A Fast and Accurate Predictive Model for Thermal Environments Using Machine Learning

Jess, Brandi Jo

Award date:
2023

Awarding institution:
Coventry University

A FAST AND ACCURATE PREDICTIVE MODEL FOR THERMAL ENVIRONMENTS USING MACHINE LEARNING

BRANDI JESS

A thesis submitted in partial fulfillment of the University's requirements for the Degree of Doctor of Philosophy

January 2023

Coventry University

ABSTRACT

MOTIVE: Heating and cooling systems are large contributors to energy consumption. In order to save energy and cost, the controls of these systems must be optimised. This provides an opportunity for a machine learning approach, which has the potential to provide fast thermal predictions from measurements within a thermal environment. Unfortunately, state-of-the-art simulators are either too slow or provide little resolution of the thermal environment. Furthermore, the modelling of thermal systems is often approached differently depending on the environment or sector. This thesis aims to bridge a gap in the thermal modelling literature between sectors, namely the automotive and residential sectors, and introduce a common, fast approach for modelling various thermal aspects of an environment using machine learning algorithms.

METHOD: Two case studies are investigated. The first is a car cabin, with data from 5 different experiments on the same vehicle in a controlled environment (climatic wind tunnel). The second is a house, which provides a larger dataset with real-world observations from 18 different houses, resulting in 18 models. The data used for analysis is split into vectors of states and controls, which are measurable variables and do not include knowledge of the structure of the space, for example, insulation levels and geometry. A range of machine learning approaches are applied and compared for the two case studies, making use of both hyperparameter search and cross validation methods.

RESULTS: The top performing models for both case studies were found to be based on linear regression. The resulting linear regression model for the car cabin is fast (0.007 milliseconds per predicted second), and yields good accuracy (NRMSE 0.3%) for multi-step ahead predictions, which exceeds the performance of the traditional physics-based model. For the house, a regularised regression (lasso) model provides a good accuracy across the 18 house models that were built (average NRMSE 6.8%), again providing a fast result (average 0.0014 milliseconds per predicted second). Furthermore, the models are able to differentially predict the thermal environment in various locations (for example, footwell vs. head for the car and kitchen vs. bedroom for the house).

CONCLUSION: The resulting fast and accurate predictions based on machine learning can be utilized to optimise thermal systems and its controllers. Implementing a fast, accurate thermal model such as the ones proposed in this thesis can accelerate the adoption of techniques, such as deep reinforcement learning, for climate control in various settings.

ACKNOWLEDGMENTS

First and foremost, I would like the thank my Director of Studies, Prof. James Brusey for helping me develop as a researcher and for providing continuous support throughout this journey. In addition, I would like to thank my second and third supervisors, Prof. Elena Gaura and Dr. Alison Halford, for their insightful comments and words of encouragement.

A special thank you to my other co-authors, Dr. Kojo Sarfo Gyamfi, Alberto Maria Merlo, and Matteo Maria Rostagno, for making publishing a positive experience. I would also like to thank Coventry University for providing me with the financial support needed to conduct this research.

A sincere thank you to all of my peers at the Centre for Computational Science and Mathematical Modelling, past and present, for providing a positive working environment, with challenging discussions on work and much needed breaks for laughter.

I would like to thank my parents for their love and support, and for encouraging me to always follow my dreams. To my family and friends, thank you for acting as beacons of light and providing much needed distractions. Last but not least, I would like to thank my partner for his thoughtful advice and unwavering support throughout this journey.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AC air conditioning

AI artificial intelligence

ANN artificial neural network

AT air temperature

AV	air velocity
CFD	computational fluid dynamics
CWT	climatic wind tunnel
EV	electric vehicle
HVAC	heating, ventilation, and air conditioning system
KNN	K-nearest neighbours
LR	linear regression
LSTM	long-short term memory
MAE	mean absolute error
ML	machine learning
MLP	multilayer perceptron
MPC	model-based predictive control
MRT	mean radiant temperature
MSE	mean squared error
NN	neural network
NRMSE	normalized root mean squared error
PID	proportional, integral, derivative
R&D	research and development
ReLU	rectified linear unit
RF	random forest
RH	relative humidity
RL	reinforcement learning
RMSE	root mean squared error
RNN	recurrent neural network
SGD	stochastic gradient descent
ST	surface temperature

1 INTRODUCTION

THESIS IN A SENTENCE The implementation of a data-driven machine learning model for predicting the thermal aspects of an environment, such as a car cabin and house, which does not require detailed information about the physical space.

1.1 MOTIVATION

In 2020, the UK produced an estimated 405.5 million tonnes carbon dioxide equivalent, contributing to around 1 % of the global total [BE&IS22]. The 2015 Paris Agreement aims to limit global warming to well below 2, preferably to 1.5 °C, compared to pre-industrial levels [CC]. In line with this goal, the UK has set a target to reduce emissions by at least 80 % by 2050 in the Climate Change Act of 2008 [Gov08]. It is estimated that the transport, energy supply, business, and residential sectors make up roughly 80 % of the UK's total net greenhouse gas emissions. In order to reduce the energy use in these sectors, it is important to first understand how energy is used.

Heating and cooling is a large contributor of the UK's energy use and in order to reduce this, the ability to accurately model the system is essential. Heating, ventilation, and air conditioning systems (HVACs) rely on a fast and accurate simulator for thermal data within an environment to ensure correct decisions are made with regards to heating and cooling, as well as to ensure that the environment is appropriately constructed. Therefore, the benefit of these simulations is twofold. First, the simulations will help to reduce the need for high emission, real-time studies and enhance our understanding of how a thermal environment is heated or cooled. Second, the simulations can be utilised to form a better control system and, therefore, a reduction in energy waste.

Although a few studies in thermal environments have examined the use of machine learning methods for predictions of thermal aspects, such as temperature and relative humidity, there has been little attention on how a simulator can be used in practice by making use of existing data and, therefore, saving cost and energy during the research and development (R&D) process. The focus of many of these studies is on the thermal comfort of occupants or on optimizing the control system itself. However, the performance of a simulator is instrumental in building an optimal control system. As such, this thesis provides additional insight into how machine learning methods can be implemented to simulate thermal variables within various envi-

ronments. To provide this insight, this work will examine two case studies: a car cabin and house. This thesis analyses the applicability of machine learning methods in thermal predictions in order to improve the current state-of-the-art methods and use the vast amounts of data available for better learning and thermal predictions. This chapter will detail the importance of modelling thermal environments using machine learning methods and how this investigation will contribute towards the goals of the UK, and the world at large, to reduce greenhouse gas emissions.

1.1.1 *Car cabin*

Transport is the largest emitting sector accounting for approximately 27 % of the UK's total emissions in 2019, with 61 % of that coming from cars or taxis [Tra21]. The transport emissions saw a drop of 19.2 % in 2020, largely due to nationwide lockdowns during the COVID-19 pandemic, bringing the sectors total emissions to 24 % [BE&IS22]. Despite the decrease in emissions from 2019 to 2020, it is clear that vehicle emissions remains an issue that needs to be addressed. To accomplish these goals, the UK aims to ban the sale of new petrol and diesel cars by 2030 [Cam20]. Therefore, it is an important time for electric vehicles (EVs) in the market. In addition to being two- to four-times more efficient than petrol and diesel vehicles, EVs can help to reduce reliance on fuel and greenhouse gas emissions [IEA21]. In over a year, just one electric car on the roads can save an average 1.5 million grams of CO_2. Although the movement towards EVs has begun, it is slow. In 2020, EVs accounted for just 4.6 % of global car sales and about 1 % of global car stock [IEA21]. One barrier to adoption is range anxiety, or the fear of running out of charge before arriving at the destination. The HVAC is the largest auxiliary load and has a significant impact on range, especially during very hot or cold weather [FR00]. However, the climate control system remains essential for maintaining reasonable comfort and defogging the windshield. Thus, minimising the energy cost of delivering climate comfort and windshield visibility can not only save energy, but also contribute to the uptake of EVs.

The automotive industry is facing many new challenges, such as increasing fuel economy of vehicles and autonomous driving. Vehicles' sub-systems are becoming increasingly complex with rapid changes in expectations and technology. With these new demands comes the need for an abundance of testing during the R&D stages, which often means high costs and potential delays in production. The high costs are due to repetitive testing and experimental data collection from concept to production, to the validation of improvements and the assessment of performance. The world's top 20 car manufacturers spent $71.7 billion on R&D in 2019–20; Tesla alone spent $1.1 billion [Aus21]. There is pressure on the automotive industry to develop

sustainable, low-carbon vehicles at an affordable price. In order to achieve this, industry could reduce the reliance on expensive, repetitive testing. Certain variables of thermal systems, such as cabin humidity and HVAC fresh air purge are difficult to capture using existing physics based transient simulations [MN20]. This challenge, on the other hand, can be seen as an opportunity for an accurate and robust thermal model using artificial intelligence (AI) or machine learning (ML) to learn from existing data, both saving production time and cutting development costs.

The first step towards optimising a vehicle's energy use is to accurately model the system. The optimisation of a car cabin's thermal system requires computationally fast simulation for several key reasons. One being a rigorous assessment requires diverse simulated environmental conditions and the final optimised solution must work in the full range of possible situations. Another key reason is the duration distribution for simulations needs to align with the duration of typical car journeys (approximately 22 min, according to the Travel Survey (2020) [Tra20]). Optimisation approaches for the control logic, such as Reinforcement Learning, must experience each possible environment for a typical journey many times to converge on a solution. Furthermore, if the cabin configuration were to be optimised (for example, changing the vent location or using a different type of heating unit), the control logic may need to be re-optimised for each new configuration. In summary, the viability of such optimisation crucially depends on the performance of the simulator, as well as the optimisation algorithm chosen.

1.1.2 *Residential*

The ability to accurately model the thermal aspects of a house room-by-room can contribute towards occupants comfort and health, as well as lead to energy and cost savings. There are two main sectors that contribute to the emissions of houses: energy and residential. The energy sector, which includes the production of electricity and other energy, is responsible for 21 % of UK greenhouse gas emission. The residential sector is responsible for another 16 %, with emissions from the use of natural gas for heating and cooking being the greatest contributor [BE&IS22].

Residential heating and cooling consumes more than 50 % of residential energy consumption and, on top of this, much of that energy is wasted. In present times, where energy costs are on the rise, this is problematic for many. By understanding how the energy is used and wasted in houses, ways to save energy can be discovered. In particular, understanding the heat transfer in building can identify opportunities for insulation, sealing leaks, and optimising residential heating and cooling systems. Through the accurate forecast of thermal

variables, such as temperature, buildings and homes could see an increase energy savings.

Similar to the automotive industry, the regulations on homes have also highlighted the importance of reducing emissions in line with the Future Homes Standard, which will come into effect in the UK in 2025. New homes must produce 70 to 80 % less carbon emissions. Additionally, new offices and shops will need to cut emissions by 27 %.

Thermal predictions are important not only for energy savings, but also aiding in the design of control systems. The ability to predict temperatures in a house on a room-by-room basis can also ensure that only rooms that are occupied are heated, and further, that the heated rooms are able to maintain a comfortable temperature.

1.1.3 *Aims*

The intent of this thesis is to build a simulator for a thermal environment using machine learning algorithms. The ML-based simulator will be beneficial for building optimal control systems, such as using reinforcement learning (RL) or model-based predictive control (MPC), which can help to ensure thermal comfort is maintained and energy is saved. The ML methods are used to predict various thermal aspects of different locations within an environment without knowledge of the structure of the space, for example, insulation levels and geometry. Rather, the ML models are data-driven, making use of existing data, such as state and control variables, to learn from experience. In order to test if this model is generalisable, two case studies, a car cabin and house, are investigated using the same modelling approach. The resulting models can be used to optimise HVAC systems, making use of existing data to model the thermal aspects of an environment. Although numerous studies have identified that machine learning could be used for thermal predictions, little analytical attention has been paid to the comparison across differing thermal environments. This issue is addressed by directly comparing the machine learning method for a car cabin and a house, providing insight on features that are most influential to the thermal predictions. Furthermore, this thesis aims to build a simulator based on ML which can remain stable for longer term predictions.

1.2 RESEARCH QUESTIONS

In response to the issues above, the key research questions (RQs) that are investigated throughout this thesis are as follows:

1. Can a data-driven ML model, with little details about the physical environment itself, be used to model thermal aspects of an environment?

   a) Can the top performing ML model for a car cabin provide faster and more accurate predictions than the industry standard? If so, to what extent?
   b) Can the top performing ML model for a house provide a fast and accurate prediction of thermal variables?

2. Which ML model provides the most accurate next step predictions for the thermal aspects of various locations in different thermal environments?
   a) Which ML methods are the most accurate for a car cabin?
   b) Which ML methods are the most accurate for a house?
   c) Are the ML techniques that perform best for the two case studies the same? If not, why are they different?

3. Can the top performing ML model a provide stable, long-term prediction?
   a) Is the model stable for a car cabin?
   b) Is the model stable for a house?

4. What thermal and environmental features are most important in achieving a high accuracy ML thermal model?
   a) What features are most important for a car cabin?
   b) What features are most important for a house?
   c) What are the similarities and differences in the features for the two case studies and why?

The following section will detail each of the research questions aims and contributions.

1.3 CONTRIBUTIONS TO KNOWLEDGE

This thesis investigates the use of machine learning methods for predicting thermal variables within an environment, namely for a car cabin and house. The research shows that data-driven, machine learning models are able to quickly and accurately model thermal aspects, and through the two case studies, shows the potential for widespread use of these models in thermal environments.

This thesis provides contributions to knowledge in the following areas:

1. The first contribution of this thesis is to bridge a gap across different sectors. Often, thermal environments are considered and modelled using different approaches depending on the environment in-hand. In order to answer RQ1, this research adapts a similar approach to two different settings, a car cabin and house, showing that a unified approach could be adopted for thermal simulations.

a) For the car cabin, the chosen ML model is compared to the industry standard physics-based approach. This comparison provides a contribution to the field as it explores the suitability of ML techniques as an alternative to the industry standard, a physics-based simulator.

b) For the house, a simplified ML model is proposed. That is, the model does not include details such as room geometry, insulation levels, occupancy levels, etc. The approach taken attempts to model this environment by simply learning from observable data, using a vector of states (for example, indoor temperature and relative humidity) and controls (for example, weather and gas consumption) to predict the next state. An examination of the speed and accuracy is done.

2. In response to RQ2, the results of various machine learning methods are compared and discussed. Multiple ML methods are tried, ranging from a simple linear regression to a more complex multilayer perceptron (MLP), in order to fully examine and conclude what model should be selected for the data at hand.

   a) This thesis not only provides an examination of two thermal environments, a car cabin and a house, adapting the same approach, but also has an analytical focus on making multiple thermal predictions for various areas within a space, which provides another contribution to the field as a more complete picture of how the thermal environment varies across locations (for example, head vs foot in a car cabin or the living room vs kitchen in a house) is obtained.

   b) The discovered top model for the car cabin and house will be compared, drawing upon similarities and differences and discussing these results. This contribution again helps bridge a gap across sectors by conducting a deep analysis of how and why the chosen final models may differ for various thermal environments as well as an examination of the accuracy of these models.

3. After finding the top performing model for next step predictions, RQ3 considers how stable the performance is for longer term predictions. In particular, the car cabin should be able to simulate the time of an average car journey, which is approximately 22 minutes, and the house should be able to simulate for a longer period, for example for a full day or a week.

4. A further contribution of this thesis is an analysis of important features for modelling thermal aspects of an environment, in response to RQ4. This also connects back to the first contribution as a gap is being bridged across sectors in terms of what features

are most important to the models. The important features of each individual model are described and then differences are discussed.

a) The most important features for a car cabin and for a house will be introduced and discussed.

b) The comparison between important variables is dependent on the data available from each case study. Therefore, the variables may not always have a directly comparable alternative, however, the general variables are discussed and considered for each model and comparisons are made wherever possible.

This contribution could also lead to quicker, cheaper R&D in manufacturing, helping to reduce the carbon footprint and speed up delivery of new and improved HVAC systems for both cars and houses. Furthermore, if the techniques used here are applicable to both the thermal aspects of a car cabin and a house, then the scope of use for machine learning in thermal modelling could be widespread.

This study is important because the ML techniques are data-driven, only needing previous state values and control values, not requiring complex mathematical derivations or detailed knowledge of the physical space, to learn and make fast and accurate thermal predictions. The application to two different thermal environments demonstrates that the modelling techniques could potentially be generalisable.

1.4 PUBLICATIONS

The following publication resulted from the work on which this thesis is based:

- Jess B, Brusey J, Rostagno MM, Merlo AM, Gaura E, Gyamfi KS. "Fast, detailed, accurate simulation of a thermal car-cabin using machine-learning." In: *Frontiers in Mechanical Engineering* 8. (2022). ISSN: 2297-3079. DOI: 10.3389/fmech.2022.753169.

This journal article is based off the work in Chapter 4 and an open-sourced copy can be found through the associated DOI link [Jes+22].

1.5 THESIS STRUCTURE

Figure 1.1 shows the general layout of this thesis, in particular how the ML pipeline is constructed and can be evaluated using a control system. The remainder of this thesis is structured as follows:

- Chapter 2 will review the current literature and highlight gaps, building a stage for the research done in this thesis. The topics include thermal modelling, including the modelling approaches

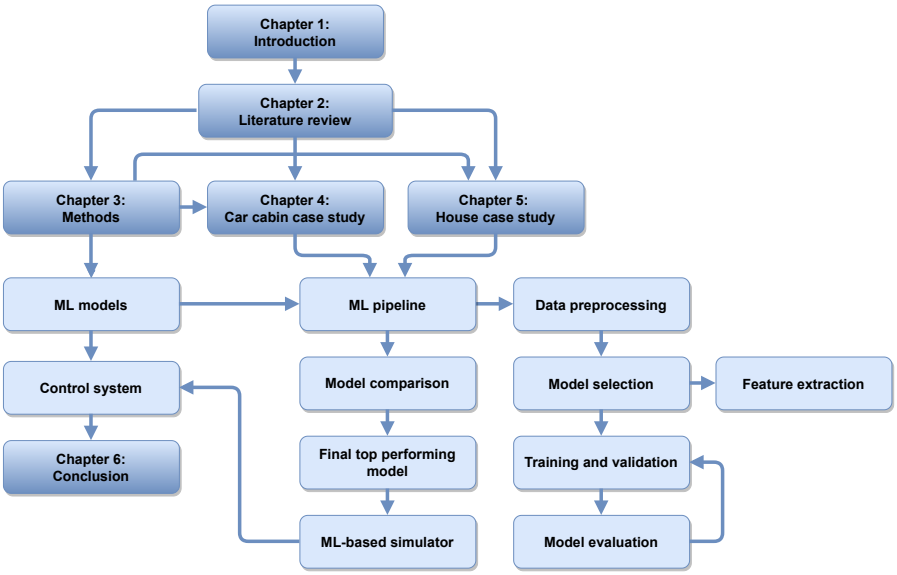

Figure 1.1: The structure of this thesis.

and applications, as well as an introduction to the existing methods from the literature. Methods cover a range of machine learning approaches, time series analysis, multi-output modelling, and cross validation methods.

- Chapter 3 will detail the methods used for the analysis in this thesis, including an outline of how analysis is done and what new methods have been contributed. This chapter aims to build a recipe for how analysis was conducted across both case studies.

- Chapter 4 includes the application of machine learning methods to car cabin data from climatic wind tunnel (CWT) trials. In this chapter, the thermal aspects of different areas within a car cabin will be modelled using data from the same car in five different experiments. The resulting top ML model will then be utilised in a simulator to make longer term predictions. The model will also be compared to a physics-based model, showing the potential for ML models in practice.

- Chapter 5 examines the application of the models from Chapter 4 by applying a similar approach to data from multiple houses. For this application, a model of the thermal aspects of various rooms will be built for each of the 18 houses used. An evaluation of the next-step, as well as a longer run predictions will be presented. The resulting model will be compared with the car cabin case study, including a discussion on the similarities and differences in the models themselves, as well as the most important features.

- Finally, Chapter 6 will summarise the findings, detail limitations, and propose future avenues of research that have arisen from this thesis.

1.6 ACKNOWLEDGEMENT OF CONTRIBUTED WORK

This thesis is reliant on research conducted by other researchers, namely:

- The Design OptiMisation for efficient electric vehicles based on a USer-centric approach, or DOMUS, project provided the basis for this research [BR20]. In particular, Centro Ricerche Fiat S.C.p.A., or CRF, provided the climatic wind tunnel experimental data used for the car cabin case study, as well as the results of a physics-based modelling approach, which allowed for the comparison to the model adopted in this thesis.
- The data used for the house case study is a publicly available dataset, the REFIT Smart Home dataset [Fir+17].

2

LITERATURE REVIEW

This chapter covers relevant literature on the modelling of thermal aspects of environments. Specific models which have been used for thermal environments will be analysed, including the details of the models (that is, what variables are used and what is being predicted). Furthermore, this chapter will also introduce and explain time series and multi-output regression methods that have been used throughout literature. This chapter will show the relationship of this work to current literature and highlight gaps, problems, and shortcomings.

The literature review aims to provide an overview of current theories and debates within the topic area. This will include explaining what thermal models are, where they are used, what they include, and how they are implemented. More specifically:

- An explanation of thermal modelling, including literature on thermal modelling;
- An overview of the variables typically used to build thermal models and why these are important;
- An analysis of where thermal modelling has been used in practice across the transport and residential sectors;
- An introduction of existing physics and machine learning-based methods for thermal modelling, as well as the difference in modelling choices for various sectors across relevant literature;
- Introduction to multi-output learning and why this is useful in this domain;
- Discussion on time series analysis and how it is used for thermal modelling;
- An overview of relevant cross validation methods with examples of their use.

This chapter will provide a strong argument as to why this research is needed and how it fills a gap in the current literature. One of the major contributions of this thesis is to bridge a gap across thermal modelling in different sectors and, by analysing the different uses per sectors, builds a strong foundation for the remaining research done in this thesis. This chapter also contributes to the idea of using machine learning to provide fast, accurate thermal models by explaining in detail why machine learning (ML) is appropriate in this application, how these models have been applied to date, and how this could be applied in practice to make advances in technologies.

2.1 WHAT ARE THERMAL MODELS?

Thermal models are models relating to the dynamics of heating and cooling, for example, heating, ventilation, and air conditioning systems (HVACs). The ability to model thermal environments is important as it can inform the future choice of products and enable adjustments to achieve peak performance, leading to improved energy efficiency. The thermal environment involves anything that affects the thermal exchange of the heat transfer between different systems. There are various ways to model a thermal system. Thermal capacity is based on a variety of factors that affect both internal and external environments and models have been developed to predict how a thermal environment will change based on these variables.

2.1.1 *Thermal variables*

There are several aspects of a thermal environment that can affect heat transfer, including:

- air temperature (AT),
- air velocity (AV): the rate of motion of the air in a given direction,
- mean radiant temperature (MRT): the average temperature of surfaces surrounding a point, accounting for the ability to emit and absorb heat (emissivity), and
- relative humidity (RH): the actual water vapour in the air relative to the maximum water vapour the air can hold at the same temperature.

2.1.2 *Thermal comfort models*

Many thermal comfort models, as well as some of the environment variables mentioned previously, rely on personal variables (that is, measurable, on-body parameters). Some of the most commonly used standards for thermal comfort are the ISO 14505 [Isoa] based on work by Nilsson [Nil04], as well as ISO 7730 [Isob] and ASHRAE 55 [Ash20], which are both based on work by Fanger et al. [Fan+70]. ISO 14505 specifies a method for assessing thermal comfort in vehicles. ISO 7730 presents a method for calculating thermal sensation and discomfort for those exposed to moderate thermal environments. ASHRAE 55 specifies the range of indoor environmental conditions necessary to achieve acceptable comfort for occupants, specifically for buildings. The intended goal of these standards is to improve the design, operation, and commissioning of thermal environments, with a particular interest in car cabins and buildings.

Although the focus of this thesis is on a machine learning simulator, it is important to address how this could relate to thermal comfort models. The ML simulator can provide an accurate forecast of thermal conditions, which provides crucial insights that can be used to optimize the car's HVAC system for comfort. Through the simulation of various scenarios, including changes in external conditions and HVAC configurations, the simulator can help identify optimal settings that prioritize comfort. Moreover, it can aid in the development and testing of HVAC control algorithms, allowing for precise fine-tuning of strategies to achieve the best possible comfort levels while ensuring energy efficiency. While this simulator acts as a virtual testing environment for exploring control strategies, it cannot be directly used for evaluating changes to the physical environment, such as changes to the seat materials, window configurations, or insulation levels. A way that this might be done, however, is to use another modelling tool, such as computational fluid dynamics (CFD), to derive sufficient examples to train a new simulator. Alternatively, the modification can be done to a prototype vehicle and then further experience data can be obtained to train the new simulator.

2.1.3 *Control systems*

HVAC systems work by continuously monitoring and controlling the temperature in an environment. An HVAC control system regulates the operation of the HVAC. This is typically achieved by comparing the actual state (for example, temperature) with the target state (for example, set point temperature) and taking action as necessary (for example, starting the blower).

A control system is a set of devices that directs or regulates either itself or another system in order to provide a desired response. Two key examples of control systems that are related to this work are the thermostat controls in a house and in a car, which both control an HVAC system. Control systems can be manual, such as an HVAC that will continue to blow air at the same pace until it is adjusted or switched off, or automatic, such as setting a thermostat temperature to 22 °C where the system then adjusts the speed and distribution in order to maintain the set temperature. In an automatic temperature control system, a suitable sensor (for example, a thermocouple) is used to measure the temperature. Thermal control systems usually contain a heat source, a heat spreader, a temperature sensor, and a controller. Control systems can also be open- or closed-loop. An open-loop system means that the control action is independent of the output, whereas a closed-loop control system is dependent and makes adjustments based on the output.

There are various types of control strategies that can be utilised, from a simple on/off to proportional, integral, derivative (PID) and model-

based predictive control (MPC) controllers. PID controllers are the simplest automatic controller that allows for the use of past, present, and future error, and, therefore, are common controllers used in HVAC systems [AM08, Chapter 10]. PID controllers allow for the adjustment of a desired temperature through the use of a set point temperature. A PID controller adjusts the control output based on the error, or, in this case, the difference between the current temperature and the set point temperature. The proportional component only relies on the error between the set point and current temperatures. The integral component integrates, or sums, the error over time, while the derivative component is proportional to the derivative of the error, or the rate of change of the error. Before a PID controller can be applied, the control parameters (proportional, integral, and derivative gain/reset) must be tuned. It is also possible to use a P or PI controller, which utilises the proportional or proportional and integral parameters respectively. A disadvantage of the PID controller is the reaction time for adjusting the current temperature, in particular when the signal exceeds its target (overshoot) as this can only be corrected slowly and cannot be forced down. Lomas et al. [Lom+18] studied the energy efficiency of these controls and found that the savings was often low quality or non-existent.

The MPC controller is a control strategy based on numerical optimisation that uses a model of a system to predict future system behaviour [RMD20]. MPC provides a cost effective approach and can provide energy savings over the PID controllers [Ora+18]. Using prediction, the MPC determines an optimal output by solving a constrained optimization problem. A benefit of an MPC controller is that it is one of the few control methods that has the ability to directly consider constraints and uncertainties [RH11].

Common computer packages, such as LabVIEW [BMN06], SIMULINK [Doc20], and Modelica [Modoo], are often used to construct and test these complex physical systems.

Another approach to the HVAC control problem is reinforcement learning (RL) [Ram+20]. RL is a sub-discipline of ML that tries to learns an optimal mapping from situation to action given some reward structure [SB18]. In the setting of HVAC models, an RL agent can directly interact with the environment to learn the dynamics from raw experience. The RL agent can then make decisions in order to achieve a pre-defined goal, such as keeping a room in a house comfortably heated, by using a defined reward function.

The control systems mentioned in this section all crucially rely on an accurate model of the temperature to provide accurate control functionality.

2.2 THERMAL MODEL APPLICATIONS

Thermal models have been used widely across literature for topics including:

- weather [SW05];
- components in electrical devices [Bah+22];
- personal space heaters [Kat+18];
- high performance computers [Zha+18];
- water heating systems [KPD99];
- smartphone thermal management [Jai+21];
- heat shields on aerospace craft [RPM15];
- manufacturing of building materials [Mir+18];
- river temperature [RHNv15].

Two of the most frequently explored thermal modelling applications include the HVAC systems in a car cabin and house, which are explored in more detail in this section.

2.2.1 *Car cabin*

The primary aim of thermal models for car HVAC systems is to ensure that components are sufficiently powerful to cope with the expected range of conditions and to ensure that the cabin is cooled or warmed sufficiently quickly. For this reason, relatively simple 0D or 1D thermal models are commonplace in industry [DM19; Mar+14].

The energy cost of the climate system has become more important in recent work [Laj17; KB14a], along with the realisation that air temperature is not the only influence on occupants' thermal comfort. As a result, other measurements have been considered for thermal modelling, such as mean radiant temperature [KK20] and air velocity [Kam+13].

More recently, work using artificial neural networks in automotive applications has been done, mainly focusing on the automotive air conditioning system. Apart from modelling the temperature and other thermal variables, work has also been done to predict the performance of the system, as well as cooling capacity [HE06; DDM19; Kam+13]. Other work, such as that by Ng et al. [Ng+14b], made use of multilayer perceptron (MLP) and radial basis network with experimental data to predict the average air temperature in the car cabin. In order to use the ISO 14505 model to estimate thermal comfort, 3 modalities are required (air temperature, mean radiant temperature, and air velocity)

for at least 3 locations (head, torso, and foot) for each passenger (a total of $3 \times 3 \times 4$ variables). Furthermore, for safety reasons the windscreen may also need to be checked for fogging, which will require the relative humidity inside the cabin and windshield temperature. Predicting this many thermal variables is a big step up from simply predicting the average air temperature.

2.2.2 *House*

In thermal modelling, there is an abundance of research conducted for commercial or residential buildings [Afr+17; KXZ14]. Bastida et al. [Bas+19] adapted a dynamic physics-based thermal model of a house using Simulink and a PI controller, requiring detailed information, including the thermal storage capacity (calculated through the house geometry and properties of the materials). This model requires detailed information about the environment that may not always be readily available, for example, it may be unknown what type of or how much insulation is inside a cavity wall.

Aguilera, Andersen, and Toftum [AAT19] used various measurements, such as weather, occupancy levels, whether windows were open, floor area, and construction year, to predict the indoor air temperature of a house. The results showed that the heating set point on the thermostatic radiator valve was the most important feature in predictions, while building related data had less effect. However, this model treated the prediction of air temperature as a classification problem (a range of 2 °C per predicted class), which would not provide the precision needed for some applications.

2.3 PHYSICS-BASED THERMAL MODELS

Traditional approaches to thermal modelling are based on physical or mathematical models. Thermal systems, when formulated as differential equations, are mainly linear with respect to their inputs. Given a simple thermal system that involves a body (for example, a container of water) with temperature $y(t)$, an external environment that maintains a uniform temperature y_0, and some insulating barrier (for example, the outer wall of the container) with coefficient k, the rate of change of temperature of the body $\frac{dy}{dt}$ is proportional to the difference between the inside and outside temperature,

$$\frac{dy}{dt} = -k(y - y_0). \tag{2.1}$$

This is known as Newton's model and forms the basis for lumped thermal models where the components parts, or lumps, are considered to have a single uniform temperature. The coefficient k could potentially be expanded to consider the surface area and the unit thermal resistivity of the dividing layer.

Throughout the literature, deterministic models are presented, (that is, 0D, 1D, or 3D models). A 3D CFD model, solves fundamental flow equations and can allow for simulations under various conditions, including extreme conditions. The 3D models rely on heat transfer processes, such as convection and conduction, as well as the thermal storage capacity, calculated through geometry and materials information. However, 3D CFD approaches are complex and computationally expensive.

The 0D models, also referred to as a lumped parameter model, are given by ordinary differential equations and are only dependent on time, thus reducing the complexity and computational demands when compared to a 3D model. Lumped parameter thermal models are based on heat transfer and electric current flow theory and provide a simplified state space, or model of the physical system. The 1D models, in addition to a time dependency, also include a spatial dimension and are solved through differential-algebraic equations. Throughout the literature, 1D models are commonly used to represent a thermal environment. This approach can determine the relative humidity and temperature within the environment without the need for complex CFD or experimental models. Despite the simplifications, the 1D model is still complex due to the complexity of the environment and the other components involved. For 0D and 1D models, physical variables of properties are seen as grouped for elements of the model. While these models offer a simpler and less computationally expensive approach, they fail to provide detailed information about the flow and thermal fields of the environment.

Cui et al. [Cui+18] built a physics-based model to predict a difference in temperature between an upstairs and downstairs in a house using detailed information about thermal resistance, thermal capacitances, and heat resources.

Although physics-based models can provide accurate models, they may not be suitable for real-time predictions due to the computational demand.

2.4 HYBRID THERMAL MODELS

In addition to the work on physics-based and ML-based models, work has been done to adapt a hybrid approach to modelling, sometimes referred to as grey-box models. The hybrid models draw upon the knowledge from physics-based approaches while using the data-driven techniques from ML to aide in the process. For example, the coefficients of the physics-based models can be estimated by learning from the data itself, rather than being derived from mathematical equations. Kirchgässner, Wallscheid, and Böcker [KWB21] proposed a hybrid method where the ML model is used to estimate the partial differential equations used in traditional lumped-parameter models. Singh and

Abbassi [SA18] adapted a hybrid approach for the thermal modelling of the cabin in machinery, which was based on CFD and artificial neural network (ANN). Furthermore, Attoue et al. [Att+19] adapted a hybrid approach for the short-term predictions of thermal aspects of a building that produced an error of less than 0.2 °C for temperatures.

While hybrid approaches are able to encompass some benefit from both physics-based and data driven models, the level of physics and mathematical detail required is quite high. When comparing the performance of each of these types of models separately, Shahdi et al. [Sha+21] found that the ML predictions were comparable to, and sometimes outperforming, the traditional physics-based model. This indicates the potential for the simpler, data-driven models in thermal modelling.

2.5 MACHINE LEARNING THERMAL MODELS

This section introduces ML methods that have been utilised throughout thermal modelling literature. ML is a branch of artificial intelligence (AI) that uses algorithms involving gradual refinement to produce a classifier or predictor that aims to work well on data that has not been seen during training. More recently, machine learning-based models have been proposed for thermal modelling. ML models are particularly suitable for problems where processes are not completely understood or where it is infeasible to run the physical models at desired resolutions in space and time. These data-driven models can estimate thermal variables of interest from readily available sensor data. The ML models rely solely on empirical data as they do not incorporate any thermodynamic theory, geometry, or material information. Therefore, the ML thermal models aim to provide a model of the environment that is derived without knowledge of the physical system but, rather, is obtained entirely from measurement data.

