



Intisar Omar

MACHINE LEARNING (ML) APPROACHES TO MODEL
INTERDEPENDENCIES BETWEEN DYNAMIC LOADS AND
CRACK PROPAGATION

SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING, CRANFIELD UNIVERSITY
Mechanical Engineering

A Degree of Doctor of Philosophy in the Artificial Intelligence for
Structural Health Monitoring

Academic Year: 2020 - 2023

Supervisor: Muhammad Khan
Associate Supervisor: Andrew Starr
September 2023



SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING, CRANFIELD UNIVERSITY
Mechanical Engineering

PhD

Academic Year 2020 - 2023

Intisar Omar

MACHINE LEARNING (ML) APPROACHES TO MODEL
INTERDEPENDENCIES BETWEEN DYNAMIC LOADS AND
CRACK PROPAGATION

Supervisor: Muhammad Khan
Associate Supervisor: Andrew Starr
September 2023

This thesis is submitted in partial fulfilment of the requirements for
the degree of PhD.

© Cranfield University 2023. All rights reserved. No part of this
publication may be reproduced without the written permission of the
copyright owner.

Academic integrity declaration

I declare that:

- the thesis submitted has been written by me alone.
- the thesis submitted has not been previously submitted to this university or any other.
- that all content, including primary and/or secondary data, is true to the best of my knowledge.
- that all quotations and references have been duly acknowledged according to the requirements of academic research.

I understand that to knowingly submit work in violation of the above statement will be considered by examiners as academic misconduct.

Abstract

The application of machine learning in structural health and crack prediction is of paramount importance, as it offers the potential to enhance the accuracy, efficiency, and reliability of detecting and predicting damage in various materials and structures. This research presents an in-depth exploration of machine learning (ML) applications in the field of Structural Health Monitoring (SHM) across various materials, including composites, metals, and polymers. The study identifies the current challenges in implementing ML in SHM, such as data sparsity, interpretability of ML models, overfitting, and the absence of general guidelines for ML model selection.

The research analyses the dynamic response data of different materials and establishes significant crack depth predictors for materials such as aluminum, concrete, and 3D-printed Acrylonitrile Butadiene Styrene (ABS). It further investigates and validates selected ML models to predict crack depth in different materials. The models' performance is evaluated using Mean Squared Error (MSE) on both training and test sets, demonstrating their ability to capture meaningful patterns within the data and make reasonably accurate predictions. A significant contribution of this study is the proposal of an automated model utilizing the H2O library for crack propagation prediction in ABS materials. This model demonstrates the potential of automation in SHM, offering substantial benefits for structural integrity assessment, maintenance strategies, and materials design in various industries. This research concludes with recommendations for future research, including the exploration of advanced ML algorithms, investigation of additional predictive features, and evaluation of the models in different real-world scenarios.

Keywords:

Machine learning, Structural health monitoring and assessment, ABS, Aluminum, Concrete, Automated Machine learning, H2O library, Python.

Acknowledgements

With a sorrowful heart, I dedicate this endeavour to the memory of my dear brother, who departed from this world on the 18th of July 2023. As I present my research, I am reminded of his unwavering encouragement and his belief in my abilities. Though he is no longer with us, his spirit continues to inspire me to strive for excellence.

I would like to express my sincere gratitude to my supervisors, Dr. Muhammad Khan, and Prof. Andrew Starr, for their invaluable guidance, unwavering support, and expert advice throughout the course of my Ph.D. research. Their exceptional knowledge, dedication, and mentorship have been instrumental in shaping the direction and success of this thesis. I am truly fortunate to have had the opportunity to work under their supervision.

I am also deeply indebted to my family & husband for their unwavering support, encouragement, and belief in my abilities. Their love, patience, and understanding have been the pillars of my strength throughout my academic pursuits. I am grateful for their sacrifices, understanding, and continuous motivation.

Finally, I would like to express my gratitude to all the individuals, institutions, and organizations that have contributed to my research in any capacity. Your assistance, participation, and resources have been invaluable and have significantly enhanced the quality and impact of this work.

To all those who have supported me directly or indirectly, I offer my deepest appreciation. Your belief in my abilities and your unwavering support have been integral to the completion of this thesis. Thank you for being a part of this incredible journey.

Intisar Omar

LIST OF PUBLICATION

The key contributions and publications from this thesis are given below.

1. Omar I, Khan M & Starr A (2022) Compatibility and challenges in machine learning approach for structural crack assessment, Structural Health Monitoring, 21 (5) 2481-2502.
2. Omar I, Khan M & Starr A (2023) Suitability analysis of machine learning algorithms for crack growth prediction based on dynamic response data, Sensors, 23 (3) Article No. 1074.
3. Omar I, Khan M & Starr A (2023) Comparative analysis of machine learning models for predicting crack propagation under coupled load and temperature, Applied Sciences, 13 (12) Article No. 7212.
4. Omar I, Khan M, Starr A, & Abou Rok Ba K (2023). Automated Prediction of Crack Propagation Using H2O AutoML. Sensors.
5. Jiang J, Omar I, & Khan M (2023). Predicting Wear Damage in Moving Mechanical Contacts: Comparative Analysis of Regression Algorithms and Feature Selection Techniques. ISA Transactions. (Waiting for the publication).
6. Presented conference paper titled 'A critical overview of Machine learning approaches to structural health monitoring and assessment: smart composite' to CM 2021.
7. Omar I, Khan M & Starr A (2023). Prediction Methods to Model Interdependencies Between Crack Propagation and Dynamic Response. The 29th International Congress on Sound and Vibration (ICSV29).

Table of Contents

Academic integrity declaration.....	i
Abstract	ii
Acknowledgements	iii
LIST OF PUBLICATION	iv
Table of Contents	v
List of Figures.....	viii
List of Tables	x
List of Equations.....	xi
1 INTRODUCTION.....	12
1.1 Background.....	12
1.2 Motivation	12
1.3 Aim and Objectives	14
1.4 Thesis Structure.....	15
2 LITERATURE REVIEW	17
2.1 Introduction	17
2.2 Structural Health Monitoring Approaches	18
2.2.1 Inverse Model-Driven Approaches	18
2.2.2 Forward Model-Driven Approaches.....	21
2.2.3 Data-based Approaches.....	23
2.3 Machine Learning Applications in Crack Mechanics.....	26
2.4 The Importance of Feature Engineering and Tools.....	29
2.5 Machine Learning for Crack Assessment in Smart Structures.....	32
2.5.1 Composite Structure.....	33
2.5.2 Metal Structure	35
2.5.3 Polymer Structure	37
2.6 Common challenges of machine learning	39
2.6.1 Availability and Quality of Data.....	39
2.6.2 Overfitting in Machine Learning Models	40
2.6.3 Interpretability of ML Models	41
2.6.4 Selection of Machine Learning Methods	42
2.7 Summary	44
3 Suitability analysis of Machine learning algorithms for crack growth prediction.....	46
3.1 Introduction	46
3.2 Support Vector Regression.....	48
3.3 Random Forest	51
3.4 Gradient Boosting Regression	54
3.5 Ridge Regression	58
3.6 Lasso Regression	60
3.7 Neural networks.....	61

3.8 Summary	69
4 Data Collection and Preprocessing	70
4.1 Introduction	70
4.2 Experimental scheme and setup.....	70
4.2.1 Aluminum	70
4.2.2 ABS.....	71
4.2.3 Concrete.....	73
4.3 Signal processing and Data collection	74
4.4 Data Preparation for Machine learning models.....	76
4.5 Training and validation Datasets Division	81
4.6 Model Training and Evaluation	83
4.7 Summary	84
5 Results and Discussions: Results and Discussions: Application of the proposed ML Models.....	85
5.1 Introduction	85
5.2 Machine learning Models	85
5.3 Application of the proposed ML Models Results	87
5.3.1 Training and Validation with all Features.....	88
ABS Models:	88
Aluminum Models:.....	88
Concrete Models:	91
5.3.1 Training and Validation with Selected Features	92
5.4 Discussion	96
5.5 Summary	99
6 Automated Prediction of Crack Propagation Using H2O AutoML: Towards Enhanced Structural Health Monitoring.....	100
6.1 Introduction	100
6.2 Previous Academic Work.....	101
6.3 Methodology	103
6.3.1 Dataset Description and Preprocessing:	103
6.3.2 Feature Engineering and Model Selection and Configuration:	104
6.3.3 Model Training and Evaluation:.....	105
6.4 Results and Discussion.....	106
6.4.1 Insights into Crack Propagation Behavior in ABS Materials:	106
6.4.2 AutoML Models Selection.....	107
6.4.3 Hyperparameter tuning and training of AutoML model.....	108
6.4.4 Performance Evaluation of the Selected Automated Model:	110
6.5 Comparison with Machine learning Approaches:.....	113
6.6 Advantages of the Automated Model:	114
6.7 Practical Implications:	115
6.8 Limitations and Future Directions:	115
6.9 Summary	116

7 Conclusions.....	118
7.1 Achievement of Objectives	118
7.2 Novel Contributions.....	119
7.2.1 Enhancing Structural Health Monitoring: Comprehensive Analysis of Diverse Materials and Machine Learning Models for Accurate Crack Propagation Prediction.....	119
7.2.2 Suitability Analysis of Machine Learning Algorithms for Crack Growth Prediction Based on Dynamic Response Data	119
7.2.3 Automated Prediction of Crack Propagation Using H2O AutoML: Towards Enhanced Structural Health Monitoring	120
7.3 Future Research Directions	120
REFERENCES.....	122
Appendices.....	158
Appendix A Ethical approval letter	158
Appendix B Python code of machine learning models.....	159
Appendix C Collected data of ABS to train and validate selected models. .	165
Appendix D Collected data of Aluminum to train and validate selected Models.	194
Appendix E Collected data of Concrete to train and validate selected Models.	203
Appendix F Collected ABS data to train and validate Automated model.....	210

List of Figures

Figure 2-1 Rytter's Hierarchy for Structural Health Monitoring and Assessment and Material Science Tasks.....	17
Figure 2-2 Machine learning workflow.....	24
Figure 2-3 The categories of machine learning with commonly adopted algorithms.....	25
Figure 2-4 The major concerns in the crack mechanics' field.....	27
Figure 4-1 The Geometry of Aluminum Beam.....	71
Figure 4-2 The Geometry of ABS Beam.....	72
Figure 4-3 The Geometry of Concrete Beam.....	73
Figure 4-4 The experimental setup.....	74
Figure 4-5 Signal Processing.....	75
Figure 4-6 The Flow Chart of Data Analysis Steps.....	77
Figure 4-7 The correlation between crack depth (mm) and natural frequency (Hz) for a- Aluminum b-ABS c- Concrete.	80
Figure 4-8 The correlation between Crack depth (mm) and amplitude (mm) for a- Aluminum b- ABS c- Concrete.....	80
Figure 4-9 The correlation between natural frequency (Hz)and amplitude (mm) in a- Aluminum b-ABS c- Concrete.....	81
Figure 4-10 5-Fold cross validation schematic.	82
Figure 5-1 The actual against predicted of ABS crack depth in case of all features.	89
Figure 5-2 The actual against predicted of Aluminum crack depth for selected machine learning models.....	90
Figure 5-3 The actual against predicted of Concrete crack depth for selected machine learning models.....	92
Figure 5-4 The actual against predicted of ABS crack depth based on selected features.....	93
Figure 5-5 The actual against predicted of Concrete crack depth based on selected features.	95

Figure 6-1 Flowchart of the process of AutoML H2O Application.	104
Figure 6-2 The parameters of AutoML operation.....	108
Figure 6-3 Code to identify optimal hyperparameters for all selected models.	109
Figure 6-4 The preformance of H2O selected models based on RMSE metric	111
Figure 6-5 The actual against predicted of ABS crack depth a. GBM AutoML b. Stacked Ensemble Best of Family AutoML c. CLM AutoML d. DRF AutoML.	113

List of Tables

Table 3-1 The most important requirements and parameters for ML approaches 67

Table 4-1 The Experimental Conditions and Parameters..... 76

Table 4-2 Spearman’s correlation coefficient between features and crack depth 79

Table 5-1 The evaluation of training and validation for selected models. 87

Table 5-2 The evaluation of training and validation based on selected features. 94

Table 6-1 The importance of features for AutoML models. 106

Table 6-2 The performance of H2O models based on different regression metrics. 111

List of Equations

(3-1)..... 48

(3-2)..... 48

(3-3)..... 49

(3-4)..... 49

(3-5)..... 52

(3-6)..... 55

(3-7)..... 55

(3-8)..... 55

(3-9)..... 56

(3-10)..... 56

(3-11)..... 56

(3-12)..... 58

(3-13)..... 58

(3-14)..... 58

(3-15)..... 60

(3-16)..... 62

(3-17)..... 62

(3-18)..... 63

(3-19)..... 63

(3-20)..... 64

(3-21)..... 64

(3-22)..... 65

(4-1)..... 75

(4-2)..... 75

(4-3)..... 77

(4-4)..... 83

1 INTRODUCTION

1.1 Background

Cracks in structures pose a significant threat to their safety and longevity, leading to catastrophic failures and loss of life. Thus, understanding the mechanisms of crack propagation and predicting the behavior of structures under dynamic loads is of extreme importance. Traditional analytical methods for modeling crack propagation are often limited to simple geometries and linear elastic materials. However, real-world structures are complex and exhibit nonlinear behavior under dynamic loads, making it challenging to accurately predict crack growth and propagation.

Machine learning (ML) approaches have emerged as promising tools for modeling the interdependencies between dynamic loads and crack propagation in complex structures. By using data-driven techniques, these methods can learn from the behavior of the system under various conditions and make accurate predictions about its future behavior. ML approaches have been used successfully in various fields, including finance, healthcare, and image recognition. The application of ML techniques in structural engineering is relatively new, but the potential benefits are significant.

1.2 Motivation

The study's motivation arises from the pressing need to enhance the accuracy of predicting crack growth and propagation under dynamic loads in complex structures within the field of structural health monitoring and assessment. The conventional analytical methods employed for modeling crack propagation are often restricted to simple geometries and linear elastic materials. However, real-world structures are intricate and exhibit nonlinear behavior when subjected to dynamic loads, posing a formidable challenge in accurately predicting crack growth and propagation. The precise anticipation of crack growth and propagation plays a pivotal role in ensuring the safety and durability of structures.

In recent years, ML approaches have garnered increasing interest for addressing structural engineering problems. ML techniques offer several advantages over traditional analytical methods, primarily by leveraging data to learn and make precise predictions about intricate systems. By utilizing data-driven techniques, these ML methods can assimilate information from a range of conditions and accurately forecast the future behavior of the system. This potential presents an avenue to overcome the limitations of traditional analytical methods and provide more precise predictions of crack growth and propagation in materials science.

In addition to enhancing prediction accuracy, machine learning (ML) approaches present the opportunity for significant time and cost savings. Traditional analytical methods require exhaustive data collection, meticulous model development, and rigorous validation processes, all of which can be exceptionally time-consuming and expensive. However, it is important to note that ML methods do require substantial data collection; they cannot completely circumvent this necessity. Nevertheless, the key distinction lies in the learning process. Unlike traditional methods, ML approaches have the ability to autonomously learn from the provided data and subsequently generate accurate predictions with minimal human intervention. While the training process in ML can indeed be lengthy, the efficiency gained in prediction accuracy, coupled with the reduced need for extensive manual model development and validation, results in notable time and cost savings. This intrinsic capability of ML to learn from data and make predictions autonomously reinforces its effectiveness, making a compelling argument for its adoption in various fields. This capability has the potential to streamline and expedite the process of modeling and predicting crack growth and propagation, resulting in significant reductions in time and cost requirements. However, the application of ML in materials science and crack prediction also presents several challenges. One of the primary challenges is the availability and quality of data. ML algorithms heavily rely on large and high-quality datasets to learn effectively. Obtaining such datasets with detailed information on crack growth and propagation under dynamic loads can be challenging and may require extensive experimental or simulation efforts. Additionally, the interpretation of complex ML models and the extraction of meaningful insights from them can be

difficult, hindering the adoption and acceptance of these models in the materials science community. The integration of ML approaches with existing analytical models and the development of hybrid methodologies poses further challenges, as they require careful consideration of model compatibility, data integration, and validation procedures. Furthermore, the black-box nature of some ML algorithms can limit their explainability, making it challenging to understand the underlying factors driving predictions. Explainability is crucial in materials science, as it helps researchers and engineers gain insights into the fundamental mechanisms of crack growth and propagation, contributing to the development of better materials and structural designs. Machine learning holds great promise for enhancing crack growth and propagation predictions in materials science. However, we must overcome some challenges before fully realizing its potential. The research purposes to address several challenges, including ensuring the availability of sufficient relevant data, enhancing the models' clarity and comprehensibility, integrating them seamlessly with existing methods, and ensuring they can offer clear explanations for their predictions[1]–[8]. Overcoming these challenges will enable the development of robust and reliable ML models that can accurately predict crack behavior under dynamic loads, ultimately enhancing the safety and durability of structures in various applications.

1.3 Aim and Objectives

The overall scientific aim of this research is to investigate machine learning approaches to model the interdependencies between applied load and crack propagation in mechanical structures, aiming to generalize an accurate crack prediction model for different materials (composite, metal, and polymer) and reduce the associated computational effort. To achieve the aim of this research, the author emphasizes the following research objectives:

1. Identify the current challenges in machine learning approaches for structural health monitoring and assessment.
2. Analyse the compatibility of dynamic response data of different material structures with the prominent ML approaches.

3. Develop an ML model based on the suitable approach/es to relate the interdependencies in between the dynamic response and crack propagation in structures .
4. Evaluate the efficiency and accuracy of the developed ML model.
5. Validate the proposed model with experiments on arbitrary values (composite, metal, and polymer).

1.4 Thesis Structure

Chapter 2: Literature Review_ summarize relevant literature on machine learning approaches, dynamic loads, and crack propagation. Discuss related work in the field, highlighting key findings, methodologies, and any identified gaps in the existing literature that the current research aims to address.

Chapter 3: Suitability Analysis of Machine Learning Algorithms for Crack Growth Prediction Based on Small-Moderate Dynamic Response Data _ Evaluate and analyse the suitability of various machine learning algorithms for crack growth prediction using small to moderate dynamic response data. Discuss the selection criteria, methodology, and justification for the chosen algorithms. Present findings and insights from the analysis.

Chapter 4: Methodology and Data Preparation _ Describe the dataset used for crack prediction in FDM 3D printed ABS and 2024 Sheet Aluminium specimens and Concrete. Explain the steps taken to preprocess and clean the data. Outline how the dataset was divided into training and testing sets.

Chapter 5: Results and Discussion_ Introduce the machine learning models employed. Present the results of training and testing the models. Include the evaluation of the accuracy and reliability of the crack prediction models. Summarize the key findings and insights from the crack prediction application.

Chapter 6: Automated Model using H2O Library _ Discuss the development of a generalization model using the H2O library. Explain the methodology,

implementation, and evaluation of the model. Present the results and discuss its performance compared to previous models.

Chapter 7: Conclusion _Summarize the main findings and contributions of the research. Propose future directions and potential areas for further research.

2 LITERATURE REVIEW

2.1 Introduction

SHM is a field of study focused on the monitoring, assessment, and maintenance of the structural integrity of engineered systems, including bridges, buildings, and aircraft. SHM utilises various techniques to track changes in structural performance, identify potential failures, and predict future structural behavior. A hierarchy of levels required for structural health monitoring and assessment has been defined by Rytter [9], which is illustrated in Figure 2-1. The hierarchy includes multiple levels that are required to perform SHM effectively. Each level builds upon the previous level, and a higher level of desired detail leads to increased difficulty. The levels include data acquisition, data processing, feature extraction, damage identification, damage characterization, and prognosis. Prognosis is a unique level that requires an understanding of the physics of the damage and is different from other hierarchy levels [10].

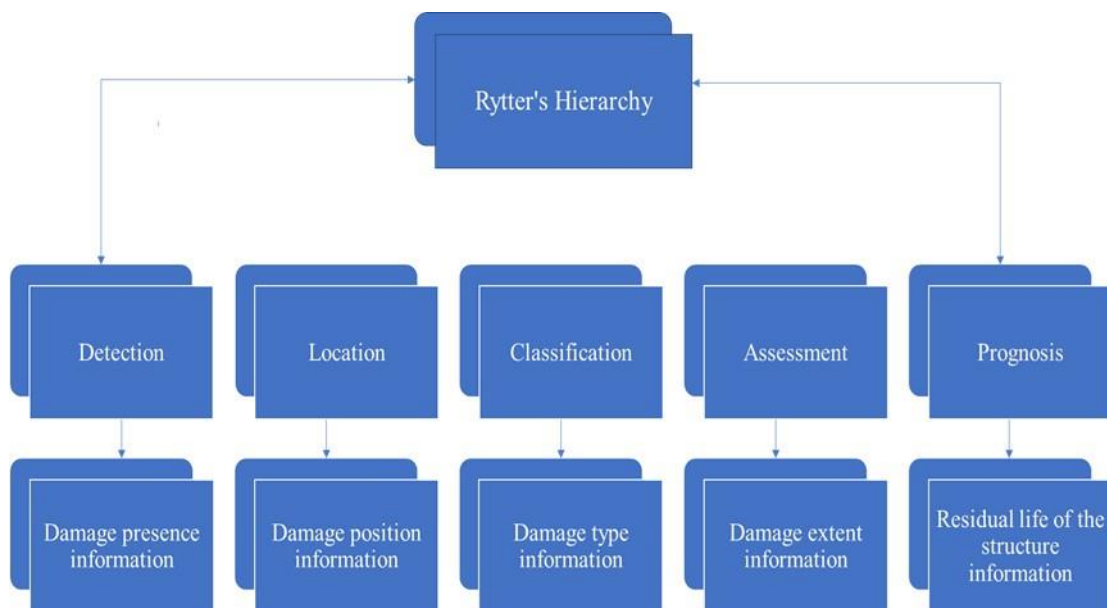


Figure 2-1 Rytter's Hierarchy for Structural Health Monitoring and Assessment and Material Science Tasks.

SHM approaches provide early warnings of damage occurrences to support asset management decisions, which eliminates unscheduled maintenance and in-service failures. SHM can present various economic benefits by reducing system failures, increasing the efficiency of maintenance, and providing cost-effective data-based designs. This leads to an extension of the structure design life and improves safety, making it widely implemented across different industries, such as civil infrastructure, manufacturing, aerospace, and power generation. The process of SHM involves employing statistical analysis, damage-sensitive feature extraction, and dynamic responses to monitor the structure using non-destructive methods such as vibration/model analysis, X-rays, ultrasonic testing, and optical methods. However, the development of robust SHM technologies tends to have three main challenges. First, the damage is not directly measurable. Second, confounding factors such as changes in boundary conditions and/or obscure load patterns in data associated with damage, and environmental conditions need to be eliminated before implementing health decision strategies. Third, numerous techniques of structural health monitoring require damage situation data from all damage scenarios of interest, often in a set of operational conditions, which may not be feasible practically or economically or would cause safety concerns. To monitor the structure in the process of structural health monitoring and assessment, statistical analysis, damage sensitive feature extraction, and dynamic responses are employed. In the following section, we will explore various SHM approaches that have been developed to assess and monitor the integrity and condition of structures over time.

2.2 Structural Health Monitoring Approaches

2.2.1 Inverse Model-Driven Approaches

Inverse model-driven approaches have been increasingly used as a promising solution for SHM. SHM involves the continuous monitoring of the condition of structures and their components to ensure their safe and efficient operation. Inverse model-driven approaches use mathematical models to reconstruct the state of the system based on measurements taken from sensors installed on the

structure. The purpose of inverse modeling is to obtain information about the system's state that would otherwise be difficult or impossible to measure directly[11], [12].

One of the key advantages of inverse model-driven approaches is their ability to incorporate multiple sources of information, including measurements of temperature, vibration, and strain, to obtain a comprehensive understanding of the structure's condition. This information can be used to identify potential problems, such as cracks or deformations, that may compromise the structure's integrity. Inverse model-driven approaches can also provide insight into the underlying physical mechanisms that contribute to structural damage, which can inform the design of more robust and resilient structures[11].

Inverse model-driven approaches can be used in a variety of structural monitoring and assessment scenarios, such as bridge health monitoring, wind turbine blade inspection, and structural integrity assessment of offshore platforms. For example, in the case of bridge health monitoring, inverse models can be used to estimate the stress and strain fields within the bridge structure based on measurements of deflection and acceleration. This information can then be used to detect the presence of cracks, deformations, and other anomalies that may indicate the need for maintenance or repair[11], [13]–[15].

One of the key applications of inverse model-driven approaches in SHM is in the analysis of integrated thermal and vehicular response. A study by Kromanis et al. [16] introduced a measurement interpretation approach that merges the regression-based thermal response prediction methodology with a novel methodology for predicting traffic-induced response. The approach first removes both environmentally and operationally induced trends from measurement time series of structural response [10]–[12], [17], [18]. The resulting time series is then analysed using anomaly detection techniques. Another study by Dang et al [19] developed a practical end-to-end framework that makes use of physical features embedded in raw data and an elaborated hybrid deep learning model for SHM. The method combines various signal processing techniques such as the autoregressive model, discrete wavelet transforms, and empirical mode

decomposition. The hybrid deep learning model is designed based on the capacity of capturing local information and the prominent ability to learn long-term dependencies. Moreover, the development of smart sensing technology approaches in SHM of bridge structures has been explored by [20]. The research provides a detailed understanding of bridge monitoring, focusing on sensors utilized and all kinds of damage detection according to bridge nature and environmental degradation under static and dynamic loading. These studies demonstrate the potential of Inverse Model-Driven Approaches in providing real-time solutions for decision making in SHM, despite the high computational cost associated with these approaches.

Despite the numerous benefits of inverse model-driven approaches, there are some challenges that must be addressed to ensure their effective and reliable use in SHM. One of the key challenges is the complexity of the mathematical models used in inverse modeling. These models require a detailed understanding of the physical behavior of the structure, including the material properties and geometry of the components, as well as the loading conditions and boundary conditions. Another challenge is the need for high-quality and reliable sensor data, which can be difficult to obtain in challenging environments, such as offshore platforms or wind turbines. Inverse model-driven approaches offer a promising solution for structural health monitoring and assessment, allowing for the integration of multiple sources of information and the detection of structural damage in real-time [13], [21], [22]. However, to ensure their effective and reliable use, the complexity of the mathematical models used in inverse modeling and the quality of sensor data must be addressed [8]. ML also faces a similar challenge in terms of needing high-quality and reliable data for training and prediction. Just like in the context of structural health monitoring, where sensor data needs to be accurate and trustworthy, machine learning models rely on clean and relevant data to make accurate predictions and decisions. Whether it's offshore platforms, wind turbines, or any other application, the availability of good data is crucial for the success of machine learning models.

2.2.2 Forward Model-Driven Approaches

Forward model-driven approaches in SHM involve the use of mathematical models to predict the behavior of a structure under various conditions. These models are typically based on the physical properties of the structure, such as its geometry, material properties, and loading conditions. By simulating the response of the structure to various stimuli, such as external forces or temperature changes, these models can be used to identify potential structural issues before they become serious problems [23]. One of the key advantages of forward model-driven approaches is their ability to provide a comprehensive understanding of the behavior of a structure under different conditions. By simulating the response of the structure to various stimuli, these models can identify potential issues that may not be apparent through visual inspection or other monitoring techniques. This allows engineers and other stakeholders to make informed decisions about the maintenance and repair of the structure, potentially reducing downtime and extending the life of the structure. Another advantage of forward model-driven approaches is their ability to incorporate data from multiple sources. By combining data from sensors, visual inspections, and other sources with the output of the model, engineers can develop a more complete understanding of the structure's behavior. This can help identify potential issues early on, allowing for more efficient and effective maintenance and repair[24], [25]. There are a variety of different techniques used in forward model-driven approaches, including finite element analysis, computational fluid dynamics, and structural dynamics modeling. These techniques rely on advanced computational methods to simulate the behavior of the structure under various conditions and can be tailored to the specific needs of the structure and the monitoring program. In recent years, the forward model-driven approach has been instrumental in advancing our understanding of structural behavior, particularly in the context of seismic activities. Xiong et al [26] employed this methodology to assess the seismic performance of steel box girder bridges, emphasizing the integration of the bridge deck with the orthotropic steel deck, forming a steel-concrete composite structure. Their research combined finite element analysis (FEA) with field measurements to derive their conclusions.

Similarly, [27] utilized finite element (FE) analysis and operational modal analysis (OMA) to evaluate the seismic vulnerability of a masonry arch bridge on the Kalka Shimla Mountain Railway line, a region known for its severe seismic activity. Their findings provide valuable insights into the seismic performance of such structures. On a different note, Tatarciuc et al. [28] used finite element analysis to assess the stability of inlay-retained dental bridges, demonstrating the versatility of the forward model-driven approach beyond traditional civil engineering applications. Liu et al [29] delved deep into the finite element analysis of cable-stayed bridges, focusing on their behavior under seismic excitations. Their study underscored the importance of considering appropriate initial shapes and highlighted the intricate interaction between the deck and the stay. Collectively, these studies underscore the versatility and efficacy of the forward model-driven approach in understanding and predicting structural behavior under various conditions. Previous studies demonstrate the potential of forward model-driven approaches in structural health monitoring and assessment. By utilizing advanced computational models, integrating sensor data and real-time monitoring systems, and incorporating machine learning techniques, engineers can gain a more comprehensive understanding of the behavior of structures under different conditions. As these approaches continue to evolve, they are likely to become an increasingly important tool in the field of structural engineering[8], [14], [30]. Forward Model-Driven Approaches, while effective in many applications, have their limitations. One significant drawback is their dependency on accurate and comprehensive mathematical models that describe the underlying physical processes. Constructing these models can be a complex and time-consuming task, requiring in-depth domain knowledge and expertise. Additionally, these models may not always capture all the intricate details of real-world systems, leading to discrepancies between the model's predictions and the actual behavior of the system. This limitation becomes especially critical in situations where the system's dynamics are poorly understood or where external factors, such as environmental conditions, introduce uncertainties. Furthermore, these approaches often struggle with adapting to dynamic and changing environments, as model updates can be challenging to implement in real-time.

