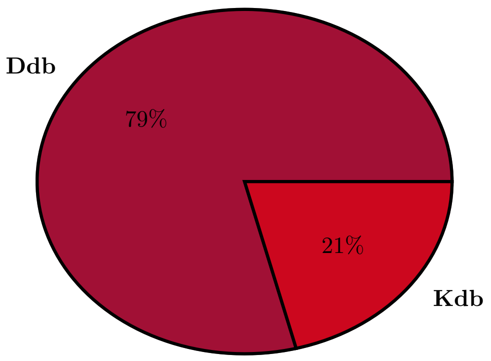
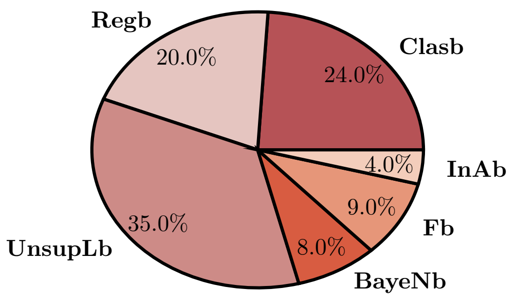


Figure 2.3: FDD Classification according to Zhao et al. [8].

Knowledge driven-based methods (Kdb) develop detection models or rules based on prior expertise that generally has clear physical meanings. They seek to simulate the diagnostic thinking of domain experts. In contrast, Data driven-based (Ddb) methods detect faults by finding changes in patterns in the measurements of selected variables. They rely mainly on pattern similarities and use machine learning algorithms to automatically extract patterns; which may not have clear physical interpretations. This classification uses a disaggregated interpretation of *black-box models*, describing in more detail the existing methods. Figure 2.4 describes the classes and subclasses distribution according to the literature review of Zhao et al. [8].



(a) FDD classification distribution.



(b) FDD subclasses distribution.

Figure 2.4: FDD methods trends according to modern classification [8].

Zhao et al. [8] categorization separates detection and diagnosis phases; also inheriting classes from the machine learning nomenclature. Figure 2.3 exhibits this classification. It is noteworthy that the authors identify AI algorithms within the inference-based techniques of the diagnostic phase.

2.2.2 Development of FDD methods

FDD protocols typically involve techniques for deducing the underlying faults from measured data, given a single fault can lead to several symptoms. Likewise, different faults can produce similar effects. This represents the fundamental complexity of the FDD inference process [8].

The general FDD process in HVAC can be summarized by [9]:

- Selecting features that use the available measurements.
- Detecting steady-state operations and filtering the data accordingly. This encompasses removing noise, errors, outliers, etc.

- Modeling the steady-state fault-free behavior of the system at the current operating conditions.
- Classifying the current operations as faulty or not.

Commonly, FDD methods for HVAC systems use measured temperature and/or pressure at various locations of the system, moreover, other implementations also utilize thermodynamic. Frequently, the detection system runs continuously, while the diagnostic system is triggered only upon the detection of a fault. In other applications, the detection and diagnostic systems run in parallel, and in particular instances, the detection and diagnostics are performed in a single step [13]. To collect the data, it is a common practice to wait until the system reaches a steady state within a defined tolerance range. Some methods use a steady-state detector, others require a system to run for a given period of time, moreover, certain protocols rely on user experience to determine whether steady operation has been achieved [18].

On each system, a subset of the tests is done with no fault present, typically at several sets of driving conditions that correspond to the driving conditions for fault tests. Common driving conditions are: ambient air temperature, humidity and (external fluid) return temperature. The reason for conducting the no-fault tests is that the fault effects on performance are a key concern for the researchers, evidently, they cannot be properly assessed without a baseline for comparison. A common and advisable practice is to develop a *normal model* from the no-fault test's measurements, using techniques such as multiple linear regression to predict capacity and COP to assess degradation [17]. Rogers et al. [9] report that third-order polynomials are better suited to be fault-free models than the neural networks due to a lack of significant non-linearity, as suggested by [20, 21]. The first reason to develop such models is that it significantly reduces bias error because it obviates the problem of trying to match the test conditions between fault and no-fault tests exactly. The second is that it reduces the random error associated with the comparison of two test results at the same conditions [17].

2.2.2.1 FDD protocols evaluation

Evaluating FDD tools performance requires knowledge of accuracy in detecting and correctly diagnosing faults across a range of fault types and fault intensities, and under a range of operating conditions. The assessment is further complicated by the many approaches taken and the functionalities of existing FDD tools. In particular, there is currently no standard method for determining how well the FDD performs in HVAC [17]. As a response, Rogers et al. [9] provide three fundamental questions to assess FDD protocols: first, how they were validated; how many sensors are required; which faults may be diagnosed. For a general case, any protocol that detects and isolates without an indication of the fault's magnitude delivers five possible outcomes, these are illustrated by Figure 2.5. There are protocols that are not

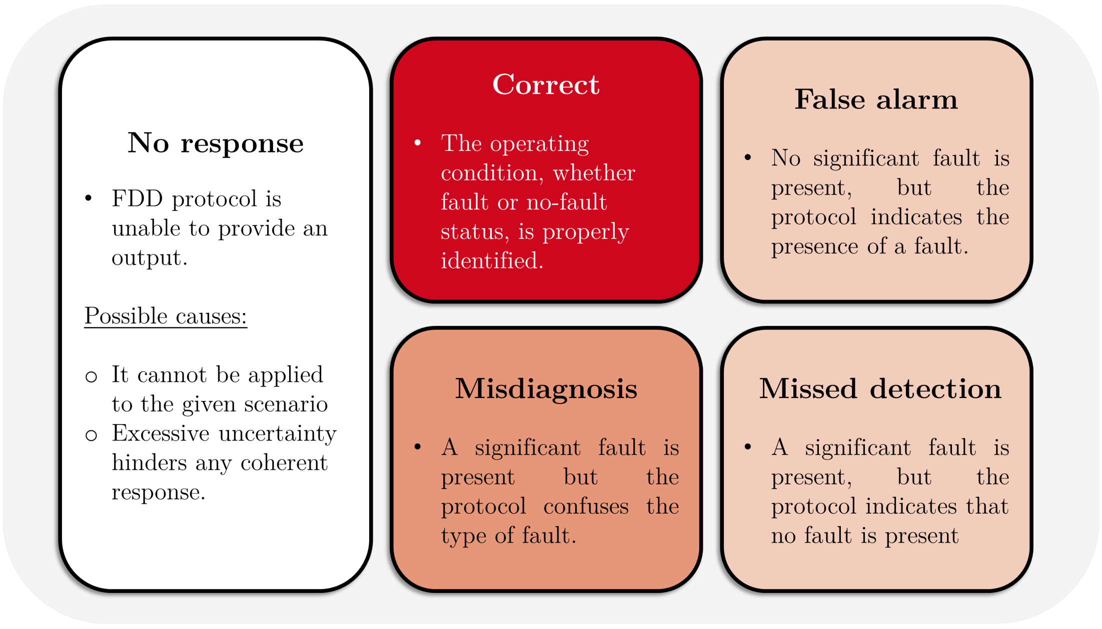

No response
• FDD protocol is unable to provide an output.
Possible causes:
o It cannot be applied to the given scenario
o Excessive uncertainty hinders any coherent response.
Correct
• The operating condition, whether fault or no-fault status, is properly identified.
False alarm
• No significant fault is present, but the protocol indicates the presence of a fault.
Misdiagnosis
• A significant fault is present but the protocol confuses the type of fault.
Missed detection
• A significant fault is present, but the protocol indicates that no fault is present

Figure 2.5: FDD protocol outputs [17].

intended to diagnose all of the faults. In these cases, the remnant faults could be missed, misdiagnosed or fail to provide any output [17]. Although that approach is a comprehensible simplification, the synergistic nature of the faults must be taken into consideration [22]. Studies such as Hu et al. [19], provide insight into the compensatory and synergistic effects of multiple simultaneous faults.

In brief, a comprehensive AFDD method should be able to diagnose all fault sources simultaneously. If only one fault is diagnosed and repaired, the system will continue to operate with an undiagnosed fault that could cause the system to fail again [22]. Apart from the desire for robust systems, this characteristic attends to a probabilistic reason. When one fault occurs, there is an increased probability that another will occur in parallel, because faults often result from low-quality installation, lack of maintenance, or harsh operating conditions [19].

Metrics

Once the results for a given set of test cases are generated, statistics are generated to provide overall performance indicators [17]. From the five previous possible outputs from an FDD system, "no response" will not be considered because the selected approaches selected ensure an output. Yuill and Braun [18] utilized the following metrics in their research, assessing an existing protocol with experimental data:

$$False\ Alarm\ Rate = \frac{misclassified\ nofault\ cases}{all\ actual\ nofault\ cases} \tag{2.5}$$

$$Missed\ Detection\ Rate = \frac{misclassified\ nofault\ cases}{all\ actual\ fault\ cases} \tag{2.6}$$

$$Misdiagnosis\ Rate = \frac{misclassified\ fault\ cases}{all\ cases\ correctly\ diagnosed\ as\ faulty} \tag{2.7}$$

The aforementioned authors [18] remark that false alarms are a specially serious error that an FDD protocol can make since it could trigger service being done on a properly working system. Likewise, they state that misdiagnoses can lead to the wrong corrective action, thus opening the possibility of a greater negative impact than taking no action. At last, it is concluded that missed detection could be considered the least serious error for a protocol to deliver since it does not result in unnecessary and potentially detrimental service.

2.2.3 Common Faults in Heat Pumps

In the study from 1998, Breuker and Braun [23], a database of 6000 commercial units from 1989 to 1995 were analyzed, from which five soft faults were selected for further research. The authors selected as the most common faults: refrigerant leakage, condenser fouling, evaporator fouling, liquid line restriction, compressor valve leakage. Those faults are still considered the most common faults in HVAC. This is demonstrated throughout more recent HVAC FDD studies, for example: Mehrabi and Yuill (2017, 2018) [24, 25], Hu et al. (2021a,2021b) [19, 7]. In addition, Bellanco et al. (2021, 2022) [6, 26], where non-condensables and refrigerant overcharge are added.

Bellanco et al. [6] and Rogers et al. [9], among other authors, generalize *fouling* to account for reversible operation modes (as in heat pumps) or faults in the fan. The proposed nomenclatures are: *outdoor unit fouling/outdoor mechanical component failure*, *indoor unit fouling/indoor mechanical component failure* or *low airflow*. Due to the fact that most soft faults are mechanical, the previous titles allow the grouping of several causes that produce the same effects on the equipment. For instance: incrustation of dirt, debris, or leaves; blockage of any kind; or malfunction of heat exchanger's fan.

Acknowledging the precedent nomenclature, it is possible to consider refrigerant leakage and fouling among the most common and costliest faults in heat pumps; particularly, for air-to-water heat pumps as claimed to insurances and manufacturers. According to Madani's results [27], from the analysis of 37.000 faults reported to manufacturers over the period 2010–2012 in Sweden.

The results of Kim and Katipamula [15] estimate that a refrigerant undercharging in the range of 25% can lead to an average reduction of 20% in cooling capacity and 15% in energy efficiency. Furthermore, an undercharge of about 25% would cause an average penalty in seasonal energy efficiency ratio (SEER) of about 16 % and a cost penalty of US$60 per year per refrigeration ton of rated capacity. These penalties could be considered cost savings associated with improving refrigerant charge levels and are very significant.

For evaporator fouling, a reduction in airflow of 50 % decreases the average capacity by 14%, whereas the energy efficiency decreases by 12%. The average SEER value decreases by 10% and annual cost increases by US$24 per refrigeration ton. For condenser fouling, a 50% reduction in airflow decreases the average capacity by 9%, whereas energy efficiency decreases by 22%. The SEER value decreases by 20% and annual cost increases by US$80 per refrigeration ton. Evaporator fouling has more influence on capacity than on efficiency, while condenser fouling has more impact on efficiency [15].

2.2.3.1 Evaporator Fouling

Fouling is defined by Awais and Bhuiyan [28] as the accumulation of unwanted or undesirable deposits on heat transfer surfaces, which result in thermal conductivity and pressure drop detriments by the presence of corrosion, erosion, incrustation, or bacterial growth. Notwithstanding, in the context of HVAC FDD, fouling is referred to as the effects of low flow through a heat exchanger (HX) occasioned by previously mentioned phenomenons.

Research has demonstrated that the main consequence of fouling is not the reduction of surface heat transfer coefficient (in certain cases slightly improved), instead, the dominant effect is the increased pressure drop. This reduces the flow of air or water through the HX reducing the overall heat transfer coefficient (UA) [17]. This research will take into account only the low outdoor airflow of the HX responsible for capturing heat in the vapor-compression cycle, i.e., evaporator fouling (EF). Nonetheless, it has been reported that EF and condenser faults can produce the same patterns at high intensities. This observation indicates that the expansion device plays an essential role in the system's response to certain faults [29].

Du et al. [29] conducted a comparison study between different types of HVAC devices. They concluded that there are substantial similarities between the repercussions of the same faults in heating and cooling modes, while noting specific sensitivities that must be accounted for in each mode. This is an important caveat to bear in mind, because fewer studies are conducted on heating mode, as demonstrated in Mehrabi and Yuill [24, 25], where relationships for EF could not be provided. Response of systems to faults may differ from equipment to equipment due to differences in their overall type, design or component selection (mainly in expansion valve and accumulators), and control system [30, 29, 7].

General effects of EF are [31, 30, 29, 25, 32, 6]:

- Decrease of evaporator's temperatures as a consequence of a drop in evaporator saturation temperature (e.g liquid line, suction line, superheat, exit airflow temperatures).

- Lower refrigerant's mass flow rate.
- Compressor's discharge temperature increase.
- Raise in condensing temperature.
- Reduction of COP and capacity.
- Boost in compressor's speed.

Several researchers have proposed simulation of air-side fouling by covering portions of the face of the heat exchanger with paper (e.g., [25, 6]). Others have limited the fan speed to induce the reduction in airflow (e.g., [32, 26]). The present work will focus on evaporator fouling and refrigerant leakage further on, including refrigerant undercharge within leakage.

2.2.3.2 Refrigerant Leakage

Refrigerant leakage (RL) is depicted by an insufficient quantity of working fluid within the vapor-compression cycle. This could happen during commissioning or service because of inadequate charging, or due to the rupture of a pipe, seal, valve, among other components [6]. The International Institute of Refrigeration in his 24th technical note reports losses up to 10% per year of refrigerant on commercial and residential air conditioning [33]. The experimental replication of this fault is straightforward in the cases of undercharge or leakage [29, 32]. Nevertheless, compressor's valve or 4-way valves leakages (CVL) can induce recirculations that could have different impacts. This fault is modeled by bypassing the refrigerant in specific sections of the cycle. CVL will not be further considered in this study.

General effects of RL [31, 30, 34, 29, 25, 32, 6]:

- Significant degradation of COP due to diminished capacity.
- Condenser temperature and subcooling decrease.
- Increase of evaporator's and compressor's temperatures (superheat, suction, discharge temperatures) as a result of the reduction in refrigerant mass flow.
- Power consumption is somewhat affected, however, the consequences will vary depending on the expansion valve and speed regulation.

2.2.4 FDD Challenges

The primary bottlenecks to FDD implementation in the field are the high initial costs of additional sensors, and the need for customization of software solutions for each specific building or equipment [22, 15, 9, 8]. Another difficulty in applying the existing approaches is in handling multiple faults that occur simultaneously because the state variables can depend on more than one fault along with the operating conditions [22, 9, 18]. Kim and Katipamula [15] identified the following general challenges for FDD applications:

- Lack of automated tools and processes to automatically map data sources to AFDD tools. This impedes scaling AFDD services and also increases the cost of deployment. Many researchers have recognized this as one of the key challenges to overcome. Low-cost reliable sensing certain type of measurements (air flow, pressure, power, etc.) are necessary.
- Most AFDD techniques, especially rule-based ones and classifiers, rely on simple thresholds of the features to identify faults. If the thresholds are not properly selected or are not general enough, too many false alarms may be generated or faults may be misidentified. [15, 9].
- Detection of faults and diagnosing the cause of the faults are two important steps in the AFDD process. However, without an estimate of fault severity and of the energy and cost impact, building operators lack the knowledge to prioritize whether to address and/or repair the fault. Many studies have focused on the identification of faults and diagnosis of their causes, still, there is a relative void in studies that focus on estimating the fault impact.
- Considering the spectrum of black-box models, parameters, data preprocessing and other techniques, it is necessary to develop methods that eliminate the need for manual model identification or algorithm training. This provides flexibility for adapting the methods to the change in the configurations of HVAC systems. Some self-training models have been developed in the literature to adapt to the changes of the system within time, clearly, these models are to be developed after system installed. This goal is accomplished by slowly developing the model as different operating conditions become available [9]. As illustrated by Bode et al. [35] in his "real-world" application study, determining the no-fault status of an installed system after several years of operation, is a case on its own.

Zhao et al. [8] expand the previous list adding the shortcomings for data-driven methods. They distinguish *incomplete information* and *uncertainty* as the major issues. *Incomplete information* refers to the lack of sensors, normal data, faulty data and physical parameters. *Uncertainty* in this context entails measurement errors, probabilistic relations among symptoms and faults, as well as inaccurate knowledge. Additionally, provide relevant insight into:

- Inadequacy of current data storage practices: where states that data commonly is only temporary saved.
- Poor accuracy of a large portion of measurement devices.

- Fault propagation by control loops.
- Lack of public databases: as a benchmark and base for new normal data and faulty data of some typical buildings developments.

Specific challenges for the residential sector are listed by Roger et al. [9]:

- The majority of methods use an extensive set of laboratory experiments to characterize the behavior. Nevertheless, these experimental results are not available for all systems. Even when they are available, these models may not capture the installation variations (for example in split-systems further apart than expected). Installed systems will probably need to be benchmarked during the commissioning process.
- The required sensor package must be significantly simplified. This simplification involves reducing the number of sensors, accounting for the type of sensors used, and considering where these sensors must be installed. This would be an important part of increasing the cost-effectiveness of FDD methods.
- There is not an established benchmark for HVAC applications. In terms of square footage and location, similar households could be compared through such framework. By leveraging the data from many homes, researchers could provide insight to specific homes via anomaly-detection methods to faulty or even less efficient cases. The utility of process-history approaches would be significantly improved with an expanded dataset.

Up to this point, the marketplace has been slow to offer these high-level features. Primarily, it has been justified by the high costs associated with providing site-specific solutions relative to the savings potential. The additional costs are due to additional sensor requirements and labor to engineer and program these applications [22]. It is essential to understand the benefit of FDD throughout the value chain. Unequivocally, the increase in costs from FDD implementation must be accounted by someone, still, multiple parties could share the costs once these benefits are reckoned [9]. Examples of incentives could be:

- Electric-grid operators could provide an incentive in the form of a cash rebate for customers who install the FDD system. The reduced peaks in electricity demands facilitate a more stable grid.
- Manufacturers could offer an FDD-enabled system to the dealer at a discounted price to receive feedback about their services.
- The dealer, installer, and service company could also pay for access to the FDD data. This information from the value chain (excluding final consumer) could provide a direct source of improvement opportunities, also assuring a history of reliability.

2.2.5 Virtual Sensors

One of the key areas of improvement is the development of low-cost AFDD algorithms that reduce the number of sensors. Faults have effects over the entire cycle, which makes it necessary to measure several variables in order to detect a set of faults; moreover, the degradation of the performance of a system. Studies have relied mostly on techniques to create analytical “virtual” sensing as derived parameters [36, 37, 38, 15, 39] called *virtual sensors*. These parameters are equations formulated from energy and mass balances, numerical approximations or empirical relationships. Among the benefits of the virtual sensors (VS), Kim and Lee [22] identify:

- The derived indicators can provide a check on the accuracy of an installed sensor, even enabling virtual calibration. The combination of this features results in more robust FDD systems.
- The diagnostic approaches based on these methods can identify and isolate specific faults using only a number of low-cost physical sensors.
- The FDD process can be simplified through simple thresholds between the output of the virtual sensor and the real sensor in order to detect faults.
- VS can be modeled to represent features specific to a fault, hence, insensitive to other possible faults; this is known as "decoupled virtual sensors".
- These methods could simplify the diagnostic task. Fault diagnoses would result directly from the deviation of decoupled features from expected values.

Virtual sensors are based on the early work of Li and Braun [36], where the first equations were presented. This methodology treats each component of the vapor-compression cycle independently in order to assess specific features and faults. The outputs are VS which can be used as an input in other equations. This work will reference to the most updated version of some of these equations [39], for further concerns review the early works of Li and Braun or the posterior studies from Kim and Braun [36, 37, 38, 15, 39].

Equation 2.8 is referred to as *Power virtual sensor*. Evidently, it is an ANSI/ARI Standard 540-1999 10 coefficient polynomial for fixed speed compressors, where c_i are the polynomial coefficients (i=[1 ; 10]), S is the suction dew point temperature and D represents discharge dew point temperature. Kim and Braun [38] studies demonstrated that is an appropriate estimate even in faulty conditions; as long as faults in the compressor are not being accounted for.

Equation 2.9 represents the *Refrigerant’s mass flow virtual sensor*, where α_{loss} depicts a model for compressor’s heat loss, h_{dis} compressor’s discharge enthalpy, h_{suc} compressor’s suction enthalpy. Kim and Braun [38] note that α_{loss} is generally very small ($< 5\%$), thus it will be neglected for further implementations in this study.

Equation 2.10 depicts *Evaporator's volume airflow*, which introduces h_{liq} is the liquid line enthalpy, $h_{\text{e,out}}$ stands for the enthalpy at the exit of the evaporator, $h_{\text{a,in}}$ and $h_{\text{a,out}}$ the air enthalpy at the inlet and outlet respectively. This equation and the previous (2.9) both rely on enthalpies that can be calculated using thermodynamical relationships of pressures measurements. The authors allow this within a low-cost sensor approach since Li and Braun [37] established a method to position surface-mounted thermocouples to obtain saturation temperatures in the condenser and evaporator. Nevertheless, this method is limited to systems with fin and tubes heat exchangers, whereas an air-to-water heat pump will not entirely apply.

Equations 2.11 is the *Heating capacity virtual sensor*, which presents h_{dis} the discharge enthalpy, $h_{\text{c,out}}$ is the enthalpy at the exit of the condenser. Finally, 2.12 relates to the *Coefficient of performance* virtual sensor, this is the ratio between the capacity and the power consumption. These last VS are known as the performance virtual sensors.

$$\begin{aligned}\dot{W}_{\text{vs}} = c_1 + c_2 \cdot S + c_3 \cdot D + c_4 \cdot S^2 + + c_5 \cdot S \cdot D + c_6 \cdot D^2 + \\ + c_7 \cdot S^3 + c_8 \cdot S^2 \cdot D + c_9 \cdot S \cdot D^2 + c_{10} \cdot D^3\end{aligned} \tag{2.8}$$

$$\dot{m}_{\text{ref,vs}} = \frac{\dot{W}_{\text{vs}} \cdot (1 - \alpha_{\text{loss}})}{h_{\text{dis}}(P_{\text{c}}, T_{\text{dis}}) - h_{\text{suc}}(P_{\text{e}}, T_{\text{suc}})} \tag{2.9}$$

$$\dot{V}_{\text{a,vs}} = \dot{m}_{\text{ref,vs}} \cdot v_{\text{a}} \cdot \frac{h_{\text{liq}}(P_{\text{cond}}, T_{\text{liq}}) - h_{\text{ev,out}}(P_{\text{evap}}, T_{\text{ev,out}})}{h_{\text{a,in}} - h_{\text{a,out}}} \tag{2.10}$$

$$\dot{Q}_{\text{h,vs}} = \dot{m}_{\text{ref,vs}} \cdot v_{\text{a}} \cdot h_{\text{dis}}(P_{\text{dis}}, T_{\text{dis}}) - h_{\text{cond,out}}(P_{\text{cond}}, T_{\text{cond,out}}) \tag{2.11}$$

$$COP = \frac{\dot{Q}_{\text{h,vs}}}{\dot{w}_{\text{vs}}} \tag{2.12}$$

2.3 Machine Learning

Already in 2005, Katimapula and Brambley [13] recognized methods that took advantage of historical data, in their classification named *Process history-based FDD* explained in Section 2.2.1.3. Within these methods, they identify the *black-box* models, whose deployment is now largely extended. With this idea, Zhao et al. [8] propose another classification, where the previous category evolved to *Data Driven-based* models and its subcategories are inherited from a research domain called *machine learning.* The upcoming concepts establish a framework to explore the association of FDD and this field.

The field of machine learning (ML) is concerned with the question of how to construct computer programs that automatically improve with experience. This can be defined as: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" [40]. Thus, "Machine Learning is the science and art of programming computers so they can learn from data" [41], where data is treated as the translation of *experience*.

2.3.1 Machine Learning Concepts

The following sections aim to describe the most important concepts to consider when working with machine learning models, these notions are fundamental to assess performance and improve accuracies.

2.3.1.1 Generalization

Most Machine Learning tasks are about making predictions. Given a number of training examples, the model needs to be able to make good predictions for examples it has never seen before. Training a model implies running an algorithm to find parameters that will make it best fit the training data and make good predictions on new data. [41, 42]. This opens two possibilities:

- The model over-generalizes, performing too well on the training data but poorly on new instances, this is called *overfitting*. Overfitting happens when the algorithm is too complex relative to the amount and noisiness of the training data [41].
- The model under-generalizes, unsuccessfully adapting to any data, both training and new examples. Occurs when the model is too simple to learn the underlying structure of the data [42].

Evidently, the only way to know how a model will generalize to new cases is to try it out on unseen cases. Hence, the task on how to partition the data arises. A common strategy is to split the available examples into *training sets* and *test sets*; the latter is reserved and only used to evaluate performance. The error rate on the new cases provided by the test set is called the *generalization error*. If the error during training is low, but the generalization error is high, the current model is overfitting the training data [41].

A key aspect to consider when fractionating the data is *data leakage*. *Leakage* insinuates that information is revealed to the model which gives an advantage to make better predictions. For instance, this could happen when data from the future is leaked to the past in a time-dependent dataset. Any time that a model is given information that it should not have access to when it is making predictions in real-time in production, there is leakage. [43]

Another common approach is to further divide the training set to generate an intermediate test set, known as the *validation set*; which can be used to compare different algorithms or parameters of the same algorithm, before the final evaluation. If the generalization error is estimated directly using the test set, the result is at risk of being too biased towards the user's requirements. This is called *data snooping bias*, it unveils an important liability: if the test set does not represent in a good measure the data that the model will encounter, there is no certainty of its future performance. [41]

Additionally, Géron [41] identifies important aspects which the data must comply for any machine learning algorithm:

- Sufficient quantity representative of the cases to generalize: small *sampling noise* and adequate sampling method, where the *sampling bias* is accounted for.
- High quality: low levels of errors, outliers, noise or missing information.

2.3.1.2 Variance and Bias

The generalization error can be expressed as the sum of:

- Bias: this part of the generalization error is due to wrong assumptions. A high-bias model is likely to underfit the training data.
- Variance: implies excessive sensitivity to small variations in the training data. A model with many degrees of freedom, such as a high-degree polynomial is likely to have high variance, thus overfitting the training data.
- Irreducible error: this part is due to the noisiness of the data itself.

There is a correlation between these factors known as the *variance-bias trade off*: increasing model complexity increases its capacity to fit the training data (overfitting and increased variance), while reducing complexity can lead to underfitting (increased bias). [41]

2.3.2 Supervised Learning

In his book "Machine Learning", Mitchell (1997) [40] establishes notions of *direct* and *indirect* training, where the roles of the model (learner), user (teacher) and how much information a possible *feedback* provides, are conceptualized. These definitions established a framework for the modern classifications of learning.

Machine Learning systems can be classified according to the amount and type of supervision they get during training. There are four major categories: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning [41]. This work will cover only the first one mentioned previously, because it is the most explored in the current literature.

When the machine is trained with *labeled data* and tested with new data is called *supervised machine learning*, the fact that an example is labeled suggests that it has been previously tagged with the correct output [14]. In other words, the training set fed to the algorithm includes the desired solutions [41]. This approach requires that both the input and output data must be known in order for a supervised model to develop a mathematical function that describes the relationship between input and output. Based on this function, the model can predict the output value using previously unobserved input values.

Supervised learning can be further divided into *classification* (for discrete output values) and *regression* (for continuous output values). This type of learning is often highly interpretable; which provides a sense of reliability [16]. The considered supervised learning algorithms in this work are:

- Support Vector Machine (SVM).
- Decision Trees (DT).
- Artificial Neural Networks (ANN).

2.3.3 Algorithms

This section pursues to conceptualize the ML algorithms employed in the realization of this research. Machine learning models are basically mathematical functions that represent the relationships between different aspects of data [42]. Each represent alternative approaches to obtain accurate discrete or continuous outputs; furthermore, certain algorithms are utilized to reveal patterns among data.

Many models have important parameters which cannot be directly estimated from the data. These model parameters are referred to as *tuning parameters* [44] or *hyperparameters. Model parameters* are those variables (within the model's internal structure) that are able to modify directly from its interaction with the data, commonly from the training set. On the other hand, hyperparameters are not learned, hence, a first speculation is made for a latter tuning step. A noteworthy remark about hyperparameters lies in their potential impact on performance and prediction accuracy, different settings could represent substantial differences in prediction accuracy and generalization [42].

2.3.3.1 Support Vector Machine

Support Vector Machines (SVM) are built on statistical learning theory for structural risk minimization [45]. SVMs are capable of performing linear or nonlinear classification, regression, and outlier detection. Additionally, they are particularly well suited for the classification of complex small to medium-sized datasets [41]. SVM is a binary maximum margin classifier, defined by a boundary that is unit-separated from the nearest instances of both classes using the simplest possible representation (regularization effect) [16]. The fundamental concept of the SVM is to draw a *decision boundary* that separates the data into distinct categories. This decision boundary is known as a *hyperplane*, where the points closest to the line from both classes are known as *support vectors* [14]. SVM will not only separate the two classes, also stays as far away from the closest training instances as possible. They are the closes points to the boundaries (gray lines) that define the *street*. An SVM classifier will fit the widest possible street between the classes [41]. The goal of the model is to determine the boundary, ensuring the largest distance between it and the closest data instances of the two classes [16]. The distance between the hyperplane and support vectors is known as *margin*. This maximized margin draws an optimal hyperplane through a process called *large margin classification* [14, 41].

Although SVMs are considered linear classifiers, they are not limited to problems with linear patters. Complex distributions can be encoded using *kernel functions*, this is referred to as the *kernel trick*. A kernel function is basically a computation of high-dimensional relations of input data without the need to explicitly transform the data. It reduces the computational cost by avoiding the transformation of the data, and allows the computation of relations in an unbounded number of dimensions. Polynomial kernels and radial basis function kernels are the most commonly used [16].

2.3.3.2 Decision Trees

A Decision Tree (DT) is a model that uses *sample features* to build rules that classify data predictively. The decision tree selects the best features to segment the data in a recursive manner [46]. DTs consist of *root nodes*, decision nodes, and leaf nodes. A root node, also called a parent node, represents the entire population and divides the data into two or more nodes [16]. Root nodes can be pictured as the stem. Within this structure, each non-leaf node represents one feature, each branch of the tree represents a different value for a feature, and each *leaf node* represents a class of prediction [47]. This means that the leaf nodes are in the last layer of the tree, thus, are closer to the algorithms output. When developing a tree, decisions must be made about which features to include as input, the conditions for splitting and when to stop further branching of the tree [16]. DTs have become a very popular ML technique because of its simplicity, ease of use, and interpretability; as it can be easily visualized and explained [48, 16]. Furthermore, this algorithm is used as a base for more complex classifiers, which instead of training one instance, assemble groups of several DTs trained and grown under distinct methodologies. These combinations are known as an *ensemble*, relevant to this work are two types of ensemble learners: Random Forest and Extreme Boosting Gradient.

Random Forest

The Random Forest (RF) algorithm introduces extra randomness when growing trees. Instead of searching for the very best feature when splitting a node, it searches for the best feature among a random subset of features [41]. An RF model predicts a class by averaging the results of multiple trees, and its accuracy improves as the number of trees increases. This is accomplished through a process of sampling with replacement called *bagging* coupled with the random feature selection mentioned earlier, performed at each tree-building step to train ensembles of trees for attaining higher predictive accuracy [16]. Consequently, the algorithm results in greater tree diversity, which trades a higher bias for a lower variance, generally yielding an overall better model [41]. RF is a model able to handle encoding of more complex distributions by using highly expressive individual models whose variance is in turn constrained through voting during inference [16]. In conclusion, RF models are an improved version of population intelligence-based decision tree model. The unpredictability of random forest refers to the use of a random attribute selection strategy for training each decision tree, ensuring that there is no correlation between them [46].

Extreme Boosting Gradient

Boosting is a typical classification learning integration method in which a series of weak classifiers are learned by iteratively modifying the training data's probability distribution, and then these weak classifiers are linearly merged to generate a strong classifier. When the decision tree is the basis function, boosting is called Boosting Tree (BT). Gradient Boosting (GDBT) integrates multiple weak learners into the final predictive model, and at each iteration, a learner that minimizes loss in the direction of the steepest gradient is generated to compensate for the deficiencies of the existing model [46]. This method tries to fit the new predictor to the residual errors made by the previous predictor [41]. Unlike the random forest, which generates decision trees independently of each other, the GDBT model builds on the previous trees from the second tree onwards [46]. An optimized implementation of Gradient Boosting is available called XGBoost, which stands for Extreme Gradient Boosting. This package aims to be extremely fast, scalable, and portable [41].

2.3.3.3 Artificial Neural Networks

An Artificial Neural Network is a nonlinear informational processing device, which is built from interconnected elementary processing devices called *neurons*. Each input is multiplied by a connection weight. The products and *biases* (special additions) are summed and transformed through a *transfer function* (algebraic equations such as *log-sigmoid* , *tangent-sigmoid*, or *rectified linear unit*) to generate a final output. The process of combining the *signals* and generating the output of each connection is represented as *weight* [49]. Model weighting is adjusted until the model has the smallest possible margin of error. Due to their structure, ANNs can encode more complex representations by adding more hidden layers [49].

Multiple layers of neurons with nonlinear transfer functions allow the network to learn linear and nonlinear relationships between input and output vectors [49]. An MLP is generally composed of one pass-through *input layer*, one or more layers of threshold logic units (TLUs) known as *hidden layers*, and one final layer of TLUs called the *output layer*. TLUs are the neurons, they compute a weighted sum of its inputs, then applies the transfer function to that sum and output a result. Training a TLU in this case means finding the right values for each weight [41].

The Back-propagation algorithm (BPA) is widely used to train an ANN. BPA optimizes the weight connection by allowing the error to spread from output layers towards the hidden layer and input layer [49]. For each training instance, the backpropagation algorithm first makes a prediction (forward pass) and measures the error, then goes through each layer in reverse to measure the error contribution from each connection (reverse pass), to finally modify the connection weights to reduce the error [41].

2.3.4 Performance metrics

In a generic sense, performance metrics are linked to the concepts of distance and similarity [50]. One of the fundamental tasks in building any ML model is to define how to evaluate its performance. The achievement and degree of success require to be comprised within an objective metric to assess the compliance of the set goals. Furthermore, the definition of an appropriate metric prior to the establishment of objectives can lead to more attainable goals [42]. The selection of the performance metric should account for:

- Model learning type.
- Model phase: training, testing or evaluation.
- Data scale.
- Data distribution.

If within the data, there are significantly more examples of one group than another, some metrics will give a very distorted picture because the most represented class will dominate the statistic. Any metric that gives equal weight to each instance of a class has a hard time handling *imbalanced classes*. The extension of issues that arise from this scenario transcend to all development stages. They are problematic not only for the final evaluation stage, also when training the model. If class imbalance is not properly dealt with, the resulting model could be unable to predict the minority classes [42].

The following metrics are applicable to supervised learning, specifically to classification and regression tasks.

2.3.4.1 Classification metrics

The classification process aims to predicting class labels given input data. Evidently, in *binary classification* there are two possible output classes and in *multiclass classification* there are more than two possible classes.

The next measures define a relationship between *true positive* (TP), *true negative* (TN), *false positive* (FP), *false negative* (FN). TP are the elements that have been labelled as positive by the model, and they are actually positive, while FP are the elements that have been labelled as positive by the model, but they are negative in reality. On the other hand, FN are the elements that have been labelled as negative by the model, but they are positive. These last two are referred in statistics as: Type I and Type II error. Additionally, the total number of possible outputs (classes) is indicated by C, the total number of instances by N, i refers to row number and k alludes to column number.

Confusion matrix

Despite the confusion matrix (CM) is not a metric by itself, it provides a clear graphical representation of the posterior concepts, enclosing all the relevant information about the algorithm and classification rule performance. Basically, it is a distribution of the model's predictions, counting each individual instance and presenting it in a matrix's cell. It shows a more detailed breakdown of correct and incorrect classifications for each class. The rows of the matrix correspond to actual labels, and the columns represent the predictions. The confusion matrix is a squared matrix of size C *x* C. The classes are listed in the same order in the rows as in the columns, therefore the correctly classified elements are located on the main diagonal from top left to bottom right [51, 41, 42].

Table 2.1: Confusion matrix example.