Research in thermal modelling often makes comparisons amongst multiple ML methods for the analysis. Shahdi et al. [Sha+21] made use of several ML techniques for predicting subsurface temperatures at various depths, including ridge regression, random forest (RF), XGBoost, and a neural network (NN), which resulted in the RF and XGBoost as the top performing models. Panek and Włodek [PW22] used ML methods, such as linear regression (LR), RF, and NN, to forecast the gas consumption across a city based on metrological factors, such as temperature, wind velocity, and humidity. Warey et al. [War+20] made use of simulated CFD data in linear regression, random forest, and ANN to predict temperature within a car cabin. Various thermal modelling applications take differing approaches in terms of modelling. This section will discuss in further detail the ML models used across the literature for thermal environments and where they have achieved promising results. The ML models range from a simple LR to

regularised techniques, to neighbours- and tree-based approaches, to gradient boosting and MLP.

2.5.1 *Linear regression*

Alongside the more complex models, such as MLP, a LR is considered. In a simple LR model, for the input vector $\mathbf{x}^T = (x_1, x_2, \ldots, x_p)$ with p variables and a real-valued output y, we use the form

$$y = \beta_0 + \sum_{j=1}^{p} x_j \beta_j, \tag{2.2}$$

where β_0 is the intercept parameter and β_j is the jth coefficient. Typically the parameters β are estimated using a set of training data $(x_1, y_1) \ldots (x_N, y_N)$, where each x_i is a vector of feature measurements.

In the least squares approach, the coefficients $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$ are chosen to minimise the residual sum of squares cost function

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{2.3}$$

$$= \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2, \tag{2.4}$$

where y_i is the actual value and $f(x_i)$ is the predicted value. The residual sum of squares can be written as

$$\mathrm{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta), \tag{2.5}$$

where $\mathbf{X}$ is an $N \times (p + 1)$ matrix where each row is an input vector and $\mathbf{y}$ is a vector of N outputs in the training set. Using differentiation and the assumption that $(\mathbf{X}^T\mathbf{X})$ is positive definite, this can be solved for ordinary least squares to obtain the solution $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{y})$.

The method of least squares provides closed form estimates for the regression parameters whereas stochastic gradient descent (SGD) optimises over the log-likelihood iteratively. SGD is commonly used in other techniques, such as NN, and due to the iterative and approximate nature can result in a performance advantage. LR uses a minimal set of coefficients to model the system, which means that while overfitting is unlikely, there is a slight possibility of underfitting, that is, the model could not providing sufficient flexibility. Including interactions and polynomial terms in a LR can lead to greater complexity and, therefore, overfitting. The simplicity of LR tends to make it robust to measurement noise although noise in sensor readings used as the independent variables causes LR to underestimate the gradient, known as regression dilution [DS98, Chapter 24]. Time series noise reduction is used to reduce this effect for statistical inference.

Other work on thermal modelling has seen promising results from LR. Hintea, Brusey, and Gaura [HBG15] found LR to be the top performer in predicting car cabin temperatures in a comparison to other ML models, while Cui et al. [Cui+18] found LR provided accurate air temperature predictions for separate floors in a house.

LR models do not require a large amount of input data and are not computationally expensive. Least squares estimates often have low bias and large variance, which could negatively affect the prediction accuracy of the model. This can sometimes be resolved by shrinking or setting coefficients equal to zero, which increases the bias slightly and reduces the variance.

2.5.2 *Regularisation*

Here the regularisation techniques of ridge, lasso, and elastic net regression will be introduced [HTF01, Chapter 3]. Regularisation techniques make slight adjustments to the learning algorithm to aide in the generalisation of the model and the performance on unseen data. Regularisation is often used as a solution to overfitting, which could be caused by a small training data set relative to the number of predictors.

Ridge regression, or L_2 regularisation, imposes a penalty on the size of the regression coefficients to reduce the complexity of the model. In ridge regression, not only are the residual sum of squares minimised, but the size of parameter estimates are also penalised. The ridge estimates are defined as

$$\hat{\beta}_{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}. \tag{2.6}$$

where $\lambda \geqslant 0$ is the complexity parameter where a larger λ results in more shrinkage. An equivalent way of writing this is

$$\hat{\beta}_{ridge} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \tag{2.7}$$

$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 \leqslant t, \tag{2.8}$$

where there is a one-to-one correspondence between the parameters λ and t. The coefficients can exhibit high variance and be difficult to determine when there are a large number of correlated variable in a linear model. Introducing a size constraint, such as the one above, this can alleviate the problem. In matrix form, this can be rewritten as

$$RSS(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta, \tag{2.9}$$

where the λ parameter is the regularisation penalty, $\mathbf{X}$ is the input matrix, and $\mathbf{y}$ is the output vector. Solving for $\hat{\beta}$ gives $\hat{\beta}_{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}(\mathbf{X}^T\mathbf{y})$ where $\mathbf{I}$ is the identity matrix. The ridge regression solution is a linear function of $\mathbf{y}$ with the penalty chosen as $\beta^T\beta$. Ridge regression will bring parameters near zero, but never actually zero.

Lasso regression, also known as L_1 regularization, adds a penalty for non-zero coefficients similar to ridge. The key difference is that lasso penalizes the sum of the absolute values rather than the sum of squared coefficients, which are penalized in ridge regression. Therefore, lasso provides a built-in feature selection method as it has the ability to shrink coefficients to zero. The lasso estimate is defined by

$$\hat{\beta}_{lasso} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \tag{2.10}$$

$$\text{subject to} \sum_{j=1}^{p} |\beta_j| \leqslant t. \tag{2.11}$$

This is similar to the ridge regression problem, but the L_2 ridge penalty $\sum_1^p \beta_j^2$ is replaced with the L_1 lasso penalty $\sum_1^p |\beta_j|$. The lasso constraint makes the solution non-linear, therefore a closed form solution does not exist.

Zou and Hastie [ZH05] introduced a new penalty suggesting a compromise between the ridge and lasso regression, as well as for computational tractability. This penalty is called the elastic net penalty, which combines the penalties of ridge and lasso to give the following penalty

$$\lambda \sum_{j=1}^{p} \left(\alpha\beta_j^2 + (1-\alpha)|\beta_j| \right). \tag{2.12}$$

Elastic net adopts the selection of variables from lasso and the shrinkage of the coefficients of correlated predictors from ridge and, in addition, provides computational advantages over both methods. The α parameter controls the mix of penalties. If α is 1, this would be a L_1 lasso penalty, 0 would be a L_2 ridge penalty, and any α value between 0 and 1 provides a combination of both. A possible advantage over lasso is that when $p > n$, elastic net allows for more than N non-zero coefficients.

Regularisation methods have been applied to various thermal environments throughout the literature. Al-Obeidat, Spencer, and Alfandi [AOSA20] utilised ridge and lasso regression for the prediction of temperatures in buildings, finding that lasso was the top performer. Karevan, Mehrkanoon, and Suykens [KMS15] found elastic net as a useful feature selection technique for forecasting weather conditions.

2.5.3 *Neighbors-based models*

Nearest-neighbors methods form estimates using observations in the training set that are close in the input space. K-nearest neighbours (KNN) is a non-parametric, supervised learning technique that uses the similarity of features in the training set to predict the values of any new data point. KNN fits $\hat{Y}$ by

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i, \tag{2.13}$$

where $N_k(x)$ is the neighbourhood of x, that is, the k closest points to x_i in the training sample, using some distance, such as a uniform or Euclidean distance. Therefore, the average of the corresponding responses is calculated for k observations that are located where x_i is closest to x in the input space. KNN can be summarised then in three steps:

1. Calculate the distance between the new point and each training point,
2. Choose the k closest points, and
3. Average the k points.

KNN can be used for both classification and regression. In the regression setting, the output is an average of the k nearest neighbors, that is, if $k = 1$ then the output would be equal to the value assigned to the nearest neighbour. KNN retains all training data in memory, unlike other ML methods, which do not rely on training data to make predictions.

Badhiye, Sambhe, and Chatur [BSC13] used a KNN approach for the prediction of weather data, such as temperature and humidity, using airport weather data. The findings indicate that KNN provides accurate results, with slightly better performance in terms of mean squared error (MSE) for temperature prediction compared to humidity.

2.5.4 *Regression trees*

Tree-based methods are simple, but powerful methods that use a set of rectangles to divide up the feature space, then fitting a simple model to each. Regression trees are a variant of decision tree that use a binary recursive partitioning to split the data into partitions or branches. Let the data consist of p input features and a response for each feature, that is, for $1 = 1, 2, \ldots, N$ with $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$. A regression tree automatically decides on the splits and the shape of the resulting

tree. If the space is divided into M regions $R_1, R_2, \ldots, R_M$ and the response of the model is a constant c_m, this results in

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m). \tag{2.14}$$

To minimize the sum of squares $\sum (y_i - f(x_i))^2$, the best $\hat{c}_m$ is the average of y_i in the region R_m

$$\hat{c}_m = \text{avg}(y_i | X - i \in R_m). \tag{2.15}$$

Finding the best partition in this way is not feasible, so a greedy algorithm is often used. Using all of the available data, if we select a splitting variable j and split point s, and the pair of half planes are defined as

$$R_1(j, s) = \{X | X_j \leqslant s\} \text{ and } R_2(j, s) = \{X | X_j > s\}, \tag{2.16}$$

then we seek a j and s that solve

$$\min_{j,\ s} \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right]. \tag{2.17}$$

For any choice of j and s, the minimizations are solved by

$$\hat{c}_1 = \text{avg}(y_i | x_i \in R_1(j, s)) \text{ and } \hat{c}_2 = \text{avg}(y_i | x_i \in R_2(j, s)). \tag{2.18}$$

For each choice of j, the determination of s can be done quickly so this approach is viable. Having found the best split, the data can then be divided into two regions and the process can be repeated in each region. Regression trees benefit from easily interpretable results and do not require the predictors relationship to the response to be specified, for example, as a linear model does [KJ13, Chapter 8]. These models can also handle missing data effectively and are able to conduct feature selection (that is, if a predictor is not used in a split). However, a simple regression tree can be unstable and provide subpar predictive performance due to the definition of rectangular regions for the outcomes. To overcome these pitfalls, ensemble methods have been developed, which can offer better predictive performance.

A random forest is an ensemble approach which combines multiple decision trees using the technique of bagging. Bagging is a bootstrap algorithm used to reduce the variance of a model, particularly suitable for high variance, low bias methods, such as trees. In bagging, several regression trees are fit to bootstrap samples of the training data in parallel and the average result is taken across the trees. In the case of random forest, a further step is taken to randomly select subsets of features for each sample. The following steps are repeated for the number of chosen models to build. For each bootstrap sample $\mathbf{Z}^*$, a

random forest tree is built and each node of the tree is minimized by selecting a subset of predictors, picking the best split point, and splitting into two daughter nodes. The output is an ensemble of trees that predicts a new point x using

$$\hat{f}_{rf}^{B}(x) - \frac{1}{B}\sum_{b=1}^{B} T_b(x). \tag{2.19}$$

A downfall of random forest is that it cannot extrapolate, or predict values outside of the observed range, whereas a linear model can. It is also difficult to understand the relationship between the predictors and response due to the ensemble nature of random forests.

Adding another step to the random forest model, extremely randomized trees, or extra trees, trains each tree on the whole training set, rather than a bootstrap sample, and the splitting for the tree learner is randomized. That is, rather than computing a local optimal cut point, extra trees uses a random cut point instead. This adds another step to the randomness, in addition to the random subsets of data and random features for splits, thereby reducing the correlation between trees in the forest. This approach often reduces the variance of the model at a cost of an increase in the bias.

Ramadan et al. [Ram+21] examined several ML models for indoor temperature forecasting, finding that the extra trees regressor was one of the top performers, having also outperformed the hybrid, gray box approach. Alawadi et al. [Ala+20] tested 36 different ML models for the forecast of 3 hours of indoor temperature for a house, finding that the extra trees models performed best in terms of R-squared and root mean squared error (RMSE), and demonstrated robustness against outliers and noise in the data. Other models that had good performance included random forest, and gradient boosted machines with regression trees, which will be introduced in the following section.

2.5.4.1 *Boosting trees*

Boosting is an ensemble algorithm that combines the outputs from multiple weak learners to form a single stronger one. Originally, boosting was designed for classification problems, but these methods can also be used for regression. Gradient tree boosting is an approach that creates new models that predict the errors of prior models and combines these for a final prediction [Fri01; Mas+99]. As suggested in the name, gradient boosting uses a gradient descent algorithm to minimise the loss while adding new models. Boosting trees differ from a random forest in how the trees are grown and combined.

XGBoost, short for extreme gradient boosting, is known for its efficiency in terms of speed and memory resources, as well as its model performance. XGBoost implements the gradient boosting decision tree algorithm while providing parallel tree boosting to provide fast and accurate solutions [CG16].

In the application of thermal modelling, Liao et al. [Lia+20] found gradient boosting regression had the best accuracy for multi-step forecasting of inflow into a reservoir, when compared with ANN, support vector regression, and linear regression. Kaligambe, Fujita, and Keisuke [KFK22] utilised an XGBoost model to predict temperature, relative humidity, and CO_2 levels in a smart home, showing the potential for making predictions within residences. However, this research was conducted on just one single home with no comparison to other ML model results.

2.5.5 *Neural networks*

In recent years, recurrent neural networks (RNNs) have become known as a go-to method for time series data. An RNN processes time series step-by-step, while preserving an internal state from step-to-step. Sri Rahayu et al. [Sri+20] made use of RNN by implementing a long-short term memory (LSTM) for predicting daily temperature. Despite the strengths of this approach, the models will only accept inputs that are sequential, or time dependent, so if the aim is to also include control inputs, then these are not, by nature, handled by this approach. An alternative would be an MLP, which is a class of feed forward ANN where the connections between the nodes do not form a cycle, meaning there are no loops or feedback connections. ANN is a common tool for energy forecasting due to the ability to approximate non-linear processes with high accuracy. ANN are often displayed as arcs and nodes where each arc are associated with a weight $w_{i,j}$. For example, for 7 inputs and 4 outputs, the weights can be expressed as a matrix W with dimensions 7×4. For linear activation (or pass through) on the output nodes, the general form is

$$y = Wx + b. \tag{2.20}$$

Other activation functions and the addition of hidden layers with varying numbers of nodes per layer produce a non-linear function of arbitrary complexity and expressiveness.

The number of neurons in the input layer corresponds to the number of model inputs or predictors, while the neurons in the output layer correspond to the number of outputs of target variables. The hidden layers apply weights to the inputs and apply an activation function to provide the output. Models which have more than one hidden layer are considered Deep Neural Networks. Within each hidden layer, the number of neurons or nodes must also be determined. The number of hidden layers, and nodes within each of those layers, is often found using a search strategy, such as random or grid search. Panchal and Panchal [PP14] suggest the following criteria when selecting the number of nodes in hidden layers, the number should be:

- Between the size of the input and output layers.

- Two-thirds of the input layer size plus the output layer size.
- Less than twice the input layer size.

Depending on the application, the threshold mechanism, or activation function, can be varied. Commonly used activation functions include linear, rectified linear unit (ReLU), sigmoid, and hyperbolic tangent (tanh). ReLU is a relatively modern activation function that has had success as a hidden layer activation function for problems such as handwriting recognition. Sigmoid and tanh are commonly used, smooth, differentiable functions, whereas linear activation merely passes through the input without modification. Table 2.1 shows the formulas for these common activation functions.

Table 2.1: Definition of some common NN activation functions

Activation function	Formula
linear	$f(x) = x$
ReLU	$f(x) = \max(0, x)$
sigmoid	$f(x) = \frac{1}{1+\exp -x}$
tanh	$f(x) = \tanh(x)$

Research has been conducted on the performance of MLPs in thermal environments. Rabi, Hadzima-Nyarko, and Šperac [RHNv15] used a simple ML model to predict river temperature using only the mean air temperature as an input, finding that the MLP model outperformed LR and stochastic modelling.

2.6 MULTI-OUTPUT MODELLING

In thermal modelling, we are often interested in modelling many different variables, such as temperature and relative humidity. When the regression outputs are not correlated, a simple solution would be to build n independent models, that is, a model for each output. Each of the n models could then be used to independently predict the n outputs. However, it is likely that the output values related to the same input are themselves correlated. Therefore, it is ideal to build one regression model that is capable of simultaneously predicting all n outputs. This approach is referred to as multi-output, or multi-target, regression which maps the same set of inputs to multiple outputs [Xu+20]. The goal is to learn a function f that maps a single input sample to multiple outputs. For the case of linear regression, given

multiple outputs $\mathbf{y} = y_1, y_2, \ldots, y_m$ and inputs $\mathbf{x} = x_0, x_1, x_2, \ldots, x_p$ the equation is

$$y_m = \beta_0 + \sum_{j=1}^{p} x_j \beta_{jm} + \epsilon_m \tag{2.21}$$

$$= f_m(x) + \epsilon_m, \tag{2.22}$$

where ϵ_m is an m-dimensional vector or error that are assumed to be normally distributed with mean 0. In matrix notation, this can be represented as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2.23}$$

where $\mathbf{Y}$ is an $N \times m$ response matrix, $\mathbf{X}$ is an $N \times (p+1)$ input matrix, $\boldsymbol{\beta}$ is a $(p+1) \times m$ matrix of parameters, and $\boldsymbol{\epsilon}$ is an $N \times m$ matrix of errors. Given this, the least squares estimates match the form given before

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \tag{2.24}$$

In this case, multiple outputs do not affect other estimates, that is, the coefficient of the kth outcome is the regression estimate for $\mathbf{y}_k$ on $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_p$.

There are a few benefits of adapting this approach, first, the training time is lower as only one estimator is being built. Second, the estimator may also experience an increased generalization accuracy. Kocev et al. [Koc+09] tested the difference between single- and multi-output models for decision trees and found no significant difference in predictive power, however, the model was simpler and faster to learn for one multi-output model versus many single-output models.

2.7 TIME SERIES ANALYSIS

Time series analysis is a technique used for studying data generated sequentially over time, which is usually evenly spaced. In time series, the goal is often to make a forecast for the future. An assumption for many time series models is that the data are stationary, meaning the mean, variance, and autocorrelation do not change over time. In most real-world data, however, these statistical properties vary. The time series data could have systemic components, such as a trend or seasonality. For example, due to the seasonal nature of our climate, weather data would be non-stationary.

One approach to account for the non-stationary nature of data would be to difference transform. This will remove the temporal dependence of structures like trends or seasonality. The difference is calculated between time stamps

$$x_{\text{diff}} = x_t - x_{t-1}. \tag{2.25}$$

If the conversion back to the original scale is necessary, such as in the case of prediction, then the inverted differences can be used

$$x_{inv} = x_{diff} + x_{t-1}. \tag{2.26}$$

A lag-1 difference takes the difference between consecutive time stamps and can be adjusted for temporal components such as seasonality.

2.7.1 *Autoregressive models*

Autoregressive models assume the current value z_t is dependent on the previous values $(z_{t-1}, z_{t-2}, \ldots)$. That is, a kth order autoregressive model, or AR(k), is a multiple linear regression where the value of the series at time t is a linear function of values at times $t-1, t-2, \ldots, t-k$:

$$z_t = \beta_0 + \beta_1 z_{t-1} + \beta_2 z_{t-2} + \ldots + \beta_k z_{t-k} + \epsilon_t. \tag{2.27}$$

Moving average models assume the current value z_t is dependent on the error terms, including the current error $(\epsilon_t, \epsilon_{t-1}, \ldots)$. In order to assess the order of autoregressive and moving average models, we can examine the autocorrelation and partial autocorrelation functions of the time series data, assuming the underlying time series is stationary.

The stationary assumption means that the covariance between z_t and z_{t+k} separated by lag k must be the same for all times t. This gives us the autocovariance at lag k, defined as

$$\gamma_k = \text{cov}[z_t, z_{t+k}] \tag{2.28}$$

$$= E[(z_t - \mu)(z_{t-k} - \mu)]. \tag{2.29}$$

The autocorrelation function is a calculation of the correlation of successive time observations in the same series. An autocorrelation at lag k is

$$\rho_k = \frac{E[(z_t - \mu)(z_{t+k} - \mu)]}{\sqrt{E[(z_t - \mu)^2]E[(z_{t+k} - \mu)^2]}} \tag{2.30}$$

$$= \frac{E[(z_t - \mu)(z_{t+k} - \mu)]}{\sigma_z^2}. \tag{2.31}$$

Note that $\gamma_k = \rho_k \sigma_z^2$, therefore knowledge of the autocorrelation function $\{\rho_k\}$ and variance σ_k^2 is equivalent to knowledge of the autocovariance function $\{\gamma_k\}$. For a stationary process, the variance $\sigma_z^2 = \gamma_0$ is the same at time t and $t+k$, therefore, the autocorrelation at lag k, or the correlation between z_t and z_{t+k}, is $\rho_k = \frac{\gamma_k}{\gamma_0}$ and $\rho_0 = 1$. The calculated correlation coefficient can range from -1 to 1, representing a perfectly negative or perfectly positive relationship respectively. The autocorrelation function plot, which shows the autocorrelation coefficient ρ_k as a function of the lag k, is often used to assess whether

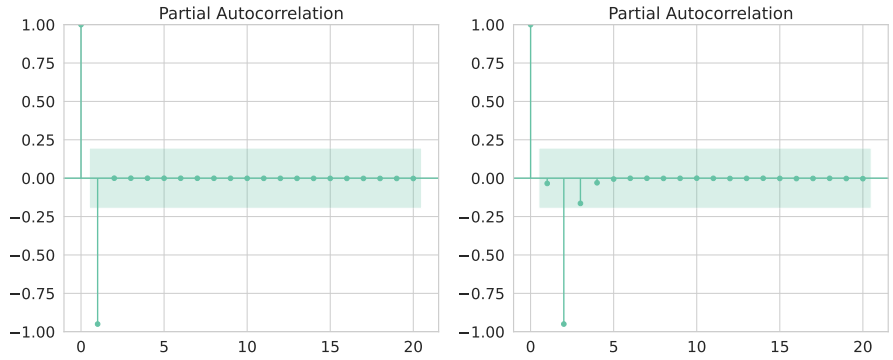

Figure 2.1: Two examples of partial autocorrelation plots. The figure on the left shows a strong negative partial autocorrelation at lag 1 only, whereas the figure on the right shows a strong negative partial autocorrelation at lag 2 only.

a time series model and the incorporation of lags is appropriate for the data. For random data, the autocorrelations will be near zero, and if the data is non-random there will be at least one significant lag. Non-random data is a good indication that time series analysis is relevant.

Partial autocorrelation, on the other hand, captures the relationship between two observations in the time series after controlling for the effects of the other variables. Essentially, the partial autocorrelation function uses a regression model to get these "direct" correlations. For an AR(p) process, the partial autocorrelations ϕ_{kk} will cut off after lag p, that is the partial correlation will be non-zero for $k \leqslant p$ and zero for $k > p$. The partial autocorrelation ϕ_{kk} of the process $\{z_t\}$ at lag k is equal to the partial correlation between variables z_t and z_{t-k} adjusted for variables $z_{t-1}, z_{t-2}, \ldots, z_{t-k+1}$ [BJ90]. The partial autocorrelations $\phi_{k1}, \phi_{k2}, \ldots, \phi_{kk}$ are the regression coefficients $\beta_1, \ldots, \beta_k$ that minimise $E[(z_t - \beta_0 - \sum_{i=1}^{k} \beta_i z_{t-i})^2]$ in the linear regression of z_t on $z_{t-1}, z_{t-2}, \ldots, z_{t-k}$.

One way to assess the partial autocorrelation function is to make a plot and visualise how many autoregressive terms are needed to explain the autocorrelation pattern in a time series. If the partial autocorrelation is significant at only two lags and the remaining lags are zero, this suggests using an autoregressive term of 2. Figure 2.1 shows two examples of partial autocorrelation plots. For the figure on the left with a strong partial autocorrelation at lag 1 only, this means to predict the next state x_{t+1} only one previous value is required, that is, the current state x_t. The figure on the right shows a strong partial autocorrelation at lag 2 only and not at lag 1, so in order to predict the next state variable x_{t+1} only the value from 2 time ago is required, that is, x_{t-1}.

The concept of lagged variables has been widely used across literature for thermal predictions. Hanoon et al. [Han+21] utilised an MLP

with lag of 4 to predict monthly air temperature and relative humidity which gave an improvement of 0.08 in the coefficient of determination over the lag 1 model. Dadour et al. [Dad+11] introduced a lag of 2 to predicting the hourly air temperature in a parked vehicle. Manero, Béjar, and Cortés [MBC19] implemented a lag in their deep learning model to predict wind speed. Doolgindachbaporn [Doo21] used a time lag to help to identify issues in the thermal data, such as unexpected temperature rises.

In previous studies, variants of autoregressive models have been used for thermal modelling. Two such models are the autoregressive model with exogenous inputs (ARX) and autoregressive moving average model with exogenous inputs (ARMAX). Gustin, McLeod, and Lomas [GML18] investigated ARX and ARMAX for the forecast of indoor temperatures during a heatwave. The findings suggest that both models are able to provide accurate forecasts, with ARX providing more consistent multi-step ahead predictions. Similarly, Saifizi et al. [Sai+13] compared ARX and ARMAX models for thermoelectric refrigerator systems, finding ARX showed better prediction results. Non-linear autoregressive model with exogenous input (NARX) is a type of RNN based on ARX. This model was investigated by Ng et al. [Ng+14b] for both one- and multi-step ahead predictions of an automotive air conditioning system, showing an MLP based NARX model provided accurate predictions and was less influenced by noise in the data. A further extension of these models is the non-linear autoregressive moving average model (NARMA), or NARMAX when exogenous variables are included, which have the advantage of including information about past errors to improve predictions. Palanthandalam-Madapusi et al. [PM+07] found strong performance of NARMAX for the 27-day prediction of solar wind conditions.

2.8 CROSS VALIDATION METHODS

Cross validation is a statistical method used to estimate the performance of machine learning models on unseen data, often used to evaluate the generalisability of a model and prevent overfitting. The procedure for cross validation involves dividing the dataset into two or more subsets, training the model on one subset, and evaluating it on the other. The process is repeated a number of times, each time using a different subset as the evaluation set and averaging the results. It is important to follow a strict and correct procedure for cross validation when working with time series data, as highlighted by Nowotny [Now14]. Careful interpretation of the results is also crucial.

When working with time series data, traditional cross validation methods, such as k-fold cross validation, may not be ideal. This is because time series data exhibits temporal dependencies, meaning that the data at a given time point is dependent on the data that

came before it. These dependencies must be taken into account when splitting the data into training and evaluation sets. In addition, the choice of test set should not be random, as this could introduce bias into the evaluation.

2.8.1 *Group k-fold*

K-fold cross validation is a widely used resampling technique in which the data is divided into k equal sized groups, or folds. Each fold is used as a validation set once, and the model is trained on the remaining $k-1$ folds. The process is repeated k times, and the results are averaged over all k iterations.

While k-fold cross validation is a simple and effective method for estimating the performance of a model, it does not account for groups within the data. Group k-fold cross validation is a variation of k-fold cross validation that ensures a group is not represented in both the training and validation sets, which can help to reduce bias in the evaluation.

2.8.2 *Time series split*

K-fold assumes the samples are independent and identically distributed, but due to the autocorrelation in time series, this assumption is often not met. To address this issue, time series cross validation is a variation of k-fold cross validation that takes into account the temporal dependencies of the data. This validation approach ensures that the test indices are higher than the training indices and does not shuffle the data, which can help to reduce bias. Time series cross validation also restricts successive training sets to be supersets of the previous folds, which helps to consider the temporal dependencies of the data.

According to Varma and Simon [VS06], a nested cross validation procedure, such as time series split, provides an almost unbiased estimate of the true error. Bergmeir and Benítez [BB12] also discusses the applicability of cross-validation methods on time series data, particularly methods based on the last block, like time series split.

2.9 CHAPTER SUMMARY

This chapter summarised some of the ideas in thermal modelling literature, ranging from applications in electrical components to weather to car cabins and houses. The modelling approach also showed great variety including physics-based models, ML models, or a combination of both using hybrid approaches. Although different modelling approaches are taken depending on the environment and the data at hand, there is promise for ML models across thermal modelling environments. In particular, this chapter highlighted that various pieces

of research have found different ML models were the top performing model, even when examining similar data. The next chapter will detail the method adopted in this thesis to the problem of thermal predictions within an environment.

3 METHODS

3.1 CHAPTER INTRODUCTION

The chapter describes the methods used for modelling thermal aspects of an environment. The aim of this chapter is to provide a "recipe" that can be followed to reproduce this research. This chapter acts as a guide as to why certain decisions and considerations were made within this thesis. This involves explaining the entire data lifecycle from the raw data, to preprocessing steps, to modelling approaches and tuning, to validation. This chapter introduces methods that are common across all analyses in this thesis, if variants are used, this is clarified in the appropriate results chapters (Chapter 4, Chapter 5).

This chapter details the steps needed to reproduce the thermal models used within this thesis. In particular, this includes:

- an outline of the methodological approach;
- an introduction to new algorithms used;
- arguments for why these methods were chosen for this work.

This thesis examines two case studies, a car cabin and house, which will be discussed in detail in their relevant results chapters (Chapter 4 and Chapter 5), however, this section will provide a guide on how decisions were made for the set-up and data preprocessing without detailing the exact data and characteristics of the specific experiments.

3.2 EXPERIMENTAL SET-UP

Despite the differing environments in the two case studies, both concepts are focused on examining thermal aspects of the environment and determining if a unified approach can be successful for both. The aim is to build a machine learning model that can accurately predict thermal aspects, such as temperature and relative humidity, for various areas within the environment. The first step is to explore what thermal data is available and needed for modelling, as well as whether to consider each input feature as a state or control variable.

3.2.1 *Thermal variables*

The first step in the methodology is extracting relevant variables from the data as some of the data that has been collected during the experiment may not be of interest for the chosen model (for example,

if the geometry of the environment is measured but is not necessary as a feature for the model being tested). As discussed in Section 2.1.1, there are multiple different thermal variables that are used for thermal modelling. In this thesis, the variables available will be dependent on the experiments and data sources. However, the variables that are in common and important for both environments are external climate data, heating, ventilation, and air conditioning system (HVAC) inputs, surface temperatures, and internal thermal data. The internal thermal data consists of air temperature and relative humidity for various areas within the environment.

3.2.2 *State and control vector*

In this thesis, the aim is to make use of the current (and possibly previous) state and control variables to predict the next state, as introduced in Section 2.7.1. Therefore, in order to build a model, a decision must be taken to classify variables as either a state $\mathbf{x}$ or a control $\mathbf{u}$. State variables are used to describe the "state" of an environment. The state is defined as the observable variables that describe the behaviour of an environment and can change during a specified process [LM01]. Control variables are considered to be exogenous inputs to the model and are either controllable, uncontrollable but measurable, or indirectly controllable. For example, although the weather may not be able to be directly controlled, this can be measured, and in a controlled environment could even be "controlled".

Given this, the control variables must be defined based on this criteria. In the data examined in this thesis, the state vectors include the variables of interest, or those which a future prediction is required, which are typically thermal variables, such as air temperature and relative humidity in the environment being examined. On the other hand, the control variables consist of external thermal variables, such as climate data, and HVAC input data. Climate data includes variables such as outdoor temperature and relative humidity, while HVAC input variables include gas consumption readings in a house, which signals if the heating is on, or the set point temperature of the HVAC in a car. These variables will be listed in further detail for each thermal environment case study within the relevant chapters (Chapter 4 and Chapter 5), however this lays out the rationale for how variables should be split between the state and control vectors.

3.2.3 *Sampling rate for input data*

A suitable sampling rate is crucial for minimise missing information and capture fast-changing conditions. For the case studies examined, it is important to consider the level of detail required, as well as the responsiveness of the HVAC systems and computational resources.

The car cabin data is sampled at either 1 or 10 second intervals, whereas the house is captured in 30 minute intervals. In Section 4.5.2, various time intervals will be considered and compared, to investigate the effect of these choices on the selected models.

3.3 DATA PRE-PROCESSING

Once the features have been identified as state or control variables, the next step is to preprocess the data. This includes handling missing data, scaling and transforming, and examining autocorrelation to identify if one or more lag is required for time series data. A lagged variable is a variable that is measured or calculated at a previous time point. Lagged variables are used to capture the effect of a variable on a current or future outcome, while accounting for any potential delays or lags in the effect.

The data pre-processing steps introduced here are quick to implement, taking seconds to arrive at the result. However, the time spent investigating how to handle missing data can take a researcher more time as this is often less straightforward and will depend on the given data.

3.3.1 *Lagged variables*

As introduced in Section 2.7.1, autoregressive models forecast the variables of interest using a linear combination of the past values of the variable. To give the model the ability to retain memory, an autoregressive model is implemented, which includes the application of lagged variables. In order to access what order model (that is, the number of lags) is appropriate for the data, the autocorrelation and partial autocorrelation plots can be analysed. Based on the findings from the autocorrelation and partial autocorrelation functions, the data can then be shifted to implement the appropriate lag. If a lag of 2 is deemed appropriate, that is, there is a strong correlation at both lag 1 and 2, this represents a correlation between values that are 1 and 2 time periods apart from the current value. This implies that the current value of a variable is dependant on its previous values, therefore, introducing a lagged variable allows the model to capture patterns within a variable itself and improve predictions. The data can then by restructured for a lag of 2 such that at any time t, not only are the current state x_t and control u_t used as inputs to predict the next state x_{t+1}, but also the previous time point, that is, the state variable x_{t-1}.

Figure 3.1 shows a simple example of a dataset with one state variable (indoor air temperature) and one control variable (outdoor air temperature) which has been transformed to include a lag of 1. Note that during the data transformation for lagging as many rows

time (t)	indoor air temperature (x)	outdoor air temperature (u)
0	18.5	9
1	18	9.5
2	17	9
3	17.5	10

time (t)	indoor air temperature (x_1)	indoor air temperature (x_2)	outdoor air temperature (u_1)
1	18.5	18	9.5
2	18	17	9
3	17	17.5	10

Figure 3.1: An example of transforming data to have 1 lagged variable where indoor air temperature is considered a state variable x and outdoor air temperature is considered a control variable u.

as there are lags will be dropped from the data. This is because there is no previous data to fill, that is, in Figure 3.1 there is no indoor air temperature reading prior to time $t = 0$ that could be used to fill the data in the right table for the state x_1. By introducing a lag into the model, this brings the issue of multicollinearity, which is discussed in the following section.