The reliance on precise models and their lack of flexibility in handling uncertainties and dynamic conditions can restrict the practical applicability of Forward Model-Driven Approaches in certain real-world scenarios [23], [31], [32].

2.2.3 Data-based Approaches

This approach uses statistical techniques to analyse measurement data in order to identify patterns and anomalies that may indicate structural degradation. According to a review by [8], [14], [30], these approaches can be used to identify patterns and anomalies in the data that may indicate structural degradation. Data-based approaches are a key component of SHM as they help to capture, process, and interpret the vast amounts of data generated by sensors. There are several data-based approaches used in SHM, including those discussed in [2], [3], [33], [34]. A general framework of data-based approaches consists of the below steps [34]–[36]:

1. **Data Acquisition:** The first step in data-based approaches is to acquire data from sensors that are placed on the structure. These sensors can measure various parameters such as strain, temperature, vibration, and displacement.
2. **Data Processing:** Once the data is acquired, it needs to be processed to extract useful information. This involves filtering, averaging, and smoothing the data to remove noise and improve its accuracy.
3. **Feature Extraction:** The next step is to extract features from the processed data that can provide insights into the structural condition. Features can include frequency response functions, modal parameters, and damage indicators.
4. **Data Analysis:** The extracted features are then analysed using statistical and machine learning techniques to identify patterns and anomalies that indicate changes in the structural condition.
5. **Structural Health Assessment:** Finally, the information obtained from data analysis is used to assess the health of the structure and determine if any maintenance or repair is required.

According to Klunnikova et al [37], machine learning and extracted features play a critical role in data-based decision-making for structural damage prediction. Figure 2-2 presents a clear chart of the machine learning workflow for SHM tasks, which includes data pre-processing and cleaning, model training, and model testing and evaluation. The authors emphasize that cleaning raw data and selecting and extracting sensitive features before training machine learning models can enhance the performance and accuracy of the model prediction.

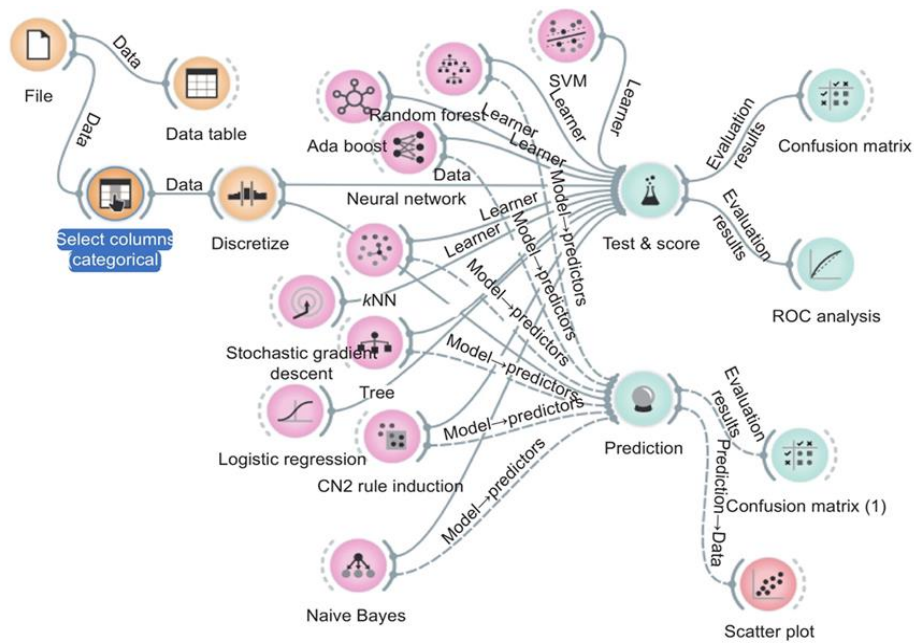


Figure 2-2 Machine learning workflow [37]

Figure 2-3 illustrates the categories of machine learning and commonly adopted algorithms within the context of structural health monitoring and assessment, as presented by [38]. It also highlights three major problems that machine learning techniques aim to solve: regression, density estimation, and classification. Regression infers an unknown function based on mapping an input to an output, while density estimation infers probability density clusters from data, and classification labels data based on an inferred decision boundary.

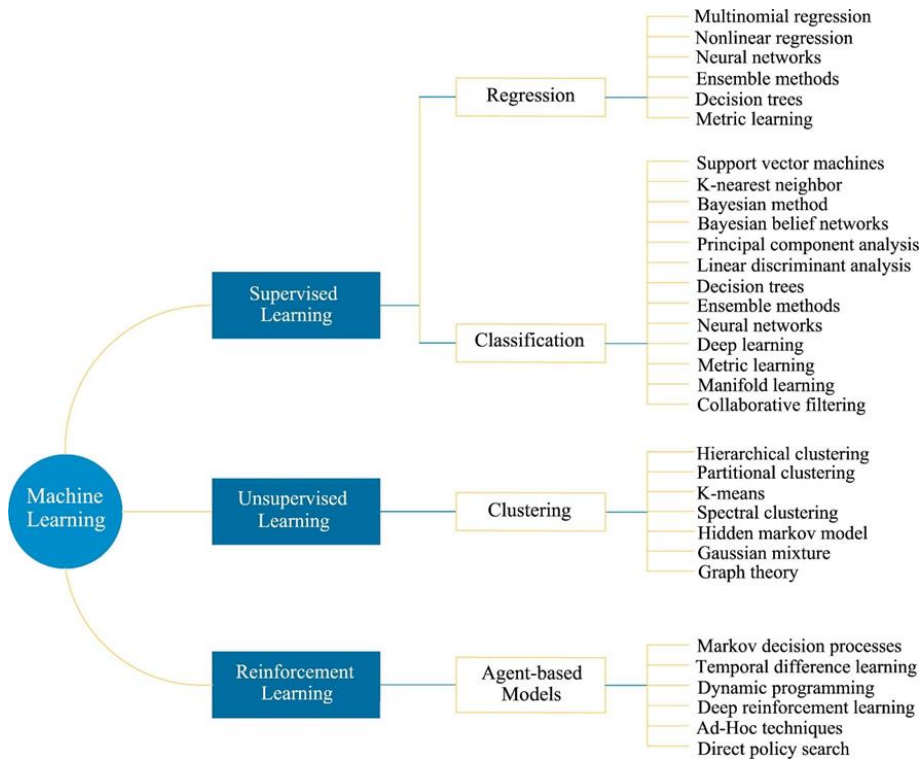


Figure 2-3 The categories of machine learning with commonly adopted algorithms [38]

Recent advancements in the realm of SHM have been significantly influenced by data-driven approaches. Bono et al [39] delved into the application of intelligent systems for SHM, emphasizing the potential of deep learning models in detecting structural changes. Their research compared two deep learning approaches applied to a test rig of a small four-storey building model, revealing that physics-informed neural networks showed heightened potential in damage detection and location. Sierra et al [40] highlighted the increasing demand for renewable energy and the consequent rise in wind energy projects. The research focused on data extraction from monitoring systems for wind turbines, particularly using fiber-Bragg-grating-based systems. Building on these advancements, Reuland et al. [41] emphasized the importance of rapid post-earthquake damage assessment. Their research offers a comprehensive review of damage-sensitive features, aiming to assess the capacity of data-driven indicators to detect, localize, and quantify the presence of nonlinearity in seismic-induced structural response. Dang et al. [42] highlighted the significance of SHM in rail transit systems, proposing an efficient image classification model for railway status identification using ultrasonic guided waves. Pollastro et al [43] introduced a semi-supervised

method with a data-driven approach using Artificial Neural Networks (ANNs) in SHM systems, showcasing its potential in detecting structural anomalies. Ahmad et al [44] proposed a sensing system comprising a vast network of RF-powered sensors that use backscatter communication, indicating the potential of this approach for 3D structural health monitoring.

One of the advantages of data-based approaches in SHM is that they can enhance continuous and real-time monitoring of the structure. This enables early detection of any damage or deterioration and can help prevent catastrophic failures. Additionally, data-based approaches are non-destructive, meaning that, they do not require physical intervention in the structure and can be applied to both new and existing structures.

However, data-based approaches also have some challenges, including the need for accurate sensor placement and calibration, the need for robust data processing and analysis techniques, and the need for effective visualization and communication of the results to stakeholders. Data-driven approaches have shown great potential in detecting and locating damage in various types of structures, contributing to the development of more effective and efficient SHM systems[3], [8], [45].

2.3 Machine Learning Applications in Crack Mechanics

Crack mechanics is a specialized field of study, primarily focusing on the analysis of crack initiation and propagation in materials. This area of research is paramount for enhancing the safety and reliability of engineered structures. According to research by [6], [46], [47], the subdomains of crack mechanics include:

1. Crack Diagnosis and Detection: This area concerns the identification of cracks on various scales (macro, meso, and micro) that might culminate in mechanical failure. Mechanical damage typically describes the defect within a structure.
2. Fracture Parameters and Mechanical Fracture: This segment encompasses parameters such as fracture energy, fracture toughness, crack propagation, and fractography, all of which pertain to mechanical

fracture. A fracture represents the division of a solid body into two or more parts.

3. Fault and Error Diagnosis and Detection: The emphasis here is on identifying deviations in mechanical systems or components. Such deviations diverge from the anticipated conditions, leading to diminished mechanical performance.
4. Failure Mode and Mechanism Identification: This domain covers the physical processes causing mechanical failure. A nuanced description of a failure mode is termed a "failure mechanism."

Delving into fracture mechanics, both empirical and analytical approaches, demands significant time and a high level of technical expertise. This makes it a challenging endeavour. Figure 2-4 highlights the pressing issues identified in contemporary fracture mechanics research. Notably, there are instances where neither empirical nor analytical solutions can address intricate engineering challenges, such as decoding nonlinear relationships in high-dimensional data. Hence, meticulous monitoring and evaluation of various mechanical structures are imperative. Such assessments ensure dependable information on their serviceability, safety, health, and integrity, considering the structure's continued performance is intertwined with the timely identification and monitoring of

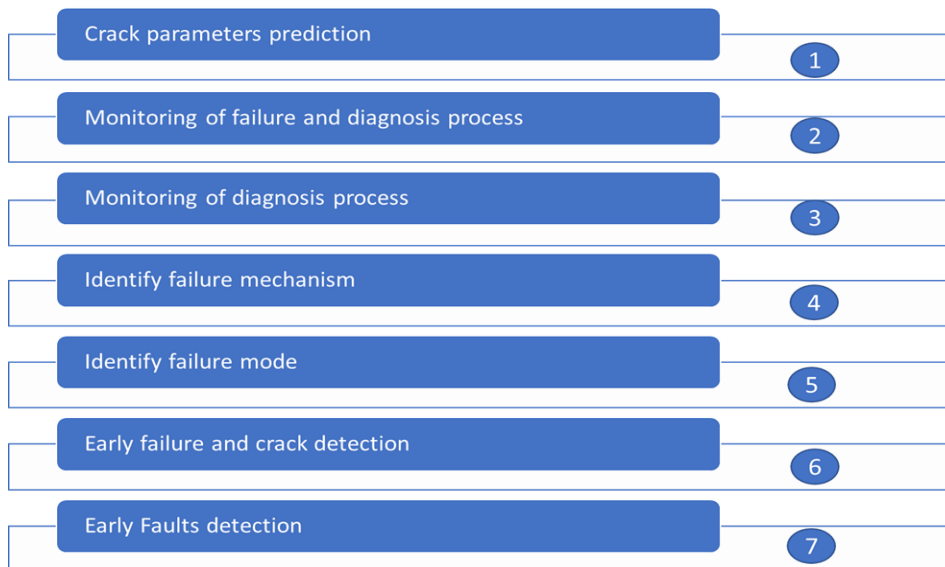


Figure 2-4 The major concerns in the crack mechanics' field.

damage. In recent times, ML has surfaced as an invaluable asset for crack analysis. ML algorithms, including ANNs, Support Vector Machines (SVMs), and Random Forests (RFs), can map relationships between crack properties and numerous inputs, like material attributes, loading conditions, and geometry. This equips engineers with predictive capabilities, enriching design, and maintenance processes. ML finds its utility in crack mechanics, notably in predicting crack initiation and propagation. For instance, ANNs have modelled relationships between stress intensity factors (SIFs) and inputs, aiding predictions about crack behaviours [48]. SVM models have been utilized to forecast the critical stress intensity factor [49]. Various studies have demonstrated the efficacy of ML in this domain [50], [51]. Another avenue where ML showcases its prowess is in the classification of cracks. RFs, for instance, have been employed for classifying cracks by shape and geometry [47]. This aids in prioritizing maintenance efforts and enhancing structural safety. Balcioglu et al. [46] also ventured into this domain, advocating for a meticulous approach to data collection for achieving accurate results. Optimizing structural designs to counteract crack initiation is another potential application of ML. ANNs, for example, have facilitated designs that boost crack resistance [52], [53]. This proactive approach could lead to structures that are intrinsically resistant to crack propagation, mitigating potential failures. Furthermore, ML assists in developing crack detection and monitoring systems. Real-time crack detection, based on structural vibration responses, has been achieved using ANNs [54], underscoring the predictive maintenance potential. However, challenges persist. Data availability remains a significant hurdle. ML demands comprehensive and representative datasets, yet acquiring such data on crack behaviours is onerous, typically requiring costly experiments or simulations. Additionally, selecting the appropriate ML algorithm, given the vast array available, is a complex task. The choice depends on the specific needs of the crack analysis and available resources. While challenges exist, ML's potential in crack mechanics is undeniable. It promises enhanced comprehension of crack behaviours, culminating in safer and more reliable engineering structures. The role of feature extraction and selection remains pivotal, ensuring accurate outcomes in damage assessment.

2.4 The Importance of Feature Engineering and Tools

Worden and Manson[34] proposed seven axioms for structural health monitoring and assessment. These axioms confirm that feature extraction through statistical classification and signal processing is fundamental for converting sensor data (raw data) into damage information. Therefore, feature extraction and selection should be performed before training the ML models to enhance the model's performance, flexibility, and efficiency. While most ML approaches use standard feature extraction and selection algorithms, some can modify features to achieve the best potential prediction performance. Feature extraction involves transforming the measured signal into features (also known as attributes), which serve as inputs for a learning algorithm. The extracted features should retain the maximum amount of information contained in the collected signals by eliminating all redundant data. The features' suitability can be quantified according to their sensitivity to cracks and noise immunity. Another constraint is determined by insensitivity to various operating and environmental conditions. In other words, if the measured signals are affected by differences in these operational and environmental conditions, the extracted features should be stable and robust under these conditions [55], [56]. To address these issues, a preliminary step to feature extraction, involving signal processing procedures to correct the collected signals and ensure a fair comparison with the baseline signal, is often required. A popular method for reducing the effects of noise is discrete wavelet transformation [56], [57]. Feature extraction involves two essential tasks that enhance the effectiveness of ML models. The first task is dimensionality reduction, which can be achieved using various methods such as Linear Discriminant Analysis (LDA), Generalized Discriminant Analysis (GDA), and Principal Component Analysis (PCA). The second task involves transforming the data into a higher-dimensional space to make the patterns sparse and separable, such as in kernel-based ML algorithms [58]. Feature selection is the process of identifying the most relevant features that can predict outputs with high accuracy [56], [59]. In analogy with SHM applications, feature selection aims to find features that offer high performance in damage detection and classification. Consequently, features that contain redundant information will be excluded from

the final dataset or include information about damage presence [56]. This way, the global dataset dimension is significantly reduced, helping avoid the curse of dimensionality [56], [60]. Concerning the number of features, adding more may cause a decrease in the accuracy of the classification algorithm. In theory, an increase in the dimensions of features should be correlated with a significant increase in the training dataset. This is impractical in most instances and requires considerable computing time. If this condition is not tested, it may cause an overfitting issue [61]. Overfitting occurs when the misclassification of training data reduces as it increases for validation data. In other words, while the classifier can accurately predict the class memberships for training data by memorizing them, it loses its ability to generalize unknown data. Cross-validation is an approach used to evaluate a classifier's accuracy and validity by testing it on different partitions of the training data [56]. Feature selection can be classified into three methods:

1. Filter methods: These classify the original features according to an importance measure such as the scores from correlation coefficients between individual features and the response variable or chi-square test. Filter methods use statistical techniques to evaluate a subset of features.
2. Wrapper methods: These recursively exclude or include features from the original features and select the best-performing feature set based on the feedback from the ML model. They use cross-validation to evaluate a subset of features.
3. Embedded methods: These are used by algorithms that have their built-in feature selection methods (e.g., LASSO and Ridge regression).

Both wrapper and filter methods effectively avoid overfitting problems by reducing highly correlated features and model complexity. However, since filter methods do not consider the classifier's performance, they can't be utilized to identify damage-sensitive features. Wrapper methods are well-suited for damage detection. Despite the importance of feature selection in the process of damage detection, it hasn't received much attention within the SHM community. This might be because the intuitive belief is that more features improve classification performance. It might also be implicitly incorporated into the feature extraction

process. However, Bagheri et al [62] presented feature selection using ANN to categorize three types of faults in immersed plates. Feature extraction was performed in time-frequency, time, and frequency domains. Eight features were selected to represent a signal. As ANN is considered a comprehensive approach, finding the optimal number of features is challenging, especially when the number of hidden neurons may also be adjustable. The authors therefore decided to assess each feature individually and compare it with a combination of four features. Although the four-feature case showed no substantial improvement over a single feature, there's no guarantee that the optimal number of features was found because other combinations weren't considered. Similar work using a probabilistic neural network was conducted by [63]. Among the six features extracted, four were chosen as the most discriminatory. Once again, no explanation was given regarding the criterion upon which this feature was selected. Jiang et al. [64] used spatial filtering to preprocess microphone signals. Pre-processed signals and the original sensor signals were fed into NN5. The study results demonstrated a comparison in classification accuracy of the NN5 model without and with spatial filtering in both training and cross-validation. Filtered signals achieved 97.7% and 97.5% accuracy in training and cross-validation, respectively, compared against 86.4% and 80.4% without spatial filtering.

In addition to the aforementioned feature selection and extraction techniques, domain-specific feature engineering techniques have been developed to address specific data structures and issues, such as Scale-Invariant Feature Transform (SIFT) features [65], Histogram of Oriented Gradients (HOG) features [66], and HAAR-like features [67], [68]. These techniques require trial and error testing and are developed to work only for very specific problems and data structures. Deep learning (DL) and neural networks (NNs) approaches automate feature engineering to enhance performance in various data mining and pattern recognition domains [69]. Due to the increase in computation power, which has made complex NNs architectures trainable, NNs have become popular, achieving exceptional feature extraction [70]. The complex architecture of deep neural networks allows for automatic feature extraction. Contrary to shallow learning or

conventional machine learning, feature extraction is performed outside the algorithmic stage. In other words, data scientists and experts, not machines, are responsible for analysing raw data and transforming it into valuable features. The extraction of sensitive features is crucial to the successful damage assessment application within the machine learning paradigm. However, in practice, it seems difficult to identify a superset of damage-sensitive features to estimate and evaluate all types of structural systems.

2.5 Machine Learning for Crack Assessment in Smart Structures

Smart materials are capable of responding to external factors such as temperature, stress, magnetic fields, and humidity by changing their fundamental properties in a controlled manner. When integrated with sensors and actuators, smart materials form intelligent structures capable of sensing and adapting their dynamic and static responses, also known as adaptive structures. These structures can continuously monitor the existence, location, and severity of damage. Machine learning can be used to assess and detect damage in these structures, with two essential stages: feature extraction and selection, and model training .[71], [72]

In parametric machine learning-based methods, the modal parameters of the monitored structures, such as damping ratios, frequencies, and mode shapes, are considered as extracted features[73]–[76]. Non-parametric machine learning-based methods use techniques such as wavelet transform, autoregressive modeling, basic statistical analysis, time-frequency methods, and principal component analysis to extract features. Popular domains for feature extraction in undamaged and damaged structures include time domain, frequency domain, time-frequency domain, modal analysis, and impedance [77]–[80].

Sensitive features can be extracted from high frequency, low-wavelength guided waves or low frequency, high wavelength structural response. These features, also known as descriptors, include duration, rise time, count to peak, peak amplitude, average frequency, energy, and peak frequency for acoustic waves, low-wavelength guided, or high-frequency. Different classifiers are used in both

nonparametric and parametric machine learning-based methods, such as fuzzy neural networks, artificial neural networks, probabilistic neural networks, support vector machine, singular value decomposition, and online sequential extreme learning machine algorithm [69], [75], [81]–[87].

The success of structural damage detection techniques based on machine learning depends primarily on the choice of classifier and extracted features. Therefore, caution must be exercised in choosing the appropriate features to analyse signals and capture the most important characteristic condition. Similarly, based on the extracted feature types, a suitable classifier must be implemented to classify them accurately. This review provides a general guideline for selecting the best combination of ML tools and damage sensitive features for assessing and detecting structural damage in different materials.

2.5.1 Composite Structure

Composite materials are made up of multiple materials with different properties, resulting in materials that have the best qualities of their components without any chemical reaction or solubility [2]. In order to train machine learning algorithms to accurately detect cracks in composite materials, it is important to identify sensitive features that can be used as inputs to the model. Several studies have investigated this issue and identified potential sensitive features. For instance, Khan et al; Yuan et al [88], [89] used a machine learning approach to detect damage in composite laminates such as glass-fibre-reinforced plastic (GFRP) and carbon-fibre-reinforced plastic (CFRP). They identified that features related to the stiffness and modal parameters of the structure were sensitive to damage, including changes in natural frequencies, mode shapes, and modal damping ratios. These features were extracted from experimental data using a combination of modal analysis and digital image correlation techniques. Another study by Abueidda et al; Feng et al. [90], [91] used a convolutional neural network to detect delamination in composite laminates. They found that the location and size of the delamination were important sensitive features, as well as the amplitude and frequency of the signals generated by the ultrasonic testing system. In addition, Panda et al. [92] investigated the use of acoustic emission

signals to detect damage in composite structures. They identified that features related to the amplitude, frequency, duration, and energy of the acoustic emission signals were sensitive to damage, as well as the time interval between successive events. Zhang et al [49] used a support vector machine to detect damage in composite structures using a combination of vibration and acoustic emission signals. They found that features related to both the vibration and acoustic emission signals were sensitive to damage, including changes in natural frequencies, mode shapes, and amplitude and frequency of acoustic emission signals.

Neural networks, convolutional neural networks, backpropagation, K-nearest neighbor, and support vector machines have been applied in the health monitoring and assessment of composite-based structures, including wind turbines, bridges, and high-rise buildings[93]. For instance, Soyoz et al. [94] used a neural network approach to evaluate long-term structural parameters, ageing, mass, and stiffness of the Jamboree Road overcrossing in Irvine, California. Sung et al [95] developed a model that combined the Levenberg-Marquardt algorithm with neural network and generalization method to locate low-velocity impact damage in composite laminates. Chetwynd et al. [96] used a multilayer perceptron neural network (MLP) for damage detection in a stiffened curved carbon fiber reinforced panel, achieving an overall classification accuracy of 90.9%. Hoang et al. [97] compared six machine learning algorithms, including Classification Tree, Naïve Bayesian classifier, Radial Basis Function Neural Network, Backpropagation Artificial Neural Networks, least squares support vector machine, and Support Vector Machine, for automatic recognition of Asphalt pavement cracks based on image processing. The study reported high rates of classification accuracy of 91.91% and 92.62% for the support vector machine and least square's support vector machine, respectively. Das et al. [98] used one-class support vector machines to detect and classify four types of damage in composite laminates, achieving better classification accuracy with time-embedded features than with time-frequency features. Godin et al. [99] employed unsupervised and supervised classifiers for clustering acoustic emission (AE) signals obtained from unidirectional polyester/glass composite

tests. They successfully classified the AE signals related to interfacial decohesion and matrix cracking. Li et al [100] performed cluster analysis of AE signals collected from the damage initiation and development of epoxy/glass woven composites using PCA and K-means++ clustering algorithm.

The aforementioned studies demonstrate that there are multiple sensitive features that can be used to train machine learning algorithms to detect cracks in composite materials, including features related to stiffness, modal parameters, ultrasonic signals, acoustic emission signals, and vibration signals. By carefully selecting and extracting these features from experimental data, it is possible to train accurate and robust machine learning models for crack detection in composite materials[2].

2.5.2 Metal Structure

Detecting cracks in metal materials is crucial for preventing catastrophic failures in structures and engineering components. Machine learning techniques have been employed to detect cracks in metal materials. The sensitivity of the features used to train machine learning models is crucial for the performance of these models. In this regard, various studies have investigated the use of different features to train machine learning models to detect cracks in metal materials [101]–[105]. Machine learning has flexible approaches to the modelling of fatigue crack growth because of its multivariable learning ability and excellent nonlinear approximation [106], [107]. Wang et al. [106] proposed a fatigue crack growth calculation method based on three different machine learning algorithms including genetic algorithms, optimized backpropagation network, radial basis function network and extreme learning machine. Experimental data (including specimen type, size, initial crack length, loading type) of different materials (including 7050-T7451 aluminium alloy, Ti6Al4V titanium alloy, ADB610 steel and D16 alumina) were used for training and testing the proposed ML methods. They suggested the following procedures for developing the well-trained model of machine learning for fatigue crack growth. Raja et al [107] investigated the relationship between stress intensity factor range ΔK and da / dN fatigue crack growth rate da/dN using three machine learning algorithms (curve-fitting model,

extreme learning machine, and backpropagation neural networks) with the experimental fatigue crack growth data of cryo-rolled Al 2014 alloy. The result pointed out that the extreme learning machine is the quickest and best model for predicting unstable crack growth regions compared with the other two used ML models. González et al [108] employed NNs and FEM to identify damage in steel moment frame structures. The mode shape and frequencies (collected by finite element model for five-story office buildings) were used as input to train the ML model. The model output predicted the stiffness and mass of the structure to deliver a damage index at each story. Another study investigated three machine learning models (K-nearest neighbour regression, polynomial regression, and ridge regression) using experimental data of metal plate to predict fatigue crack growth rate in stage II and stage III regions model of K-nearest neighbour regression was identified as better suited for their application. They also founded that the accuracy of the machine learning model is strongly reliant on the selection of training data, prepossessing of experimental data [109]. Thus, the most essential sensitive features can be used for metal structures as the following list:

1. **Texture Analysis:** Texture features, such as grey-level co-occurrence matrix (GLCM) and local binary patterns (LBP), can be used to analyse the texture of metal surfaces and identify cracks.
2. **Frequency Analysis:** Frequency-based features, such as Fast Fourier Transform (FFT) and Wavelet Transform (WT), can be used to analyse the frequency components of metal surfaces, and identify changes caused by cracks.
3. **Acoustic Emission (AE):** AE features can be used to detect the high-frequency signals generated by crack propagation in metals.
4. **Image Analysis:** Image features, such as edge detection, can be used to analyse images of metal surfaces and identify cracks based on changes in the image intensity.

5. **Vibration Analysis:** Vibration features, such as natural frequencies and mode shapes, can be used to detect cracks in metal structures by analysing changes in the vibration response.
6. **Material Properties:** Material properties, such as the elastic modulus and yield strength, can be used to detect changes in metal properties caused by cracks.

It is important to note that the selection of sensitive features depends on the type of metal material being analysed and the specific characteristics of the crack being detected. Moreover, the combination of multiple features can lead to a more robust and accurate machine learning models.

2.5.3 Polymer Structure

Polymer structures are widely used in various applications, including aerospace, automotive, civil engineering, and medical devices, due to their unique properties such as lightweight, high strength, and durability. However, these structures are also susceptible to various types of damages, including cracks, delamination, and material degradation. Such damages can compromise the performance and safety of the polymer structure, potentially leading to catastrophic failures. In recent years, machine learning techniques have been employed for the detection of cracks in polymer materials. A ridge-regularized multiple linear regression was used to predict the damage location, damage severity, and fundamental behavior of ABS structures. This study utilized raw experimental data (both damaged and undamaged data of aluminium and ABS under coupled mechanical loads at various temperatures) from references [110], [111] to train the model. The model indicated that tip amplitude and natural frequency are the most important predictive features for ABS. Furthermore, the model results confirmed that the crack position seems to be of little importance for both materials. The crack position was not identified through the theoretical model. Therefore, future studies should train data using theoretical models to achieve more rigorous ML models [112]. Tibaduiza et al [113] proposed a damage detection and classification approach, which was evaluated and tested on a carbon fibre reinforced polymer (CFRP) structure. This approach involves using data collected

from a structure under different states through a piezoelectric sensor network, leveraging hierarchical nonlinear PCA, discrete wavelet transforms, and machine learning algorithms (K-nearest neighbour and decision tree). The researchers emphasized the importance of properly selecting training data when working with machine learning algorithms; otherwise, the trained machine's results might deviate from system expectations. RFs, ANNs, and Haar wavelet discrete transform were applied to predict the severity and location of a crack in the Bernoulli cantilever beam [114].

Analysis of the prediction results revealed that crack depth was more challenging to predict accurately than its location. The dataset of eight Haar wavelet coefficients yielded more precise predictions on crack locations, while the dataset of eight natural frequency parameters produced more accurate predictions on crack depth. Chen et al [115] developed a damage identification scheme by combining hierarchical cluster analysis based on complex network theory and waveform chain code (WCC) analysis. Waveform chain code analysis was performed using the principal component analysis-reduced frequency response function (PCA-reduced FRF), and the areas under the slope differential value curves were calculated as damage-sensitive WCC features. Unsupervised machine learning using hierarchical cluster analysis was then applied to these damage-sensitive features. A rectangular Perspex plate was studied using this newly developed damage identification scheme. Experimental results demonstrated that the proposed scheme could successfully distinguish all damage conditions from the undamaged state with 100% accuracy. Regarding damage severity and location identification, the scheme could detect damage severity with a damage index as low as 0.17.