		Predicted					
		Class 1	Class 2	Class 3	. . .	**Class k**	
	Class 1	c_{11}	c_{12}	c_{13}	. . .	c_{1k}	
	Class 2	c_{21}	c_{22}	c_{23}	. . .	c_{2k}	
Actual	Class 3	c_{31}	c_{32}	c_{33}	. . .	c_{3k}	
	.	.	.	.	.	.	
	.	.	.	.	.	.	
	.	.	.	.	.	.	
	Class i	c_{i1}	c_{i2}	c_{i3}	. . .	$\mathbf{c}_{ik}$	
							N

Accuracy

The accuracy reflects how often the classifier makes the correct prediction, it is the probability that the model prediction is correct. Represents the ratio between the number of correct predictions and the total number of predictions [42, 51]

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\sum_{i=1}^{C} c_{ii}}{N} \tag{2.13}$$

Precision

The precision is the fraction of TP elements divided by the total number of positively predicted units (diagonal cell divided by the sum of the column). Expresses the proportion of units our model classifies as positive that are actually positive [51].

$$Precision = \frac{TP}{TP + FP} = \frac{c_{ii}}{\sum_{i=1}^{C} c_{ik}} \tag{2.14}$$

Recall

The recall is the fraction of TP elements divided by the total number of positively classified units (diagonal cell divided by the sum of the row). Recall measures the model's predictive accuracy for the positive class, it measures the ability of the model to find all the positive units in the dataset. Recall is also known as *sensitivity* or *true positive rate* (TPR) [51].

$$Recall = \frac{TP}{TP + FN} = \frac{c_{ii}}{\sum_{k=1}^{C} c_{ik}} \tag{2.15}$$

Balanced Accuracy

The formula of the balanced accuracy is essentially an average of recalls. First is evaluated the recall for each class, then the values are averaged in order to obtain the Balanced Accuracy score. The value of recall depicts the likelihood for each class of each individual class to be classified correctly. Hence, the balanced accuracy provides an average measure of this concept across the different classes [51].

$$Balanced\ Accuracy = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) = \frac{1}{C} \cdot \sum_{i=1}^{C} \frac{c_{ii}}{\sum_{k=1}^{C} c_{ik}} \tag{2.16}$$

2.3.4.2 Regression metrics

In regression tasks, the model learns to predict numeric scores. The *distance* between the actual value and the prediction is calculated to obtain a measure of the *error* in the prediction [42]. Botchkarev [50] groups the following metrics as "Primary metrics" given they are used for constructing further numerical indicators. Essentially, primary metrics involves three steps: calculating point distance, performing normalization and aggregating point results over a data set.

Root Mean Squared Error

The root-mean-square error (RMSE) is the most commonly used metric for regression tasks, it is defined as the square root of the average squared distance between the actual score and the predicted score. This equation computes the Euclidean distance between the vector of the true scores and the vector of the predicted scores, averaged by $\sqrt{n}$, where n is the number of data points, y_i denotes the true score for the i_{th} data point and $\hat{y}_i$ denotes the predicted value. RMSE describes an error range in which the predictions of a regression model lie. Although RMSE can be affected by large outliers, it is able to withstand them better than other primary metrics such as: Mean Absolute Error, Mean Squared Error, etc, because of an attenuating effect of the square-root function. [42, 50]

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.17}$$

Coefficient of determination

The coefficient of determination (R^2) can be viewed as a measure of the proportion of the sum of squares of deviation of the y_i about their mean. Hence, R^2 measures the goodness of fit, in the sense of comparing a model with another in which none of the independent variables appear. In practical applications, R^2 is used as a metric of the usefulness of a regression equation in the sense of comparing two models [52], generally the predictions are compared with the expected output of a constant horizontal line. This measure is of common use within optimization functions in hyperparameter tuning. While the RMSE only accounts for the mean difference of actual values and predictions, R^2 considers the variation of the data, providing a metric of how well the predictions emulate the actual trend. To illustrate the difference, suppose the case of two similar curves with an offset. In this case, R^2 will not give insight of this event, while RMSE will provide a result that shows the offset.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.18}$$

Where n is the number of observations, and $\bar{y}$ is the mean value of the dependent variable.

2.3.5 Machine Learning Libraries

2.3.5.1 Scikit-learn

Scikit-learn (SKL) is a Python module integrating a wide range of ML algorithms for medium-scale supervised and unsupervised problems. This package focuses on machine learning using a general-purpose high-level language. Emphasis is put on ease of use, performance, documentation, and API consistency. SKL harnesses this rich environment to provide state-of-the-art implementations of many well-known ML algorithms while maintaining an easy-to-use interface tightly integrated with the Python language. It depends only on *Numpy* and *Scipy* libraries to facilitate distribution. Additionally, Scikit-learn provides over 300 pages of user guide [53].

Scikit-learn comprises a wide variety of machine learning algorithms, using a consistent, task-oriented interface, thus enabling easy comparison of methods for a given application. Since it relies on the scientific Python ecosystem, it can easily be integrated into applications outside the traditional range of statistical data analysis [53].

In his internal structure, objects are specified by the interface, not by inheritance. Hence, to facilitate the use of external objects with SKL, inheritance is not enforced. Instead, code conventions provide a consistent interface. The central object is an *estimator*, that implements a *fit* method, accepting as arguments an input data array and, optionally, an array of labels. Supervised learning estimators can implement a *predict* method. Some estimators, referred to as *transformers*, implement a *transform* method, returning modified input data. Estimators may also provide a *score* method, which is an evaluation of the goodness of fit [53].

2.3.5.2 Hyperopt - Scikit learn

Hyperopt-scikit learn (HPSKL) is a module build over the idea that: "the choice of classifier (also applies for regressors) and even the choice of preprocessing module can be taken together to represent a single large hyperparameter optimization problem". When there is no preference over the classifier, generally the selection is made based on the one that provides greater accuracy. In this light, the choice of classifier can be seen as hyperparameter. Likewise, the choice and configuration of preprocessing components can be included in this optimization *pipeline* [54].

This approach is possible given the size of data sets and the speed of computers have increased to the point where it is often easier to fit complex functions to data using statistical estimation techniques than it is to design them by hand. The fitting of such functions, which in this case would be the training of ML algorithms, remains a relatively arcane art, typically mastered in the course of a graduate degree and years of experience, according to Komer [54]. Considering that algorithms like RF or SVM have a small enough number of hyperparameters that manual compared to ANN, there is an increased probability that manual tuning, or *grid search* approaches provided satisfactory results until this point.

HPSKL uses Hyperopt to describe a search space over possible configurations of Scikit-learn components (DT, SVM, ANN), including preprocessing (scalers, dimension reductors), classification, and regression modules. The Hyperopt library offers optimization algorithms for search spaces that arise in algorithm configuration [54]. To use Hyperopt, the user must define:

- Search domain.
- Objective function.
- Optimization algorithm.

The search domain is specified via random variables, whose distributions should be chosen so that the most promising combinations have high prior probability. The objective function maps a joint sampling of the random variables defined in the search domain to a scalar-valued score that the optimization algorithm will try to minimize [54].

The optimization algorithm is defined and implemented through the *fmin* function, whose call carries out the simple analysis of finding the best-performing configuration, and returns that to the caller. The optimization algorithms present in Hyperopt are: random search, annealing search, and tree of parzen estimators. Testing has showed that HPSKL implementation is viable, however, it can be of slow convergence [54].

HPSKL provides a parameterization of a search space over *pipelines*, that is, of sequences of preprocessing steps and classifiers. Hyperopt description language allows us to differentiate between conditional hyperparameters (which must always be assigned) and non-conditional hyperparameters (which may remain unassigned when they would be unused). We make use of this mechanism extensively so that Hyperopt's search algorithms do not waste time learning by trial and error [54].

HPSKL defines an *estimator* class with a *fit* method and a *predict* method. The *fit* method of this class performs hyperparameter optimization, and after it has completed, the predict method applies the best model to test data. Each evaluation during optimization performs training on a large fraction of the training set, estimates test set accuracy on a validation set, and returns that validation set score to the optimizer. At the end of search, the best configuration is retrained on the whole data set to produce the classifier that handles subsequent predict calls [54].

Testing of HPSKL implementation in three different benchmark datasets (20-Newsgroups, MNIST, and Convex Shapes) shows that the scores of HPSKL are relatively good on each data set. Moreover, the results indicate that Hyperopt's optimization algorithms are competitive with human experts. From these outcomes, the difficulty and importance of hyperparameter search is highlighted [54].

2.3.5.3 Automated Data-Driven Modeling tool

The Automated Data-Driven Modeling tool (ADDMo) was designed for building's energy systems optimization and control, example applications are: component of a grey-box model development, forecasting, and set-point alteration. Fundamentally, ADDMo is a software tool developed to generate and optimize regression models. It automates data preprocessing, feature engineering, and model selection tasks. Studies comparing ADDMo results and a manual modeling approach via Scikit-learn revealed that obtains better results in all tested use cases [55].

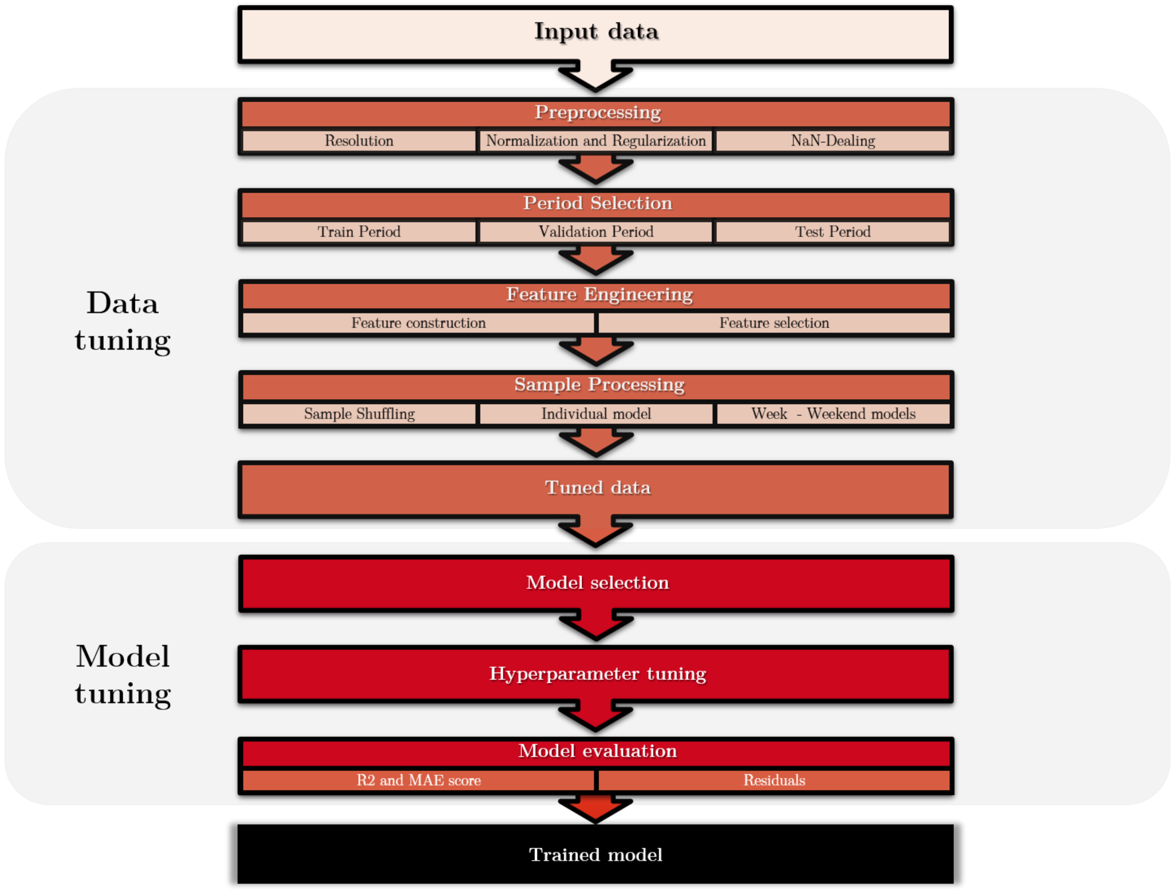


Figure 2.6: ADDMo workflow.

ADDMo was built under the consideration that achieving the optimal model is a highly time-consuming endeavor; it requires a large amount of computational power. Particularly, manual modeling additionally requires expert knowledge in black box modeling and the implementation of various tuning methods, which result in increase workload. There is an immense amount of possibilities for tuning, which leaves the user in uncertainty whether the respective tuning approach is going to be beneficial. This could lead to unsuccessful tuning attempts or incomplete tuning methods (early stopped) [55].

ADDMo executes most of the time-consuming and error-prone processes of data-driven modeling. It is structured to provide mechanisms to the following challenges of black box modeling [55]:

- Preprocessing of initial data.
- Selection of proper training and test data periods.
- Selection and creation of optimal features.
- Selection of a model.
- Hyperparameter tuning.
- Overfitting and underfitting.
- Trade-off between accuracy and computational costs.

These challenges are addressed in two stages: *Data Tuning* and *Model Tuning.* Data tuning comprises:

- Preprocessing: involving resolution, scaling and normalizing, and “Not a Number" (NaN) dealing.
- Period selection.
- Feature construction.
- Feature selection.
- Sample processing.

On the other hand, Model tuning entails:

- Model and tool selection.
- Hyperparameter tuning.
- Training, testing, and evaluation of the final model’s performance.

Finally, ADDMo implements a variety of models, referred to as *model families*, which are supposed to summary most machine learning approaches. These are: Multilayer Perceptron (ANN), epsilon-Support vector Regression (SVR), Random Forest (RF), Gradient Tree

Boosting (GB), and Least Absolute Shrinkage and Selection Operator (Lasso). In conclusion, ADDMo framework represents a comprehensive and versatile tool, with high specificity in several steps of the modeling process, and proved effectiveness [55].

3 Methodology

This chapter is divided in three sections. First, Section 3.1 details the base model and a reference heat pump that defines input parameters and a baseline to compare performance trends; both in no-fault and fault conditions for heating mode. Section 3.2 describes the fault modeling strategies, the evaluation of the simulation model, and its limitations. At last, Section 3.3 enters the machine learning specifics, characterizing the strategies to develop a FDD protocol, and finishes with the criteria to select the most suitable method.

3.1 Use Case

As introduced in Section 2.2.2, the development of a FDD protocol conventionally requires of two models: the no-fault (normal) model and the fault model. To generate the necessary data for the black-box model, the use of simulation models was deemed as an adequate approach. Amasyali et al. [47] depicts that only a 19% of the data used in data-driven energy consumption prediction studies came from simulation models, as observable in Figure 3.1 where *PBM* stands for *Public Benchmark*, *SIM* means *Simulated data*, and *Real* represents *Real data*. Nonetheless, Bellanco et al. [6] currently report an increased use of virtual environments in heat pumps fault behavior research. The need for large datasets and reference libraries for HVAC components and buildings drives the trend towards simulation models in data-driven AFDD [56].

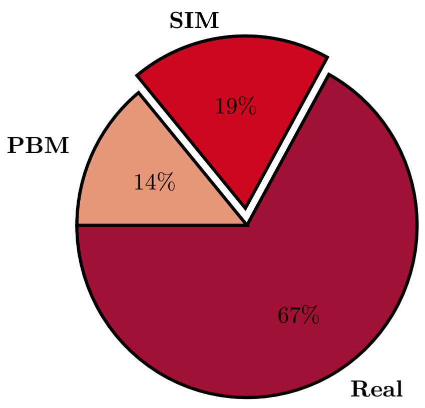


Figure 3.1: Data sources for data-driven methods [47].

Yuill and Braun [57] concluded that simulation models are the best method for providing AFDD evaluation input data. AFDD tool performance involves several hundred scenarios, which calls for immense laboratory time and high skilled technicians. Demonstrations of how labor-intensive this processes can become are found in studies such as Hu et al. [19]: where over one hundred experiments carried out to characterize multiple simultaneous faults at one ambient condition. For these reasons, simulation models validated with experimental data become a necessary alternative, because they are scalable, rapid, adaptable and cost-effective [56]. Experimental data is still required, however, the effort will be optimized.

Figure 3.2 represents the sequence of the upcoming subsections: first, the virtual environment is depicted, upon which the data is generated after further development; next, a real heat pump is presented to reference the modeling and performance. At last, the initial parameters for the specialization of the simulation are established.

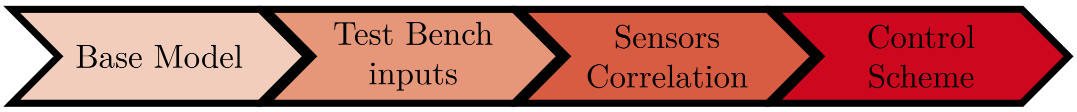


Figure 3.2: Model development outline.

3.1.1 Base simulation model

The base model is built in Dymola [58] software, with components from the TIL Suite library [59] in Modelica language [60]. This virtual environment emulates an air-to-water heat pump utilizing a simple vapor-compression cycle. Figure 3.3 portrays Dymola's graphical user interface, it allocates a simplified graphic of the heat pump main components. Each component represents their physical counterpart through several input variables, parameters, and calculation approaches.

The model consists of:

- Fin-and-tube evaporator
- Reservoir
- Fixed-speed compressor
- Brazed plate condenser
- Separator
- FXO expansion valve

The separator component is a computational formulation for numerical stability, its role differs from the real counterpart. This element is considered a virtual element which shall remain unaltered. Likewise, the *SIM* square (at the upper-right side) holds multiple pertinent setups; only applicable to the simulation. In particular, it allows specifying the simulation fluids: the refrigerant is R290 (propane), the hydraulic circuit fluid is liquid water, and ambient air is represented as moist air gas. The fluid models are taken from TIL media libraries [59].

Figure 3.4 illustrates the connections of all the elements within the model, how the main input variables associate with the corresponding component, and distinguishes the key parameters that are relevant for this study (lighter color). The scheme discriminates between major groups and parameters by a color code. Certain names provided were relabeled to assure an easy interpretation. The component-specific configuration approximates to the methodology described by Sterling et al. [61].

3.1.2 Reference test bench

To approximate to a reduced sensor configuration, an existing experimental test bench is selected, given its wide range of available measurements and suitable attributes for emulating faults. This device was constructed based on the work of Klebig [62] who pursued the construction of a modular-structured heat pump for low-GWP refrigerants. The strategy will be to take advantage of the existing sensors for modeling, data generation, and algorithm

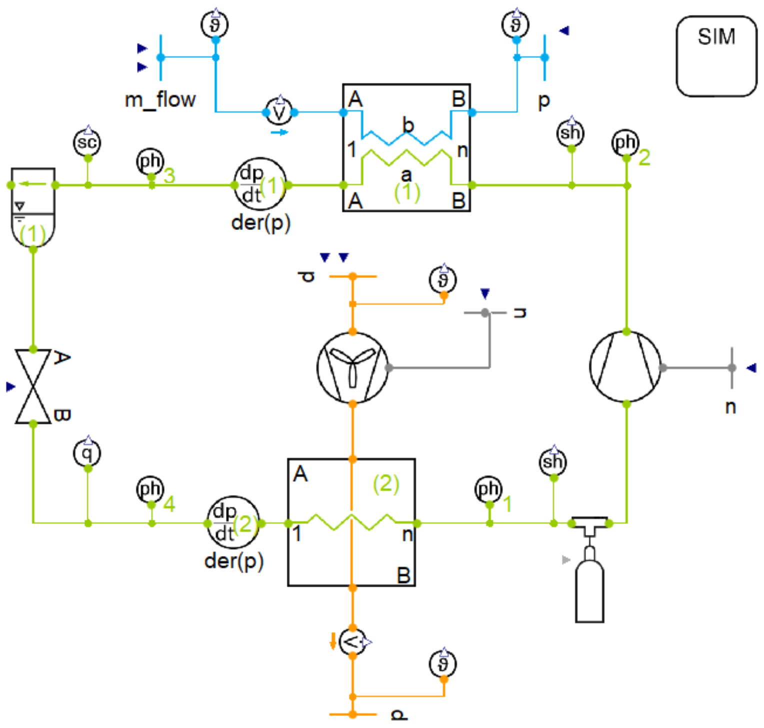


Figure 3.3: Base heat pump simulation model.

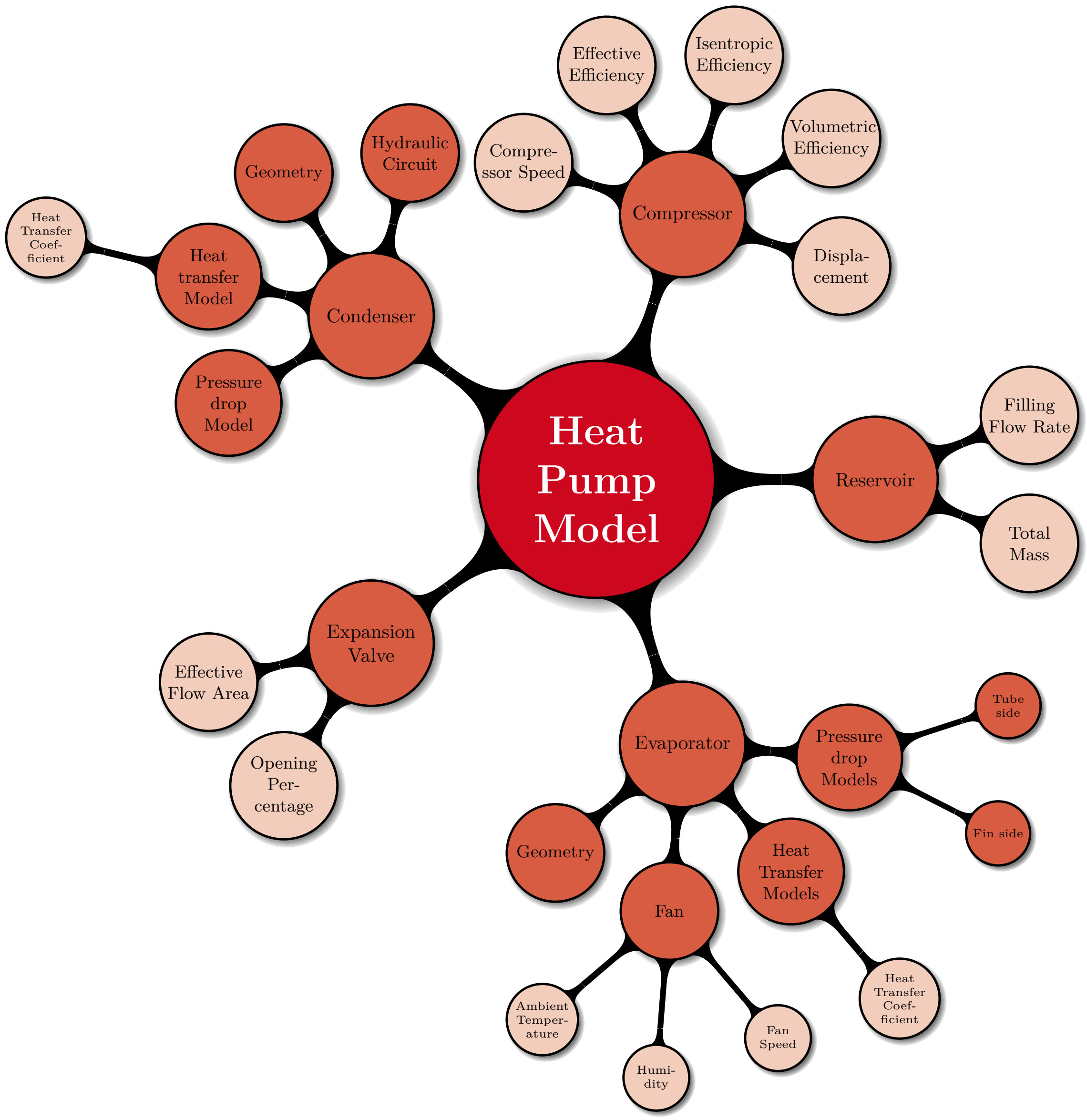
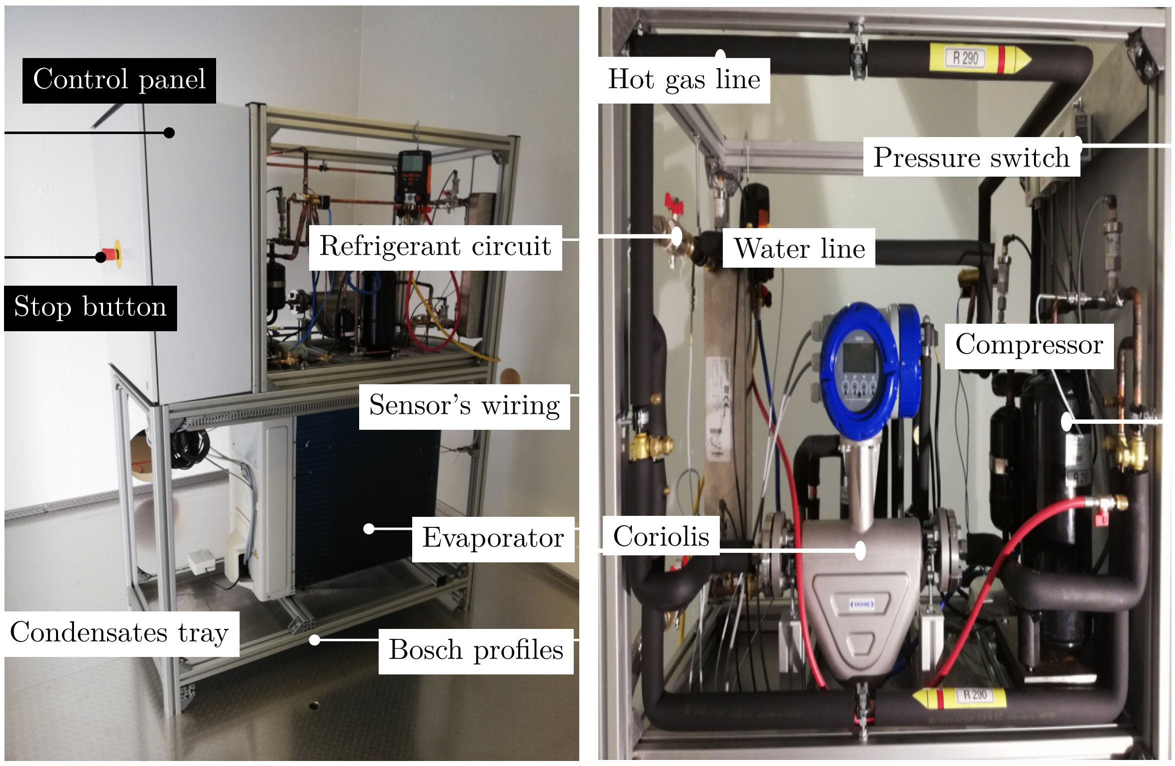


Figure 3.4: Base model breakdown.

training. Leveraging on the results of an extensive sensor configuration, reduced sensor alternatives will be explored.

Figure 3.5 shows the current distribution of the components of the test bench [62]. It is noteworthy to mention that the evaporator's case only contains a fin-and-tube HX and a fan. Table 3.1 provides an overview of the main components' specifications. This heat pump features inverter technology to regulate compressor's speed, control loops to maintain a constant superheat through the expansion valve opening manipulation, and fan speed setpoint based on the voltage signal; which also allows to carry on adaptations depending on the requirement.



(a) Test bench overview **(b)** Refrigerant cycle close-up

Figure 3.5: Experimental heat pump test bench [62].

With the intention to objectively assess the quality of the generated data, certain datasets gathered by Kleipass [63] are set as a baseline, for the outputs of both no-fault and fault models. The study carried out several experimental trials with the forenamed heat pump under numerous fault and no-fault conditions. A key note at this stage: this data will merely serve to establish context to the simulation output; by no means, the simulation model will intend to act as a precise digital counterpart of the physical heat pump. The validation of the simulation model, in order for it to replace the steady-state outputs of the equipment (under strict accuracy standards) is outside of the scope of this work. A fair resemblance

Table 3.1: Heat pump reference model specifications.

Component	Capacity	Type	Brand	Model
Evaporator	6,5 kW	Fin-and-tube	Daikin	ERGA08DAV
Compressor	7,475 kW	Rolling Piston	Hitachi Highly	WHP07600PSD
Condenser	10 kW	Brazed Plates	SWEP	B8LASHx30/1P-SC-M
Expansion Valve	-	EEV	Danfoss	ETS 6 - 18
Reservoir	3,4 l	-	EFM	-

that specifically mimics the trends of no-fault and fault trials is conceived as sufficient for the development of this research.

3.1.3 Component characterization

The Modelica language [60] is object-oriented, which opens broad possibilities for customization. This feature that has been exploited in works such as [64, 65, 66] to develop heat pump applications and data for FDD. This subsection details the configuration of the components with the most adjustable parameters, according to the current codification and the representation theory they follow.

3.1.3.1 Evaporator

From the various alternatives that this component provides, it is essential to focus on heat transfer and pressure drop models, combined with geometry. Appendix A.1 portrays the window with the selected options. The chosen heat transfer model is the constant overall heat transfer coefficient, generally represented by the letter U; although titled *alpha* (α_{ev}) within the TIL library. This approach is implemented for the fin side and the tube side. Regarding that pressure drop should be small, the common simplification of considering pressure drop zero is applied. The heat transfer is calculated through finite elements methods.

Appendix A.2 exhibits the variables and values assigned. The assigned values were either obtained from datasheets or estimated based on the test bench. Therefore, most input values were measured on-site. In the case of fin thickness, fin pitch, parallel tube distance, and tube wall thickness, the introduced number is an estimation.

3.1.3.2 Compressor

The compressor component is based on the efficient compressor model of Fösterling [67], which incorporates the rotational speed (η_{comp}), the addition of all compressor's strokes (*displacement*), and three fixed efficiencies: volumetric efficiency (η_{vol}), isentropic efficiency (η_{is}), and effective isentropic efficiency (η_{mec}). These efficiencies give an approximation to the physical phenomena that result in a deviation from ideal compression. The effective isentropic efficiency is better known as *electro-mechanical efficiency*, it pursues to weigh in the irreversibilities from mechanical frictions, magnetic effects, electrical losses, etc. Appendix A.3 depicts the displacement and efficiency inputs. For the displacement, a lesser value than the test bench's compressor is taken, given early trials with the model demonstrated higher capacities than those exhibited by the reference heat pump. In contrast, the efficiencies are taken from Klebig's [62] calculations.

3.1.3.3 Condenser

As with the evaporator, similar assumptions are taken with the condenser. The heat transfer model selected is a constant overall heat transfer coefficient, also represented by *alpha* (α_{cond}). Accordingly, the pressure drop is deemed to be negligible, for both refrigerant and water sides. Figure A.4 illustrates the previous considerations. Additionally, Figure A.5 gives a deeper insight into the geometrical values of the plate heat exchanger. For this component, a similar plate HX is taken as the reference: SWEP B8LASHx30/1P-SC-M. The number of plates, length, and width inputs are extracted from the manufacturer's data sheet, while the remaining values are the default values from TIL Suite.

Finally, it is pertinent to mention some particularities of the hydraulic circuit. The water mass flow ($\dot{m}_{\text{water}}$) has a negative value due to the boundary definitions of the model, that is, everything leaving the control volume is considered positive. The water mass flow will remain constant for all simulations, fixed at 0,1 kg/s or 6 l/min. Likewise, the outlet pressure of 100 kPa ($P_{\text{w,o}}$) will be constant across this research. Both values are considered a good approximation of real conditions for a heat pump of these characteristics.

3.1.3.4 Other components

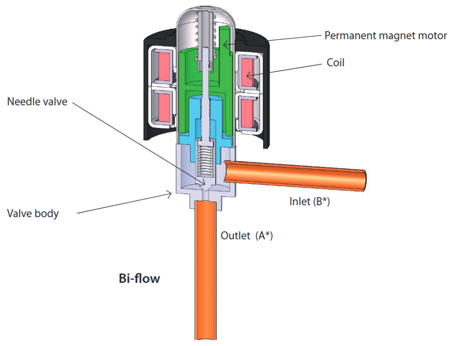


Figure 3.6: Danfoss EEV ETS 6-18 [68].

From the breakdown showed in Figure 3.4, remarks about the reservoir and expansion valve are pending. In reality, the most important attribute of a reservoir is the volume it can collect, notwithstanding, this is not accurate regarding the model. As seen on the figure, the only variables that this virtual environment considers are the refrigerant mass (m_{ref}) and the filling flow rate ($\dot{m}_{\text{fill}}$). These parameters must be addressed with caution. The refrigerant mass parameter represents the total refrigerant mass to be used within the system. Consequently, the reservoir component serves as an indicator of the refrigerant charge, not as a vapor separator or a refrigerant's reserve as its real counterpart. The filling flow rate is disregarded from this point onward, granted it is only useful for dynamic behavior studies at starts. This study will only focus on steady-state behavior.

The expansion valve component parameters are the effective flow area and the opening percentage. From the manufacturer's data sheet [68], it is known that the installed EEV has an opening range from 10 to 100% and an orifice of 1,8 mm, which results in a total opening area of 2,54 mm^2. Figure 3.6 depicts a cross-sectional view of the EEV. This figure exhibits a key aspect of this type of valve: the needle. Even though the valve is said to be at 100% opening, the orifice area will be less than 2,54 mm^2 because a portion of that surface is blocked by the needle. Further in this work, the *effective flow area* will be referenced as *opening area* (S_{op}).Table 3.2 outlines the initial parameters for the base simulation model.

Table 3.2: Summary of initial input parameters.

Component	**Parameter**	**Value**
Evaporator	α_{ev}	300 $W/(m^2 \cdot K)$
Fan	n_{fan}	15 Hz
	η_{vol}	0,7
Compressor	η_{is}	0,75
	η_{mec}	0,95
	displacement	20 cm^3
	α_{cond}	3000 $W/(m^2 \cdot K)$
Condenser	$\dot{m}_{\text{water}}$	0,1 kg/s
	$P_{\text{w,o}}$	100 kPa
Expansion Valve	S_{op}	0,2 mm^2

3.1.4 Control system development

The control system of the heat pump model consists of a superheating control and a flow temperature control. While the configuration of the parameters of the previous sections is straightforward, ensuring a minimum match (physical device – virtual environment) regarding the sensors needs a closer inspection. The piping and instrumentation scheme of the test bench (3.7) shows that important temperature sensors (represented by letter ϑ) are absent in the base model. Temperature measurements at the inlet and outlet of each component are of high importance for a faster examination; moreover, a one-to-one comparison. These will be aggregated together with the control scheme.

To make the regulation possible, the limPID component from the AixLib library [70] is incorporated. AixLib is a Modelica library of models for building performance simulations. The limPID controller allows for reverse action, this means that, for a constant set point, an increase in measurement signal x decreases the control output signal y. In other words, this scheme covers both negative and positive correlations between input and output.

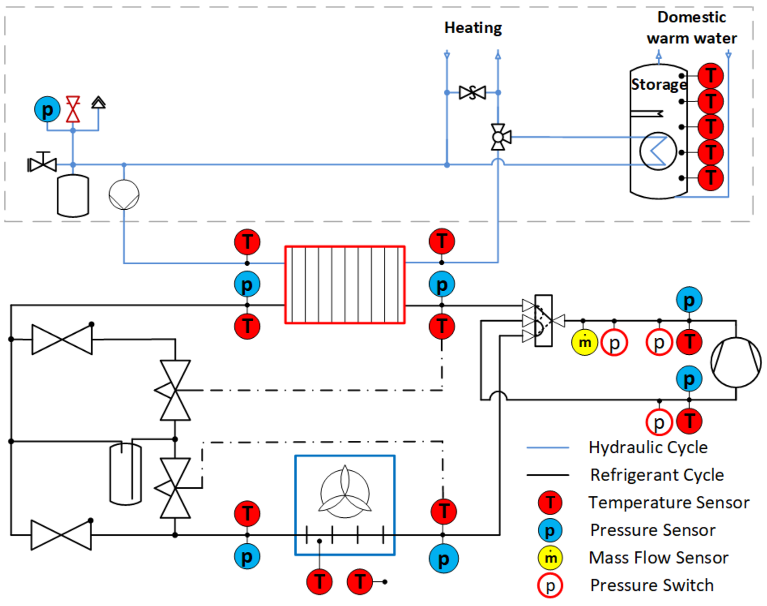


Figure 3.7: Test Bench piping and instrumentation diagram [69].

Figure 3.8 portrays the base model configuration once the *limPID* component has been added. The blue arrows belong to the inputs, while the white arrow denotes the output. The upper blue arrow is the set-point, objective value of the controller. The lower blue arrow is the input signal, providing feedback from the output signal effects. The expansion valve now resembles an EEV which opening regulates the superheat. The superheat set-point of the reference heat pump is 10°C, which is correspondingly fixed in the simulation. The compressor's speed is controlled as a function of the desired supply water temperature (outlet, $T_{\mathrm{w,o}}$). For all simulations, the set-point will be a temperature increase of $\triangle T = 5°C$ from the return water temperature (inlet, $T_{\mathrm{w,i}}$). The conjunction of the simultaneous action of both controllers results in a simplified version of a real control scheme. In real-case applications, more variables are taken into consideration for performance regulation.

As a final remark, the optimization of the proportional, integral, and derivative parameters is to be executed until stability and reduced simulation time are achieved. Strictly fast convergence is beyond the scope of this research.

3.1.5 Description of available processing capabilities

Until this point several characteristics have been laid down, to finish this section, it is adequate to indicate the computational power to be employed. Appendix A.1 presents the available capacities for the development of this study, addressing the basic hardware and software aspects to reproduce the research. Evidently, this work does not exploit any special processing capabilities.

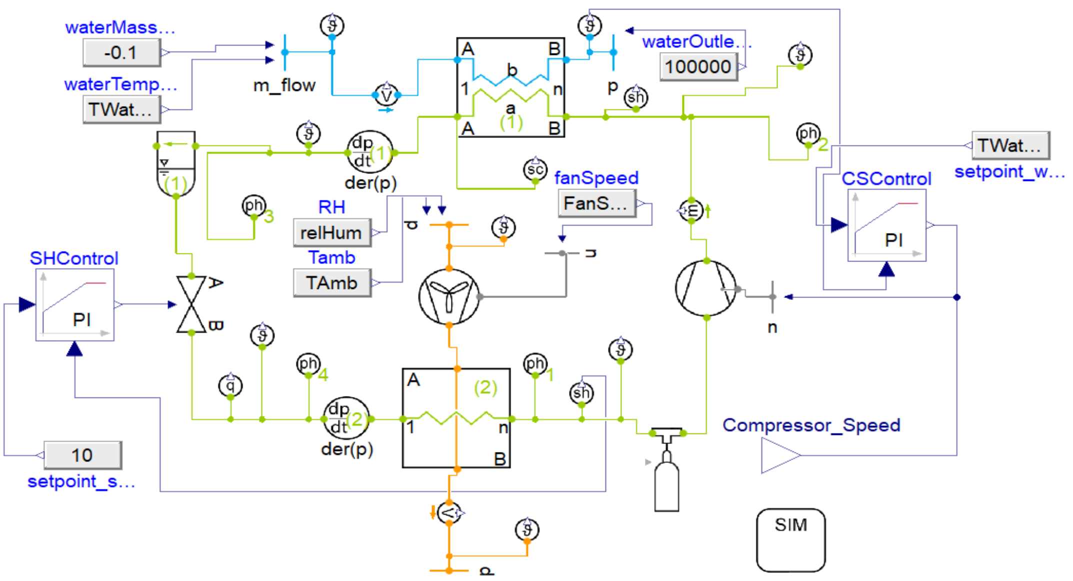


Figure 3.8: Simulation model with control scheme.