3.3.2 *Multicollinearity in predictive models*

With time series data, in particular data with lagged variables, the presence of multicollinearity is expected due to the high correlation of a variable to itself. Multicollinearity is when the independent variables in the model are highly correlated. This can be problematic when determining the true effect of the variables on the response. However, multicollinearity in the data does not affect the models prediction accuracy or goodness-of-fit [Kut+04, Chapter 7.6]. Therefore, as the predictions are the main focus of this thesis, the multicollinearity problem does not need to be directly addressed.

However, multicollinearity would be a problem when looking at the feature importances of these models. This will be discussed further in Section 3.9.

3.3.3 *Missing data*

An important step in pre-processing is considering how to handle missing data. Ultimately, missing data must either be dropped or filled using methods such as imputation. This could include approaches such as linear interpolation or nearest neighbors imputation. If the dataset is reasonably large, it may be acceptable to drop the missing data. The decision of how to handle missing data depends on why data is missing [Mol+14]. For the data in this thesis that is collected in a controlled environment for a set period of time, missing data is not an issue, but for real-world data, data loss can happen for many reasons. If the missing data is not missing at random, then imputing

data could potentially introduce bias into the machine learning (ML) model. Due to the nature of sensor data, missing data could occur as a result of power outages or connectivity issues. To address this concern of introducing bias into the model, if a low proportion of data is missing and the dataset is reasonably large, the entries with missing data will be dropped. Dropping data is done after variables are lagged to ensure that the correct previous timestamps are used.

3.3.4 *Scaling and transforming*

The scaling technique chosen normalises the data by subtracting the minimum value of the feature from the current value, and dividing by the range of the feature

$$y_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

where x is the data and x_i is the ith observation. This scaler is chosen as it preserves the original distribution of the data and does not reduce the importance of any outliers. For the analysis in this thesis, the data scaling is implemented using Python's sklearn `MinMaxScaler()` command and then using `fit_transform()` to transform over all of the data [Ped+11].

Although some models do not require rescaling, others are very sensitive to unscaled data, such as neural network (NN). Scaling is necessary for numerical stability during the optimisation process and allows parameter coefficients to become directly comparable. Therefore, scaling and transforming is applied to the data for all models for consistency. Scaling is performed prior to the handling of missing data, however, due to the nature of the scaler this should not produce any issues as the min-max scaling approach allows for predictions outside the min-max range.

3.4 MODEL SELECTION

Given the current state $y(t)$, a transient simulation must identify $y(t + \Delta t)$ for some small increment in time Δt, say 1 s. An Euler simulation is a numerical approximation that assumes for a small Δt,

$$\frac{\Delta y}{\Delta t} \approx \frac{dy}{dt} = -k(y - y_0). \tag{3.2}$$

Given this approximation,

$$\begin{aligned} y(t + \Delta t) &\approx y(t) + \frac{\Delta y}{\Delta t} \cdot \Delta t && (3.3)\\ &= y(t) - k\Delta t \cdot y(t) + k\Delta t y_0 && (3.4)\\ &= (1 - k\Delta t) \cdot y(t) + k\Delta t y_0 && (3.5) \end{aligned}$$

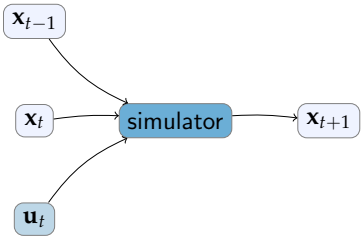

Figure 3.2: A model phrased as a recursive function, which is trained with one-step examples.

which shows that $y(t + \Delta t)$ is a linear function of $y(t)$. Therefore, assuming Δt is small, a simple linear regression between $y(t)$ and $y(t + \Delta t)$ would yield the key coefficients in what otherwise appears to be a complex relationship between the internal temperature, the external temperature, and the thermal resistivity of the dividing wall. In summary, the simple thermal problem can be modelled using a linear correspondence between the current state $y(t)$ and the next state $y(t + \Delta t)$. Finding the coefficients for such a dynamical system is termed system, or model, identification.

Although a simple linear correspondence may be sufficient for a simple system as the complexity of the model increases, non-linearities will appear. Furthermore, some effects, such as radiative heat transfer, are proportional to the difference in the fourth power of temperatures and thus seem to demand a more flexible modelling method. As suggested by work on non-linear autoregressive network systems, some form of NN or recurrent neural network (RNN) may be appropriate [Eng+19; Ng+14a].

A key insight in this work is the realisation that many physical systems can be predicted using only the current state and control inputs. In some cases, the prior state is also needed (for example, as a proxy for the velocity of a moving object where the state contains just its position). Therefore, the structure of the simulator is a transfer function of the form

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t-1}).$$

for a single previous state (time $t - 1$), corresponding to a network structure as shown in Figure 3.2.

3.5 HYPERPARAMETER SEARCH

Many machine learning models have parameters that can be configured, either regarding the model itself or the objective function. The choices of configurations of the hyperparameter values is known as the search space or hyperparameter space. The search space can contain categorical, continuous, or conditional hyperparameters. Depending on the model, these search spaces can become complex and have condi-

tional hyperparameters, such as when several optimisation algorithms could be configured which, in themselves, contain hyperparameters.

A hyperparameter search is important to select a well-performing model. However, hyperparameters searches can often take a long time and be computationally expensive.

There are multiple hyperparameter search methods used in practice. One of these is a grid search, which is an exhaustive search through manually specified hyperparameters. For each value added to a hyperparameter the number of evaluations increases exponentially, which is especially problematic for models with many hyperparameters, such as multilayer perceptron (MLP). An alternative is to try a random search, which consists of applying random sampling to the search space to select a chosen amount of models. A random search is useful as it is simple to implement and is quicker than an exhaustive grid search.

While grid and random searches are valid choices for hyperparameter searches, they are often time consuming and can spend time searching spaces that have generally poor performance. Another alternative is to use Bayesian optimisation, which makes informed decision on where to search next in the hyperparameter space. The goal of Bayesian optimisation approaches is to minimise - or maximise, depending on the metric - the objective function using information from past evaluations to inform the next hyperparameter values. One available tool is the `hyperopt` library in Python which adapts a Tree of Parzen Estimators approach to Bayesian optimisation [BYC13].

Reproducibility is important not only to ensure correct results, but also to aid in understanding and transparency. Adapting this approach, this section will detail how each of the machine learning models used throughout this thesis were implemented, including details of the hyperparameter searches. The hyperparameter search is conducted using `hyperopt` with a training set of 80 % and testing set of 20 %, conducting 1000 evaluations of hyperparameter settings (or number of models fit) [BYC13]. As the data within this thesis is time series, the first portion of data is used for the training set and shuffling is applied, then the subsequent data is used as the testing set. The models tested include linear regression, regularisation techniques, neighbors-based models, tree-based models, boosting methods, and MLP, which have been detailed and introduced within the thermal modelling setting in Section 2.5. The following sections detail how the hyperparameter search is used to tune the parameters of each model. All models were implemented using scikit-learn in Python [Ped+11].

3.5.1 *Linear and regularisation methods*

Linear regression and its related regularisation methods are fairly straightforward models with minimal parameters to optimise. This

section will detail the hyperparameters for linear, ridge, lasso and elastic net regression.

Ordinary least squares linear regression simply finds the best fitting line for the data. Therefore, linear regression does not have any hyperparameters to tune and the base `LinearRegression()` is used fit to the data [Ped+11].

For the regularisation methods below, it is worth noting that scikit-learn denotes the penalty parameter as alpha, rather than lambda λ, and the ratio of penalties as L_1 rather than alpha α. For clarification, the sections below use the terms λ and α for the penalty and ratio parameters, respectively, as defined in Section 2.5.2 not as scikit-learn has defined them.

Ridge regression is implemented using the `RidgeCV()` function [Ped+11]. For ridge regression, there are three hyperparameters to optimise. The main hyperparameter of interest is the regularisation penalty lambda, which controls the amount of shrinkage. When λ is 0, this is equivalent to a linear regression. The search space for the parameter λ is a log normal distribution with mean 0.2 and standard deviation 2, which constrains the value of λ to be positive and with most values close to zero. Alongside this, the solver can also be varied, with the chosen options including cholesky, least-squares (lsqr), stochastic average gradient descent (sag), and the adjusted stochastic gradient descent (saga). The cholesky solver uses linear algebra to obtain a closed-form solution, whereas lsqr is a fast, iterative approach that uses regularised least-squares. The sag solver utilises stochastic average gradient descent, and saga provides an unbiased, improved version of this. The final hyperparameter to be optimised is whether to fit an intercept for the model or not.

Lasso regression is implemented using the `MultiTaskLassoCV()` function [Ped+11]. For lasso regression, the main hyperparameter to be optimised is the regularisation penalty, λ. The search space for λ is a log normal distribution with mean 0.2 and standard deviation 2, which is the same as for the ridge parameter. Lasso has the additional benefit of acting as a feature selection method by shrinking some coefficients to zero.

For elastic net regression, three hyperparameters can be optimised, namely λ, the L_1 ratio α, and whether to fit an intercept. Similar to ridge and lasso, λ is selected from a log normal with mean 0.2 and standard deviation 2 The α parameter is as described in Section 2.5.2, where a value of 0 indicates that the penalty is an L_2 penalty (ridge), 1 is an L_1 penalty (lasso), and any value between 0 and 1 is a ratio representing a combination of both. Therefore, in order to obtain this ratio, the value for α is selected from a uniform distribution with minimum 0 and maximum 1. Elastic net regression is implemented using the `MultiTaskElasticNetCV()` function [Ped+11].

3.5.2 *K-nearest neighbours*

K-nearest neighbours (KNN) is a non-parametric method which approximates by averaging over the K closest observations in the same neighbourhood. The size of each neighbourhood is determined such that the mean squared error (MSE) is minimised. For this thesis a KNN regression is utilised, although this technique can also be used for classification.

KNN is implemented using the `KNeighborsRegressor()` function [Ped+11]. In this thesis, four KNN hyperparameters are tuned during the search, including the choice of algorithm, the weight function, the leaf size, and the number of neighbors. The algorithm to use for computing the nearest neighbors is either ball tree, k-dimensional tree, or a brute-force search. The weight function is chosen as either uniform, giving equal weights to points in each neighbourhood, or distances, where closer neighbors provide a greater influence. The leaf size and number of neighbors are selected as an integer between 1 and 50.

3.5.3 *Regression tree*

A decision tree is a non-parametric supervised learning model that makes predictions based on simple decisions rules that are inferred from the data features. A decision tree is often referred to as a weak learner and is susceptible to overfitting. When decision trees are used for a regression problem, they are often referred to as regression trees.

The decision tree algorithm is implemented using the function `DecisionTreeRegressor()` [Ped+11]. Regression trees have multiple hyperparameters to optimise. The six hyperparameters being optimised here are maximum depth, minimum number of samples required to split an internal node, minimum number of samples at a leaf node, minimum weighted fraction of the sum total weights at a leaf node, maximum number of features when finding the best split, and maximum number of leaf nodes. The following search spaces are used for the hyperparameters:

- max depth: integer between 1 and 50 or none
- min samples leaf: uniform with minimum 0.01 and maximum 0.49,
- min samples split: uniform with minimum 0 and maximum 1,
- min weight fraction leaf: uniform with minimum 0.01 and maximum 0.49,
- max features: base-2 logarithm or square root,
- max leaf nodes: an integer between 2 and 100 or none.

3.5.4 *Random forest*

Random forest is an extension of decision trees which makes use of bagging, or bootstrap aggregation, which is an ensemble learning technique used to reduce the variance of a prediction model. Random forest builds multiple decision trees in parallel and provides an average estimation of the individual trees. During the ensemble construction, random features are dropped to try to reduce the correlation amongst trees.

Random forest is implemented using the `RandomForestRegressor()` function [Ped+11]. Random forest has similar hyperparameters to regression trees, adding one parameter for the number of estimators. The six other hyperparameters are the same, including the maximum depth, minimum number of samples required to split an internal node, minimum number of samples at a leaf node, minimum weighted fraction of the sum total weights at a leaf node, maximum number of features when finding the best split, and maximum number of leaf nodes. The following search spaces are used for the hyperparameters:

- n estimator: an integer between 10 and 100,
- max depth: integer between 1 and 50 or none
- min samples leaf: uniform with minimum 0.01 and maximum 0.49,
- min samples split: uniform with minimum 0 and maximum 1,
- min weight fraction leaf: uniform with minimum 0.01 and maximum 0.49,
- max features: base-2 logarithm or square root,
- max leaf nodes: an integer between 2 and 100 or none.

3.5.5 *Extremely randomised trees*

Extremely randomised (or extra) trees implements a meta estimator that fits multiple randomised decision trees on various subsamples of the data. The average is used to improve predictive accuracy and help control over-fitting.

Extra trees is implemented using the `ExtraTreesRegressor()` function [Ped+11]. The hyperparameters for extra trees are the same as for random forest, including the number of estimators, maximum depth, minimum number of samples required to split an internal node, minimum number of samples at a leaf node, minimum weighted fraction of the sum total weights at a leaf node, maximum number of features when finding the best split, and maximum number of leaf nodes. The following search spaces are used for the hyperparameters:

- n estimator: an integer between 10 and 100,
- max depth: integer between 1 and 50 or none
- min samples leaf: uniform with minimum 0.01 and maximum 0.49,
- min samples split: uniform with minimum 0 and maximum 1,
- min weight fraction leaf: uniform with minimum 0.01 and maximum 0.49,
- max features: base-2 logarithm or square root,
- max leaf nodes: an integer between 2 and 100 or none.

3.5.6 *XGBoost*

Boosting is an ensemble method that reduces bias and variance by converting weak learners into strong ones. Gradient boosted decision trees is a generalisation of boosting that allows for the optimisation of arbitrary differentiable loss functions.

An extension of gradient boosting is extreme gradient boosting, or XGBoost, which is an ensemble method where trees are sequentially added into the model to improve the accuracy by adding a base learner to correct shortcomings of the existing base learners. At each step of the ensemble construction, the XGBoost algorithm adds a decision tree and aims to improve upon any previous steps with each iteration.

The XGBoost algorithm is implemented using the `xgboost()` package in Python [CG16]. Eight hyperparameters are optimised, including the learning rate (eta), maximum depth, minimum child weight, subsample ratio, subsample ratio of columns when constructing a tree, the minimum loss reduction gamma, the L1 regularisation term alpha, and the L_2 regularisation term lambda. The following search spaces are used for the hyperparameters:

- eta: log uniform $-7, 0$,
- max depth: integer between 1 and 100,
- min child weight: log uniform $-2, 3$,
- subsample: uniform with minimum 0.5 and maximum 1,
- subsample of columns for tree: uniform with minimum 0.5 and maximum 1,
- gamma: log uniform $-10, 10$,
- alpha: log uniform $-10, 10$,
- lambda: log uniform $-10, 10$.

3.5.7 *MLP*

Having preprocessed the data, random hyperparameter search is used to identify the best NN or MLP structure [Aur19]. The parameter space to search over is:

- number of hidden layers ($0 - -4$).
- activation for hidden layers (ReLU, sigmoid, tanh, linear).
- activation for final layer (linear).
- number of nodes in each hidden layer ($64 - -1024$).
- whether a dropout regularisation, a technique where randomly selected neurons are ignored during training [Sri+14], is used (selected from the set $0.05, 0.75$).
- batch size ($28 - -128$).
- optimizer (Adadelta, Adam, or Adamax).
- epochs ($50 - -1500$ by 10)

The final activation layer is chosen to be linear as this thesis examines data that forms a regression problem with real-valued outputs.

The network is learnt using the `Sequential` class from TensorFlow [Aba+15] and Keras [Cho+15] libraries with the Adam optimizer [KB14b] aiming at minimising the MSE. From the hyperparameter space given above, 200 variants are selected at random (with uniform distribution for number of nodes). Input and outputs are rescaled to a unit range using min-max rescaling, `MinMaxScaler()`.

3.6 TIME SERIES GROUP SPLIT CROSS VALIDATOR

Two considerations are made when performing cross-validation on time series data presented in this thesis. First, whether the data comes in the form of groups, or different trials/experiments. In the car cabin data, which will be discussed in Chapter 4, there are five different trials done on the same car in the same environment, but with different settings, so each trial therefore represents a group within the data. This will not be the same for the house data in Chapter 5, where rather than considering houses as groups in the data, an individual model will be built for each house as they are different structures.

A second consideration is that order is important for time series data as autocorrelation is often present. Therefore, the selection of folds for time series data should ensure the testing set is at a timestamp following the training set. Shuffling can be done and in some cases, such as for MLP, is necessary, but must be done with caution. The test set is always selected to be after the training set in terms of the original

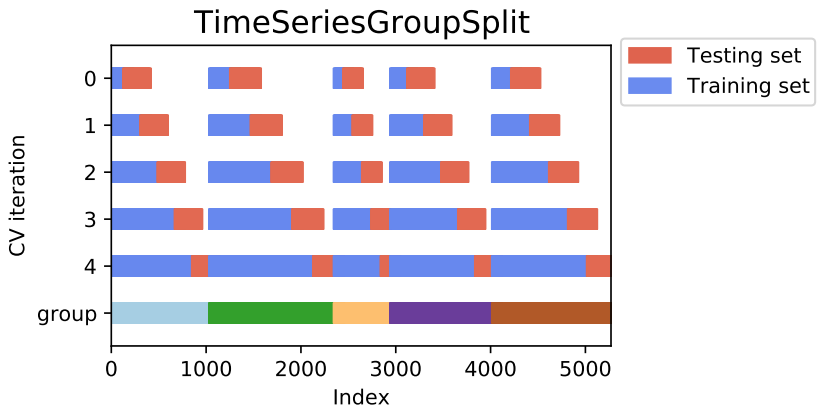

Figure 3.3: Visualisation of the Time Series Group Split cross validation splits on CWT trial data.

timestamps regardless of any shuffling performed In practice, this will mean that the first timestamps will be assigned to the training set and the following observations assigned to the test set. The training set is then shuffled, the parameters are estimated, and predictions are made on the test set.

In Section 2.8, Group K-fold and Time Series Split cross validators were introduced, which address each of these considerations individually, but fail to address both aspects. In order to address this, a Time Series Group Split cross-validator is developed, which is based on the `TimeSeriesSplit` cross-validator function in scikit-learn [Ped+11]. This requires that each successive training set is a superset of those before it within each of the groups. A similar approach to this has been pursued as a scikit-learn feature request on GitHub [Gri19], although an assumption of their approach is that groups are contingent, or together in a sequence. This does not support our data as the groups do not form a sequence. A visualisation of how the developed Time Series Group Split cross-validator works on the climatic wind tunnel (CWT) trial data from Chapter 4 is shown in Figure 3.3.

The Time Series Group Split cross-validator with shuffling works as follows:

- Split data into groups. In Figure 3.3, this is shown through the bottom bar labelled group, which contains data from our 5 different groups.
- Repeat the following for each split. In the visualisation, 5 splits are chosen (0–4).
  - For each group, take the split portion of train and test indices such that the test indices immediately follow the training indices. The blue represents the training test within each of our 5 groups followed by red which represents the testing set for each group.

– Shuffle the training indices within each group. There is no need to shuffle the test set. This means the blue training set is shuffled and the red testing set retains its order.

Model training is done in open loop, meaning the true output rather than the predicted one is used for the next step predictions [MB08].

3.7 LONG-RUN SIMULATOR USING MACHINE LEARNING

Once the model is chosen from the one-step analysis, then the performance is evaluated over a longer period of time, for example, the duration of the data available. As mentioned in Section 3.6, the training is done in open loop, but for the longer term predictions closed-loop is utilised. This means that the predicted value is used to predict the next step, rather than the observed data.

3.8 EVALUATION METRICS

In order to measure the performance of our model, various metrics can be utilised that are applicable to regression problems, including mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), normalized root mean squared error (NRMSE), and R-squared.

The MAE is a single-output method that computes the difference between the predicted and actual output values. Given some set of N measurements $Y = y_1, y_2, \ldots, y_N$ and some estimate of those values $\hat{Y} = \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N$,

$$\mathrm{MAE}_{Y,\hat{Y}} = \frac{1}{N}\sum_{i=1}^{N} |\mathbf{y}_i - \hat{\mathbf{y}}_i|. \tag{3.6}$$

The MSE is a measure of how close a fitted line is to data points, that is, the smaller the MSE, the closer the fit is to the data

$$\mathrm{MSE}_{Y,\hat{Y}} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y}_i)^2. \tag{3.7}$$

The MSE is more sensitive to outliers than the MAE. Alternatively, the RMSE can be used, which is simply the square root of the MSE

$$\mathrm{RMSE}_{Y,\hat{Y}} = \sqrt{\mathrm{MSE}_{Y,\hat{Y}}}. \tag{3.8}$$

The RMSE is often preferred because it has the same units as the quantity being estimated, and can therefore be interpreted directly. Further, the NRMSE or percentage error is the error in terms of the possible range of values

$$\mathrm{NRMSE}_{Y,\hat{Y}} = \frac{\mathrm{RMSE}_{Y,\hat{Y}}}{y_{max} - y_{min}} \cdot 100\%. \tag{3.9}$$

The NRMSE can aid in the comparison between datasets or models with different scales.

3.9 FEATURE IMPORTANCE

As discussed in Section 3.3.2, the presence of multicollinearity affect the interpretation of statistical models. If feature importance is done when multicollinearity is present, this can cause inflated standard errors of the coefficients, making it difficult to determining the contribution of a unique feature. One approach to interpret feature importances when multicollinearity is present is regularisation.

Regularisation methods are often used to prevent overfitting and improve the generalisability of a statistical model. In addition, lasso and elastic net regression can also be used as feature selection methods. Given this, these two approaches can be used to identify the most important features in a dataset where multicollinearity is present, although this should be done with caution as the standard errors of the coefficients could be inflated. One way to mitigate this is to scale the data before fitting the model, as described in Section 3.3.4, which removes the scaling effects of multicollinearity and makes the regularisation parameter more effective.

3.10 A MACHINE LEARNING FRAMEWORK

Throughout this chapter, a comprehensive array of machine learning methods applied in data analysis has been introduced, encompassing every step from handling raw data to evaluating models. The steps given below can be combined to build a machine learning framework for thermal modelling.

The experimental set-up and aspects of the data pre-processing are not featured in this framework as they require some degree of flexibility depending on the application or environment, as well as the data at hand. For example, the data pre-processing steps for lagged variables and missing data may not be the same and require human input. As an example of this, in the following chapters, one uses a lag of 1 while the other uses a lag of 2. In the same way, the missing data could be handled in various ways depending on why the data is missing and how much is missing. However, the other steps of this process could be built into a framework to be applied across various applications. More specifically, once the missing data is handled and the number of needed lags has been calculated, the framework shown in Figure 3.4 can be standardized. To determine the train/test set as well as apply the ML models, it will nee to be specified whether the data has groups present. If the data has groups, the time series group split cross validator is used, otherwise the the time series split cross

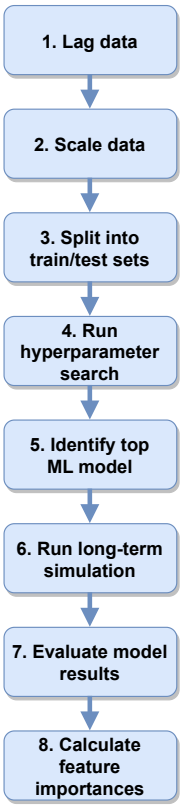

Figure 3.4: Standardized ML framework used in this thesis.

validator is utilised. In this thesis, these steps are applied to both the house and car cabin case studies.

3.11 CHAPTER SUMMARY

This chapter has introduced the methods used in the experiments in the following chapters. The process involves extracting the relevant data and classifying variables into either a state or control vector. The time series data is then analysed to determine whether lagged variables are appropriate for the data at hand. Once the lag is implemented, missing data is handled and scaling is applied. For the ML models, there are several hyperparameters which can be tuned, as laid out in this chapter. Chapter 2 showed promise for ML techniques in thermal modelling, therefore various ML models are tried and compared in this thesis. Once the ML hyperparameters are tuned, the model is built and implemented using cross validation. Multiple model metrics are calculated and compared, as well as an analysis of feature importances.

Following this approach, two case studies are pursued in Chapter 4 and Chapter 5. The next chapter will present the application of these methods to a car cabin.

4

THERMAL ML MODEL FOR CAR CABIN

4.1 CHAPTER INTRODUCTION

This chapter will introduce the experimental data for the car cabin case study, detail the analysis performed, and examine the results. More specifically the structure of this chapter is as follows:

1. An introduction to the climatic wind tunnel (CWT) experimental data (Section 4.2), as well as the preprocessing steps (Section 4.3);
2. Details of the hyperparameter search results for the tested multi-output machine learning (ML) models (Section 4.4);
3. A comparison of a variety of ML approaches, presenting the results of the chosen model for both one-step and longer term performance (Section 4.5 and Section 4.6 respectively);
4. A discussion on the most important features for the selected ML model (Section 4.7);
5. A comparison of the selected ML model to a conventional lumped thermal (0D) model, in particular examining the accuracy, speed, and capability (Section 4.8).

4.2 CLIMATIC WIND TUNNEL DATA

The car cabin data consists of five CWT trials that were conducted using a Fiat 500e vehicle. Trials 1, 3, and 4 were done at one CWT testing facility, while trials 2 and 5 were done at another. A CWT facility can be used for development and testing of vehicles by simulating realistic environmental conditions, such as outdoor temperature, wind speed, and precipitation. By utilising such facilities, manufacturers can simulate a variety of conditions without the need for on-road testing. Furthermore, the use of CWT facilities allows for repetitive, reproducible tests. The condition settings for each of the five trials are shown in Section 4.2.1, where the following are configurable: external ambient temperature (°C), vehicle heating, ventilation, and air conditioning system (HVAC) temperature (°C), air distribution (recirculation or fresh), vent (defrost/floor or neutral), moisture inside the car cabin ($g\,h^{-1}$), and car velocity ($km\,h^{-1}$).

Table 4.1: Condition settings for the 5 CWT trials. The starred recirculation or fresh items denote trials where the setting was switched halfway through.

			Part 1: HVAC and car test settings on						Part 2: Settings off
	Ambient temp. (°C)	Total duration	Duration (min)	HVAC set point (°C)	Recirc. or fresh	Defrost/floor or neutral	Moisture ($g\,h^{-1}$)	Car velocity ($km\,h^{-1}$)	Duration (min)
CWT1	-10	180	60	22	fresh	defrost/floor	0	50	120
CWT2	35	180	60	22	both*	neutral	0	100	120
CWT3	15	90	60	22	both*	defrost/floor	140	100	30
CWT4	15	180	60	16	fresh	defrost/floor	0	50	120
CWT5	35	180	60	29	both*	neutral	0	100	120

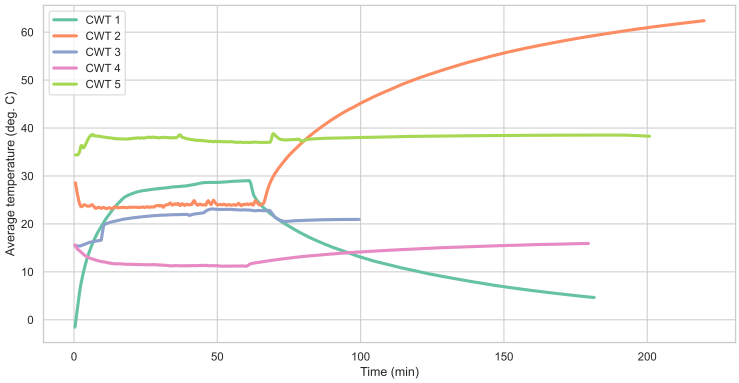

Figure 4.1: The average air temperature inside the cabin for each of the five CWT trials.

4.2.1 *CWT experimental settings*

Table 4.1 defines the configurable settings for each of the 5 CWT trials. The star (*) represents trials where the settings were varied during the test, that is, after 30 minutes the air distribution was switched from recirculated to fresh air for 30 minutes. In CWT trial 3, moisture is introduced directly into the cabin as a means to reflect the moisture typically generated by occupants, including that from respiration and perspiration.

Figure 4.1 shows the average air temperature (AT) throughout the 5 CWT trials. This is calculated by averaging the AT at the driver and passengers' three sensor locations (head, torso, and foot). For each trial, the HVAC runs for 60 minutes before the HVAC and car are switched off. This shows the temperature changes within the cabin during a time where the car sits in the external conditions without any HVAC on, equivalent to leaving a car sit in a car park. CWT trial 2 shows an increase in the average temperature after the HVAC and car are switched off. The observed response in this trial is a result of the applied solar load, simulating the effect of sunlight heating the vehicle's surfaces and leading to an increased average interior temperature.

In Figure 4.2, it is clear that there is a high correlation among many of the features. Each of these features is important for the model, that is, necessary for either thermal comfort modelling or safety measures. The main goal of this analysis is to make accurate predictions and these strong correlations will not affect the ML models capability to make predictions.

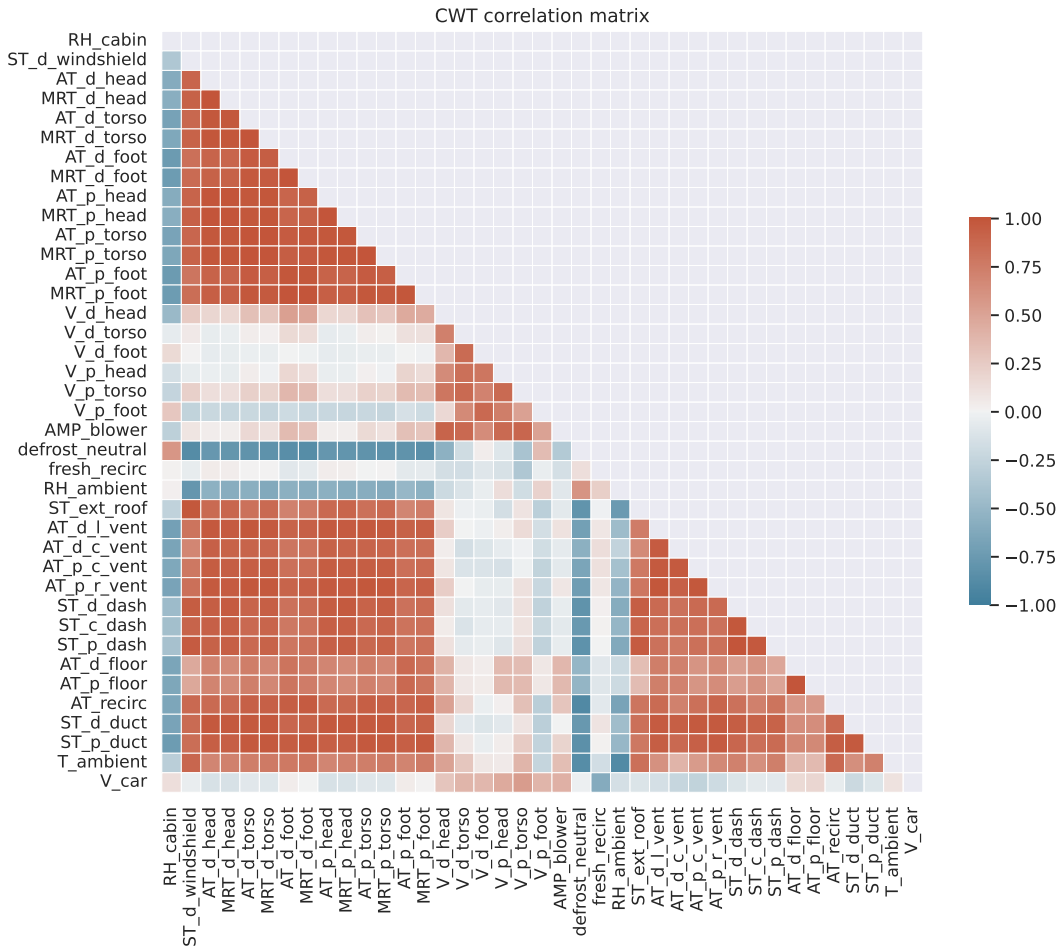

Figure 4.2: Triangular correlation matrix plot for the variables in the CWT trials where −1 indicates a perfect negative correlation, 0 indicates no correlation, and +1 indicates a perfect positive correlation.

Table 4.2: Measurement variables that comprise the control vector **u**. The air temperatures ($u_1 - u_{12}$) correspond to: vents at the side, central, floor, and near duct for driver and front passenger; recirculation inlet; and left, right, and central dashboard surface temperatures.

Control	Description	Units
u_1 - u_{12}	air and surface temperatures	°C
u_{13}	blower amperage	A
u_{14}	external roof temperature	°C
u_{15}	ambient air temperature	°C
u_{16}	ambient relative humidity	%
u_{17}	car velocity	$\mathrm{km\,h^{-1}}$
u_{18}	distribution setting	
u_{19}	vent setting	

Table 4.3: Measurement variables comprising the state **x**. Personal variables (air and mean radiant temperature and air velocity) are measured at the head, torso, and foot locations for both driver ($x_1 - x_3$, $x_7 - x_9$, $x_{15} - x_{17}$) and passenger ($x_4 - x_6$, $x_{10} - x_{12}$, $x_{18} - x_{20}$).

State	Description	Units
x_1 - x_6	air temperatures	°C
x_7 - x_{12}	mean radiant temperatures	°C
x_{13}	windshield temperature (driver's side)	°C
x_{14}	relative humidity inside cabin	%
x_{15} - x_{20}	air velocities	$\mathrm{m\,s^{-1}}$

4.2.2 *State and control vectors*

A total of 39 time series variables, measured at either 1 or 10 second intervals depending on the test facility, are divided into control (Table 4.2) and state (Table 4.3) vectors. The control variables here are either controllable (e. g., blower amperage), uncontrolled but measurable (e. g., external roof temperature), or indirectly controlled (e. g., vent temperatures). State variables include personal temperatures to allow estimation of thermal comfort (for example, via ISO 7730 [Isob] or ISO 14505 [Isoa]) and estimation of windshield fogging for safety.