Additionally, the combination of PCA-reduced FRF and mode shapes showed a positive correlation between the magnitude of the resonant peak and the displacement of the impact point in identifying different damage locations of the plate. The selection of sensitive features for machine learning-based crack detection in polymer materials should be tailored to the specific application and the type of polymer material in question. A careful selection of these features can

enhance the accuracy and reliability of machine learning models for crack detection.

2.6 Common challenges of machine learning

To transition ML in SHM applications from conceptual research to practical implementation, several obstacles must be addressed. This section provides a comprehensive discussion of these challenges and proposes areas for future exploration.

2.6.1 Availability and Quality of Data

One of the main contributors to the success of ML in various fields is its capability to obtain the necessary data. Although the amount of data required to achieve reasonable performance for ML models depends on the goal and problem, it's crucial to have ample high-quality data representing the true distribution. This enables the ML algorithms to discern underlying patterns and create predictive models that are genuinely generalizable within the problem's scope. A significant challenge for ML in SHM applications is that datasets are often limited in diversity and volume. In experiments aiming to predict structural performance and response using ML techniques, data was sourced from nonlinear response history analyses conducted by other researchers [116], [117]. However, these datasets weren't publicly accessible. To comprehend a truly representative dataset of structural response requirements, an open-access repository with stringent quality control measures should be established [70]. Over time, resources like Data Centre Hub (<http://datacenterhub.org>), Structural ImageNet, and the National Hazard Engineering Infrastructure (<http://rapid.designsafe-ci.org>) will help mitigate this challenge. While there might be hundreds of data points, datasets produced by physical experiments are more varied. Previous studies have utilized data from a broad spectrum of experiments conducted by various researchers, such as [80], [118], [119]. A potential solution to the scarcity of data from physical experiments is to incorporate ML algorithms with domain knowledge. This reduces the model space complexity, subsequently decreasing the data volume needed for effective performance.

Transfer learning (TL) can address the data shortage issue. The core principle of transfer learning is leveraging knowledge from training one model in a specific domain or problem for another. Azimi et al. [120] introduced a novel convolutional neural network-based method for crack identification and localization using a type of measured compressed response data through TL-based techniques. The findings suggest that TL can be effectively employed for crack identification in similar structural systems with varied sensor types. Future research should emphasize: •Collecting and conserving diverse datasets. • Providing synthetic data. • Integrating domain knowledge into ML model design. • Utilizing transfer learning.

Another vital aspect is data quality, a prevalent challenge for ML models. The SHM community lacks standardized methods for synthesizing and collecting datasets. This absence of systematic curation can lead to issues like anomalies in the data, potentially hampering ML model performance. This is especially true for ML techniques like logistic regression, less adept at handling noise. In essence, the typical data filtering process for ML models should be meticulously merged with the domain knowledge of structural health monitoring and assessment. Many challenges related to dataset quality can be addressed if precise collection and processing techniques are established and embraced. Consequently, Hsieh et al [1] emphasized that data quality and volume are pivotal in enhancing ML crack detection models. The development of accurate, robust, and swift ground-truth labelling, and data acquisition is paramount.

2.6.2 Overfitting in Machine Learning Models

Overfitted models, which result in subpar performance outside of the data used for training and/or testing, represent a significant challenge faced by the broader ML community. In essence, one of the causes of overfitting is training on and learning from inaccurate and noisy data entries within the collected dataset. Consequently, the model may not classify the data correctly. Overfitting can also arise from nonlinear and non-parametric methods that offer more flexibility in model construction. These methods can sometimes produce unrealistic models.

Conventional ML methods attempt to mitigate overfitting through techniques such as training/testing splits, bootstrapping, bagging, and k-fold cross-validation, among others. For instance, the stochastic methodology employed by RF to generate trees was specifically designed to address the overfitting issues associated with decision trees [120], [121].

It's worth noting that overfitting is not solely a concern during model training but also during model selection. A complex nonlinear model trained on a dataset with low-dimensional features might be prone to overfitting. Within the SHM community, incorporating domain knowledge can help alleviate overfitting problems, leading to more robust crack prediction and detection applications. Combining domain knowledge with data-driven methods, such as the approach used to handle data sparsity, could prove to be a potent mix. While overfitting has been extensively studied in the ML community, it may be even more crucial in SHM applications due to the complexity of some mechanics-based relationships that data-driven models aim to emulate. As a result, ML-SHM models typically require vast amounts of data, meticulous tuning, and enhanced noise filtering processes to minimize overfitting effects [80], [122].

2.6.3 Interpretability of ML Models

Perhaps one of the primary challenges associated with ML models is interpreting the physical significance of the model parameters and elucidating the effects of features. A commonly held belief is that ML models, especially the more advanced and intricate ones, function as black boxes. In essence, it's challenging to discern the mechanics-based relationship between features (inputs) and response variables (outputs) in data-driven models. To enhance model explainability, one should conduct feature importance tests to understand their impact on the model's output variables. These can then be benchmarked against fundamental principles.

Analysis of variance (ANOVA) [116] and statistical methods like the F-test [123] can be employed to evaluate the relative strengths of relationships between features and response variables. Additionally, tools such as individual conditional expectation curves (ICE), the partial dependence plot (PD plot), and its variants

are also prevalent [124], [125]. Beyond these feature importance measures, model-specific methods like the use of class activation mapping (CAM) to visualize focal areas in the imagery of CNN models [84], [126] have been developed.

Feature attribution or visualization enhances the interpretability and comprehension of black-box models. However, there are instances where it might be prudent to sacrifice model accuracy in favour of greater explainability[127]. Recent efforts in ML interpretability have demonstrated the benefits of integrating domain knowledge into ML strategies by incorporating a physics-based loss function. One approach is to embed hard constraints using a Lagrange multiplier into the loss function [126], [128]. This technique integrates a physics-based principle into the objective function, offering a way to elucidate certain ML models. A recent study in the domain of structural health monitoring merged unlabelled simulation data with observed labeled field data using a physics-guided neural network with a loss function [122]. The adopted method introduces additional conditions representing the discrepancy between observed and simulated outputs. Collectively, these studies [121], [129]–[135] underscore the importance of merging ML models with physics-based models in various SHM applications. This integration aids in interpreting the physical significance of model parameters and understanding the influence of features on the precision of the ML model. Nonetheless, the amalgamation of physics-based models and machine learning remains a formidable challenge, especially in the realm of structural crack performance assessment. This area warrants further exploration in future research.

2.6.4 Selection of Machine Learning Methods

A variety of algorithms were implemented in the presented ML-SHM studies. Unfortunately, there is no consensus or general approach regarding the selection of ML methods. In some studies, the researchers focused on a single method [136], [137], but they seldom provided a clear and cogent reason for their choice. Other studies have concentrated on comparing the performance of ML models developed using various methods. For instance, Hsieh et al. [1] compared the

performance of eight ML-based crack segmentation models using three-dimensional pavement images and consistent evaluation metrics under varied conditions to identify potential directions and challenges for future crack detection development. However, the results of such comparative evaluations are difficult to generalize because they are highly influenced by the adopted dataset, model testing (e.g., partition point for training- testing split, performance metric), and model training (e.g., whether k-cross-validation is adopted).

Future efforts are expected to focus more on analysing domain-specific characteristics of adopted datasets and applying knowledge-informed approaches in selecting ML algorithms, rather than using a solely performance-based search. For example, multi-output models are particularly useful in predicting baseline network curves because they can predict multiple response variables. Addressing some of the above challenges by creating well-organized and systematic datasets would also assist in method selection. The advantage of having such benchmark datasets is that a standard dataset will encourage particular attention to the integration of domain knowledge and associated data models. However, performance-based model selection is often the best solution when there is no understanding of how domain knowledge can be incorporated.

An immediate strategy that can be employed to guide method selection is to start by training and evaluating the performance of a linear (basis function) ML model logistic regression for classification problems, and (OLS, LASSO, ridge) for regression problems. Except for very specific problems, it has been demonstrated that linear models work reasonably well, are easy to implement (e.g., [138], [139]), and, more importantly, offer high model interpretation and transparency. If the initial linear models do not perform well, they should be further examined before moving to more complex models. For instance, poor performance may be due to the simplicity of the linear ML model or noisy data. The first situation requires a more advanced exploration (e.g., non-parametric models) capable of capturing the data complexity. However, noise problems within the data can be addressed by filtering.

2.7 Summary

This chapter discusses the application of ML in SHM and highlights several challenges that must be addressed to take full advantage of this technology. Recent advances in computational power have increased the capability of ML in handling complex, high-dimensional data. ML offers advantages in pattern recognition and function approximation for SHM. It is pointed out that the SHM community currently faces limited access to high-quality data sources, which is crucial for the development of ML models. To overcome this limitation, a unified effort is required to curate, collect, and generate datasets to an open-source storage area that can be used and populated by practitioners and researchers. Additionally, the creation of benchmark datasets for specific SHM sub-domains is suggested to align and focus research resources. ML models can be difficult to interpret and suggest the use of importance testing to better understand the individual effects of features on the response variable. They also propose the introduction of physics-based loss functions to offer insight into ML model training and interpretation and improve robustness. Overfitting is highlighted as an issue in ML models, particularly when capturing complex mechanics-based relationships in SHM problems. To address this issue, the examination of the SHM data space is recommended, along with physics-based validation and evaluation techniques. The chapter emphasizes the importance of combining data-based procedures with SHM domain knowledge to enhance performance and provide model insights. They suggest that future research should focus on finding ways to incorporate more knowledge-informed selection strategies when selecting ML models.

The potential benefits of ML in SHM are acknowledged, this chapter highlights several significant challenges that must be addressed. These challenges include the lack of high-quality data sources, difficulty in interpreting ML models, and the issue of overfitting. The chapter provides suggestions for addressing these challenges, but their successful implementation in practice remains to be seen.

3 Suitability analysis of Machine learning algorithms for crack growth prediction

3.1 Introduction

In the field of structural health monitoring, the selection of machine learning approaches is often done randomly without clear justifications, leading to suboptimal results. The choice of a machine learning model is crucial as it affects the performance and accuracy of the predictions. Numerous scholars confirm various factors that can aid in choosing the appropriate machine learning model for better accuracy and performance [2], [7], [140], [141].

1. **Problem type:** The type of problem you are trying to solve will dictate the type of model that you should use. For example, if you are solving a binary classification problem, logistic regression, decision trees, or random forests may be appropriate. If you are solving a regression problem, linear regression or support vector regression may be appropriate.
2. **Data characteristics:** The characteristics of the data, such as the size, distribution, and structure, can affect the performance of different models. For example, linear regression is appropriate for data that is linear in nature, while decision trees and random forests are appropriate for data that is non-linear.
3. **Computational Complexity:** The computational complexity of a machine learning model is an important factor to consider, especially for large and complex datasets. Some models, such as support vector machines and deep neural networks, can be computationally expensive, and may require significant amounts of time to train and make predictions.
4. **Hyperparameter tuning:** Hyperparameter tuning is a process of adjusting the parameters of a machine learning model to optimize its performance on a specific task. The parameters that are optimized in hyperparameter tuning are not learned from the data, but instead set prior to training the model. Proper hyperparameter tuning can help to significantly improve the performance of a machine learning model.

5. **Overfitting and Underfitting:** Overfitting and underfitting are common problems that can affect the performance of machine learning models. Overfitting occurs when a model is too complex and has learned the noise in the data rather than the underlying patterns. Underfitting occurs when a model is too simple and has not learned the patterns in the data. Proper validation techniques, such as cross-validation, can help to prevent overfitting and underfitting.
6. **Model Interpretability:** The interpretability of a machine learning model is an important factor to consider, especially for models that will be used in decision-making applications. Some models, such as decision trees and random forests, are easy to interpret, while others, such as neural networks, are more complex and difficult to interpret.

It's worth noting that, choosing the appropriate machine learning model for better accuracy and performance is a complex process that involves considering multiple factors. The problem statement determines the type of machine learning problem: supervised (labeled data) or unsupervised (unlabelled data). The target variable (output) determines the type of problem: categorical (classification), numerical (regression), or a set of input bunches (clustering). The size of the training data affects the risk of underfitting and overfitting, and the choice of low-variance high-bias (small data, many features) or high-variance low-bias (large data, few features) models. Linearity of data affects the choice of linear models (linear regression, SVM) or non-linear models (neural networks, random forests). The number of features also affects model selection, with SVM being a common choice for high feature, low observation data. The type of problem, the characteristics of the data, the performance metric, computational complexity, hyperparameter tuning, overfitting and underfitting, and model interpretability are some of the key factors that should be considered when selecting a machine learning model[7].

Therefore, this chapter analyses some of the most commonly used machine learning techniques in structural health monitoring studies. This analysis focuses more on regression machine learning algorithms that work well with small or

moderate datasets. This analysis can increase the probability of selecting suitable machine learning approaches and assist in interpreting and understanding the results of the model. By considering these factors and understanding the problem statement, a clearer path toward selecting the appropriate machine-learning approach can be determined.

3.2 Support Vector Regression

SVR is a powerful tool for regression analysis because it can handle non-linear relationships between the input and output variables. It does this by transforming the input data into a higher-dimensional feature space where the data becomes linearly separable. In this feature space, the SVR algorithm finds the optimal hyperplane that maximizes the margin between the data points and the hyperplane. This hyperplane is then used to make predictions for new data points[142], [143]. SVR is also a robust model because it has a built-in mechanism for handling outliers and noise in the data. This is done by introducing a "slack variable" that allows some of the data points to be on the wrong side of the hyperplane (i.e., they are misclassified). The algorithm then tries to minimize the sum of the slack variables, which helps to reduce the impact of outliers and noise on the model[142]–[147]. The SVR equation can be represented in the following form:

$$f(x) = wx + b + \varepsilon \quad (3-1)$$

Where:

$f(x)$ is the predicted output value for a given input x , w is the weight vector that defines the hyperplane, b is the bias term, and ε is the error term, also called the "residual" or "slack variable". In order to find the optimal values of w and b , the SVR algorithm minimizes the following objective function:

$$\text{Minimize: } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n (\varepsilon_i + \varepsilon_i^*) \quad (3-2)$$

subject to:

$$y_i - (wx + b) \leq \varepsilon_i \quad (wx + b) - y_i \leq \varepsilon_i \quad \varepsilon_i \geq 0 \quad (3-3)$$

Where:

C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the error, $\sum(\varepsilon_i)$ is the sum of the error terms for all data points, and y_i is the actual output value for a given input x_i .

The objective function tries to minimize the sum of the squared weight vector and the slack variable while satisfying the constraints on y_i and ε_i . In the case of non-linearly separable data, SVR uses a kernel function to transform the input data into a higher-dimensional feature space where it is linearly separable. The most popular kernel functions are the linear and radial basis function (RBF) kernels. The equation for the SVR model with the RBF kernel is:

$$f(x) = \sum(\alpha_i y_i K(x, x_i)) + b \quad (3-4)$$

where $K(x, x_i)$ is the RBF kernel, α_i are the Lagrange multipliers, and x_i, y_i are the training examples.

It's worth noting that the equation presented here is a simplified form of SVR, in practice the optimization problem to find the optimal values of w and b is a quadratic programming problem that can be solved by optimization solvers.

For small and moderate datasets, the optimization solvers that can be used to solve the quadratic programming problem in SVR [147]–[151] include:

1. **Sequential Minimal Optimization (SMO):** This is a popular optimization algorithm that is used to solve the quadratic programming problem in SVR. This algorithm is also a popular choice for small and moderate datasets as it is relatively simple to implement and fast to converge. The SMO

algorithm breaks down the large optimization problem into smaller subproblems that are easier to solve.

2. **Conjugate Gradient Descent:** This algorithm is well-suited for small and moderate datasets as it can converge quickly, and it is efficient in terms of computational time.
3. **Quasi-Newton methods:** These methods such as BFGS and L-BFGS are known for their fast convergence, and they can be a good option for small and moderate datasets as they are less sensitive to the initial guess.
4. **Stochastic Gradient Descent (SGD):** This algorithm is efficient for large datasets, but it can also work well with small and moderate datasets. It can be useful when data is non-linearly separable, and the dataset is too large to be handled by other optimization solvers.

It is worth noting that the choice of optimization solver will also depend on the characteristics of the data, such as the number of features and the presence of outliers. It may be necessary to try different optimization solvers to find the one that works best for a given problem.

When using SVR, there are several factors to consider [7], [141], [142], [145]:

1. **Choosing the appropriate kernel:** SVR uses a kernel function to map the input data into a higher dimensional space, where a linear regression can be performed. Different kernel functions can lead to different results, so it is important to choose the right kernel based on the data characteristics.
2. **Regularization parameter:** The regularization parameter C controls the trade-off between finding a smooth decision boundary and classifying the training points correctly. A high value of C leads to overfitting, while a low value of C may result in underfitting.
3. **Gamma parameter:** The gamma parameter controls the radial basis function used by the kernel. It affects the influence of each training example on the prediction. A high value of gamma results in a complex

model that may overfit the data, while a low value of gamma results in a simple model that may underfit the data.

4. **Scaling of the input data:** It is important to scale the input data before training an SVR model, as the magnitude of the input features can affect the performance of the model.
5. **Model performance evaluation:** It is important to evaluate the performance of the SVR model using appropriate metrics such as mean squared error or R-squared, and to use cross-validation techniques to avoid overfitting.
6. **Data imbalance:** SVR is sensitive to data imbalance, so it is important to handle data imbalance, either by oversampling the minority class or under sampling the majority class, or by using appropriate loss functions.
7. **Outliers:** SVR is sensitive to outliers, so it is important to detect and handle outliers before training the model.

These are some of the key factors to consider when using SVR. Proper consideration of these factors can help ensure the best performance of the SVR model for the specific data and problem at hand. SVR has some limitations too, one of them is that it is sensitive to the scale of the input variables, meaning that it can be affected by outliers if the input data is not properly scaled.

Additionally, it also has a high computational cost for large datasets and high-dimensional feature space. SVR is a powerful and robust regression model that can handle non-linear relationships between the input and output variables. It can be an effective tool for predicting continuous variables and is particularly useful in cases where the data has outliers or noise [147], [151]–[153].

3.3 Random Forest

Random Forest is a supervised learning algorithm used for both regression and classification problems. It is an ensemble learning method that combines multiple decision trees to produce a more accurate and stable prediction.

The main idea behind random forests is to generate multiple decision trees using bootstrapped samples of the training data and randomly selecting a subset of features for each split. The predictions from each tree are then combined to produce a final prediction for the entire forest[154], [155].

The prediction for a single tree in a random forest can be represented using the following equation:

$$y = \sum (w_i * 1[x \in R_i]) \quad (3-5)$$

where:

- y is the prediction for a given input x .
- w_i is the weight assigned to the i -th terminal node (or leaf node) in the decision tree.
- $1[x \in R_i]$ is an indicator function that returns 1 if x belongs to the region R_i and 0 otherwise:
- $\sum$ denotes the sum over all the terminal nodes in the tree.

The prediction for the entire random forest is obtained by averaging the predictions from all the trees in the forest. This can be represented using the following equation:

where:

- y is the prediction for a given input x .
- y_i is the prediction for the i – th tree in the forest.
- N is the number of trees in the forest.
- $\sum$ denotes the sum over all the trees in the forest.

The combination of multiple decision trees in a random forest helps to reduce the variance and increase the stability of the predictions. It also helps to mitigate the problem of overfitting, which is a common problem with decision trees. Additionally, random forests can handle a large number of features and non-linear

relationships between features and target variables, making them a popular choice for many machine-learning tasks. Random forests have several key benefits that make it a popular choice for machine learning problems. One of the main advantages is their ability to handle missing values and noisy data more effectively than decision trees. This is due to the random selection of features and samples in the training set, which reduces the impact of individual outliers or missing values. Random forests also offer a way to assess feature importance, which can be useful in identifying the most important features for predictions. Another benefit of random forests is their interpretability. Unlike some other machine learning algorithms, the prediction of a random forest is simply the average of the predictions made by each individual decision tree in the forest. This makes it easy to understand how the model is making predictions and to determine which features are the most influential[155].

It's worth noting that, when using random forests for regression problems, there are several factors to consider:

1. **Overfitting:** Like any machine learning algorithm, random forests can overfit the data if the number of trees is too large or if the trees are too deep. To prevent overfitting, it's important to use cross-validation to tune the number of trees and the maximum depth of each tree.
2. **Correlated features:** Random forests work best when the features are independent and uncorrelated. If the features are highly correlated, this can reduce the overall accuracy of the model. It's important to examine the correlation between features and, if necessary, remove or combine correlated features.
3. **Outliers:** Random forests are robust to outliers, but outliers can still have a large impact on the final predictions. It's important to identify and handle outliers before training the random forest.
4. **Feature scaling:** Random forests do not require feature scaling but scaling the features can sometimes improve the performance of the model.

It's important to consider the effect of feature scaling on the final predictions.

5. **Class imbalance:** If the response variable is imbalanced, the random forest may not perform well. It's important to balance the class distribution or use a weighted loss function to ensure that the model is not biased towards the majority class.
6. **Computational cost:** Random forests can be computationally expensive, especially when the number of trees is large, or the number of features is high. However, parallel implementations are available to help reduce computation time. Additionally, the number of trees in the forest can be controlled to strike a balance between accuracy and computation time.

By keeping these factors in mind, you can ensure that you are using random forests in the most effective way for regression problems[154]–[156].

3.4 Gradient Boosting Regression

Gradient Boosting Regression (GBR) is a powerful machine learning algorithm that can be used for both regression and classification problems. It is particularly effective for small and moderate datasets, where other algorithms such as random forests may struggle to provide accurate results. One of the key benefits of GBR for small and moderate datasets is its ability to model complex non-linear relationships between the input features and the output target. This is achieved by combining multiple weak models, such as decision trees, into a single strong model, using gradient descent optimization. Another advantage of GBR is its ability to handle missing values and noisy data. Unlike some other algorithms, such as linear regression, GBR can still provide accurate results even when the data contains missing values or noisy features [157]–[159].

However, GBR can also be computationally expensive, especially when the number of trees is large, and the number of features is high. To mitigate this issue, it is important to tune the hyperparameters of the GBR model, such as the learning rate and the number of trees, to achieve the best balance between accuracy and computation time [159], [160].

The GBR equation is based on the concept of gradient descent optimization, which is a method to minimize a loss function by iteratively adjusting the model parameters. The general form of the equation is:

$$f(x) = f(x) + h(x, \theta) \quad (3-6)$$

where $f(x)$ is the current approximation of the target function, $h(x, \theta)$ is a weak learner that models the residual error of the previous iteration, and θ represents the parameters of the weak learner.

In the case of GBR for regression problems, the target function is usually defined as the mean squared error between the predicted output and the true output. The weak learner is typically a decision tree that models the relationship between the input features and the residual error.

In each iteration, the residual error is calculated by subtracting the predicted output from the true output, and the parameters of the decision tree are adjusted to minimize the residual error. This process is repeated until the residual error reaches a minimum or a stopping criterion is met.

In the end, the final prediction is obtained by summing up all the weak learners' predictions. The final equation for the GBR model is:

$$f(x) = f_0(x) + h_1(x, \theta_1) + h_2(x, \theta_2) + \dots + h_T(x, \theta_T) \quad (3-7)$$

$$y_{new} = \frac{1}{k} \sum_{i=1}^k y_i \quad (3-8)$$

Where y_{new} is the predicted target value for the new data point. y_i is the target value of the $i - th$ neighbor found. K is the number of nearest neighbors chosen for the prediction. The distance metric is an important aspect of the K-Nearest Neighbor (KNN) algorithm[161]. The distance metric is used to determine the similarity between the test instance and the training instances. The most commonly used distance metrics for KNN regression include [162]:

1. **Euclidean Distance:** The Euclidean distance is the most commonly used distance metric in KNN regression. The equation for Euclidean distance is given by:

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2} \quad (3-9)$$

where x and y are two instances, x_i and y_i are the i -th feature of instances x and y , and n is the number of features.

2. **Manhattan Distance:** The Manhattan distance is another commonly used distance metric in KNN regression. The equation for Manhattan distance is given by:

$$d(x, y) = |x_1 - y_1| + |x_2 - y_2| + \dots + |x_n - y_n| \quad (3-10)$$

3. **Minkowski Distance:** The Minkowski distance is a generalization of the Euclidean and Manhattan distances. The equation for distance is given by:

$$D(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}} \quad (3-11)$$

where p is a positive integer that controls the weight of each feature. If $p = 1$, the Minkowski distance becomes the Manhattan distance, and if $p = 2$, the Minkowski distance becomes the Euclidean distance. It is important to keep in mind that the choice of the distance metric can greatly impact the performance of the KNN regression. The distance metric should be selected based on the characteristics of the data and the problem being solved. The performance of KNN regression is heavily dependent on the value of k . A small value of k can lead to overfitting, while a large value of k can lead to underfitting. To find the optimal value of k , cross-validation techniques can be used. KNN regression works well for small and moderate datasets because of its simplicity and ease of implementation. It does not require any prior assumptions about the distribution of the data and can handle non-linear relationships between the features and the target variable. However, KNN regression can be computationally expensive when working with large datasets, as the number of nearest neighbors needs to be calculated for

each test data point. Some considerations to bear in mind when applying KNN to regression problems [7], [162]–[165]:

1. **Scale of the data:** The scale of the data can affect the performance of KNN regression. It's a good practice to scale the data to a similar range before training the model.
2. **Number of neighbors (K):** The number of neighbors (K) is a critical hyperparameter that needs to be set before training the KNN model. A small value of K will lead to a more complex and flexible model, while a large value of K will lead to a smoother and less flexible model. It's important to select the value of K using cross-validation.
3. **Distance metric:** The distance metric used to measure the distance between the data points can affect the performance of the KNN regression model. Commonly used distance metrics include Euclidean, Manhattan, and cosine distance.
4. **Curse of dimensionality:** The curse of dimensionality is a well-known problem in KNN regression. As the number of features increases, the distance between the data points becomes increasingly large, making it more difficult for the model to identify patterns in the data. To mitigate this issue, dimensionality reduction techniques, such as PCA can be used.
5. **Outliers:** Outliers can have a significant impact on the performance of KNN regression. It's important to identify and handle outliers in the data before training the model.
6. **Time complexity:** The time complexity of KNN regression is $O(N)$, where N is the number of data points in the training set. This can be computationally expensive for large datasets, and alternative regression models, such as linear regression or decision trees, may be more suitable. It's crucial to be aware of several key factors when applying KNN to regression problems, such as the scale of the data, the number of neighbors, the distance metric used, the impact of high dimensions, the presence of outliers, and the computational time required.

3.5 Ridge Regression

Ridge Regression is a regularization technique that is used to address the problem of overfitting in linear regression models. It is a type of linear regression that adds a penalty term to the loss function to reduce the magnitude of the coefficients, which helps to avoid overfitting [166], [167]. The penalty term is defined as the sum of the squares of the coefficients multiplied by a scalar value, lambda, which is known as the regularization strength. The equation for Ridge Regression can be represented as:

$$y = X * \beta + \varepsilon \quad (3-12)$$

where y is the target vector, X is the feature matrix, β is the coefficient vector, and ε is the error term.

The goal of Ridge Regression is to minimize the residual sum of squares (RSS), while also shrinking the coefficients towards zero. The regularization term, represented by λ , is added to the RSS to give the following optimization equation:

$$\sum (y_i - x_i * \beta)^2 + \lambda * \sum \beta^2 \quad (3-13)$$

where λ is the *regularization* strength. The larger the value of λ , the greater the shrinkage of the coefficients [161], [163], [168], [169]. Ridge Regression has a closed form solution, which can be computed using linear algebra. The coefficients can be calculated as follows:

$$\beta = (X^T * X + \lambda I)^{-1} * X^T * y \quad (3-14)$$

where I is the identity matrix.

The regularization term helps to reduce the magnitude of the coefficients, which helps to avoid overfitting. The value of lambda determines the amount of regularization applied to the model. A high value of lambda results in a stronger regularization effect, which reduces the magnitude of the coefficients, while a low value of lambda results in a weaker regularization effect, which allows the coefficients to be larger. In Ridge Regression, the coefficients are estimated using an optimization algorithm that minimizes the objective function. The

optimization algorithm used is usually gradient descent or its variants. When using Ridge Regression to solve prediction problems, it's important to keep below factors in mind when using ridge regression to solve prediction problems with small and moderate datasets[7], [163], [169]–[171]:

1. **The scale of the data:** The scale of the data can affect the performance of the ridge regression model. It's important to standardize the data before using it in the model.
2. **The choice of regularization parameter:** The regularization parameter, also known as the hyperparameter, is a crucial factor that affects the performance of the ridge regression model. The regularization parameter determines the degree of shrinkage applied to the coefficients, and it should be carefully chosen to avoid underfitting or overfitting.
3. **The choice of the distance metric:** Ridge Regression can be used with different distance metrics, such as Euclidean distance, Manhattan distance, or Mahalanobis distance, depending on the nature of the data. The choice of the distance metric is crucial and can affect the performance of the model.
4. **The curse of dimensionality:** The curse of dimensionality is a phenomenon that occurs when the number of features is larger than the number of samples. In this situation, ridge regression may not perform well, and other techniques, such as feature selection or dimensionality reduction, may need to be applied.
5. **The presence of outliers:** Outliers can have a significant impact on the performance of ridge regression. It's important to identify and handle outliers before using the model.
6. **The time complexity:** The time complexity of ridge regression can be high for large datasets, especially if the number of features is large. It's important to consider the computational resources available before using ridge regression.