3.2 Data generation

The process of gathering information, useful to train machine learning algorithms able to detect and diagnose faults, starts with a functional simulation model. Afterward, the fault modeling approaches and time granularity of the simulations are pondered. Next, the simulation scope is clearly defined as the addition of the previous subsections, in combination with summaries of assumptions and limitations. Lastly, the evaluation of the output is detailed.

3.2.1 Fault Modeling

The described normal / faulty models method from Section 2.2.2. are combined with Sterling et al. [61] suggestion of representing faults through parameters. First, the no-fault simulation model will mimic the trends or normal operation collected by Kleipass [63], adjusting parameters for a better fit if necessary. This no-fault simulation acts as the described normal model. Thereafter, existing parameters are modified to emulate the trends from the faulty condition experiments.

Bellanco et al. [6] assembled a summary table with multiple experimental fault modeling approaches implemented throughout the literature. For fouling of a fin-and-tubes heat exchanger, the most classical methods are blocking the surface or deliberately reducing the airflow across the HX. These techniques date from Breuker and Braun (1998) [23] to Kim and Lee (2021) [22]. Likewise, Kleipass [63] used the blocking technique to reproduce the evaporator fouling. By means of three cardboards of different sizes, with heights of 22, 37, and 47,5 cm, the evaporator surface was coated from bottom to top. The cardboards were long enough to cover the longitude of the front face of the evaporator, nonetheless, this HX has an inward bend that remains unblocked.

On the other hand, the refrigerant leakage fault is emulated by extracting refrigerant from the circuit on each trial. The amount and steps vary depending on the fault impacts in which the researcher is interested. Table 3.3 resumes the methodologies implemented by this author which are relevant to this work.

Table 3.3: Outline of reference experimental data [63].

Fault type	Emulation approach	Fault levels	Fault impact range	Working conditions	Ambient temperature in °C
Evaporator Fouling	Surface blockage	4	[0, 22, 38, 48] %	[35, 45, 55, 65] °C	10
Refrigerant Leakage	Refrigerant extraction	6	[0,75 ; 1] kg	[35, 45, 55, 65] °C	10

Those trials were selected for the following reasons:

1. The fault types are aligned with those within the defined scope
2. The working conditions (supply water temperatures) cover a range translatable to most residential heating needs.
3. The current model parameters provide, at first glance, pathways to emulate these faults in similar steps.

The first fault modeling strategies consist on adjusting fan speed n_{fan} and refrigerant mass m_{ref} at the four working conditions of Table 3.3, both are considered *fault emulation parameters*. The comparison with experimental data will shed light on the following steps.

3.2.2 Extension of simulation conditions

On the previous segment it was discussed how the simulation attempts to resemble the experimental behavior, however, the available data only takes into account one ambient condition. In Section 2.1 it was discussed the major influence of the temperatures which interact with the heat exchangers. Moreover, it was detailed that performance of air-to-water heat pumps is susceptible to changes in outdoor temperatures. Therefore, limiting the ambient temperature (T_{amb}) for all simulations to the one used in the experiments constricts the exploration that this study could carry out on successive stages. As a result, other ambient temperatures are considered.

The reference location for this research is the city of Aachen, Germany (N 50° 46' 34,86" E 6° 5' 1,90"). The distribution of temperatures for the typical year of this locality, restricted to those where heating applications are more likely (under 20°C), show that the majority of temperatures lay within the [2 ; 16] °C range.

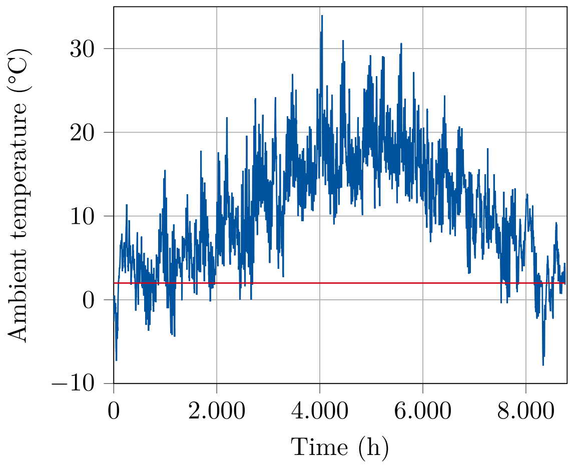


Figure 3.9: Exclusion of temperatures to avoid frost conditions.

Throughout the reviewed literature of FDD, there were scarce mentions of frost formation. An explanation could be that frost does not suffice neither hard nor soft fault definitions. Frost hinders performance once the conditions of low temperature and humidity on the evaporator's surface are met. This phenomenon does not evolve with time, nor will in an interruption of service; however, it could lead to loss of indoor comfort conditions. Yoon et al. [30] comments: "imposing faults while frosting was occurring was not considered due to the unpredictable nature of frost formation". These reasons stand behind the exclusion of frost formation and ambient temperatures below 2°C from this work. Figure 3.9 displays the frequency of temperatures to be excluded from the scope of this study.

During the rating and testing stages of air-to-water heat pumps is common to define primarily an ambient temperature and supply water temperature to assess the performance of the device; other variables such as humidity, water mass flow, or compressor speed are occasionally specified as well. Typically, these main conditions are coded as *AXX-WXX*, which reads "Air...°C - Water...°C", and referred as *experimental points*. Those points are deliberately defined by the testing and quality divisions of manufacturers under a wide variety of considerations: components, customer profile, expected performance, common complaints, frequent issues, etc. While this work does not try to replicate any industrial practice, it harnesses this subject to introduce the discussion of data distribution and representativity.

In Section 2.3.1.1, there were identified two requirements for the data: representativity and high quality. What it is intended to replicate is an approximation to the normal performance of a residential air-to-water heat pump in heating mode. The question of how to represent this does not have a clear answer or methodology in the current state of the art. For this reason, two strategies are established.

3.2.2.1 Points' matrix

Imitating the industrial tendency, combinations across 8 ambient temperatures ([2, 4, 6, 8, 10, 12, 14, 16] °C) and 4 working conditions ([35, 45, 55, 65] °C) are established. With the simulation outputs, a matrix of measurements for each code is composed. Figure 3.4 portrays an example of supply water temperature of 35°C and certain variables selected arbitrarily.

Table 3.4: Example of no-fault points matrix for one working condition.

Point	$T_{\textbf{dis}}$ **in °C**	$T_{\textbf{suc}}$ **in °C**	$T_{\textbf{SH}}$ **in °C**	$T_{\textbf{SC}}$ **in °C**	$\dot{W}$ **in kW**	$n_{\textbf{comp}}$ **in Hz**	$n_{\textbf{fan}}$ **in Hz**	$m_{\textbf{ref}}$ **in kg**
A2W35	64	2	10	8,9	0,88	49	15	1
A4W35	63	4	10	8,9	0,83	46	15	1
.	.	.	.	.	.		.	.
.	.	.	.	.	.		.	.
.	.	.	.	.	.		.	.
A16W35	58	15	10	8,8	0,56	33	15	1

For the no-fault condition, 32 samples are expected. Given these are steady-state simulations, once reached convergence, only the values from the last time step of the simulation are taken into account. For the fault scenarios, 4 fault intensities ([0, 22, 37, 48]% blockage) are defined for evaporator fouling, while 6 fault intensities are set ([0,75; 0,8; 0,85; 0,9; 0,95; 1] kg m_{ref}). Contemplating a common no-fault condition (0% blockage and 1 kg), the combined number of samples on the entire points' matrix is 286.

3.2.2.2 Typical year simulation

As mentioned, the preceding method is comparable with existing industrial practices, nonetheless, how useful this data is to train a model is to be determined. The amount of data on an experimental trial is usually larger because of higher variability from the random errors mentioned in Section 2.2.2. Still, to be valid, the results must be bounded to a specific range. This variation allows for multiple states within the same point that could be advantageous for a black-box model to have a wider spectrum to characterize a given state, instead of a binary discretization. In other words, the behavior of a point is represented by a cluster of acceptable samples, not with a yes (sample belong to the point) or no (sample does not belong to the point). Indeed, simulation models can be programmed to account for some uncertainty, however, they tend to converge to the same results under equal conditions. Thus, not accounting for any variation. The question, therefore, is the impact on the representativeness of data from that variation. To answer that question, this approach is developed as well.

According to Amasyali and El-Gohary [47] research, short-term analysis (e.g., sub-hourly, hourly, or daily) is more suitable for the operational standpoint, while long-term is preferable for strategic and projection matters. Their survey concludes a 57% of studies use an hour-by-hour granularity, a summary is presented in Figure 3.10. Similarly, the present work chooses an hourly resolution to model the steady-state behavior of a residential heat pump.

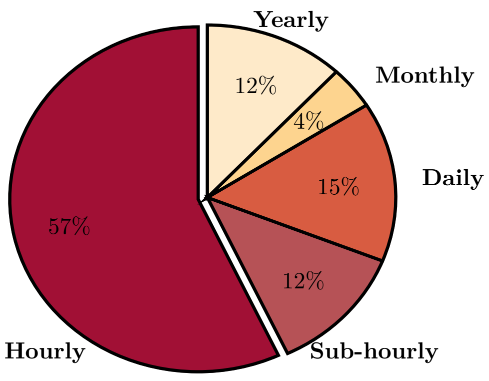


Figure 3.10: Common temporal granularities in literature [47].

Via weather data (ambient temperature and humidity) of a typical year of Aachen, simulations of one-year periods on an hourly basis are executed at several working conditions and fault severities. For the results of this virtual environment, it is presumed that, at the end of every 60 min interval, steady-state has been reached. This strategy introduces:

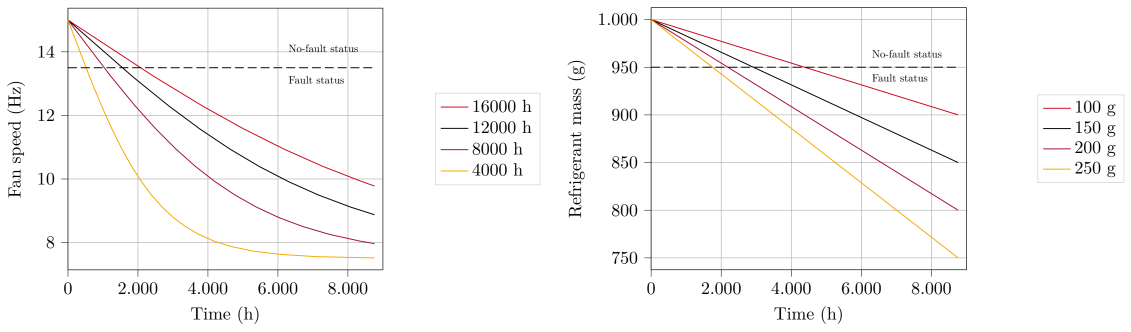
- Broad ambient temperature range.
- Uncertainty at each time step derived from the model's convergence time; necessary to adapt a response to a new input.
- Fault evolution as a function of time.

The hypothesis that sustains this methodology is that these factors help to provide a closer representation of an air-to-water heat pump's real performance. With respect to the fault progression, which is at the core of the soft fault definition, this research adopts techniques analogous to Pelella et al. [71].

For the evaporator fouling, the authors specified an amount of hours to reach 90% of a certain maximum fault intensity. Then, observing the behavior of fouling described by [28], emulated this fault through a hyperbolic tangent function. The reduction of fan speed through time is represented by the equation 3.1:

$$n_{\text{fan}} = n_{\text{fan,max}} - n_{\text{fan,min}} \cdot \tanh\left(\frac{\pi \cdot t}{2 \cdot t_{\text{max,f}}}\right) \tag{3.1}$$

Where n_{fan} is the fan speed at any hour of the typical year, $n_{\text{fan,max}}$ represents the fan speed under no-fault conditions, $n_{\text{fan,min}}$ portrays the fan speed that corresponds to the reduced airflow which allegedly emulates the desired fault intensity. t denotes the hour of the year and $t_{\text{max,f}}$ the number of hours at which the evaporator is 90% fouled, according to the selected fault intensity. Once $n_{\text{fan,min}}$ has been defined as the lowest speed that matches the maximum blockage, $t_{\text{max,f}}$ is set to be around half-year (4000 h), year (8000 h), year-and-half (12000 h), and two-years (16000 h). These four progressions are simulated for every working condition.



(a) Evaporator fouling emulation. **(b)** Refrigerant leakage emulation.

Figure 3.11: Temporal degradation of fault emulation parameters.

Concerning refrigerant leakage, the fault progression is emulated using a linear relationship between the no-fault refrigerant mass and the expected leakage throughout the year. The International Institute for Refrigeration provides a reference loss of 10% for commercial and residential equipment [33], this matches the selected values from Kleipass experiments [63]; also, it looks beyond that typical value into more severe leakages. Equation 3.2 depicts the loss of refrigerant across the year, where $m_{\text{ref,max}}$ is the no-fault refrigerant charge and $m_{\text{ref,min}}$ the remaining amount of refrigerant at the end of the year. $m_{\text{ref,min}}$ is set to be 5%, 10% , 15%, 20%, and 25%, which matches the refrigerant leakage intensities from Mehrabi and Yuill [24]. Figure 3.19 shows how the fault emulation

$$m_{\text{ref}} = m_{\text{ref,max}} - m_{\text{ref,min}} \cdot \frac{t}{8760} \tag{3.2}$$

3.2.3 Simulation model evaluation

To conclude this section, it is deemed important to highlight specific points of the simulation model for future readers to bear in mind.

1. The results disregard transient behavior. The objective supply temperature is +5°C higher than the return temperature, which is under the assumption of a constant heating power demand of 2 kW. For each working mode, the supposition is that once a steady state is reached, the return temperature is constant. This could happen in a system with a tank. Once the tank has been heated to the desired temperature, the return water

will have been previously preheated in comparison to the district supply temperatures (e.g., 20°C). These considerations are based on Ling et al. [64], which suggests that: "residential heating system is more likely to operate in steady state conditions given the prolonged use and rather constant thermal load of households".

2. The compressor's speed lower limit is 30 Hz. This speed is considered low enough to allow for control actions from the PID, simultaneously, is the smallest value to ensure stable simulations.

3. The evaporator's surface temperature is not measured in the virtual environment. The development of this feature within the simulation is neglected.

4. The fan is assumed to be fixed-speed. In real-case scenarios, fan speed is regulated to control superheat as well as the expansion valve. Hence, assessing a fault in air-flow merely from fan speed will not be as straightforward because diminished forced ventilations can be adaptations from the system.

5. The components within the simulation model are an approximation of real elements, the accuracy of this estimation is to be determined. Moreover, the existing parameters in the virtual environment could not be the best match for this use case. In the scope of the present research, the development of additional parameters or physical models is contemplated.

6. The generated data will be preponderantly about fault conditions. One possible way to conceptualize this is by visualizing no-fault conditions as a special case of fault conditions. Thus, there are more possible fault scenarios than no-fault scenarios. That consideration leads to imbalanced datasets, where more fault samples are available than no-fault. This is addressed further on.

7. The efforts are placed into ascertaining performance under winter conditions to assess heating mode performance. Although summer conditions are not neglected in the study, they are accessory. The evaluation of simulated data is focused on cold periods.

How well does the virtual environment outputs mimic the experimental trends is assessed qualitatively with the aid of the covariance matrix (Equation 3.3), which features the variance (Equation 3.5) of each variable in the main diagonal and the covariance (Equation 3.4). The variance is used as a measure of the changes of the variable through the different working conditions or fault intensities. The covariance provides a numerical representation of a trend, a positive covariance means an up-trend, while a negative indicates a down-trend; the magnitude is neglected in most of the following applications. In addition, the relationships

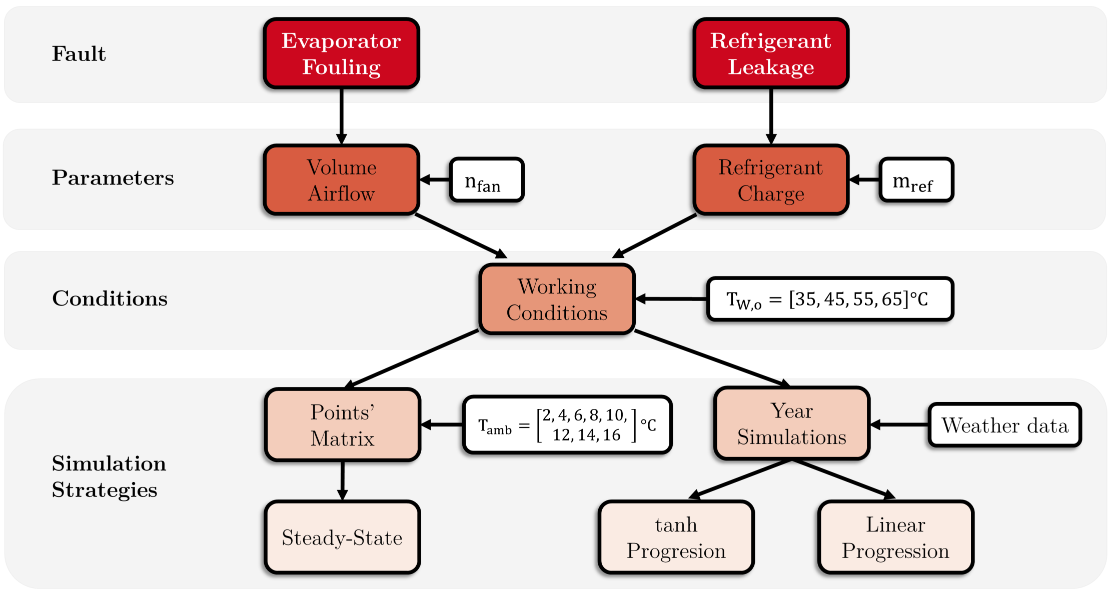


Figure 3.12: Data generation stage breakdown.

developed by [24, 25, 34] are considered to provide more context to the analysis.

$$\begin{pmatrix} Var(x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Var(y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Var(y) \end{pmatrix} \tag{3.3}$$

$$\mathrm{Cov}(x,y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1} \tag{3.4}$$

$$\mathrm{Var}(x) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1} \tag{3.5}$$

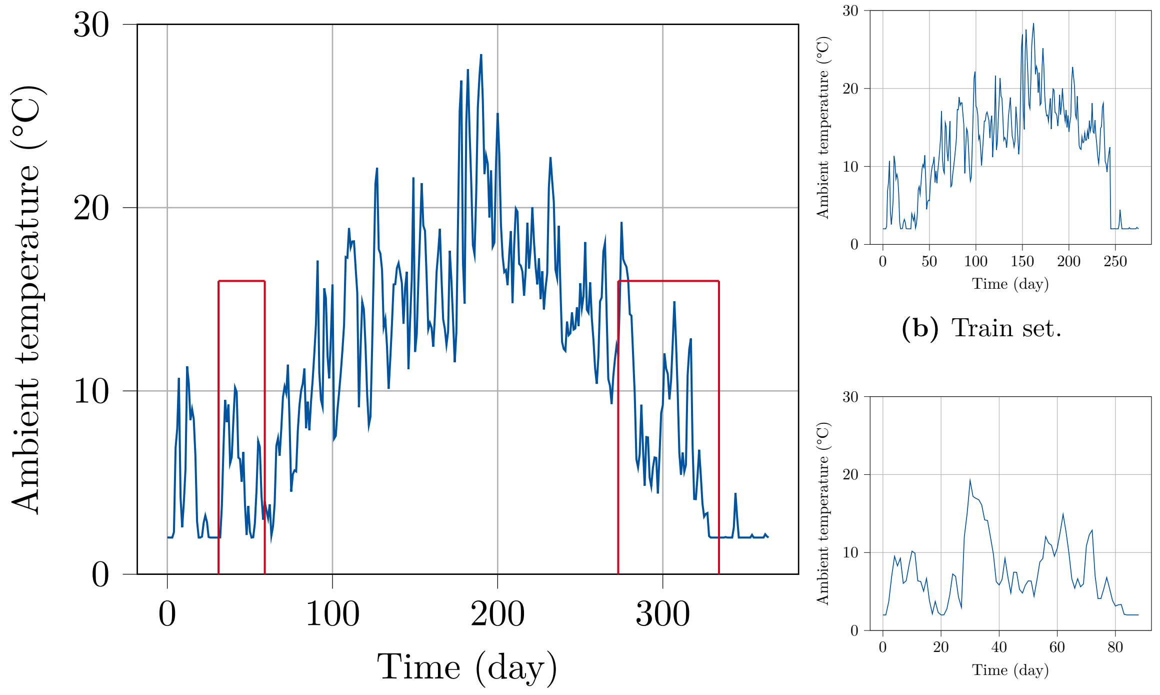
Finally, 3.12 is provided to summarize the steps described in this section.

3.3 FDD algorithm development

The aim of this research is to create a fault detection and diagnosis algorithm that can be executed at the core of a tool, on-field or online. This section details the methodology to develop such an algorithm, first, preparing the data, then prototyping and evaluating to select the most suitable. Certain arguments exposed in the previous chapter are further discussed to characterize this research approaches.

3.3.1 Data's preprocessing

Once the data is generated, usual operations to exclude non-important features and outliers are carried on. Additionally, typical year simulations demand an extra step regarding *period selection*. The modeling strategy for the fault trials establishes a starting no-fault status that degrades throughout the typical year. From the entire year, the most relevant temperatures to this work are considered to be in the range of [2 ; 16] °C according to Section 3.2.2.



(a) Typical year mean daily ambient temperature.

(b) Train set.

(c) Test set combining February and October.

Figure 3.13: Typical year period selection.

Section 2.3.1.1 elucidates on splitting the dataset to examine the generalization error from the models. A common practice is to allocate around 20 to 30% of the data in the test set. It is noteworthy to recall that both sets must accurately represent the interested phenomena because an arbitrary or random partition has the potential to avoid poor performance for data misrepresentation. A split of 75 : 25 train/test is chosen, in which the group reserved for testing comprises hours from mid-January to mid-February and October. These periods were selected under the consideration that their combination provides an adequate representation of no-fault hours and temperature variation to assess operation in winter conditions. Figure 3.13 exhibits the selected periods and the joint temperature curves, meanwhile, Figure 3.14 illustrates the resultant, where is clearly represented the testing temperature profile.

Thereafter, the datasets are merged into three different groups: one combination of all data in a single set (ALL dataset) maintaining the splits as depicted in Figure 3.16, and two subgroups organized by working condition and fault evolution. The combination set joins training and testing data by appending each subset to the side correspondent of the threshold, in Figure 3.16 right (blue) for training and left (light blue) test data. These subsets represent the same data, the objective is to ascertain if there is an optimal set that meets the performance of a model trained on all the generated data. This examination allows elucidating how different working modes and fault severities impact the MLA implementation. As a final note, the displayed representations utilize a daily granularity to deliver a tidier appearance, however, the actual datasets have an hourly granularity.

3.3.2 Feature engineering

According to Zheng and Casari [43]: "*Feature Engineering* is the process of formulating the most appropriate features given the data, the model, and the task".

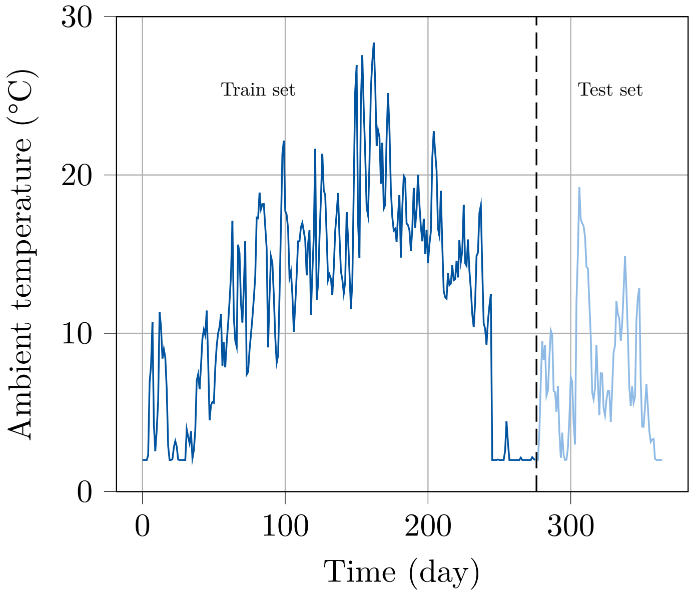
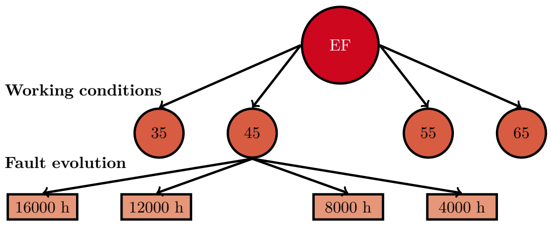
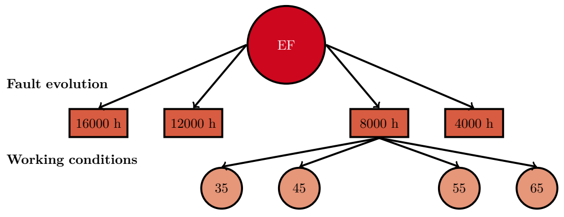


Figure 3.14: Train - test split of typical year simulations.



(a) Grouped by working condition.



(b) Grouped by fault evolution.

Figure 3.15: Datasets combination example.

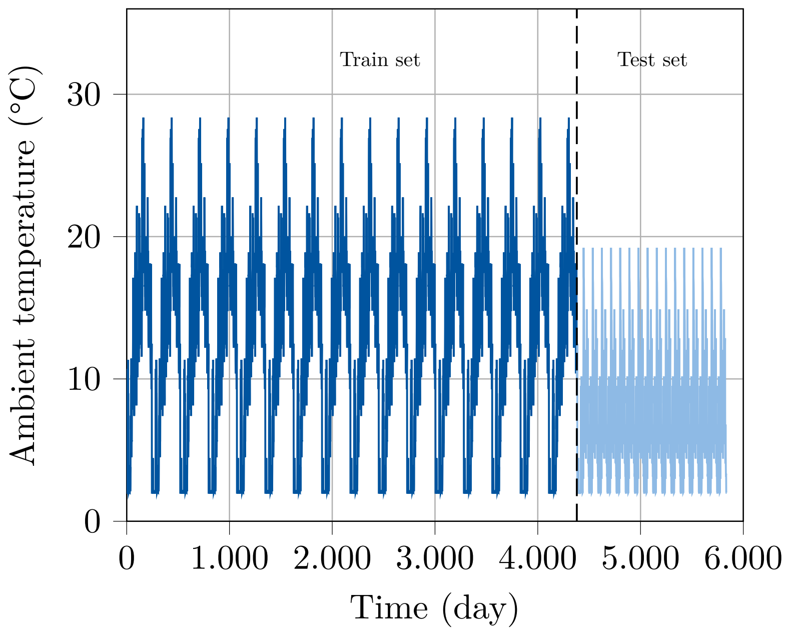


Figure 3.16: All datasets combination split.

The previous definition exposes how heavily the Feature Engineering depends on the context. This has substantial implications in fitting the algorithms to the problem. Dunning [72] refers to this procedure as the translation of domain expertise. Then, the manipulation of features can also be understood as the transmission of evident patterns (to the user) that the estimators would require more data to realize. These definitions come from authors purely dedicated to machine learning, the treatment of this step regarding the application in heat pumps and FDD could vary. The main sub-processes within feature engineering are *feature extraction* (generation of new features) and *feature selection*. By some definitions, the use of virtual sensors could be *feature extraction* strategy. They introduce variables that harness thermodynamic relations and first-principle equations. On the other hand, the techniques to extract *the best* features, first, aim to suppress redundant and non-independent variables. Afterward, on optimization steps, a new set of features could be selected testing directly the model's performance; not the relationships between the variables.

Having briefly conceptualized these techniques, it is necessary to recall that this work seeks to obtain the best performance on variables feasible in the context of a residential-size heat pump. This requirement constrains the universe of measurements, consequently, reduces the features mostly to temperatures. As a result, the features are *preselected*. Considering the implementation of virtual sensors discussed in Section 2.2.5, some features are also *extracted*. As a result, the present research does not neglect the importance of feature engineering, it still tangentially addresses certain aspects of the process.

Taking into account the cost of pressure readings and the difficulty to precisely determining thermodynamic states, two feature sets will be considered for the estimators:

- All features group: this set comprises typical variables of the vapor-compression cycle: compressor speed n_{comp}, air outlet temperature $T_{\text{a,o}}$, ambient temperature T_{amb}, condenser outlet temperature $T_{\text{cond,o}}$, discharge temperature T_{dis} , evaporator inlet temperature $T_{\text{ev,i}}$, suction temperature T_{suc}, water inlet temperature $T_{\text{w,i}}$, water outlet temperature $T_{\text{w,o}}$, condensing temperature T_{cond}, evaporating temperature T_{evap}, subcooling T_{SC}, superheating T_{SH}, power consumption sensor $\dot{W}_{\text{vs}}$, mass flow sensor $\dot{m}_{\text{ref,vs}}$, dry air volume flow sensor $\dot{V}_{\text{a,vs}}$, heating capacity sensor $\dot{Q}_{\text{h}}$, COP. It will be considered as a baseline to reference the accuracy lost from only implementing temperature sensors.
- Temperatures group: this is the ideal set, which includes inexpensive measurements: compressor speed n_{comp}, air outlet temperature $T_{\text{a,o}}$, ambient temperature T_{amb}, condenser outlet temperature $T_{\text{cond,o}}$, discharge temperature T_{dis} , evaporator inlet temperature $T_{\text{ev,i}}$, suction temperature T_{suc}, water inlet temperature $T_{\text{w,i}}$, water outlet temperature $T_{\text{w,o}}$, power consumption sensor $\dot{W}_{\text{vs}}$, heating capacity sensor $\dot{Q}_{\text{h}}$, COP. Heating capacity and COP are maintained, although their measurements could present obstacles.

3.3.3 Algorithm selection

From the well-known papers "the lack of a priori distinctions between learning algorithms" [73] and "the unreasonable effectiveness of data" [74], it is pertinent to this work to extract a few ideas.

First, how well any algorithm performs is determined by the probability that the internal logic from which it runs is aligned with the distribution that governs the looked patterns within the data [75]. That is, as phrased by Geron [41]: "without any assumption about the data, there is no reason to prefer one model over any other. There is no model that is a priori guaranteed to work better". As presented in Section 2.3.3, this work selected ubiquitous algorithms to test their performance on air-to-water residential heat pumps FDD: SVM, DT, ANN. This affirmation is based, in particular, on the reviews from Matetic et al. [16] and Amasyali and El-Gohary, where the 76% of works dealt with one or several of these algorithms [47]; as portrayed in Figure 3.17. The same supervised learning algorithms (DT, SVM, ANN) are implemented in classification and regression in this study.

Second, data by itself can lead to several answers without requiring the "perfect model". The characteristics that a dataset must comply described in Section 2.3.1.1 persist: quantity, quality and representativeness. Indeed, a *brute force* approach of using all data available may theoretically function. Notwithstanding, the burdens of using all data available in terms of extra processing power and complexity could hinder the implementation of these methods. This raises the question of which specific attributes a FDD dataset for training data-driven models needs to have. The methodology to be followed is to use all data available (features and samples included) for training at the beginning as a ground zero to, subsequently, proceed to optimization the datasets.

Following the literature predominant tendency [47, 8, 14, 16], this study selects supervised learning methods to address FDD. That opens two paths: a classification and regression.

3.3.3.1 Classifiers

Zhao et al. [8] reports a common preference to channel FDD protocols as a classification task. The discretization of results allows creating synthetic *bins* (classes), such as: *fault* or *no-Fault* in the detection phase, *evaporator fouling* or *refrigerant leakage* in the diagnostic phase. Classifiers rely heavily on the labels given to the data to properly distribute and predict the pattern of each defined class. Binary classifications are known as *single-class.* To illustrate this: fault detection is an example of single-class classification, the only possible outputs are *fault* or *no-fault*, while fault diagnosis is an example of multiclass classification because of the many possible faults. Another example is if detection and diagnostics are executed on a single step, where in most scenarios a multiclass classifier is necessary.

The criteria to label the data should be defined with caution, accounting for false alarms, sensitivity, and indication for service. Given the labels establish the classes, the labeling criteria constitute the boundary between fault and no-fault. Without the proper assessment, the output's discretization could hinder evaluating the severity of the fault. As it were an alarm indicator, the classifier will show the presence, not the magnitude. How to interpret this alarm will depend entirely on the labeling criteria. As an alternative, fault intensities

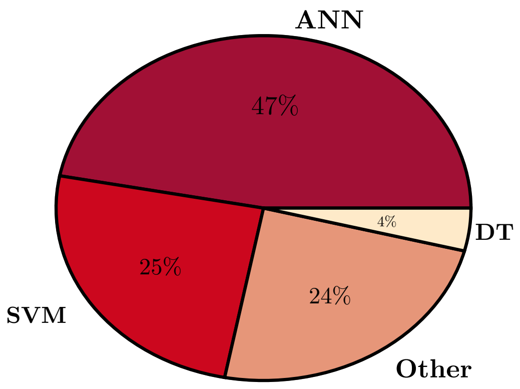


Figure 3.17: Machine learning algorithms utilization [47].

may be defined as classes. This comes with the possibility of increasing the complexity of the model or diminishing performance, among other effects.

Figure A.6 depicts the basic step-by-step to train the classifiers using a decision tree as an example. First, the necessary libraries are imported such as Numpy [76], Pandas [77], and Scikit-Learn [53].

1. Next, the dataset is loaded and split between samples (X) and labels (y), in this example the labels (Fault status) are the last column. Then, it is divided into the training set and the test set maintaining the class proportion (strata) of the original dataset. Also, the rows are shuffled to avoid any data leakage related to the progression of the faults. It is intended that the FDD algorithm detects the faults at any time, not after a progression. Considering that the data could have an important variation, a validation set is established as a sub-group from the training set. To finish the preprocessing phase, the data is standardized by subtracting the mean of each feature from every value, afterward dividing by the standard deviation. This is not necessary for the decision trees, still, it is a requirement for SVM and ANN.

2. The following lines show the code for the training, prediction, and evaluation. In the latter, the confusion matrix and balanced accuracy are calculated, together with precision, recall, and harmonic mean (f1-score) from the *classification report.* To ensure that the results are not derived from a special split between the training and validation sets, a *cross validation* is carried on with a *stratified k fold* iterator. This last function takes the input dataset, splits it into k groups (folds), trains the estimator with one, and tests with the others, based on the specified metric. The *make scorer* method transforms the balanced accuracy metric into a utility function (i.e. the greatest the better). In the end, it returns a list with the score from each fold. This list is saved along with its arithmetic mean, as the score of the preliminary training.

3. At last, a matrix specifying values of some hyperparameters is described for a rudimentary optimization. The *grid search cv* method will find the optimal combination of hyperparameters, through a specified score, obtained with a procedure similar to the previous. The best model is saved, and the evaluation is executed until an acceptable balanced accuracy is reached; if possible.

3.3.3.2 Regressors

When the effort required to label the data is pondered, having a continuous value as an output, whose interpretation is not concealed behind some criteria, is appealing. Notwithstanding, a regression task implies a trade-off: labeling effort vs *signal selection* and *threshold definition.*

The feature selected as an output, which should be directly correlated with the corresponding fault, is called *signal*. By *threshold definition* is expressed the mechanisms to detect and diagnose faults, generally, the residual between a no-fault signal and a fault signal is bounded to an interval. To illustrate this last point: a no-fault model could be developed to permanently have an indication of how the system is expected to perform, if the difference between the output and an actual measurement exceeds a limit, there is a fault. More complex protocols harness statistical methods, unsupervised learning algorithms, or other regressors, to set these boundaries [8]. As a first attempt, it is deemed logical to explore of the most simple path, once analyzed its limitations, that understanding will shed light on further methods. Therefore, a residual-threshold approach is set to develop an FDD protocol.

The process of choosing the signal must be threaded lightly. Similar to the argumentation over virtual sensors of Section 2.2.5, this feature should be insensitive to any fault other than the one it is meant to diagnose (i.e., be decoupled). In addition, the input measurements ought to be inexpensive, as well as the reference signal; although, this last point is optional. A decoupled VS could be used as a reference signal. However, the thermodynamic relationships needed to compute (some derived from pressure measurements) and then imitated by a black box model, prevent this to be the preferred approach in this work. Signals derived from measurements are conceived as a more suitable path.

A straightforward technique that assures decoupled measurements is to select the variables directly affected by faults (e.g., air volume flow and refrigerant mass), nevertheless, these evident variables are usually not simple nor affordable to measure. Kim and Lee [22] implement virtual sensors for air dry volume flow and refrigerant charge, in part, due to the obstacles to obtaining the real measurements. To circumvent these adversities: first, the signals selected are air volume flow for evaporator fouling and refrigerant mass for refrigerant leakage; second, two models on each case are developed. A reference model, trained exclusively with no-fault data, and a fault model, trained with a combination of no-fault and fault data, are generated in both cases. With these algorithms, an analysis of the residual's magnitude and statistical distribution is carried on to set the corresponding thresholds.

In Section 2.2.2, a reference to the *normal* models is made, building upon Rogers et al. [9] statements. While research points in the direction of polynomial multivariate regression for these models, this work deems that a more suitable strategy is to explore the selected estimators' capacities to their fullest. This provides methodological simplicity and thorough analysis.

Finally, similar to the classifiers, Figure A.7 exhibits the essential breakdown to train the regressors; once more, using a decision tree as an example. The steps are the same, however, now a feature is loaded in y instead of labels. Thus, they are normalized as well, compressing the values in a range of 0 to 1. Excepting the different metrics than in the preceding case, the procedure continues analogously.