In order to keep the variables names easy to interpret and meaningful, where the variables are not fully named, the following shorthand has been used throughout this chapter:

- d: driver
- p: passenger
- c: central

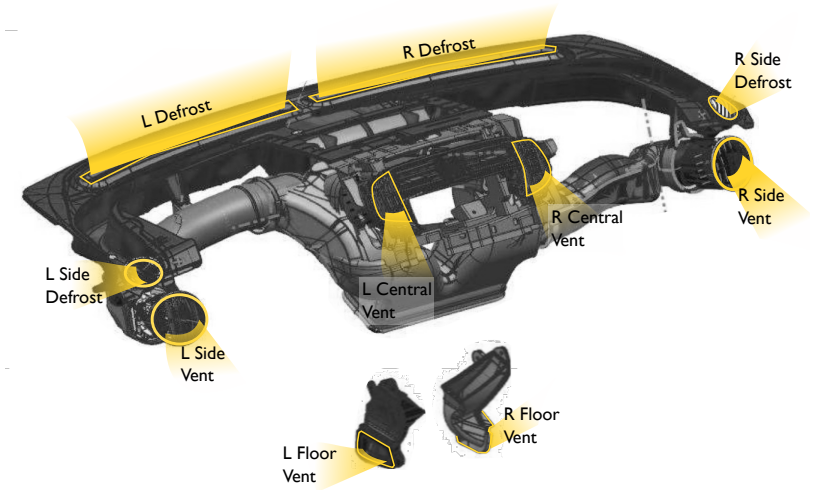

Figure 4.3: Vent outlets on instrument panel and in footwell for the CWT trials.

- l: left
- r: right
- ext: external

Figure 4.3 and Figure 4.4 show the locations for the vents and cabin sensors, respectively, that were used during the CWT trials.

Note that the data does not include the solar load. However, by including other related thermal variables such as surface and windshield temperatures, we indirectly account for the effects of solar irradiance. These variables serve as proxies for solar load. It is important to recognize that while indirect measurements help capture some aspects of solar irradiance's influence, they may not fully replicate the intricate dynamics of direct solar radiation. Thus, the model's predictions might have limitations in scenarios with significantly varying solar conditions, such as sudden changes in cloud cover or seasonal variations in solar intensity. Nevertheless, this practical approach simplifies implementation, using alternative data sources to capture some of the effects of solar load.

4.3 DATA PRE-PROCESSING

There was no missing data for any of the CWT trials, possibly because the experiments were performed in a well-controlled environment by experienced technicians. Two CWT trials (2 and 5) were sampled at 1 second, whereas the others (1, 3, and 4) were sampled at 10 seconds. To account for this, rolling means of window 50 and 5 were applied to the 1 and 10 second interval data, respectively, and then the 1 second

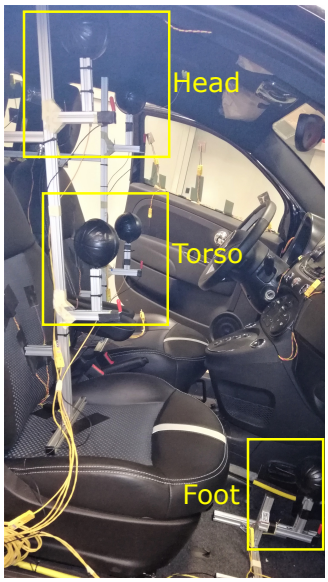

Figure 4.4: Sensors at the head, torso, and foot locations for the CWT trials.

data was subsampled to 10 second intervals. Furthermore, noise was present in some of the data so the data is smoothed using a rolling average. The smoothing is applied prior to subsampling as this is likely to be more accurate. Finally, the features are normalised from their various ranges to a common scale (that is, between 0 and 1).

Figure 4.5 shows the data from CWT 1, as an example of what the CWT data can look like. The data is split into six graphs showing different types of measurements (that is, air temperature (AT), relative humidity (RH), mean radiant temperature (MRT), surface temperature (ST), vent temperature, and air velocity (AV)) at various locations. Similar plots for the other four CWT trials can be found in Appendix B.

To test how many lags would be appropriate, as discussed in Section 2.7.1, the autocorrelation and partial autocorrelation are examined for the time series data, as shown in Figure 4.6. The autocorrelation and partial autocorrelation function plot shows that a lag 1 is best for the CWT data, meaning that only one previous time point (that is, the current value) is correlated with the next one.

After applying a lag of 1 to the data, the resulting time series data shapes are shown in Table 4.4. This is the final data that will be used during modelling.

4.4 HYPERPARAMETER SEARCH RESULTS

This section shows the results of the hyperparameter search, as laid out in Section 3.5, for the CWT data. Figure 4.7 provides an overview of the results from the hyperparameter search for the multilayer perceptron (MLP) where the final activation layer is linear. The main results shown here are that for those network structures that perform poorly, dropout

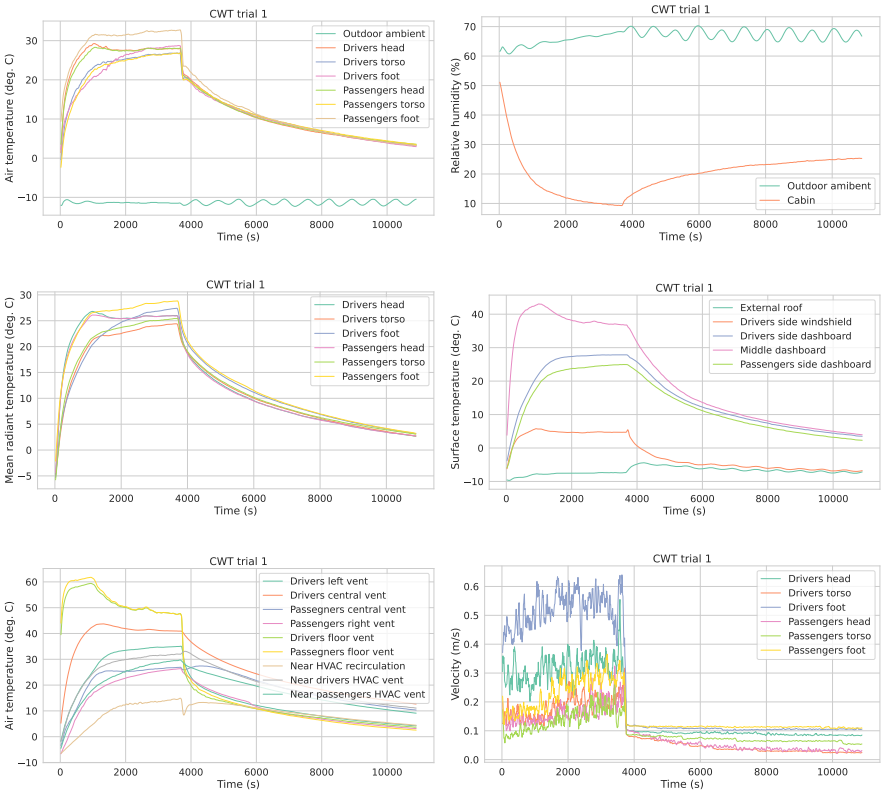

Figure 4.5: Measurements from CWT trial 1, smoothed using a rolling average with a window of 5 seconds. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

Table 4.4: The dimensions of the CWT data by trial.

CWT trial	No. rows	No. columns
1	1086	39
2	1315	39
3	594	39
4	1074	39
5	1201	39
Total	5270	39

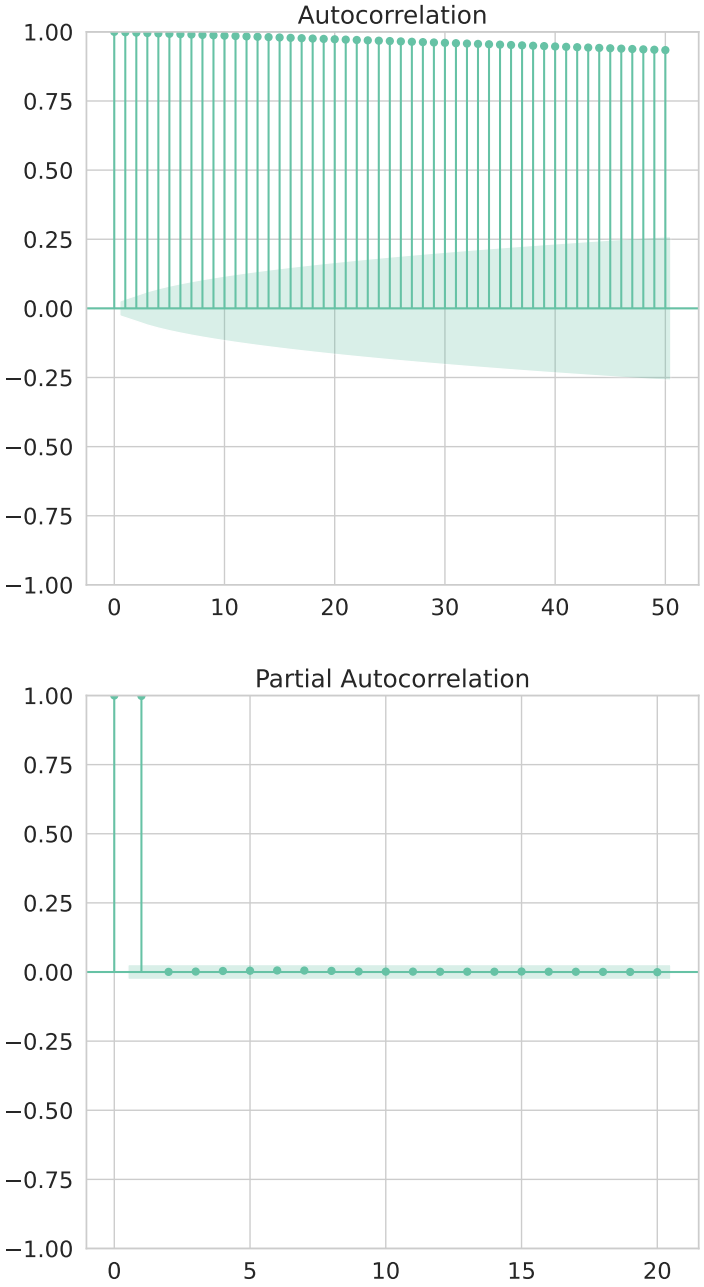

Figure 4.6: Autocorrelation and partial autocorrelation function plots for the air temperature sensor at the drivers head.

Figure 4.7: Overview of hyperparameter search showing the loss (mean squared error (MSE)) for with and without dropout and separated for each hidden activation function. For better visualisation, 8 data points are removed as their loss was between 0.1 and 0.7.

helps considerably. Dropout is a regularisation technique that disables connections in the network with a fixed probability. This approach is often helpful in dealing with large, highly correlated inputs by making the network less reliant on individual inputs and thus more robust to noise in individual inputs or missing values.

The top 10 performers of the hyperparameter search for the MLP are shown in Table 4.5. The MSE is given as mean plus or minus the standard deviation over the Time Series Group Split cross-validation, with a lower value indicating a better fit model. One of the top performers is a simple perceptron system (single layer of linear activation). This is equivalent to a linear function between inputs and outputs, which thus suggests that linear regression (LR) may also be effective.

The hyperparameter search for each model resulted in the following best parameters. Note if the parameter is not defined below then the default parameter for the model was used.

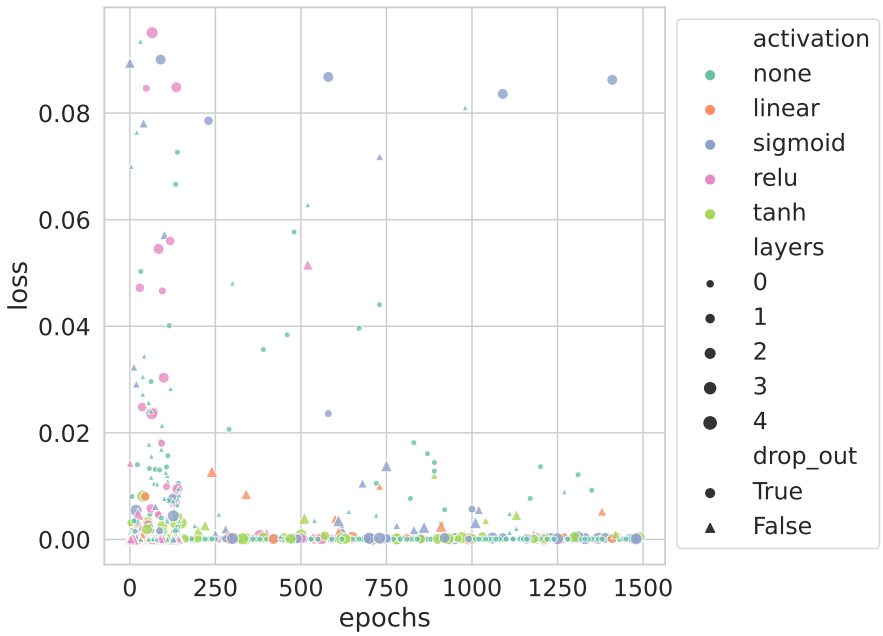

Figure 4.8: Overview of hyperparameter search showing the increase in epochs generally minimises the loss (MSE). For better visualisation, 8 data points are removed as their loss was between 0.1 and 0.7.

Table 4.5: The top 10 MLP models from the hyperparameter search results where Adamax is the optimizer.

Activation	batch size	dropout	epochs	layers	MSE
none	30	True	1480	0	0.000 049
none	41	True	1160	0	0.000 050
none	86	True	830	0	0.000 050
none	27	False	62	0	0.000 051
none	3	False	92	0	0.000 051
none	50	True	1190	0	0.000 052
none	112	True	620	0	0.000 053
relu	24	False	110	1	0.000 053
none	9	False	134	0	0.000 053
none	41	False	58	0	0.000 053

- LINEAR: no hyperparameters.
- LASSO: $\lambda = 0.000016$.
- RIDGE: $\lambda = 0.0014$ using the cholesky solver.
- ELASTIC NET: $\lambda = 0.000016$ and an α ratio of 0.99.
- KNN: Brute algorithm with leaf size of 13, 4 neighbors, and using the distance as weights.
- DECISION TREE: using the square root method for max features with a max depth of 6, max leaf nodes of 87, min samples leaf of 0.0102, min samples split of 0.0211, and min weight fraction leaf of 0.0111.
- RANDOM FOREST: using the square root method for max features with a max depth of 17, max leaf nodes of 94, min samples leaf of 0.0178, min samples split of 0.0353, min weight fraction leaf of 0.0249, and 16 estimators.
- EXTRA TREES: using max features equal to the number of features with a max depth of 33, max leaf nodes of 58, min samples leaf of 0.0106, min samples split of 0.000 12, min weight fraction leaf of 0.0106, and 27 estimators.
- XGBOOST: $\alpha = 0.0010$, $\gamma = 0.000046$, $\lambda = 0.000086$, 0.97 column sample by tree, learning rate of 0.2287, min child weight of 6.72, max depth of 83, and subsample of 0.8352.
- MLP: 0 hidden layers with a linear final activation, no drop out, using the Adamax optimizer, a batch size of 91, and 610 epochs.

4.5 COMPARISON OF MULTI-OUTPUT MODEL RESULTS FOR NEXT-STEP PREDICTIONS

The top performer is LR with a mean RMSE of 0.003 98 and a mean MAE of 0.002 59 for next step (10 s) predictions of min-max rescaled data, as shown in Table 4.6. The numbers in parentheses represent the standard deviation, for example the linear models RMSE 0.003 98(44) could also be rewritten as 0.00398 ± 0.00044. The RMSE and MAE scores are both close to zero, indicating that the LR model has high accuracy. The R^2 value indicates that 99.5 % of the variability observed in the response is explained by the LR model. The LR model has the lowest RMSE, although this is not significantly different from the results for ridge regression. Lasso and elastic net both provided a slightly higher RMSE of 0.004 84. The results for elastic net and lasso regression are similar because of the optimised hyperparameters. The elastic net and lasso regression lambda penalty parameters were both found to be 0.000 016, and the α parameter, which represents the L_1 ratio, for

Table 4.6: A comparison of the 10 multi-output ML models for the car cabin. The computation time is given as total seconds both train the model and make predictions. The metrics include root mean squared error (RMSE), mean absolute error (MAE) and the coefficient of determination (R^2), which are averaged across the last 3 folds of the 5-fold cross validation on the scaled data. The results are arranged by mean RMSE, with the standard deviation in parentheses.

Model	Time (s)	RMSE	MAE	R^2
Linear	0.1	0.003 98(44)	0.002 59(41)	0.995(2)
Ridge	0.0	0.003 99(42)	0.002 58(39)	0.995(1)
Lasso	76.0	0.004 84(49)	0.003 10(44)	0.996(1)
Elastic net	90.0	0.004 84(49)	0.003 10(44)	0.996(1)
MLP	120.0	0.0071(14)	0.0051(11)	0.993(2)
XGBoost	870.0	0.0192(35)	0.0114(20)	0.982(6)
KNN	0.2	0.027(11)	0.0178(68)	0.954(27)
Extra trees	0.8	0.0311(79)	0.0221(54)	0.937(38)
Decision tree	0.1	0.041(13)	0.0274(88)	0.815(135)
Random forest	0.5	0.063(15)	0.0418(82)	0.549(455)

elastic net was found to be 0.99, meaning the elastic net model is 99 % L_1 lasso penalty and 1 % L_2 ridge penalty. These hyperparameters result in virtually identical models for lasso and elastic net regression. The MLP is less accurate with a mean RMSE of 0.0071. The tree-based and neighbours-based models have significantly less accurate results than the linear, regularised, and MLP models.

Furthermore, the LR model is fast with a total computation time to train the model and make predictions of 0.1 seconds, the time to make predictions only being 0.0003 seconds of the total time. The ridge regression model is slightly quicker with a total computation time of 0.04 seconds. The XGBoost is the most computationally expensive with a total computation time of over 870 seconds. Note that the prediction error for several hours of predictions, that is, multi-step ahead predictions, is likely to be larger than the results shown here, which are for one-step predictions.

Despite the similar time and loss results, the LR is preferred over the ridge regression model as LR is a more robust model. Due to this and the fact that thermal systems are mainly linear, LR is chosen as the model for this data.

4.5.1 *Comparison of smoothed and unsmoothed data*

The previous analysis in this chapter was conducted using smoothed data, that is, a rolling mean with a window of 5 seconds. In order to investigate the effect that this smoothing had on the results, this section will examine the accuracy of the machine learning model on smoothed data against its unsmoothed counterpart.

Table 4.7: A comparison of smoothed and unsmoothed data for the LR multi-output model of the car cabin. The metrics include RMSE, MAE and the coefficient of determination (R^2), which are averaged across the last 3 folds of the 5-fold cross validation on the scaled data.

Smoothed?	Time (s)	RMSE	MAE	R^2
Yes	0.1	0.003 98(44)	0.002 59(41)	0.995(2)
No	0.2	0.015 70(67)	0.010 16(97)	0.876(13)

The results in Table 4.7 show that smoothing the data improves the predictive performance of the linear regression model. The smoothed model has an RMSE of 0.003 98 while the unsmoothed model has an RMSE of 0.0170, showing that the smoothed model has a significantly lower RMSE compared to the unsmoothed model. The smoothed model has smaller errors, on average, between the predicted and actual values, suggesting that smoothing the input data may increase accuracy and be able to better capture the underlying patterns in the data.

Unsmoothed data is able to provide fine-grained details, which can be especially useful when real-time responses are required, but can also amplify noise signals. On the other hand, smoothed data reduces noise, providing a more stable model and easier interpretability, however smoothed data may also lead to delayed responses to changes. The decision to use smoothed or unsmoothed data in the modelling process is a balance of trade-offs. Smoothed data is chosen in this thesis as it provides a more stable model for making both short and long-term predictions, with increased performance in terms of RMSE, MAE, and coefficient of determination.

4.5.2 *Comparison of various time intervals*

This chapter has examined the car cabin data for a 10 second time interval between data points. In this section, various time intervals, including 20, 30, and 60 seconds are tested and compared, as shown in Table 4.8. Due to the limited time frame available for the data, resampling at these intervals means the sample size of the data will be smaller for longer time intervals. In particular, recall that the 10 second interval contains 5270 samples, reducing the sample size to 2635 for 20 seconds, 1755 samples for 30 seconds, and 875 samples for 60 seconds.

4.6 LONG-RUN PREDICTIONS

Figure 4.9 shows a comparison of prediction output with measurement data for the AT at the driver's head position during CWT1 trial. The correspondence is remarkable since there is no divergence between two curves, that is, there is only a small error between them, even

Table 4.8: A comparison of the 10, 20, 30, and 60-second time intervals for the LR multi-output model of the car cabin. The metrics include RMSE, MAE and the coefficient of determination (R^2), which are averaged across the last 3 folds of the 5-fold cross validation on the scaled data.

Time interval (s)	RMSE	MAE	R^2
10	0.003 98(44)	0.002 59(41)	0.995(2)
20	0.006 86(94)	0.004 71(75)	0.983(5)
30	0.009 46(140)	0.006 73(106)	0.961(13)
60	0.014 11(183)	0.010 35(126)	0.893(47)

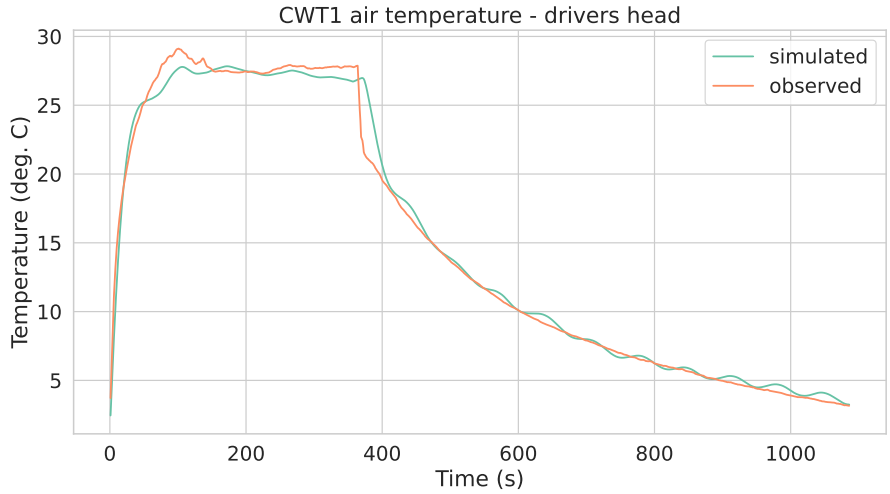

Figure 4.9: Comparison of prediction produced curve for the air temperature at the driver's head versus measurement data from CWT1 trial (without smoothing)

over the extended period of the test (around 3 h). Furthermore, the temperatures vary over a large range during the trial from almost 0 °C at the beginning to a peak of nearly 30 °C.

Other sensor modalities are reproduced with similar accuracy. Again, this is striking as the RH varies over a wide range during the trial. The model manages to track it almost perfectly just on the basis of the initial state and the control inputs.

The AV measurements tend to vary considerably. These measurements are smoothed during processing and thus the model produces a smooth estimate of the AV. The correspondence here is quite consistent through the whole period.

4.6.1 *Differences between head, torso, and foot temperatures*

A key benefit of the ML-based predictions is the ability to differently estimate different parts of the cabin space. For example, the tempera-

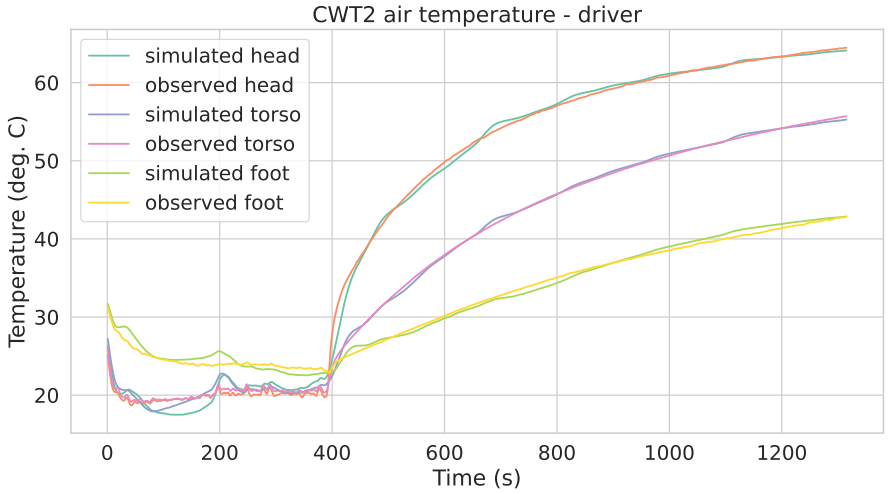

Figure 4.10: The ML model based on LR correctly and independently tracks driver's head, chest, and foot temperatures over a 3 h trial (CWT2) with only small errors

ture at the head may be much hotter than the footwell, which can have an effect on a passengers thermal comfort. Therefore, it is interesting to see whether the model is able to independently and differently track the AT (for example) at the head, torso and foot. Figure 4.10 and Figure 4.11 demonstrate that the predictions are very close tho the actual values.

In Figure 4.10, for example, the AT in the footwell is higher than at the head and torso locations during the first part of the trial. During the long term progression, where the HVAC was turned off after approximately 400 seconds, temperatures for all three locations (head, torso, and foot) differed and were closely predicted by the ML model. The temperature predictions in the footwell are not quite as accurate as the predictions for the head and torso during the beginning of the trial (0 s to 4000 s). A reason for this may be that the model has not captured some characteristics of the footwell, such as the fact that the area is more enclosed and therefore may be more insulated. This may not be detrimental as a perfect prediction may not be needed [HS18].

4.6.2 *Examples showing room for improvement*

Figure 4.12 shows some response differences for AT at the driver's head in the predictions compared to the data from CWT3 trial. Note that the vertical range is small and this may appear to magnify errors. A striking aspect of this result is that the final temperature converged upon is very close to the true final temperature.

Another example is the windshield temperature predictions for CWT3 trial, shown in Figure 4.13. Here the temperatures are quite

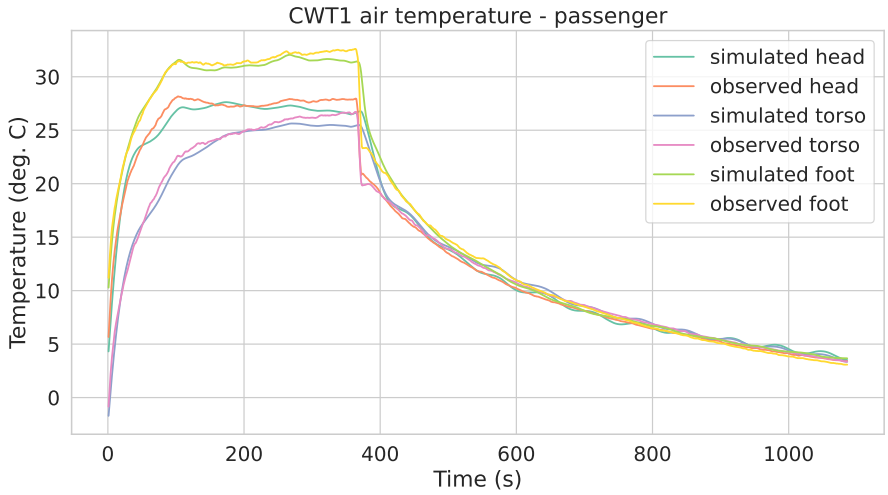

Figure 4.11: Passenger side air temperatures are reasonably accurately tracked with clear differences during the early phase between head, torso, and foot for CWT1 trial

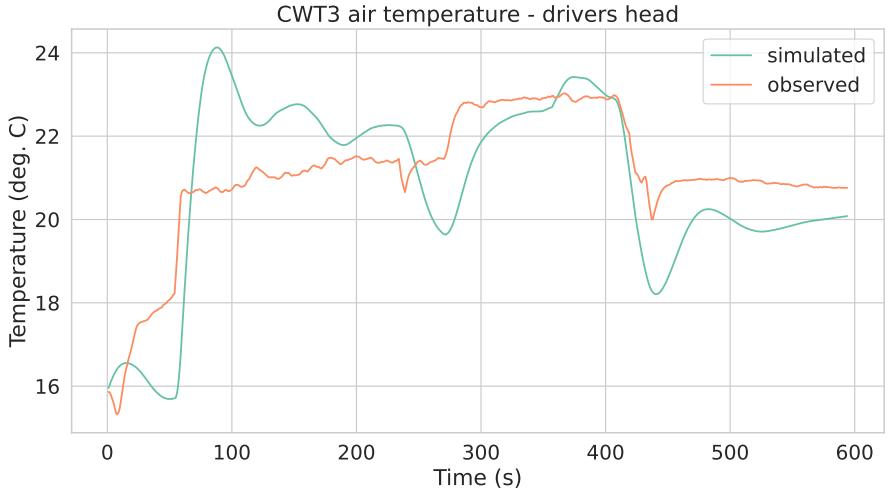

Figure 4.12: Comparison of measurement and simulated data for CWT3 trial's air temperature at the driver's head

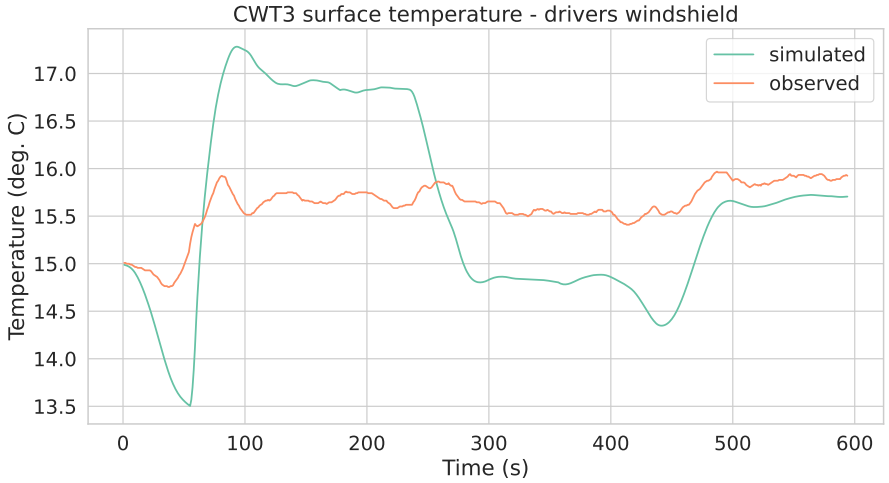

Figure 4.13: Comparison of measurement and simulated data for CWT3 trial's windshield temperature driver's side

stable during the CWT trial but vary considerably in the predictions. Again, the vertical range is small but the error is up to 3 K.

For Figure 4.12 and Figure 4.13, recall that CWT3 is the only trial with moisture added into the cabin. The average normalized root mean squared error (NRMSE) across all sensors for the full trial duration is 0.16 (or 16%) for CWT3, whereas the other four CWT trials are in the range of 0.041 (4.1%) to 0.082 (8.2%). It is possible that providing more CWT trial data where moisture is added into the cabin to train the model on may help to improve these predictions.

4.6.3 *NRMSE results*

Rather than trying to understand the accuracy of the predictions based on examining individual graphs, it is generally more appropriate to summarise the error in terms of the RMSE or NRMSE. Table 4.9 shows results obtained by producing predictions through the entire duration of each of the CWT trials using the LR model. The RMSE and NRMSE shown here is the mean alongside the standard deviation over the 5 trials. Note that NRMSE is shown as a proportion rather than a percentage. For example, an NRMSE of 0.003 corresponds to 0.3 %. The RMSE and NRMSE values here are for the full trials (around 3 h) and thus will be somewhat larger than the 10 second prediction RMSE or corresponding MSE used during training. The units for the RMSE depend on the sensor, as specified in Table 4.3. Note that the displayed decimal places are adjusted according to the standard deviation.

The average NRMSE performance over all sensors over the full trial duration is 0.011 (or 1.1 %), which is comparable but slightly larger than the 10 second NRMSE (0.8 %). The average AT RMSE is 0.25 °C or 0.5 %.

Table 4.9: Performance of ML model in terms of error for each sensor, including surface temperature (ST), mean radiant temperature (MRT), air temperature (AT), relative humidity (RH), and velocity (V). The table is arranged by mean NRMSE with the standard deviation in parentheses. The average air temperature (AT) is based on comparing the average of head, torso and foot air temperatures for driver and front passenger with that of the predicted values.

Sensor	RMSE	NRMSE
ST - drivers windshield	0.19(7)	0.003(1)
MRT - passengers torso	0.26(9)	0.004(1)
MRT - passengers foot	0.20(7)	0.004(1)
MRT - drivers head	0.3(1)	0.004(2)
AT - drivers torso	0.25(8)	0.004(1)
MRT - drivers torso	0.28(9)	0.005(1)
MRT - passengers head	0.3(1)	0.005(2)
average AT	0.25(9)	0.005(2)
MRT - drivers foot	0.24(5)	0.0049(9)
AT - passengers torso	0.28(10)	0.005(2)
AT - drivers head	0.3(1)	0.006(2)
AT - passengers head	0.4(1)	0.006(2)
AT - passengers foot	0.27(6)	0.007(1)
AT - drivers foot	0.28(7)	0.007(2)
RH - cabin	0.5(3)	0.012(7)
AV - drivers head	0.012(9)	0.02(2)
AV - drivers torso	0.011(3)	0.022(5)
AV - passengers torso	0.010(1)	0.023(3)
AV - drivers foot	0.011(3)	0.024(6)
AV - passengers head	0.014(3)	0.025(5)
AV - passengers foot	0.016(3)	0.025(4)

4.6.4 *Compute performance*

Calculating a single time step (10 s) with the LR model developed in this thesis is extremely fast. The reason for this fast performance is that the calculation can be performed with a single matrix multiplication. Also, LR, unlike the neural network, does not require rescaling of inputs and outputs. On a PC with Intel(R) Core(TM) i7-4790 CPU, 3.60GHz processor, 1000×3 hour predictions were calculated in 76.5 seconds. This corresponds to $0.007\,\mathrm{ms\,s^{-1}}$ (elapsed time per predicted second).

Compute performance becomes critical when attempting to use machine learning to optimise a control algorithm. In past work, around 9 years of simulated time was required to find the optimal control strategy. The time to compute 9 years of simulated time using this proposed LR approach is around 33 minutes.

4.6.5 *Discussion*

These results show high accuracy for the long-run predictions using the LR model over all the sensors. The smallest errors are less than the expected error in the thermocouple sensor while the largest errors (around $0.04\,\mathrm{m\,s^{-1}}$ or 4 % of the range) are within 5 % of the accuracy of the high-level model. The results reflect that short-term errors tend to disappear over time. This is surprising because, in many predictions, small errors accumulate when producing a prediction that runs over an extended period. This reflects the relative simplicity of the model causing it to be extremely robust. Possible threats to validity of the LR prediction results are as follows:

- These results are specific to the range of parameters varied during the CWT trials. For example, only 2 distribution modes were switched between during the experiments, therefore it would not be possible to use this model to make predictions for other distribution modes.
- If new components, such as radiant panels, are added, this may impact the thermal dynamics of the car cabin, therefore the current model would not support predictions of an environment with such features. In order to provide predictions for this environment, real-world or computational fluid dynamics (CFD) based data is required. Given a sufficient amount of training data, this approach could then be applied to environments with these additions.
- A better estimate of the performance on unseen data might be possible using k-fold cross-validation. Since the full data-set was used to both learn the linear regression coefficients and to assess performance, the true performance on unseen data

may be slightly worse than estimated here. Note, however, that overfitting is unlikely for this method.