3.6 Lasso Regression

Lasso Regression is a linear regression method that combines the benefits of both Ridge Regression and Subset Selection. It aims to address the issues of overfitting and multicollinearity in the data by introducing a penalty term to the regression coefficients [7], [168]. The penalty term is a product of the absolute value of the regression coefficients and a scalar, lambda. The equation for Lasso Regression can be expressed as:

$$\text{minimize } f(\theta) = \sum_{i=1}^n (y_i - \theta^T x_i)^2 + \lambda \sum_{j=1}^m |\theta_j| \quad (3-15)$$

Where $f(\theta)$ is the objective function, y_i is the target value, $\theta^T x_i$ is the predicted value, θ is the regression coefficients, λ is the penalty term, n is the number of observations, and m is the number of features. The Lasso Regression algorithm aims to find the optimal value of θ that minimizes the objective function, subject to the constraint of the penalty term. The value of lambda determines the strength of the penalty term and affects the sparsity of the regression coefficients. A high value of lambda will lead to sparse coefficients, and a low value of lambda will lead to a less sparse solution [172]–[175]. It's important to carefully consider below factors and perform appropriate experiments to determine the optimal settings for a particular problem:

1. **Feature Selection:** Lasso regression can perform automatic feature selection by shrinking the coefficients of unimportant features to zero. This is beneficial when the number of features is large compared to the number of samples, as it can help to reduce the complexity of the model and prevent overfitting.
2. **Regularization Strength:** The regularization strength of lasso regression is controlled by the hyperparameter alpha, which determines the amount of shrinkage applied to the coefficients. A larger value of alpha will result in stronger shrinkage, which can reduce the variance of the model but increase the bias.
3. **Computational Complexity:** Lasso regression can be computationally expensive, especially for large datasets with many features. The choice of

solver and optimization method can affect the computational complexity of the model.

4. **Model Interpretability:** Lasso regression can provide interpretable results by selecting a subset of features and assigning non-zero coefficients to them, which can be helpful for identifying the important features for prediction.
5. **Outliers:** Lasso regression can be sensitive to outliers, as the regularization applied to the coefficients can reduce their magnitude. Outliers can have a significant impact on the results of lasso regression and should be detected and treated appropriately.

3.7 Neural networks

A Neural Network is a machine learning algorithm inspired by the structure and function of the human brain. It is designed to recognize patterns and make predictions based on the input data. Neural Networks consist of a series of interconnected processing nodes, also known as artificial neurons, that work together to perform a specific task. Neural Networks can be used for various types of problems, including classification, regression, image, and speech recognition, and more[176]–[178]. The input data is passed through a series of layers in the network, each of which applies a set of weights and biases to the data to produce a prediction. The weights and biases are adjusted during the training phase, where the network is provided with labeled data and the error is calculated. The goal is to minimize the error and improve the accuracy of the prediction.

There are several neural network architectures that can be used for regression problems with small and moderate datasets, including:

3.8.1. Single Layer Perceptron: This is the simplest form of a neural network, with a limited number of hidden layers. This architecture is known as Shallow neural network and can be used for linear regression problems. A basic building block of a shallow neural network is the artificial neuron or perceptron, which takes input values, processes them through a set of weights, and outputs a

prediction [54], [176]. The equation for a single perceptron can be represented as:

$$y = f(\sum w_i x_i + b) \quad (3-16)$$

where y is the output prediction, x_i is the i -th input, w_i is the weight associated with the i -th input, b is the bias term, and f is the activation function. In a shallow neural network, the input data is fed into an input layer, which passes the data through a series of hidden layers before finally arriving at an output layer. The hidden layers consist of artificial neurons, each with its own set of weights and bias. The weights and bias are updated during the training process to optimize the performance of the network. The activation function is a non-linear function that transforms the output of the perceptron into a prediction. Common activation functions used in shallow neural networks include sigmoid, hyperbolic tangent, and rectified linear unit (ReLU). Thus, a shallow neural network works by processing input data through a series of artificial neurons, each with its own set of weights and bias, to produce an output prediction. The network is trained to adjust the weights and bias to minimize the error between the actual and predicted outputs[54], [179], [180].

3.8.2. Multilayer Perceptron (MLP): is a type of feedforward neural network with one or more hidden layers. MLPs have been widely used in a variety of applications including image classification, speech recognition, and natural language processing. They have the ability to learn complex relationships between inputs and outputs and can handle large amounts of data. However, they can also be prone to overfitting, especially when the number of hidden nodes is large, and the dataset is small. To prevent overfitting, techniques such as regularization, early stopping, and dropout can be used. The activation function used in the hidden layer is typically a non-linear function such as the sigmoid or ReLU function [22], [180]. The equation for the activation function in the hidden layer of an MLP is given by:

$$f(x) = \max(0, x) \quad (3-17)$$

The ReLU activation function or the sigmoid activation function. Where x is the input data, and $f(x)$ is the activation value. The equation for the output layer of an MLP is given by:

$$y = \sum w_i * f(x_i) + b \quad (3-18)$$

where y is the predicted output, w_i is the weight for the i – th hidden node, $f(x_i)$ is the activation value for the i – th hidden node, and b is the bias term. MLP uses non-linear activation functions in the hidden layer to capture complex relationships between the inputs and outputs, and a linear combination of these activations in the output layer to make predictions. The weights and biases in the hidden and output layers are learned through training, allowing the network to make accurate predictions for unseen data. MLPs can be trained using different optimization algorithms, such as gradient descent, stochastic gradient descent, and Adam. The choice of optimization algorithm depends on the complexity of the problem and the size of the dataset[22], [177], [178], [181]. MLPs are considered a simple and flexible type of neural network that can be used for a wide range of regression problems with small to moderate datasets. They are easy to understand and implement, making them a popular choice for both researchers and practitioners.

3.8.3. Convolutional Neural Network (CNN): CNN is a type of deep neural network that is specifically well-suited for image and audio processing and have been successful in a variety of applications. However, it is important to consider the complexity of the problem, the type of data, and the desired output when choosing a CNN architecture. Proper data pre-processing and techniques to prevent overfitting such as regularization and early stopping are also important to achieve good performance[147], [182]. The network is composed of multiple layers including convolutional, pooling, and fully connected layers. The equation for the convolutional layer is given by:

$$y = (x * w) + b \quad (3-19)$$

where y is the output, x is the input, w is the convolutional kernel, and b is the bias term. The convolutional kernel is used to extract features from the input data, and the bias term is used to shift the output. The equation for the pooling layer is given by:

$$y = \max(x) \quad (3-20)$$

where y is the output, and x is the input. The pooling layer is used to reduce the spatial dimension of the data and control overfitting. The equation for the fully connected layer is given by:

$$y = \sum w_i * x_i + b \quad (3-21)$$

where y is the predicted output, w_i is the weight for the i – th node, x_i is the activation value for the i – th node, and b is the bias term. CNNs use convolutional layers to extract features from the input data, pooling layers to reduce the spatial dimension, and fully connected layers to make predictions. The weights and biases in each layer are learned through training, allowing the network to make accurate predictions for unseen data[183], [184]. CNNs have been widely used in a variety of applications including image classification, object detection, and image segmentation. They have been successful in learning complex relationships between inputs and outputs, especially when dealing with images and audio data. However, they can also be computationally expensive and may require large amounts of training data to perform well [128], [185]–[187].

3.8.4. Recurrent Neural Network (RNN): RNN is a type of neural network that are well-suited for sequential data. RNNs are particularly useful for time series data and sequential data such as speech, video, and text. The key feature of RNNs is their ability to model long-term dependencies in the data. RNN consists of a hidden state that is updated at each time step based on the current input and the previous hidden state[188]. The hidden state is used to make a prediction for the next time step. The key challenge with RNNs is the vanishing gradient problem, where the gradient used to update the parameters of the network becomes very small over time. This can make it difficult for the network

to learn long-term dependencies in the data. To address the vanishing gradient problem, RNNs can be combined with gating mechanisms such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs). These gating mechanisms allow the network to control the flow of information through the hidden state, allowing it to learn longer-term dependencies in the data[188]–[192]. The equation for a simple RNN can be expressed as follows:

$$h(t) = f(W(h(t-1)) + W(x(t)) + b) \quad (3-22)$$

where $h(t)$ is the hidden state at time t , $W(h(t-1))$ and $W(x(t))$ are the weights for the previous hidden state and the current input, b is the bias, and f is the activation function. Another important aspect of RNNs is their ability to handle variable-length sequences. In traditional feedforward networks, the input must have a fixed size. This can be a problem for sequential data, where the length of the sequences can vary. RNNs can handle sequences of varying lengths by using a special symbol to represent the end of the sequence. One of the main applications of RNNs is natural language processing (NLP)[189]. RNNs are used to model sequences of words, allowing them to capture the context and meaning of the words in a sentence. This can be used to perform tasks such as sentiment analysis, language translation, and text generation. Therefore, RNNs are a powerful tool for regression problems with sequential data. They can handle variable-length sequences, capture long-term dependencies in the data, and be used for a variety of applications such as NLP. To achieve good performance, it is important to consider the vanishing gradient problem and use techniques such as LSTM or GRUs to prevent it from affecting the performance of the network. Proper data pre-processing and techniques to prevent overfitting such as regularization and early stopping are also important to achieve good performance. Some of the factors that should be considered when using ANN with small or moderate datasets[189], [191], [192]:

1. **Overfitting:** Small datasets are susceptible to overfitting, where the model becomes too specialized for the training data, leading to poor generalization on new data. To prevent this, techniques such as regularization and early stopping can be used.

2. **Data Pre-processing:** The quality of the data is important in ANN. Careful pre-processing such as normalization, feature scaling, and data augmentation can help improve performance.
3. **Model Architecture:** A model with too many parameters can easily be overfit to a small dataset, while a model with too few parameters can have limited capacity. A good starting point is to use a simple, small network and gradually increase its complexity if necessary.
4. **Cross-Validation:** Cross-validation techniques can help prevent overfitting by evaluating the model's performance on unseen data.
5. **Feature Selection:** Feature selection techniques such as backward elimination and principal component analysis can be used to reduce the number of inputs, reducing overfitting, and making it easier for the model to generalize.
6. **Ensemble Methods:** Ensemble methods such as bagging, and boosting can be used to combine multiple models to improve performance.
7. **Transfer Learning:** Transfer learning, where pre-trained models can be used as a starting point, can be helpful when the dataset is small, as it enables the new model to take advantage of the knowledge and features learned from the pre-trained model. Transfer learning can be used to save time and reduce the computational cost of training a new model from scratch, especially when the pre-trained model is well suited for the new problem. For example, if the pre-trained model is a convolutional neural network that has been trained on a large image dataset, it can be used as a starting point for a new model that will be trained on a smaller dataset of images. The pre-trained model provides the new model with a set of learned filters and feature maps that can be used as a starting point, reducing the need for the new model to learn these features from scratch. The key to successful transfer learning is to choose a pre-trained model that is well-suited to the new problem, and to fine-tune the model to the new data. This typically involves retraining the model on the new data, updating the model's weights, and possibly modifying the architecture to better fit the new problem.

Thus, transfer learning is a powerful technique for addressing the challenges of small and moderate datasets, allowing models to take advantage of the knowledge and features learned from pre-trained models, reducing the need for large amounts of training data, and enabling the models to generalize better to new data. Table 3- 1 illustrates the most important requirements and parameters for all approaches that have been discussed in this chapter.

Table 3-1 The most important requirements and parameters for ML approaches

Approach	Requirements	Most Important Parameters
SVR	- Numeric Data - Feature Scaling - Handling Nonlinear Data - Tuned Hyperparameters	- Kernel type (linear, radial basis function, etc.) - Regularization parameter (C) - Kernel-specific parameters (gamma, degree etc.)
RF	- Numeric and categorical data - Missing data handling - Feature scaling not required - Robust to overfitting	- Number of trees (n_estimators) - Maximum depth of trees (max_depth) - Minimum samples per leaf (min_samples_leaf) - Maximum features per split (max_features)
GBR	- Numeric data - Feature scaling - Handling nonlinear data - Tuned hyperparameters	- Number of trees (n_estimators) - Learning rate (learning_rate) - Maximum depth of trees (max_depth) - Subsample fraction (subsample)

Ridge	- Numeric data - Feature scaling - Handles multicollinearity	- Regularization strength (alpha)
Lasso	- Numeric data - Feature scaling - Feature selection	- Regularization strength (alpha)
NNs	- Numeric Data - Large Datasets - Computational Resources - Extensive Tuning Required	- Number of layers, neurons per layer, activation functions. - Learning rate, batch size, epochs, early stopping, etc.
KNN	- Feature scaling - Distance Metric: Choice of distance metric (e.g., Euclidean, Manhattan, etc.) should be appropriate for the data. - Computation: Can be computationally expensive for large datasets.	- Number of Neighbors (k): Number of nearest neighbors to consider for prediction. - Weighting Scheme: Optional weighting of neighbors (uniform, distance-based, etc.). Algorithm used to compute nearest neighbors (auto, ball_tree, kd_tree, brute).

3.8 Summary

It is crucial for researchers to choose appropriate machine learning algorithms based on the type of data, study material, and desired outcomes. In addition, interpretability is a significant factor to consider. While highly accurate predictions are desirable, some machine learning models may lack interpretability, making it challenging to comprehend the reasoning behind the predictions or explain them to others. Hence, researchers should weigh both the predictive accuracy and interpretability of a model when selecting a machine learning algorithm for their needs. The significance of this chapter lies in its analysis of frequently employed machine learning algorithms within the realm of SHM. This analysis serves to equip aspiring researchers with the capability to discern appropriate ML algorithms for their specific issues. The subsequent section delves into comprehensive insights regarding the meticulous data collection and pre-processing procedures undertaken.

4 Data Collection and Preprocessing

4.1 Introduction

This chapter offers a detailed account of the experimental methodology employed in prior investigations, specifically targeting the examination of three distinct materials: 2024 Sheet Aluminium, ABS, and concrete. The chapter encompasses an elaborate description of the experimental scheme and setup employed for each material, encompassing crucial parameters and configurations utilized during the experiments. Furthermore, the chapter delves into the data processing procedures, emphasizing the exploration of correlations between the collected features and the predicted value of crack depth.

The contribution of this chapter lies in its provision of valuable insights into the experimental procedures utilized in previous studies, particularly focusing on the exploration of various materials [193]–[196]. Additionally, the chapter highlights the significance of data processing techniques in uncovering correlations between the collected features and the predicted crack depth.

4.2 Experimental scheme and setup

4.2.1 Aluminum

Experimental specimens of aluminum were prepared with different crack depths (0.25 mm, 0.5 mm, and 1 mm) in a rectangular shape as shown in Figure 4-1. The crack was intentionally positioned at 4.5 mm away from the fixed end, as it resulted in maximum stress concentration in the fillet region. The aim was to analyse the behavior of the aluminum beam under uniform and non-uniform temperatures, with heating rates of 2 °C /min, 5 °C /min, and 8 °C /min applied to study the effect of heating duration on modal parameters.

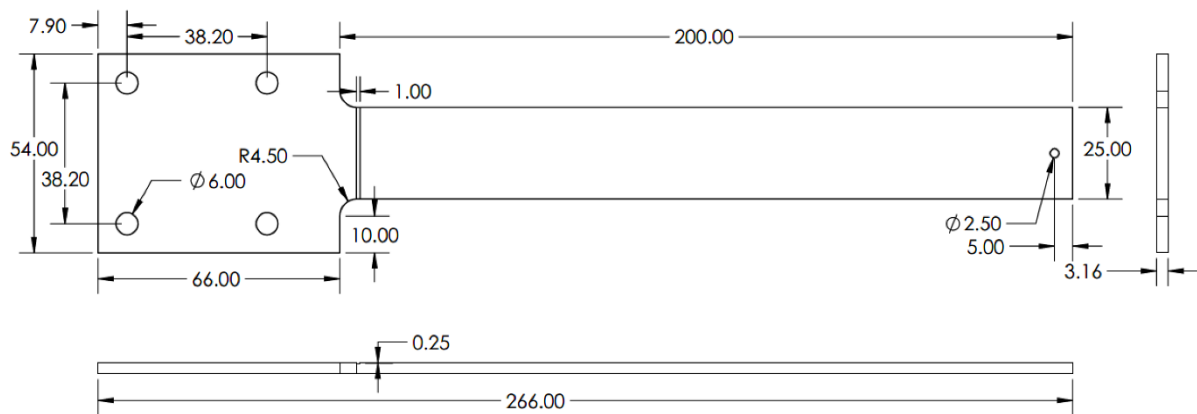


Figure 4-1 The Geometry of Aluminum Beam

The experimental setup consisted of three units. The vibrating mechanism included a signal generator, power amplifier, and mechanical shaker, which provided controlled vibrations to the aluminum specimens. The thermal unit incorporated a thermal couple, PID temperature controller, and thermal mat to regulate the heat supply and maintain the desired temperatures. The data acquisition unit involved an accelerometer, data acquisition card, and National Instrument Signal Express software, enabling the analysis of the dynamic response of the system. By monitoring the structural response using the accelerometer, valuable insights were obtained regarding the behavior of the aluminum specimens under different thermal conditions.

4.2.2 ABS

The experimental setup for the ABS tests included a shaker, signal generator, power amplifier, and accelerometer. The shaker provided the necessary vibrations to the specimens, while the signal generator and power amplifier controlled the vibration parameters. The accelerometer captured the acceleration data during the experiments, providing crucial information about the specimens' behavior. Furthermore, a digital microscope was utilized to monitor the crack depth during the tests, facilitating a comprehensive understanding of the fracture behavior of the ABS beams.

In the case of ABS material, the experimental scheme was divided into two main parts: bending vibration tests and dynamic mechanical analysis (DMA) tests. Cantilever beams made of ABS were subjected to constant thermal loads using

a mica band heater. The initial step involved measuring the natural frequency of the beams using an accelerometer. This measurement helped determine the optimal maximum amplitude for the beam based on the resonant frequency, significantly reducing the experiment time. In order to enhance stress concentration and capture the dynamic response of the beam, an initial seeded crack was strategically placed near the fixed end. The beam used in the experiments had a thickness of 3 mm. The seeded crack had a width of 0.2 mm and a depth of 0.5 mm as shown in Figure 4-2. This positioning ensured maximum stress concentration at the crack site. To monitor and record the dynamic behavior of the beam, a lightweight accelerometer from PCB® company was attached to the free end of the beam. This accelerometer, weighing only 0.6 g, was selected for its ability to accurately measure and capture the acceleration data during the experiments. Subsequently, the beams were continuously vibrated at this resonant frequency, and the number of cycles until fracture was recorded, aiming to identify the optimal combination of 3D printing parameters that resulted in the highest number of cycles before fracture.

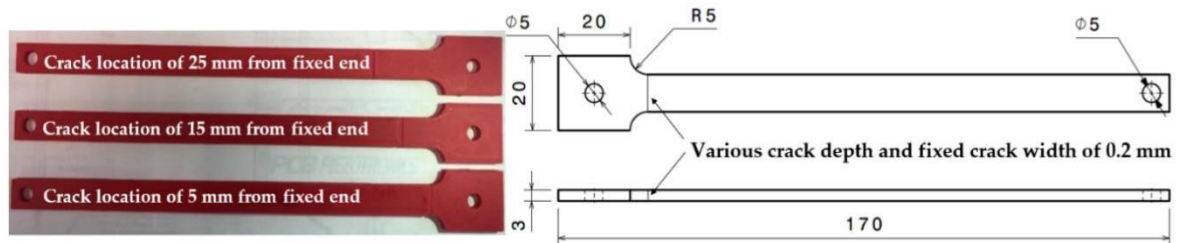


Figure 4-2 The Geometry of ABS Beam

In the DMA tests, specimens were cut from the fractured beams and analysed using a Q800 instrument. These specimens were clamped in the chamber of the instrument, and the temperature was ramped up at a rate of 3.00 °C /min. The storage modulus of the ABS specimens was then measured under a load oscillation frequency of 1 Hz within the temperature range of 30 to 70 °C. This allowed for an assessment of the mechanical properties of the ABS material at different temperatures.

4.2.3 Concrete

In regard to concrete specimens, the experimental investigation involved a cantilever beam configuration, as depicted in Figure 4-3. The beam had a thickness of 25 mm, and an initial seeded crack with widths and depths of 2 mm, 4 mm, and 6 mm was located near the fixed end of the beam to achieve maximum stress concentration. The crack dimensions were precisely measured at a distance of 5 mm from the fixed end, while an extra-lightweight accelerometer was affixed to the free end of the beam.

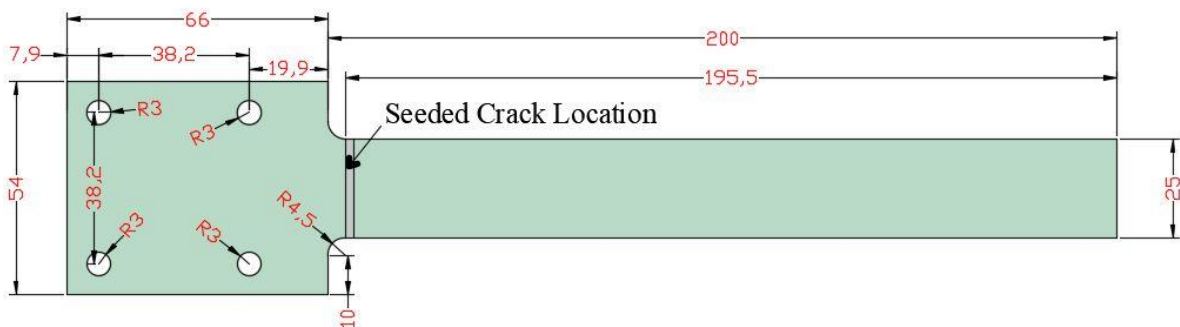


Figure 4-3 The Geometry of Concrete Beam

To explore the behavior of different concrete mixes, including steel fibre reinforced concrete (SFRC), several mixtures were prepared, each varying in seeded crack depths and fibre contents. The dynamic response testing procedure entailed subjecting the SFRC sample to controlled vibrations using a shaker, which generated a sinusoidal output signal synchronized with the fundamental natural frequency of the specimen. Additionally, a temperature load was applied using a mica band heater, while real-time acceleration data was acquired through the accelerometer.

The experimental process was conducted in sequential stages, comprising the determination of the fundamental natural frequency, the generation of a sinusoidal motion at the identified frequency with a defined displacement amplitude, and the subsequent analysis of the resulting displacement amplitude to assess the characteristics of the seeded crack. It was observed that the continuous force vibration led to variations in crack depth, consequently influencing the dynamic response behavior of the entire system. Throughout the experiment, the displacement amplitude displayed in real time by the Signal

Express program served as a criterion to pause or resume the procedure. These steps were consistently applied to all the tested seeded crack beams. The all-experimental setup is illustrated in Figure 4-4. More details about the experimental setup and methods can be found in references [193]–[198].

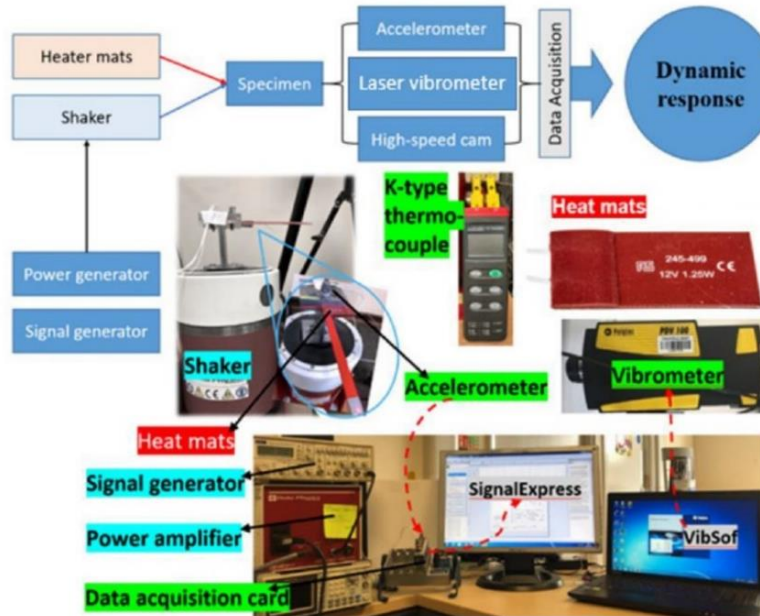


Figure 4-4 The experimental setup

4.3 Signal processing and Data collection

The modal response of both ABS, and the aluminum beam were measured using an accelerometer. The acquired acceleration and time data were processed in National Instrument Signal Express and imported into MATLAB for further analysis as shown in Figure 4-5.

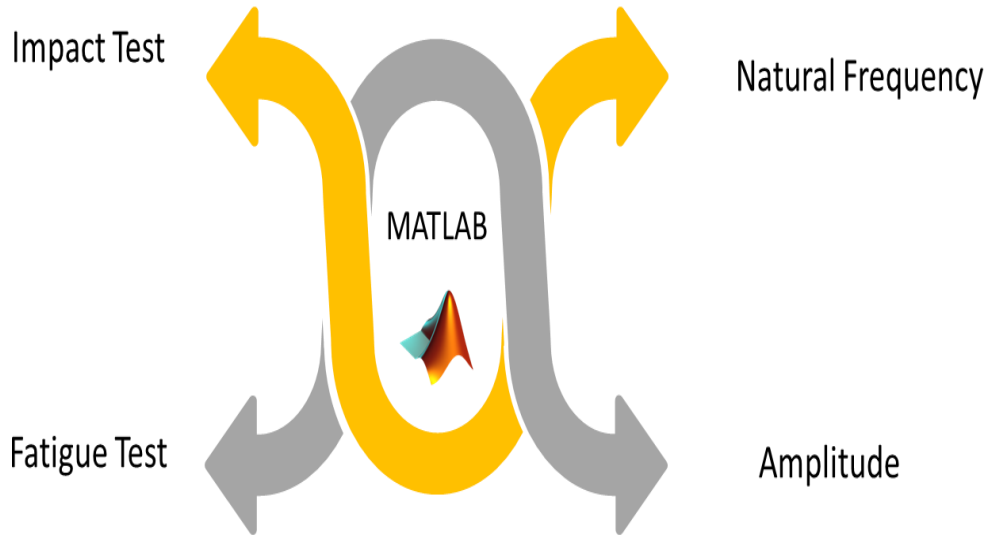


Figure 4-5 Signal Processing

The natural frequency and modal amplitude were computed using specific equations based on the acquired data [193]–[195]. The equation 4-1 computes the natural frequency of the specimen. where, t_i and t_j represent the time instances of the i^{th} and j^{th} peak amplitudes. The vibrations occurring at the natural frequency were responsible for the initial highest amplitude of the beam's oscillation, which effectively reduced the duration of the experiment. As shown in equation 4-2, the amplitude of displacement at the crack tip was determined by a quadratic integral connection with the accelerometer-measured amplitude of acceleration. Table 4.1 indicates experimental conditions and parameters[194], [198].

$$f = \frac{i - j}{t_i - t_j} \quad (4-1)$$

$$y_i(L) = \frac{1}{2} \frac{acc_{i,peak} - acc_{i,trough}}{(2\pi f_i)^2} \quad (4-2)$$

Table 4-1 The Experimental Conditions and Parameters.

Material	Temperature C ⁰	Crack Location mm	Structural Response
Composite Control Mix SFRC	20 40 60	5	Natural Frequency
Polymer ABS	20 50 60 70	5 15 25	
Metal aluminum 2024-T3	20-25 50-100 150-200	4.5 5 10	Amplitude

4.4 Data Preparation for Machine learning models

In this study, Jupyter Notebook was utilised as a tool for creating and sharing documents that contain live code, equations, visualizations, and narrative text. It is commonly used by practitioners and researchers in the fields of data analysis, machine learning, and scientific computing. Before using a dataset for machine learning, data analysis and feature extraction techniques (shown in Figure 4-6) are employed to select the most relevant and informative features for the task at hand. Techniques such as visualizing the correlations between the features and the target variable are useful, and visualization libraries such as Pandas' scatter matrix and Altair which, is a declarative visualization library for Python. It supports a wide range of chart types, including bar charts, line charts, scatter plots, and heatmaps. It also provides support for faceting and layering, which allows users to create more complex visualizations.

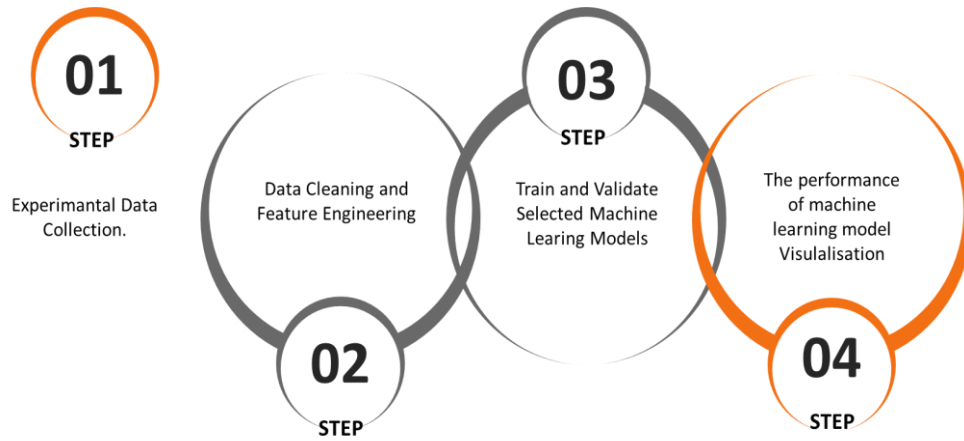


Figure 4-6 The Flow Chart of Data Analysis Steps

These tools can aid in understanding data and selecting a suitable machine learning model. However, it's important to consider the nature of the different machine learning models and how they handle different types of problems. The research utilized experimental data of three materials collected from previous studies [193], [195], [197], [198]. The experimental data consists of four features: temperature C^0 , crack location mm, amplitude mm, natural frequency Hz, and a predicted value: crack depth mm. Pre-processing of the data was conducted using libraries such as Pandas and NumPy to clean and format the data, perform basic statistical analysis, and use techniques such as Variance Threshold to drop features that have constant or near-constant values. The data were also scaled using libraries such as Standard Scaler to improve the training time of the model and ensure that the features were at a comparable scale. According to Chatterjee [199], Spearman's correlation coefficient is alternative measures of correlation that can capture non-linear relationships and are suitable for ranking or ordinal data [200]–[202]. Spearman's correlation coefficient (ρ) is calculated based on the ranks of the variables. Let's denote the ranks of X and Y as $R(X)$ and $R(Y)$, respectively. The Spearman correlation coefficient is given by:

$$\rho = 1 - [(6 * \Sigma(D^2)) / (n * (n^2 - 1))] \quad (4-3)$$

Where: Σ denotes the sum of the respective values.