3.3.3.3 Estimator assessment

Concerning the analysis of algorithms performance, the procedure employs the equations and methods of Section 2.3.4. In relation to the imbalance characteristic of the datasets, a metric that ponders equally is defined by the balanced accuracy (Equation 2.16) as the standard measure to select the better performing classifier. Naturally, a deeper analysis is needed to ascertain the performance, hence, with the confusion matrix technique 2.1 that exploration is made in combination with the rest classification metrics presented. Based on them, the selection of the classifier among all the algorithms tested with variations in training sets, features, and optimization is executed. In relation with the regressors, the two-models configuration (no-fault vs fault) disregards the imbalance. One model is trained exclusively with no-fault data, while the other with the imbalance data. To determine the best regressor combination, RMSE and R^2 (Equations 2.17 and 2.18 respectively) are considered. While RMSE denotes a measure of the distance between actual values and measurements, R^2 indicates how well the variation is emulated. The combination of both metrics gives a comprehensive understanding of performance, hence, both are taken into account indistinctively. This section concludes with the election of the highest-ranking classifier and regressor individually. Pondering which approach among the two yields the best results is done in the next section.

3.3.3.4 Estimator optimization

Over the training stage of estimators (i.e., classifiers and regressors) a basic optimization is done to preliminarily examine the performance of the algorithms. Evidently, without deeper hyperparameter tuning, asserting their aptness is premature. Therefore, Hyperopt-Sklearn (HPSKL) and ADDMo described in Section 2.3.5 are implemented for classifiers and regressors respectively.

Appendix A.8 portrays the usage of HPSKL. The example develops a classifier, although the procedure for regressors is equivalent. First, a search space is defined, for this example is displayed a multi-layer perceptron classifier with a single hidden layer, and is set to have between 1 and 1000 neurons. Second, the classifier type is specified, with certain basic hyperparameters to constrain the optimization possibilities further. Next, the *HyperoptEstimator* object is created with the search space and classifier type as input, additionally, the iterations, iteration time (referred to in the figure as *evals* and *trials*), and the number of processing cores to be used, are declared. Finally, the *fit* method is called with the training datasets, the results are evaluated as in 3.3.3.1 and 3.3.3.2.

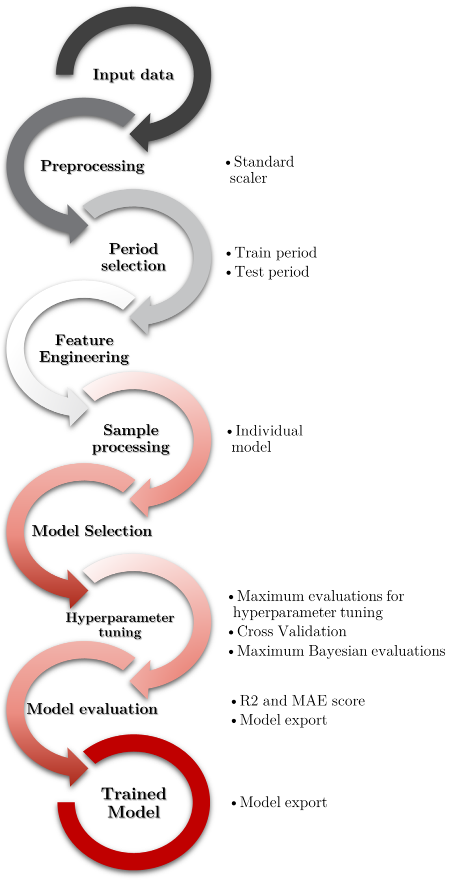
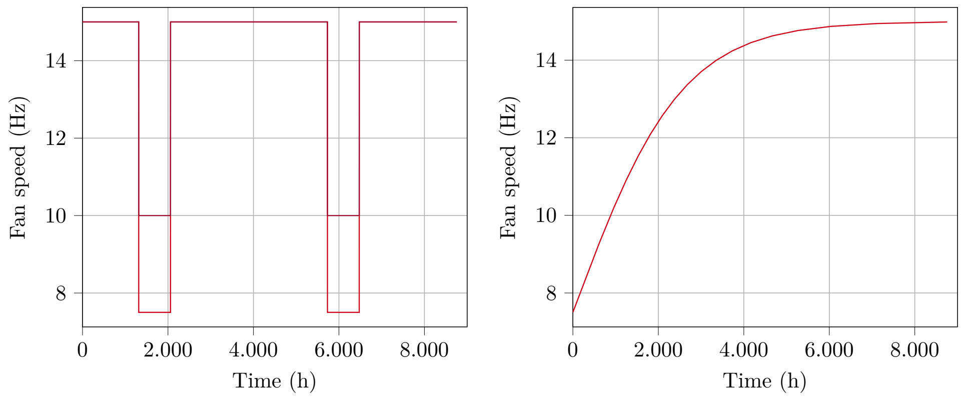


Figure 3.18: ADDMo implementation procedure.

Section 2.3.5.3 exhibits the potential and versatility of ADDMo, still, it does not support classifiers. For that reason, HPSKL, a simple implementation of the Hyperopt library [78], is used. Furthermore, HPSKL will also be used to train the no-fault models for the regression strategy. Unlike the fault models, that task is deemed too simple for ADDMo. Figure 3.18 illustrates the procedure followed. First, the signal is specified within the *Shared Variables script*, next, the *StandardScaler* method of the preprocessing is equaled to *True*.

3.3.4 FDD algorithm evaluation

Ultimately, fault detection and diagnosis protocols converge to a classification problem, where the classes are the identified faults. Nonetheless, this fact does not imply that the best strategy is a supervised-classification task. To determine the most suitable approach, several pathways are explored. Considering how broad HVAC FDD and Machine Learning methods are, distinctions are set for clarity's sake. Evidently, statistical indicators such as accuracy, precision, and recall are relatable with missed detection rate, false alarm rate, and misdiagnosis rate; all of which can be deduced from a confusion matrix. Still, it is deemed necessary to split the discussion regarding algorithms metrics, from the one that targets FDD results in order to maintain a simple thread. Thus, To examine the results in an HVAC FDD context, the nomenclature and metrics from Section 2.2.2.1 are used. With the objective of determining the best data-driven FDD algorithm, the classifier, and regressors are tested on new datasets:



(a) Medium and maximum fault imposing. **(b)** Inversion of hyperbolic tangent.

Figure 3.19: Final evaluation sets.

1. Faults at medium and high intensities are imposed individually during a year with no evolution, similar to a step-function. This evaluates the performance in a no-fault / fault / no-fault scenario. This validation seeks to evaluate whether the algorithms can detect a fault without any progression.
2. The evolution of the fault is inverted. Instead of starting at a no-fault level and progressing into a severe fault status, the year will start at a fault and progress towards no fault. The objective is to discard any learned relationship between the time of the year and FDD. Similarly, reversing the trend allows focusing the behavior of the algorithms on the minority class, if any.

4 Results and discussion

This chapter comprises the data generated by the developed virtual environment, the outcomes from training the estimators, and the core models for an FDD algorithm. In Section 4.1 the operation of the virtual heat pump is detailed for normal conditions, evaporator fouling, and refrigerant leakage. Its correspondence with the experimental trends is also discussed. Section 4.2 explores the capacities of the algorithms through data and feature combinations, classification and regression approaches are carried on sequentially, implementing findings regarding the data and estimator behavior from one step to the other. This chapter finalizes with the strategy and model selection.

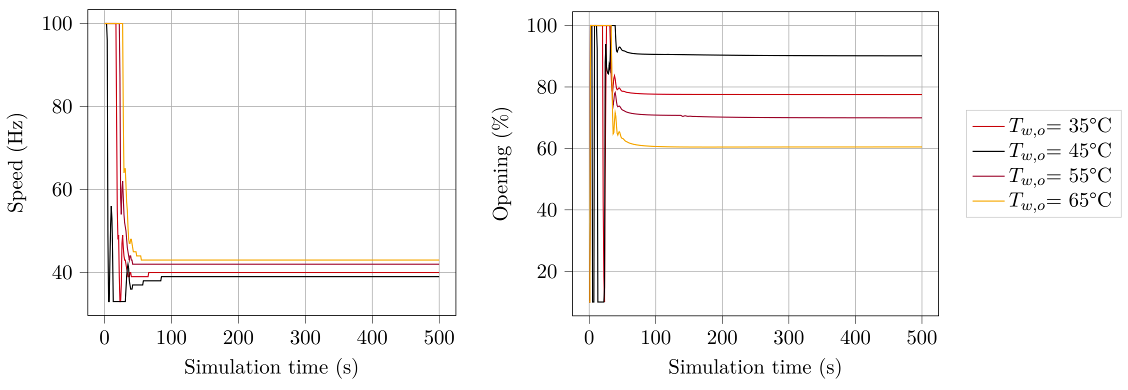
4.1 Simulation model results

The base simulation model described in Section 3.1.1 is a functional representation of a basic heat pump, featuring a fixed-orifice expansion valve and a fixed-speed compressor. Notwithstanding, a closer depiction of reality requires incorporating elements, in particular, the control system. This regulation loop is created and tested within this work, hence, it is presented as a result of this research. Next, the results of the fault modeling strategies are summarized and pondered by the covariance matrix. With the no-fault operation and fault-imposing techniques validated, the datasets are assembled based on working conditions, ambient temperatures, and fault intensities. This data is the core of the training of the algorithms.

4.1.1 Control system

The control scheme configuration is successfully carried out without sophisticated methods to set the proportional, integral, and derivative parameters. The first control loop set is the expansion valve, starting with the proportional parameter (P). Once the response is able to converge to the desired superheat in steady-state conditions in a range of 300 time units (simulation time in seconds), the integral parameter (I) is adjusted to improve stability and fasten convergence. The limPID component receives *integral time* (y_i), which is the inverse of the integral term. Subsequently, the same procedure is applied to the compressor's control loop. Since the operation of both loops executes properly in sequence, posterior re-configurations of the expansion valve control settings were not needed.

Figure 4.1 exhibits the output signals of the control schemes implemented for all working conditions. A stable output is achieved after 100 s in conditions similar to the experiments. From these results, it was deemed unnecessary to set the derivative parameter in both controllers.

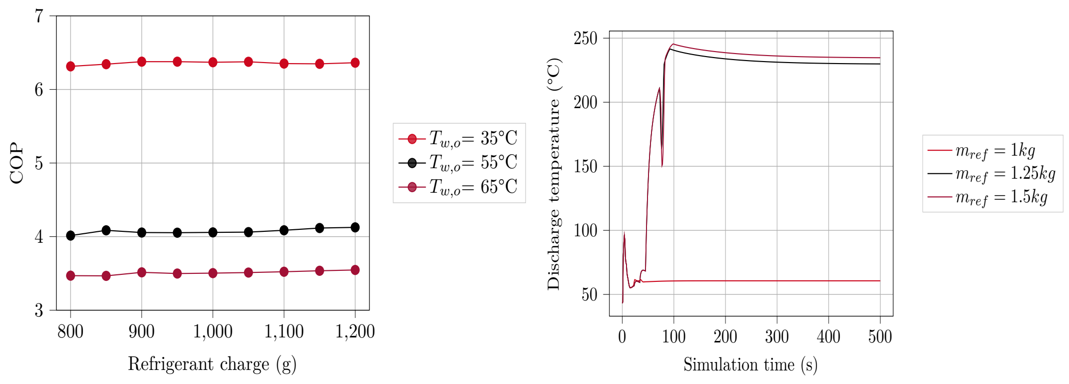


(a) Compressor's control response. **(b)** EEV's control response.

Figure 4.1: Control system response overview at $T_{\text{amb}} = 10$°C.

4.1.2 Fault modeling

In Section 3.2.2.1, the no-fault condition of the virtual environment was defined as the state in which the evaporator surface is free of any kind of blockage, while at the same time, the system is charged with 1 kg of refrigerant. Mehrabi and Yuill [24] establish two definitions of *rated refrigerant charge*: one is stated by the manufacturer, and the other is the charge which leads to the higher COP.



(a) Test bench's COP for different refrigerant charges. **(b)** Simulated T_{dis} comparison for different refrigerant charges.

Figure 4.2: COP and T_{dis} validations regarding the refrigerant charge in an experimental and a virtual environment, trials performed at $T_{\text{amb}} = 10$.

Taking into account that the reference test bench is an experimental assembly, the first definition is not applicable. Figure 4.2a shows that in a range from [0,8;1,2] kg and a condition of T_{amb}= 10 °C the COP remains almost unaltered. On the other hand, the simulations did not run properly with m_{ref} over 1 kg. This error is not displayed with an error message or warning in Dymola, however, the superheat drops to constant zero in the base model; an uncommon phenomenon in dry evaporators. Likewise, the developed model presents misleading values for discharge temperature, with excessively high temperatures (over 200°C), as depicted in Figure 4.2b. Since the scope of this work does not regard refrigerant overcharge, also, at 1 kg COP is approximately constant in a coherent range, it is established that $m_{\text{ref,rated}}$=1kg.

Whereas the refrigerant charge is set in a straightforward manner, the airflow through the evaporator has additional challenges. First, as seen in Figure 3.7, this use case does not have a reference air volume flow measurement. Second, the component in Dymola does not provide the fan's geometrical parameters as inputs. Third, the actual fan speed is defined through an analogic [0, 10] V signal, while the correspondent speed is not measured. To make up for this lack of information, the default parameters of the base model are kept: n_{fan}=15 Hz and $\dot{V}_{\text{nom}}$=1,18 m^3/s.

In the following figures, a comparison between the experiments and simulations results are displayed. The validation procedure is described in Section 3.2.3.

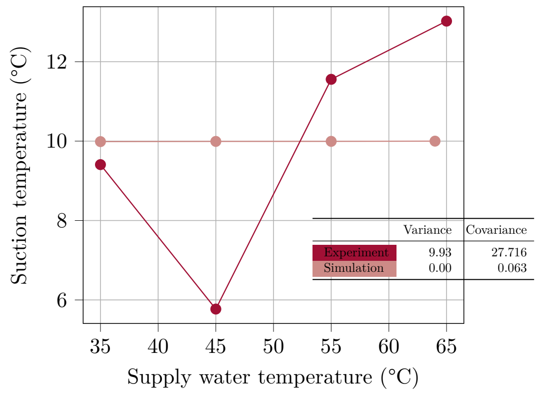
$$\begin{pmatrix} \text{Var}(T_{\text{w,o}}) & \text{Cov}(T_{\text{w,o}}, T_{\text{SH,exp}}) & \text{Cov}(T_{\text{w,o}}, T_{\text{SH,sim}}) \\ \mathbf{Cov}(T_{\text{SC,exp}}, T_{\text{w,o}}) & \mathbf{Var}(T_{\text{SC,exp}}) & \text{Cov}(T_{\text{SC,exp}}, T_{\text{SC,sim}}) \\ \mathbf{Cov}(T_{\text{SC,sim}}, T_{\text{w,o}}) & \text{Cov}(T_{\text{SC,sim}}, T_{\text{SH,exp}}) & \mathbf{Var}(T_{\text{SC,sim}}) \end{pmatrix} \tag{4.1}$$

	Variance	**Covariance**
Experiment	$\mathbf{Var}(T_{\text{SC,exp}})$	$\mathbf{Cov}(T_{\text{SC,exp}}, T_{\text{w,o}})$
Simulation	$\mathbf{Var}(T_{\text{SC,sim}})$	$\mathbf{Cov}(T_{\text{SC,sim}}, T_{\text{w,o}})$

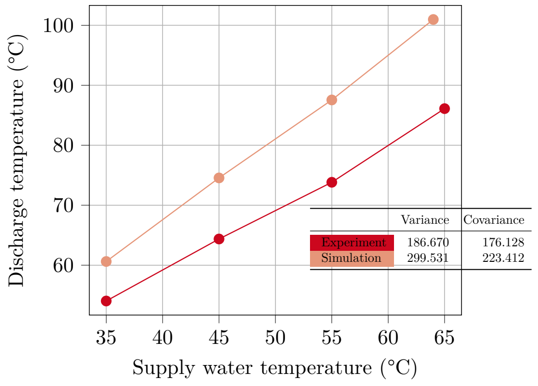
The covariance matrix (Equation 3.4) is rearranged to only exhibit in a table the variances of the analyzed features and the covariance against working condition or fault intensity (Equation 4.1. The variance of the independent variable or the covariance between experiments and simulations is disregarded.

4.1.2.1 No fault simulations

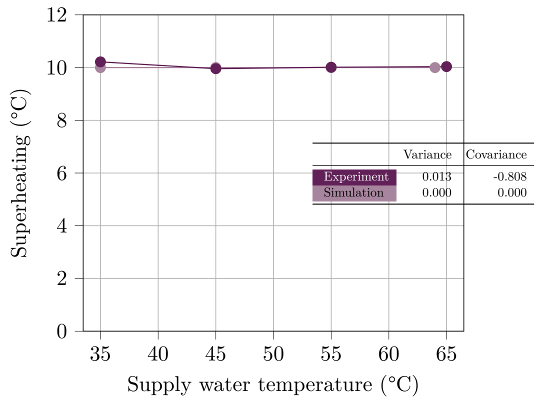
A prior step to examine the fitting of fault modeling strategies is the no-fault model resemblance of the simulations with the use case. Figure 4.3 depicts a comprehensive comparison



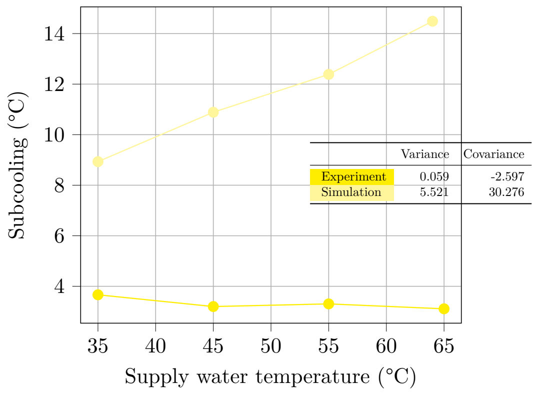
(a)



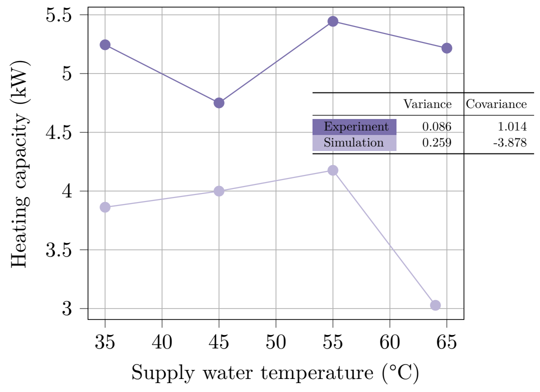
(b)



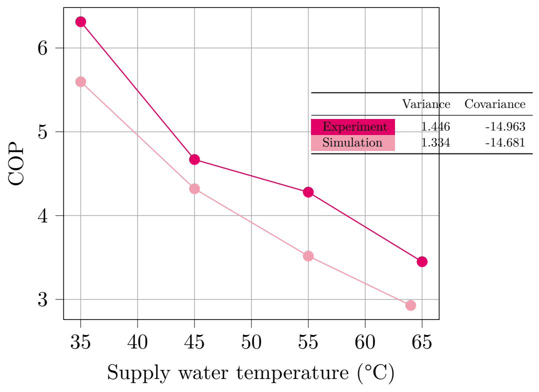
(c)



(d)

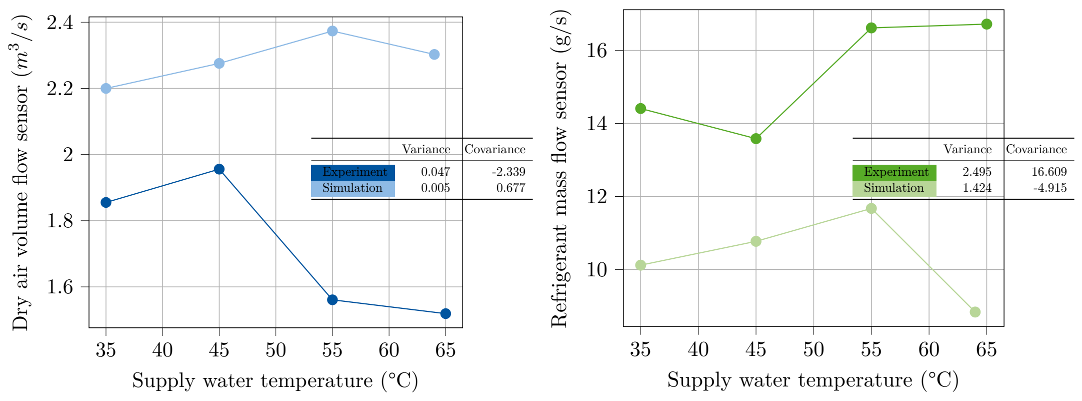


(e)



(f)

Figure 4.3: No-fault experimental-simulation comparison at $T_{\text{amb}} = 10°\text{C}$.



(g) **(h)**

Figure 4.3: No-fault experimental-simulation comparison at $T_{\text{amb}} = 10°\text{C}$ (cont.).

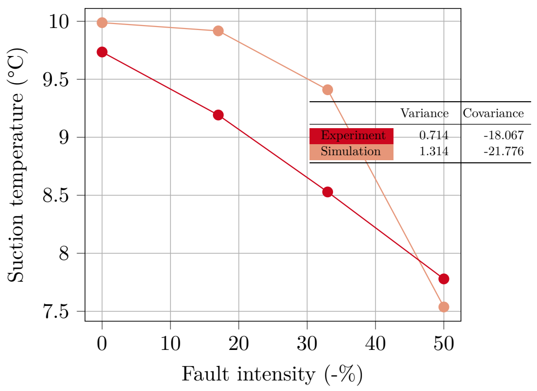
between: T_{suc}, T_{dis}, T_{SH}, T_{SC}, $\dot{Q}_{\text{h}}$, COP, $\dot{V}_{\text{a,vs}}$, and $\dot{m}_{\text{ref,vs}}$. These variables were deemed the most representative, either for their capability to represent the thermodynamic cycle or sensitivity to faults. In the majority of graphs, there is a point that breaks the tendency. Experiment features are represented by darker colors and occupy the first row of the covariance table. Regarding the data collected in the laboratory trials, the working condition $T_{\text{w,o}}$ = 45°C is not aligned in several variables with the other conditions. In the same way, the simulated features have an out-of-trend working condition $T_{\text{w,o}}$ = 65°C. Table 4.1 condenses the information of Figure 4.3, and resumes the comparison in this state. The qualitative analysis results show that the no-fault simulation is considered appropriate for this work. As last keynotes: the heating capacity and refrigerant mass flow tendencies ($\dot{Q}_{\text{h}}$ and $\dot{m}_{\text{ref,vs}}$) for the simulation are deemed uptrend after considering the last working condition an outlier.

4.1.2.2 Evaporator fouling

The strategy to model this fault began with the reduction of the fan speed to match a specific blockage step. First the boundaries were fixed (no-fault / maximum fault), next the middle values. Early in the process, it became evident that merely the decrease of air velocity through the evaporator would not have enough impact. The hypothesis that completes the fouling approach is that the overall heat transfer coefficient (α_{ev}) is attenuated with the lessened

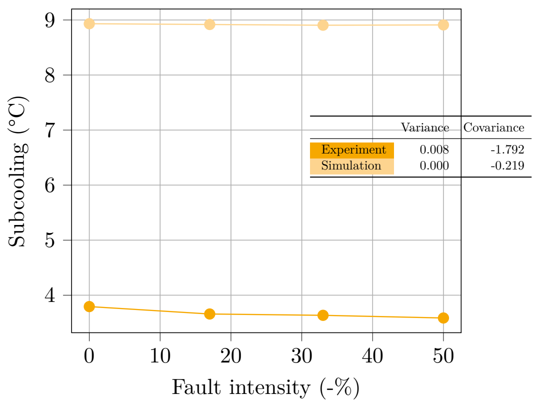
Table 4.1: No-fault state comparison

	T_{suc}	T_{dis}	T_{SH}	T_{SC}	$\dot{Q}_{\text{h}}$	COP	$\dot{V}_{\text{a,vs}}$	$\dot{m}_{\text{ref,vs}}$
Experimental	↑	↑	–	–	↑	↓	↓	↑
Simulation	–	↑	–	↑	↑	↓	–	↑
Result	×	√	√	×	√	√	×	√



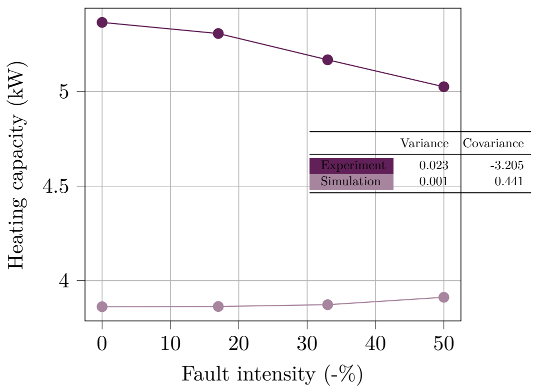
	Variance	Covariance
Experiment	0.714	-18.067
Simulation	1.314	-21.776

(a)



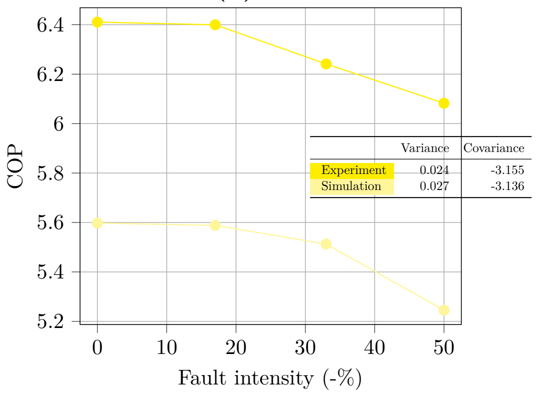
	Variance	Covariance
Experiment	0.008	-1.792
Simulation	0.000	-0.219

(b)



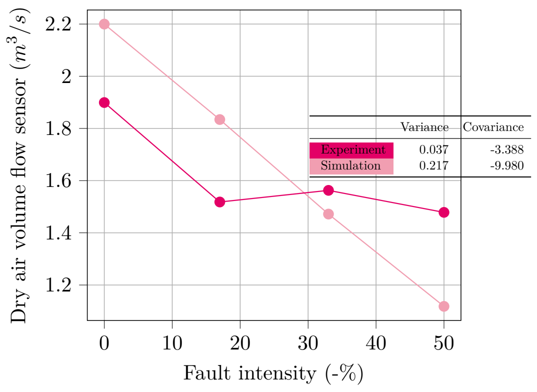
	Variance	Covariance
Experiment	0.023	-3.205
Simulation	0.001	0.441

(c)



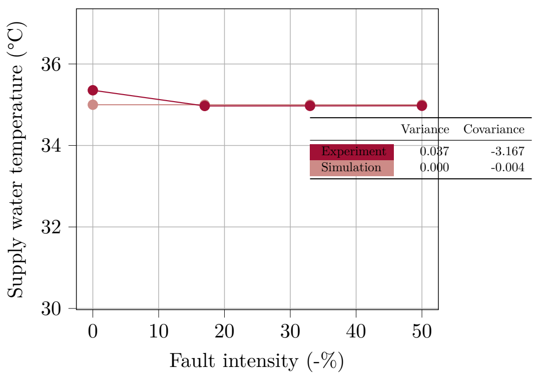
	Variance	Covariance
Experiment	0.024	-3.155
Simulation	0.027	-3.136

(d)



	Variance	Covariance
Experiment	0.037	-3.388
Simulation	0.217	-9.980

(e)



	Variance	Covariance
Experiment	0.037	-3.167
Simulation	0.000	-0.004

(f)

Figure 4.4: Evaporator fouling experimental-simulation comparison at A10W35.

Table 4.2: Evaporator fouling equivalences and methods.

Blockage in %	**Fan Speed in Hz**	**Heat transfer coefficient in $W/(m^2.K)$**	**Air Volume Flow in m^3/s**	**Fault Intensity in %**
0	15	300	1,2	0
22	12,5	250	0,98	-17
37	10	200	0,78	-34
48	7,5	150	0,58	-50

airflow.

More information on the effects of fouling on the air-side heat transfer can be found in the research from Bell et al. [79]. As a result, a combination of decrements in n_{speed} and α_{ev} are implemented to imitate the effects of blocking the evaporator. Table 4.2 summarizes the fault progression, in addition, describes the parities established between different variables. All of these features are used indistinctly to represent evaporator fouling. The air volume flow ($\dot{V}_{\text{a}}$) from this table should not be mistaken by the dry air volume flow virtual sensor ($\dot{V}_{\text{a,vs}}$).

Table 4.3: Evaporator fouling trends comparison

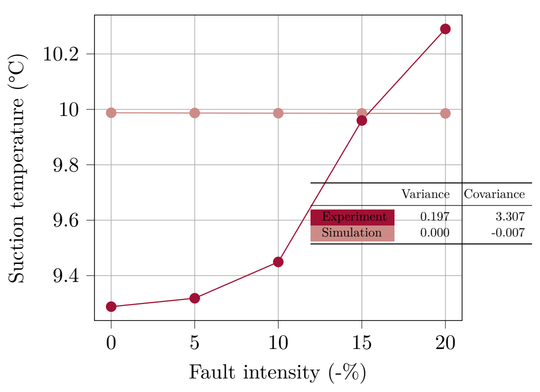
	$T_{\mathbf{suc}}$	$T_{\mathbf{SC}}$	$\dot{Q}_{\mathbf{h}}$	COP	$\dot{V}_{\mathbf{a,vs}}$	$T_{\mathbf{w,o}}$
Experimental	$\downarrow$	–	$\downarrow$	$\downarrow$	$\downarrow$	–
Simulation	$\downarrow$	–	–	$\downarrow$	$\downarrow$	–
Result	$\surd$	$\surd$	$\times$	$\surd$	$\surd$	$\surd$

Figure 4.4 displays the behavior followed by experiments and simulations on the most affected parameters. The discharge temperature (T_{dis}) is not represented because it lacks significant variation; the same applies for the refrigerant mass flow ($\dot{m}_{\text{ref,vs}}$). These results present a closer fit than the no-fault conditions, this is evident by the similar values of variance and/or covariance in each variable.

Table 4.3 resumes the tendencies and evaluates each feature, as done in the previous section. The slight uptrend in the heating capacity ($\dot{Q}_{\text{h}}$) for the simulations is neglected, still, the supply water temperature is validated to show that the demand is covered. These outcomes prove that there is an adequate match between experimental trends and simulations for evaporator fouling.

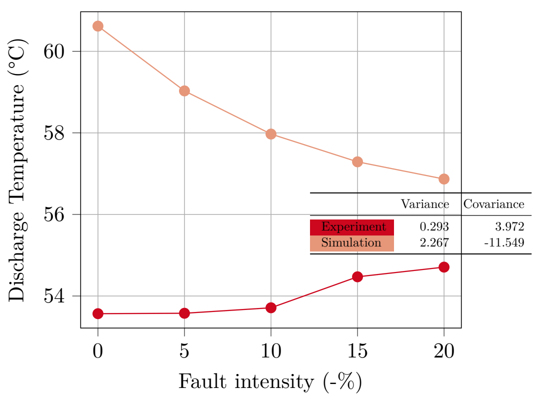
4.1.2.3 Refrigerant leakage

The refrigerant leakage simulation variables depict opposite or non-correlatable trends from their experimental counterpart, as observable in Figure 4.5. The absence of matching in the majority of relevant features derives from the exclusion of this fault from the research.



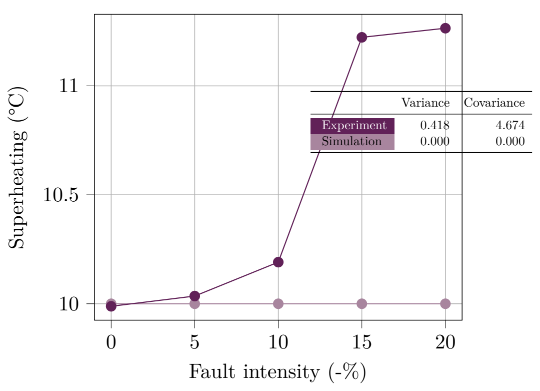
	Variance	Covariance
Experiment	0.197	3.307
Simulation	0.000	-0.007

(a)



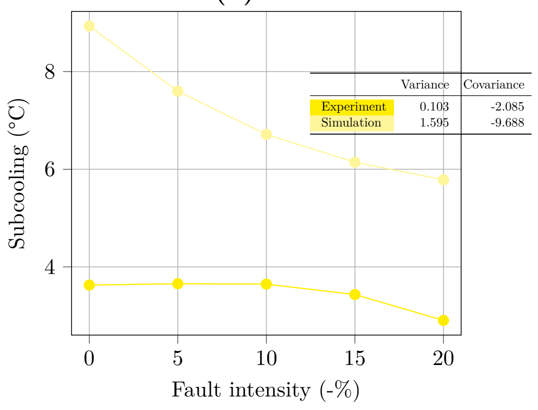
	Variance	Covariance
Experiment	0.293	3.972
Simulation	2.267	-11.549

(b)



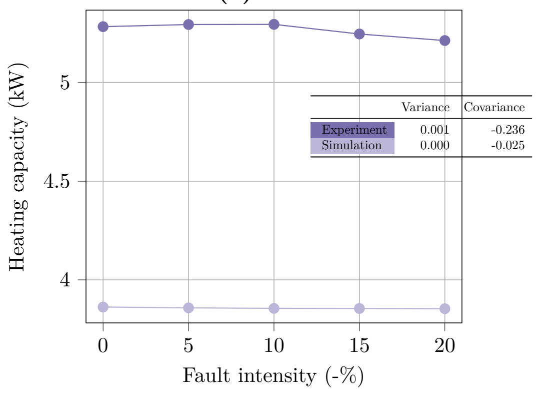
	Variance	Covariance
Experiment	0.418	4.674
Simulation	0.000	0.000

(c)



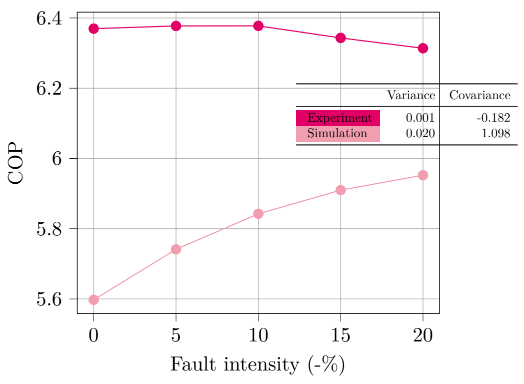
	Variance	Covariance
Experiment	0.103	-2.085
Simulation	1.595	-9.688

(d)



	Variance	Covariance
Experiment	0.001	-0.236
Simulation	0.000	-0.025

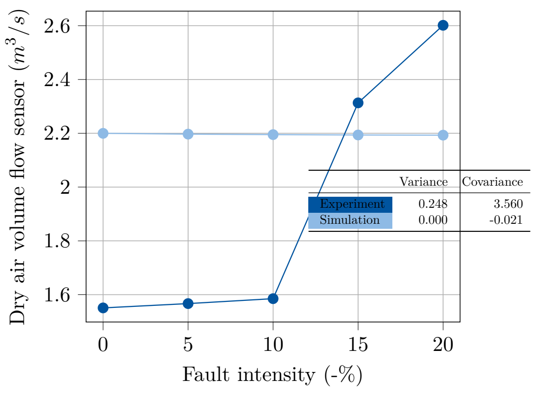
(e)



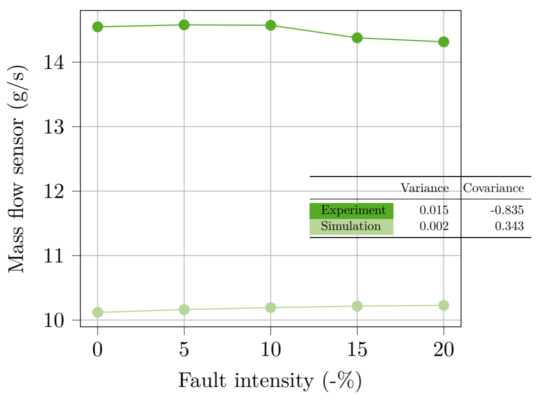
	Variance	Covariance
Experiment	0.001	-0.182
Simulation	0.020	1.098

(f)

Figure 4.5: Refrigerant leakage experimental-simulation comparison at A10W35.



(g)



(h)

Figure 4.5: Refrigerant leakage experimental-simulation comparison at A10W35 (cont.).

The exploration of the virtual environment did not yield an alternative approach that, in combination with the refrigerant mass reduction, mimicked the effects of leakage maintaining the physical coherence of the parameters; unlike the previous case. Table 4.4 compares the features and provides a simple inspection of the criterion to the rejection of these simulations.

4.1.3 Datasets

The preceding results lead to a reduction of the projected samples in Section 3.2.2. From this point onward, the focus relies on evaporator fouling data. The first subsection, which outlines a starting approach to training strategies, deals with the establishment of a fault threshold for classifiers. These criteria are maintained for the typical year simulation and the rest of this work. The discrete variable *Fault State* ($F.S$) takes a dummy value of "0" to represent a no-fault status, and "1" to denote evaporator fouling.