When applying the proposed ML thermal model to a real car environment, distinct challenges arise compared to the controlled CWT environment. External factors, such as sunlight, wind, and ambient temperature changes, have dynamic effects on the car cabin's thermal conditions in real-world scenarios. Dealing with uncertainties and variations becomes crucial since real-world conditions are less predictable than controlled settings. A robust model is essential to handle unforeseen events that might affect prediction accuracy. Additionally, occupant behavior, like opening windows or adjusting HVAC controls, significantly impacts the car cabin's thermal state. To account for these human-driven factors, the model needs to accommodate such actions. Understanding and addressing these challenges will determine how effectively the ML thermal model predicts thermal variables in real-world car cabin conditions.

4.7 FEATURE IMPORTANCE

Feature importance is performed using the regularization method of elastic net to examine how each input feature contributes to the model predictions. For models based on linear regression, as elastic net is, feature importance can be interpreted through the coefficients of the model. Elastic net regression allows for coefficients to be shrunk to zero, therefore, this can act as a feature selection method. Note that the coefficients were found using standardized data. Due to the multi-output structure of the model, 39 coefficients are given for each of the 20 outputs. Therefore, an average is calculated for each feature, resulting in 39 feature importance scores. Here it is chosen to visualise the elastic net results only as the lasso results are very similar, with the elastic net ratio heavily favouring the lasso penalty (0.99). In Figure 4.14, only one variable is shrunk all the way to 0, that is the MRT at the passenger's torso.

The feature importance can also be filtered based on variables of interest. Figure 4.15 shows the feature importances for predicting the AT at the drivers head using elastic net regression. As expected, the most important feature for predicting the current AT at the drivers head is the previous AT at the same position. The next most important values include the AT at the passengers head and drivers torso, as well as the MRT at the drivers head and passengers foot. Although the importance for many of the variables is very small, the only value that is shrunk all the way to zero is for the MRT at the passengers torso.

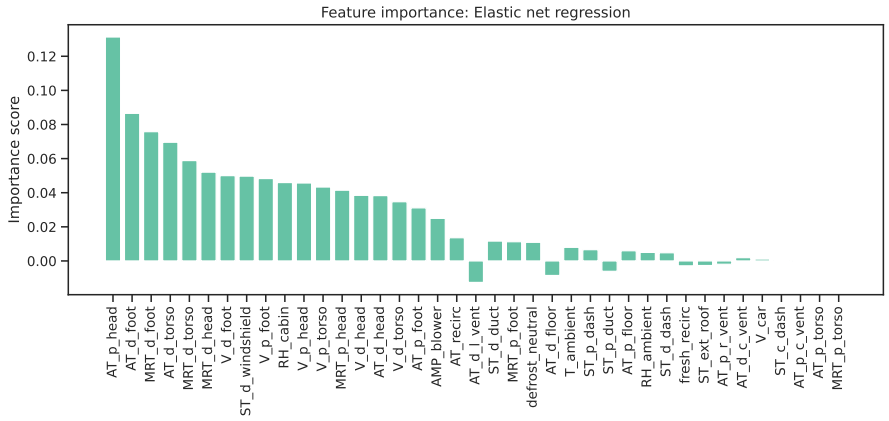

Figure 4.14: Feature importance for elastic net model on the car cabin data across the predictions of all output variables.

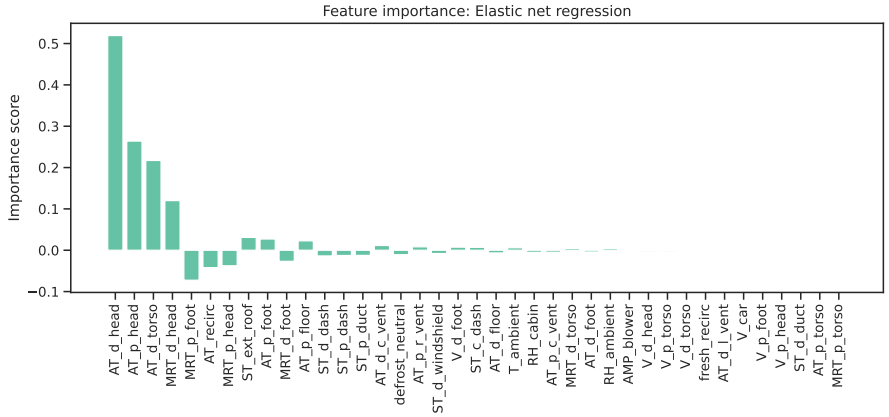

Figure 4.15: Feature importance for elastic net regression model on the car cabin data for predicting only air temperature at the drivers head.

4.8 COMPARATIVE PERFORMANCE OF PHYSICS AND MACHINE LEARNING-BASED MODELS

To allow for a comparison of the ML model results, the experimental results from a physics-based simulation model have been provided by experts in the field via the DOMUS project [BR20]. This section will explain the basics of the physics-based model that was implemented in order to obtain the results, making comparisons to the LR model in terms of accuracy, speed, and capability.

4.8.1 *Physics-based model*

The physics-based model uses AMESim version 17 software [Sie18] to model the North America Fiat 500e BEV climate system. AMESim is a programming environment developed for the object-oriented modelling of complex physical systems. The libraries that were used to simulate the thermal aspects of the car cabin include: thermal, thermo-hydraulic, two-phase flow, heat, and air-conditioning.

The model of the car cabin climate system includes three main sections that are connected together, that is:

1. the HVAC system, including an evaporator, air ducts, and positive temperature coefficient (PTC) heater;
2. a two-phase flow loop, including a compressor, heat exchangers, and thermal expansion valves;
3. two coolant loops, which provide battery and power train thermal management.

Geometry and material data is required for the AMESim model, which was provided by the data sheet for each component, such as the evaporator, to ensure that the virtual components are identical to the physical ones. Furthermore, some of the parameters were calibrated using information on the data sheets, such as the heat exchange coefficient, which was derived from the Nusselt number, similar to the work done by Ciacci [Cia19].

Two separate models are produced for the different air modes: one is tri-level for heating and the other is vent only for cooling. For each of these models, the heat exchange coefficient is calibrated (that is, for warm-up and cool-down). No pressure drop is considered from the HVAC to the car cabin and the air flow rate is variable depending on the adopted strategy.

In summary, the DOMUS project provided results from the physics-based models that include average air temperature in the cabin and simulation compute time results [BR20].

A 1D physics-based model may have limitations in capturing localized temperature variations compared to a CFD model, which offers a

more detailed discretized representation of the air within a car cabin. However, to achieve faster simulation results and provide a simplified representation of the car cabin's thermal behavior, a 1D model was chosen over CFD. This choice also reduces the data and assumptions required for modelling, aligning with the approach used in the ML thermal model to achieve efficiency and practicality in capturing the thermal environment.

4.8.2 *Accuracy*

The average NRMSE over all sensors being estimated for the LR-based model is 1.1 %. The best estimated sensor (ST at drivers' windshield) has an NRMSE of 0.3 % while the worst (AV at passenger's foot) has a 2.5 % NRMSE. The error of the estimate of the average AT over the front bench of the car cabin is 0.25 °C (0.5 %).

The physics-based model NRMSE is 1.5 % for the average cabin air temperature. When the air conditioning (AC) loop is on, the model performs about 3 % for the high pressure and 6 % for the low pressure.

4.8.3 *Computational speed*

To compare the computational speed, the amount of elapsed time to compute 1 second of predicted time is calculated. The results for the physics-based models are:

- $7.6\,\mathrm{ms\,s^{-1}}$ in warm up protocol.
- $250\,\mathrm{ms\,s^{-1}}$ in cool down protocol.

The result for the ML model is:

- $0.007\,\mathrm{ms\,s^{-1}}$.

Note that the ML-based model does not attempt to make predictions for the cooling loop, but is able to model different parts of the car cabin (driver and passenger's head, torso and foot).

Based on the above results, the speed-up for the LR-based model compared with the physics-based model (during warm up) is 1400-fold.

4.8.4 *Capability*

The two models have different sets of capabilities and this should be taken into account when considering other performance aspects.

The LR-based model has certain capabilities that are not available in the physics-based model:

- The ability to provide properties needed to make use of a holistic comfort model for both front bench occupants. Specifically, the

ML model estimates temperature, MRT, and AV at the head, torso, and foot positions for both occupants.

- The ML model is able to estimate windshield glass temperature and RH within the cabin, which allows for the estimation of safety in terms of windshield fogging.

The physics-based model, on the other hand, has capabilities not available in the LR-based model:

- It simulates the HVAC system more fully, including the AC loop, rather than requiring the air vent temperature as input. Note that the ML model does simulate the blower.
- It is a physics-based approach and thus is likely to generalise more readily to circumstances not seen in the CWT trials.
- It supports additional components, such as the radiant panels.

Due to these differences in capabilities, the choice of model will depend on the application.

4.9 CHAPTER SUMMARY

The key results are:

1. The ML cabin model has an average NRMSE over all sensors of 1.1 %. The RMSE for the average AT for the front two occupants is 0.25 °C (0.5 %) over all trials.
2. The ML cabin model computes a second of predicted time in 0.007 ms.

The chosen ML model is based on LR and gives an average RMSE of 0.003 98 for next step predictions. The LR model is able to closely track AT, RH, and AV dynamics over multi-step predictions within the cabin clearly demonstrating the viability of the method. This model provides a solid basis for work where it is not necessary to add components, such as radiant panels. Furthermore, this work is remarkable in that it provides a model that is capable of accurately simulating the thermal dynamics at multiple car seating positions and to do so with a compute performance that is much faster than traditional physics-based approaches. This opens the way for numerical optimisation approaches that were previously considered infeasible to build car cabin HVAC controllers and redesign the car cabin features. Future work is required to provide data that can enable the prediction of optional components including radiant panels, heated seats, and special glazing.

In order to make use of the ML-based model, additional work is needed, as follows:

1. A separate model of the HVAC system is needed to provide vent outlet temperatures.

2. To properly simulate components, such as the radiant panels, further experimental data is needed with this added. This data might be produced based on the CFD simulation, for example.

3. More information could also be included in the model such as contact heat (that is, heated seats) and the accumulation of CO_2 in the cabin [Ang+19].

Minimising unnecessary energy consumption is central to the design of modern electric vehicles, with the heating and cooling system is the largest auxiliary load. However, personal comfort depends on this HVAC system and is critical to customer satisfaction, while some of this functionality is also needed for safety (for example, defogging the windscreen). Therefore, it is important to minimise energy use under the constraint of maintaining acceptable comfort and safety. The methods presented in this paper model the thermal environment within a car cabin to help identify whether comfort and safety requirements can be met and at what energy cost.

The key comparisons between the physics- and LR-based model results are:

1. The physics-based cabin model predicts the cabin average AT within ± 1 K (1.52 %).

2. The physics-based cabin model predicts AC pressure with an average error less than 0.6 bar (3 %) at high pressure and 0.1 bar (6 %) (steady state) at low pressure. The physics-based cabin model computes a second of predicted time in 0.25 s (worst case - when the AC compressor is on) or 0.0076 s (AC compressor off).

The LR-based model is sufficiently fast and accurate to suggest that this is a promising method. The physics-based model, being physics-based, may still be preferred for some applications.

5

THERMAL MACHINE LEARNING MODEL FOR A HOUSE

5.1 CHAPTER INTRODUCTION

In this chapter, a second case study is pursued to predict the thermal aspects of houses. In contrast to the car cabin case study, this case study includes data collected from various rooms within different houses in Loughborough, UK, resulting in a model being built for each individual house. The aim of this case study is to investigate whether the machine learning (ML) models used in Chapter 4 could be generalisable to other thermal environments, and, in addition, allows the approaches to be tested on a larger, real-world dataset.

Overall this chapter will include the following:

1. An introduction to the REFIT dataset (Section 5.2), as well as a description of the preprocessing steps (Section 5.3);
2. The results of the hyperparameter search for the tested multi-output ML models (Section 5.4);
3. A comparison of several multi-output ML approaches, including the results of the chosen model for both one-step and longer term performance (Section 5.5 and Section 5.6 respectively);
4. An analysis of the most important features for the selected ML model (Section 5.7);
5. A comparison of the selected ML model for the house to the selected model for the car cabin, including a comparison of the most important features (Section 5.8).

5.2 REFIT SMART HOME DATA

The REFIT Smart Home data [Fir+17] is a publicly available dataset from researchers at Loughborough University, the University of East Anglia, and University of Strathclyde. Contained in the REFIT dataset are 20 houses within the East Midlands region of the UK, near Loughborough [Fir+15]. For each house, a building survey was performed, which involved the collection of data regarding the houses' building geometry, construction materials, occupancy, and energy services. In addition, multiple sensors and devices were installed to monitor total energy usage, gas consumption, and multiple temporal variables, such as temperature, humidity, light level, motion, and individual appliance energy usage. Alongside this data, which was collected from

September 2013 to April 2015, the climate data during this period was also available from a local weather station.

From the data collected, the following variables collected at each house that are of interest to this research:

- Gas consumption for the whole house,
- Air temperature (AT) for various rooms or spaces in the house,
- Relative humidity (RH) for various spaces in the house,
- Radiator valve surface temperature (ST) for various spaces in the house,
- Outdoor AT,
- Outdoor RH,
- Outdoor wind speed,
- Outdoor total rainfall.

For AT, RH, and radiator ST in each house, the number of sensors varies depending on the size of the house and the data available. For this analysis two houses were excluded: house 2 due to a large portion of missing data and house 5 as gas consumption data was not available. For the sake of this thesis, the houses are then renumbered where house 3 becomes house 2, house 6 becomes house 4, etc.

From the remaining 18 houses, sixteen of the single family dwellings were detached and two were semi-detached. The houses were built during various periods: 1850-99 (1), 1919-44 (2), 1945-64 (2), 1965-74 (6), 1975-80 (1), 1981-90 (3), 1991-95 (1), or after 2002 (2). Most dwellings were two-stories (16) while only two were three-stories. In terms of the number of rooms, these included two (1), three (12), four (3), and five (2) bedrooms with either one (11), two (4), or three (3) bathrooms. Note that the houses used in this case study are mainly detached dwellings, which is not representative of the UK housing stock, and because these dwelling types have the most exposed walls, they may also lose more heat than other dwelling types. Other factors, such as the time period homes were built, could also have an effect on heat transfer within the homes as the standards for building materials has varied throughout the years.

5.2.1 *State and control vectors*

For the analysis, all variables must be classified into either the control $\mathbf{u}$ or state $\mathbf{x}$ vectors, as defined in Section 3.2.2. The selected control vector $\mathbf{u}$ and state vector $\mathbf{x}$ are shown in Table 5.1 and Table 5.2 respectively. The number of AT, RH, and ST variable available depend on the house and are therefore represented by the lengths a, r, and s,

Control	Description	Units
u_1	outdoor air temperature	°C
u_2	outdoor relative humidity	%
u_3	outdoor wind speed	$m\,s^{-1}$
u_4	outdoor total rainfall	mm
u_5	gas consumption	m^3
u_{5+1}	radiator surface temperature	°C
u_{5+2}	radiator surface temperature	°C
⋮	⋮	⋮
u_{5+s}	radiator surface temperature	°C

Table 5.1: Measurement variables that comprise the control vector $\mathbf{u}$, where s is the number of radiator surface temperature sensors available and the vector contains one gas consumption reading and four weather variables.

State	Description	Units
x_1	room air temperature	°C
x_2	room air temperature	°C
⋮	⋮	⋮
x_a	room air temperature	°C
x_{a+1}	room relative humidity	%
⋮	⋮	⋮
x_{a+r}	room relative humidity	%

Table 5.2: Measurement variables comprising the state $\mathbf{x}$, where a is the number of air temperature sensors and r is the number of relative humidity sensors available.

respectively. For the house, the aim to keep the variables that are most similar to those used in the car cabin model in Chapter 4.

Table 5.3 details the final count of the state $\mathbf{x}$ and control $\mathbf{u}$ variables for each of the 18 houses. The number of rooms with air temperature, relative humidity, and surface temperature sensor readings vary by house. Furthermore, some rooms where the air temperature is measured are either heated, but the radiator surface temperature is not measured, or are simply unheated (for example, a garage). Rather than using only the rooms where all three sensor readings are available, it is chosen that all available data will be used in the models.

House	$\mathbf{x}$	x_a	x_r	$\mathbf{u}$	u_s
0	15	11	4	13	8
1	10	7	3	12	7
2	12	8	4	16	11
3	15	11	4	21	16
4	16	12	4	20	15
5	12	9	3	14	9
6	10	7	3	17	12
7	11	8	3	13	8
8	14	10	4	15	10
9	8	6	2	15	10
10	15	10	5	18	13
11	12	8	4	12	7
12	15	11	4	21	16
13	13	9	4	15	10
14	10	7	3	16	11
15	11	9	2	14	9
16	12	9	3	14	9
17	12	8	4	12	7

Table 5.3: The size of the state vector $\mathbf{x}$ and control vector $\mathbf{u}$ for each house. The state vector is made up of a air temperatures (x_a) and r relative humidities (x_r), while the control is made up of s radiator surface temperatures (u_s), four climate variables (u_{1-4}), and one total gas consumption (u_5) for the house. For example, house 0 has 15 state variables comprised of 11 air temperatures and 4 relative humidities, and 13 control variables comprised of 8 surface temperatures, as well as the 5 variables consistent across each house (u_{1-5}).

5.3 DATA PRE-PROCESSING

This section details the pre-processing steps taken for the house data, as introduced in Section 3.3, after defining the state and control vectors. The REFIT data comes from houses within the community, and due to the nature of sensors (for example, a loss of connection or hardware damage), there are various missing data points that need to be addressed. First, a calculation is done to check how many data points are missing per house. To prepare the data for analysis, the data is then resampled to every 30 minutes using the mean value. Some of the sensors needed to be replaced after a period of time, in order to account for this, the sensors were grouped by room and measurement type (such as, air temperature or relative humidity) and the average reading was taken. This resulted in, for example, only one air temperature reading per room. The next step is to scale and transform the data, as described in Section 3.3.4, yielding a normalised form of the data.

Once the data is normalised, the variables of interest are extracted from the data. The aim is to build a simple model for each house based on easily measurable data. Therefore, the main variables of interest include room ATs, room RHs, radiator STs, and total gas consumption for each house. These variables are extracted and joined with the climate data, specifically the outdoor AT, outdoor RH, wind speed, and total rainfall. These can be divided into state and control vectors, as defined in Table 5.1 and Table 5.2. The AT and RH readings from various rooms comprise the state vector $\mathbf{x}$, while the climate, radiator STs, and gas consumption comprise the control $\mathbf{u}$. The next step in pre-processing is to examine the autocorrelation and partial autocorrelation functions to determine how many lags are appropriate for the state variables, as described in Section 2.7.1 and represented in Figure 3.1. A representative example of how the autocorrelation plots look is selected and shown in Figure 5.1. The figure shows the autocorrelation and partial autocorrelation functions for the air temperature in space 287 in house 15, which indicates a strong partial correlation at both 1 and 2 lags. Therefore, the data is converted to a form which has a lag of 2 for the state vector.

Table 5.4 shows the final dimensions of the data for each house after pre-processing, including combining the house data with climate data and introducing the lag. In total, across all 18 houses used for analysis there are 267 761 rows of data. The number of columns for each house depends on multiple factors, including the size of the house, the number of sensors actually installed, and the availability of the sensor data, as explained in Table 5.3. The number of rows per house ranges from 5036 to 20845, again depending on the sensor data available for the house.

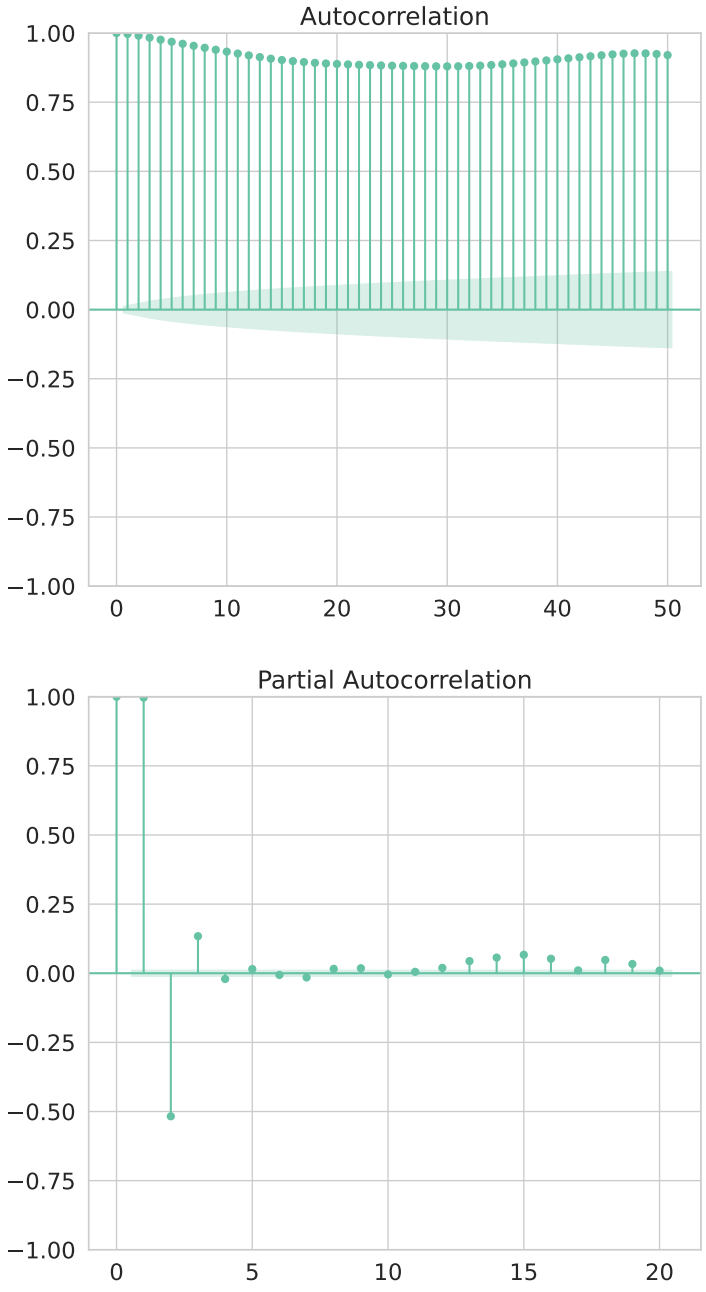

Figure 5.1: Autocorrelation and partial autocorrelation function plots for the air temperature in space 287 within house 15, which is representative of the other spaces.

House ID	No. rows	No. columns
0	17843	43
1	11148	32
2	5036	40
3	13570	51
4	12952	52
5	18684	38
6	16236	37
7	20845	35
8	12643	43
9	15988	31
10	14364	48
11	14465	36
12	8190	51
13	14371	41
14	21950	36
15	15604	36
16	20776	38
17	13096	36

Table 5.4: The dimensions of the pre-processed house data for the final 18 houses used for analysis, after being combined with the relevant weather data.

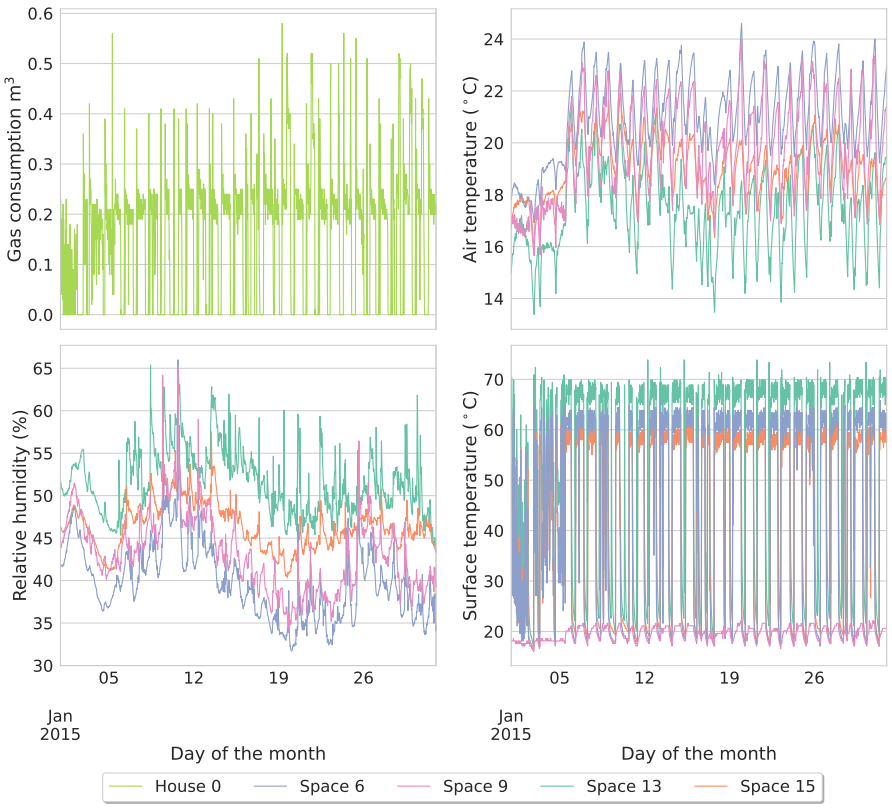

Figure 5.2: Visualisation of data from house 0 in January 2015. The figure on the top left shows the gas consumption readings. The top right figure shows air temperature readings, bottom left shows relative humidity, and bottom right shows radiator surface temperature, which all show four different spaces within the house, namely the kitchen (space 6), living room (space 9), bedroom (space 13), and en suite bathroom (space 15). See Figure 5.3 for a zoomed view.

Figure 5.2 shows a sample of the data from house 0. The plot shows January 2015, which saw an average outdoor temperature of 5.0 °C, ranging from −3.6 °C to 15.9 °C. Therefore, it is likely that the heating would have been utilized during this month. The sensors readings show four rooms, which are the only rooms in the house where AT, RH, and radiator ST were all collected. Figure 5.2 shows a difference in the sensor readings by room, for example the bedroom (space 13) appears to be cooler and more humid than the other rooms. A baseline gas usage may be contributed from cooking using a gas hob if it is available in the house, however, the main contribution to an increase gas consumption is for heating purposes. Therefore, any upward spike for gas consumption can be seen as an indication that the house has the heating turned on. Similarly, an increase in radiator ST demonstrates when the heating is turned on for that specific radiator in a room.

For the living room (space 9) radiator ST, Figure 5.3 also shows that the radiator does not have much variation in the ST, signalling that either the radiator does not get warm or is not turned on in this room.

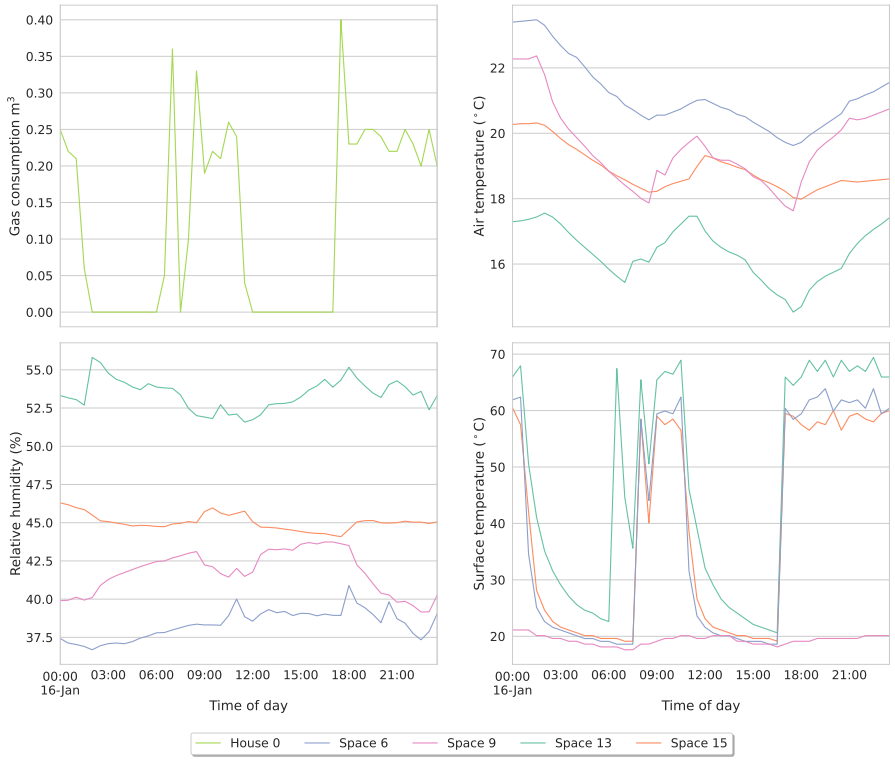

Figure 5.3: Visualisation of data from house 0 on 16 January 2015. The figure on the top left shows the gas consumption readings. The top right figure shows air temperature readings, bottom left shows relative humidity, and bottom right shows radiator surface temperature, which all show four different spaces within the house, namely the kitchen (space 6), living room (space 9), bedroom (space 13), and en suite bathroom (space 15).

Examining the corresponding ATs shows that the temperature in space 9 is quite high compared to the rooms where the radiator ST is warmer. This could be due to the home having a secondary heating source, such as an electric heater or fireplace. The bedroom (space 13) exhibits the highest radiator ST, highest humidity, and lowest air temperature. This could be due to a variety of reasons, such as damp clothes being placed on the radiator to dry, causing an increase in humidity and a lower air temperature.

Once we have pre-processed the house data, a hyperparameter search for the selected ML models can be conducted, as described in Section 3.5. For the house data, rather than considering each house to be a group, a separate model will be built for each of the 20 homes because the underlying structures would vary considerably across houses. As such, for each house the Time Series Split cross validator is used, as introduced in Section 2.8.2.

5.4 HYPERPARAMETER SEARCH RESULTS

For the house case study, a separate model was built for each of the 18 houses. A hyperparameter search is, therefore, done for each house with the proposed ML models, following the procedure from Section 3.5. Due to the computational demand of the XGBoost and multilayer perceptron (MLP) models, the hyperparameter searches were done on a subsample of the data using every fourth row of data. The remaining models used the full data for the hyperparameter searches. A summary of the best parameters for the 18 models are presented here. Note if the parameter is not defined below then the default parameter for the model was used.

- LINEAR: no hyperparameters.
- LASSO: λ was found to be small, with a minimum of 3.3×10^{-22} and maximum of 7.4×10^{-5}. The mean value is 2.6×10^{-5} and standard deviation 2.4×10^{-5}.
- RIDGE: λ had a minimum value of 1.6×10^{-34} and maximum of 0.34. The mean λ was 0.042 with standard deviation 0.10. The Cholesky solver was used for 9 models, lsqr for 4, saga for 3, and sag for 2.
- ELASTIC NET: λ minimum 1.0×10^{-21}, maximum 7.5×10^{-5}, mean 2.6×10^{-5} standard deviation 2.3×10^{-5}. The α ratio was above 0.5 for 15 models, with 13 of these having a value over 0.90 and a further 11 models having a value of over 0.99. The mean α ratio is 0.82 with a standard deviation of 0.31.
- KNN: every model was found to use using distances as weights. The brute algorithm was used for 8 models, kd tree for 8 and ball tree for 2. The leaf sizes were in the range of 1 to 48 with an average leaf size of 21 and standard deviation of 16, while the number of neighbors was in the range of 4 to 25 with an average leaf size of 13 and standard deviation of 6.
- DECISION TREE: the majority of the models (16) used the number of total features for the number of features considered for the best split, while 2 use the log2 method. The maximum depth of the tree had values in the range of 8 to 49 with an average max depth value of 29 and standard deviation 14. The maximum leaf nodes had values from 3 to 94 with an average of 69 and standard deviation of 20. The minimum number of samples required at a leaf node ranged from 0.010 to 0.212 with mean 0.033 and standard deviation 0.060. The minimum number samples required to split an internal node were between 4.6×10^{-5} and 0.89 with a mean 0.63 and standard deviation 0.20. Lastly, the minimum weighted fraction of the sum of all inputs samples

requires at the leaf node had values between 0.010 and 0.490 with a mean 0.038 and standard deviation 0.110.

- RANDOM FOREST: the number of features considered for the best split was done using the total number of features for 3 models, the square root for 10 models, and the log2 for 4 models. The maximum depth of the tree had values in the range of 8 to 48 with an average max depth value of 31 and standard deviation 12. The maximum leaf nodes had values from 42 to 95 with an average of 71 and standard deviation of 15. The minimum number of samples required at a leaf node ranged from 0.010 to 0.011 with mean 0.010 and standard deviation 2.3×10^{-4}. The minimum number samples required to split an internal node were between 7.2×10^{-5} and 0.23 with a mean 0.011 and standard deviation 0.009. Lastly, the minimum weighted fraction of the sum of all inputs samples requires at the leaf node had values between 0.010 and 0.020 with a mean 0.014 and standard deviation 0.004.

- EXTRA TREES: for the number of features considered for the best split, 16 models used the total number of features and 2 used the square root. The maximum depth of the tree had values in the range of 8 to 49 with an average max depth value of 31 and standard deviation 13. The maximum leaf nodes had values from 39 to 93 with an average of 71 and standard deviation of 15. The minimum number of samples required at a leaf node ranged from 0.010 to 0.019 with mean 0.011 and standard deviation 0.0027. The minimum number samples required to split an internal node were between 9.2×10^{-5} and 0.039 with a mean 0.015 and standard deviation 0.012. Lastly, the minimum weighted fraction of the sum of all inputs samples requires at the leaf node had values between 0.010 and 0.018 with a mean 0.011 and standard deviation 0.003.

- XGBOOST: alpha ranges from 4.6×10^{-5} to 0.28 with mean 0.018 and standard deviation 0.066, gamma ranges from 4.6×10^{-4} to 0.0015 with mean 2.8×10^{-4} and standard deviation 3.6×10^{-4}, lambda ranges from 7.9×10^{-5} to 68 with mean 5.5 and standard deviation 16, column sample by tree ranges from 0.71 to 1 with mean 0.93 and standard deviation 0.085, learning rate ranges from 0.047 to 0.30 with mean 0.11 and standard deviation 0.072, min child weight ranges from 0.97 to 20 with mean 8.6 and standard deviation 6.7, max depth ranges from 1 to 98 with mean 45 and standard deviation 34, and subsample ranges from 0.50 to 0.93 with mean 0.72 and standard deviation 0.13.