D represents the difference between ranks ($R(X) - R(Y)$). n is the number of data points. The value of ρ ranges between -1 and 1, where: $\rho = 1$ indicates a perfect monotonic increasing relationship. $\rho = -1$ indicates a perfect monotonic

decreasing relationship. $\rho = 0$ indicates no monotonic relationship. Table 4-2 presents spearman's correlation coefficient between natural frequency, amplitude, temperature, crack location and crack depth for three materials. Upon examination of the compiled data, several observations can be discerned: In the case of ABS material, the most profound correlation a negative one is apparent between the parameter 'Natural Frequency' and the predicted depth of the crack. This relationship suggests that as 'Natural Frequency' increases, the anticipated crack depth decreases. When considering the Aluminum material, similar to ABS, 'Natural Frequency' exhibits the most substantial negative correlation with the forecasted crack depth. However, it is also noteworthy to point out the existence of a robust positive correlation between 'crack location' and the predicted crack depth. This indicates that as 'crack location' values rise, so does the anticipated depth of the crack. For Concrete material, the correlations between the various parameters and the predicted crack depth appear to be comparatively tenuous. Among these, 'Natural Frequency' displays the most pronounced albeit negative correlation. This proposes a converse relationship between 'Natural Frequency' and the predicted crack depth. Each of these observed correlations provides distinct insights into the influence exerted by these parameters on the predicted crack depth for the respective material types. These correlations underline the importance of considering different parameters when analysing the impact on crack progression across various material types.

Results show that ABS, concrete, and aluminum exhibit similar patterns in natural frequency, with frequency decreasing as crack depth increases (as shown in Figure 4-7) in a nonlinear manner. Additionally, the impact of temperature on natural frequency is more pronounced in aluminum than in concrete and ABS. This is due to the higher elastic modulus and greater consistency of isotropic sheet metal properties relative to additive layer manufactured ABS and concrete [7], [112].

Table 4-2 Spearman's correlation coefficient between features and crack depth

Material	Features	ρ Value
ABS	<i>Crack Location</i>	<i>0.1089</i>
	<i>Temperature</i>	<i>0.0294</i>
	<i>Natural Frequency</i>	<i>-0.7580</i>
	<i>Amplitude</i>	<i>-0.2906</i>
Aluminum	<i>Crack Location</i>	<i>0.7839</i>
	<i>Temperature</i>	<i>-0.7439</i>
	<i>Natural Frequency</i>	<i>-0.8755</i>
	<i>Amplitude</i>	<i>0.5577</i>
Concrete	<i>Elastic value</i>	<i>-0.0229</i>
	<i>Natural Frequency</i>	<i>-0.4257</i>
	<i>Amplitude</i>	<i>-0.1229</i>
	<i>Temperature</i>	<i>-0.0309</i>

As crack depth and temperature increase in aluminum specimens, the amplitude also increases, consistent with the expected effect of reduced natural frequency resulting from a crack in the specimen.

In contrast, ABS exhibits the opposite behavior: as crack depth increases, natural frequency drops similarly to aluminum, but amplitude decreases instead of increasing, as shown in Figure 4-8. This discrepancy can be explained by the different temperature conditions during the experiments. Maximum temperature for aluminum specimens is 200°C, well below the temperature threshold for any significant material transformations during the short test duration. In contrast, the maximum temperature for ABS samples is 70°C, which is close to the glass transition temperature.

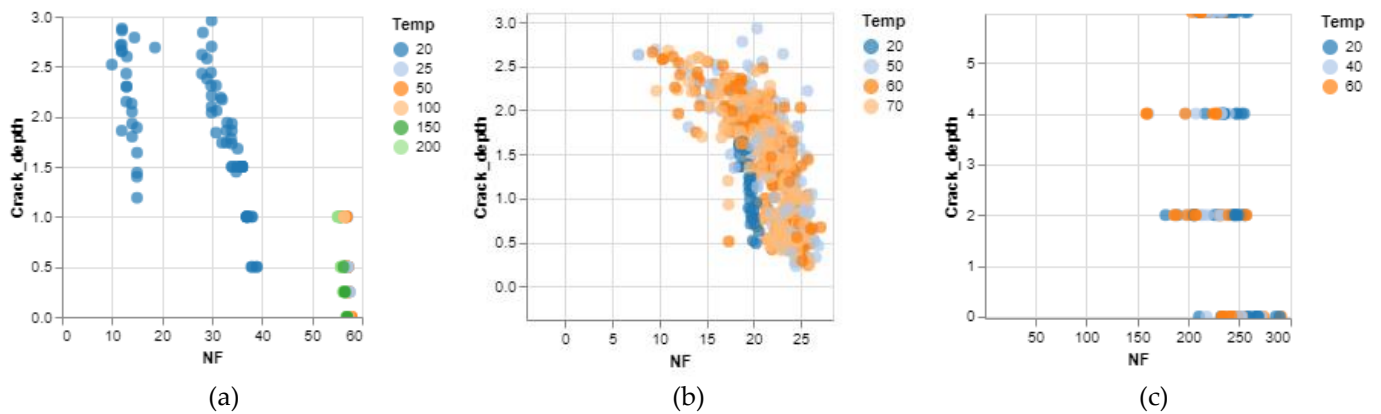


Figure 4-7 The correlation between crack depth (mm) and natural frequency (Hz) for a- Aluminum b-ABS c- Concrete.

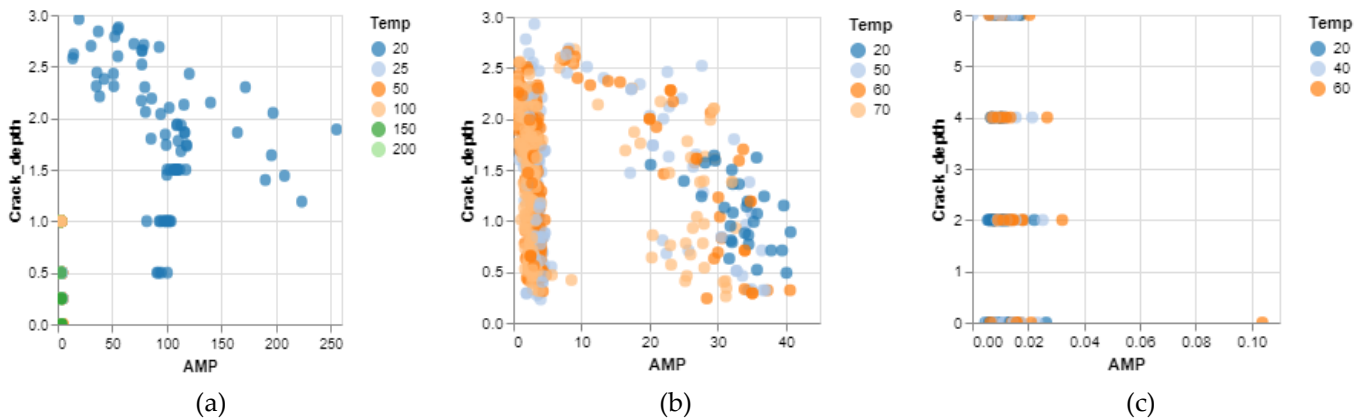


Figure 4-8 The correlation between Crack depth (mm) and amplitude (mm) for a- Aluminum b- ABS c- Concrete.

Figure 4-9 provides a more comprehensive view of the frequency-amplitude relationship in the experimental data, revealing an inverse relationship between frequency and amplitude in aluminum and concrete, while ABS shows a positive relationship. No clear relationship is evident between crack depth and position in ABS and concrete data, but in aluminum specimens, increasing crack position results in increased amplitude and decreased frequency.

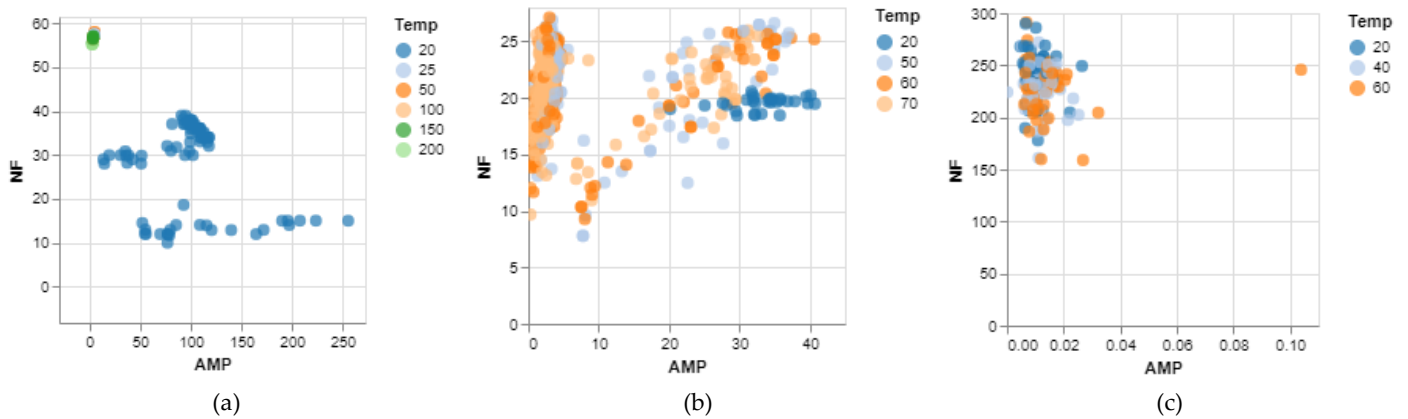


Figure 4-9 The correlation between natural frequency (Hz) and amplitude (mm) in a- Aluminum b-ABS c- Concrete.

After cleaning and preparing the data, they are typically split into a training set and a validation set. The training set is used to train the model, while the validation set is used to evaluate the performance of the model on unseen data, ensuring that the model generalizes well to new, unseen data.

4.5 Training and validation Datasets Division

To enable proper comparison of the coefficients and account for different material behavior, three identical but independent regression models were trained - one for Aluminium, one for concrete, and one for ABS. The collected data (as shown in Appendix C, D, and E) were randomly and blindly split into training and test sets with a 70/30 split. The choice of a 70% training data split ensures a sufficiently large dataset for training the machine learning models. Having a substantial amount of data for training helps the models capture the underlying patterns and variations in crack propagation accurately. It provides a robust foundation for the models to learn from and develop predictive capabilities. Allocating 30% of the dataset to the validation set allows for a substantial portion of data to evaluate the trained models' performance. Sizable test data helps assess the generalization ability of the models, determining how well they perform on unseen data. This split ensures a meaningful evaluation of the models' predictive accuracy and provides statistical confidence in their performance metrics. The 70/30 split strikes a balance between mitigating overfitting and underfitting issues. If the training set were too small (e.g., 50/50 split), the models might not have enough data to learn effectively, potentially resulting in poor

generalization and overfitting. Conversely, if the training set were too large, the risk of underfitting increases as the models may not adequately capture the complexity of crack propagation patterns. The 70/30 split attempts to optimize this balance[2], [203]–[206]. Model hyperparameters were adjusted to minimize mean squared error and achieve satisfactory performance.

This approach ensured that the model was always evaluated on previously unseen data, making the evaluated performance more representative of a real-world predictive problem. To prevent any compromise to the model's performance due to breaking the "test on unseen data" principle during training, a K-fold cross-validation methodology was utilized. This methodology is illustrated in Figure 4-10 and ensured that the model was evaluated using unseen data during training. In the subsequent chapter, the performance of the proposed machine learning models will be evaluated. This evaluation will primarily focus on the features delineated in Table 4-2. These features have been selected due to their strong relevance and potential impact on the performance of the models.

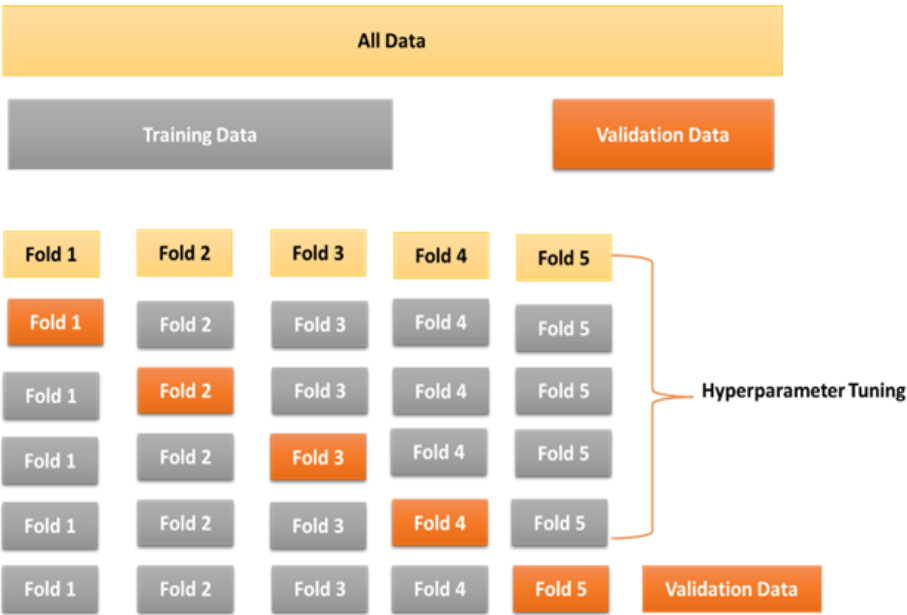


Figure 4-10 5-Fold cross validation schematic.

4.6 Model Training and Evaluation

It is important to note that the benefits and drawbacks mentioned above are general considerations for the respective machine learning models. The specific performance of ML models in predicting crack propagation would depend on factors such as the quality and size of the dataset, appropriate feature selection, and careful hyperparameter tuning [7], [8]. To enable a proper comparison and account for the diverse material behavior discussed earlier, three identical but independent models were trained and validated with unseen data. The proposed models were evaluated for training and validation accuracy using a MSE metric. MSE is a commonly used statistical metric for evaluating the performance of regression models. MSE measures the average squared difference between the predicted values and the actual values of the target variable. For each data point in the test set, the model generates a predicted value based on the input features. The squared difference between the predicted value and the corresponding actual value is calculated. These squared differences are then averaged across all data points in the test set as shown in Equation . MSE provides an estimate of the average squared error between the predicted values and the true values. It gives higher weight to larger errors due to the squaring operation. The value of MSE is always non-negative, with lower values indicating better performance. A value of 0 indicates a perfect fit, where the predicted values match the actual values exactly. In the context of the study, MSE was used as an evaluation metric to assess the performance of the machine learning models in predicting crack propagation. Lower MSE values indicate that the model's predicted crack propagation values closely match the actual values, suggesting better predictive accuracy.

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (4-4)$$

Where $\hat{y}_i$ represents the predicted value for the i^{th} observation. y_i represent the actual value for the i^{th} observation. It's worth noting that while MSE provides a useful measure of model performance, it is not the only metric available. Depending on the specific objectives of the study, other metrics such as mean absolute error (MAE), R-squared (coefficient of determination), or domain-

specific metrics could also be considered to evaluate and compare the performance of the models. Essentially, these metrics represent the average predictive error of the models[7].

4.7 Summary

This chapter provides a comprehensive overview of the experimental methodology used in previous studies to investigate three different materials: 2024 Sheet Aluminium, ABS, and concrete. The chapter describes in detail the experimental design and setup employed for each material, highlighting the important parameters and configurations utilized during the experiments. Furthermore, the chapter focuses on the data processing procedures, with a specific emphasis on exploring the correlations between the collected features and the predicted crack depth. The chapter identifies the most influential predictive features for each material. For example, it is found that temperature has a more significant impact on the natural frequency of aluminum compared to concrete and ABS. This difference can be attributed to the higher elastic modulus and consistent isotropic properties of aluminum sheet metal, in contrast to the additive layer manufactured ABS and concrete. The forthcoming section aims to present significant findings and discussions regarding the performance of all the proposed models in predicting crack depth.

5 Results and Discussions: Results and Discussions: Application of the proposed ML Models

5.1 Introduction

This chapter delves into the exploration and implementation of various machine learning models to predict crack propagation in different materials such as ABS, aluminum 2024-T3, and concrete, based on datasets that have discussed in previous chapter. The models discussed include Ridge Regression, Lasso Regression, K-Nearest Neighbors (KNN), Support Vector Regressor, Random Forest Regressor, and Gradient Boosting Regressor. Each of these models has been carefully chosen for their respective capabilities to efficiently handle different aspects of the data, like linearity, multicollinearity, non-linear relationships, and high-dimensional data.

5.2 Machine learning Models

The selection of suitable machine learning models for the given dataset is a critical decision that revolves around the dataset's characteristics and the specific objectives of the analysis. The dataset under consideration comprises four to five features, and its size ranges from small to moderate. This section will delve into the rationale behind choosing machine learning models that align with the data's attributes.

Regarding the datasets pertaining to ABS and aluminum 2024-T3 alloys, the following models were chosen: Ridge Regression, Lasso Regression, and KNN. These models were selected with careful consideration of the dataset's features and size. Ridge Regression, a linear regression model with L2 regularization, emerged as a suitable candidate due to its ability to counteract overfitting and address multicollinearity among the features. Notably, its computational efficiency and provision of interpretable coefficients are advantageous. Lasso Regression, similar to Ridge Regression, utilises L1 regularization to encourage sparsity in coefficients and facilitate feature selection. Due to its capacity for automated identification of pertinent features, particularly within the domain of predicting crack propagation, Lasso Regression emerges as a judicious and

valuable selection. However, both Ridge and Lasso Regressions may struggle with intricate nonlinear patterns, while also encountering challenges posed by multicollinearity. KNN algorithm, operating as a non-parametric approach, was also integrated. Its reliance on the proximity of training samples in the feature space makes it adept at accommodating complex relationships present in the data. KNN's simplicity in both understanding and implementation is noteworthy. However, sensitivity to the choice of the number of neighbors (K) and the distance metric, as well as its computational demands during prediction, require consideration. These chosen models present simplicity, ease of interpretation, and efficient performance, rendering them ideal for swift execution or deployment on resource-constrained systems[7], [8], [158], [195].

Regard to composite data, the crack depth of each concrete type was clustered based on their respective elastic modulus, which varied with changes in temperature. The researcher selected Support Vector Regressor, Random Forest Regressor, and Gradient Boosting Regressor to predict crack depth for composite data based on their ability to handle complex datasets with non-linear relationships between the features and the target variable and many hyperparameters that can be fine-tuned to improve performance. The Random Forest Regressor, an ensemble learning method that combines multiple decision trees, was utilised. It handles high-dimensional data effectively, mitigates overfitting, and captures complex relationships between input features and crack propagation. However, Random Forest Regressor can be computationally expensive, especially for large datasets. It may struggle to identify subtle patterns and can be challenging to interpret due to its ensemble nature. SVR was also utilised, as it is effective in handling both linear and nonlinear relationships. It can manage high-dimensional data and, by selecting appropriate kernel functions, capture complex patterns. However, SVR can be sensitive to the choice of hyperparameters and kernel functions. It may also face limitations in dealing with very large datasets due to computational requirements. Gradient Boosting Regressor, which constructs an ensemble of weak prediction models to sequentially correct errors, was another model used. It handles complex relationships well, performs admirably with high-dimensional data, and typically

exhibits high prediction accuracy. However, Gradient Boosting Regressor can be computationally intensive and may require careful hyperparameter tuning. Without proper regularization, it is also prone to overfitting [208], [209]. The chosen machine learning models are aligned with the specific attributes of the datasets and the targeted analysis objectives. The chosen machine learning models have been thoughtfully tailored to the data characteristics and specific requirements. By acknowledging the data's size, feature count, linearity, and complexity, these models offer a balance between predictive power and interpretability. Nevertheless, the trade-offs associated with computational demands and sensitivity to hyperparameters warrant cautious optimization and selection of the final model.

5.3 Application of the proposed ML Models Results

Table 5-1 summarizes the performance metrics for training and validation (with all features) predictions for the three materials.

Table 5-1 The evaluation of training and validation for selected models.

Material	Learning model	Training MSE “mm”	Validation MSE “mm”	Baseline model “mm”
Aluminum	K- Neighbors Regressor	0.189	0.079	0.6838
	Lasso	0.180	0.15	
	Ridge	0.170	0.14	
ABS	K- Neighbors Regressor	0.195	0.086	0.4398
	Lasso	0.195	0.182	
	Ridge	0.194	0.182	
Concrete	Gradient Boosting Regressor	4.100	0.597	4.971
	SVR	4.401	3.114	
	Random Forest Regressor	3.903	0.509	

5.3.1 Training and Validation with all Features

ABS Models: The results show that the K Neighbors Regressor model has a Training MSE of 0.195, indicating that, on average, the model's predictions are off by 0.195 squared units from the actual values in the training set. The validation MSE of the same model is 0.086, indicating that the model's predictions are off by 0.086 squared units from the actual values in the test set. These values indicate that the K Neighbors Regressor model performs well on both the training and test sets, suggesting that it has not overfit to the training data. The Lasso model's training MSE is 0.195, and the validation MSE is 0.182. Both values are lower than the MSE of the baseline model (0.4398), indicating that the Lasso model outperforms the baseline. The results for the Ridge model show that it has a Training MSE of 0.194 and a validation MSE of 0.182. These values are also lower than the MSE of the baseline model (0.4398), indicating that the Ridge model outperforms the baseline. Overall, all three models (K Neighbors Regressor, Lasso, and Ridge) outperform the baseline model, suggesting that they have learned meaningful patterns in the data and can make reasonably accurate predictions. Figure 5-1 shows the actual against predicted of ABS crack depth for selected machine learning models in case of all features.

Aluminum Models: The results show that the K Neighbors Regressor model has a training MSE score of 0.189 and a validation MSE score of 0.079, which is lower than the MSE of the baseline model (0.8907), indicating that the K Neighbors Regressor model outperforms the baseline. The lower the MSE score, the better the performance of the model. The Ridge model has a training MSE score of 0.170 and a validation MSE score of 0.149, both of which are lower than the MSE of the baseline model. The Lasso model has a training MSE score of 0.180 and a validation MSE score of 0.159, both of which are also lower than the MSE of the baseline model. In all cases, the validation MSE score is lower than the MSE of the baseline model, suggesting that the models have learned meaningful patterns in the data and can make reasonably accurate predictions. Figure 5-2 shows the actual against predicted of aluminum crack depth for selected machine learning models.

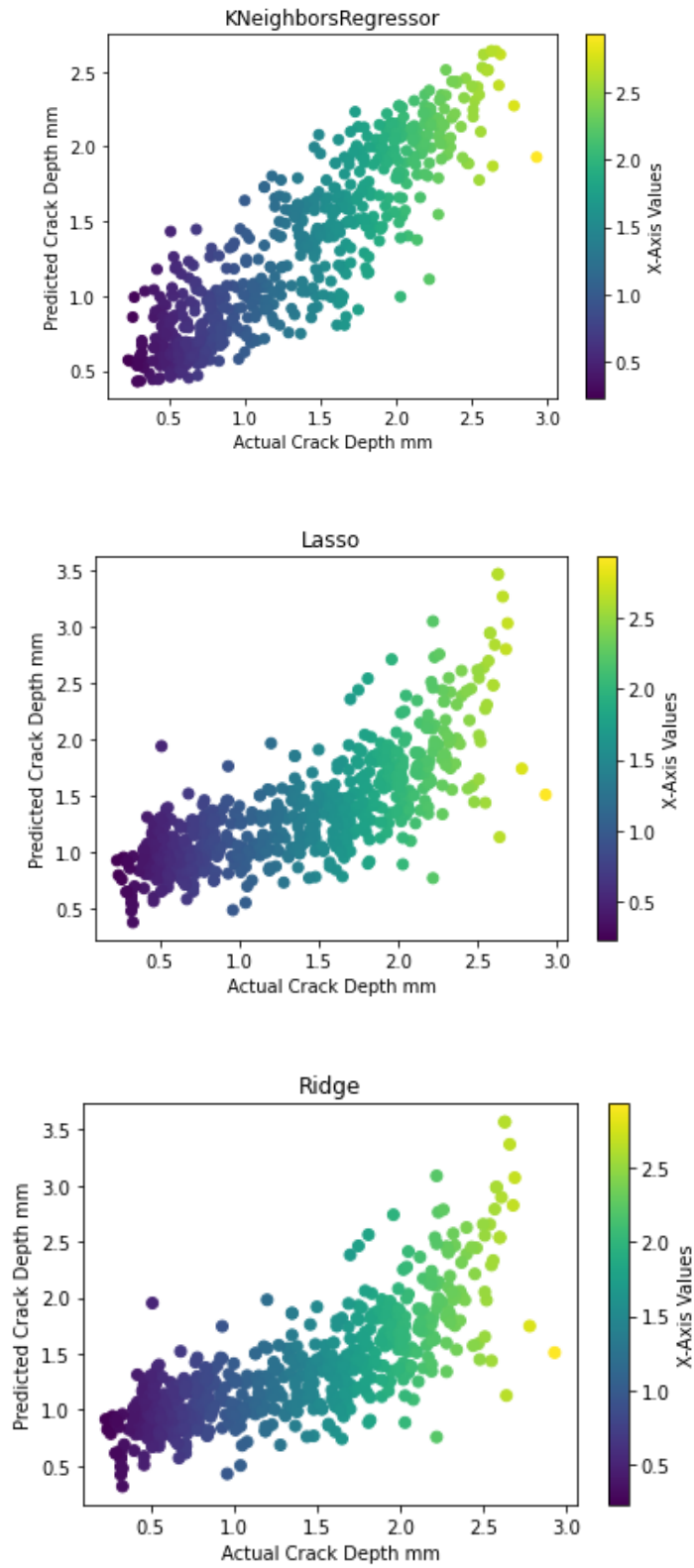


Figure 5-1 The actual against predicted of ABS crack depth in case of all features.

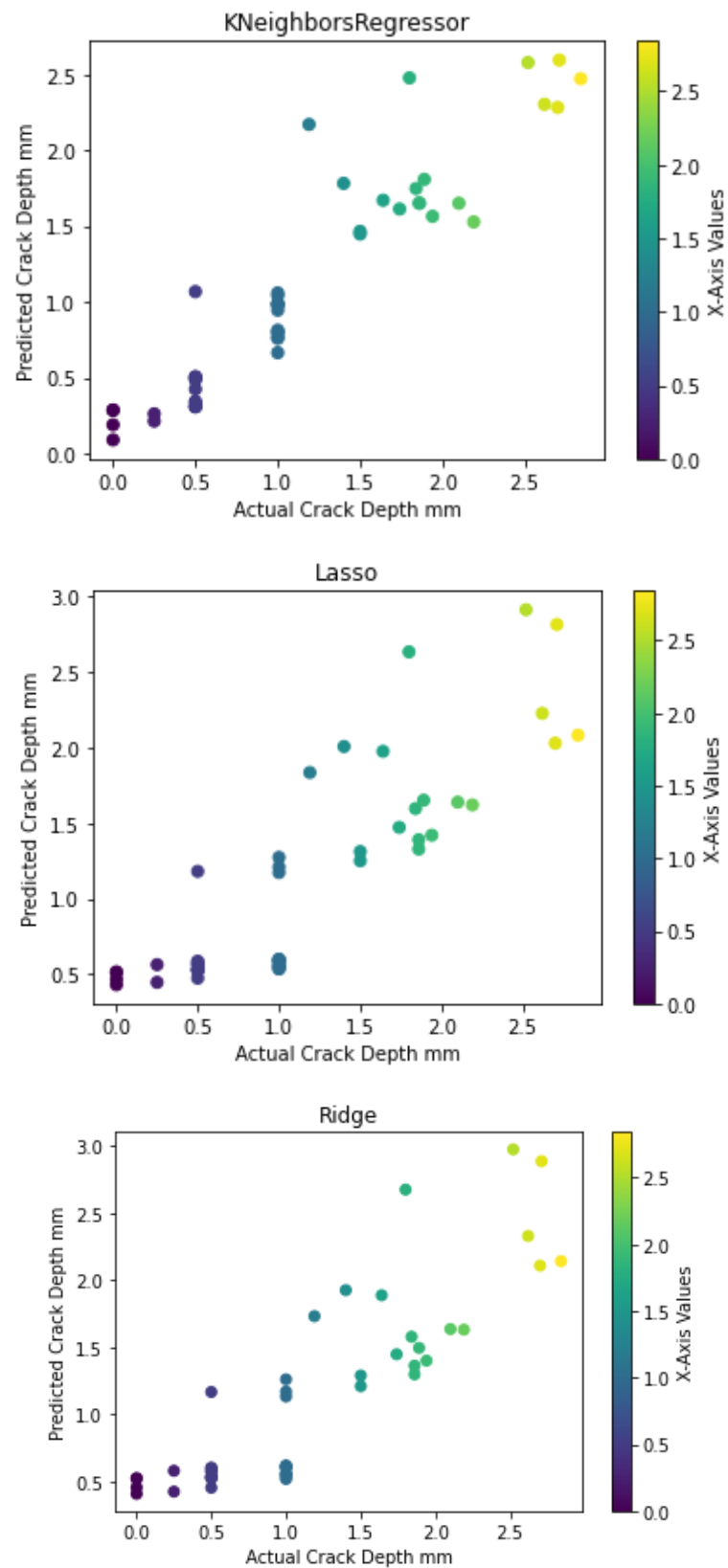
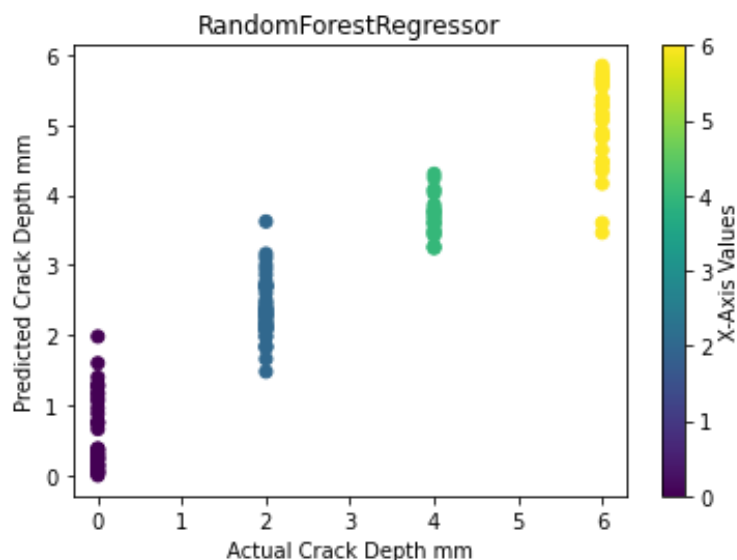


Figure 5-2 The actual against predicted of Aluminum crack depth for selected machine learning models.