4.1.3.1 Points Matrix

The Points Matrix is the combination of simulations at different ambient temperatures, with multiple working conditions, and diverse fault intensities, in steady-state. To set up this database, a characterization is necessary to split the data into "fault" and "no-fault". This

Table 4.4: Refrigerant leakage state comparison

	$T_{\mathbf{suc}}$	$T_{\mathbf{dis}}$	$T_{\mathbf{SH}}$	$T_{\mathbf{SC}}$	$\dot{Q}_{\mathbf{h}}$	COP	$\dot{V}_{\mathbf{a,vs}}$	$\dot{m}_{\mathbf{ref,vs}}$
Experimental	–	↑	↑	↓	–	–	↑	–
Simulation	–	↓	–	↓	–	↑	–	–
Result	√	×	×	√	√	×	×	√

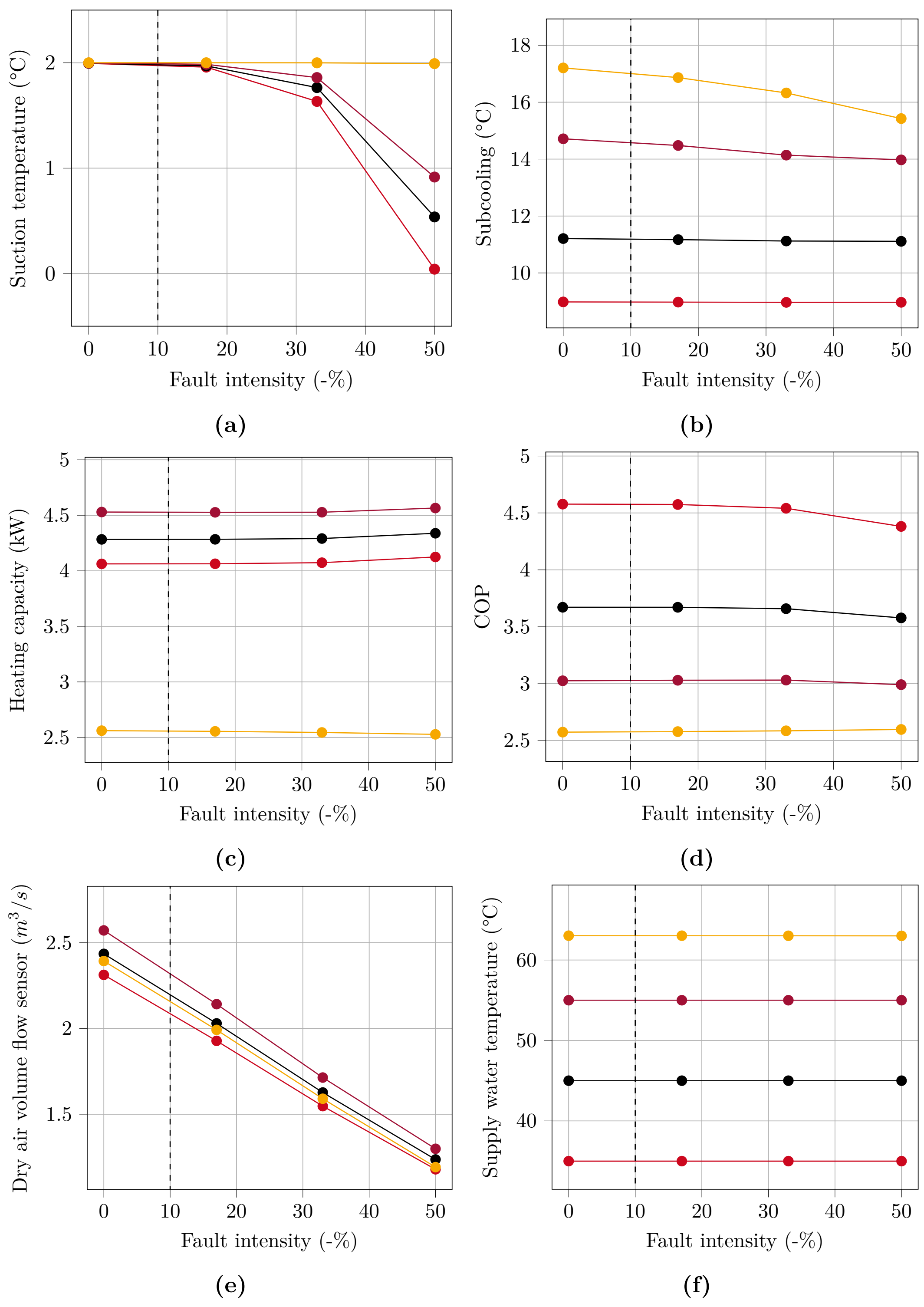
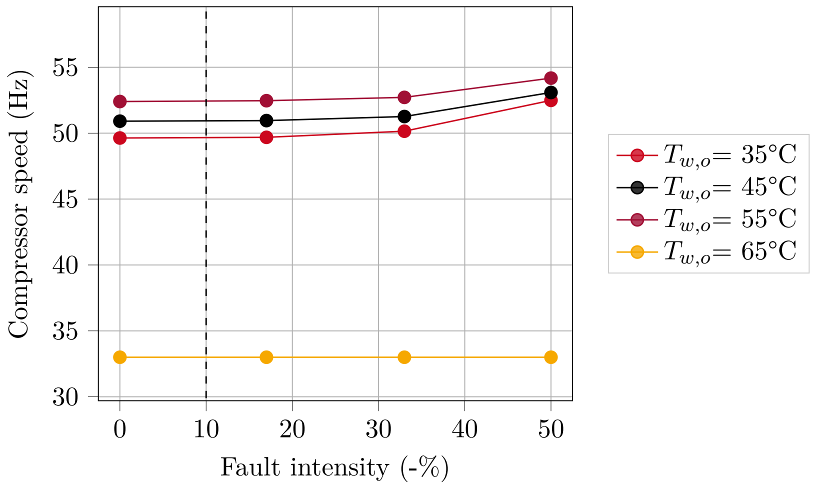


Figure 4.6: Evaporator fouling effects summary at $T_{\text{amb}} = 2$ °C.



(g)

Figure 4.6: Evaporator fouling effects summary at T_{amb} = 2 °C (cont).

must be done under strict criteria in order for the algorithms to find patterns and emulate them. For this purpose, the simulations are plotted to evaluate the behavior of the virtual heat pump in all the working conditions at specific variables.

Figure 4.6 and Appendix B.6 portray T_{amb}= 2 °C and T_{amb}= 16 °C, the coldest and warmest ambient temperature conditions tested, respectively. This figure is displayed to exhibit the effects of fouling at the most critical condition simulated. It is noticeable a significant decrease of suction temperature (Figure 4.6a) and subcooling (Figure 4.6b). Moreover, the dry air volume flow virtual sensor (Figure 4.6e) depicts a clear linear downtrend, which could be useful for detecting this fault. Regarding the simulated working conditions, this figure also reveals a particular behavior for $T_{\text{w,o}}$= 65 °C (represented by color yellow). The large decrease of heating capacity (Figure 4.6c) and the low compressor's speed (4.6g) are indicators of abnormalities within the model for this work condition. Comparing, for example, with °C. $T_{\text{w,o}}$= 45 °C and $T_{\text{w,o}}$= 55 °C, more typical pattern for all variables is distinguishable. It is possible that this occurs because they are the middle working conditions, hence, the adjustments done to assure a better match favor these conditions than others. Despite this, $T_{\text{w,o}}$= 65 °C is still considered for training the algorithms. Its values could portray a machine under a high strain, and the variation of its results could benefit the generalization of the algorithms.

The subfigures of Figure 4.6 have a dashed line at 10% fault intensity that sets the threshold to separate the no-fault and fault states. Any operation where a fault of an effect below 0,12 m^3/s is present, is considered normal. From the aforementioned figures, it is clear that this criterion is inclined to high sensitivity, given its aims to address the fault before any significant degradation has taken place. With this split, the *Fault State*($F.S$) feature is set.

Table 4.5 illustrates a subset of the Points Matrix, grouped by working condition $T_{\mathrm{w,o}}$=35°C. There are displayed three ambient temperatures considered coldest, middle and warmest, along with features previously introduced. In spite, the air outlet temperature from the evaporator ($T_{\mathrm{a,o}}$) and the power consumption ($\dot{W}$) are presented for the first time. From this dataset, the most relevant features are selected for training the algorithms. The last columns: m_{ref}, n_{fan}, and $F.S$ are depicted for reference purposes, this data must not be leaked to the algorithms. The fault state variable represents the signal for these algorithms, in the case of the classifiers, it denotes the classes. The performance of the algorithms trained with the Points' Matrix is described in Appendix B.1. The features utilized are defined in Section 3.3.2.

Table 4.5: Points' matrix example for working condition $T_{\mathrm{w,o}}$=35°C.

	T_{amb}	n_{comp}	$T_{\mathrm{a,o}}$	T_{dis}	$T_{\mathrm{ev,i}}$	T_{suc}	T_{SC}	T_{SH}	$\dot{W}$	$\dot{m}_{\mathrm{ref,vs}}$	$\dot{V}_{\mathrm{a,vs}}$	COP	m_{ref}	n_{fan}	F.S
0	16	33,91	14,79	58,31	5,98	15,98	8,87	10,0	0,57	10,06	2,16	6,70	1000	15,0	0
1	16	34,01	14,54	58,32	5,87	15,87	8,84	10,0	0,57	10,06	1,80	6,68	1000	12,5	1
2	16	34,59	14,19	58,52	5,23	15,23	8,82	10,0	0,58	10,06	1,44	6,55	1000	10,0	1
3	16	36,53	13,62	59,30	3,16	13,16	8,83	10,0	0,62	10,06	1,09	6,14	1000	7,5	1
4	10	39,76	8,85	60,62	-0,01	9,99	8,93	10,0	0,69	10,12	2,20	5,60	1000	15,0	0
5	10	39,84	8,62	60,63	-0,08	9,92	8,92	10,0	0,69	10,12	1,83	5,59	1000	12,5	1
6	10	40,39	8,28	60,81	-0,59	9,41	8,90	10,0	0,70	10,14	1,47	5,51	1000	10,0	1
7	10	42,51	7,74	61,57	-2,46	7,54	8,91	10,0	0,75	10,20	1,12	5,25	1000	7,5	1
8	2	49,63	0,94	64,00	-8,01	1,99	8,98	10,0	0,89	10,45	2,31	4,58	1000	15,0	0
9	2	49,68	0,72	64,01	-8,04	1,96	8,98	10,0	0,89	10,45	1,93	4,57	1000	12,5	1
10	2	50,15	0,41	64,14	-8,37	1,63	8,97	10,0	0,90	10,47	1,55	4,54	1000	10,0	1
11	2	52,49	-0,09	64,86	-9,96	0,04	8,97	10,0	0,94	10,56	1,18	4,38	1000	7,5	1

An overview of the base code used to train and test the algorithms is shown in Appendix A.6. The first evaluated group was the Decision Trees (DT), these algorithms have an interesting attribute called *feature_importances_*. From all the features introduced, the DTs carry a selection and classify over those features. This allows to have a sense of the most important variables, which can guide further preprocessing steps, such as deleting features of low importance. Nonetheless, each algorithm is different and could not perform appropriately with those features; i.e., perhaps only relevant to DTs.

To train each algorithm, the datasets are shuffled and split into train, validation, and test set. For the last set, the balanced accuracy is calculated and reported. It is expected a decline in performance when the algorithms are exposed to new data such as that derived from typical year simulations or experiments, thus, only those algorithms with scores over 0,7 are optimized and tested with these sets. The optimizations are performed with HPSKL, except the ANNs, which are trained directly because of their large amount of hyperparameters. In some cases, the algorithms are retrained to match the available experimental features; where, for example, air outlet temperature is not measured.

On the whole, the algorithms trained with the Points Matrix failed to provide better than random chance scores on typical year and experimental sets. Notwithstanding, important lessons are drawn:

- Decision trees are oversimplifying the patterns, in cases only taking one parameter. Regarding this underperformance, ensemble methods are used instead in the next steps.
- The similar performance of SVMs and ANNs demonstrates that the data does not have enough quality to generalize to other datasets. This is a motivation to continue with the typical year data strategy.

4.1.3.2 Typical year simulations

These simulations are built on the lessons learned with the steady-state results that compose the Points Matrix. To address the time-dependent variations of the typical year weather data, an element from the Modelica Standard Library [60] was added to the model from Figure 3.8. The final virtual environment highlights the component *Combitimetable*. Through a fixed arrangement of input variables for each hour of the year, the Combitimetable controls the simulation and provides the desired variations. This component reads a tabular array that contains: time (t (h)), ambient temperature (T_{amb} (K)), humidity (RH (%)), water return temperature set point ($T_{\text{w,i}}$ (K)), water supply temperature set point ($T_{\text{w,o}}$ (C)), superheat set point (T_{SH} (K)), fan speed (n_{fan} (Hz)), refrigerant charge (m_{ref} (kg)), and evaporator's overall heat transfer coefficient (α_{ev} ($W/(m^2 \cdot K)$)), then connects each feature to the corresponding component, as illustrated by Figure 4.7. The Combitimetable goes over the array row-by-row, hence, each component receives a value in each time step. To account for the simulation convergence time, the first sample of each simulation is repeated over 100 simulation seconds or rows of the Combitimetable, hence, the simulations are of 8860 time intervals. These first 100 samples are discarded on the preprocessing stage.

For the progression of fouling in the evaporator, Equation 3.1 is rewritten as:

$$n_{\text{fan}} = 15 - 7,5 \cdot \tanh\left(\frac{\pi \cdot t}{2 \cdot t_{\text{max,f}}}\right) \implies t_{\text{max,f}} \in \{4.000, 8.000, 12.000, 16.000\}h \tag{4.2}$$

$$\alpha_{\text{ev}} = 300 - 150 \cdot \tanh\left(\frac{\pi \cdot t}{2 \cdot t_{\text{max,f}}}\right) \implies t_{\text{max,f}} \in \{4.000, 8.000, 12.000, 16.000\}h \tag{4.3}$$

Figure 4.8 presents the results from $T_{\text{w,o}}$=35°C at the softest degradation ($t_{\text{max,f}}$=16.000 h). In Figure 4.8a is distinguishable the fault threshold of 10%, however, the effects are not clearly observable given the ambient temperature variations that affect performance; a common phenomenon that occurs on these devices. From this figure, it is noteworthy the high variations that the dry air volume flow virtual sensor experiences. In the Points Matrix,

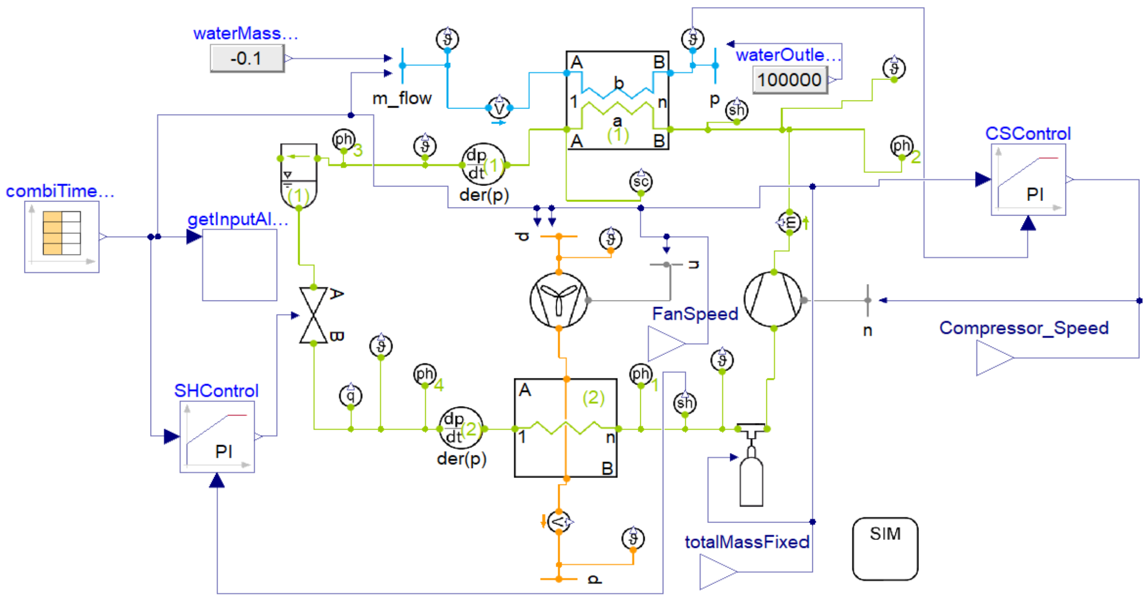


Figure 4.7: Simulation model for typical year simulations.

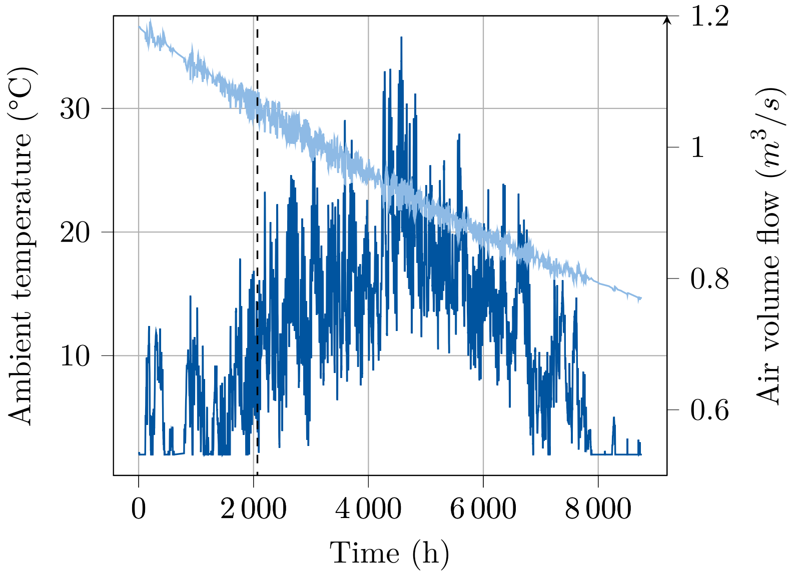
this feature seemed to provide an evident indication of evaporator fouling, nevertheless in these conditions, it fails to provide a clear correlation.

4.2 Fault Detection and Diagnosis

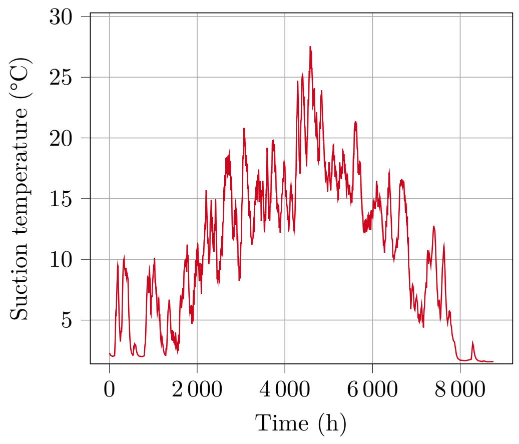
At this stage, quality datasets have been generated under several assumptions in order to train algorithms capable of discerning between a normal operation and a machine working with a fouled evaporator. First, low-variation data (Points Matrix) is examined, which builds foundational knowledge for the high-variation data (Typical year). Repurposing certain data generated, the multiclass capabilities of the classifiers are additionally evaluated. Finally, the regressors are created, optimized, and set for a final evaluation, fully under the scope of FDD metrics. This phase establishes the base ground for the discussion of the FDD algorithm, assessing strategies, assumptions, and techniques with their outcomes.

4.2.1 Single class classifiers

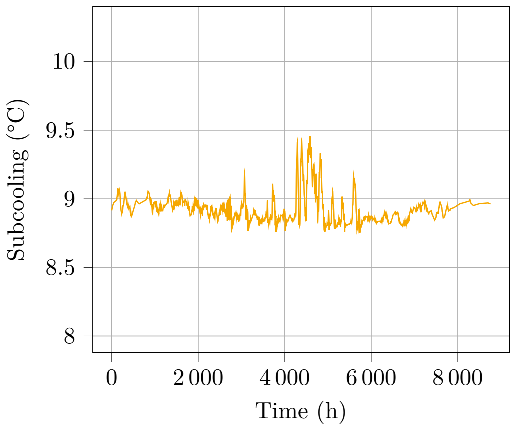
Figures from 4.9 and 4.10 exhibit the scores of the classifiers according to Section 3.3.1, the base format for these tables is given by Figure 4.9a. Recalling Figure 3.14, where representative fractions of the data are divided are displayed, if each fraction is treated as individual sets, it is possible to organize a training-testing scheme. The purpose is to try to find conditions where those fractions are more effective. This provides enhanced context to the performance analysis. The vertical axis of Figure 4.9a stacks the testing fractions, from top to bottom,



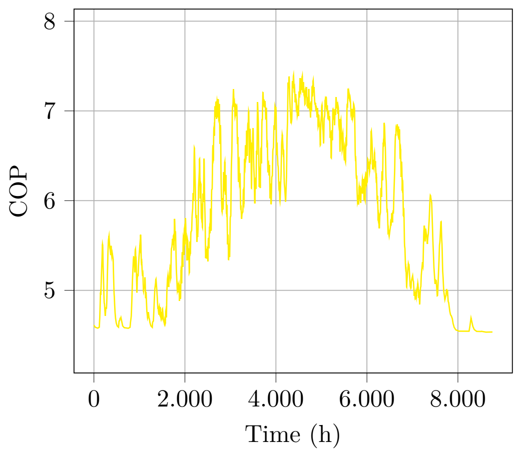
(a) Where the ambient temperature is depicted with blue and the air volume flow with light blue.



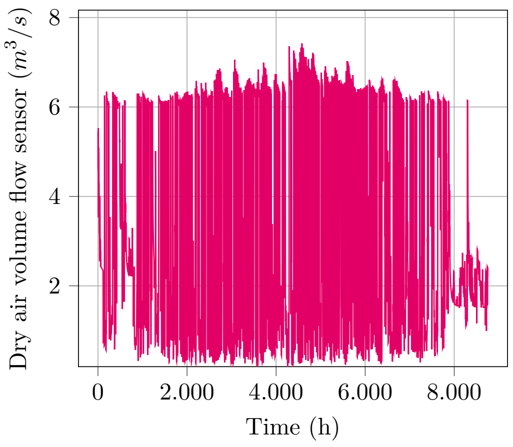
(b)



(c)

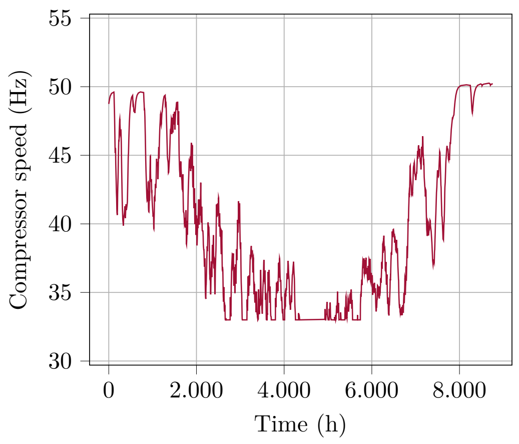


(d)



(e)

Figure 4.8: Evaporator fouling effects summary at $T_{\mathrm{w,o}} = 35$ °C on a typical year simulation with 16.000 h to max fault EF.



(f)

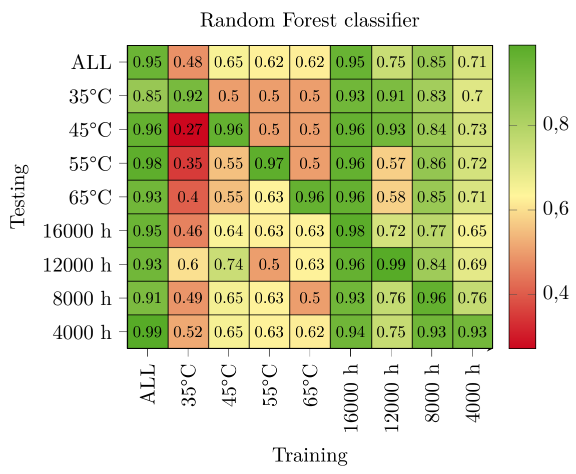
Figure 4.8: Evaporator fouling effects summary at $T_{\mathrm{w,o}} = 35$ °C on a typical year simulation with 16.000 h to max fault EF (cont).

the sets are organized by total combination (as depicted in Figure 3.16), working condition, and fault evolution; these last two are ordered from the least to the most demanding state. The main diagonal of this array gives the classic score of an algorithm trained and tested with the same dataset, the variations lay outside the main diagonal.

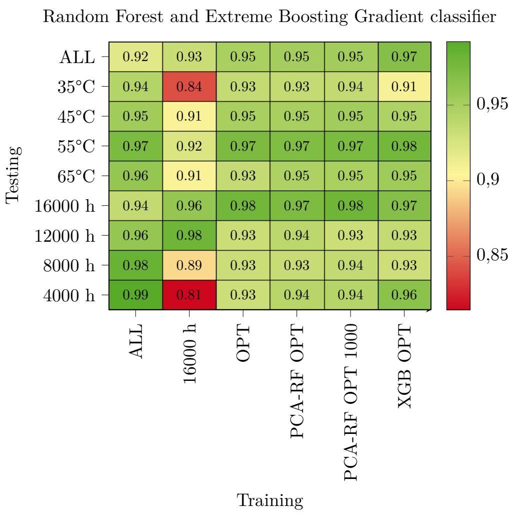
This first figure leads to a significant discovery. Excluding the algorithms trained with all available, the best performance is found on those algorithms trained with the 16.000 h train set. The key characteristic of this subset is that it has more no-fault samples, given it is the softest progression. In other words, the second most effective train set, which is 4 times smaller than the first, is the one that represents better the no-fault state. Hereafter, the other classifiers are trained only with all the data available and the 16.000 h train set; the test sets for all the individual sets are maintained.

Having reduced the 9 train sets to 2, additional trials are executed to test optimization methods and assess the performance with a reduced set of features; Figures 4.9b, 4.10b, and 4.10d address that second interest. HPSKL has three characteristics that are relevant to highlight in stage: first, its classes *any_classifier* and *any_proprocessing*, which can suggest preprocessors (e.g., scalers or dimension reductors) and classifiers; second, the number of iterations. Throughout all the executed trials, it was found that no substantial improvement is reached after 200 iterations. This is evidenced by comparing the scores of the classifiers called *OPT 200* and *OPT 1000*, where the number describes the iteration number. It is implicit that all classifiers not trained with all data available (labeled by *ALL* in the figures) are trained with the 16.000 h set.

At this stage, with over 35.000 samples and at least 12 features, the speed of the SVM could become an issue to consider. While the algorithm trained with all features and the 16.000 h



(a) RF all features scores.



(b) RF and XBG only temperatures scores.

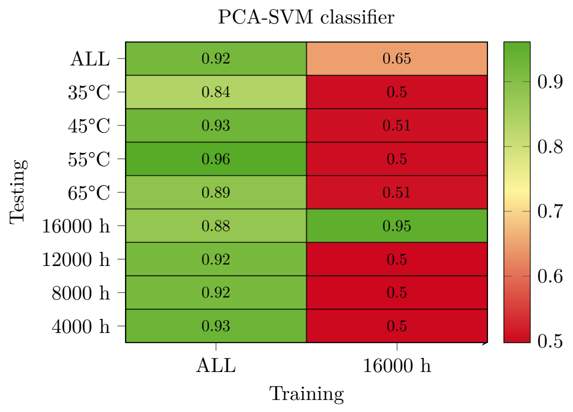
Figure 4.9: Random Forest single class scores.

train set reports a poor performance in all the other test sets (Figure 4.10a), there is a major increase in performance when the features are reduced (Figure 4.10b). Table 4.6 summarizes Figures 4.9 and 4.10 scores, only noting the highest mean balanced accuracy in each type. The conclusion of the single class scores is ANN at the top of the ranking, with RF and XBG as close seconds. These results demonstrate that it is possible to achieve high accuracies only with temperature features. Moreover, the estimators seem to benefit more from the number of instances represented than from the fault intensity that the samples describe. While the representation of intensity can influence the over or underestimation of the prediction, for the highest accuracy are required more diverse instances.

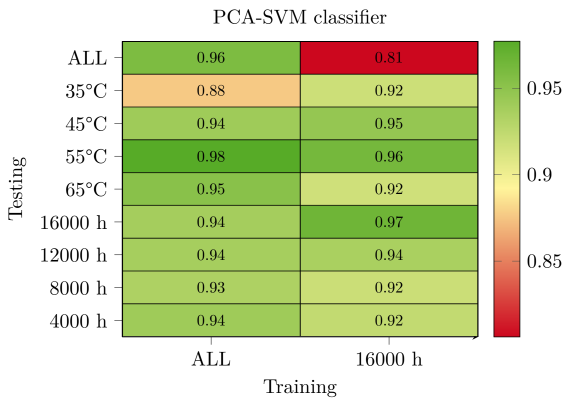
Table 4.6: Mean balanced accuracy scores for single class best classifiers.

	ALL Features		**Temperatures**	
Classifier	**All sets**	**16.000 h**	**All sets**	**16.000**
RF	0,94	0,95	0,96	0,95
XBG	–	–	–	0,95
PCA-SVM	0,91	0,57	0,94	0,92
MLP	0,98	0,96	0,98	0,97

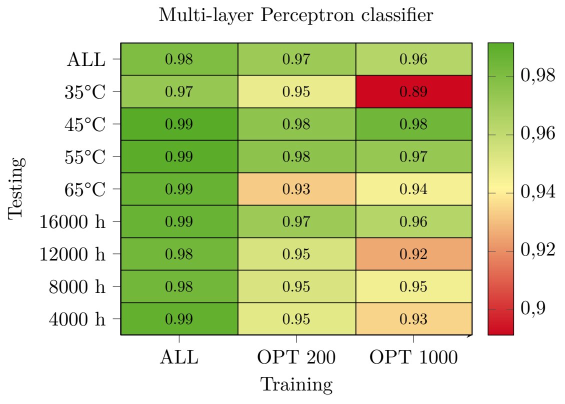
In Table 4.6 Extreme Boosting Gradient is only displaying results with the set of 16000 h. This estimator is considered after a trial with HPSKL, where implementing the *any_classifier* class, it was given the task of suggesting the best classifier after 1000 iterations. This procedure was done to test the capacities of the library and to explore which relationships could be automatically made. In the end, the library provided a suitable algorithm, taken into account for the latter stages.



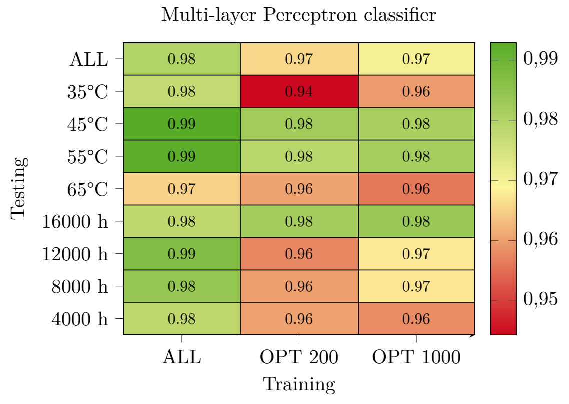
(a) PCA-SVM all features scores.



(b) PCA-SVM only temperatures scores.



(c) MLP all features scores.



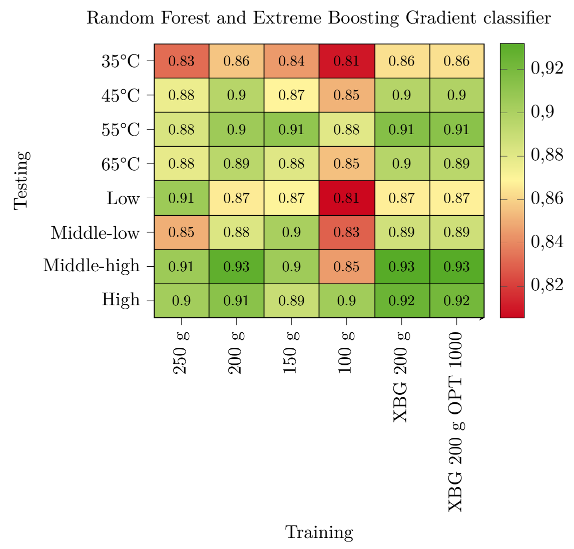
(d) MLP only temperatures scores.

Figure 4.10: SVM and ANN single class scores.

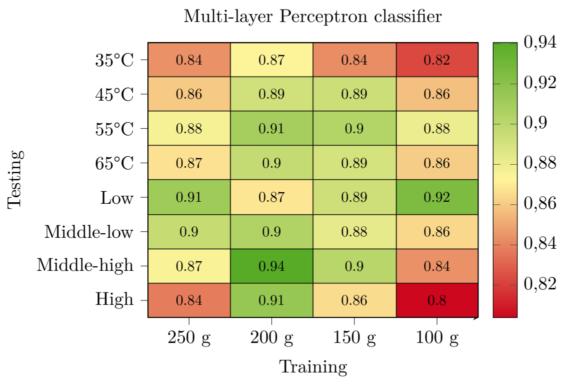
4.2.2 Multiclass classifiers

Although the results from the refrigerant leakage simulations did not yield an acceptable emulation, the reduction of refrigerant mass in the virtual environment produce deviations from normal operations. Although refrigerant leakage is not precisely being emulated, considering the trends from the experiment trials (Figure 4.5), there is a fault within the system that could be used to test how well does the algorithms perform in a multiclass scenario. Hence, the resultant datasets derived from decreasing the refrigerant mass are repurposed to set the *Others faults* class.

This third category has the objective of testing what effects have an extra class on the performance of the classifiers. For this step, the combination of all datasets is not considered, only the combination of the 16.000 h train set and different annualized leakages. The test sets are merged according to their fault intensity for the fault evolution group: 4.000 h + 250 g (High), 8.000 h + 200 g (Middle-high), 12.000 h + 150 g (Middle-low), 16.000 h + 100 g (Low). Regarding the features, as the preceding section proved that high accuracies are achieved only with temperatures, only the reduced feature set is considered.



(a) RF and XBG.



(b) MLP.

Figure 4.11: Multiclass classifiers scores.

In this stage, the size of the datasets was demonstrated to be excessively large for the SVM. The training, prediction, and optimization require longer periods of time than the other algorithms. Accounting also that the SVM ranked last on the single class validation, this algorithm is excluded from the multiclass trials. To conclude, XBG and ANN have a mean score of 0,9 and 0,89 for RF.

4.2.3 Regressors

The development of the regressors benefits from the experiences with the classifiers, which provided substantial insight into data quality and algorithms' training. First, the *normal* no-fault model is developed. These models are trained with a 4-year data set that comprises all the working conditions considered, maintaining the train-test split according to Figure 3.16. Addressing the fact that the regressors require more precise outputs, particularly given the small magnitudes of air volume flow, a feature is created to enhance the prediction performance. The generation of this feature was especially motivated given that the *evaporator's surface temperature* ($T_{\mathrm{ev,surf}}$) represents another inexpensive measurement, in fact, it is a common sensor in real heat pumps. Equations 4.4, 4.5, and 4.6 are empirical relationships derived from Kleipass [63] experiments. They relate ambient temperature, supply water temperature, and fan speed; the latter as a measure of fault. The fan speed variable is not a real speed, instead, it is related to the emulation of the blockage as depicted in Table 4.2.

$$T_{\text{surf}} = T_{\text{amb}} + (0.0288 \cdot n_{\text{fan}}^2 - 0.8248 \cdot n_{\text{fan}} + 11.829) \implies T_{\text{w,o}} = 35°C \tag{4.4}$$

$$T_{\text{surf}} = T_{\text{amb}} + (-0.002 \cdot n_{\text{fan}}^3 + 0.067 \cdot n_{\text{fan}}^2 - 0.7512 \cdot n_{\text{fan}} + 6.47) \implies T_{\text{w,o}} \in \{45, 55\}°C \tag{4.5}$$

$$T_{\text{surf}} = T_{\text{amb}} + (0.0356 \cdot n_{\text{fan}}^2 - 0.9486 \cdot n_{\text{fan}} + 9.1855) \implies T_{\text{w,o}} = 65°C \tag{4.6}$$

Table 4.7: No-fault model regressor

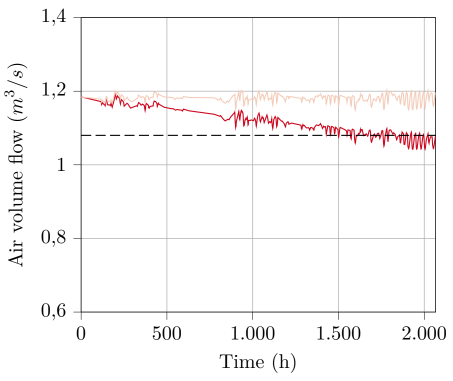
	RF	**XBG**	**MLP**	**SVR**	**RF-TEMP**	**XBG-TEMP**	**MLP-TEMP**	**SVR-TEMP**
RMSE	8.10^{-5}	4.10^{-4}	0,002	0,021	0,002	0,001	0,003	0,018
R^2	0,999	0,998	0,999	0,976	0,964	0,985	0,999	0,982

Table 4.7 describes the $RMSE$ and R^2 scores of the algorithms, first with the "All features" group to define a baseline, then with the "Temperatures features" group. In this application, every estimator had acceptable performance. ANNs and tree ensembles are in the top positions by a few decimals. Additionally, the regression approach shows that the "Temperatures" group yields accurate results. Figure 4.12 confirms that the fault threshold is suitable based on the ANN regressor performance, it illustrates the fault behavior in the 16.000 h to maximum fault for the no-fault model and the actual value. This proves that this model is insensitive to the evaporator's fouling.

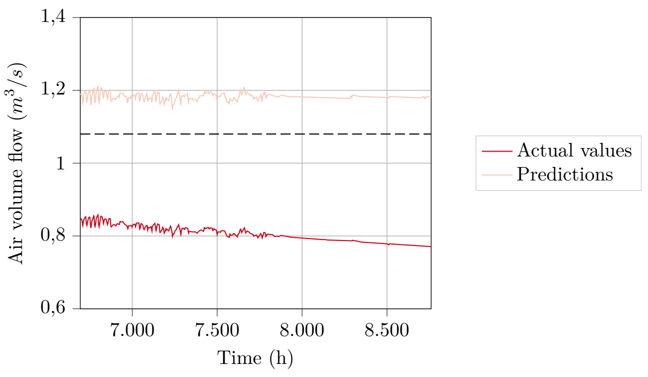
Subsequently, the fault model is developed only considering the Temperatures feature set and implementing ADDMo to find the best configuration of algorithms and features. Table 4.8 shows the results of the ANN trained under different sets. Unlike the classifiers, the regressors require more than one data group to emulate the fault accurately. Only the group that combines the two extreme progressions 16.000 h + 4.000 h (least and maximum fault) provides acceptable; i.e., low RMSE and near 1 R^2 scores. The performance of the ANN is examined against the test subset of each group, however, a broader test with all the data is reserved for those with an adequate score on the previous test. This condition is fulfilled just by 16.0000 h + 4.000 h. For the final evaluation, this model and the one trained with all data available are considered.

Table 4.8: Fault model MLP regressor.