- MLP: 4 models had 0 hidden layers, 8 had 1 layer, 4 had 2, 1 had 3, and 1 had 4. 3 models used the Adam optimizer, while the

Table 5.5: Mean and standard deviation of the computation time in seconds to train the model, RMSE, MAE, and coefficient of determination (R^2) of different multi-output ML models across all 18 houses for the last 3 folds of the 5-fold cross validation on the scaled data. The values are arranged by mean RMSE.

Model	Time (s)	RMSE	MAE	R^2
Ridge	11(19)	0.017 11(440)	0.010 31(288)	0.948 87(2601)
Lasso	21(8)	0.017 47(427)	0.010 71(285)	0.949 59(2468)
Elastic net	21(8)	0.017 48(425)	0.010 71(284)	0.949 54(2468)
Linear	0.13(6)	0.017 72(467)	0.010 28(286)	0.944 03(3492)
MLP	300(410)	0.021 16(641)	0.013 89(496)	0.924 27(6502)
XGBoost	100(97)	0.024 44(877)	0.015 10(533)	0.907 27(5576)
KNN	2.2(14)	0.057 70(1231)	0.043 92(983)	0.562 55(17896)
Extra trees	3.6(17)	0.069 09(1940)	0.054 15(1637)	0.318 47(40155)
Random forest	4.6(40)	0.071 05(1844)	0.055 69(1530)	0.293 84(36027)
Decision tree	0.67(36)	0.082 57(2276)	0.065 12(1999)	0.055 32(74937)

remaining all utilised the Adamax optimizer, 2 of these having 0 layers and 1 having 1. Of those models with hidden layers, 5 used a linear activation, 5 used tanh, and 4 used ReLU.

5.5 COMPARISON OF MULTI-OUTPUT MODELS FOR NEXT-STEP PREDICTIONS

Table 5.5 shows results for the various ML models applied to the house data. The results shown here are averaged across all 18 houses, the full results per house can be found in Appendix C. Table 5.5 includes the mean and standard deviation of the root mean squared error (RMSE), mean absolute error (MAE), and R^2 for all 18 houses in the analysis.

Table 5.5 shows that the methods based on linear regression (LR) (including regularised methods) are the top performers for next step (30 min) predictions. Ridge regression has a mean RMSE of 0.0171 and mean MAE of 0.0103, which is slightly lower than lasso (RMSE 0.0175, MAE 0.0107), elastic net (RMSE 0.0175, MAE 0.0107), and linear (RMSE 0.0177, MAE 0.0103) regression. There is quite a large disparity between the quickest and slowest models with an average computation time to train the model and make predictions of 0.13 s for LR and of 300 s for MLP. For LR, the average time taken to make predictions only was 0.39 milliseconds with a standard deviation of 0.06 milliseconds across the 18 houses. Due to LR being the fastest and simplest model, this is the model selected for the house data.

Figure 5.4 shows the boxplot of RMSE for linear regression over the last 3 folds of the 5 fold Time Series Split cross validation. The boxplot shows there is a significant difference between the RMSE score for different houses, such as house 0 and 1. House 15 has the smallest average loss while house 17 has the largest.

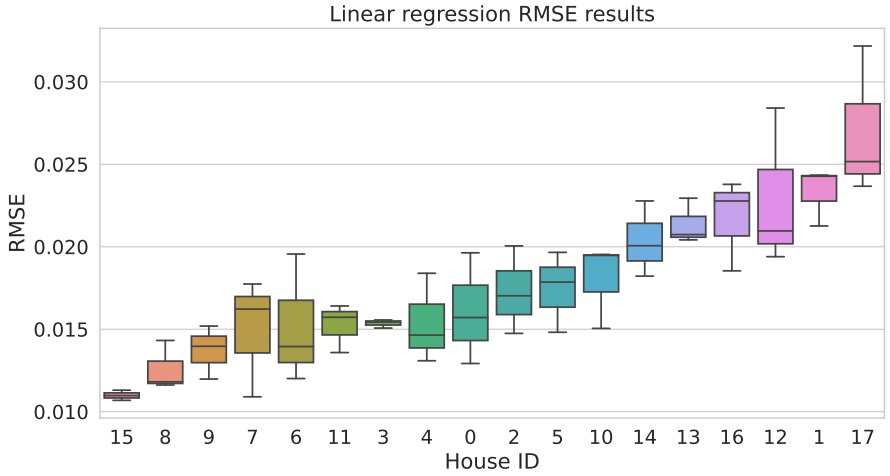

Figure 5.4: Boxplot of the RMSE results for all 18 houses over the final 3 folds of the 5 fold cross validation for linear regression.

Table 5.6: Mean and standard deviation of the computation time in seconds to train the model, RMSE, MAE, and coefficient of determination (R^2) of the multi-output LR models with and without average horizontal solar irradiance (Wh/m^2) across all 18 houses for the last 3 folds of the 5-fold cross validation on the scaled data.

Solar irradiance?	Time (s)	RMSE	MAE	R^2
No	0.13(6)	0.017 72(467)	0.010 28(286)	0.944 03(3492)
Yes	0.11(4)	0.017 69(474)	0.010 24(287)	0.944 11(3534)

5.5.1 *Addition of solar irradiance*

The missing solar data can be considered important as solar irradiance can significantly influence indoor temperatures and humidity levels. While the model could potentially indirectly capture some of the effect of solar irradiance through the outdoor air temperature, relative humidity, and wind speed, the direct impact of sunlight on indoor temperatures and heat gain is not fully accounted for without explicit solar data. Further, the solar load can cause heating of indoor surfaces, such as walls and windows, which in turn radiate heat into the room, leading to localized temperature variations and an effect on the overall thermal comfort inside a room. Radiators that are fixed to walls could indirectly capture some aspects of the solar load in the room as the sunlight heats the walls and the radiator may absorb some of this heat.

Table 5.6 presents the results of adding the average horizontal solar irradiance into the LR model. This data comes from the same source as the other climate data, which was collected at the Loughborough University campus weather station. The results are averaged across all 18 houses showing very similar RMSE, MAE, and coefficient of determination results for the models with and without the addition of solar irradiance.

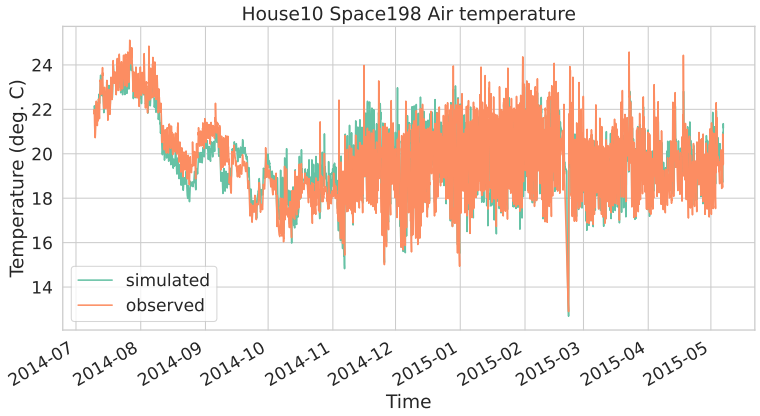

Figure 5.5: The observed and predicted values for the air temperature in space 200 within house 10 over the time period for all available data.

Thermal comfort modelling often depends on solar irradiance as comfort is a complex phenomenon influenced by various factors, including thermal variables, personal factors, activity levels, and psychological factors. Therefore, comprehensive thermal comfort assessments may benefit from incorporating other variables and considering the subjective nature of comfort. However, a simplified thermal model can still provide benefits, including being able to identify potential discomfort zones, areas with high or low temperatures, or humidity levels outside a defined comfort range.

In the subsequent sections of this chapter, the thermal model does not include the average horizontal solar irradiance to maintain symmetry with the car cabin model. There is also a lack of information regarding the exposure of each individual room to direct sunlight, such as the location within the house and potential shading from external factors like trees.

5.6 LONG-RUN PREDICTIONS

Figure 5.5 and Figure 5.6 show the long term, multi-step ahead predictions for the AT and RH in space 198, respectively, which represents the living room in house 10. These figures show that the predictions are able to closely follow the observed data over a longer period of time, which is just short of a full year. It is difficult to see the step-by-step predictions in these plots, however, it can be seen as an indication that the predictions are close to the observed values. The time taken to predict over the full length of each experiment was on average 4.8×10^{-8} seconds, or 0.048 µs, per predicted second with a standard deviation of 0.008 µs.

In order to see more closely what occurs in the day-to-day predictions, Figure 5.7 shows the AT predictions for four different rooms

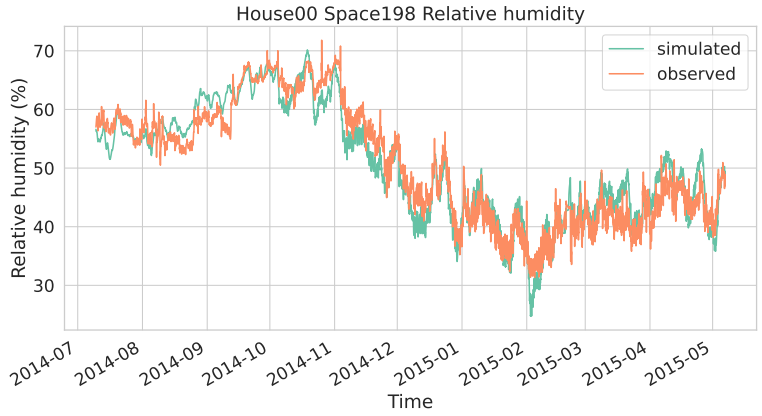

Figure 5.6: The observed and predicted values for the relative humidity in space 198 within house 10 over the time period for all available data.

in house 0 throughout January 2015, which corresponds to the plots in Figure 5.2. The rooms include the kitchen, living room, bedroom, and bathroom. Figure 5.7 shows that the LR model is able to track the variations in each of the rooms, for example, the model is able to capture that the kitchen tends to have a higher temperature than the bedroom. The errors also tend to be a bit larger when the temperature is lower than average, for example, at the beginning of January 2015, for almost a week, the temperatures in the house are much lower than the remainder of the month. The shape and profile of the plots is similar, but the values are sometimes shifted. Possible reasons for the shift down in the temperature could be because of the number of occupants in the room or because of missing factors, such as a layer of snow on the roof.

Recall that the observed sensor data was collected every 30 minutes, therefore it may be of interest to look closer at a specific day to better visualise the error given by the predictions. Figure 5.8 shows the observed and predicted AT for the living room in house 0 on the 16th of January 2015. This shows that the predictions are able to follow the peaks and valleys of the AT throughout this day, but the error is relatively large, particularly in the early hours of the day. This could possibly be related to the occupancy of the living room.

5.6.1 *NRMSE results*

For each house, the normalized root mean squared error (NRMSE) was calculated per input feature. Table 5.7 shows the top and bottom 5 NRMSE values for the features across all 18 houses. The average RMSE across all sensors is 1.82 and the average NRMSE is 0.068. TheNRMSE values range from 0.034 to 0.123 (3.4 % to 12.3 %), which is quite a large range of errors. The lowest NRMSE is for the air temperature in

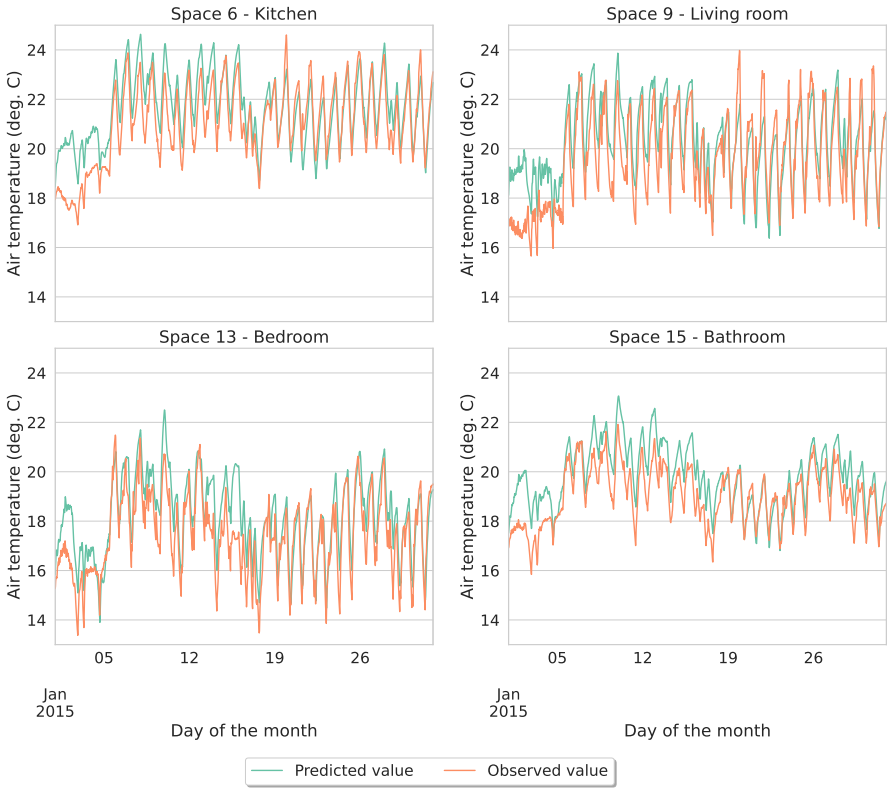

Figure 5.7: The observed and predicted values for the air temperature in four rooms of house 0 throughout January 2015. The rooms depicted in the top row are the kitchen (space 6) and living room (space 9), and in the bottom row is the bedroom (space 13) and bathroom (space 15).

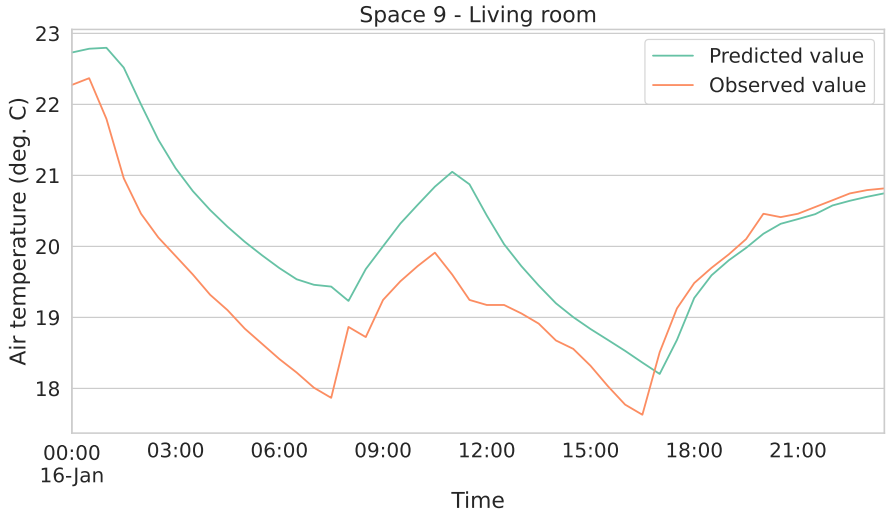

Figure 5.8: The observed and predicted values for the air temperature in the living room (space 9) of house 0 on the 16th of January 2015.

Table 5.7: Top 5 and bottom 5 NRMSE values for the house sensors, along with the RMSE across all 18 houses, arranged by NRMSE.

Sensor	RMSE	NRMSE
Space 237 Air temperature	0.384 442	0.033 761
Space 233 Air temperature	0.463 957	0.034 213
Space 311 Air temperature	0.476 685	0.034 282
Space 214 Air temperature	0.586 675	0.037 899
Space 66 Air temperature	0.589 714	0.038 221
⋮	⋮	⋮
Space 184 Air temperature	1.542 609	0.111 468
Space 124 Relative humidity	7.892 841	0.112 395
Space 108 Relative humidity	5.403 345	0.112 467
Space 229 Air temperature	2.048 509	0.122 960
Space 104 Relative humidity	4.272 015	0.123 234

space 237 at 3.4 %. The corresponding RMSE shows that the predicted values are, on average, within 0.38 °C of the actual value. The highest NRMSE is for the relative humidity in space 104, with a NRMSE of 12.3 %. The difference between the lowest and highest error is quite large.

In addition to looking at the RMSE and NRMSE averages across all responses, an analysis for the prediction of just air temperatures is examined. The average AT was calculated for each house by averaging the RMSE and NRMSE across the available AT variables. Table 5.8 shows the results for the average air temperature predictions in each house. The average RMSE and NRMSE across all 18 houses are 1.8 °C and 6.8 %, respectively. The errors for the multi-step predictions are larger than for the next-step predictions, which is expected, especially when predicting for nearly a year worth of data. The standard deviations for RMSE are also quite large, showing that for some predictions the error is close to zero, while others could be closer to 3 °C.

5.7 FEATURE IMPORTANCE

Figure 5.9 shows the feature importance across the prediction of all next states for space 10, which is a bedroom, in house 0. In this model, the coefficients for two climate variables were shrunk to zero, wind speed and total rainfall. Twelve other houses had a coefficient of zero for the wind speed and eight others had a coefficient of zero for total rainfall. Despite the low importance in these models, other models saw wind speed to be in the top four most important variables. This could be due to a characteristic of the house, that is, it is possible the

Table 5.8: The average air temperature RMSE and NRMSE values for each house, with the standard deviation in parenthesises.

House	RMSE	NRMSE
0	1.982(1405)	0.070(13)
1	2.325(1930)	0.082(16)
2	1.492(1205)	0.068(12)
3	1.309(1034)	0.062(10)
4	1.647(1539)	0.067(27)
5	1.991(2170)	0.061(21)
6	1.911(1014)	0.078(9)
7	1.971(1683)	0.073(15)
8	1.723(1430)	0.063(16)
9	1.764(1086)	0.073(21)
10	1.679(1404)	0.058(14)
11	1.988(1531)	0.073(18)
12	2.079(1769)	0.074(26)
13	2.287(2150)	0.065(20)
14	2.062(1721)	0.071(14)
15	1.372(1055)	0.066(15)
16	1.408(1551)	0.056(19)
17	1.927(1397)	0.069(15)

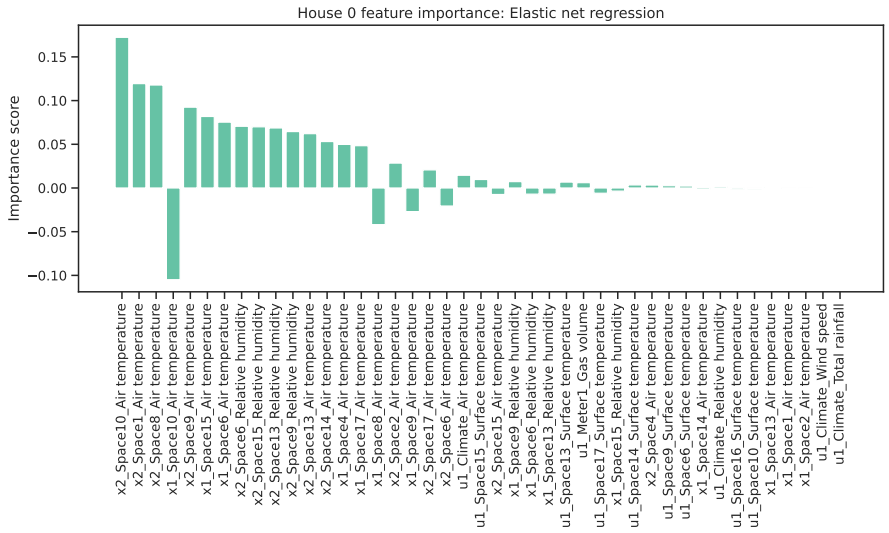

Figure 5.9: Feature importance for elastic net regression model on the data from house 0 across the predictions of all output variables.

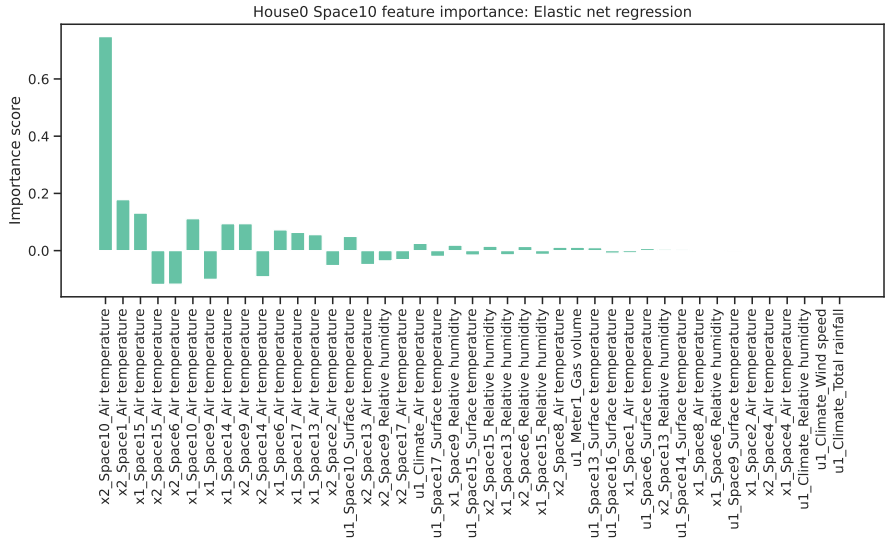

Figure 5.10: Feature importance for elastic net regression model on the data from house 0 for predicting only the air temperature in.

house where the feature was deemed importance has poor insulation or is positioned with little protection, such as that provided by trees or other houses.

In Figure 5.10, the feature importances are shown for house 10 when predicting only the AT in space 10, a bedroom. For all rooms, from hallways to kitchens to living rooms, it was found that the most recent state of the variable being predicted was the most important for predicting the next state. The second most recent state had a varying effect on the prediction of the next state, that is, it was second most important for six models, between second and tenth most important for nine models, and eleventh and onwards for another three models. This shows that the importance of the lag 2 is less certain than the importance of lag 1.

5.8 COMPARISON WITH CAR CABIN RESULTS

There are multiple aspects of the car cabin and house case studies which differ. First, the car cabin data comes from a controlled environment, therefore it may experience less unexpected events (for example, a large gust of wind) and therefore may not be symbolic of a real world setting. In this aspect, the house data provides an application in the real world as the data is from houses within the community. Another difference is the sampling rate of the data, with the car cabin having a 10 second time interval and the house having a 30 minute interval. The variable time intervals allow us to see if the same approach can make predictions for differing gaps between time points. These considerations must be taken into account when comparing the results from the models.

The car cabin analysis in Chapter 4 found that the top performing models were based on linear models with linear regression producing very similar results to the regularisation techniques (ridge, lasso, and elastic net regression). A similar result is produced for the house case study. Both case studies were able to produce a reasonable accuracy in terms of both next-step and multi-step ahead predictions.

5.8.1 *Feature importance comparison*

The car cabin and house are different environments with a different set of variables, however comparisons can be made about the influence certain variables have on a predicted value. For example, the hypothesis is that due to the time series nature of the data, it is believed that the previous values will be the most important in predicting the next value. The analysis of feature importances for both the car cabin (Section 4.7) and house (Section 5.7) confirmed this hypothesis. Although the house implemented a lag of 2 and the car cabin used a lag of 1, a common theme amongst feature importances is that the most recent value of the variable being predicted is found the most important for predicting the next value.

5.9 CHAPTER SUMMARY

In this chapter, the REFIT Smart Home data was introduced and pre-processed for the use in a ML thermal model. A lag of 2 was implemented for this data and a range of ML models were fit using Time Series Split cross validation. The LR model was the chosen model for this data, showing that the LR model could capture the temperature changes within various rooms of each house. The errors for next-step (RMSE 0.017) were smaller than for the longer term predictions (RMSE 1.82). Furthermore, the hypothesis that the current value of any feature would be the most important for predicting the next value has been confirmed. This case study provided another setting to adapt the same modelling approach the car cabin case study in Chapter 4, which was introduced in Chapter 3.

6 CONCLUSIONS

6.1 DISCUSSION

In this thesis, two case studies were pursued to develop a thermal modelling approach using machine learning. The case studies were conducted in a car cabin and house, both using the same methodological approach. The research questions that were investigated are listed below, including an answer to the question and a brief summary of the key results.

6.1.1 *Research questions*

1. Can a data-driven, machine learning model, with little details about the physical environment itself, be used to model thermal aspects of an environment?
    a) Can the top performing machine learning (ML) model for a car cabin provide faster and more accurate predictions than the industry standard? If so, to what extent?
    b) Can the top performing ML model for a house provide a fast and accurate prediction of thermal variables?

To address these questions, a comparison of the chosen ML model based on linear regression was compared to a physics-based 0D model for the car cabin, both in terms of accuracy and speed, as shown in Section 4.8. For the physics-based model, the normalized root mean squared error (NRMSE) was 1.5 % for the temperature compared to an NRMSE of 0.5 % for the average air temperature in the ML model. The average NRMSE for the ML model over all sensors, including temperatures, velocities, and humidities, is 1.1 %. Comparing the physics-based and ML models in terms of temperature, showed that the ML model can indeed provide more accurate predictions over the physics-based model. Furthermore, for the physics-based model when the air conditioning (AC) loop is on, the model performed about 3 % for the high pressure and 6 % for the low pressure. The linear regression (LR) shows an improvement over this with the best feature (surface temperature at driver's windshield) having an NRMSE of 0.3 % and worst feature (air velocity at passenger's foot) having an NRMSE of 2.5 %.

In terms of the amount of elapsed time to compute 1 second of simulated time, the physics-based model resulted in speeds of 7.6 ms s^{-1} for warm-up mode and 250 ms s^{-1} for cool-down mode. Comparing the warm-up speed to the speed of the ML model, which is 0.007 ms s^{-1},

shows a 1400-fold speed up in predictions. Therefore, the chosen ML model is able to outperform the physics-based model.

Furthermore, Section 5.5 built an ML model for a house using the same approach. The results for the house show a good accuracy, with an average NRMSE of across all sensors of 0.068 and a very fast average prediction speed of $0.048\,\mu s\,s^{-1}$.

2. Which machine learning model provides the most accurate next step predictions for the thermal aspects of various locations in different thermal environments?
    1. Which ML methods are the most accurate for a car cabin?
    2. Which ML methods are the most accurate for a house?
    3. Are the ML techniques that perform best for the two case studies the same? If not, why are they different?

For both the car cabin and house case study, a LR was chosen to model the data, although the results were comparable to other models, such as ridge, lasso, and elastic net regression. The LR was selected as the model is simple, but robust, and thermal systems are mainly linear with respect to their inputs. The next-step predictions were found using normalised data.

In the car cabin case study, the LR model resulted in an root mean squared error (RMSE) of 0.0040 and R^2 value of 0.995. The house case study found that the LR model produced an RMSE of 0.017 and R^2 value of 0.950. The R^2 values for the car cabin and house indicate that most of the variability in the response can be explained by the selected regression models (99.5 % or 95.0 %, respectively). In both case studies, the LR model showed a similar result could be achieved using the regularisation methods, while the tree- and neighbours-based methods showed poor performance on the data, and the multilayer perceptron (MLP) results were in between.

3. Can the top performing machine learning model a provide stable, long-term prediction?
    1. Is the model stable for a car cabin?
    2. Is the model stable for a house?

Both case studies examined a multi-step, longer term prediction using the respective LR models. These results showed that the LR model was able to provide open loop predictions with good accuracy. Figure 4.10 showed the multi-step ahead predictions for a climatic wind tunnel (CWT) trial. The model is able to closely and differentially predict the head, torso, and foot air temperature for the driver. The long run predictions achieved an NRMSE of 1.1 %, showing a stable model for the car cabin.

For the house case study, Figure 5.8 showed an example of the predicted versus observed values in a day. It is striking that the model is able to capture the peaks and dips in temperature throughout a day, however the error is relatively large across long term predictions across all houses (NRMSE 6.8 %). Despite the larger error in the house case study, this is still a promising method for the long run prediction of thermal variables.

4. What thermal and environmental features are most important in achieving a high accuracy machine learning thermal model?
    1. What features are most important for a car cabin?
    2. What features are most important for a house?
    3. What are the similarities and differences in the features for the two case studies and why?

When using the full set of features to predict all possible outputs, the results show that the individual importances for features are similar and have small importance score. However, when isolating an output of interest, it is found that the current value of that feature is the most important in predicting the next value. In Section 4.7, it is shown that the air temperature at the drivers head is the most important for the prediction of the next air temperature at the drivers head. Similarly, in Section 5.7, the results show that the most recent value of air temperature in space 10 (a bedroom) is the most important for predicting the next value of the same feature. In the house case study, a lag of 2 was used for modelling, therefore, a previous value of the air temperature is also used for prediction. The importance of this previous value varied depending on the room and house.

6.1.2 *Limitations*

To produce meaningful results that can be applied to real world problems, models require appropriate, robust data that reflect everyday experiences and events. For example, the car data had only one experiment with moisture present inside the cabin, which meant that the model had minimal data to train on for that type of condition inside the cabin. Similarly, the house case study mainly included detached houses, which is not representative of the UK housing market and may experience more heat loss. Therefore, further testing is required on houses that are not detached, such as terraced or semi-detached houses, to ensure the approach is suitable for these cases.

6.1.3 *Advances to current understanding*

The advancements made in this thesis are potentially beneficial to a wide range of thermal environments, ranging from the temperature of

a swimming pool to the humidity in a space craft. This thesis adapted a common approach to two different thermal environments in order to examine if a unified approach was suitable. Not only did this thesis adapt a common approach, but it also aimed to simplify what could often be seen as complex modelling, for example, physics-based models that require detailed and sometimes unavailable data, such as geometry and material information. This thesis has shown that a simplified process can speed up the predictions of thermal variables without necessarily compromising the quality of the results. A further contribution is that one single model is built to predict the thermal variables multiple locations. In the car cabin, this is done through the prediction of thermal variables at the head, torso, and foot of the occupants, and for the house, this is through the prediction of various rooms. Adapting this approach leads to fast predictions that are able to differentiate between these various locations.

6.2 FUTURE WORK

The long-term vision for this work is that we can design, build, and control thermal environments (including houses and vehicles) more optimally based on virtual experiments. While virtual environments or simulation models have been feasible, using them for optimisation is difficult as they are hard to construct and often too computationally slow. Given this work, future work can now go ahead including:

1. optimising the control logic for vehicles to reduce energy cost while maintaining comfort;
2. optimising the control logic for houses in the same way;
3. optimising the physical design of both vehicles and houses towards the same goal.

6.3 SUMMARY

The work in this thesis adapted a machine learning approach to thermal modelling. The aim was to simplify the modelling process while providing fast and accurate predictions of various thermal variables. Two case studies, namely a car cabin and a house, were investigated and modelled using LR. The resulting thermal predictions for various spaces showed promise for a ML approach, specifically using a linear model such as LR. Furthermore, this modelling approach shows a great opportunity for use in a wide range of other thermal environments.

A

ETHICAL APPROVAL

Below is the ethical approval form from Coventry University, approved on 20 September 2022.