Concrete Models: For all models (Gradient Boosting Regressor, SVR, and Random Forest Regressor), the Mean Squared Error (MSE) is used to evaluate their performance, which measures the average squared difference between the predicted and actual values. The lower the MSE, the better the performance of the model. The results show that the Gradient Boosting Regressor model has a training MSE of 0.208 and a validation MSE of 0.014, indicating that the model performs well on both the training and test sets and has not overfit to the training data. The SVR model has a training MSE of 0.182 and a validation MSE of 0.094, indicating that the model's predictions are off by 0.182 and 0.094 squared units from the actual values in the training and test sets, respectively. The Random Forest Regressor model has a training MSE of 1.649 and a test MSE of 0.189. Although the Test MSE is lower than the baseline model, it is higher than the Test MSE of the other two models, suggesting that the model may have overfit to the training data. Overall, the Gradient Boosting Regressor and SVR models seem to be the best performers based on these results. They have the lowest Test MSE values and are less likely to overfit compared to the Random Forest Regressor model. Figure 5-3 shows the actual against predicted of Concrete crack depth for selected machine learning models.



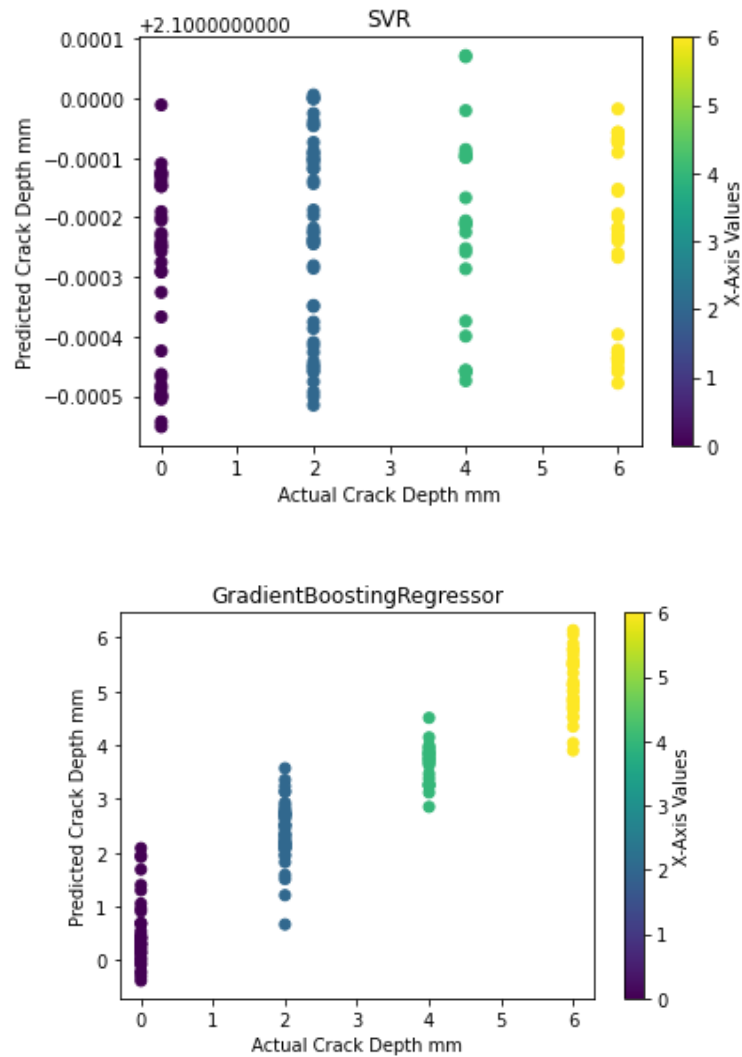


Figure 5-3 The actual against predicted of Concrete crack depth for selected machine learning models.

5.3.1 Training and Validation with Selected Features

ABS Model: In the analysis of the prediction models' performance, the K Neighbors Regressor, Lasso, and Ridge regression techniques were employed to predict a target variable based on selected features, namely natural frequency, and amplitude. . Figure 5-4 shows the actual against predicted of ABS crack depth for selected machine learning models based on selected features. The K Neighbors Regressor model demonstrated a comparatively superior performance as evidenced by the training MSE of 0.196 and the test MSE of 0.097. These low MSE values suggest that the model was successful in capturing underlying

patterns within the data, resulting in more accurate predictions. This indicates that the model's capacity to identify relationships between the selected features and the target variable was effectively harnessed.

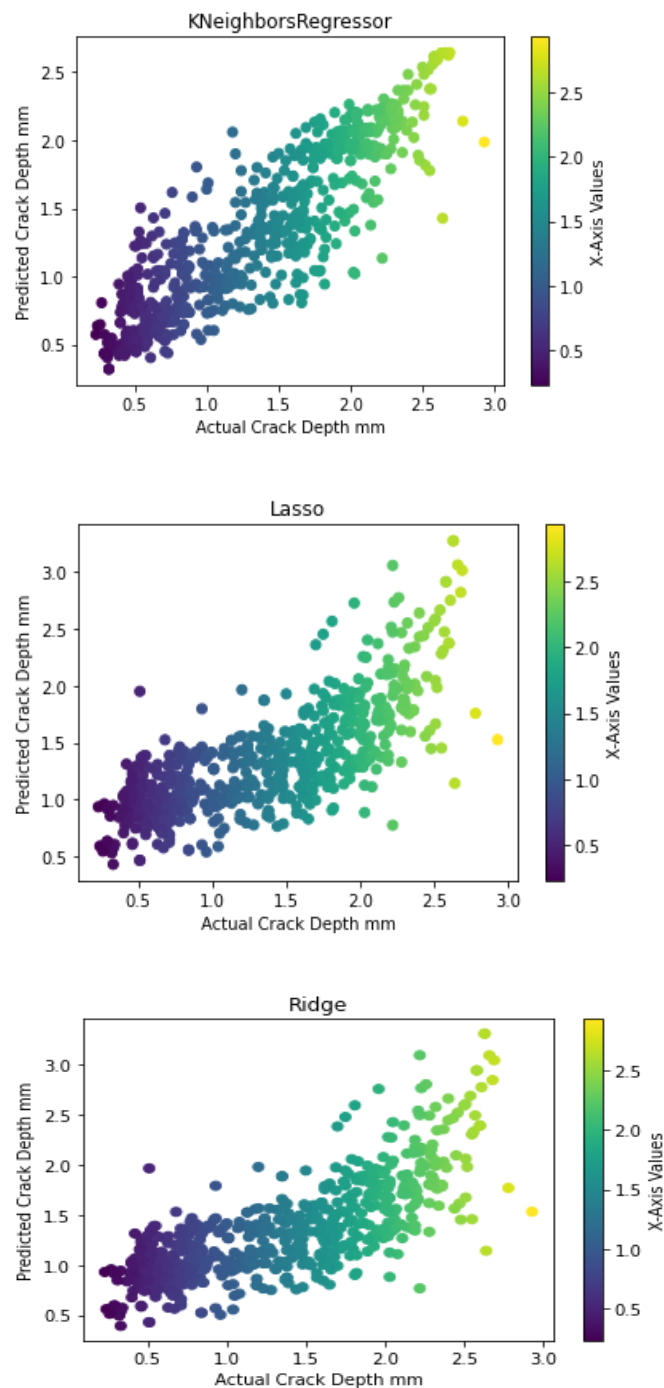


Figure 5-4 The actual against predicted of ABS crack depth based on selected features.

The K Neighbors Regressor's test MSE being substantially lower than that of the baseline (0.437) implies a meaningful enhancement over a naive prediction approach. On the other hand, both Lasso and Ridge regression models yielded similar results in terms of test MSE, both recording a value of 0.186. While their performance in the test phase was comparable, it's important to note that both models outperformed the baseline model, indicating a successful utilization of regularization techniques to optimize the predictive performance. However, it's worth highlighting that the training MSEs for Lasso and Ridge (as shown in Table 5-2) were slightly better than that of the K Neighbors Regressor, implying that the latter model may have encountered some degree of overfitting in its training phase. In contrast, Lasso and Ridge models, which incorporate regularization, managed to strike a balance between fitting the training data and generalizing to unseen data.

Table 5-2 The evaluation of training and validation based on selected features.

Material	Learning model	Training MSE “mm”	Validation MSE “mm”	Baseline Model “mm”
ABS	K- Neighbors Regressor	0.196	0.097	0.437
	Lasso	0.194	0.186	
	Ridge	0.189	0.186	
Concrete	Gradient Boosting Regressor	1.209	0.217	2.799
	SVR	1.376	1.374	
	Random Forest Regressor	1.054	0.141	

Concrete Model:

The models were trained and tested on a selected set of features, primarily focusing on natural frequency as shown in Figure 5-5 . The Random Forest Regressor exhibited promising results, achieving a training MSE of 1.054 and a Test MSE of 0.141. This suggests that the model generalizes well to unseen data, as the test error is relatively low. Comparing this to the baseline MSE of 2.799,

the Random Forest Regressor demonstrates its ability to provide significant improvements in prediction accuracy.

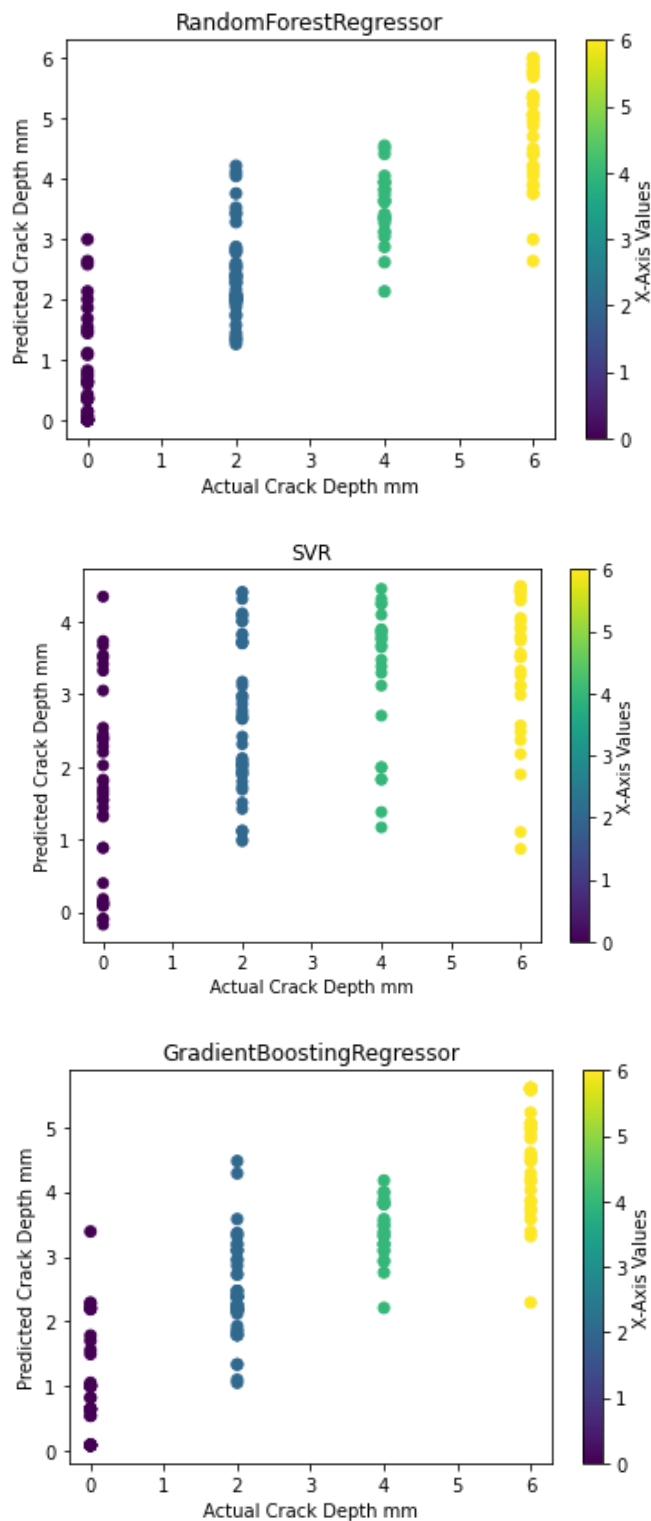


Figure 5-5 The actual against predicted of Concrete crack depth based on selected features.

Its ensemble approach harnesses the power of decision trees and randomness, enabling it to capture complex relationships within the data effectively. On the other hand, the SVR model exhibited higher training and test MSE values of 1.376 and 1.374, respectively. These results indicate that the model may be overfitting to the training data, as it struggles to generalize to the test data. The hyperparameters of the SVR model should be further optimized to potentially enhance its predictive performance. While the SVR model's performance is currently suboptimal compared to the Random Forest Regressor, fine-tuning its hyperparameters and possibly exploring different kernel functions could lead to improved results. The Gradient Boosting Regressor showed competitive performance with a training MSE of 1.209 and a Test MSE of 0.217. This suggests that the model effectively balances bias and variance, resulting in relatively low errors on both training and test sets. The hyperparameters of the Gradient Boosting Regressor should also be investigated to determine whether further optimization can lead to better results. Given that the test MSE is considerably lower than the baseline MSE, the Gradient Boosting Regressor proves to be a valuable model for predicting outcomes based on the selected natural frequency feature. The Random Forest Regressor and Gradient Boosting Regressor models display strong predictive capabilities on the selected natural frequency feature, outperforming the baseline MSE. While the SVR model initially struggles with overfitting, it presents an opportunity for improvement through hyperparameter optimization. The choice between these models should be driven by a comprehensive analysis of their respective strengths and weaknesses and the specific requirements of the predictive task at hand.

5.4 Discussion

The most important predictive features for Aluminium are natural frequency, crack position, and temperature. For ABS and concrete, natural frequency and amplitude dominate, while the position of the crack appears to be of little importance. This may seem counterintuitive, as theory suggests that the closer the crack is to the fixed position of the specimen, the more pronounced the effect on dynamic response. However, the effect of crack position is captured by the model under the natural frequency and amplitude terms, resulting in a small and

relatively unimportant crack position coefficient. Using all of the features as predictors yields the lowest MSE but removing crack location does not significantly increase the error in ABS and concrete models. Removing amplitude has a much more significant effect on the error, increasing it by approximately 60% relative to the all-features baseline. The most significant effect is seen by removing natural frequency from the feature pool, which increases the error significantly. This suggests that natural frequency is the dominant feature in the prediction of crack depth, and its effect on the model error is greater than the rest of the features combined. In comparing the results of the studies presented with other related research, it becomes evident that the identified predictive features for different materials, such as Aluminium, ABS, and concrete, are consistent with previous findings in the field. Natural frequency and amplitude emerge as a significant factor in predicting material behavior, aligning with established theoretical expectations [110], [112], [193], [195], [210]–[212]. The reinforcement of this feature's importance in the current study further solidifies its relevance in material characterization. The computational time and complexity associated with training algorithms exhibit variations contingent upon several factors, including dataset size and algorithmic implementation. When dealing with small to moderate datasets, linear regression techniques such as Ridge and Lasso demonstrate efficient training times, characterized by a complexity of $O(p^3)$, where p corresponds to the number of features. In contrast, the kNN algorithm, which is non-parametric in nature, requires minimal training time due to its reliance on storing data points. However, the computational expense arises during prediction when KNN necessitates distance calculations, resulting in a complexity of $O(n d)$, with n referring to the number of training instances and d representing the number of features. SVR, an algorithm based on support vector machines, generally exhibits reasonable training time for small to moderate datasets. Nevertheless, the process of solving the quadratic programming problem can become time-consuming for larger datasets. Random Forest Regressor, an ensemble algorithm, exhibits efficient training for small to moderate datasets with a complexity of $O(M * n * \log(n))$, where M denotes the number of trees and n represents the number of training instances. On the other hand,

Gradient Boosting Regressor, another ensemble algorithm, typically entails higher computational requirements, especially when considering larger numbers of iterations. Consequently, its complexity generally surpasses that of Random Forests. The selected algorithms can typically achieve training within an acceptable timeframe for small and moderate datasets. However, the actual time and complexity involved are contingent upon specific dataset characteristics, implementation nuances, and the availability of computational resources [7], [8], [161], [195], [197], [198]. In a critical analysis of table 5-1 and 5-2, it's evident that model performance varies with material types and learning algorithms. While some models consistently perform well across materials (e.g., K-Neighbors Regressor on ABS), others exhibit material-specific effectiveness. The choice of model and its hyperparameters should be tailored to the specific material characteristics and the trade-off between training and generalization. Furthermore, hyperparameter tuning for models as SVR might alleviate overfitting concerns is in line with recent research by [23], [145], [151], [210], [213], [214]. These tables emphasize the importance of selecting appropriate models for different materials and highlight the need for rigorous evaluation and optimization to ensure accurate predictions in material science applications. The models utilized in this research, including KNN, ridge regression, and lasso regression, are generally considered to be white-box models [143], meaning that they are relatively easy to interpret and understand. These models make their predictions based on a set of weights or coefficients that are applied to the input features and can be examined, to understand how the model is making its decisions. This makes it possible to understand the importance of each feature and how it contributes to the overall prediction, as well as to identify any potential biases in the model. However, it is important to note that even white-box models can be difficult to fully interpret, especially when working with large or complex datasets. It may also be challenging to fully understand the reasoning behind the model's predictions when working with more complex models, such as deep neural networks. In these cases, it may be necessary to rely on other methods, such as feature importance analysis or model agnostic techniques such as LIME [215]–[217], to gain a better understanding of the model's decision-making process. All

proposed models may have some capability usage for crack prediction with small datasets, they each have their own limitations and may not be the best choice for all cases. It may be useful to try multiple approaches and compare their performance, in order to determine the most effective method for the given data.

5.5 Summary

The chapter presents an in-depth explanation of how each model operates, highlighting their unique strengths and weaknesses, and the implications of these characteristics in handling ABS, aluminum, and concrete datasets. The crack depth in each concrete type is also examined based on the variations in their respective elastic modulus as the temperature changes. The chapter provides comprehensive insights into the performance metrics used for evaluation, specifically the MSE, while drawing attention to the relevance of feature selection and hyperparameter tuning in determining the specific performance of the models. In addition to the selection and application of suitable machine learning models, the chapter introduces formal verification techniques used to assess the models' robustness and reliability, including formal specification, model checking, sensitivity analysis, and statistical testing. It delves into how these techniques are adapted to the unique requirements of predicting crack propagation in varying conditions. The chapter concludes with a discussion on the computational efficiency and complexity associated with each of these machine learning models, considering factors such as the size of the dataset and algorithmic implementation. By illustrating the various methods and metrics used in model selection, assessment, and comparison, this chapter aims to provide a thorough understanding of how machine learning can be leveraged to predict crack propagation in different materials, offering valuable insights into the practicalities of implementation and performance evaluation. However, while the machine learning models in this chapter demonstrated promising predictive capabilities, it's important to recognize their limitations, particularly in addressing the complex dynamics of crack propagation. This limitation serves as a driving force behind our exploration automated machine learning in the subsequent chapter, wherein we tackle the intricate challenge of enhancing structural health monitoring through a more specialized and refined predictive framework.

6 Automated Prediction of Crack Propagation Using H2O AutoML: Towards Enhanced Structural Health Monitoring

6.1 Introduction

Crack propagation is a critical phenomenon in materials science and engineering, with implications for structural integrity, reliability, and safety in various applications. Accurately predicting crack propagation behavior is essential for ensuring the structural performance and longevity of engineering components as investigated in previous chapter. However, there is a growing need for automated models that can efficiently and accurately predict crack propagation. The aim of this chapter is to develop a machine learning-based automated model that can accurately predict crack propagation behavior in any materials. By leveraging the powerful capabilities of the H2O library, the aim is to harness the potential of machine learning algorithms to analyse complex crack propagation patterns and provide reliable predictions. The H2O library is a popular and versatile tool for machine learning in Python. It offers a wide range of algorithms, automated hyperparameter tuning, and feature engineering capabilities, making it well-suited for our crack propagation prediction task. By utilizing the H2O library, we can streamline the process of model development and leverage its efficiency and scalability for handling large datasets [218]–[221]. This research contributes to the field of crack propagation prediction by demonstrating the effectiveness of an automated approach using the H2O library. By automating the crack propagation prediction process, this chapter aims to enhance the efficiency and accuracy of crack propagation analysis, thereby facilitating improved structural integrity assessment and maintenance strategies. In the following sections, we provide an overview of existing crack propagation prediction methods and highlight the limitations of traditional approaches. We discuss the relevance of automated models in addressing these limitations and explore the potential of the H2O library in predicting crack propagation.

The methodology, experimental results, and implications of this research will be discussed, followed by a comprehensive analysis of the findings. Through this

work aims to advance the understanding of crack propagation in ABS materials and provide valuable insights for practical applications in engineering and materials science.

6.2 Previous Academic Work

Crack propagation prediction has been a subject of extensive research in the field of materials science and engineering. Various approaches have been explored to understand and predict crack growth behavior, ranging from analytical methods to numerical simulations. However, these traditional approaches often face limitations in capturing the complexities of crack propagation, highlighting the need for automated models that can overcome these challenges and provide accurate predictions. Analytical methods, such as fracture mechanics principles and stress intensity factor (SIF) analysis, have long been used to estimate crack growth behavior. These methods provide valuable insights into crack behavior under simplified conditions. However, they often rely on simplifying assumptions and are limited in their ability to account for complex crack geometries and heterogeneous materials [222]. As a result, the accuracy of predictions is compromised when dealing with real-world scenarios. Numerical methods, particularly finite element analysis (FEA), have been widely employed to simulate crack propagation [18], [222]–[226]. FEA-based models offer more realistic representations of material behavior and crack growth dynamics. However, these models require significant computational resources and manual intervention in defining crack paths, making them time-consuming and less efficient for large-scale applications [227]–[232].

The application of ML techniques for damage and crack detection in engineering materials has witnessed a burgeoning interest in recent years, driven by the quest for enhancing safety and structural integrity across a wide range of industrial applications. Numerous studies have demonstrated the efficacy of ML in crack detection across various materials. For instance, Wang et al [106] introduce a fatigue crack growth calculation method based on extreme learning machine (ELM), radial basis function network (RBFN), and genetic algorithms optimized back propagation network (GABP). Regan et al [233] study harnesses statistical

features for acoustics-based damage detection in enclosed cavities like wind turbine blades, utilizing machine learning algorithms such as Logistic Regression (LR) and Support Vector Machine (SVM) for decision-making via binary classification. Experimental data from a wind turbine test rig demonstrated that a Cubic SVM can effectively generalize to various damage types, locations, and severities with over 98% accuracy, highlighting the potential of this approach in the advancement of SHM. Miorelli et al [234] leverage Guided Wave (GW) Structural Health Monitoring (SHM) simulation models to generate a comprehensive database and train a machine learning-based inversion algorithm, aiming to improve the detection and characterization of defects. Feng et al [91] present an effective method for damage detection in hydro-junction infrastructure, utilizing a modified Inception-v3 deep convolutional neural network with transfer learning. The model trained on a set of high-definition images, outperforms an SVM model with a remarkable test accuracy of 96.8%. While the method proves to be efficient and accelerates training through GPU utilization, the lack of a unified image dataset for hydro-junction infrastructure presents challenges for comprehensive comparison and future improvements. Despite the proven effectiveness of ML in crack and damage detection, a key challenge lies in the requirement for significant human intervention in terms of feature selection, model selection, and hyperparameter tuning. The process is time-consuming and necessitates extensive expertise in ML, thereby limiting its wider adoption. In light of these limitations, the advent of Automated Machine Learning (AutoML) platforms, such as the H2O library, presents a promising avenue for expediting and enhancing the ML model development process. AutoML automates the end-to-end process of applying ML, including automatic data pre-processing, feature selection, model selection, and hyperparameter tuning, thus reducing the need for human intervention and expert knowledge. Although AutoML has been explored in various fields such as healthcare, finance, and retail [235]–[246], its application in structural health monitoring, particularly for crack and damage detection, remains under-explored. Moreover, the H2O library, despite its powerful capabilities, has not been thoroughly investigated for this specific use-case. Therefore, it is pertinent to conduct a comprehensive study on the

application of the H2O AutoML library for crack detection in engineering materials. This not only contributes to the growing body of literature on the integration of AutoML in structural health monitoring but also provides crucial insights into the practical implications of using the H2O library in this critical field of study. The promising potential of AutoML to revolutionize the process of crack and damage detection makes this an exigent area of research.

6.3 Methodology

6.3.1 Dataset Description and Preprocessing:

The dataset employed in this research comprises empirical observations of crack progression in ABS materials. The dataset is representative of typical crack propagation scenarios encountered in ABS materials as shown in Appendix F. In the process of conducting the experiments, the sample was secured onto the vibrating platform and subjected to varying degrees of heat. This platform was then tasked with exerting mechanical forces, while impact tests were initiated to identify the primary vibrational frequency of the sample. A laser vibrometer was employed to capture measurements. Subsequently, a vibration test was implemented at this identified frequency. Should there be any manifestation of crack development, the amplitude of the beam tip's displacement would decline, prompting a halt to the shaker in order to register the new frequency. Additional impact tests were carried out to ascertain this new frequency, which was subsequently programmed into the shaker. The cycle of these steps continued until the sample endured a catastrophic failure as a result of crack progression. The dataset includes information on various parameters such as temperature, natural frequency, amplitude, and crack depth. The crack propagation measurements were obtained from previous studies [194], [196], [198]. Prior to model development (as shown in Figure 6-1), The data is processed and refined to ensure it is optimized for training the automated machine learning model. Data preprocessing involves outliers, handling missing values, and data normalization. Outliers are identified and treated using robust statistical methods. Missing values are imputed using appropriate techniques such as mean imputation or

regression-based imputation. Data normalization is performed to scale the features to a consistent range, reducing the impact of varying magnitudes.

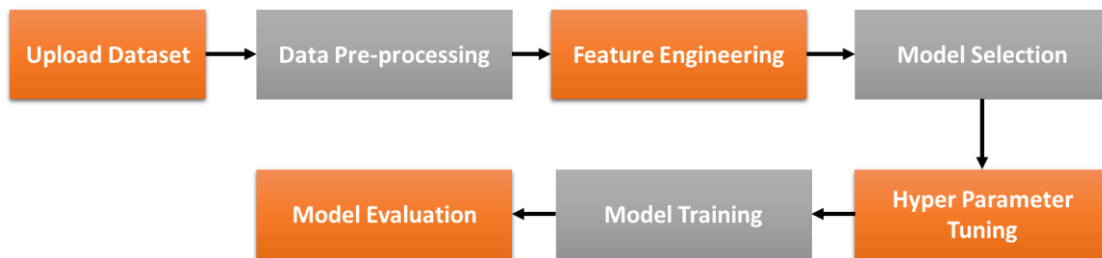


Figure 6-1 Flowchart of the process of AutoML H2O Application.

6.3.2 Feature Engineering and Model Selection and Configuration:

The importance of feature engineering in data interpretation cannot be understated. For the analysis of crack propagation in ABS materials, a variety of features are selected or engineered to accurately depict the material traits. Techniques such as correlation analysis and feature importance ranking are employed to discern the features that most significantly influence crack propagation. The H2O library offers an extensive suite of machine learning algorithms tailored for regression tasks. Given the specific nature of the crack propagation issue, suitable algorithms are chosen from the offerings of H2O. Several model parameters — including the learning rate, regularization strength, and the number of layers — are adjusted to maximize model efficiency while minimizing overfitting. Validation methods, such as k-fold cross-validation, are utilized to assess the models and identify the most efficient one.