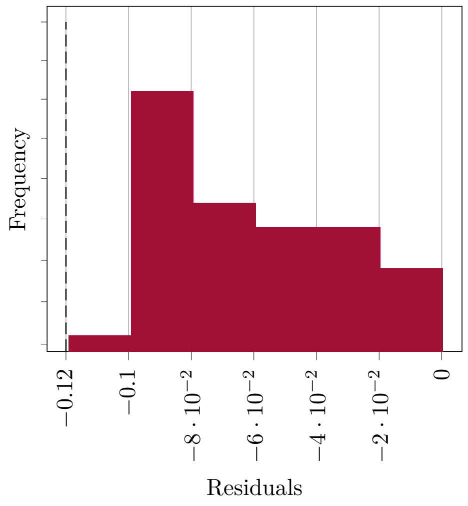
	ALL	**16.000 h**	**12.000 h**	**8.000 h**	**4.000 h**	**16.000 + 4.000 h**	**12.000 + 8.000 h**
RMSE	-	1,847	1,557	1,324	0,964	0,250	1,226
R^2	-	-1,338	-0,693	-0.317	0,002	0,931	-0,127
RMSE	0,171	-	-	-	-	0,226	-
R^2	0,974	-	-	-	-	0,940	-



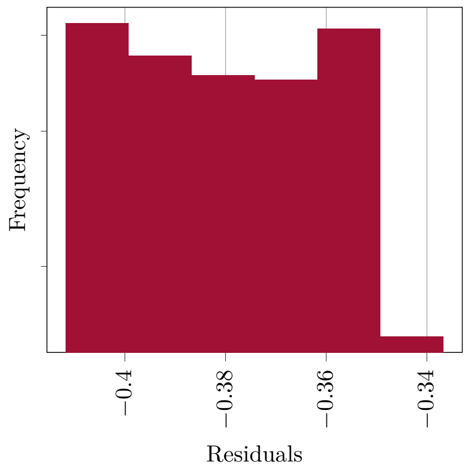
(a) Start of the fault at the beginning of the year.



(b) Maximum fault level at the end of the year.



(c) Residuals under no-fault / least fault conditions

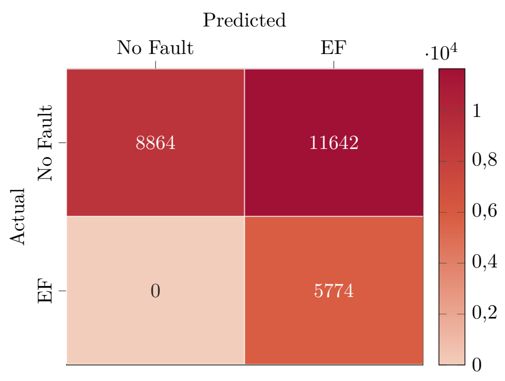


(d) Residuals under maximum fault conditions.

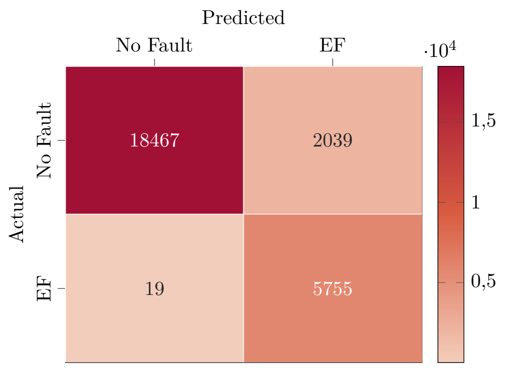
Figure 4.12: Validation of fault threshold for regressors.

4.2.4 Final Evaluation

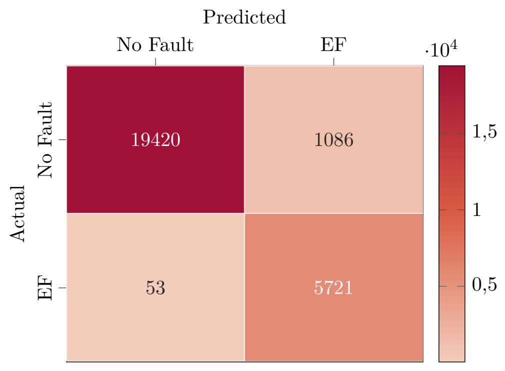
In all the preceding implementations, ANNs have excelled, ranking first in both classification and regression tasks. Based on those previous results, only the MLP regressors and classifiers are considered. To assess the resulting FDD algorithm, the three datasets described in Section 3.3.4 are prepared: fault progression inversion, medium fault injection, and high fault injection. The outcomes of the algorithms in the three sets are similar, hence, the overall results are depicted in Figures 4.13 and 4.14; also in Appendixes B.8 and B.9. However, the results are split by working conditions to examine any possible biases. These are found in the $T_{\mathrm{w,o}}$=35°C and $T_{\mathrm{w,o}}$=65°C conditions, where no-fault condition is largely misclassified, in comparison with the other working conditions. Regarding the classifiers, the same phenomena occur without distinction of working condition. All the estimators exhibit a high false alarm rate. This could indicate a bias towards the fault state prediction, which was the majority class in most training tests. The only exception to this is the no-fault regressor, trained only with no fault data. Moreover, the fault regressor and classifiers were tested in sets with predominant fault instances.



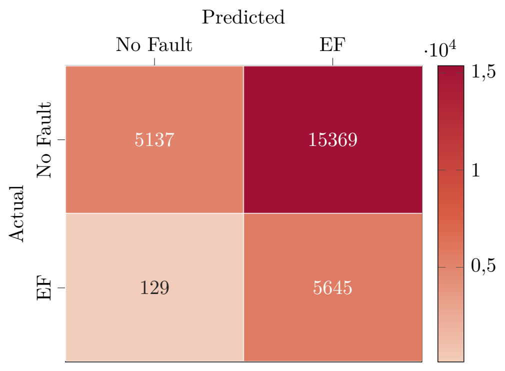
(a) $T_{\mathrm{w,o}}$=35°C



(b) $T_{\mathrm{w,o}}$=45°C



(c) $T_{\mathrm{w,o}}$=55°C



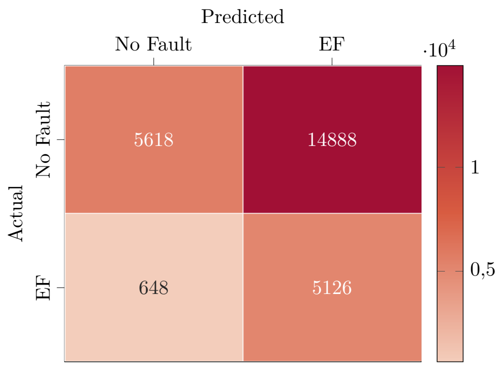
(d) $T_{\mathrm{w,o}}$=65°C

Figure 4.13: MLP regressor performance on evaluation set.

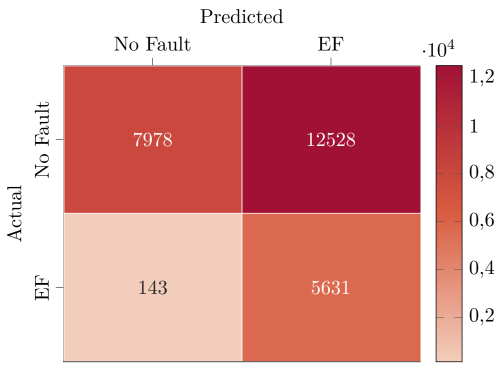
Table 4.9 summarizes the results of the best two estimators of each strategy through the FDD metrics defined in 2.2.2.1. Evidently, the regressors outperform the classifiers. It is noteworthy to mention that the misdiagnosis rate is not applicable, due to it requires more classes than the ones that the use case reflect. In multiclass scenarios, for example, evaporator fouling and refrigerant leakage, it should be taken into account. As a result, the two-model regressor approach is selected as the most suitable FDD algorithm.

Table 4.9: Regressors and classifiers comparison.

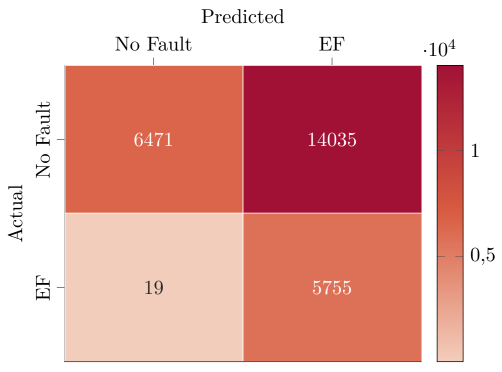
	MLP REG	**ANN 16.000h + 4000h**	**MLP CLAS**	**RF CLAS**
Correct rate	71%	70%	48%	43%
False alarm rate	29%	29%	52%	57%
Missed detection rate	0%	1%	1%	1%



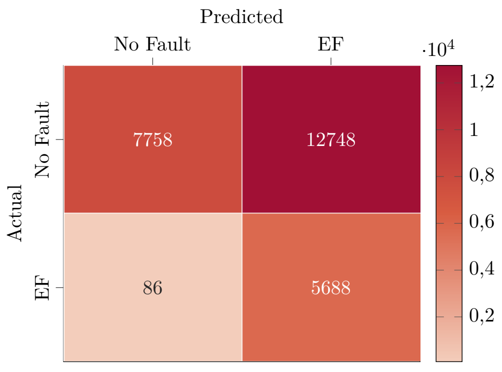
(a) $T_{\mathrm{w,o}}$=35°C



(b) $T_{\mathrm{w,o}}$=45°C



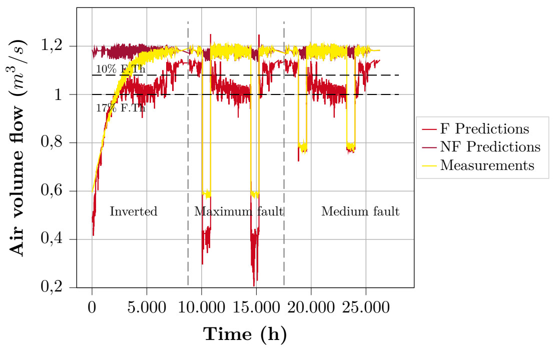
(c) $T_{\mathrm{w,o}}$=55°C



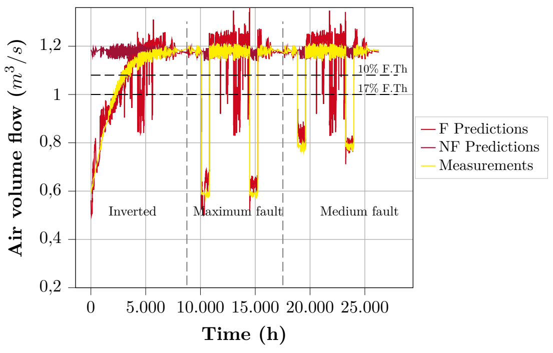
(d) $T_{\mathrm{w,o}}$=65°C

Figure 4.14: MLP classifier performance on evaluation set.

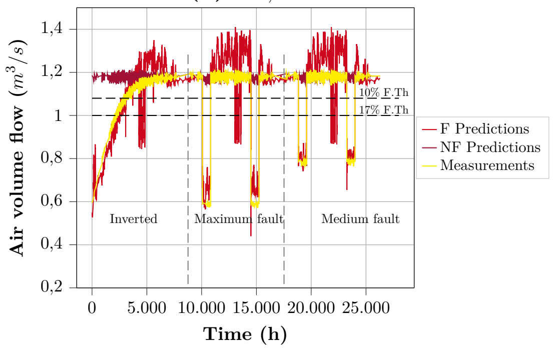
To illustrate the predictions of the regressors, Figure 4.15 portrays actual values and predictions in all working conditions. This figure reveals that the selected fault threshold could be conservative for appropriate detection, forcing the FDD protocol to be excessively sensitive and resulting in a high false alarm rate. The minimal missed detection rate confirms the false alarm—missed detection tradeoff, thus allowing for some correction to balance the performance. In Figure 4.6 is showed that the effects of evaporator fouling are more notorious after a fault intensity of 20%. Therefore, a fault threshold (portrayed as black dashed lines in Figure 4.15) of 17% is a suitable alternative, still maintaining a strict boundary; being the current fault threshold of 10% (residual > 0,12 m^3/s). This modification has the potential to improve the results from regressors and classifiers. Nonetheless, Figure 4.15 exhibits that attention must also be paid to the performance in summer periods, this adds another important consideration to enhance the test groups. The current hypothesis is that test sets comprised of no-fault (winter+summer), fault in summer, and fault in winter, equally represented, could lead to improved results. This hypothesis bases on the fact that the evaluation and optimization were done on purely winter period conditions, with a low representation of no-fault conditions. As a final note, the generated data (i.e., virtual environment results) should be further validated in the aforementioned periods.



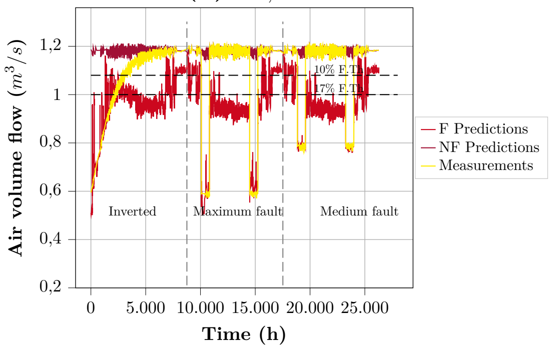
(a) $T_{\mathrm{w,o}}$=35°C



(b) $T_{\mathrm{w,o}}$=45°C



(c) $T_{\mathrm{w,o}}$=55°C



(d) $T_{\mathrm{w,o}}$=65°C

Figure 4.15: Regressor models predictions in the evaluation set.

4.2.5 Individual performance of algorithms

4.2.5.1 Support Vector Machine performance

During the initial stages the Support Vector Machine algorithms performed adequately, notwithstanding, the lack of defined feature selection strategies could have hindered their results on the final stages. Moreover, the sizes of the datasets during the majority of trials from the typical year simulations increased the computational time required to train these algorithms. The recommendations regarding SVMs point to implement them for small to medium sets, due to the fact that the computational complexity function O bases on the square (or even the third) power of the total number of samples times the number of features; this can be expressed as $O(m^2 \times n)$ [41]. To illustrate this, the simplest data set in the typical year simulations featured 4 working conditions of 8760 samples and at least 8 features $O((8760 \cdot 4)^2 \times 8)$, thus the basic computational complexity function is $O(9,8 \cdot 10^9)$. Even though HPSKL performed a Principal Component Analysis, the training time was significant, specially against the other algorithms whose speed were not equally affected. Evidently, automated machine learning algorithms are helpful for the optimization stages. Nonetheless, the analysis of the *best model* cannot be neglected. As suggested by SVM performance, an algorithm that requires a Principal Component Analysis (PCA) is one that could benefit from another iteration of the feature selection process, as performed in two stages in AD-DMo. This recommendation must be taken in the proper context, accounting for the fact that different algorithms benefit from different features (SVM with DT features case from Table 4.5). Between the hyperparameters of the best performing SVM classifier are a fourth degree polynomial kernel and a C parameter of 388.

4.2.5.2 Decision Tree performance

The Decision trees are the only classifiers during the steady-state phase that oversimplified the problem; particularly in one scenario, given the linear relationship between the fault and the dry air virtual sensor. From the literature review, this behavior was anticipated, hence, the ensemble alternatives that build up from the agglomeration of these algorithms were prepared. The Random Forest, performs as a fast, easy to optimize algorithm, which led to complying results. Similarly, Extreme Gradient Boosting, recommended by HPSKL, shows to be suitable for this use case. The performance of both algorithms rank among the bests, surpassed slightly by the ANNs. This behavior corresponds to trends found in the literature, specially in [48]. A relevant hyperparameter to mention regarding the random forests is the number of estimators, in the case of the best classifier, are 100 estimators.

4.2.5.3 Artificial neural networks performance

The Artificial Neural Networks are the best performing algorithm among all the estimators, at every stage, in each approach. While the performance was very similar to that from the trees in the multiclass trials, it was not tested whether the algorithm could benefit from an enhancement in complexity, including another hidden layer. Regardless of this, Guo et al. [80] comments that the common assumption "more layers, better performance" could not be entirely true, given the increase in complexity in the ANN architecture will be beneficial only if the problem requires it. Aggregating complexity to the ANN can serve other purposes rather than only improve accuracy. A single ANN can be trained as a regressor and classifier, moreover, can be its own no-fault and fault model. This implementation would require multi-output ANNs, which were not considered in this use case for the sake of comparison. Between the hyperparameters of the best performing ANN are "Adam" solver, one layer of 12 neurons in the no-fault regressor, three layers of 4, 96, and 96 neuron in the fault regressor, and three layers of 105, 240, and 5 neurons in the classifier.

Referring to the multiclass trials, these were carried on to test the potential of a single algorithm to handle multiple faults categories. Evidently, an aggregation of several estimators, each trained for a specific fault, is possible; this is called a *voting classifier* [41]. However, it was deemed relevant to explore whether one of these algorithms could handle this task on their own. The results point that RF, XBG, and ANNs can perform with high accuracies in multiclass tasks. Whether this approach is advantageous for multiple simultaneous faults or to address fault intensities are questions for future research.

To conclude this section, it is worth mentioning aspects about the sequence in which the classification and regression approaches were carried on. Pondering that the regression task was decided from the beginning to be a binary model (no-fault and fault) the optimization of aspects inherent to the data itself and not the algorithms seemed more appropriate from a single estimator strategy; because of simplicity. Significant lessons were learned to apply in the regressors. It is thought that, the inverse case would have been equivalent, this is to have taken on the regression road first. The factor that possibly makes a substantial difference, is that the regression approach harness complete no fault datasets (8760 samples each), while the regressors only a few samples (from 500 to 2066 samples).

5 Conclusions and outlook

This thesis develops a simulation model of an air-to-water heat pump, emulating a simple control system to maintain a constant superheat and adapt the compressor's speed to the required supply water temperature. Superheat is regulated by varying the expansion valve opening (similar to an electronic expansion valve), while the compressor's speed is controlled through variations in frequency. This configuration is achieved in Dymola with independent PID controller modules.

The reference model is set approximating the characteristics of an existing heat pump test bench, additionally, fault modeling techniques are established. Two of the most common faults in heat pumps are selected, from which only the no-fault status and the evaporator fouling complied. These strategies adhere to physical interpretations of the variables. The selected fault modeling techniques are to decrease refrigerant mass and fan speed; later, it is realized that the overall heat transfer coefficient should be decreased as well for a better match. In those cases where the data was unavailable, the closest reference to the variables or default values from Dymola is utilized.

The base Dymola model shows a deeper need for additional component development to model refrigerant leakage in comparison with evaporator fouling. Methodologies such as the one proposed by Song et al. [21] through the use of soft (virtual) sensors with manufacturer's data could be a starting point for much broader fault modeling strategies in virtual environments. Still, Dymola showed many utilities regarding mathematical computations and programming techniques suitable for this application. A significant limitation is the simulation time, for example, the typical year simulations required 90 min on average each. Simulating several faults for different working conditions and fault intensities, together with the corresponding no-fault state, can take up to several days with the current computational power. Improvements for any base model within this regard are parallelization and optimization of computational resources.

Accounting for the estimations in the model's configuration, the goal is to emulate the trends of no-fault and fault behaviors from experiments performed with the reference machine. This means that the present work limits itself to mimicking these trends, not to provide an accurate match. The mimicking is qualitatively assessed through variance matrixes in each of the chosen cases. As the matrixes indicate, the current approach is the first milestone to a more complex validation, these results only establish a functional reference. This means, that the achieved simulation results can be improved to be closer to the actual reference heat pump. Special focus is required in the thorough fit of the model no-fault behavior. This is considered necessary to reliably assert the match of experimental, simulation, and real-installation operations.

Next, datasets for a range of ambient temperatures, supply water temperature, and fault intensities, are generated. Subsequently, the data is preprocessed and disposed to train the Support Vector Machines, Decision Trees (with ensembles), and Artificial Neural Networks. Afterward, two strategies are followed: classification and regression. To settle the best approach, a final evaluation is defined, composed of three sets. First, the faults are imposed without progressions and at specific times of the year at two fault intensities, high for the first set and medium for the second set. The third set inverts the train fault progression to assess whether the performance is actually detecting the fault, or matching ambient temperatures with fault states.

According to the results from all the estimators, to assess evaporator fouling, the only measurements required are: compressor speed n_{comp}, air outlet temperature $T_{a,o}$, ambient temperature T_{amb}, condenser outlet temperature $T_{cond,o}$, discharge temperature T_{dis} , evaporator inlet temperature $T_{ev,i}$, evaporator surface temperature $T_{ev,surf}$, suction temperature T_{suc}, water inlet temperature $T_{w,i}$, water outlet temperature $T_{w,o}$, power consumption sensor $\dot{W}_{vs}$, heating capacity sensor $\dot{Q}_h$, and COP. Although this subset could be further refined for EF, most of these variables translate to inexpensive sensors. Certain could have technical difficulties such as $T_{a,o}$ and $\dot{Q}_h$, however, being able to exclude pressure readings or associated thermodynamic variables is an advance towards a better instrumentation of residential-size heat pumps; without a major increase in costs.

During the final evaluation, the performance of estimators dropped from above 95% of balanced accuracy to around 70%. This drop has several causes: first, the classifiers were biased towards the majority class, second, the regressors did not perform well during the summer times, and third, the fault threshold that divided the no-fault and fault classes was excessively conservative (10% of fault intensity). The metrics used to define the final results are the correct, false alarm, and missed detection rates. A key point revealed in this evaluation is that the biases and assumptions introduced in the training process must be carefully addressed. Once recognized, the evaluation of the algorithms should be prepared accordingly, to reduce the bias in the final results and provide the most impartial scores.

The regressors, which featured a binary approach of no-fault reference model and fault model, where the difference between each denotes the fault after the threshold, scored the best. Their scores were of 71% correct rate, 29% false alarm, and 0% missed detection. Clearly, these results suggest that the fault boundary can be adjusted to balance the performance. In particular, given that the literature regards more negatively a high false alarm rate than a high missed detection rate, due to the loss of confidence in the system from the users, and possible unnecessary services.

The main conclusions of this thesis are:

- The artificial neural networks proved to be the best estimator, both in classification and regression approach, according to the followed methodology.
- Evaporator fouling can be detected only with temperature sensors as features for the machine learning algorithms.
- The dataset split (training and testing) require an equivalent representation of no fault in summer period, no fault in winter period, fault in summer period, and fault in winter period, to provide reliable results. The formulation of imbalance datasets results can deliver misleading scores.

On the other hand, there are three aspects, which this thesis does not provide conclusive results, that are deemed as noteworthy:

- Determining if the typical year approach is the best to approximate real data. The exhibited methodology was deemed more suitable than the steady-state Points Matrix, considering the number of samples and represented variation. Additionally, it is possible for a heat pump to measure and store the mean of its readings hour-by-hour with the right interfaces and controllers. Thus, collecting and storing datasets similar to the ones presented in this research. Notwithstanding, it is still necessary a proof of concept, whether it be with experimental at different ambient conditions, or from an installed household-heat pump, to validate the right approach.
- The impact of the fault progression. While the real progression of faults can adopt many forms, the optimal degradation sequence for training the algorithms is to be determined in further research. This point could gain more importance when dealing with multiple simultaneous faults, where synergistic and attenuating effects between faults occur.
- Minimum necessary number of instances. It is observable, through the experiences with classifiers and regressors, that the instances in a dataset can be optimized. This could lead to the generation of more refined sets, where a broader range of cases is characterized without the requirement of simulating or collecting entire years at a time. Nonetheless, this thesis also features a case where the algorithms are trained with insufficient samples for a proper generalization. Therefore, a specific number of samples is not found within this work, still, it is demonstrated that is possible to converge to an optimum.

Throughout the reviewed literature, there were noticed several efforts in the development of algorithms, ensembles, and architectures for HVAC FDD targeted at numerous applications (e.g., vehicle ventilation [46], boilers [81], buildings [48, 82, 35, 8]) apart from heat pumps. Nonetheless, few inputs were found in topics common across estimators, such as good practices regarding data generation, training, and testing. The full potential of any algorithm is only revealed under the light of these three pillars, still, most of the recommendations limit to feature selection. Therefore, to advance in the specifics of every use case, there is an important need for fundamental guidelines to cover, among others:

- The compilation of benchmark data to address multiple ambient conditions, heat pumps, installations, and demands.
- Validated fault progression behaviors to model.
- Data distribution for training and testing.

Up to this point, there have been layout strategies for virtual environment development, fault modeling, data generation, algorithm training, and evaluation. These steps have the potential to be replicated in an industry setting, in particular, with the findings of this work regarding necessary features for EF FDD. Although only outdoor air flow was successfully analyzed in full, the indicated measurements together with virtual sensors could be implemented for a comprehensive FDD. This matter must be defined during the design phase, considering that the placement of the sensors is of high importance. Next, the evaluation of this measuring system should come along with the quality tests of the heat pump. Afterward, with the corresponding data from the laboratory trials, a simulation model can be validated to extrapolate conditions and generate the datasets for the machine learning algorithm. As in this research, a base virtual environment with the proper settings (e.g., geometries, capacities, dimensions, etc.) can be easily adapted across several heat pump models.

During the MLA phase, each R&D team has to decide which approach and algorithm find most suitable to ensure reliability and evolve in time. While the ANNs performed the best and has the most potential architecture and optimization-wise, decision trees ensembles are a decent second. Subsequently, trained algorithms can be deployed in an online (e.g., cloud based) application or on-site analysis device. For any of those applications, sufficient data must be collected, data storage protocols need to be strictly implemented as well, where memory considerations are taken into account. Zhao et al.[8] and Bode et al. [35] report several issues regarding incomplete and lost data. The tool should feature a preprocessing module in order to deal with this.

Having trained algorithms, data collection, storage, and preprocessing guaranteed, the last topic to treat is *data drift.* Installation variables, such as tube length in a split configuration, or presence of a tank, must be accounted as parameters within the simulation model. This is a start to address the matter of setting the no-fault model and retraining. As exhibited by Bode et al. [35], establishing a reference status for on-field systems is a difficult task, because to assess the no-fault state can lead to neglect installation or service faults. A strategy could be to closely monitor the system for a period of time (whether new or existing) and define the normal operation consequently. The present research works with a fault threshold, where a margin of deviance from laboratory trials is expected. This could be a strategy to determine initial *healthy* operation. Nonetheless, the aging of the systems and how to prepare the models to adapt this phenomenon is a current research gap.

As a final remark, the application of fault detection and diagnosis techniques in residential-size heat pumps incentive the implementation of Internet-of-Things interfaces within these equipments for data monitoring and storage. Another alternative are the Energy Management Systems for households, which avoid the addition of these interfaces directly into the machine. As mentioned in Section 2.2.4, the information derived from the supervision of these equipments can bring benefits to the value chain, from grid operators, to manufacturers, sellers, and final users. These enhancements can translate into improved reliability and durability of the heat pumps.

Bibliography

[1] IEA. World energy balances (database), 2022. www.iea.org/reports/world-energy-balances-overview (accessed 23.05.2023).

[2] IEA. Heating, 2022. www.iea.org/reports/heating (accessed 17.05.2023).

[3] IEA. World energy balance: Germany, 2022. www.iea.org/countries/germany (accessed 17.05.2023).

[4] IEA. Global heat pump sales continue double-digit growth, 2023. www.iea.org/reports/heating (accessed 17.05.2023).

[5] European Comission. Repowereu: A plan to rapidly reduce dependence on russian fossil fuels and fast forward the green transition, 2022. www.ec.europa.eu/commission/presscorner/detail/en/IP_22_3131 (accessed 16.05.2023).

[6] I. Bellanco, E. Fuentes, M. Vallès, and J. Salom. A review of the fault behavior of heat pumps and measurements, detection and diagnosis methods including virtual sensors. *Journal of Building Engineering*, 39:102254, 2021.

[7] Yifeng Hu, David P. Yuill, Amir Ebrahimifakhar, and Ali Rooholghodos. An experimental study of the behavior of a high efficiency residential heat pump in cooling mode with common installation faults imposed. *Applied Thermal Engineering*, 184:116116, 2021.

[8] Yang Zhao, Tingting Li, Xuejun Zhang, and Chaobo Zhang. Artificial intelligence-based fault detection and diagnosis methods for building energy systems: Advantages, challenges and the future. *Renewable and Sustainable Energy Reviews*, 109:85–101, 2019.

[9] A. P. Rogers, F. Guo, and B. P. Rasmussen. A review of fault detection and diagnosis methods for residential air conditioning systems. *Building and Environment*, 161:106236, 2019.

[10] Eric Granryd, Jngvar Ekroth, Per Lundqvist, Ake Melinder, Bjorn Palm, and Peter RoWin. *Refrigerating Engineering*, volume II. Department of Energy Technology, Division of Applied Thermodynamics and Refrigeration. Royal Institute of Technology, KTH, Stockholm, 2002.

[11] Walter Grassi. *Heat Pumps: Fundamentals and Applications.* Green Energy and Technology Ser. Springer International Publishing AG, 2018.

[12] Eric Granryd, Jngvar Ekroth, Per Lundqvist, Ake Melinder, Bjorn Palm, and Peter RoWin. *Refrigerating Engineering*, volume I. Department of Energy Technology, Division of Applied Thermodynamics and Refrigeration. Royal Institute of Technology, KTH, Stockholm, 2003.

[13] Srinivas Katipamula and Michael Brambley. Review article: Methods for fault detection, diagnostics, and prognostics for building systems—a review, part i. *HVAC&R Research*, 11(1):3–25, 2005.

[14] Vijay Singh, Jyotirmay Mathur, and Aviruch Bhatia. A comprehensive review: Fault detection, diagnostics, prognostics, and fault modelling in hvac systems. *International Journal of Refrigeration*, 2022.

[15] Woohyun Kim and Srinivas Katipamula. A review of fault detection and diagnostics methods for building systems. *Science and Technology for the Built Environment*, 24(1):3–21, 2017.

[16] Iva Matetić, Ivan Štajduhar, Igor Wolf, and Sandi Ljubic. A review of data-driven approaches and techniques for fault detection and diagnosis in hvac systems. *Sensors*, 23(1), 2023.

[17] David P. Yuill and James E. Braun. Evaluating fault detection and diagnostics protocols applied to air-cooled vapor compression air-conditioners. 2012.

[18] David P. Yuill and James E. Braun. Evaluating the performance of fault detection and diagnostics protocols applied to air-cooled unitary airconditioning equipment. 19(7):882–891, 2013.

[19] Yifeng Hu, David P. Yuill, Seyed Ali Rooholghodos, Amir Ebrahimifakhar, and Yuxuan Chen. Impacts of simultaneous operating faults on cooling performance of a high efficiency residential heat pump. *Energy and Buildings*, 242:110975, 2021.

[20] Minsung Kim, Seok Ho Yoon, W. Vance Payne, and Piotr A. Domanski. Development of the reference model for a residential heat pump system for cooling mode fault detection and diagnosis. 24(7):1481–1489, 2010.

[21] Yang Song, Davide Rolando, Javier Marchante Avellaneda, Gerhard Zucker, and Hatef Madani. Data-driven soft sensors targeting heat pump systems. *Energy Conversion and Management*, 279:116769, 2023.

[22] Woohyun Kim and Je-Hyeon Lee. Fault detection and diagnostics analysis of air conditioners using virtual sensors. *Applied Thermal Engineering*, 191:116848, 2021.

[23] Mark S. Breuker and James E. Braun. Common faults and their impacts for rooftop air conditioners. 4(3):303–318, 1998.

[24] Mehdi Mehrabi and David Yuill. Generalized effects of refrigerant charge on normalized performance variables of air conditioners and heat pumps. *International Journal of Refrigeration*, 76:367–384, 2017.

[25] Mehdi Mehrabi and David Yuill. Generalized effects of faults on normalized performance variables of air conditioners and heat pumps. *International Journal of Refrigeration*, 85:409–430, 2018.

[26] I. Bellanco, F. Belío, M. Vallés, R. Gerber, and J. Salom. Common fault effects on a natural refrigerant, variable-speed heat pump. *International Journal of Refrigeration*, 133:259–266, 2022.

[27] Hatef Madani. The common and costly faults in heat pump systems. *Energy Procedia*, 61:1803–1806, 2014.

[28] Muhammad Awais and Arafat A. Bhuiyan. Recent advancements in impedance of fouling resistance and particulate depositions in heat exchangers. *International Journal of Heat and Mass Transfer*, 141:580–603, 2019.

[29] Zhimin Du, Piotr A. Domanski, and W. Vance Payne. Effect of common faults on the performance of different types of vapor compression systems. *Applied Thermal Engineering*, 98:61–72, 2016.

[30] Seok Ho Yoon, W. Vance Payne, and Piotr A. Domanski. Residential heat pump heating performance with single faults imposed. *Applied Thermal Engineering*, 31(5):765–771, 2011. MNF 2009 Special Issue.

[31] Kim Minsung, William Payne, Piotr Domanski, and Christian Hermes. Performance of a residential heat pump operating in the cooling mode with single faults imposed (nistir 7350), 2006-09-01 2006.

[32] Derek Noel, Philippe Riviere, C. Teuillieres, O. Cauret, and D. Marchio, editors. *Experimental Characterization of Fault Impacts on the Functioning Variables of an Inverter Driven Heat Pump*, 2018.

[33] International Institute of Refrigeration. 24th informatory note on refrigeration technologies: Containment of refrigerants within refrigeration, air conditioning and heat pump systems. international institute of refrigeration 2014.

[34] Piotr Domanski, Hugh Henderson, and William Payne. Sensitivity analysis of installation faults on heat pump performance, 2014-10-08 2014.

[35] Gerrit Bode, Simon Thul, Marc Baranski, and Dirk Müller. Real-world application of machine-learning-based fault detection trained with experimental data. *Energy*, 198:117323, 2020.

[36] Haorong Li and James E. Braun. Decoupling features and virtual sensors for diagnosis of faults in vapor compression air conditioners. *International Journal of Refrigeration*, 30(3):546–564, 2007.

[37] Haorong Li and James E. Braun. Virtual refrigerant pressure sensors for use in monitoring and fault diagnosis of vapor-compression equipment. pages 597–616, 2009.

[38] Woohyun Kim and James E. Braun. Development and evaluation of virtual refrigerant mass flow sensors for fault detection and diagnostics. *International Journal of Refrigeration*, 63:184–198, 2016.

[39] Woohyun Kim and James E. Braun. Development, implementation, and evaluation of a fault detection and diagnostics system based on integrated virtual sensors and fault impact models. *Energy and Buildings*, 228:110368, 2020.

[40] Tom Mitchell. *Machine learning*. McGraw-Hill series in computer science. McGraw-Hill, New York, NY, international ed., [reprint.] edition, 1997.

[41] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly, 2nd edition, 2019.

[42] A. Zheng. *Evaluating Machine Learning Models: A Beginner's Guide to Key Concepts and Pitfalls*. O'Reilly Media, 2015.

[43] Alice Zheng and Amanda Casari. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media, Inc., 1st edition, 2018.

[44] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer New York, 2013.

[45] Amir Mosavi, Mohsen Salimi, Sina Faizollahzadeh Ardabili, Timon Rabczuk, Shahaboddin Shamshirband, and Annamaria R. Varkonyi-Koczy. State of the art of machine learning models in energy systems, a systematic review. *Energies*, 12(7), 2019.

[46] Qiang Lei, Chensi Zhang, Junye Shi, and Jiangping Chen. Machine learning based refrigerant leak diagnosis for a vehicle heat pump system. *Applied Thermal Engineering*, 215:118524, 2022.

[47] Kadir Amasyali and Nora M. El-Gohary. A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews*, 81:1192–1205, 2018.

[48] Muhammad Waseem Ahmad, Monjur Mourshed, and Yacine Rezgui. Trees vs neurons: Comparison between random forest and ann for high-resolution prediction of building energy consumption. *Energy and Buildings*, 147:77–89, 2017.

[49] M. Mohanraj, S. Jayaraj, and C. Muraleedharan. Applications of artificial neural networks for refrigeration, air-conditioning and heat pump systems—a review. *Renewable and Sustainable Energy Reviews*, 16(2):1340–1358, 2012.

[50] Alexei Botchkarev. A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:045–076, 2019.

[51] Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview.

[52] James P. Barrett. The coefficient of determination: Some limitations. *The American Statistician*, 28(1):19,20, 1974.

[53] F. Pedregosa, 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, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[54] Brent Komer, James Bergstra, and Chris Eliasmith, editors. *Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn*, 2014.

[55] Martin Rätz, Amir Pasha Javadi, Marc Baranski, Konstantin Finkbeiner, and Dirk Müller. Automated data-driven modeling of building energy systems via machine learning algorithms. *Energy and Buildings*, 202:109384, 2019.

[56] Yanfei Li and Zheng O'Neill. A critical review of fault modeling of hvac systems in buildings. *Building Simulation*, 11(5):953–975, 2018.

[57] David P. Yuill and James E. Braun. Effect of the distribution of faults and operating conditions on afdd performance evaluations. *Applied Thermal Engineering*, 106:1329–1336, 2016.

[58] Dassault Systèmes. Dymola (dynamic modeling laboratory). `www.3ds.com/products-services/catia/products/dymola/` (accessed 21.04.2023).

[59] TLK-Thermo GmbH. Til suite. `www.tlk-thermo.com/index.php/de/til-suite` (accessed 21.04.2023).

[60] Modelica Association. Modelica. `www.modelica.org/modelicalanguage.html` (accessed 21.04.2023).

[61] Raymond Sterling, Gregory Provan, Jesús Febres, Dominic O'Sullivan, Peter Struss, and Marcus M. Keane. Model-based fault detection and diagnosis of air handling units: A comparison of methodologies. *Energy Procedia*, 62:686–693, 2014.

[62] Tim Klebig. Entwicklung eines modularen kältekreises zur klassifizierung der leistungsfähigkeit von low-gwp kältemitteln. Master's thesis, RWTH Aachen, October 2020.

[63] Johanna Kleipass. Experimentell gestützte untersuchung und bewertung neuronaler netze für die fehlerdetektion in wärmepumpenkältekreisen. Bachelor's thesis, RWTH Aachen, March 2023.

[64] Jiazhen Ling, Hongtao Qiao, Abdullah Alabdulkarem, Vikrant Aute, and Reinhard Radermacher. Modelica-based heat pump model for transient and steady-state simulation using low-gwp refrigerants. 2014.