Certificate of Ethical Approval

Applicant:	Brandi Jess
Project Title:	Fast, details, accurate simulation of thermal environments using machine learning

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:	20 Sep 2022
Project Reference Number:	P141185

B

CWT TRIAL MEASUREMENT DATA PLOTS

This appendix provides plots of the measurement data collected during the car cabin CWT experiments 2 through 5, similar to Figure 4.5 that shows data for CWT 1. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

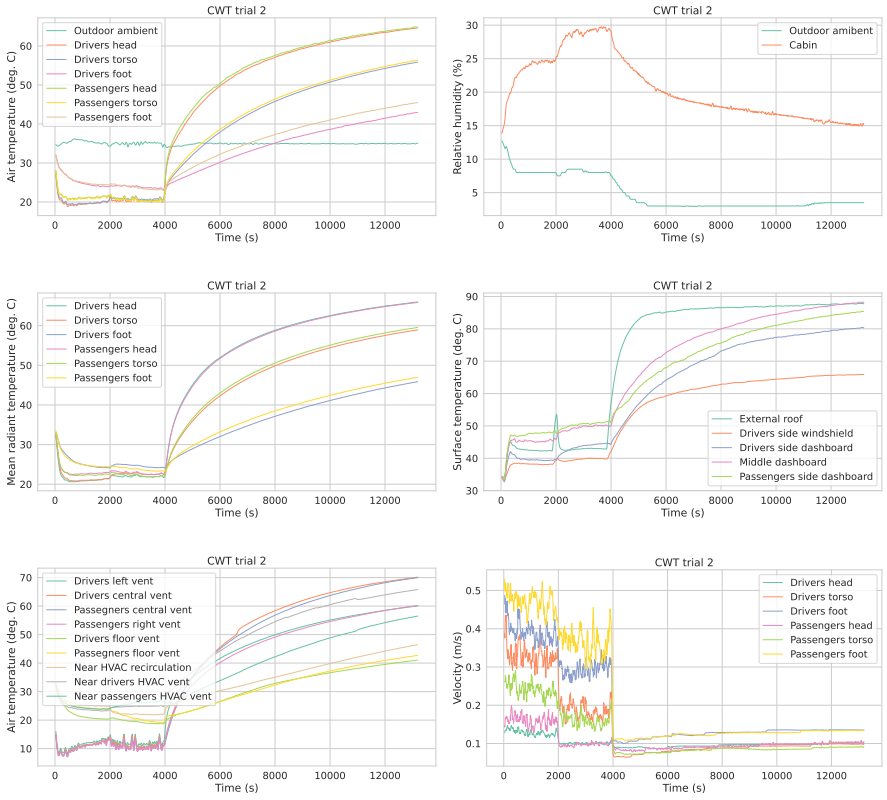

Figure B.1: Measurements from CWT trial 2. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

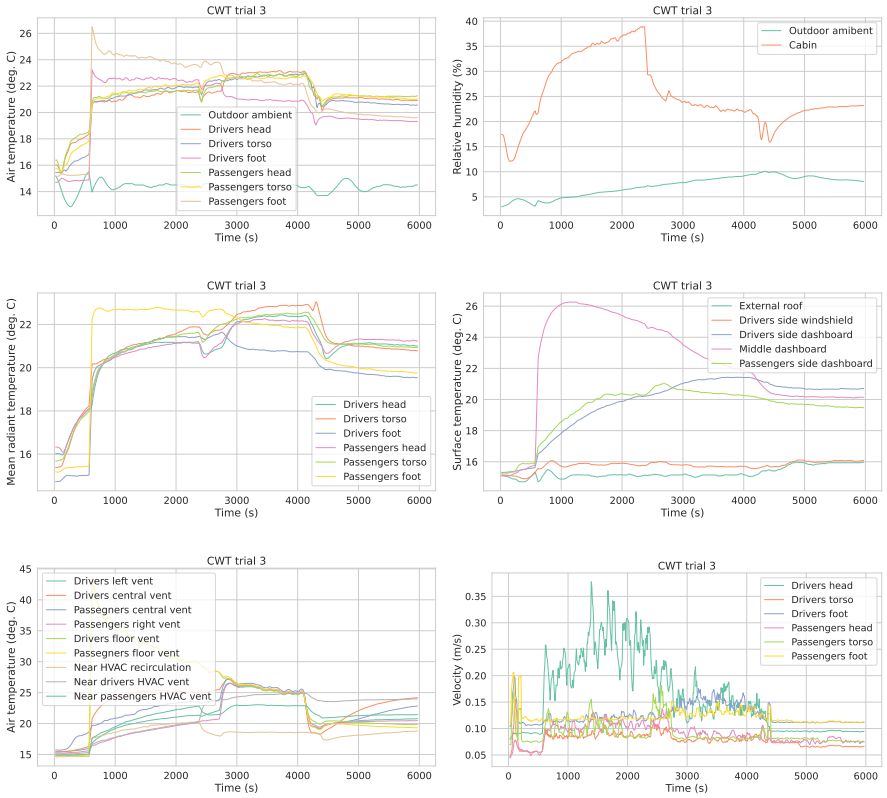

Figure B.2: Measurements from CWT trial 3. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

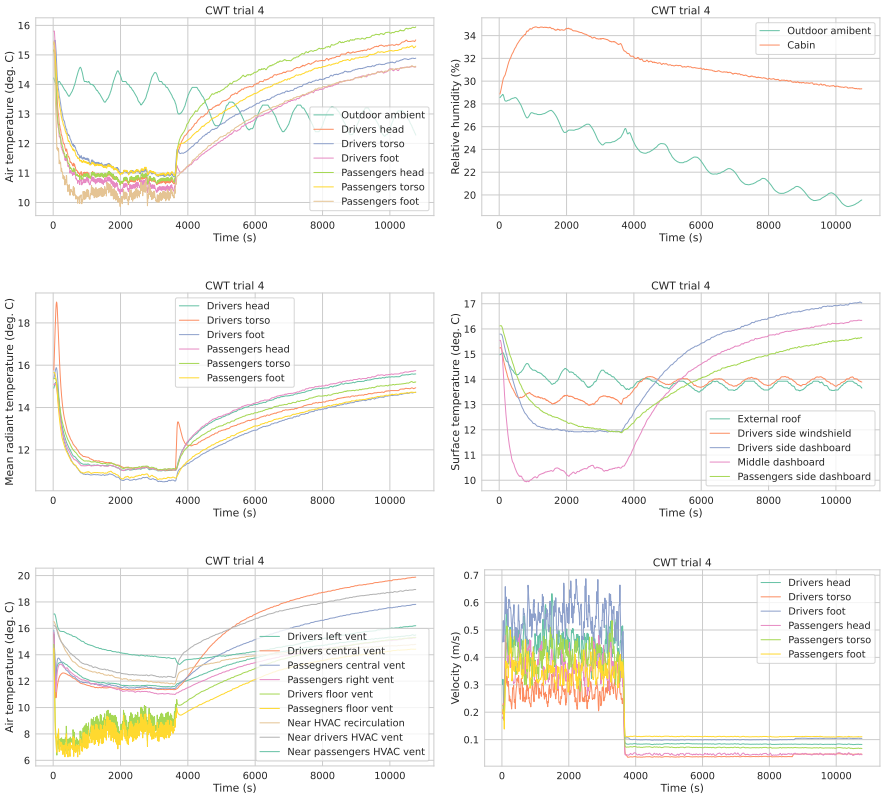

Figure B.3: Measurements from CWT trial 4. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

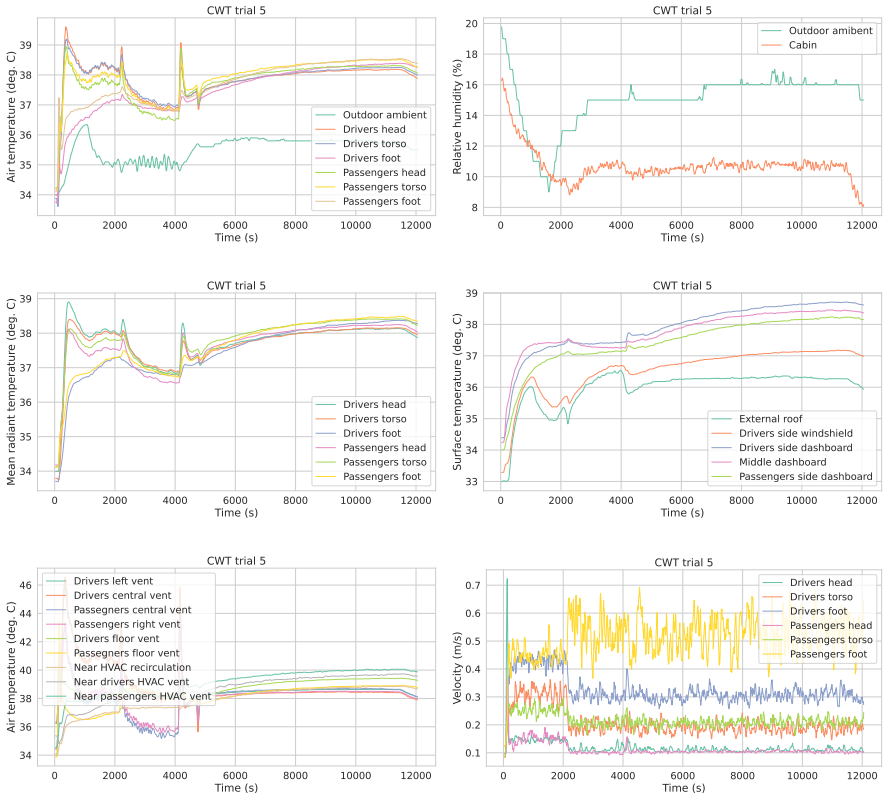

Figure B.4: Measurements from CWT trial 5. From left to right, top to bottom this includes air temperatures, relative humidities, mean radiant temperatures, surface temperatures, vent air temperatures, and air velocities.

C

ML MODEL PERFORMANCE PER HOUSE

This appendix shows the full results of the machine learning models per house (0–17). A summary of these results was provided in Table 5.5. The computation time is given in seconds and measures both the model training and prediction time. The mean squared error (MSE), RMSE, mean absolute error (MAE), and the coefficient of determination (R^2) are averaged across the last 3-folds of the 5-fold TimeSeriesSplit cross validation.

Table C.1: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 0.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.2	0.000 33 ± 0.000 13	0.016 09 ± 0.002 75	0.009 68 ± 0.002 29	0.967 68 ± 0.018 33
Ridge	0.1	0.000 31 ± 0.000 11	0.016 01 ± 0.002 61	0.009 74 ± 0.002 14	0.969 05 ± 0.015 91
Lasso	32.0	0.000 24 ± 0.000 03	0.014 28 ± 0.001 12	0.008 49 ± 0.000 32	0.978 24 ± 0.003 08
Elastic net	32.0	0.000 24 ± 0.000 03	0.014 28 ± 0.001 12	0.008 49 ± 0.000 32	0.978 24 ± 0.003 08
K-neighbors	2.2	0.003 76 ± 0.001 07	0.057 48 ± 0.008 78	0.043 45 ± 0.006 35	0.641 65 ± 0.114 28
Decision tree	1.3	0.006 04 ± 0.001 69	0.073 18 ± 0.012 27	0.056 03 ± 0.007 40	0.428 28 ± 0.086 88
Random forest	13.7	0.005 38 ± 0.001 66	0.068 38 ± 0.012 77	0.051 94 ± 0.007 46	0.495 31 ± 0.087 90
Extra trees	1.8	0.005 11 ± 0.001 34	0.065 71 ± 0.011 76	0.050 10 ± 0.007 19	0.500 38 ± 0.101 72
XGBoost	365.6	0.001 27 ± 0.000 74	0.030 19 ± 0.008 93	0.019 39 ± 0.003 79	0.913 55 ± 0.003 12
MLP	71.2	0.001 36 ± 0.001 40	0.029 57 ± 0.015 52	0.018 73 ± 0.010 37	0.839 42 ± 0.181 57

Table C.2: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 1.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 64 ± 0.000 07	0.023 30 ± 0.001 44	0.013 71 ± 0.001 11	0.929 73 ± 0.024 86
Ridge	0.7	0.000 64 ± 0.000 07	0.023 35 ± 0.001 38	0.013 75 ± 0.001 10	0.929 35 ± 0.025 67
Lasso	10.0	0.000 64 ± 0.000 07	0.023 37 ± 0.001 34	0.013 81 ± 0.001 06	0.929 42 ± 0.025 33
Elastic net	11.1	0.000 64 ± 0.000 07	0.023 37 ± 0.001 34	0.013 81 ± 0.001 06	0.929 42 ± 0.025 33
K-neighbors	5.2	0.003 62 ± 0.001 26	0.058 01 ± 0.009 44	0.042 46 ± 0.007 24	0.671 34 ± 0.035 05
Decision tree	0.7	0.005 79 ± 0.002 20	0.073 46 ± 0.014 07	0.054 78 ± 0.009 55	0.470 98 ± 0.067 90
Random forest	15.3	0.005 32 ± 0.002 57	0.069 38 ± 0.017 13	0.050 28 ± 0.012 21	0.539 40 ± 0.061 63
Extra trees	2.3	0.004 64 ± 0.002 00	0.064 26 ± 0.014 10	0.046 21 ± 0.010 94	0.598 88 ± 0.029 63
XGBoost	9.5	0.000 81 ± 0.000 12	0.026 43 ± 0.001 94	0.015 34 ± 0.001 04	0.918 01 ± 0.015 13
MLP	130.7	0.000 78 ± 0.000 02	0.026 08 ± 0.000 17	0.017 08 ± 0.000 58	0.917 06 ± 0.026 50

Table C.3: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 2.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.2	0.000 51 ± 0.000 12	0.017 28 ± 0.002 17	0.009 84 ± 0.001 56	0.939 97 ± 0.017 72
Ridge	0.0	0.000 52 ± 0.000 12	0.017 28 ± 0.002 18	0.009 83 ± 0.001 56	0.939 95 ± 0.017 77
Lasso	6.6	0.000 52 ± 0.000 13	0.017 48 ± 0.002 25	0.009 96 ± 0.001 59	0.939 40 ± 0.018 34
Elastic net	7.2	0.000 52 ± 0.000 13	0.017 47 ± 0.002 25	0.009 95 ± 0.001 59	0.939 41 ± 0.018 34
K-neighbors	0.3	0.002 14 ± 0.000 63	0.043 60 ± 0.005 46	0.032 08 ± 0.003 69	0.648 13 ± 0.148 49
Decision tree	0.0	0.005 25 ± 0.000 95	0.070 39 ± 0.006 37	0.056 73 ± 0.005 25	0.077 56 ± 0.338 76
Random forest	4.4	0.002 58 ± 0.000 67	0.047 60 ± 0.004 74	0.035 76 ± 0.003 30	0.583 29 ± 0.161 13
Extra trees	2.2	0.002 27 ± 0.000 65	0.043 89 ± 0.005 17	0.032 96 ± 0.003 17	0.639 45 ± 0.151 87
XGBoost	48.0	0.000 74 ± 0.000 19	0.021 60 ± 0.003 42	0.013 08 ± 0.001 26	0.909 84 ± 0.029 13
MLP	71.8	0.000 64 ± 0.000 15	0.020 94 ± 0.002 72	0.014 42 ± 0.001 31	0.916 77 ± 0.028 66

Table C.4: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 3.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.3	0.000 35 ± 0.000 01	0.015 36 ± 0.000 21	0.009 08 ± 0.000 15	0.954 81 ± 0.016 74
Ridge	0.1	0.000 35 ± 0.000 01	0.015 26 ± 0.000 18	0.008 90 ± 0.000 07	0.955 11 ± 0.016 68
Lasso	31.2	0.000 35 ± 0.000 02	0.015 66 ± 0.000 11	0.009 36 ± 0.000 13	0.954 25 ± 0.016 65
Elastic net	33.2	0.000 35 ± 0.000 02	0.015 66 ± 0.000 11	0.009 36 ± 0.000 13	0.954 26 ± 0.016 67
K-neighbors	2.0	0.002 79 ± 0.000 78	0.049 59 ± 0.007 06	0.037 83 ± 0.005 38	0.638 76 ± 0.076 11
Decision tree	1.1	0.005 43 ± 0.000 74	0.070 39 ± 0.004 84	0.055 62 ± 0.003 85	0.302 46 ± 0.204 38
Random forest	2.0	0.004 40 ± 0.000 76	0.060 55 ± 0.003 95	0.048 55 ± 0.004 42	0.337 40 ± 0.357 18
Extra trees	6.9	0.003 76 ± 0.000 41	0.056 70 ± 0.002 86	0.044 83 ± 0.002 84	0.459 97 ± 0.232 61
XGBoost	39.2	0.000 41 ± 0.000 06	0.016 37 ± 0.000 76	0.009 90 ± 0.000 63	0.941 58 ± 0.030 37
MLP	60.1	0.000 46 ± 0.000 02	0.018 49 ± 0.000 25	0.012 56 ± 0.000 12	0.940 79 ± 0.020 29

Table C.5: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 4.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.2	0.000 38 ± 0.000 09	0.015 38 ± 0.002 23	0.008 66 ± 0.001 13	0.955 26 ± 0.011 22
Ridge	62.9	0.000 37 ± 0.000 08	0.015 32 ± 0.002 12	0.008 63 ± 0.001 07	0.955 96 ± 0.010 27
Lasso	37.1	0.000 34 ± 0.000 02	0.015 24 ± 0.000 92	0.008 87 ± 0.000 75	0.959 33 ± 0.004 56
Elastic net	35.8	0.000 34 ± 0.000 02	0.015 24 ± 0.000 92	0.008 87 ± 0.000 75	0.959 33 ± 0.004 56
K-neighbors	1.1	0.005 15 ± 0.001 91	0.067 08 ± 0.014 27	0.050 55 ± 0.010 63	0.423 86 ± 0.156 98
Decision tree	0.6	0.009 32 ± 0.003 54	0.090 36 ± 0.019 26	0.069 90 ± 0.014 49	−0.071 97 ± 0.328 85
Random forest	1.9	0.008 68 ± 0.002 99	0.086 29 ± 0.018 00	0.066 83 ± 0.013 73	0.023 32 ± 0.293 68
Extra trees	1.2	0.009 77 ± 0.003 97	0.090 68 ± 0.022 35	0.071 72 ± 0.017 92	−0.069 70 ± 0.368 88
XGBoost	224.6	0.001 74 ± 0.001 29	0.034 30 ± 0.016 51	0.019 85 ± 0.009 31	0.818 01 ± 0.121 51
MLP	69.0	0.000 40 ± 0.000 08	0.017 06 ± 0.002 72	0.010 28 ± 0.001 01	0.951 43 ± 0.012 73

Table C.6: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 5.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.2	0.000 52 ± 0.000 09	0.017 45 ± 0.002 00	0.008 24 ± 0.001 32	0.952 88 ± 0.009 23
Ridge	28.9	0.000 40 ± 0.000 10	0.015 19 ± 0.002 16	0.008 20 ± 0.001 39	0.963 89 ± 0.013 37
Lasso	18.0	0.000 42 ± 0.000 11	0.016 10 ± 0.002 35	0.009 02 ± 0.001 62	0.961 11 ± 0.015 47
Elastic net	18.4	0.000 42 ± 0.000 11	0.016 11 ± 0.002 35	0.009 02 ± 0.001 62	0.961 10 ± 0.015 47
K-neighbors	2.0	0.003 85 ± 0.001 74	0.057 55 ± 0.014 58	0.043 65 ± 0.011 43	0.636 61 ± 0.164 10
Decision tree	0.7	0.009 56 ± 0.005 91	0.089 80 ± 0.028 97	0.070 53 ± 0.023 79	0.138 89 ± 0.418 35
Random forest	2.3	0.009 88 ± 0.007 29	0.086 17 ± 0.035 41	0.068 26 ± 0.030 07	0.216 96 ± 0.485 34
Extra trees	5.1	0.008 84 ± 0.006 41	0.081 96 ± 0.033 36	0.064 23 ± 0.028 00	0.284 84 ± 0.426 73
XGBoost	103.2	0.000 96 ± 0.000 69	0.025 81 ± 0.011 56	0.015 16 ± 0.007 16	0.924 47 ± 0.046 68
MLP	218.4	0.000 51 ± 0.000 05	0.018 23 ± 0.001 61	0.009 16 ± 0.001 46	0.954 34 ± 0.009 14

Table C.7: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 6.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 28 ± 0.000 11	0.015 17 ± 0.003 20	0.007 72 ± 0.000 63	0.969 67 ± 0.005 92
Ridge	0.1	0.000 19 ± 0.000 03	0.012 64 ± 0.000 92	0.007 60 ± 0.000 59	0.975 26 ± 0.009 25
Lasso	14.9	0.000 20 ± 0.000 02	0.013 16 ± 0.000 65	0.008 52 ± 0.001 03	0.975 97 ± 0.006 92
Elastic net	15.1	0.000 21 ± 0.000 02	0.013 18 ± 0.000 66	0.008 53 ± 0.001 05	0.975 95 ± 0.006 91
K-neighbors	1.4	0.003 90 ± 0.001 98	0.057 41 ± 0.014 92	0.043 83 ± 0.011 32	0.634 14 ± 0.123 37
Decision tree	0.6	0.007 16 ± 0.003 26	0.077 99 ± 0.017 73	0.061 52 ± 0.013 24	0.270 74 ± 0.342 36
Random forest	1.2	0.007 98 ± 0.004 45	0.076 80 ± 0.024 06	0.061 53 ± 0.018 98	0.114 44 ± 0.599 56
Extra trees	3.9	0.007 48 ± 0.004 22	0.074 64 ± 0.023 63	0.059 65 ± 0.018 56	0.177 99 ± 0.532 37
XGBoost	67.4	0.000 62 ± 0.000 41	0.021 09 ± 0.007 96	0.013 14 ± 0.004 37	0.940 81 ± 0.027 64
MLP	206.7	0.000 30 ± 0.000 11	0.016 31 ± 0.003 25	0.010 20 ± 0.001 41	0.966 67 ± 0.006 62

Table C.8: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 7.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 32 ± 0.000 12	0.014 96 ± 0.002 93	0.008 20 ± 0.001 91	0.970 99 ± 0.012 58
Ridge	44.8	0.000 20 ± 0.000 07	0.012 14 ± 0.002 35	0.007 99 ± 0.001 76	0.981 19 ± 0.007 98
Lasso	18.4	0.000 24 ± 0.000 11	0.013 22 ± 0.003 10	0.008 69 ± 0.002 20	0.978 34 ± 0.010 67
Elastic net	19.0	0.000 24 ± 0.000 10	0.013 43 ± 0.002 93	0.008 81 ± 0.002 12	0.977 99 ± 0.010 22
K-neighbors	2.4	0.005 27 ± 0.003 33	0.061 47 ± 0.016 04	0.048 21 ± 0.014 36	0.457 12 ± 0.382 98
Decision tree	0.9	0.008 05 ± 0.003 31	0.080 59 ± 0.017 01	0.064 09 ± 0.015 06	0.090 25 ± 0.578 05
Random forest	5.9	0.007 17 ± 0.003 14	0.070 64 ± 0.014 65	0.056 73 ± 0.013 99	0.230 04 ± 0.510 35
Extra trees	5.2	0.007 35 ± 0.003 21	0.071 92 ± 0.014 78	0.058 11 ± 0.014 45	0.173 91 ± 0.559 88
XGBoost	298.9	0.001 07 ± 0.000 79	0.023 66 ± 0.008 06	0.015 06 ± 0.005 60	0.897 38 ± 0.082 73
MLP	1799.3	0.000 39 ± 0.000 14	0.017 43 ± 0.003 10	0.011 12 ± 0.002 91	0.962 28 ± 0.015 55

Table C.9: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 8.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 25 ± 0.000 07	0.012 58 ± 0.001 23	0.007 09 ± 0.000 35	0.966 78 ± 0.003 79
Ridge	0.1	0.000 25 ± 0.000 07	0.012 58 ± 0.001 23	0.007 09 ± 0.000 35	0.966 78 ± 0.003 79
Lasso	26.6	0.000 27 ± 0.000 07	0.013 98 ± 0.001 13	0.008 07 ± 0.000 35	0.965 03 ± 0.003 70
Elastic net	27.0	0.000 27 ± 0.000 07	0.013 98 ± 0.001 13	0.008 07 ± 0.000 35	0.965 03 ± 0.003 70
K-neighbors	1.1	0.003 44 ± 0.000 32	0.054 09 ± 0.003 14	0.041 46 ± 0.001 59	0.381 01 ± 0.126 15
Decision tree	0.5	0.007 14 ± 0.001 01	0.078 80 ± 0.004 76	0.062 03 ± 0.004 21	−0.117 17 ± 0.168 47
Random forest	4.6	0.005 49 ± 0.000 37	0.065 96 ± 0.001 76	0.052 95 ± 0.001 80	−0.008 07 ± 0.168 46
Extra trees	3.1	0.005 01 ± 0.000 24	0.063 37 ± 0.001 20	0.050 20 ± 0.001 36	0.093 87 ± 0.163 13
XGBoost	114.5	0.000 57 ± 0.000 07	0.018 80 ± 0.000 72	0.011 94 ± 0.000 34	0.910 61 ± 0.013 91
MLP	94.6	0.000 30 ± 0.000 08	0.014 69 ± 0.001 22	0.009 17 ± 0.000 49	0.957 76 ± 0.003 48

Table C.10: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 9.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 21 ± 0.000 03	0.013 71 ± 0.001 32	0.008 89 ± 0.001 12	0.975 89 ± 0.002 20
Ridge	0.0	0.000 21 ± 0.000 03	0.013 71 ± 0.001 32	0.008 89 ± 0.001 12	0.975 89 ± 0.002 20
Lasso	11.5	0.000 22 ± 0.000 04	0.013 79 ± 0.001 42	0.008 92 ± 0.001 17	0.975 62 ± 0.001 99
Elastic net	11.1	0.000 22 ± 0.000 04	0.013 79 ± 0.001 42	0.008 92 ± 0.001 17	0.975 62 ± 0.001 99
K-neighbors	2.6	0.003 47 ± 0.001 14	0.054 81 ± 0.008 73	0.041 98 ± 0.007 07	0.530 79 ± 0.142 63
Decision tree	0.4	0.006 35 ± 0.002 25	0.073 79 ± 0.013 09	0.058 09 ± 0.010 40	0.222 60 ± 0.214 94
Random forest	4.4	0.005 32 ± 0.002 09	0.065 38 ± 0.012 81	0.052 03 ± 0.010 17	0.279 83 ± 0.301 78
Extra trees	2.5	0.004 70 ± 0.001 83	0.062 03 ± 0.011 93	0.049 10 ± 0.009 84	0.376 98 ± 0.245 86
XGBoost	48.0	0.000 28 ± 0.000 07	0.015 59 ± 0.002 25	0.010 07 ± 0.001 74	0.966 35 ± 0.007 71
MLP	419.8	0.000 24 ± 0.000 04	0.014 71 ± 0.001 52	0.010 16 ± 0.001 83	0.972 88 ± 0.002 00

Table C.11: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 10.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 45 ± 0.000 09	0.018 02 ± 0.002 10	0.011 05 ± 0.001 52	0.937 79 ± 0.013 71
Ridge	0.1	0.000 46 ± 0.000 09	0.018 08 ± 0.002 13	0.011 08 ± 0.001 55	0.936 51 ± 0.014 02
Lasso	32.3	0.000 48 ± 0.000 10	0.019 29 ± 0.002 51	0.011 99 ± 0.001 79	0.934 71 ± 0.014 61
Elastic net	33.3	0.000 48 ± 0.000 10	0.019 29 ± 0.002 51	0.011 99 ± 0.001 79	0.934 71 ± 0.014 61
K-neighbors	3.7	0.003 72 ± 0.001 58	0.056 94 ± 0.009 44	0.043 84 ± 0.008 68	0.532 58 ± 0.156 67
Decision tree	0.0	0.024 97 ± 0.014 42	0.139 40 ± 0.034 35	0.119 44 ± 0.033 11	−2.283 71 ± 1.615 70
Random forest	2.1	0.007 77 ± 0.005 03	0.076 65 ± 0.018 92	0.061 40 ± 0.018 71	−0.008 79 ± 0.591 10
Extra trees	1.8	0.010 08 ± 0.006 40	0.086 57 ± 0.020 66	0.071 12 ± 0.020 58	−0.345 38 ± 0.747 89
XGBoost	50.1	0.001 03 ± 0.000 68	0.025 26 ± 0.006 07	0.015 28 ± 0.005 25	0.880 26 ± 0.067 73
MLP	86.3	0.000 47 ± 0.000 08	0.019 02 ± 0.002 09	0.012 16 ± 0.001 52	0.936 89 ± 0.013 62

Table C.12: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 11.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 43 ± 0.000 06	0.015 24 ± 0.001 20	0.008 15 ± 0.001 04	0.950 51 ± 0.005 76
Ridge	31.9	0.000 43 ± 0.000 06	0.015 30 ± 0.001 22	0.008 19 ± 0.001 06	0.950 51 ± 0.005 77
Lasso	18.5	0.000 45 ± 0.000 05	0.016 46 ± 0.001 02	0.009 11 ± 0.000 95	0.948 74 ± 0.005 89
Elastic net	18.3	0.000 45 ± 0.000 05	0.016 46 ± 0.001 02	0.009 11 ± 0.000 95	0.948 74 ± 0.005 89
K-neighbors	1.1	0.005 32 ± 0.002 52	0.062 08 ± 0.012 51	0.048 72 ± 0.010 80	0.497 63 ± 0.176 66
Decision tree	0.5	0.010 36 ± 0.003 64	0.090 82 ± 0.014 60	0.074 17 ± 0.013 63	−0.002 82 ± 0.224 92
Random forest	5.0	0.007 85 ± 0.003 34	0.072 72 ± 0.014 16	0.059 800 ± 0.013 120	0.275 49 ± 0.232 61
Extra trees	4.8	0.007 32 ± 0.003 18	0.069 96 ± 0.013 32	0.057 33 ± 0.012 32	0.319 33 ± 0.219 72
XGBoost	54.1	0.000 97 ± 0.000 42	0.022 71 ± 0.005 89	0.013 32 ± 0.004 42	0.910 18 ± 0.026 94
MLP	94.1	0.000 46 ± 0.000 06	0.016 76 ± 0.001 46	0.009 53 ± 0.001 22	0.947 71 ± 0.007 17

Table C.13: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 12.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 89 ± 0.000 48	0.022 92 ± 0.003 93	0.013 72 ± 0.001 92	0.893 53 ± 0.058 08
Ridge	0.8	0.000 69 ± 0.000 23	0.022 82 ± 0.003 49	0.014 40 ± 0.002 11	0.916 93 ± 0.029 22
Lasso	17.2	0.000 55 ± 0.000 07	0.020 35 ± 0.001 31	0.012 86 ± 0.000 92	0.934 76 ± 0.010 96
Elastic net	18.9	0.000 54 ± 0.000 08	0.020 22 ± 0.001 35	0.012 75 ± 0.000 91	0.934 32 ± 0.011 63
K-neighbors	2.8	0.004 26 ± 0.000 64	0.061 75 ± 0.002 89	0.047 78 ± 0.002 49	0.476 60 ± 0.041 05
Decision tree	0.4	0.007 63 ± 0.002 16	0.083 62 ± 0.010 78	0.064 85 ± 0.006 82	0.008 43 ± 0.218 26
Random forest	0.9	0.007 08 ± 0.001 58	0.079 53 ± 0.007 38	0.062 36 ± 0.005 12	0.134 62 ± 0.135 55
Extra trees	0.6	0.006 68 ± 0.001 94	0.077 09 ± 0.009 85	0.060 25 ± 0.007 27	0.163 20 ± 0.176 21
XGBoost	106.4	0.001 11 ± 0.000 51	0.028 24 ± 0.004 73	0.018 71 ± 0.003 44	0.880 41 ± 0.044 16
MLP	280.6	0.001 08 ± 0.000 20	0.029 61 ± 0.002 87	0.020 94 ± 0.002 27	0.868 33 ± 0.021 56

Table C.14: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 13.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 82 ± 0.000 04	0.021 37 ± 0.001 12	0.012 63 ± 0.001 08	0.926 60 ± 0.002 52
Ridge	0.0	0.000 82 ± 0.000 04	0.021 32 ± 0.001 09	0.012 52 ± 0.001 03	0.927 07 ± 0.002 58
Lasso	22.8	0.000 84 ± 0.000 05	0.022 28 ± 0.001 44	0.013 44 ± 0.001 31	0.924 53 ± 0.003 19
Elastic net	23.8	0.000 84 ± 0.000 05	0.022 28 ± 0.001 44	0.013 44 ± 0.001 31	0.924 53 ± 0.003 19
K-neighbors	12.4	0.004 11 ± 0.001 00	0.059 95 ± 0.007 03	0.045 94 ± 0.006 59	0.565 19 ± 0.111 16
Decision tree	0.8	0.007 88 ± 0.001 73	0.084 52 ± 0.009 89	0.065 91 ± 0.007 58	0.167 70 ± 0.187 25
Random forest	2.6	0.007 04 ± 0.001 23	0.074 50 ± 0.007 58	0.059 85 ± 0.007 09	0.299 26 ± 0.119 45
Extra trees	5.3	0.006 91 ± 0.001 34	0.074 18 ± 0.008 22	0.059 34 ± 0.007 56	0.314 05 ± 0.125 55
XGBoost	71.7	0.000 98 ± 0.000 11	0.024 58 ± 0.002 43	0.014 92 ± 0.001 85	0.911 71 ± 0.007 74
MLP	125.3	0.000 85 ± 0.000 05	0.022 64 ± 0.001 54	0.013 85 ± 0.001 06	0.923 09 ± 0.003 42

Table C.15: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 14.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.2	0.000 71 ± 0.000 16	0.020 35 ± 0.001 87	0.011 62 ± 0.000 92	0.933 93 ± 0.022 56
Ridge	0.0	0.000 72 ± 0.000 16	0.020 39 ± 0.001 85	0.011 68 ± 0.000 90	0.933 79 ± 0.022 42
Lasso	18.2	0.000 73 ± 0.000 16	0.021 35 ± 0.001 78	0.012 45 ± 0.000 74	0.932 19 ± 0.023 12
Elastic net	18.4	0.000 73 ± 0.000 16	0.021 35 ± 0.001 78	0.012 45 ± 0.000 74	0.932 19 ± 0.023 12
K-neighbors	21.8	0.005 21 ± 0.001 12	0.069 77 ± 0.007 76	0.052 10 ± 0.007 07	0.533 02 ± 0.027 02
Decision tree	1.4	0.008 10 ± 0.001 93	0.086 42 ± 0.011 69	0.067 02 ± 0.009 42	0.285 15 ± 0.028 61
Random forest	4.5	0.006 52 ± 0.002 17	0.075 21 ± 0.014 66	0.058 42 ± 0.012 78	0.438 38 ± 0.099 83
Extra trees	2.9	0.005 92 ± 0.001 84	0.072 03 ± 0.013 62	0.055 31 ± 0.011 82	0.502 58 ± 0.082 51
XGBoost	103.2	0.000 86 ± 0.000 04	0.023 67 ± 0.000 36	0.014 42 ± 0.000 62	0.919 24 ± 0.014 19
MLP	246.9	0.000 88 ± 0.000 11	0.024 81 ± 0.000 96	0.015 49 ± 0.000 86	0.918 26 ± 0.019 37

Table C.16: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 15.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 22 ± 0.000 01	0.010 99 ± 0.000 25	0.007 51 ± 0.000 18	0.961 38 ± 0.016 12
Ridge	0.5	0.000 21 ± 0.000 02	0.011 50 ± 0.000 64	0.008 13 ± 0.000 32	0.961 34 ± 0.016 70
Lasso	17.9	0.000 21 ± 0.000 02	0.011 39 ± 0.000 81	0.008 04 ± 0.000 42	0.961 69 ± 0.016 42
Elastic net	19.0	0.000 21 ± 0.000 02	0.011 39 ± 0.000 81	0.008 05 ± 0.000 42	0.961 67 ± 0.016 43
K-neighbors	3.1	0.001 87 ± 0.000 24	0.040 11 ± 0.004 32	0.031 25 ± 0.002 99	0.703 39 ± 0.072 14
Decision tree	0.8	0.003 37 ± 0.000 53	0.052 52 ± 0.005 15	0.041 43 ± 0.003 63	0.442 86 ± 0.176 74
Random forest	7.6	0.003 04 ± 0.000 65	0.049 11 ± 0.007 06	0.038 14 ± 0.004 94	0.513 12 ± 0.150 13
Extra trees	4.7	0.002 56 ± 0.000 60	0.043 77 ± 0.006 95	0.033 89 ± 0.004 74	0.596 45 ± 0.128 33
XGBoost	5.6	0.000 62 ± 0.000 43	0.021 06 ± 0.009 16	0.013 67 ± 0.004 65	0.918 02 ± 0.029 44
MLP	775.1	0.000 82 ± 0.000 36	0.022 39 ± 0.004 56	0.019 50 ± 0.005 19	0.847 03 ± 0.111 87

Table C.17: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 16.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.000 86 ± 0.000 21	0.021 70 ± 0.002 27	0.012 86 ± 0.001 46	0.937 14 ± 0.012 65
Ridge	23.5	0.000 85 ± 0.000 21	0.021 72 ± 0.002 25	0.012 88 ± 0.001 45	0.937 31 ± 0.012 71
Lasso	21.5	0.000 85 ± 0.000 19	0.022 58 ± 0.001 94	0.013 86 ± 0.001 48	0.936 30 ± 0.013 42
Elastic net	23.0	0.000 85 ± 0.000 19	0.022 58 ± 0.001 94	0.013 86 ± 0.001 48	0.936 30 ± 0.013 42
K-neighbors	8.1	0.003 77 ± 0.001 17	0.057 94 ± 0.010 28	0.044 16 ± 0.009 38	0.687 10 ± 0.072 87
Decision tree	0.7	0.008 02 ± 0.002 31	0.086 52 ± 0.013 97	0.066 81 ± 0.011 53	0.301 68 ± 0.145 70
Random forest	3.4	0.006 20 ± 0.002 04	0.074 38 ± 0.014 47	0.057 86 ± 0.012 58	0.493 30 ± 0.125 20
Extra trees	6.0	0.005 48 ± 0.001 75	0.069 35 ± 0.013 42	0.053 76 ± 0.011 82	0.547 96 ± 0.107 37
XGBoost	153.5	0.001 10 ± 0.000 24	0.024 64 ± 0.002 91	0.015 28 ± 0.002 94	0.920 96 ± 0.019 04
MLP	543.1	0.000 99 ± 0.000 19	0.025 11 ± 0.002 47	0.015 87 ± 0.001 96	0.926 00 ± 0.004 40

Table C.18: A comparison of the 10 multi-output ML models. The computation time is given as seconds to train the model and the metrics include MSE, RMSE, MAE, and the coefficient of determination (R^2), which are averaged across the last 3-folds of the 5-fold cross validation. The results here are for house 17.