The H2O AutoML framework encompasses a diverse range of machine learning models, including but not limited to, XGBoost Gradient Boosting Machines (GBM), H2O GBM, Random Forests (default and Extremely Randomized Trees versions), Deep Neural Networks, and Generalized Linear Models (GLM). Importantly, H2O is compatible with the widely-used XGBoost learning algorithm, enabling the integration of third-party algorithms and facilitating GPU-accelerated training. The current iteration of H2O AutoML is capable of training and, if required, cross-validating a series of models, inclusive of three pre-defined XGBoost GBM models, an H2O GLM grid, a default H2O Random Forest, five specific H2O GBMs, a near-default H2O Deep Neural Net, an H2O Extremely

Randomized Trees model, and random grids of XGBoost GBMs, H2O GBMs, and H2O Deep Neural Nets. These pre-defined models serve as efficient, reliable benchmarks for each algorithm. The sequence of algorithms can be user-customized, initiating with models that consistently perform at a high level (such as pre-set XGBoost models), followed by a tuned GLM for immediate benchmarking. Subsequently, the algorithm introduces model diversity by integrating a selection of Random Forests, GBM, and Deep Learning models. Upon the training and ranking of these pre-set models, a random search commences across these algorithms. The time allocation for each algorithm during the AutoML run is pre-determined, favoring some algorithms such as XGBoost GBM, H2O GBM over H2O Deep Learning, based on their perceived or calculated value. These techniques and methodologies are not limited to the current analysis but present significant potential for application in future research endeavors. They constitute a comprehensive toolkit for researchers, facilitating the tackling of diverse problem sets [247], [248].

6.3.3 Model Training and Evaluation:

The selected model is trained on the preprocessed dataset using H2O's training functions. The training process involves iteratively optimizing the model's parameters using the training data. Model performance is evaluated using appropriate evaluation metrics, such as mean squared error (MSE), root mean squared error (RMSE), or R-squared. The dataset is split into training and testing sets to assess the model's generalization ability and prevent overfitting. The model is iteratively refined by adjusting hyperparameters and retraining until satisfactory performance is achieved. To assess the accuracy and reliability of the automated model, several performance metrics are considered. In addition to MSE and RMSE, metrics such as mean absolute error (MAE), root mean squared logarithmic Error (RMSLE), and mean residual deviance are calculated. These metrics provide insights into the model's ability to capture the variance in crack propagation behavior and make accurate predictions. Additionally, visualization techniques, such as scatter plots or residual analysis, are employed to further assess the model's performance and identify potential areas for improvement. Through this methodology, we aim to develop an automated model using the H2O

library that accurately predicts crack propagation in ABS materials. The dataset preprocessing, feature engineering, and model training processes ensure the model's ability to capture the underlying patterns in crack growth behavior. The performance evaluation metrics provide a quantitative assessment of the model's predictive capabilities, contributing to enhanced understanding and analysis of crack propagation in ABS materials.

6.4 Results and Discussion

6.4.1 Insights into Crack Propagation Behavior in ABS Materials:

Analyzing the results from a comparative feature importance assessment involving multiple predictive models namely, Generalized Linear Model (GLM_1), Distributed Random Forest (DRF_1), and Gradient Boosting Machine (GBM_2), we uncover significant insights into the behavior of crack propagation in ABS materials. In this analysis as shown in table 6-1 , each row of the table represents an individual feature, while each column corresponds to a distinct predictive model.

Table 6-1 The importance of features for AutoML models.

<i>Features</i>	<i>GLM_1</i>	<i>DRF_1</i>	<i>GBM_2</i>
<i>Amplitude</i>	0.560706	0.439450	0.310581
<i>Natural frequency</i>	0.383545	0.534284	0.681861
<i>Temperature</i>	0.051501	0.012806	0.002864
<i>Crack location</i>	0.004248	0.013460	0.004693

The constituent values within the table denote feature importance scores, typically calculated based on the contribution of each feature towards the model's predictive output. Notably, a higher value suggests a more significant contribution of the corresponding feature to the prediction. Upon examining the comparative table 6-1:

- The `amplitude` feature demonstrates the highest importance for the GLM_1 and DRF_1 models, albeit less so for the GBM_2 model. This implies that amplitude variations significantly influence crack propagation in ABS materials according to the GLM and Random Forest models.

- The `natural frequency` feature, second in terms of importance for the GLM_1 model, emerges as the most crucial for the GBM_2 model. This feature also exhibits significant relevance for the DRF_1 model, suggesting that natural frequency plays a considerable role in understanding crack behavior across different models.
- The `temperature` feature showcases lesser significance across all three models when compared with the `amplitude` and `natural frequency` features, indicating a relatively lesser role of temperature variations in the prediction of crack propagation behavior.
- The `crack location` feature has been determined as the least important amongst all the features across all the models, suggesting a limited role of crack location in influencing the crack propagation behavior in ABS materials according to these models. A noteworthy consideration is that feature importance does not indicate the direction (positive or negative) of a feature's influence on the response, but merely denotes the magnitude of its influence. Moreover, the scales of feature importance between different models may not be directly comparable due to the distinct methodologies each model employs for calculating feature importance. Thus, while comparing feature importance across models, the focus should be on the ranking of features within each model, rather than on a direct comparison of the absolute values across models.

6.4.2 AutoML Models Selection

The practical aspects of utilizing H2O's AutoML in Python, and the significance of the corresponding code snippet as shown in Figure 6-2, cannot be understated. Essentially, this segment of code initiates the AutoML process, furnishing vital parameters that steer the operation. In essence, it signifies the point of entry into H2O's robust automated machine learning capabilities. Critical parameters guiding the operation of the AutoML process include:

```

1 # Perform AutoML
2 aml = H2OAutoML(max_models=3,
3                 seed=1,
4                 stopping_metric='AUTO',
5                 sort_metric='AUTO')
6 aml.train(y=target, training_frame=train)

```

Figure 6-2 The parameters of AutoML operation

- **max_models:** This parameter sets a limit on the number of individual models to train. By setting this to 3, we ensure that the AutoML process will stop after training three models. This setting helps to control the computational resources and time spent during the AutoML process, depending on the complexity and size of the dataset.
- **seed:** This parameter sets the seed for the pseudo-random number generator used by H2O's AutoML. It ensures that the randomness in the AutoML process, such as random hyperparameter selection, can be reproduced across multiple runs, improving the consistency and interpretability of our results.
- **max_runtime_secs:** This parameter sets a limit on the total time spent in the AutoML process, in seconds.
- **stopping_metric='AUTO' and sort_metric='AUTO':** These parameters control the metric used to compare and rank different models. By setting these to 'AUTO', we allow H2O's AutoML to automatically choose the most appropriate metric based on the task.

Following the initialization of the H2OAutoML object, the AutoML process proceeds with hyperparameter optimization.

6.4.3 Hyperparameter tuning and training of AutoML model

Hyperparameter tuning, an essential process in ML, involves determining the optimal configuration for model performance. While traditional manual tuning can be resource and time-intensive, recent advances in AutoML have significantly streamlined the process. This research applied H2O's AutoML on selected dataset, each with distinct features and complexities, to ensure the robustness of our results. A key aspect of our experimentation was the algorithm's capacity for automatic hyperparameter tuning, which optimizes a model's performance by

fine-tuning its configuration. The underlying principle of hyperparameter tuning in H2O's AutoML is a method known as grid search. The framework builds multiple models with varying hyperparameters, facilitating the evaluation of different combinations to identify the one that yields the best model performance.

Notably, the hyperparameter tuning process in H2O's AutoML is random, ensuring a diverse search space, and it automatically ranks models based on the selected evaluation metric. Following the hyperparameter optimization, H2O's AutoML proceeded with model training using the identified optimal hyperparameters as shown in Figure 6-3. This ensured that the resultant models were not only theoretically optimized based on hyperparameter configurations but were also empirically validated through rigorous training processes. H2O's AutoML successfully executed both hyperparameter tuning and model training, resulting in the selection of optimal model parameters that yielded superior results as shown in Table 6-2.

```
# Get the best model
best_model = h2o.get_model(best_metric)

# Print the hyperparameters for the best model
print("Best Model Hyperparameters:")
print(best_model.params)

# Get hyperparameters for all models in the AutoML Leaderboard
hyperparameters = {}
for model_id in model_ids:
    model = h2o.get_model(model_id)
    hyperparameters[model_id] = model.params

# Convert hyperparameters to a Pandas DataFrame for easier tabular represent
import pandas as pd
hyperparameters_df = pd.DataFrame.from_dict(hyperparameters, orient='index')

# Print the hyperparameters table
print("Hyperparameters for All Models:")
print(hyperparameters_df)

# Deploy model for prediction
model_path = h2o.save_model(model=best_model, path="./model", force=True)
loaded_model = h2o.load_model(model_path)
prediction = loaded_model.predict(data)
```

Figure 6-3 Code to identify optimal hyperparameters for all selected models.

This automated process significantly reduced manual intervention, minimizing bias and human error, and resulted in a consistent improvement in predictive

performance across various tasks. Moreover, the entire process, from data preprocessing to model training, was conducted within a reasonable timeframe, emphasizing the efficiency of the framework. H2O's AutoML managed to simplify the process, making it accessible to both novice and experienced data scientists.

This result aligns with the premise that the democratization of ML processes is a practical and necessary evolution, especially as the world becomes increasingly data-driven. The results demonstrate that the H2O's AutoML process, with its integrated hyperparameter tuning and robust training mechanism, serves as a powerful tool in the ML toolkit, yielding high-performing models and simplifying the ML deployment process. Future work should explore the scalability of this approach in larger, more complex datasets and real-world applications.

6.4.4 Performance Evaluation of the Selected Automated Model:

The developed automated model using the H2O library demonstrates promising performance in predicting crack propagation in ABS materials. Evaluations of the model's accuracy were carried out using multiple performance metrics as outlined in Table 6-2. The GBM model achieves an RMSE of 0.24784, MSE of 0.0614249, MAE of 0.209033, RMSLE of 0.117522, and a mean residual deviance of 0.0614249. These values suggest that the GBM model offers the best performance among the models listed in terms of accuracy and fit as shown in Figure 6-4.

Conversely, the Stacked Ensemble model records an RMSE of 0.259332, MSE of 0.0672531, and an MAE of 0.217328. The RMSLE value of 0.129487 and mean residual deviance of 0.0672531 further support its commendable performance, although it slightly lags behind the GBM model.

Table 6-2 The performance of H2O models based on different regression metrics.

Model	RMSE "mm"	MSE "mm"	MAE "mm"	RMSLE "mm"	mean residual- deviance "mm"
GBM_2_AutoML	0.24784	0.0614249	0.209033	0.117522	0.0614249
Stacked Ensemble Best Off family	0.259332	0.0672531	0.217328	0.129487	0.0672531
GLM_1_AutoML	0.315497	0.0995384	0.263245	0.151175	0.0995384
DRF	0.349196	0.121938	0.297976	0.162201	0.121938

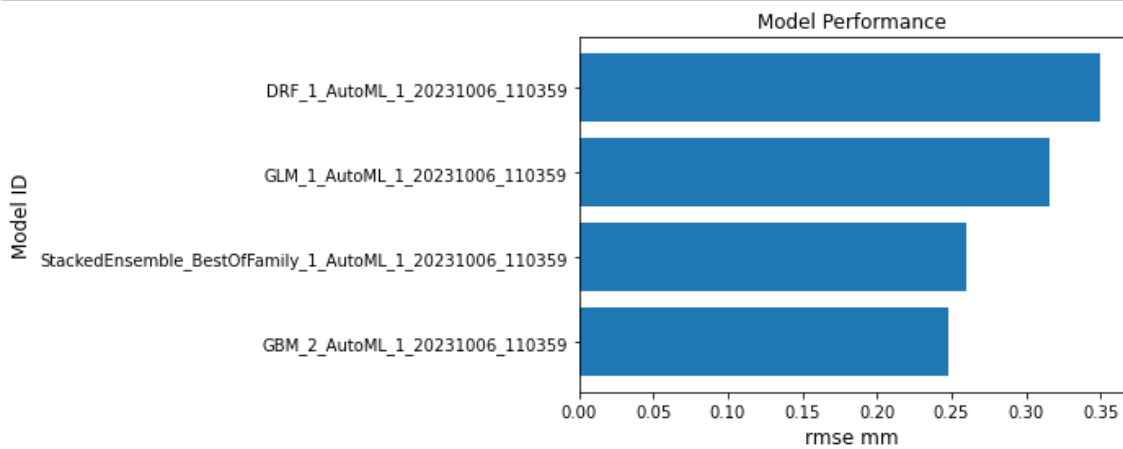
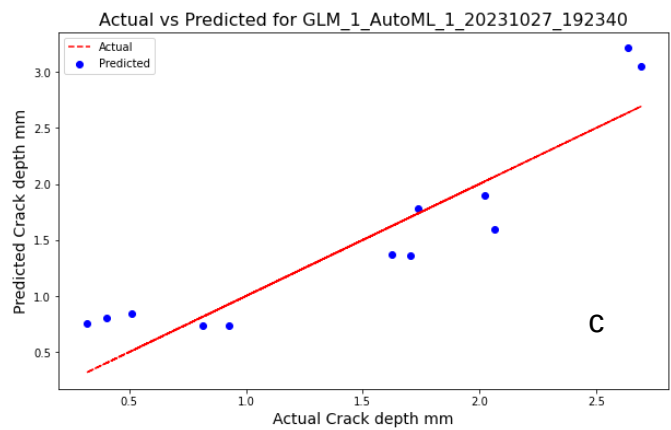
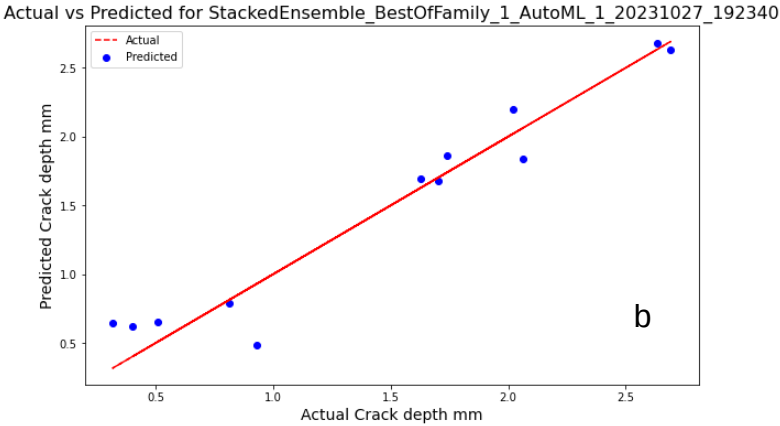
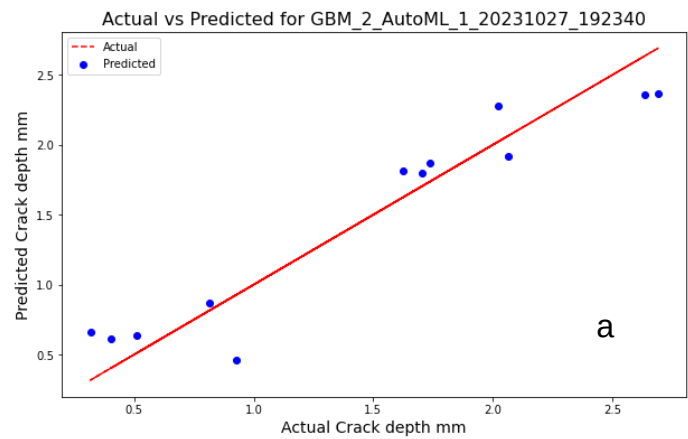


Figure 6-4 The preformance of H2O selected models based on RMSE metric

The DRF model exhibits higher RMSE, MSE, MAE, RMSLE, and mean residual deviance values of 0.349196, 0.121938, 0.297976, 0.162201, and 0.121938, respectively. These metrics suggest that while the DRF model is robust, it may not be as precise as the GBM or Stacked Ensemble models. The GLM model showcases RMSE, MSE, MAE, RMSLE, and mean residual deviance values of 0.315497, 0.0995384, 0.263245, 0.151175, and 0.0995384, respectively. Figure 6-5 illustrates the performance of the GBM, Stacked Ensemble, DRF, and GLM models over a series of iterations. It can be observed that the GLM model, although valuable, might have higher prediction errors compared to the other models, making it potentially less suitable for this dataset. The GBM model stands out as the most accurate based on the provided metrics, with all models offering valuable insights into crack propagation in ABS materials. These findings

underscore the potential of machine learning approaches in structural health monitoring, setting a promising direction for future research.



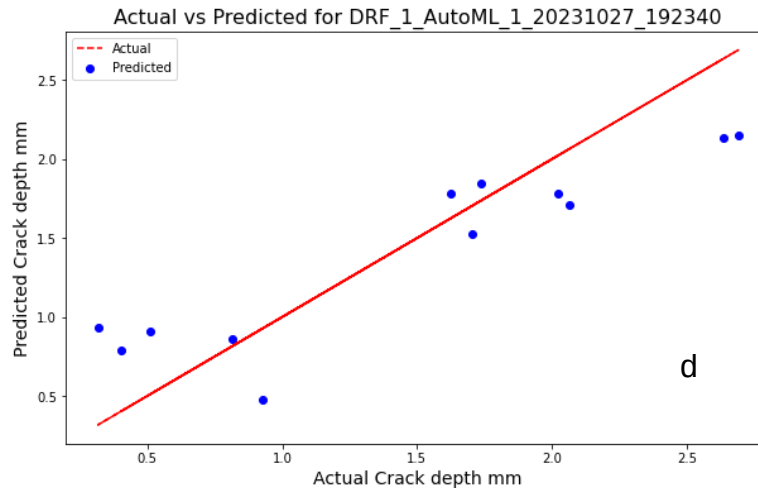


Figure 6-5 The actual against predicted of ABS crack depth a. GBM AutoML b. Stacked Ensemble Best of Family AutoML c. CLM AutoML d. DRF AutoML.

6.5 Comparison with Machine learning Approaches:

H2O's AutoML is renowned for its advanced capabilities in tackling intricate predictive challenges. This tool harnesses a diverse range of machine learning algorithms, such as Deep Neural Networks, H2O GBM, GLM, and Random Forests. Additionally, it seamlessly integrates with the popular XGBoost model, facilitating the incorporation of external algorithms and GPU-powered training. The latest version of H2O AutoML is designed to train and, where required, cross-validate models. This includes a set of XGBoost GBM models, an H2O GLM grid, the standard H2O Random Forest, multiple specific H2O GBMs, a H2O Deep Neural Net that closely follows default settings, an H2O Extremely Randomized Trees model, and random configurations of XGBoost GBMs, H2O GBMs, and H2O Deep Neural Nets [249]. In contrast, conventional regression algorithms such as Linear Regression (LR), Back-Propagation Neural Network (BPNN), Classification and Regression Tree (CART), and Support Vector Regression (SVR) have been employed to forecast a range of material properties. This includes predicting the self-repairing ability of Engineered Cementitious Composite (ECC) [250]. These algorithms have demonstrated potential, with ensemble techniques such as bagging, AdaBoost, and stacking further enhancing prediction precision. Yet, they demand manual hyperparameter adjustments and might not be as adept as AutoML solutions in processing

extensive datasets or intricate data interrelations. In research focused on forecasting crack growth in aviation aluminum alloy, the SVR algorithm was utilized. This model, trained using data from fatigue tests on crack length expansion, proved adept at accurately forecasting crack expansion between three holes [210]. In a separate investigation, reinforcement learning was deployed to determine the optimal times for structure inspections and decommissions to prevent failures. This research compared two distinct regression algorithms: neural networks (NN) and k-nearest neighbors (KNN) [251]. The results favored the KNN algorithm in terms of performance. While traditional regression techniques have yielded encouraging outcomes in predicting crack growth in diverse materials, AutoML solutions, such as H2O, present numerous benefits. These encompass efficient processing of vast datasets, automated hyperparameter adjustments, and the capability to discern intricate data patterns. Nonetheless, the selection of an algorithm should align with the task's specific demands, such as dataset size and intricacy, interpretability necessities, and available computational resources.

6.6 Advantages of the Automated Model:

The automated model developed using the H2O library for crack propagation prediction in ABS materials offers several advantages over traditional approaches. Firstly, the model demonstrates superior accuracy in predicting crack lengths, as indicated by the low RMSE and MAE values. This accuracy can significantly contribute to improving structural integrity assessment and maintenance strategies by enabling proactive measures to prevent catastrophic failures [7]. Secondly, the automated model reduces the dependency on manual intervention and subjective decision-making, as commonly encountered in analytical and numerical methods. By leveraging machine learning algorithms, the model can autonomously learn from the data and capture complex relationships, leading to more reliable predictions without the need for simplifying assumptions [249].

Additionally, the model's efficiency and scalability are noteworthy. The H2O library provides efficient handling of large datasets and offers automated

hyperparameter tuning, accelerating the model development process. The ability to process extensive datasets efficiently enhances the model's robustness and widens its applicability to real-world scenarios [7], [249].

6.7 Practical Implications:

The application of the automated model in predicting crack propagation in ABS materials has significant practical implications for industries such as automotive, aerospace, and consumer goods manufacturing. Enhanced crack propagation analysis can lead to improved product design, material selection, and manufacturing processes, ultimately ensuring the safety and reliability of engineering components. The model's predictive capabilities enable the identification of critical scenarios and the estimation of remaining useful life, facilitating more informed decision-making regarding maintenance and repair strategies. By accurately predicting crack growth behavior, proactive measures can be taken to mitigate potential risks, reduce downtime, and optimize maintenance costs. Furthermore, the insights gained from the automated model offer valuable guidance for materials scientists and engineers in the development of new materials and the optimization of existing ones. Understanding the influential factors on crack propagation in ABS materials can aid in the design of materials with enhanced fracture toughness, durability, and resistance to crack propagation, leading to improved product performance and longevity.

6.8 Limitations and Future Directions:

While the automated model presents significant advancements in crack propagation prediction, it is important to acknowledge its limitations. The model's performance heavily relies on the quality and representativeness of the training dataset. Inadequate or biased data may result in suboptimal predictions. Therefore, the availability of comprehensive and diverse datasets for model training and validation is crucial. Moreover, the model's generalizability to different ABS materials and environmental conditions should be further investigated. The variability in material composition, manufacturing processes, and loading conditions may influence crack propagation behavior. Future research should focus on expanding the dataset to incorporate a broader range

of ABS materials and considering additional factors such as temperature, humidity, and chemical exposure. Furthermore, the interpretability of the automated model warrants attention. Although machine learning models excel in predictive accuracy, understanding the underlying mechanisms and factors contributing to crack propagation can be challenging. Efforts should be made to develop explainable AI techniques that provide insights into the model's decision-making process, allowing engineers and scientists to gain a deeper understanding of the crack propagation phenomenon. The automation of crack propagation prediction can be extended to other materials and structural components beyond ABS. The methodology presented in this study can serve as a foundation for developing automated models for different materials, allowing for broader applications in diverse engineering domains.

6.9 Summary

The automated model utilizing the H2O library presents a significant advancement in crack propagation prediction for ABS materials. The model's accuracy, efficiency, and scalability offer substantial benefits for structural integrity assessment, maintenance strategies, and materials design in various industries. The accurate predictions provided by the automated model enable engineers to make proactive decisions regarding maintenance and repair strategies, ultimately enhancing the safety and reliability of engineering components. By accurately forecasting crack propagation behavior, potential risks can be mitigated, downtime can be minimized, and maintenance costs can be optimized. The efficiency and scalability of the automated model streamline the crack propagation analysis process. With the H2O library's powerful capabilities, large datasets can be processed efficiently, enabling robust and reliable predictions. This efficiency allows for the analysis of extensive data sets, facilitating a comprehensive understanding of crack propagation behavior in ABS materials.

Furthermore, the model's ability to capture complex relationships between crack behavior and various influencing factors provides valuable insights into the underlying mechanisms of crack propagation. This deeper understanding

enables engineers and scientists to make informed decisions regarding materials design, manufacturing processes, and structural improvements, leading to enhanced product performance and longevity. While there are limitations and areas for future research, such as the need for diverse and representative datasets and the interpretability of the model, the automated model demonstrates the potential to revolutionize crack propagation analysis in ABS materials and beyond. Continued research and development in this field will further advance the automation of crack propagation prediction, enabling its application in a broader range of materials and contributing to enhanced safety, reliability, and performance of engineering components across industries. The automated model utilizing the H2O library offers a powerful tool for accurate crack propagation prediction in ABS materials. Its impact extends to structural integrity assessment, maintenance strategies, and materials design, facilitating informed decision-making and advancing materials science and engineering practices. With ongoing research and development, the automation of crack propagation prediction will continue to drive improvements in safety, reliability, and performance in various industrial applications. The subsequent chapter concludes the main findings and contributions of the research. It Proposes future directions and potential areas for further research.

7 Conclusions

This research aimed to explore the application of ML in structural health monitoring and assessment and material science, focusing on different materials such as composite, metal, and polymer. The objectives set at the beginning of the study have been successfully achieved, and the novel contributions of the research have been highlighted. This chapter presents the conclusions drawn from the study, summarizing the findings and discussing the implications of the research.

7.1 Achievement of Objectives

Objective 1: Identify the current challenges in machine learning approaches for structural health monitoring and assessment: The study identified several key challenges in the application of ML in SHMA. These include the limited availability of high-quality data, the difficulty in interpreting ML model results, the issue of overfitting, and the lack of general guidelines for the selection of ML models. The study proposed potential solutions to these challenges, such as the creation of open-source datasets, the use of importance testing and physics-based loss functions, and the development of knowledge-informed selection strategies.

Objective 2: Analyse the compatibility of dynamic response data of different material structures with the prominent ML approaches: The research analyzed the suitability of various ML algorithms for different materials and data types. It was found that the choice of algorithm should consider the type of data available, the material being studied, and the desired knowledge outcomes. The study also highlighted the importance of interpretability in addition to predictive accuracy when selecting an ML algorithm.

Objective 3: Develop an ML model based on the suitable approach/es to relate the interdependencies between the dynamic response and crack propagation in structures: The study successfully developed ML models using various regression techniques to predict crack depth in different

materials. The models were able to capture meaningful patterns within the data and make reasonably accurate predictions.

Objective 4: Evaluate the efficiency and accuracy of the developed ML model: The performance of the developed models was evaluated using MSE on both training and test sets. All models surpassed the baseline model, indicating their effectiveness in predicting crack depth.

Objective 5: Validate the proposed model with experiments on arbitrary values (composite, metal, and polymer): The models were validated using experimental data from previous research. The results confirmed the models' ability to accurately predict crack depth in different materials.

7.2 Novel Contributions

The study has made several noteworthy contributions to the fields of structural health monitoring, assessment, and material science:

7.2.1 Enhancing Structural Health Monitoring: Comprehensive Analysis of Diverse Materials and Machine Learning Models for Accurate Crack Propagation Prediction

The research significantly advances the field by conducting a comprehensive analysis involving multiple materials, diverse machine learning models, and accounting for dynamic load and temperature impacts. Through validation using experimental data, the study provides valuable insights into crack propagation prediction. The findings hold potential for optimizing resource allocation and cost-effective maintenance through precise identification of critical areas and accurate prediction of crack propagation rates. This targeted strategy minimizes unnecessary maintenance, optimizes resource utilization, and lowers costs, thereby ensuring the enduring reliability and durability of structures across various materials.

7.2.2 Suitability Analysis of Machine Learning Algorithms for Crack Growth Prediction Based on Dynamic Response Data

This research analyses the requirements and characteristics of the most commonly used machine learning algorithms in SHM as in Chapter 3. The aim is

to assist researchers in selecting models that are both effective and comprehensible. This analysis can aid researchers in improving their comprehension of the model's predictions. The contribution of this research to the field of SHM lies in the examination of the compatibility between dynamic response data from diverse material structures and prominent machine learning approaches. The study involves an exploration of the data's characteristics alongside the utilisation of the most prevalent machine learning algorithms within the field.

7.2.3 Automated Prediction of Crack Propagation Using H2O AutoML: Towards Enhanced Structural Health Monitoring

The research introduces an innovative automated model that leverages the H2O library as in Chapter 6. This model empowers researchers to swiftly construct accurate crack propagation prediction models without necessitating an extensive background in machine learning. The automation streamlines the modeling process and fosters the broader adoption of machine learning techniques.

However, it remains paramount for researchers to thoroughly assess the compatibility of diverse machine learning algorithms, considering factors such as available data types, specific materials under examination, and desired learning outcomes. The study's findings bear practical implications for specialists dedicated to structural health monitoring and maintenance. They hold the potential to drive remarkable advancements within the field of structural engineering and maintenance practices. By bridging the gap between theoretical research and real-world applications, this study stands as a beacon of progress within the broader context of material science and structural integrity. The subsequent chapter delves into the future research directions and avenues.

7.3 Future Research Directions

Although this study has made significant strides in understanding and applying machine learning in structural health monitoring and assessment, there are several avenues for future research that could further enhance the potential of ML in this field and address the identified challenges.

1. Expanding the Dataset: One of the primary challenges identified in this study was the limited availability of high-quality data. Future research could focus on collecting, curating, and generating diverse datasets for different materials and conditions. This would not only enhance the training of ML models but also allow for more robust and generalizable models.
2. Investigating Additional Features: The study identified several influential predictive features for each material. Future research could investigate the influence of additional features, such as material composition or environmental factors, on crack formation and propagation.
3. Deep Transfer Learning: Researchers can explore the use of transfer learning. In order to improve the performance and interpretability of CNNs for crack detection. This can be achieved by fine-tuning pre-trained CNN models on concrete, polymer, or metal images, allowing the model to quickly learn relevant features for crack detection. By addressing these areas, future research could significantly contribute to the advancement of ML applications in structural health monitoring and assessment, leading to more accurate and reliable predictions, and ultimately, safer, and more efficient structures.