[65] Bertrand Dechesne, Stephane Bertagnolio, and Vincent Lemort. Development of an empirical model of a variable speed vapor injection compressor used in a modelica-based dynamic model of a residential air source heat pump. *IOP Conference Series: Materials Science and Engineering*, 90:012031, 2015.

[66] Michael Jokiel, Michael Bantle, Christian Kopp, and Espen Halvorsen Verpe. Modelica-based modelling of heat pump-assisted apple drying for varied drying temperatures and bypass ratios. *Thermal Science and Engineering Progress*, 19:100575, 2020.

[67] Sven Försterling. *Vergleichende Untersuchung von CO2-Verdichtern in Hinblick auf den Einsatz in mobilen Anwendungen*. Cuvillier Verlag, 2004.

[68] Danfoss. Electric expansion valves: Type ets 6, 2019.

[69] Tim Klebig, Janik Horst, Christian Vering, Valerius Venzik, and Dirk Müller. Development of a modular refrigeration cycle to classify the performance of flammable refrigerants. In *15th IIR-Gustav Lorentzen conference on Natural Refrigerants*. 2022.

[70] D. Müller, M. Lauster, A. Constantin, M. Fuchs, and P. Remmen. Aixlib: An open-source modelica library within the iea-ebc annex 60 framework.

[71] Francesco Pelella, Luca Viscito, and Alfonso William Mauro. Combined effects of refrigerant leakages and fouling on air-source heat pump performances in cooling mode. *Applied Thermal Engineering*, 204:117965, 2022.

[72] Ted Dunning. Practical feature engineering: Strata data conference, 2019. www.learning.oreilly.com/videos/oreilly-strata-data/9781492050681/9781492050681-video327354/ (accessed 05.13.2023).

[73] David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, 1996.

[74] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

[75] David H. Wolpert. What is important about the no free lunch theorems?, 2020.

[76] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi,

Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[77] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[78] James Bergstra, Dan Yamins, and David D. Cox. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *12th Python in Science Conference (Scipy 2013)*. 2013.

[79] Ian H. Bell, Eckhard A. Groll, and Holger König. Experimental analysis of the effects of particulate fouling on heat exchanger heat transfer and air-side pressure drop for a hybrid dry cooler. *Heat Transfer Engineering*, 32(3-4):264–271, 2011.

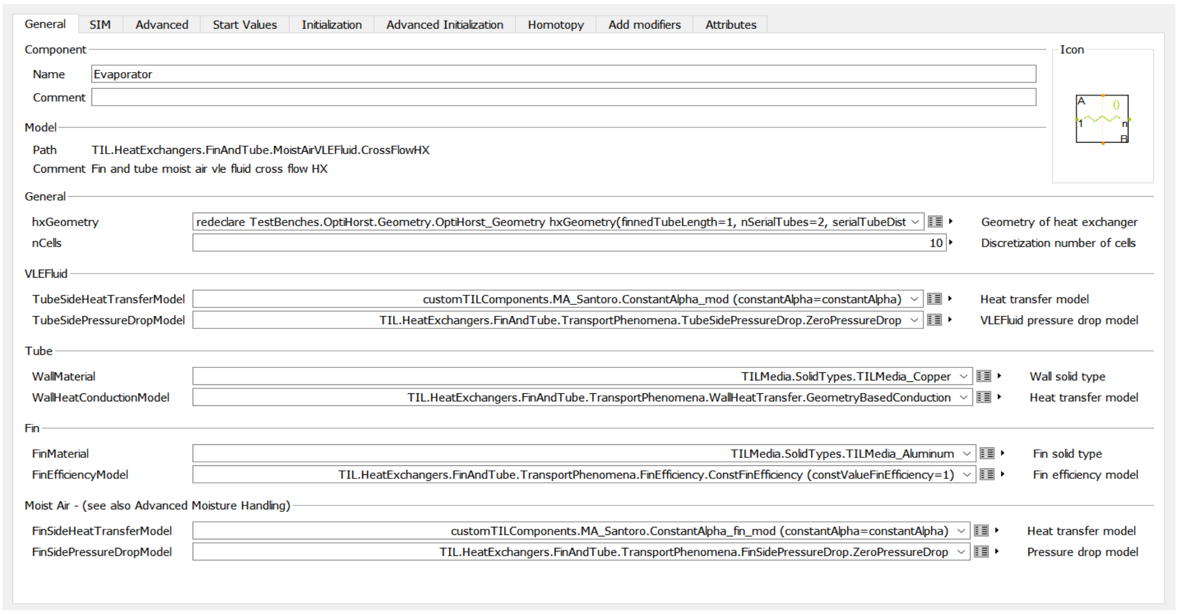
[80] Yabin Guo, Zehan Tan, Huanxin Chen, Guannan Li, Jiangyu Wang, Ronggeng Huang, Jiangyan Liu, and Tanveer Ahmad. Deep learning-based fault diagnosis of variable refrigerant flow air-conditioning system for building energy saving. *Applied Energy*, 225:732–745, 2018.

[81] Rony Shohet, Mohamed S. Kandil, and J. J. McArthur. Machine learning algorithms for classification of boiler faults using a simulated dataset. *IOP Conference Series: Materials Science and Engineering*, 609(6):062007, 2019.

[82] Clayton Miller, Zoltán Nagy, and Arno Schlueter. A review of unsupervised statistical learning and visual analytics techniques applied to performance analysis of non-residential buildings. *Renewable and Sustainable Energy Reviews*, 81:1365–1377, 2018.

Appendix

A Appendixes to Chapter 3



Component

Name: Evaporator

Comment:

Model

Path: TIL.HeatExchangers.FinAndTube.MoistAirVLEFluid.CrossFlowHX

Comment: Fin and tube moist air vle fluid cross flow HX

General

hxGeometry	redeclare TestBenches.OptiHorst.Geometry.OptiHorst_Geometry hxGeometry(finnedTubeLength=1, nSerialTubes=2, serialTubeDist	Geometry of heat exchanger
nCells	10	Discretization number of cells

VLEFluid

TubeSideHeatTransferModel	customTILComponents.MA_Santoro.ConstantAlpha_mod (constantAlpha=constantAlpha)	Heat transfer model
TubeSidePressureDropModel	TIL.HeatExchangers.FinAndTube.TransportPhenomena.TubeSidePressureDrop.ZeroPressureDrop	VLEFluid pressure drop model

Tube

WallMaterial	TILMedia.SolidTypes.TILMedia_Copper	Wall solid type
WallHeatConductionModel	TIL.HeatExchangers.FinAndTube.TransportPhenomena.WallHeatTransfer.GeometryBasedConduction	Heat transfer model

Fin

FinMaterial	TILMedia.SolidTypes.TILMedia_Aluminum	Fin solid type
FinEfficiencyModel	TIL.HeatExchangers.FinAndTube.TransportPhenomena.FinEfficiency.ConstFinEfficiency (constValueFinEfficiency=1)	Fin efficiency model

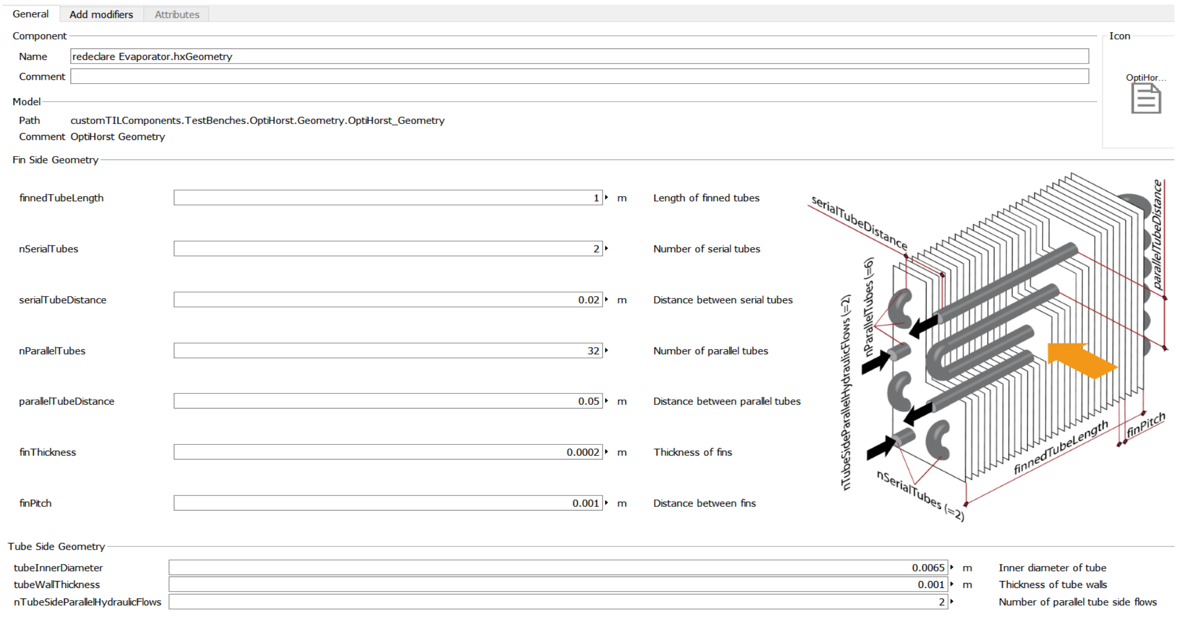
Moist Air - (see also Advanced Moisture Handling)

FinSideHeatTransferModel	customTILComponents.MA_Santoro.ConstantAlpha_fin_mod (constantAlpha=constantAlpha)	Heat transfer model
FinSidePressureDropModel	TIL.HeatExchangers.FinAndTube.TransportPhenomena.FinSidePressureDrop.ZeroPressureDrop	Pressure drop model

Figure A.1: Evaporator component general configuration.

Table A.1: Summary of computational power

Category	Item	Specification
Computer's Hardware	RAM	8 Gb
	Processor	Intel(R) Core(TM) i5-3570 CPU @ 3.40GHz
Computer's Software	Operating System	Windows 10 Education 64 bits
	Dymola	version 2021x
	TIL Suite	version 3.12.0
	AixLib	version 1.2.1
	Python	version 3.9.13
	Jupyter Notebook	version 6.4.12
	Scikit-Learn	version 1.0.2
	Hyperopt	version 0.2.7
	Hyperopt-Scikitlearn	version 1.0.3
	ADDMo	version 0.1



General | Add modifiers | Attributes

Component

Name: redeclare Evaporator.hxGeometry

Comment:

Icon: OptiHor...

Model

Path: customTILComponents.TestBenches.OptiHorst.Geometry.OptiHorst_Geometry

Comment: OptiHorst Geometry

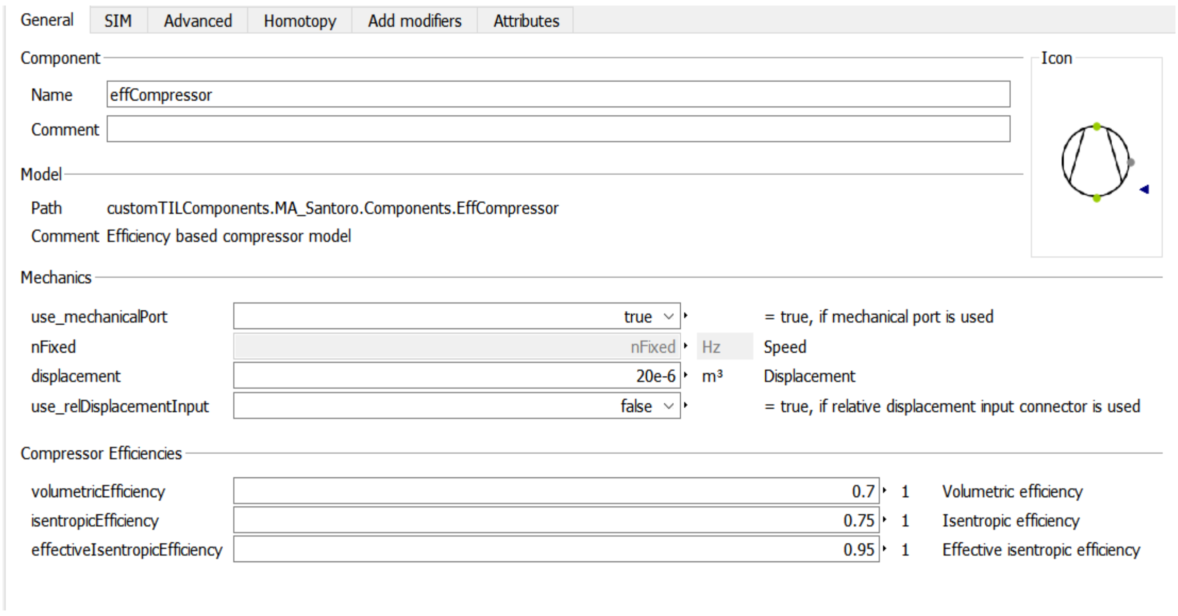
Fin Side Geometry

finnedTubeLength	1	m	Length of finned tubes
nSerialTubes	2		Number of serial tubes
serialTubeDistance	0.02	m	Distance between serial tubes
nParallelTubes	32		Number of parallel tubes
parallelTubeDistance	0.05	m	Distance between parallel tubes
finThickness	0.0002	m	Thickness of fins
finPitch	0.001	m	Distance between fins

Tube Side Geometry

tubeInnerDiameter	0.0065	m	Inner diameter of tube
tubeWallThickness	0.001	m	Thickness of tube walls
nTubeSideParallelHydraulicFlows	2		Number of parallel tube side flows

Figure A.2: Evaporator component geometry specifications.



General | SIM | Advanced | Homotopy | Add modifiers | Attributes

Component

Name: effCompressor

Comment:

Icon

Model

Path: customTILComponents.MA_Santoro.Components.EffCompressor

Comment: Efficiency based compressor model

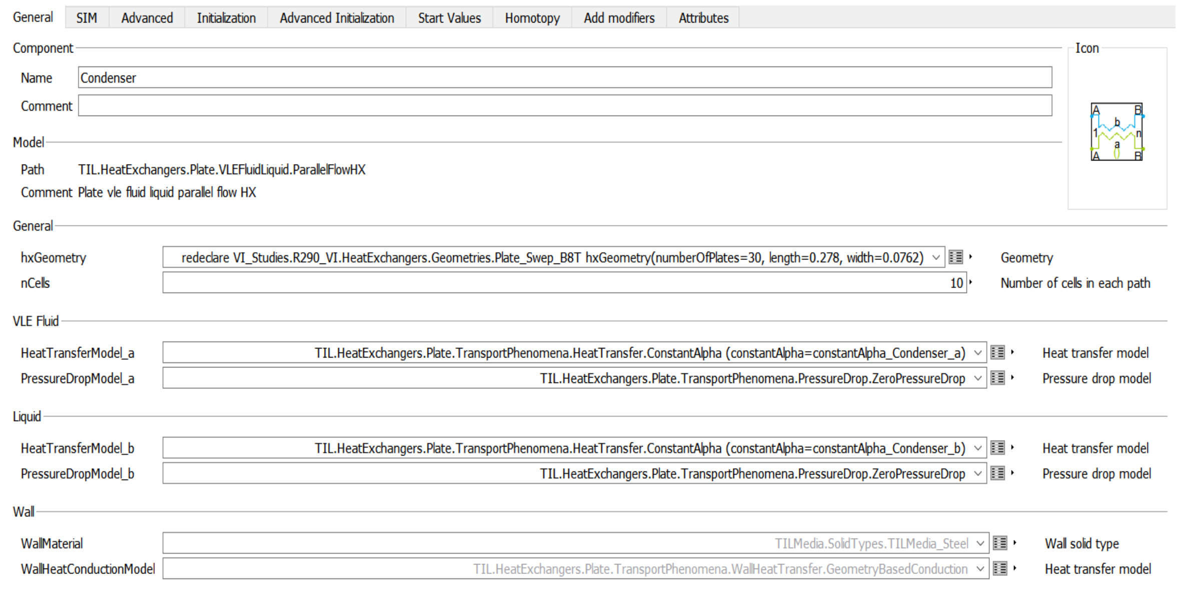
Mechanics

use_mechanicalPort	true		= true, if mechanical port is used
nFixed	nFixed	Hz	Speed
displacement	20e-6	m³	Displacement
use_relDisplacementInput	false		= true, if relative displacement input connector is used

Compressor Efficiencies

volumetricEfficiency	0.7	1	Volumetric efficiency
isentropicEfficiency	0.75	1	Isentropic efficiency
effectiveIsentropicEfficiency	0.95	1	Effective isentropic efficiency

Figure A.3: Evaporator component geometry specifications.


General | SIM | Advanced | Initialization | Advanced Initialization | Start Values | Homotopy | Add modifiers | Attributes

Component

Name	Condenser
Comment	

Icon

Model

Path	TIL.HeatExchangers.Plate.VLEFluidLiquid.ParallelFlowHX
Comment	Plate vle fluid liquid parallel flow HX

General

hxGeometry	redeclare VI_Studies.R290_VI.HeatExchangers.Geometries.Plate_Swep_B8T hxGeometry(numberOfPlates=30, length=0.278, width=0.0762)	Geometry
nCells	10	Number of cells in each path

VLE Fluid

HeatTransferModel_a	TIL.HeatExchangers.Plate.TransportPhenomena.HeatTransfer.ConstantAlpha (constantAlpha=constantAlpha_Condenser_a)	Heat transfer model
PressureDropModel_a	TIL.HeatExchangers.Plate.TransportPhenomena.PressureDrop.ZeroPressureDrop	Pressure drop model

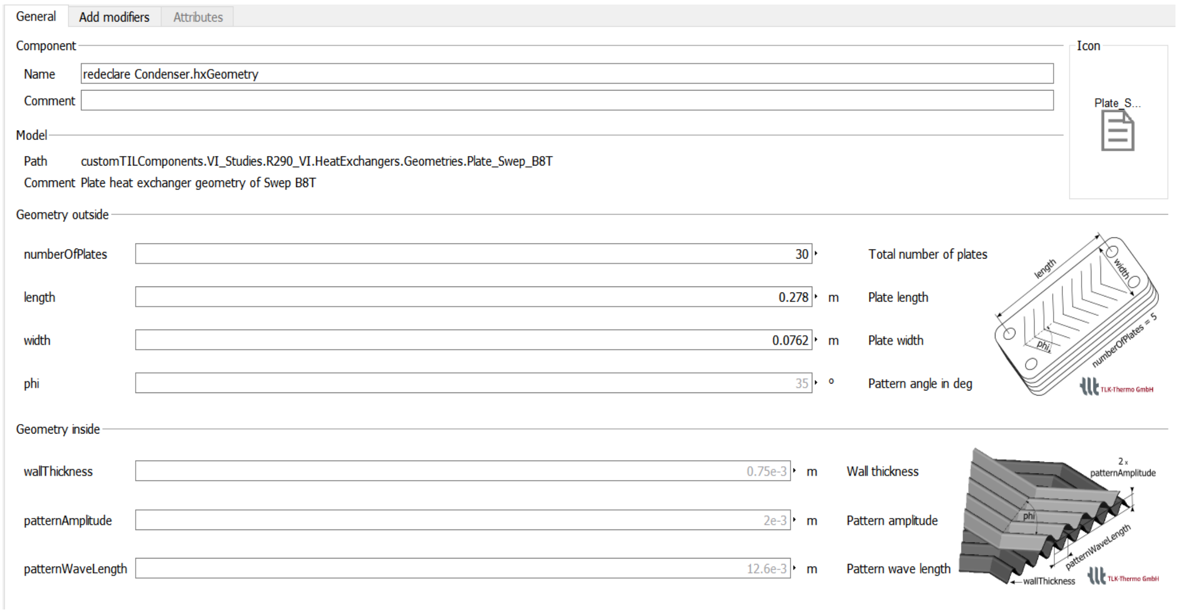
Liquid

HeatTransferModel_b	TIL.HeatExchangers.Plate.TransportPhenomena.HeatTransfer.ConstantAlpha (constantAlpha=constantAlpha_Condenser_b)	Heat transfer model
PressureDropModel_b	TIL.HeatExchangers.Plate.TransportPhenomena.PressureDrop.ZeroPressureDrop	Pressure drop model

Wall

WallMaterial	TILMedia.SolidTypes.TILMedia_Steel	Wall solid type
WallHeatConductionModel	TIL.HeatExchangers.Plate.TransportPhenomena.WallHeatTransfer.GeometryBasedConduction	Heat transfer model

Figure A.4: Condenser component general configuration.


General | Add modifiers | Attributes

Component

Name	redeclare Condenser.hxGeometry
Comment	

Icon

Plate_S...

Model

Path	customTILComponents.VI_Studies.R290_VI.HeatExchangers.Geometries.Plate_Swep_B8T
Comment	Plate heat exchanger geometry of Swep B8T

Geometry outside

numberOfPlates	30		Total number of plates
length	0.278	m	Plate length
width	0.0762	m	Plate width
phi	35	°	Pattern angle in deg

Geometry inside

wallThickness	0.75e-3	m	Wall thickness
patternAmplitude	2e-3	m	Pattern amplitude
patternWaveLength	12.6e-3	m	Pattern wave length

Figure A.5: Condenser component geometry specifications.

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import StratifiedKFold, train_test_split
from sklearn.model_selection import GridSearchCV, cross_val_score
from sklearn.metrics import confusion_matrix, classification_report, make$_s$corer
from sklearn.metrics import balanced_accuracy_score
np.random_seed(42)
Loading data and preprocessing
dataset = pd.read_excel(”dataset.xlsx”)
X = dataset.iloc[:, : −1]
y = dataset.iloc[:, −1]
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size = 0.3, random_state = 42, shuffle = True, stratify = y)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
    test_size = 0.3, random_state = 42, shuffle = True, stratify = y_train)
scaler = StandardScaler()                    ▷ Samples are centered and scaled with this method
X_train_scaled = scaler.fit_transform(X_train.to_numpy(dtype = ”float64”))
X_val_scaled = scaler.transform(X_val.to_numpy(dtype = ”float64”))
X_test_scaled = scaler.transform(X_test.to_numpy(dtype = ”float64”))
y_train = y_train.to_numpy(dtype = ”float64”)
y_val = y_val.to_numpy(dtype = ”float64”)
Training
tree_clf = DecisionTreeClassifier(max_depth = 2, random_state = 42)
tree_clf.fit(X_train, y_train)
y_pred = tree_clf.predict(X_val)
Evaluation of predictions
cm = confusion_matrix(y_val, y_pred)
report = classification_report(y_val, y_pred,
    target_names = [”Fault”, ”NoFault”])
ba = balanced_accuracy_score(y_val, y_pred)
ba_scorer = make_scorer(balanced_accuracy_score)
skf = StratifiedKFold(n_splits = 10, shuffle = True, random_state = 42)
tree_CV_scores = cross_val_score(tree_clf, X_val, y_val,
    scoring = ba_scorer, cv = skf)
tree_CV_scores_mean = tree_scores.mean()
tree_CV_scores_sigma = tree_scores.std()
Optimization
param_grid = [{′max_depth′ : [2, 4, 6], ′max_features′ : [2, 4, 6],
    ′criterion′ : [′gini′,′ entropy′], ′max_leaf_nodes′ : [2, 4, 6],
    ′min_samples_leaf′ : [2, 4, 6], ′min_samples_split′ : [2, 4, 6]}]
grid_search = GridSearchCV(tree_clf, param_grid, cv = 10,
    scoring = ba_scorer, return_train_score = True)
grid_search.fit(X_train, y_train)
opt_tree = grid_search.best_estimator_
opt_tree_params = grid_search.best_params_
```

Figure A.6: Classifier training example.

```
.                                                              ▷ Same imports as the classifier
.
.
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import r2_score, mean_squared_error
Loading data and preprocessing
y = dataset.loc[:, "Airvolumeflow"]
.                                                              ▷ Same steps as the classifier
.
.
scaler_sample = MinMaxScaler()                  ▷ Samples and signals are compressed in range
scaler_signal = MinMaxScaler()
.
.
.
y_train = y_train.to_numpy(dtype = "float64")
y_val = y_val.to_numpy(dtype = "float64")
y_test = y_val.to_numpy(dtype = "float64")
y_train_scaled = scaler_signal.fit_transform(y_train.reshape(−1, 1))    ▷ Signal arrays must
be shaped as column-vectors
y_val_scaled = scaler_signal.fit_transform(y_val.reshape(−1, 1))
y_test_scaled = scaler_signal.fit_transform(y_test.reshape(−1, 1))
Training
tree_reg = DecisionTreeRegressor(max_depth = 2, random_state = 42)
tree_reg.fit(X_train, y_train)
y_pred = tree_reg.predict(X_val)
Evaluation of predictions
r2 = r2_squared_error(y_val, y_pred)
rmse = mean_squared_error(y_val, y_pred
    , squared = False)                  ▷ squared=False → RMSE, squared= True → MSE
tree_CV_scores = cross_val_score(tree_reg, X_val, y_val, cv = 10,
    scoring = "neg_mean_squared_error"")          ▷ With -RMSE, 0 is the greatest value
tree_CV_scores_mean = tree_scores.mean()
tree_CV_scores_sigma = tree_scores.std()
Optimization
.
.
.
grid_search = GridSearchCV(tree_reg, param_grid, cv = 10,
    scoring = "neg_mean_squared_error", return_train_score = True)
grid_search.fit(X_train, y_train)
opt_tree = grid_search.best_estimator_
opt_tree_params = grid_search.best_params_
```

Figure A.7: Regressor training example.

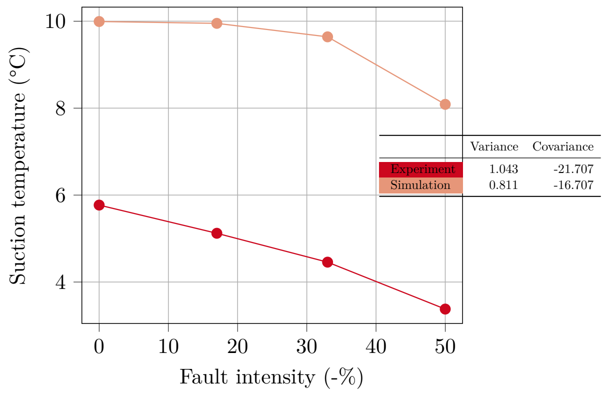
```
.                                                    ▷ Same imports as any estimator
.
.
from hyperopt import hp, tpe
from hpsklearn import HyperoptEstimator, mlp_classifier, any_preprocessing
from hyperopt.pyll import scope

Loading data and preprocessing
.
.
.

Optimization
hdl_search_space = hp.choice('hidden_layer_sizes',
   [scope.int(hp.qloguniform("1.1", np.log(1), np.log(1000), 1))])
clf = mlp_classifier("my_mlp", random_state = 42, shuffle = True,
   solver =' adam', hidden_layer_sizes = hl_search_space)
mlp_opt_clf = HyperoptEstimator(classifier = clf, regressor = None,
   preprocessing = any_preprocessing("my_pre"), algo = tpe.suggest,
   max_evals = 200, trial_timeout = 500, seed = np.random.seed(42), n_jobs = −1)
mlp_opt_clf.fit(X_train_scaled, y_train)