Model	Time	MSE	RMSE	MAE	R^2
Linear	0.1	0.001 52 ± 0.000 35	0.027 00 ± 0.003 71	0.016 34 ± 0.000 35	0.867 95 ± 0.047 55
Ridge	0.8	0.001 19 ± 0.000 10	0.023 42 ± 0.001 68	0.016 14 ± 0.000 72	0.903 82 ± 0.003 34
Lasso	17.9	0.001 20 ± 0.000 08	0.024 48 ± 0.001 27	0.017 29 ± 0.000 43	0.902 98 ± 0.003 32
Elastic net	17.7	0.001 20 ± 0.000 08	0.024 48 ± 0.001 27	0.017 29 ± 0.000 43	0.902 98 ± 0.003 32
K-neighbors	2.1	0.004 85 ± 0.000 19	0.066 97 ± 0.000 25	0.049 84 ± 0.001 56	0.519 71 ± 0.114 09
Decision tree	0.5	0.007 44 ± 0.000 36	0.083 63 ± 0.002 07	0.063 30 ± 0.000 73	0.263 78 ± 0.185 96
Random forest	1.0	0.007 18 ± 0.000 63	0.079 58 ± 0.004 50	0.059 74 ± 0.000 19	0.331 84 ± 0.070 62
Extra trees	4.8	0.006 36 ± 0.000 77	0.075 55 ± 0.004 73	0.056 53 ± 0.000 26	0.397 66 ± 0.080 88
XGBoost	8.3	0.001 94 ± 0.000 53	0.036 00 ± 0.007 77	0.023 28 ± 0.003 55	0.849 54 ± 0.028 55
MLP	80.9	0.001 33 ± 0.000 05	0.026 98 ± 0.001 24	0.019 80 ± 0.001 09	0.890 10 ± 0.007 63

BIBLIOGRAPHY

[Aba+15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. URL: https://www.tensorflow.org/ (visited on 09/10/2021) (cit. on p. 44).

[Afr+17] Abdul Afram, Farrokh Janabi-Sharifi, Alan S. Fung, and Kaamran Raahemifar. "Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system." In: *Energy and Buildings* 141 (2017), pp. 96–113. ISSN: 03787788. DOI: 10.1016/j.enbuild.2017.02.012 (cit. on p. 16).

[AAT19] Aguilera, Andersen, and Toftum. "Prediction of Indoor Air Temperature Using Weather Data and Simple Building Descriptors." In: *International Journal of Environmental Research and Public Health* 16 (22 Nov. 2019), p. 4349. ISSN: 1660-4601. DOI: 10.3390/ijerph16224349 (cit. on p. 16).

[AOSA20] Feras Al-Obeidat, Bruce Spencer, and Omar Alfandi. "Consistently accurate forecasts of temperature within buildings from sensor data using ridge and lasso regression." In: *Future Generation Computer Systems* 110 (2020), pp. 382–392. ISSN: 0167739X. DOI: 10.1016/j.future.2018.02.035 (cit. on p. 21).

[Ala+20] Sadi Alawadi, David Mera, Manuel Fernández-Delgado, Fahed Alkhabbas, Carl Magnus Olsson, and Paul Davidsson. "A comparison of machine learning algorithms for forecasting indoor temperature in smart buildings." In: *Energy Systems* (2020). ISSN: 18683975. DOI: 10.1007/s12667-020-00376-x (cit. on p. 24).

[Ang+19] R. A. Angelova, D. G. Markov, I. Simova, R. Velichkova, and P. Stankov. "Accumulation of metabolic carbon dioxide (CO2) in a vehicle cabin." In: *IOP Conference Series: Materials Science and Engineering* 664.1 (2019). ISSN: 1757899X. DOI: 10.1088/1757-899X/664/1/012010 (cit. on p. 74).

[Ash20] ANSI Ashrae. "Standard 55-2020: Thermal environmental conditions for human occupancy." In: *American Society of Heating, Refrigerating, and Air-Conditioning Engineers, Inc. Atlanta* (2020) (cit. on p. 12).

[AM08] Karl Johan Astrom and Richard M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. USA: Princeton University Press, 2008. ISBN: 0691135762 (cit. on p. 14).

[Att+19] Attoue, Shahrour, Mroueh, and Younes. "Determination of the Optimal Order of Grey-Box Models for Short-Time Prediction of Buildings' Thermal Behavior." In: *Buildings* 9.9 (2019), p. 198. ISSN: 2075-5309. DOI: 10.3390/buildings9090198 (cit. on p. 18).

[Aur19] Géron Aurélien. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. OReilly, 2019 (cit. on p. 44).

[Aus21] Richard Austin. *Top 20 global carmakers spend another $71.7 bn on R&D as electric vehicle rollout gathers pace*. 2021. URL: https://www.bdo.co.uk/en-gb/news/2021/top-20-global-carmakers-spend-another-71-7bn-on-r-and-d-as-electric-vehicle-rollout-gathers-pace (visited on 05/22/2022) (cit. on p. 2).

[BSC13] Sagar S. Badhiye, Nilesh U. Sambhe, and P. N. Chatur. "KNN Technique for Analysis and Prediction of Temperature and Humidity Data." In: *International Journal of Computer Applications* 61.14 (2013), pp. 7–13. ISSN: 09758887. DOI: 10.5120/9994-4847 (cit. on p. 22).

[Bah+22] Raihana Bahru, Mohd Faiz Muaz Ahmad Zamri, Abd Halim Shamsuddin, and Mohd Ambri Mohamed. "Simulation design for thermal model from various materials in electronic devices: A review." In: *Numerical Heat Transfer, Part A: Applications* 0.0 (2022), pp. 1–26. DOI: 10.1080/10407782.2022.2083842 (cit. on p. 15).

[Bas+19] Hector Bastida, Carlos E. Ugalde-Loo, Muditha Abeysekera, Meysam Qadrdan, and Jianzhong Wu. "Thermal Dynamic Modelling and Temperature Controller Design for a House." In: *Energy Procedia* 158 (2019), pp. 2800–2805. ISSN: 18766102. DOI: 10.1016/j.egypro.2019.02.041 (cit. on p. 16).

[BB12] Christoph Bergmeir and José M. Benítez. "On the use of cross-validation for time series predictor evaluation." In: *Information Sciences* 191 (2012), pp. 192–213. ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2011.12.028 (cit. on p. 31).

[BYC13] James Bergstra, Daniel Yamins, and David Cox. "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures." In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 1. Atlanta, Georgia, USA: PMLR, 2013, pp. 115–123. URL: https://proceedings.mlr.press/v28/bergstra13.html (cit. on p. 39).

[BMN06] Rick Bitter, Taqi Mohiuddin, and Matt Nawrocki. *LabVIEW: Advanced programming techniques*. Crc Press, 2006 (cit. on p. 14).

[BJ90] George Edward Pelham Box and Gwilym Jenkins. *Time Series Analysis, Forecasting and Control*. USA: Holden-Day, Inc., 1990. ISBN: 0816211043 (cit. on p. 29).

[BR20] James Brusey and Matteo Rostagno. *Efficient cabin model for simulating thermal and acoustic behaviour of car cabins*. Deliverable D1.5. DOMUS, May 22, 2020. URL: https://www.domus-project.eu/wp-content/uploads/2020/06/D1.5-PubSum.pdf (visited on 10/06/2021) (cit. on pp. 9, 71).

[BE&IS22] Department for Business Energy & Industiral Strategy. *Final UK greenhouse gas emissions national statistics: 1990 to 2020*. Tech. rep. UK Government, 2022. URL: https://www.gov.uk/government/statistics/final-uk-greenhouse-gas-emissions-national-statistics-1990-to-2020 (visited on 12/08/2022) (cit. on pp. 1–3).

[Cam20] Peter Campbell. *UK set to ban sale of new petrol and diesel cars from 2030*. 2020. URL: https://www.ft.com/content/5e9af60b-774b-4a72-8d06-d34b5192ffb4 (visited on 01/21/2021) (cit. on p. 2).

[CG16] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2016, pp. 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785 (cit. on pp. 24, 43).

[Cho+15] Francois Chollet et al. *Keras*. 2015. URL: https://github.com/fchollet/keras (visited on 10/15/2020) (cit. on p. 44).

[Cia19] Carlo Ciacci. "1D Modelling and Analysis of Thermal Conditioning Systems for Electric Vehicles." Masters. University of Windsor, 2019 (cit. on p. 71).

[CC] United Nations / Framework Convention on Climate Change. *Paris Agreement 2015*. United Nations. URL: https://treaties.un.org/Pages/showDetails.aspx?objid=0800000280458f37&clang=_en (visited on 12/13/2021) (cit. on p. 1).

[Cui+18] B Cui, J Dong, J Munk, N Mao, and T Kuruganti. "A simplified regression building thermal model of detached two-floor house in U.S. for virtual energy storage control." In: *17th International Refrigeration and Air Conditioning Conference at Purdue* July (2018) (cit. on pp. 17, 20).

[Dad+11] I. R. Dadour, I. Almanjahie, N. D. Fowkes, G. Keady, and K. Vijayan. "Temperature variations in a parked vehicle." In: *Forensic Science International* 207.1-3 (2011), pp. 205–211. ISSN: 03790738. DOI: 10.1016/j.forsciint.2010.10.009 (cit. on p. 30).

[DDM19] Santanu Prasad Datta, Prasanta Kumar Das, and Siddhartha Mukhopadhyay. "An optimized ANN for the performance prediction of an automotive air conditioning system." In: *Science and Technology for the Built Environment* 25.3 (2019), pp. 282–296. ISSN: 2374474X. DOI: 10.1080/23744731.2018.1526014 (cit. on p. 15).

[Doc20] Simulink Documentation. *Simulation and Model-Based Design*. 2020. URL: https://www.mathworks.com/products/simulink.html (visited on 10/03/2022) (cit. on p. 14).

[Doo21] Atip Doolgindachbaporn. "Data Driven Transformer Thermal Modelling for Identification of Thermal Issues." PhD. University of Southampton, 2021, p. 310 (cit. on p. 30).

[DM19] Aisling Doyle and Tariq Muneer. "Energy consumption and modelling of the climate control system in the electric vehicle." In: *Energy Exploration and Exploitation* 37.1 (2019), pp. 519–543. DOI: 10.1177/0144598718806458 (cit. on p. 15).

[DS98] Norman R. Draper and Harry Smith. *Applied Regression Analysis*. Wiley Series in Probability and Statistics. Wiley, 1998. ISBN: 9780471170822. DOI: 10.1002/9781118625590 (cit. on p. 19).

[Eng+19] Peter Engel, Sebastian Meise, Andreas Rausch, and Wilhelm Tegethoff. "Modeling of Automotive HVAC Systems Using Long Short-Term Memory Networks." In: *ADAPTIVE 2019: The Eleventh International Conference on Adaptive and Self-Adaptive Systems and Applications*. May. 2019, pp. 48–55. ISBN: 978-1-61208-706-1 (cit. on p. 38).

[Fan+70] Poul O Fanger et al. "Thermal comfort. Analysis and applications in environmental engineering." In: *Thermal comfort. Analysis and applications in environmental engineering*. (1970) (cit. on p. 12).

[FR00] Robert B. Farrington and John P. Rugh. "Impact of Vehicle Air-Conditioning on Fuel Economy, Tailpipe Emissions, and Electric Vehicle Range." In: *Earth Technologies Forum*. Washington DC, 2000, pp. 1–10. URL: http://www.nrel.gov/docs/fy00osti/28960.pdf (cit. on p. 2).

[Fir+15] Steven Firth, Tom Kane, Vanda Dimitriou, Tarek Hassan, Farid Fouchal, Michael Coleman, and Lynda Webb. *REFIT Smart Home dataset*. Tech. rep. 2015. URL: https://dx.doi.org/10.17028/rd.lboro.2070091 (cit. on p. 75).

[Fir+17] Steven Firth, Tom Kane, Vanda Dimitriou, Tarek Hassan, Farid Fouchal, Michael Coleman, and Lynda Webb. "REFIT Smart Home dataset." In: (June 2017). DOI: 10.17028/rd.lboro.2070091.v1. URL: https://repository.lboro.ac.uk/articles/dataset/REFIT_Smart_Home_dataset/2070091 (cit. on pp. 9, 75).

[Fri01] Jerome H. Friedman. "Greedy function approximation: A gradient boosting machine." In: *The Annals of Statistics* 29.5 (2001). ISSN: 0090-5364. DOI: 10.1214/aos/1013203451 (cit. on p. 24).

[Gov08] UK Government. *Climate Change Act 2008, c. 27*. 2008. URL: https://www.legislation.gov.uk/ukpga/2008/27/contents (visited on 03/10/2021) (cit. on p. 1).

[Gri19] Olivier Grisel. *Group aware Time-based cross validation*. 2019. URL: https://github.com/scikit-learn/scikit-learn/issues/14257 (visited on 07/01/2021) (cit. on p. 45).

[GML18] Matej Gustin, Robert S. McLeod, and Kevin J. Lomas. "Forecasting indoor temperatures during heatwaves using time series models." In: *Building and Environment* 143 (2018), pp. 727–739. ISSN: 03601323. DOI: 10.1016/j.buildenv.2018.07.045. URL: https://linkinghub.elsevier.com/retrieve/pii/S0360132318304530 (cit. on p. 30).

[HS18] David Ha and Jürgen Schmidhuber. "Recurrent World Models Facilitate Policy Evolution." In: *Advances in Neural Information Processing Systems 31*. https://worldmodels.github.io. Curran Associates, Inc., 2018, pp. 2451–2463. URL: https://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution (cit. on p. 64).

[Han+21] Marwah Sattar Hanoon, Ali Najah Ahmed, Nur'atiah Zaini, Arif Razzaq, Pavitra Kumar, Mohsen Sherif, Ahmed Sefelnasr, and Ahmed El-Shafie. "Developing machine learning algorithms for meteorological temperature and humidity forecasting at Terengganu state in Malaysia." In: *Scientific Reports* 11.1 (2021). ISSN: 20452322. DOI: 10.1038/s41598-021-96872-w (cit. on p. 29).

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001 (cit. on p. 20).

[HBG15] Diana Hintea, James Brusey, and Elena Gaura. "A Study on Several Machine Learning Methods for Estimating Cabin Occupant Equivalent Temperature." In: *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics*. Vol. 1. January. SCITEPRESS - Science, and Technology Publications, 2015, pp. 629–634. ISBN: 978-989-758-122-9. DOI: 10.5220/0005573606290634 (cit. on p. 20).

[HE06] Murat Hosoz and Hüseyin Metin Ertunç. "Artificial neural network analysis of an automobile air conditioning system." In: *Energy Conversion and Management* 47.11-12 (2006), pp. 1574–1587. ISSN: 01968904. DOI: 10.1016/j.enconman.2005.08.008 (cit. on p. 15).

[IEA21] IEA. *Global EV Outlook 2021*. Tech. rep. Paris, 2021. URL: https://www.iea.org/reports/global-ev-outlook-2021 (visited on 07/10/2022) (cit. on p. 2).

[Isoa] *ISO 14505-4: Ergonomics of the thermal environment - Evaluation of thermal environments in vehicles – Part 4: Determination of the equivalent temperature by means of a numerical manikin*. Standard. Geneva, CH: International Organization for Standardization, 2021 (cit. on pp. 12, 53).

[Isob] *ISO 7730: Ergonomics of the thermal environment - Analytical determination and interpretation of thermal comfort using calculation of the PMV and PPD indices and local thermal comfort criteria*. Standard. Geneva, CH: International Organization for Standardization, 2005 (cit. on pp. 12, 53).

[Jai+21] Aditi Jaiswal, Zeeshan Amjad, Suraj Jha, Nikhil Sahni, Vaisakh Punnekkattu Chirayil S B, and Renju C Nair. "Accurate Device Temperature Forecasting using Recurrent Neural Network for Smartphone Thermal Management." In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8. DOI: 10.1109/IJCNN52387.2021.9533732 (cit. on p. 15).

[Jes+22] Brandi Jess, James Brusey, Matteo Maria Rostagno, Alberto Maria Merlo, Elena Gaura, and Kojo Sarfo Gyamfi. "Fast, Detailed, Accurate Simulation of a Thermal Car-Cabin Using Machine-Learning." In: *Frontiers in Mechanical Engineering* 8 (2022). ISSN: 2297-3079. DOI: 10.3389/fmech.2022.753169. URL: https://www.frontiersin.org/articles/10.3389/fmech.2022.753169/full (cit. on p. 7).

[KFK22] Abraham Kaligambe, Goro Fujita, and Tagami Keisuke. "Estimation of Unmeasured Room Temperature, Relative Humidity, and CO2 Concentrations for a Smart Building Using Machine Learning and Exploratory Data Analysis." In: *Energies* 15.12 (2022), p. 4213. DOI: 10.3390/en15124213 (cit. on p. 25).

[KPD99] Soteris A. Kalogirou, Sofia Panteliou, and Argiris Dentsoras. "Modeling of solar domestic water heating systems using Artificial Neural Networks." In: *Solar Energy* 65.6 (1999), pp. 335–342. ISSN: 0038092X. DOI: 10.1016/S0038-092X(99)00013-4 (cit. on p. 15).

[Kam+13] Haslinda Mohamed Kamar, Robiah Ahmad, N. B. Kamsah, and Ahmad Faiz Mohamad Mustafa. "Artificial neural networks for automotive air-conditioning systems performance prediction." In: *Applied Thermal Engineering* 50.1 (2013), pp. 63–70. ISSN: 13594311. DOI: 10.1016/j.applthermaleng.2012.05.032 (cit. on p. 15).

[KB14a] Kiran R. Kambly and Thomas H. Bradley. "Estimating the HVAC energy consumption of plug-in electric vehicles." In: *Journal of Power Sources* 259 (2014), pp. 117–124. ISSN: 03787753. DOI: 10.1016/j.jpowsour.2014.02.033 (cit. on p. 15).

[KMS15] Zahra Karevan, Siamak Mehrkanoon, and Johan A.K. Suykens. "Black-box modeling for temperature prediction in weather forecasting." In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–8. ISBN: 978-1-4799-1960-4. DOI: 10.1109/IJCNN.2015.7280671 (cit. on p. 21).

[Kat+18] Katarina Katić, Rongling Li, Jacob Verhaart, and Wim Zeiler. "Neural network based predictive control of personalized heating systems." In: *Energy & Buildings* 174 (2018), pp. 199–213. ISSN: 03787788. DOI: 10.1016/j.enbuild.2018.06.033 (cit. on p. 15).

[KK20] Saboora Khatoon and Man Hoe Kim. "Thermal comfort in the passenger compartment using a 3-D numerical analysis and comparison with Fanger's comfort models." In: *Energies* 13.3 (2020). ISSN: 19961073. DOI: 10.3390/en13030690 (cit. on p. 15).

[KB14b] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. URL: http://arxiv.org/abs/1412.6980 (visited on 08/10/2021) (cit. on p. 44).

[KWB21] Wilhelm Kirchgássner, Oliver Wallscheid, and Joachim Böcker. "Thermal Neural Networks: Lumped-Parameter Thermal Modeling With State-Space Machine Learning."

In: (Mar. 2021). URL: http://arxiv.org/abs/2103.16323 (cit. on p. 17).

[Koc+09] Dragi Kocev, Sašo Džeroski, Matt D. White, Graeme R. Newell, and Peter Griffioen. "Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition." In: *Ecological Modelling* 220 (8 Apr. 2009), pp. 1159–1168. ISSN: 03043800. DOI: 10.1016/j.ecolmodel.2009.01.037 (cit. on p. 27).

[KJ13] Max Kuhn and Kjell Johnson. *Applied Predictive Modeling*. New York, NY: Springer New York, 2013. ISBN: 978-1-4614-6848-6. DOI: 10.1007/978-1-4614-6849-3 (cit. on p. 23).

[KXZ14] Andrew Kusiak, Guanglin Xu, and Zijun Zhang. "Minimization of energy consumption in HVAC systems with data-driven models and an interior-point method." In: *Energy Conversion and Management* 85 (2014), pp. 146–153. ISSN: 01968904. DOI: 10.1016/j.enconman.2014.05.053 (cit. on p. 16).

[Kut+04] Michael H Kutner, Christopher J Nachtsheim, John Neter, and William Li. *Applied linear statistical models*. 5th ed. New York, NY: McGraw-Hill Professional, Sept. 2004. ISBN: 007310874X (cit. on p. 36).

[Laj17] Antti Lajunen. "Energy Efficiency and Performance of Cabin Thermal Management in Electric Vehicles." In: *SAE Technical Papers* 2017-March.March (2017). ISSN: 01487191. DOI: 10.4271/2017-01-0192 (cit. on p. 15).

[LM01] Thomas W. Jr. Leland and G.A. (ed.) Mansoori. *Basic Principles of Classical and Statistical Thermodynamics*. Tech. rep. Chicago, IL, 2001, pp. 1–35 (cit. on p. 34).

[Lia+20] Shengli Liao, Zhanwei Liu, Benxi Liu, Chuntian Cheng, Xinfeng Jin, and Zhipeng Zhao. "Multistep-ahead daily inflow forecasting using the ERA-Interim reanalysis data set based on gradient-boosting regression trees." In: *Hydrology and Earth System Sciences* 24.5 (2020), pp. 2343–2363. ISSN: 1607-7938. DOI: 10.5194/hess-24-2343-2020. URL: https://hess.copernicus.org/articles/24/2343/2020/ (cit. on p. 25).

[Lom+18] K. J. Lomas, S. Oliveira, P. Warren, V. J. Haines, T. Chatterton, A. Beizaee, E. Prestwood, and B. Gething. "Do domestic heating controls save energy? A review of the evidence." In: *Renewable and Sustainable Energy Reviews* 93 (Oct. 2018), pp. 52–75. ISSN: 18790690. DOI: 10.1016/j.rser.2018.05.002 (cit. on p. 14).

[MBC19] Jaume Manero, Javier Béjar, and Ulises Cortés. "Deep Learning is blowing in the wind. Deep models applied to wind prediction at turbine level." In: *Journal of Physics: Conference Series* 1222.1 (2019). ISSN: 17426596. DOI: 10.1088/1742-6596/1222/1/012037 (cit. on p. 30).

[Mar+14] David Marcos, Francisco J. Pino, Carlos Bordons, and José J. Guerra. "The development and validation of a thermal model for the cabin of a vehicle." In: *Applied Thermal Engineering* 66.1-2 (2014), pp. 646–656. ISSN: 13594311. DOI: 10.1016/j.applthermaleng.2014.02.054 (cit. on p. 15).

[Mas+99] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. "Boosting Algorithms as Gradient Descent." In: *Advances in Neural Information Processing Systems*. Ed. by S. Solla, T. Leen, and K. Müller. Vol. 12. MIT Press, 1999. URL: https://proceedings.neurips.cc/paper/1999/file/96a93ba89a5b5c6c226e49b88973f46e-Paper.pdf (cit. on p. 24).

[MB08] José Maria P. Menezes and Guilherme A. Barreto. "Long-term time series prediction with the NARX network: An empirical evaluation." In: *Neurocomputing* 71.16-18 (2008), pp. 3335–3343. ISSN: 09252312. DOI: 10.1016/j.neucom.2008.01.030 (cit. on p. 46).

[Mir+18] Elham Mirkoohi, Jinqiang Ning, Peter Bocchini, Omar Fergani, and Kuo-ning Chiang. "Thermal Modeling of Temperature Distribution in Metal Additive Manufacturing Considering Effects of Build Layers , Latent Heat , and Temperature-Sensitivity of Material Properties." In: *Journal of Manufacturing and Materials Processing* 2.63 (2018). DOI: 10.3390/jmmp2030063 (cit. on p. 15).

[MN20] Pooya Mirzabeygi and Shankar Natarajan. "Artificial Neural Network Based Predictive Approach in Vehicle Thermal Systems Applications." In: *SAE International Journal of Advances and Current Practices in Mobility* 2.6 (2020), pp. 3093–3102. ISSN: 01487191. DOI: 10.4271/2020-01-0148 (cit. on p. 3).

[Modoo] Modelica Association. *Modelica - A Unified Object-Oriented Language for Physical Systems Modeling. Tutorial*. 2000. URL: http://www.modelica.org (visited on 11/05/2022) (cit. on p. 14).

[Mol+14] Geert Molenberghs, Garrett Fitzmaurice, Michael G. Kenward, Anastasios Tsiatis, and Geert Verbeke, eds. *Handbook of Missing Data Methodology*. Chapman and Hall/CRC, 2014. ISBN: 9781439854624. DOI: 10.1201/b17622 (cit. on p. 36).

[Ng+14a] Boon Chiang Ng, Intan Zaurah Mat Darus, Hishamuddin Jamaluddin, and Haslinda Mohamed Kamar. "Application of adaptive neural predictive control for an automotive air conditioning system." In: *Applied Thermal Engineering* 73.1 (2014), pp. 1244–1254. ISSN: 13594311. DOI: 10.1016/j.applthermaleng.2014.08.044 (cit. on p. 38).

[Ng+14b] Boon Chiang Ng, Intan Zaurah Mat Darus, Hishamuddin Jamaluddin, and Haslinda Mohamed Kamar. "Dynamic modelling of an automotive variable speed air conditioning system using nonlinear autoregressive exogenous neural networks." In: *Applied Thermal Engineering* 73.1 (2014), pp. 1255–1269. ISSN: 13594311. DOI: 10.1016/j.applthermaleng.2014.08.043 (cit. on pp. 15, 30).

[Nil04] Håkan Nilsson. "Comfort Climate Evaluation with Thermal Manikin Methods and Computer Simulation Models." PhD thesis. Apr. 2004 (cit. on p. 12).

[Now14] Thomas Nowotny. "Two Challenges of Correct Validation in Pattern Recognition." In: *Frontiers in Robotics and AI* 1.September (2014), pp. 1–6. ISSN: 2296-9144. DOI: 10.3389/frobt.2014.00005 (cit. on p. 30).

[Ora+18] Juraj Oravec, Monika Bakošová, Marian Trafczynski, Anna Vasičkaninová, Alajos Mészáros, and Mariusz Markowski. "Robust model predictive control and PID control of shell-and-tube heat exchangers." In: *Energy* 159 (Sept. 2018), pp. 1–10. ISSN: 03605442. DOI: 10.1016/j.energy.2018.06.106 (cit. on p. 14).

[PM+07] Harish J. Palanthandalam-Madapusi, Biju Edamana, Dennis S. Bernstein, Ward Manchester, and Aaron J. Ridley. "NARMAX identification for space weather prediction using polynomial radial basis functions." In: *2007 46th IEEE Conference on Decision and Control*. IEEE, 2007, pp. 3622–3627. ISBN: 978-1-4244-1497-0. DOI: 10.1109/CDC.2007.4434787 (cit. on p. 30).

[PP14] Foram S. Panchal and Mahesh Panchal. "Review on Methods of Selecting Number of Hidden Nodes in Artificial Neural Network." In: *International Journal of Computer Science and Mobile Computing* 3 (2014). ISSN: 2320-088X (cit. on p. 25).

[PW22] Wojciech Panek and Tomasz Włodek. "Natural Gas Consumption Forecasting Based on the Variability of External Meteorological Factors Using Machine Learning Algorithms." In: *Energies* 15 (1 Jan. 2022). ISSN: 19961073. DOI: 10.3390/en15010348 (cit. on p. 18).

[Ped+11] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830. DOI: 10.5555/1953048.2078195 (cit. on pp. 37, 39–42, 45).

[RHNv15] Anamarija Rabi, Marijana Hadzima-Nyarko, and Marija Šperac. "Modelling river temperature from air temperature: case of the River Drava (Croatia)." In: *Hydrological Sciences Journal* 60 (9 Sept. 2015), pp. 1490–1507. ISSN: 0262-6667. DOI: 10.1080/02626667.2014.914215 (cit. on pp. 15, 26).

[Ram+21] Lara Ramadan, Isam Shahrour, Hussein Mroueh, and Fadi Hage Chehade. "Use of machine learning methods for indoor temperature forecasting." In: *Future Internet* 13.10 (2021), pp. 1–18. ISSN: 19995903. DOI: 10.3390/fi13100242 (cit. on p. 24).

[Ram+20] Naren Srivaths Raman, Adithya M. Devraj, Prabir Barooah, and Sean P. Meyn. "Reinforcement Learning for Control of Building HVAC Systems." In: *2020 American Control Conference (ACC)*. 2020, pp. 2326–2332. DOI: 10.23919/ACC45564.2020.9147629 (cit. on p. 14).

[RMD20] J.B. Rawlings, D.Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design 2nd Edition*. Nob Hill Publishing, 2020. ISBN: 9780975937754 (cit. on p. 14).

[RH11] Jakob Rehrl and Martin Horn. "Temperature control for HVAC systems based on exact linearization and model predictive control." In: *2011 IEEE International Conference on Control Applications (CCA)*. IEEE, 2011, pp. 1119–1124. ISBN: 978-1-4577-1062-9. DOI: 10.1109/CCA.2011.6044437 (cit. on p. 14).

[RPM15] Sergey V. Reznik, Pavel V. Prosuntsov, and Konstantin V. Mikhaylovskiy. "Prediction of Thermophysical and Thermomechanical Characteristics of Porous Carbon - Ceramic Composite Materials of the Heat Shield of Aerospace." In: *Journal of Engineering Physics and Thermophysics* 88.3 (2015). DOI: 10.1007/s10891-015-1227-1 (cit. on p. 15).

[Sai+13] M Saifizi, M Z Ab Muin, Sazali Yaacob, and M S Mohamad. "Comparison of ARX and ARMAX models for thermoelectric refrigerator." In: *IOP Conference Series: Materials Science and Engineering* 50 (2013), p. 012032. ISSN: 1757-8981. DOI: 10.1088/1757-899X/50/1/012032 (cit. on p. 30).

[SW05] William R. Santee and Robert F. Wallace. "Comparison of weather service heat indices using a thermal model." In: *Journal of Thermal Biology* 30.1 (2005), pp. 65–72. ISSN:

0306-4565. DOI: 10.1016/j.jtherbio.2004.07.003 (cit. on p. 15).

[Sha+21] Arya Shahdi, Seho Lee, Anuj Karpatne, and Bahareh Nojabaei. "Exploratory analysis of machine learning methods in predicting subsurface temperature and geothermal gradient of Northeastern United States." In: *Geothermal Energy* 9 (1 Dec. 2021). ISSN: 21959706. DOI: 10.1186/s40517-021-00200-4 (cit. on p. 18).

[Sie18] Siemens. *Simcenter Amesim 17*. 2018. URL: https://www.plm.automation.siemens.com/global/en/products/simcenter/simcenter-amesim.html (visited on 12/27/2022) (cit. on p. 71).

[SA18] Samrendra Singh and Hesam Abbassi. "1D/3D transient HVAC thermal modeling of an off-highway machinery cabin using CFD-ANN hybrid method." In: *Applied Thermal Engineering* 135.January (2018), pp. 406–417. ISSN: 13594311. DOI: 10.1016/j.applthermaleng.2018.02.054 (cit. on p. 17).

[Sri+20] Ike Sri Rahayu, Esmeralda C. Djamal, Ridwan Ilyas, and Abdul Talib Bon. "Daily temperature prediction using recurrent neural networks and long-short term memory." In: *Proceedings of the International Conference on Industrial Engineering and Operations Management* August (2020), pp. 2700–2709. ISSN: 21698767 (cit. on p. 25).

[Sri+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. DOI: 10.5555/2627435.2670313 (cit. on p. 44).

[SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018. ISBN: 0262039249 (cit. on p. 14).

[Tra20] Department for Transport. *2019 National Travel Survey Factsheets*. Tech. rep. UK Government, 2020. URL: https://www.gov.uk/government/statistics/national-travel-survey-2019 (visited on 05/28/2022) (cit. on p. 3).

[Tra21] Department for Transport. *Transport and environment statistics: Autumn 2021*. Tech. rep. UK Government, 2021. URL: https://www.gov.uk/government/statistics/transport-and-environment-statistics-autumn-2021/transport-and-environment-statistics-autumn-2021 (visited on 05/28/2022) (cit. on p. 2).

[VS06] Sudhir Varma and Richard Simon. "Bias in error estimation when using cross-validation for model selection." In: *BMC Bioinformatics* 7 (2006). DOI: 10.1186/1471-2105-7-91 (cit. on p. 31).

[War+20] Alok Warey, Shailendra Kaushik, Bahram Khalighi, Michael Cruse, and Ganesh Venkatesan. "Data-driven prediction of vehicle cabin thermal comfort: using machine learning and high-fidelity simulation results." In: *International Journal of Heat and Mass Transfer* 148 (2020), p. 119083. ISSN: 00179310. DOI: 10.1016/j.ijheatmasstransfer.2019.119083 (cit. on p. 18).

[Xu+20] Donna Xu, Yaxin Shi, Ivor W. Tsang, Yew Soon Ong, Chen Gong, and Xiaobo Shen. "Survey on Multi-Output Learning." In: *IEEE Transactions on Neural Networks and Learning Systems* 31.7 (2020), pp. 2409–2429. ISSN: 21622388. DOI: 10.1109/TNNLS.2019.2945133. arXiv: 1901.00248 (cit. on p. 26).

[Zha+18] Kaicheng Zhang, Akhil Guliani, Seda Ogrenci-Memik, Gokhan Memik, Kazutomo Yoshii, Rajesh Sankaran, and Pete Beckman. "Machine Learning-Based Temperature Prediction for Runtime Thermal Management Across System Components." In: *IEEE Transactions on Parallel and Distributed Systems* 29.2 (2018), pp. 405–419. ISSN: 1045-9219. DOI: 10.1109/TPDS.2017.2732951 (cit. on p. 15).

[ZH05] Hui Zou and Trevor Hastie. "Regularization and variable selection via the elastic net." In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320. ISSN: 1369-7412. DOI: 10.1111/j.1467-9868.2005.00503.x (cit. on p. 21).

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

https://bitbucket.org/amiede/classicthesis/

Final Version as of October 15, 2023 (v1.0).