Evaluation of predictions
.                                    ▷ Calculation of metrics depending on the estimator
.
.
```

Figure A.8: Estimator optimization example.

B Appendixes to Chapter 4

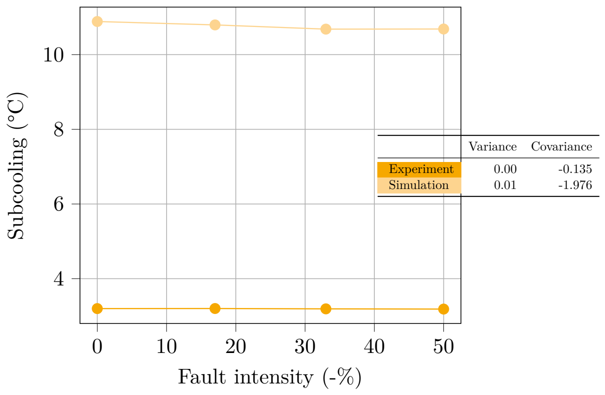
Table B.1: Points Matrix results

Classifier	Features	Hyperparameters	BA PoM	BA Y	BA Exp
DT	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, T_{cond}, T_{evap}, T_{SC}, T_{SH}, $\dot{W}$, $\dot{m}_{ref}$, $\dot{V}_{a,vs}$, $\dot{Q}_h$, COP	criterion = gini, max_depth = 2, min_samples_leaf = 1, min_samples_split = 2	0,982	0,57	0,51
DT	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$,COP	max_depth=4, max_features=5, max_leaf_nodes=6, min_samples_leaf=6, min_samples_split=2,	0,62	-	-
SVM	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, T_{cond}, T_{evap}, T_{SC}, T_{SH}, $\dot{W}$, $\dot{m}_{ref}$,$\dot{V}_{a,vs}$, $\dot{Q}_h$, COP	C = 50, kernel = poly degree = 3	0,75	-	-
SVM	$T_{evap,i}$, T_{suc}, $T_{w,o}$, COP	C = 60, kernel = rbf	0,42	-	-
SVM	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$, COP	C = 100, kernel = rbf	0,52	-	-
SVM-OPT	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, T_{cond}, T_{evap}, T_{SC}, T_{SH}, $\dot{W}$, $\dot{m}_{ref}$,$\dot{V}_{a,vs}$, $\dot{Q}_h$, COP	C = 1,165; coef0 = 0,383; kernel = linear; degree = 5; tol=0.000116	1,0	0,53	0,53
SVM-OPT	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$, COP	C=388; coef0=0.044; decision_function_shape='ovo'; degree=2; gamma='auto'; kernel='linear'; shrinking=False	1,0	0,57	-
SVM-OPT	n_{comp}, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$, COP	C=388; coef0=0.044; decision$_function_shape =' ovo'$; degree=2; gamma='auto'; kernel='linear'; shrinking=False	0,85	0,54	0,5
MLP	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, T_{cond}, T_{evap}, T_{SC}, T_{SH}, $\dot{W}$, $\dot{m}_{ref}$,$\dot{V}_{a,vs}$, $\dot{Q}_h$, COP	activation='tanh' hidden_layer_sizes=(31, 12) learning_rate='adaptive' beta_1=0.891, beta_2=0.966, momentum=0.809 power_t=0.555	1,0	0,58	0,48
MLP	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$, COP	activation='identity' beta_1=0.962, beta_2=0.994, hidden_layer_sizes=(836, 11, 30), momentum=0.921, power_t=0.486	1,0	0,5	-
MLP	n_{comp}, $T_{a,o}$, T_{amb}, $T_{cond,o}$, T_{dis}, $T_{evap,i}$, T_{suc}, $T_{w,i}$, $T_{w,o}$, $\dot{W}$, $\dot{Q}_h$, COP	activation='identity', beta_1=0.8816471453063571, beta_2=0.9761335729722351, hidden_layer_sizes=(2, 2), momentum=0.909, power_t=0.369	0,67	-	-



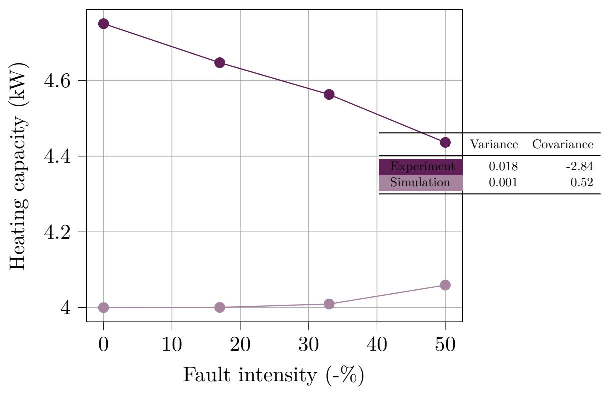
	Variance	Covariance
Experiment	1.043	-21.707
Simulation	0.811	-16.707

(a)



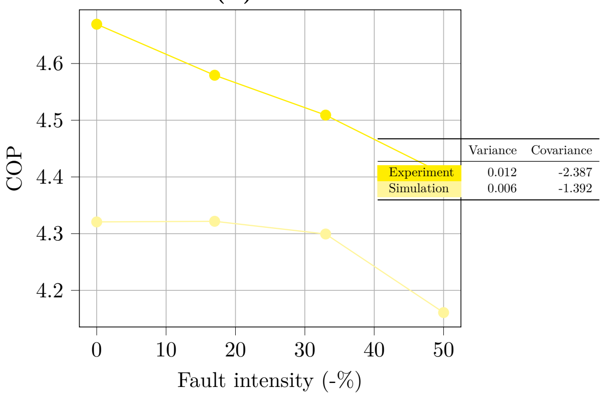
	Variance	Covariance
Experiment	0.00	-0.135
Simulation	0.01	-1.976

(b)



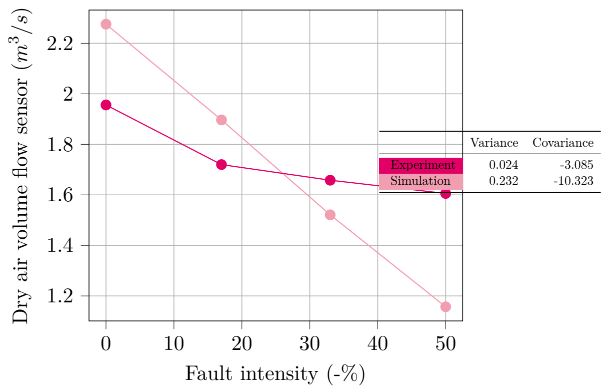
	Variance	Covariance
Experiment	0.018	-2.84
Simulation	0.001	0.52

(c)



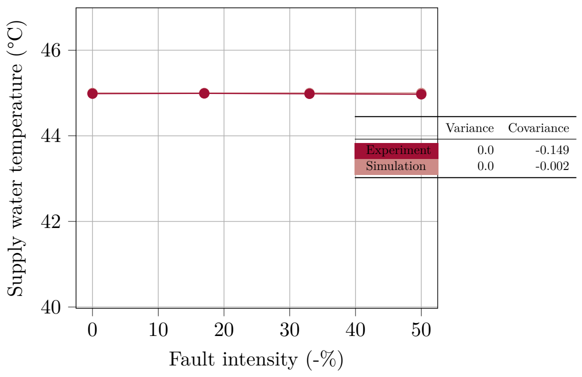
	Variance	Covariance
Experiment	0.012	-2.387
Simulation	0.006	-1.392

(d)



	Variance	Covariance
Experiment	0.024	-3.085
Simulation	0.232	-10.323

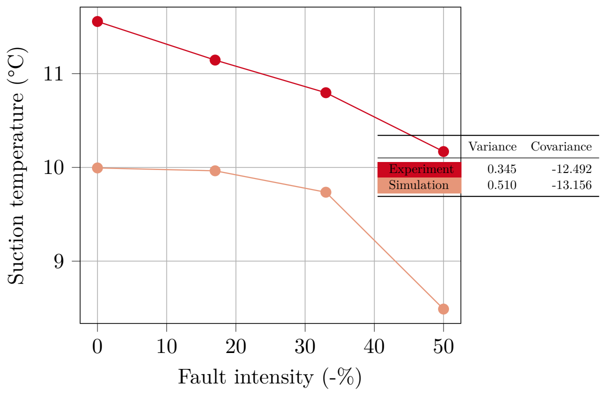
(e)



	Variance	Covariance
Experiment	0.0	-0.149
Simulation	0.0	-0.002

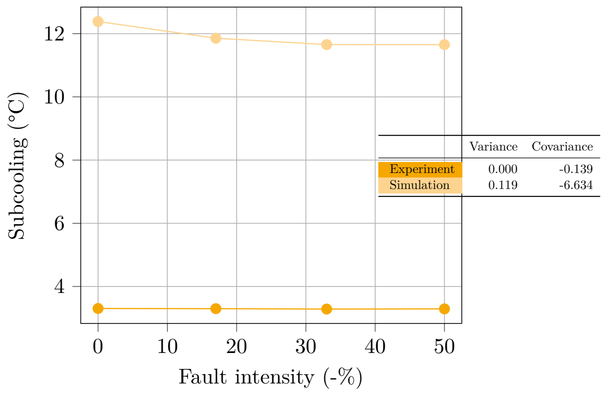
(f)

Figure B.1: Evaporator fouling experimental-simulation comparison at A10W45.



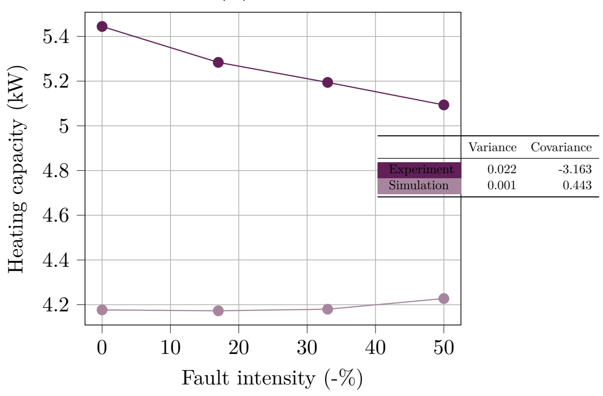
	Variance	Covariance
Experiment	0.345	-12.492
Simulation	0.510	-13.156

(a)



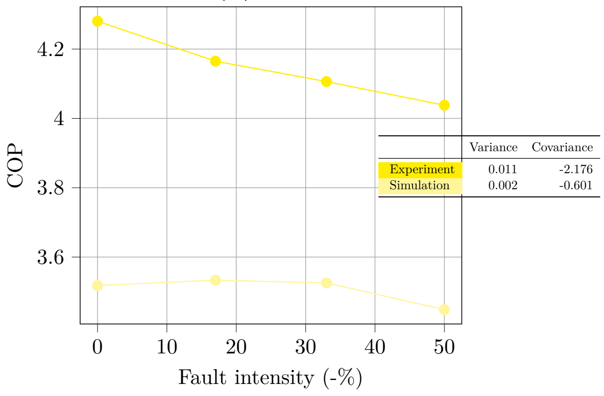
	Variance	Covariance
Experiment	0.000	-0.139
Simulation	0.119	-6.634

(b)



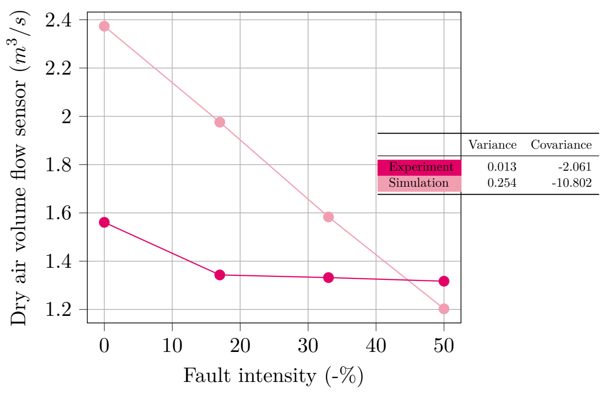
	Variance	Covariance
Experiment	0.022	-3.163
Simulation	0.001	0.443

(c)



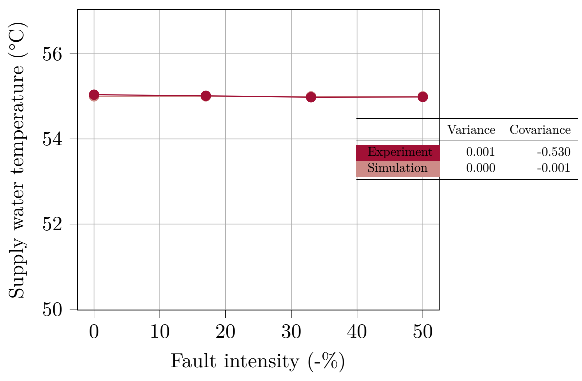
	Variance	Covariance
Experiment	0.011	-2.176
Simulation	0.002	-0.601

(d)



	Variance	Covariance
Experiment	0.013	-2.061
Simulation	0.254	-10.802

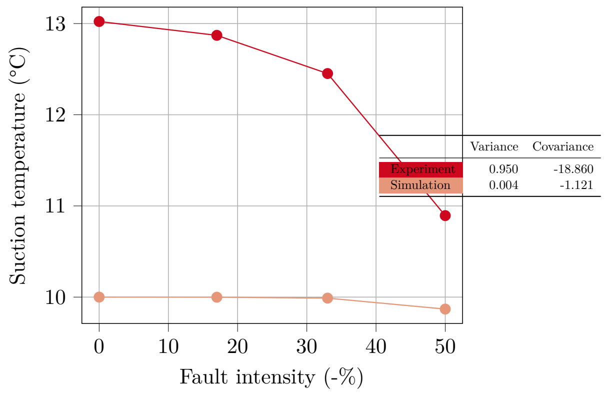
(e)



	Variance	Covariance
Experiment	0.001	-0.530
Simulation	0.000	-0.001

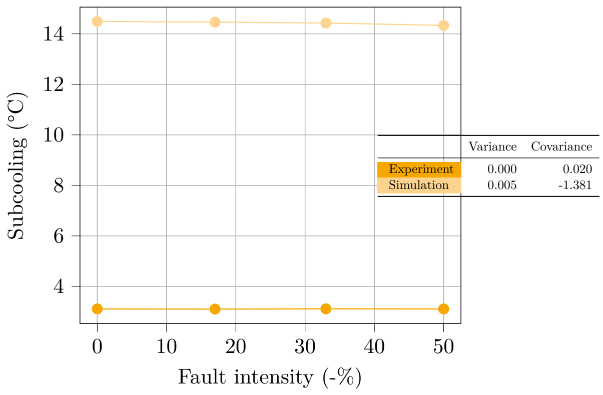
(f)

Figure B.2: Evaporator fouling experimental-simulation comparison at A10W55.



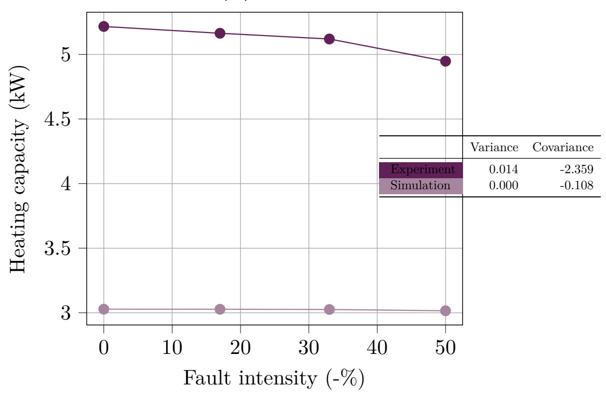
	Variance	Covariance
Experiment	0.950	-18.860
Simulation	0.004	-1.121

(a)



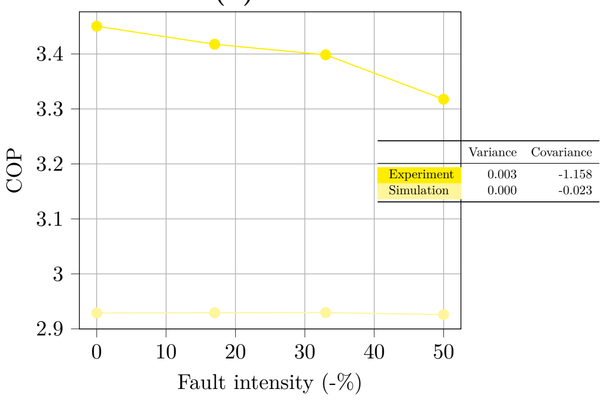
	Variance	Covariance
Experiment	0.000	0.020
Simulation	0.005	-1.381

(b)



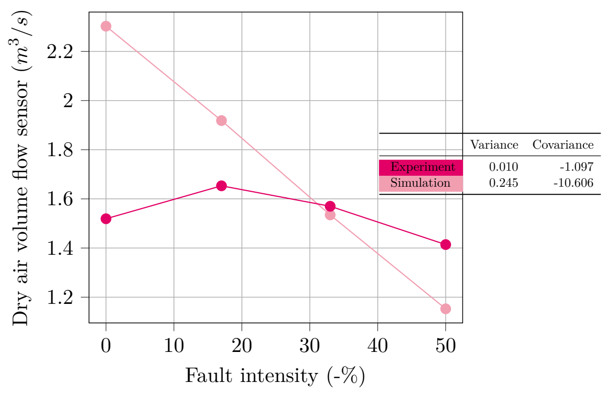
	Variance	Covariance
Experiment	0.014	-2.359
Simulation	0.000	-0.108

(c)



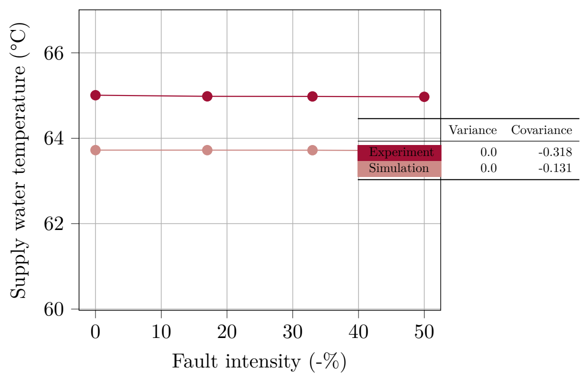
	Variance	Covariance
Experiment	0.003	-1.158
Simulation	0.000	-0.023

(d)



	Variance	Covariance
Experiment	0.010	-1.097
Simulation	0.245	-10.606

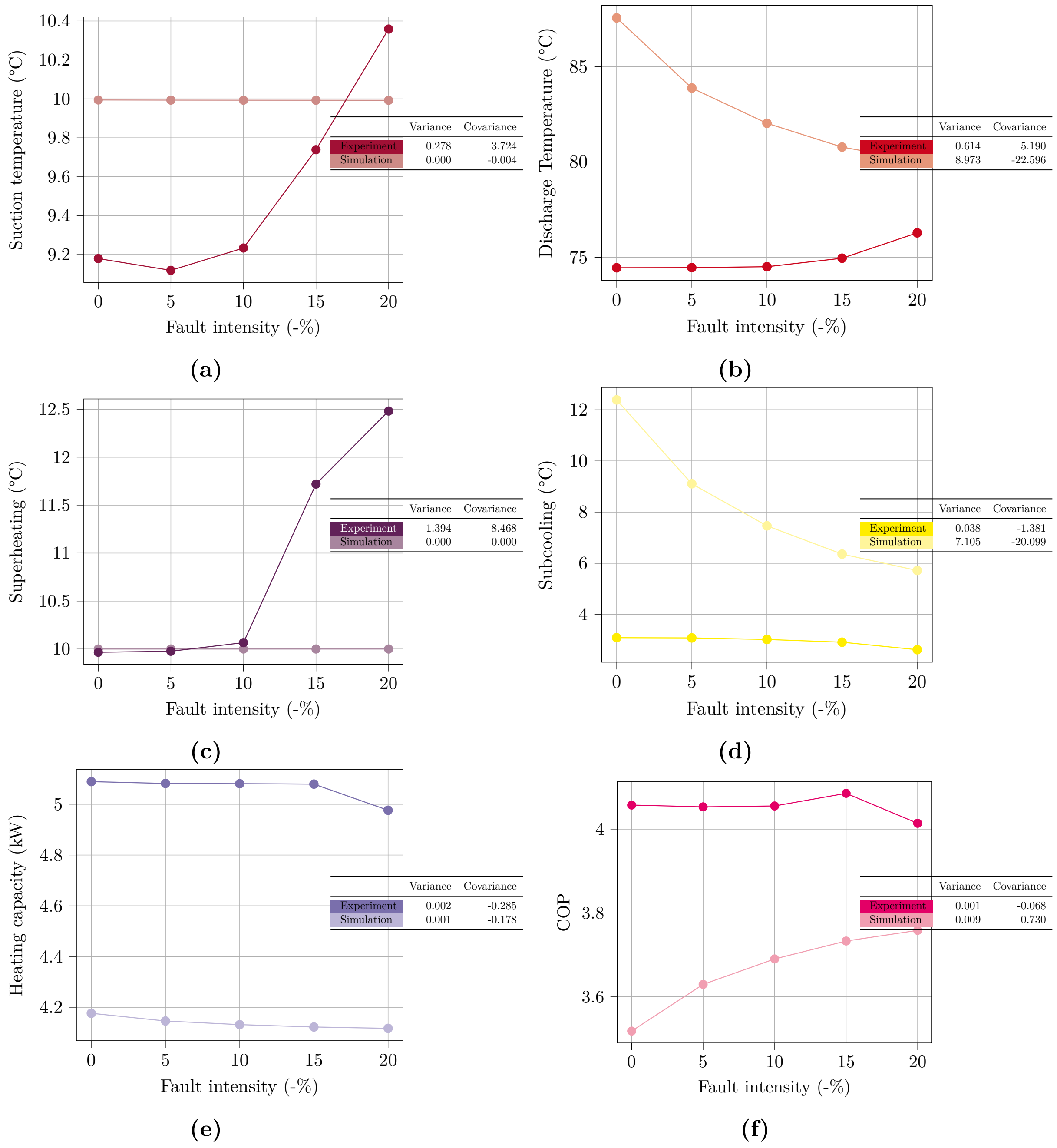
(e)



	Variance	Covariance
Experiment	0.0	-0.318
Simulation	0.0	-0.131

(f)

Figure B.3: Evaporator fouling experimental-simulation comparison at A10W65.



(a)

	Variance	Covariance
Experiment	0.278	3.724
Simulation	0.000	-0.004

(b)

	Variance	Covariance
Experiment	0.614	5.190
Simulation	8.973	-22.596

(c)

	Variance	Covariance
Experiment	1.394	8.468
Simulation	0.000	0.000

(d)

	Variance	Covariance
Experiment	0.038	-1.381
Simulation	7.105	-20.099

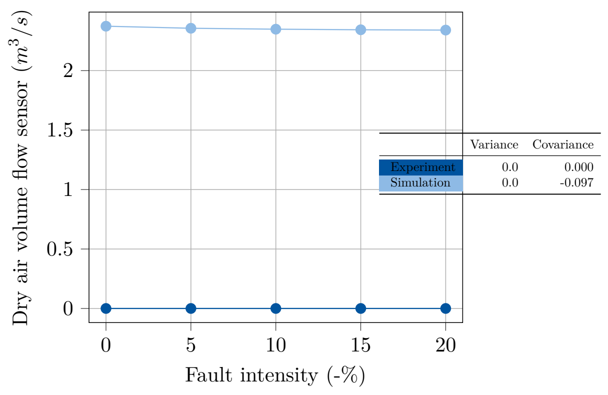
(e)

	Variance	Covariance
Experiment	0.002	-0.285
Simulation	0.001	-0.178

(f)

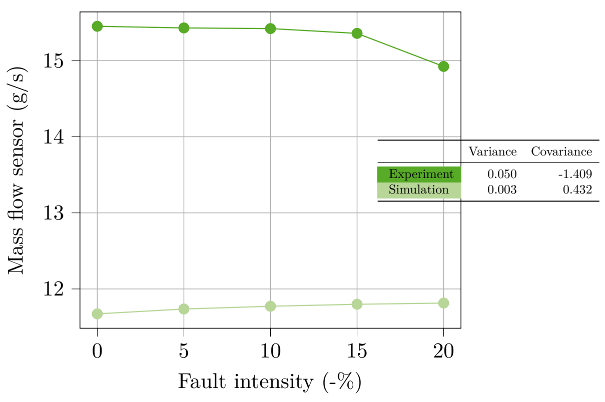
	Variance	Covariance
Experiment	0.001	-0.068
Simulation	0.009	0.730

Figure B.4: Refrigerant leakage experimental-simulation comparison at A10W55.



	Variance	Covariance
Experiment	0.0	0.000
Simulation	0.0	-0.097

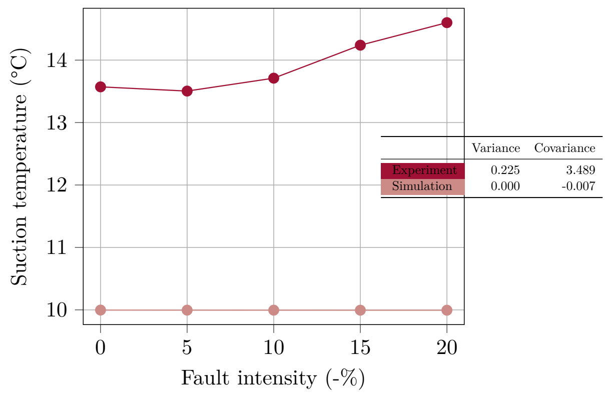
(g)



	Variance	Covariance
Experiment	0.050	-1.409
Simulation	0.003	0.432

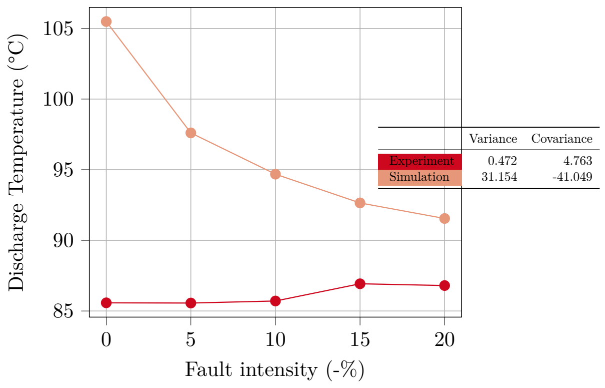
(h)

Figure B.4: No-fault experimental-simulation comparison at A10W55 (cont.).



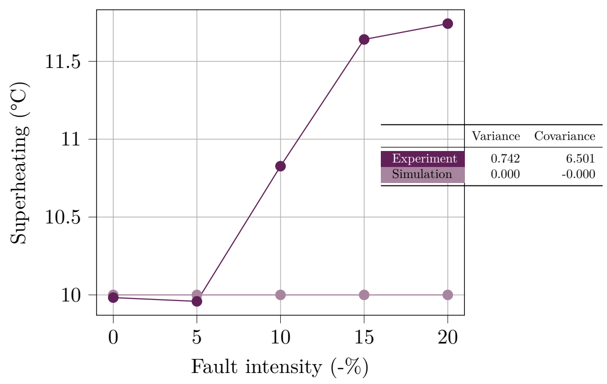
	Variance	Covariance
Experiment	0.225	3.489
Simulation	0.000	-0.007

(a)



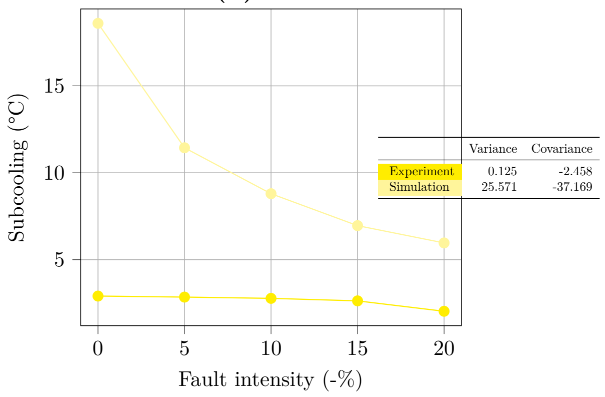
	Variance	Covariance
Experiment	0.472	4.763
Simulation	31.154	-41.049

(b)



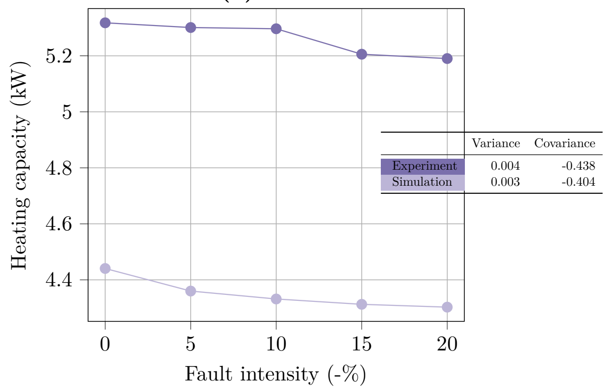
	Variance	Covariance
Experiment	0.742	6.501
Simulation	0.000	-0.000

(c)



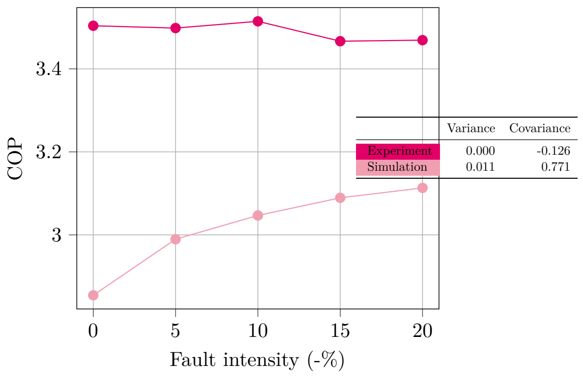
	Variance	Covariance
Experiment	0.125	-2.458
Simulation	25.571	-37.169

(d)



	Variance	Covariance
Experiment	0.004	-0.438
Simulation	0.003	-0.404

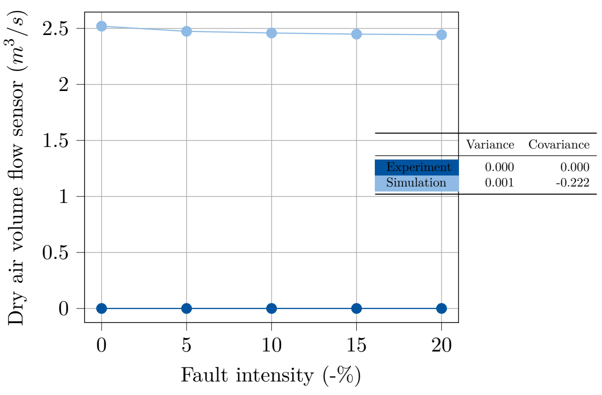
(e)



	Variance	Covariance
Experiment	0.000	-0.126
Simulation	0.011	0.771

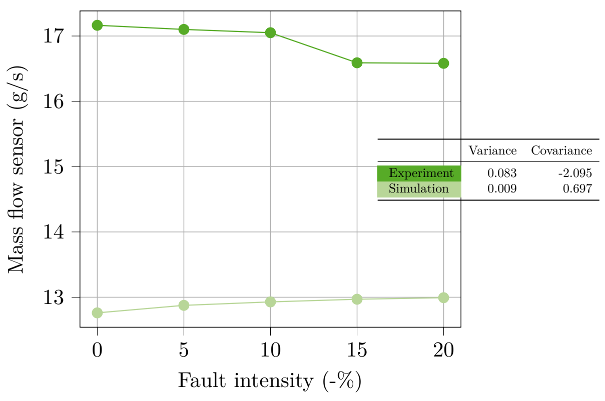
(f)

Figure B.5: Refrigerant leakage experimental-simulation comparison at A10W65.



	Variance	Covariance
Experiment	0.000	0.000
Simulation	0.001	-0.222

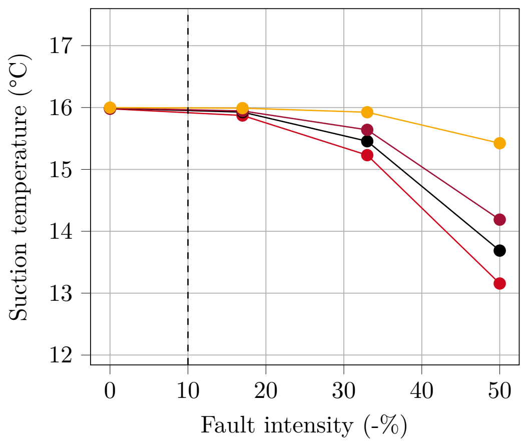
(g)



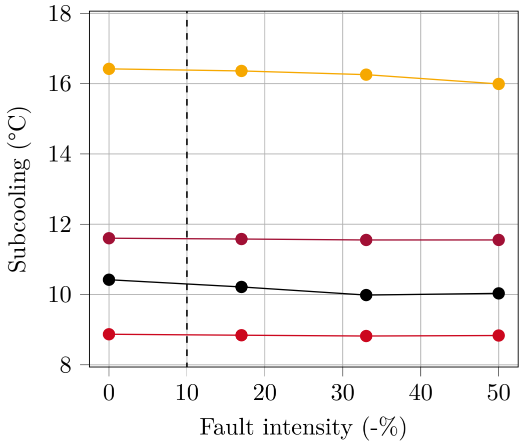
	Variance	Covariance
Experiment	0.083	-2.095
Simulation	0.009	0.697

(h)

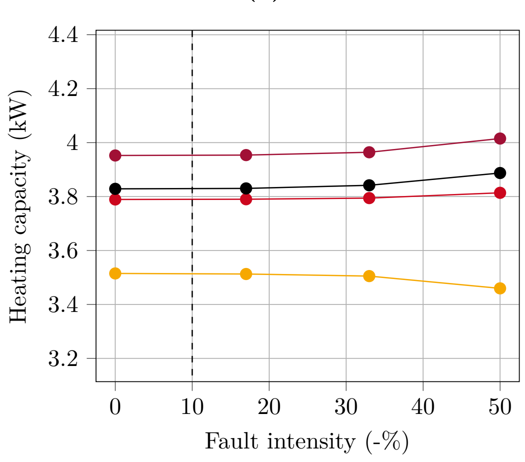
Figure B.5: No-fault experimental-simulation comparison at A10W65 (cont.).



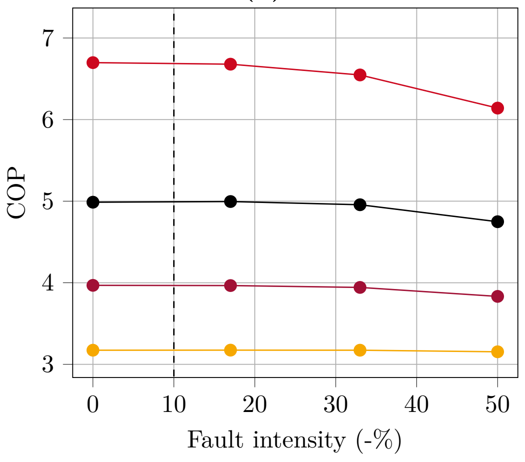
(a)



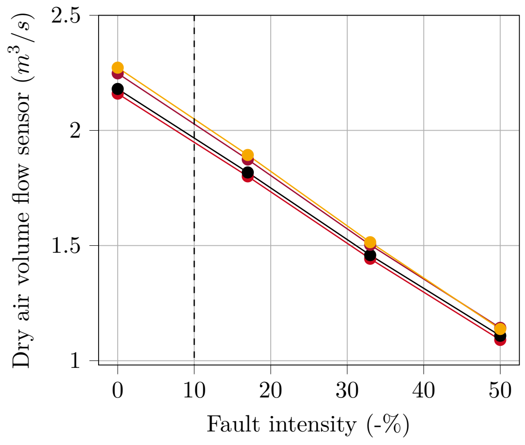
(b)



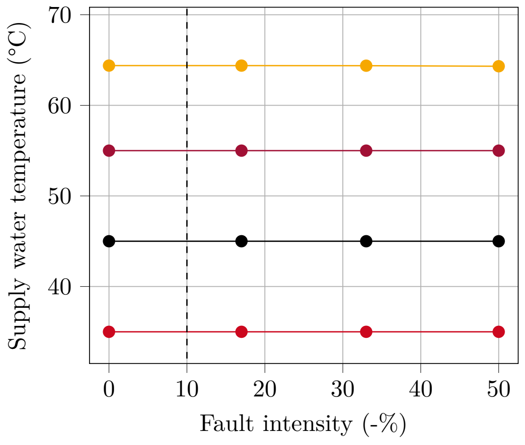
(c)



(d)

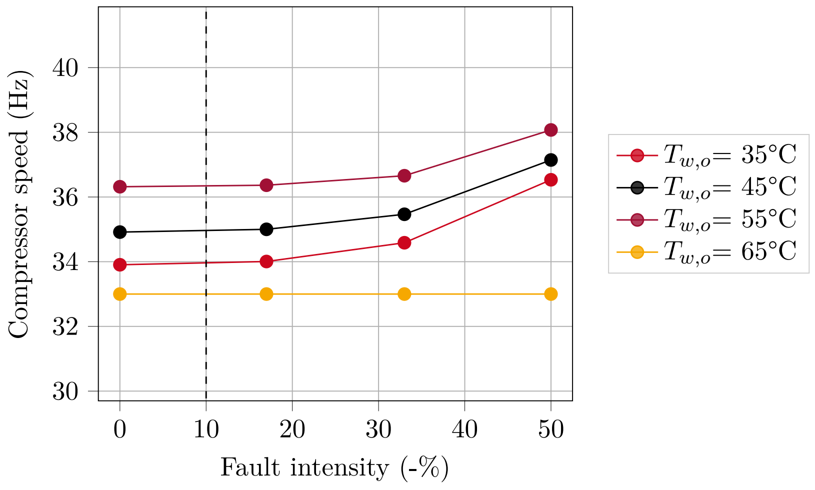


(e)



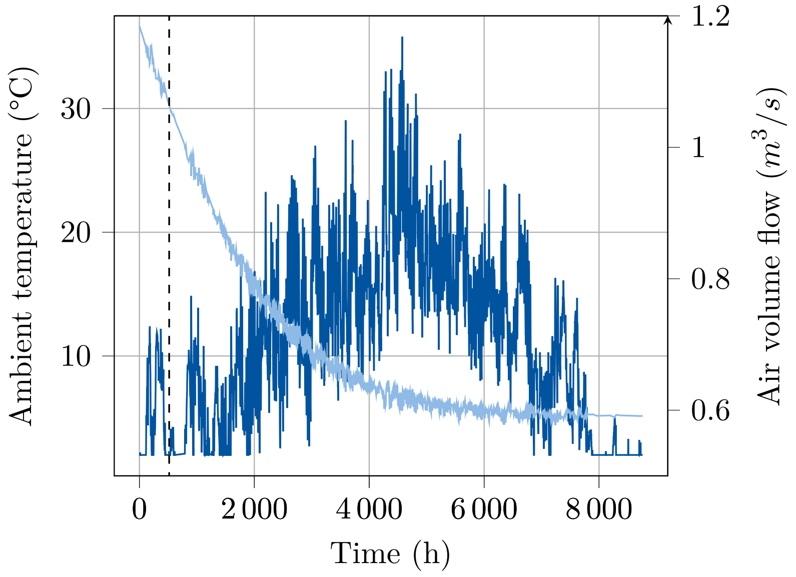
(f)

Figure B.6: Evaporator fouling effects summary at $T_{amb} = 16$ °C.

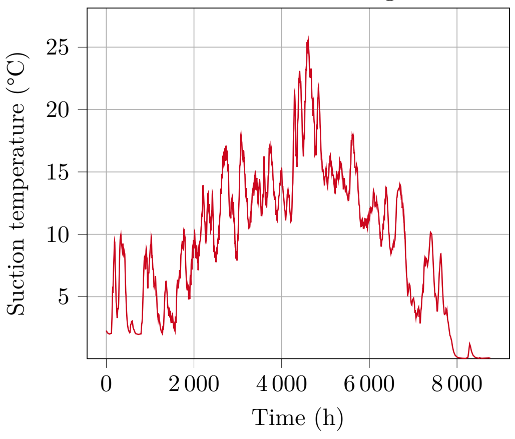


(g)

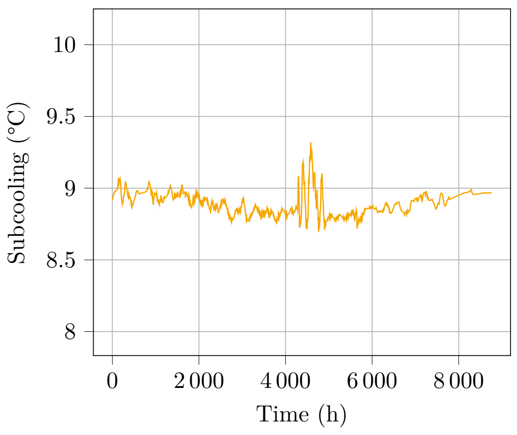
Figure B.6: Evaporator fouling effects summary at $T_{amb} = 16$ °C (cont).



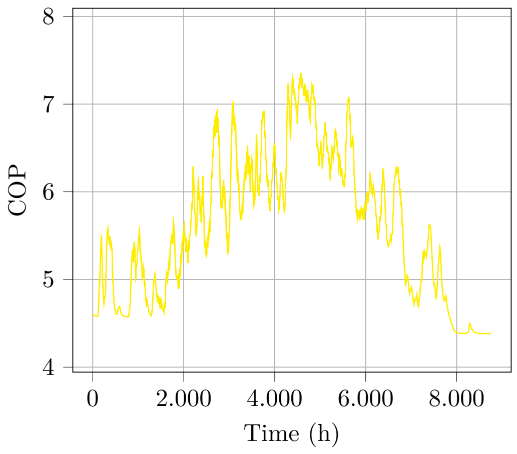
(a) Where the ambient temperature is depicted with blue and the air volume flow with light blue.



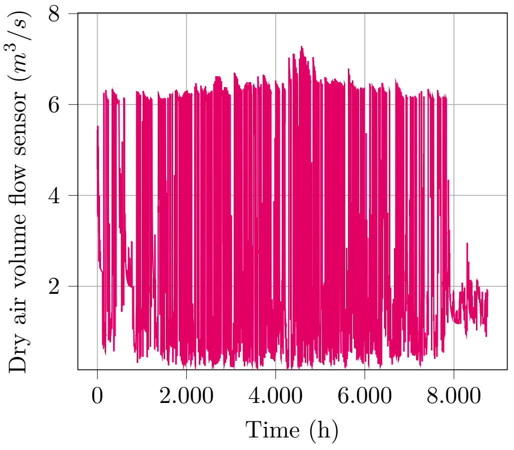
(b)



(c)

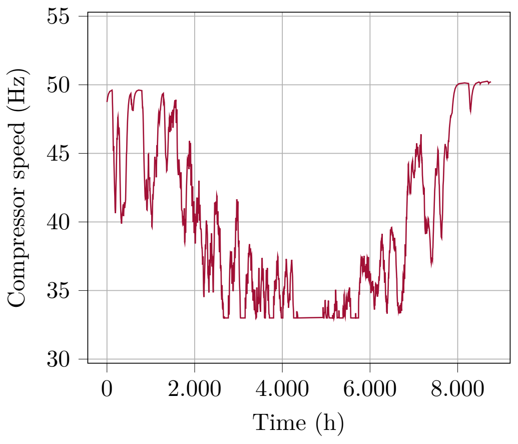


(d)



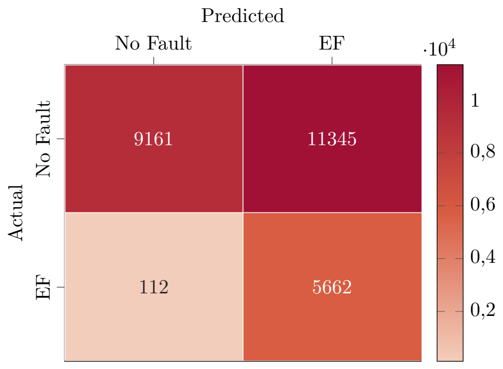
(e)

Figure B.7: Evaporator fouling effects summary at $T_{w,o} = 35$ °C on a typical year simulation with 4000 h to max fault EF.

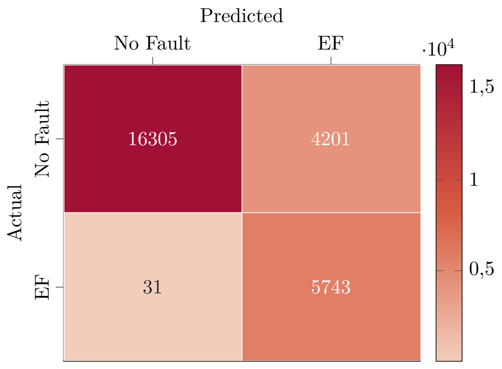


(f)

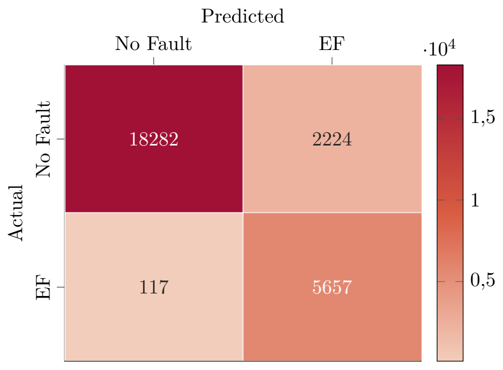
Figure B.7: Evaporator fouling effects summary at $T_{w,o} = 35$ °C on a typical year simulation with 4000 h to max fault EF (cont).



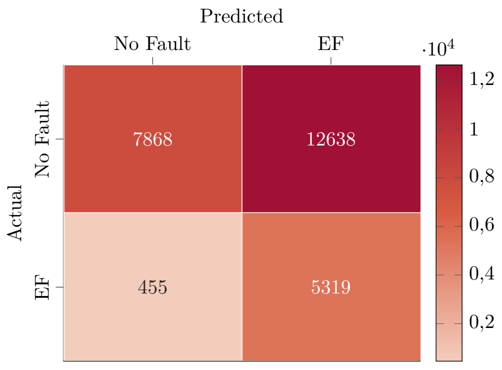
(a) $T_{w,o}$=35°C



(b) $T_{w,o}$=45°C

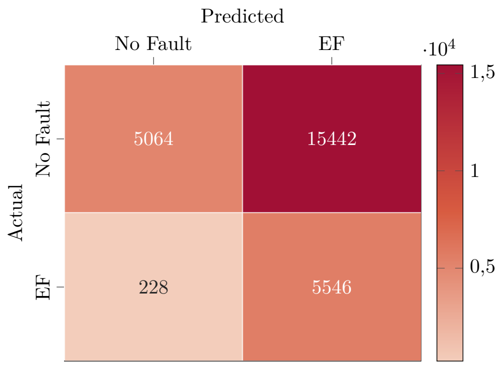


(c) $T_{w,o}$=55°C

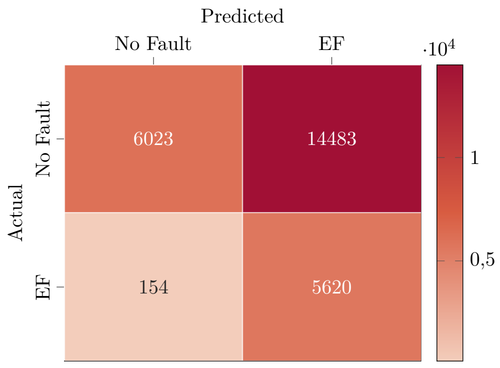


(d) $T_{w,o}$=65°C

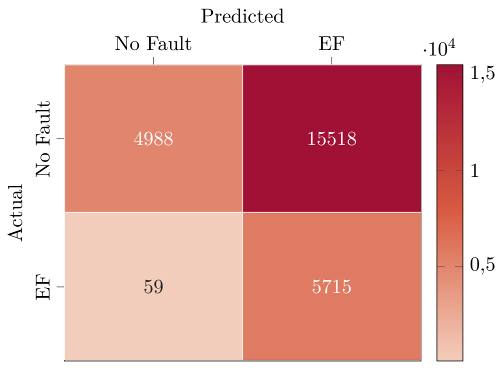
Figure B.8: MLP regressor trained with 16.000 h + 4.000 h performance on evaluation set.



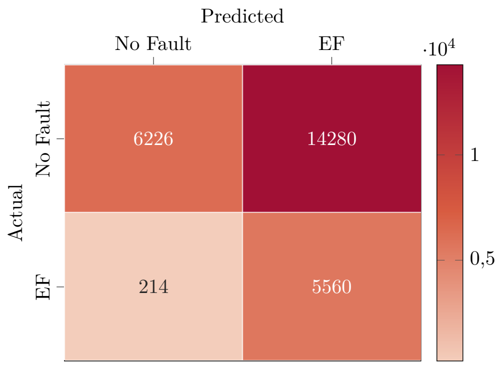
(a) $T_{w,o}$=35°C



(b) $T_{w,o}$=45°C



(c) $T_{w,o}$=55°C



(d) $T_{w,o}$=65°C

Figure B.9: RF classifier performance on evaluation set.

Declaration of Originality

I hereby declare that this thesis and the work reported herein was composed by and originated entirely from me. Information derived from the published and unpublished work of others has been acknowledged in the text and references are given in the list of sources. This thesis has not been submitted as exam work in neither the same nor a similar form. I agree that this thesis may be stored in the institute's library and database. This work may also be copied for internal use.

Aachen, Tuesday 8th August, 2023

Francisco Santoro Delgado