REFERENCES

- [1] Y.-A. Hsieh and Y. J. Tsai, "Machine Learning for Crack Detection: Review and Model Performance Comparison," *Journal of Computing in Civil Engineering*, vol. 34, no. 5, p. 04020038, 2020, doi: 10.1061/(ASCE)CP.1943-5487.0000918.
- [2] A. Khan, N. Kim, J. K. Shin, H. S. Kim, and B. D. Youn, "Damage assessment of smart composite structures via machine learning: a review," *JMST Advances*, vol. 1, no. 1–2, pp. 107–124, Jun. 2019, doi: 10.1007/s42791-019-0012-2.
- [3] M. Flah, I. Nunez, W. Ben Chaabene, and M. L. Nehdi, "Machine Learning Algorithms in Civil Structural Health Monitoring: A Systematic Review," *Archives of Computational Methods in Engineering*, 2020, doi: 10.1007/s11831-020-09471-9.
- [4] G. Toh and J. Park, "Review of vibration-based structural health monitoring using deep learning," *Applied Sciences (Switzerland)*, vol. 10, no. 5, 2020, doi: 10.3390/app10051680.
- [5] J. Sarah, F. Hejazi, R. S. M. Rashid, and N. Ostovar, "A Review of Dynamic Analysis in Frequency Domain for Structural Health Monitoring," *IOP Conf Ser Earth Environ Sci*, vol. 357, no. 1, 2019, doi: 10.1088/1755-1315/357/1/012007.
- [6] S. Nasiri, M. R. Khosravani, and K. Weinberg, "Fracture mechanics and mechanical fault detection by artificial intelligence methods: A review," *Eng Fail Anal*, vol. 81, no. November, pp. 270–293, 2017, doi: 10.1016/j.engfailanal.2017.07.011.
- [7] I. Omar, M. Khan, and A. Starr, "Suitability Analysis of Machine Learning Algorithms for Crack Growth Prediction Based on Dynamic Response Data," *Sensors 2023, Vol. 23, Page 1074*, vol. 23, no. 3, p. 1074, Jan. 2023, doi: 10.3390/S23031074.

- [8] I. Omar, M. Khan, and A. Starr, "Compatibility and challenges in machine learning approach for structural crack assessment," <https://doi.org/10.1177/14759217211061399>, vol. 0, no. 0, pp. 1–22, Mar. 2022, doi: 10.1177/14759217211061399.
- [9] A. Rytter, "Vibrational Based Inspection of Civil Engineering Structures — Aalborg University's Research Portal," Dept. of Building Technology and Structural Engineering, Aalborg University, 1993. Accessed: Feb. 22, 2022. [Online]. Available: <https://vbn.aau.dk/en/publications/vibrational-based-inspection-of-civil-engineering-structures>
- [10] C. R. Farrar and N. A. J. Lieven, "Damage prognosis: The future of structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 623–632, Feb. 2007, doi: 10.1098/rsta.2006.1927.
- [11] Friswell and Mottershead, "Inverse methods in structural health monitoring," *Key Eng Mater*, vol. 204, no. 205, pp. 201–210, 2001.
- [12] M. I. Friswell, "Damage identification using inverse methods," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851. Royal Society, pp. 393–410, Feb. 15, 2007. doi: 10.1098/rsta.2006.1930.
- [13] S. Bureerat and N. Pholdee, "Inverse problem based differential evolution for efficient structural health monitoring of trusses," *Appl Soft Comput*, vol. 66, pp. 462–472, May 2018, doi: 10.1016/J.ASOC.2018.02.046.
- [14] D. A. Tibaduiza Burgos, R. C. Gomez Vargas, C. Pedraza, D. Agis, and F. Pozo, "Damage Identification in Structural Health Monitoring: A Brief Review from its Implementation to the Use of Data-Driven Applications," *Sensors 2020, Vol. 20, Page 733*, vol. 20, no. 3, p. 733, Jan. 2020, doi: 10.3390/S20030733.

- [15] P. A. Gardner, "On Novel Approaches to Model-Based Structural Health Monitoring," Sheffield University, 2018.
- [16] R. Kromanis and P. Kripakaran, "Data-driven approaches for measurement interpretation: analysing integrated thermal and vehicular response in bridge structural health monitoring," *Advanced Engineering Informatics*, vol. 34, pp. 46–59, Oct. 2017, doi: 10.1016/J.AEI.2017.09.002.
- [17] M. I. Friswell, J. E. T. Penny, and S. D. Garvey, "Parameter subset selection in damage location," *Inverse Problems in Engineering*, vol. 5, no. 3, pp. 189–215, Oct. 1997, doi: 10.1080/174159797088027660.
- [18] M. I. Friswell and J. E. Mottershead, *Finite Element Model Updating in Structural Dynamics*, vol. 38. in Solid Mechanics and its Applications, vol. 38. Dordrecht: Springer Netherlands, 1995. doi: 10.1007/978-94-015-8508-8.
- [19] H. V. Dang, H. Tran-Ngoc, T. V. Nguyen, T. Bui-Tien, G. De Roeck, and H. X. Nguyen, "Data-Driven Structural Health Monitoring Using Feature Fusion and Hybrid Deep Learning," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 2087–2103, Oct. 2021, doi: 10.1109/TASE.2020.3034401.
- [20] A. Sivasuriyan *et al.*, "Development of Smart Sensing Technology Approaches in Structural Health Monitoring of Bridge Structures," *Advances in Materials Science and Engineering*, vol. 2021, 2021, doi: 10.1155/2021/2615029.
- [21] F.-G. Yuan, S. A. Zargar, Q. Chen, and S. Wang, "Machine learning for structural health monitoring: challenges and opportunities," p. 2, Apr. 2020, doi: 10.1117/12.2561610.
- [22] G. F. Gomes, Y. A. D. Mendez, P. da Silva Lopes Alexandrino, S. S. da Cunha, and A. C. Ancelotti, "A Review of Vibration Based Inverse

Methods for Damage Detection and Identification in Mechanical Structures Using Optimization Algorithms and ANN,” *Archives of Computational Methods in Engineering*, vol. 26, no. 4, pp. 883–897, Sep. 2019, doi: 10.1007/S11831-018-9273-4.

- [23] M. A. Hariri-Ardebili and F. Pourkamali-Anaraki, “Support vector machine based reliability analysis of concrete dams,” *Soil Dynamics and Earthquake Engineering*, vol. 104, pp. 276–295, Jan. 2018, doi: 10.1016/j.soildyn.2017.09.016.
- [24] J. J. Lee, J. W. Lee, J. H. Yi, C. B. Yun, and H. Y. Jung, “Neural networks-based damage detection for bridges considering errors in baseline finite element models,” *J Sound Vib*, vol. 280, no. 3–5, pp. 555–578, Feb. 2005, doi: 10.1016/j.jsv.2004.01.003.
- [25] S. B. Satpal, A. Guha, and S. Banerjee, “Damage identification in aluminum beams using support vector machine: Numerical and experimental studies,” *Struct Control Health Monit*, vol. 23, no. 3, pp. 446–457, Mar. 2016, doi: 10.1002/stc.1773.
- [26] C. Q. Xiong, Z. W. Zhu, and J. Jiang, “Response Surface-Based Finite Element Model Updating of Steel Box-Girder Bridges with Concrete Composite Decks,” *Advances in Civil Engineering*, vol. 2022, 2022, doi: 10.1155/2022/4298933.
- [27] V. Shimpi, M. V. R. Sivasubramanian, S. B. Singh, and D. K. Periyasamy, “Seismic vulnerability assessment and fragility curves for a multistorey gallery arch bridge,” *SN Appl Sci*, vol. 3, no. 6, pp. 1–19, Jun. 2021, doi: 10.1007/S42452-021-04652-Y/FIGURES/16.
- [28] M. Tatarciuc, G. A. Maftai, A. Vitalariu, I. Luchian, I. Martu, and D. Diaconu-Popa, “Inlay-Retained Dental Bridges—A Finite Element Analysis,” *Applied Sciences 2021, Vol. 11, Page 3770*, vol. 11, no. 9, p. 3770, Apr. 2021, doi: 10.3390/APP11093770.

- [29] M.-Y. Liu, P.-H. Wang, M.-Y. Liu, and P.-H. Wang, "Finite Element Analysis of Cable-Stayed Bridges with Appropriate Initial Shapes Under Seismic Excitations Focusing on Deck-Stay Interaction," *Earthquake Engineering*, Aug. 2012, doi: 10.5772/48440.
- [30] M. Azimi, A. D. Eslamlou, and G. Pekcan, *Data-driven structural health monitoring and damage detection through deep learning: State-of-the-art review*, vol. 20, no. 10. 2020. doi: 10.3390/s20102778.
- [31] R. Ghiasi, H. Fathnejat, and P. Torkzadeh, "A three-stage damage detection method for large-scale space structures using forward substructuring approach and enhanced bat optimization algorithm," *Eng Comput*, vol. 35, no. 3, pp. 857–874, 2019, doi: 10.1007/s00366-018-0636-0.
- [32] R. P. Finotti, A. A. Cury, and F. de S. Barbosa, "An SHM approach using machine learning and statistical indicators extracted from raw dynamic measurements," *Latin American Journal of Solids and Structures*, vol. 16, no. 2, pp. 1–17, 2019, doi: 10.1590/1679-78254942.
- [33] C. R. Farrar and K. Worden, *Structural Health Monitoring: A Machine Learning Perspective*. John Wiley and Sons, 2012. doi: 10.1002/9781118443118.
- [34] K. Worden and G. Manson, "The application of machine learning to structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 515–537, Feb. 2007, doi: 10.1098/rsta.2006.1938.
- [35] C. R. Farrar and K. Worden, "An Introduction to Structural Health Monitoring," *CISM International Centre for Mechanical Sciences, Courses and Lectures*, vol. 520, pp. 1–17, 2010, doi: 10.1007/978-3-7091-0399-9_1.

- [36] C. R. Farrar and K. Worden, "An introduction to structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 303–315, Feb. 2007, doi: 10.1098/rsta.2006.1928.
- [37] Y. V. Klunnikova, M. V. Anikeev, A. V. Filimonov, and R. Kumar, "Machine learning application for prediction of sapphire crystals defects," *Journal of Electronic Science and Technology*, vol. 18, no. 1, p. 100029, 2020, doi: 10.1016/j.jnlest.2020.100029.
- [38] H. Salehi and R. Burgueño, "Emerging artificial intelligence methods in structural engineering," *Engineering Structures*, vol. 171. Elsevier Ltd, pp. 170–189, Sep. 15, 2018. doi: 10.1016/j.engstruct.2018.05.084.
- [39] F. M. Bono, L. Radicioni, S. Cinquemani, and G. Bombaci, "A Comparison of Deep Learning Algorithms for Anomaly Detection in Discrete Mechanical Systems," *Applied Sciences 2023, Vol. 13, Page 5683*, vol. 13, no. 9, p. 5683, May 2023, doi: 10.3390/APP13095683.
- [40] J. Sierra, J. C. Perafan, C. Herrera, and C. Nieto, "Data-driven models for strain-based damage identification in composite wind turbine blades," *Materials Research Proceedings*, vol. 27, pp. 17–24, 2023, doi: 10.21741/9781644902455-3.
- [41] Y. Reuland, P. Martakis, and E. Chatzi, "A Comparative Study of Damage-Sensitive Features for Rapid Data-Driven Seismic Structural Health Monitoring," *Applied Sciences 2023, Vol. 13, Page 2708*, vol. 13, no. 4, p. 2708, Feb. 2023, doi: 10.3390/APP13042708.
- [42] D.-Z. ; Dang *et al.*, "Image Classification-Based Defect Detection of Railway Tracks Using Fiber Bragg Grating Ultrasonic Sensors," *Applied Sciences 2023, Vol. 13, Page 384*, vol. 13, no. 1, p. 384, Dec. 2022, doi: 10.3390/APP13010384.

- [43] A. Pollastro, G. Testa, A. Bilotta, and R. Prevete, "Semi-supervised detection of structural damage using Variational Autoencoder and a One-Class Support Vector Machine," Oct. 2022, doi: 10.1109/ACCESS.2023.3291674.
- [44] Abeer Ahmad *et al.*, "Dispersed passive RF-sensing for 3D structural health monitoring," *ITU Journal on Future and Evolving Technologies*, vol. 3, no. 2, pp. 535–544, Sep. 2022, doi: 10.52953/ZQYZ8264.
- [45] R. J. E. Fuentes, "On Bayesian Networks for Structural Health and Condition Monitoring," 2017.
- [46] H. E. Balcioğlu and A. Ç. Seçkin, "Comparison of machine learning methods and finite element analysis on the fracture behavior of polymer composites," *Archive of Applied Mechanics*, vol. 91, no. 1, pp. 223–239, 2021, doi: 10.1007/s00419-020-01765-5.
- [47] F. Liu, X. Jiang, X. Wang, and L. Wang, "Machine learning-based design and optimization of curved beams for multistable structures and metamaterials," *Extreme Mech Lett*, vol. 41, p. 101002, 2020, doi: 10.1016/j.eml.2020.101002.
- [48] X. Y. Zhao, C. Y. Dong, P. Zhou, M. J. Zhu, J. W. Ren, and X. Y. Chen, "Detecting surface defects of wind turbine blades using an Alexnet deep learning algorithm," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102A, no. 12, pp. 1817–1824, 2019, doi: 10.1587/transfun.E102.A.1817.
- [49] M. Zhang *et al.*, "High cycle fatigue life prediction of laser additive manufactured stainless steel: A machine learning approach," *Int J Fatigue*, vol. 128, no. July, p. 105194, 2019, doi: 10.1016/j.ijfatigue.2019.105194.

- [50] Y. Kang, C. C. Wang, and Y. P. Chang, "Gear Fault Diagnosis in Time Domains by Using Bayesian Networks," *Advances in Soft Computing*, vol. 42, pp. 741–751, 2007, doi: 10.1007/978-3-540-72434-6_75.
- [51] A. Rovinelli, M. D. Sangid, H. Proudhon, and W. Ludwig, "Using machine learning and a data-driven approach to identify the small fatigue crack driving force in polycrystalline materials," *NPJ Comput Mater*, vol. 4, no. 1, pp. 1–10, 2018, doi: 10.1038/s41524-018-0094-7.
- [52] M. Li and Z. Wang, "Deep learning for high-dimensional reliability analysis," *Mech Syst Signal Process*, vol. 139, p. 106399, 2020, doi: 10.1016/j.ymssp.2019.106399.
- [53] S. Sun, L. Liang, M. Li, and X. Li, "Vibration-based Damage Detection in Bridges via Machine Learning," *KSCE Journal of Civil Engineering*, vol. 22, no. 12, pp. 5123–5132, 2018, doi: 10.1007/s12205-018-0318-x.
- [54] L. Xu, S. F. Yuan, J. Chen, and Q. Bao, "Deep learning based fatigue crack diagnosis of aircraft structures," *Proceedings of the 7th Asia-Pacific Workshop on Structural Health Monitoring, APWSHM 2018*, pp. 410–418, 2018.
- [55] H. Sohn, "Effects of environmental and operational variability on structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 539–560, Dec. 2006, doi: 10.1098/RSTA.2006.1935.
- [56] M. El Mountassir, S. Yaacoubi, J. Ragot, G. Mourot, and D. Maquin, "Feature selection techniques for identifying the most relevant damage indices in SHM using Guided Waves," in *8th European Workshop On Structural Health Monitoring (EWSHM 2016)*, 2016.

Accessed: Feb. 22, 2022. [Online]. Available: <http://www.ndt.net/?id=20151>

- [57] P. Rizzo and F. Lanza di Scalea, "Discrete wavelet transform to improve guided-wave-based health monitoring of tendons and cables," *Smart Structures and Materials 2004: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, vol. 5391, p. 523, Jul. 2004, doi: 10.1117/12.539847.
- [58] Huang T. M., Kecman V., and Kopriva I., *Kernel Based Algorithms for Mining Huge Data Sets*. Springer-Verlag, 2006. doi: 10.1007/3-540-31689-2.
- [59] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," *Data Classification: Algorithms and Applications*, pp. 37–64, Jan. 2014, doi: 10.1201/B17320.
- [60] M. Verleysen and D. François, "The Curse of Dimensionality in Data Mining and Time Series Prediction," *Lecture Notes in Computer Science*, vol. 3512, pp. 758–770, 2005, doi: 10.1007/11494669_93.
- [61] G. C. Cawley and N. L. C. Talbot, "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation," *Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.
- [62] A. Bagheri, E. Pistone, and P. Rizzo, "Outlier Analysis and Artificial Neural Network for the Noncontact Nondestructive Evaluation of Immersed Plates," <http://dx.doi.org/10.1080/09349847.2015.1022677>, vol. 26, no. 3, pp. 154–173, Jul. 2015, doi: 10.1080/09349847.2015.1022677.
- [63] X. Zhang, G. Sun, H. Liu, and Q. Wang, "Flaw classification in ultrasonic guided waves signal using Wavelet Transform and PNN classifier," *2011 International Conference on Wireless Communications and Signal Processing, WCSP 2011*, 2011, doi: 10.1109/WCSP.2011.6096901.

- [64] Z. Jiang, Z. Zhang, and A. Maxwell, "Extraction of structural modal information using acoustic sensor measurements and machine learning," *J Sound Vib*, vol. 450, pp. 156–174, 2019, doi: 10.1016/j.jsv.2019.03.009.
- [65] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," 2004.
- [66] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," 2005. [Online]. Available: <http://lear.inrialpes.fr>
- [67] Lienhart Rainer and Maydt Jochen, "An Extended Set of Haar-like Features for Rapid Object Detection," 2002.
- [68] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," 2001.
- [69] S.-F. Jiang, C.-M. Zhang, and C. G. Koh, "Structural Damage Detection by Integrating Data Fusion and Probabilistic Neural Network," *Advances in Structural Engineering*, vol. 9, 2006.
- [70] H. Sun, H. V. Burton, and H. Huang, "Machine learning applications for building structural design and performance assessment: State-of-the-art review," *Journal of Building Engineering*, vol. 33, no. September 2020, p. 101816, 2021, doi: 10.1016/j.job.2020.101816.
- [71] A. Khan, H. S. Lee, and H. S. Kim, "Analysis of sensor-debonding failure in active vibration control of smart composite plate," *J Intell Mater Syst Struct*, vol. 28, no. 18, pp. 2603–2616, Nov. 2017, doi: 10.1177/1045389X17692052.
- [72] G. Park, C. R. Farrar, A. Rutherford, and A. N. Robertson, "Piezo-sensor self-diagnostics using electrical impedance measurements," 2004.
- [73] G. Busca, A. Cigada, and E. Zappa, "Vision device applied to damage identification in civil engineer structures," in *Conference*

Proceedings of the Society for Experimental Mechanics Series, 2014, pp. 192–203. [Online]. Available: <http://www.springer.com/series/8922>

- [74] E. Figueiredo, G. Park, C. R. Farrar, K. Worden, and J. Figueiras, “Machine learning algorithms for damage detection under operational and environmental variability,” *Struct Health Monit*, vol. 10, no. 6, pp. 559–572, Nov. 2011, doi: 10.1177/1475921710388971.
- [75] P. J. Chun, H. Yamashita, and S. Furukawa, “Bridge Damage Severity Quantification Using Multipoint Acceleration Measurement and Artificial Neural Networks,” *Shock and Vibration*, vol. 2015, 2015, doi: 10.1155/2015/789384.
- [76] B. Boashash and S. Ouelha, “Automatic signal abnormality detection using time-frequency features and machine learning: A newborn EEG seizure case study,” *Knowl Based Syst*, vol. 106, pp. 38–50, Aug. 2016, doi: 10.1016/j.knosys.2016.05.027.
- [77] U. Dackermann, J. Li, and B. Samali, “Dynamic-Based Damage Identification Using Neural Network Ensembles and Damage Index Method,” *Sensors Journal*, 2010.
- [78] B. Huang and H. S. Kim, “Frequency response analysis of a delaminated smart composite plate,” in *Journal of Intelligent Material Systems and Structures*, SAGE Publications Ltd, Jun. 2015, pp. 1091–1102. doi: 10.1177/1045389X14544141.
- [79] W. J. Staszewski, Chr. Boller, and G. R. Tomlinson, *Health monitoring of aerospace structures: smart sensor technologies and signal processing*. J. Wiley, 2004.
- [80] H. Huang and H. V. Burton, “Classification of in-plane failure modes for reinforced concrete frames with infills using machine learning,” *Journal of Building Engineering*, vol. 25, Sep. 2019, doi: 10.1016/j.jobbe.2019.100767.

- [81] R. Gutkin, C. J. Green, S. Vangrattanachai, S. T. Pinho, P. Robinson, and P. T. Curtis, "On acoustic emission for failure investigation in CFRP: Pattern recognition and peak frequency analyses," *Mech Syst Signal Process*, vol. 25, no. 4, pp. 1393–1407, May 2011, doi: 10.1016/j.ymssp.2010.11.014.
- [82] M. G. R. Sause, A. Gribov, A. R. Unwin, and S. Horn, "Pattern recognition approach to identify natural clusters of acoustic emission signals," *Pattern Recognit Lett*, vol. 33, no. 1, pp. 17–23, Jan. 2012, doi: 10.1016/j.patrec.2011.09.018.
- [83] C. M. Wen, S. L. Hung, C. S. Huang, and J. C. Jan, "Unsupervised fuzzy neural networks for damage detection of structures," *Struct Control Health Monit*, vol. 14, no. 1, pp. 144–161, Feb. 2007, doi: 10.1002/stc.116.
- [84] X. T. Zhou, Y. Q. Ni, and F. L. Zhang, "Damage localization of cable-supported bridges using modal frequency data and probabilistic neural network," *Math Probl Eng*, vol. 2014, 2014, doi: 10.1155/2014/837963.
- [85] A. Santos, E. Figueiredo, M. F. M. Silva, C. S. Sales, and J. C. W. A. Costa, "Machine learning algorithms for damage detection: Kernel-based approaches," *J Sound Vib*, vol. 363, pp. 584–599, 2016, doi: 10.1016/j.jsv.2015.11.008.
- [86] M. Silva, A. Santos, E. Figueiredo, R. Santos, C. Sales, and J. C. W. A. Costa, "A novel unsupervised approach based on a genetic algorithm for structural damage detection in bridges," *Eng Appl Artif Intell*, vol. 52, pp. 168–180, 2016, doi: 10.1016/j.engappai.2016.03.002.
- [87] V. Meruane, "Online Sequential Extreme Learning Machine for Vibration-Based Damage Assessment Using Transmissibility Data," *Journal of Computing in Civil Engineering*, vol. 30, no. 3, p. 04015042, May 2016, doi: 10.1061/(asce)cp.1943-5487.0000517.

- [88] A. Khan, H. S. Kim, and B. D. Youn, "Modeling and assessment of partially debonded piezoelectric sensor in smart composite laminates," *Int J Mech Sci*, vol. 131–132, pp. 26–36, Oct. 2017, doi: 10.1016/j.ijmecsci.2017.06.031.
- [89] M. Yuan *et al.*, "Prediction of stiffness degradation based on machine learning: Axial elastic modulus of [0m /90n]s composite laminates," *Compos Sci Technol*, vol. 218, p. 109186, Feb. 2022, doi: 10.1016/J.COMPSCITECH.2021.109186.
- [90] D. W. Abueidda, M. Almasri, R. Ammourah, U. Ravaioli, I. M. Jasiuk, and N. A. Sobh, "Prediction and optimization of mechanical properties of composites using convolutional neural networks," *Compos Struct*, vol. 227, no. August, p. 111264, 2019, doi: 10.1016/j.compstruct.2019.111264.
- [91] C. Feng, H. Zhang, S. Wang, Y. Li, H. Wang, and F. Yan, "Structural Damage Detection using Deep Convolutional Neural Network and Transfer Learning," *KSCE Journal of Civil Engineering*, vol. 23, no. 10, pp. 4493–4502, 2019, doi: 10.1007/s12205-019-0437-z.
- [92] N. Panda *et al.*, "Mesoscale informed parameter estimation through machine learning: A case-study in fracture modeling," *J Comput Phys*, vol. 420, p. 109719, 2020, doi: 10.1016/j.jcp.2020.109719.
- [93] A. Khan, N. Kim, J. K. Shin, H. S. Kim, and B. D. Youn, "Damage assessment of smart composite structures via machine learning: a review," *JMST Advances*, vol. 1, no. 1–2, pp. 107–124, 2019, doi: 10.1007/s42791-019-0012-2.
- [94] S. Soyoz and M. Q. Feng, "Long-term monitoring and identification of bridge structural parameters," *Computer-Aided Civil and Infrastructure Engineering*, vol. 24, no. 2, pp. 82–92, 2009, doi: 10.1111/j.1467-8667.2008.00572.x.

- [95] D. U. Sung, J. H. Oh, C. G. Kim, and C. S. Hong, "Impact monitoring of smart composite laminates using neural network and wavelet analysis," *J Intell Mater Syst Struct*, vol. 11, no. 3, pp. 180–190, Mar. 2000, doi: 10.1106/N5E7-M37Y-3MAR-2KFH.
- [96] D. Chetwynd *et al.*, "Damage Localisation in a Stiffened Composite Panel," 2008.
- [97] N. D. Hoang and Q. L. Nguyen, "Automatic Recognition of Asphalt Pavement Cracks Based on Image Processing and Machine Learning Approaches: A Comparative Study on Classifier Performance," *Math Probl Eng*, vol. 2018, 2018, doi: 10.1155/2018/6290498.
- [98] S. Das, A. Chattopadhyay, and A. N. Srivastava, "Classifying induced damage in composite plates using one-class support vector machines," *AIAA Journal*, vol. 48, no. 4, pp. 705–718, Apr. 2010, doi: 10.2514/1.37282.
- [99] N. Godin, S. Huguet, R. Gaertner, and L. Salmon, "Clustering of acoustic emission signals collected during tensile tests on unidirectional glass/polyester composite using supervised and unsupervised classifiers," *NDT and E International*, vol. 4, no. 37, pp. 253–264, Jun. 2004, doi: 10.1016/J.NDTEINT.2003.09.010.
- [100] L. Li, Y. Swolfs, I. Straumit, X. Yan, and S. V. Lomov, "Cluster analysis of acoustic emission signals for 2D and 3D woven carbon fiber/epoxy composites:," <http://dx.doi.org/10.1177/0021998315597742>, vol. 50, no. 14, pp. 1921–1935, Jul. 2016, doi: 10.1177/0021998315597742.
- [101] F. M. Russo, D. R. Mertz, K. H. Frank, and K. E. Wilson, "Design and Evaluation of Steel Bridges for Fatigue and Fracture REFERENCE MANUAL," 2016. [Online]. Available: <http://www.nhi.fhwa.dot.gov/training/nhistore.aspx>

- [102] E. El-Magd, H. Gese, R. Tham \$, H. Hooputra, and H. Werner, "Fracture Criteria for Automobile Crashworthiness Simulation of Wrought Aluminium Alloy Components Bruchkriterien fu È r Crash-Simulation von Komponenten aus Aluminium-Knetlegierungen in der Fahrzeugtechnik," 2001.
- [103] C. Clifton, M. Bruneau, G. Macrae, R. Leon, and A. Fussell, "Steel structures damage from the Christ church earthquake series of 2010 and 2011," 2011.
- [104] A. M. L. Adib, C. A. R. P. Baptista, M. J. R. Barboza, C. Haga, and C. C. F. Marques, "Aircraft engine bleed system tubes: Material and failure mode analysis," *Eng Fail Anal*, vol. 14, no. 8 SPEC. ISS., pp. 1605–1617, 2007, doi: 10.1016/j.engfailanal.2006.11.053.
- [105] D. L. Naik and R. Kiran, "Identification and characterization of fracture in metals using machine learning based texture recognition algorithms," *Eng Fract Mech*, vol. 219, Oct. 2019, doi: 10.1016/j.engfracmech.2019.106618.
- [106] H. Wang, W. Zhang, F. Sun, and W. Zhang, "A comparison study of machine learning based algorithms for fatigue crack growth calculation," *Materials*, vol. 10, no. 5, May 2017, doi: 10.3390/ma10050543.
- [107] A. Raja, S. T. Chukka, and R. Jayaganthan, "Prediction of fatigue crack growth behaviour in ultrafine grained al 2014 alloy using machine learning," *Metals (Basel)*, vol. 10, no. 10, pp. 1–13, Oct. 2020, doi: 10.3390/met10101349.
- [108] M. P. González and J. L. Zapico, "Seismic damage identification in buildings using neural networks and modal data," *Comput Struct*, vol. 86, no. 3–5, pp. 416–426, Feb. 2008, doi: 10.1016/j.compstruc.2007.02.021.

- [109] R. G. Kamble, N. R. Raykar, and D. N. Jadhav, "Machine learning approach to predict fatigue crack growth," *Mater Today Proc*, no. xxxx, 2020, doi: 10.1016/j.matpr.2020.07.535.
- [110] H. Baqasah *et al.*, "In-situ dynamic response measurement for damage quantification of 3D printed ABS cantilever beam under thermomechanical load," *Polymers (Basel)*, vol. 11, no. 12, Dec. 2019, doi: 10.3390/polym11122079.
- [111] B. A. Zai, M. A. Khan, K. A. Khan, and A. Mansoor, "A novel approach for damage quantification using the dynamic response of a metallic beam under thermo-mechanical loads," *J Sound Vib*, vol. 469, Mar. 2020, doi: 10.1016/j.jsv.2019.115134.
- [112] T. Fleet, K. Kamei, F. He, M. A. Khan, K. A. Khan, and A. Starr, "A machine learning approach to model interdependencies between dynamic response and crack propagation," *Sensors (Switzerland)*, vol. 20, no. 23. MDPI AG, pp. 1–13, Dec. 01, 2020. doi: 10.3390/s20236847.
- [113] D. Tibaduiza, M. Á. Torres-Arredondo, J. Vitola, M. Anaya, and F. Pozo, "A Damage Classification Approach for Structural Health Monitoring Using Machine Learning," *Complexity*, vol. 2018, 2018, doi: 10.1155/2018/5081283.
- [114] H. Hein and L. Jaanuska, "Comparison of machine learning methods for crack localization," *Acta et Commentationes Universitatis Tartuensis de Mathematica*, vol. 23, no. 1, pp. 125–142, 2019, doi: 10.12697/ACUTM.2019.23.13.
- [115] S. Chen, Z. C. Ong, W. H. Lam, K. S. Lim, and K. W. Lai, "Unsupervised Damage Identification Scheme Using PCA-Reduced Frequency Response Function and Waveform Chain Code Analysis," *International Journal of Structural Stability and Dynamics*, vol. 20, no. 8, Jul. 2020, doi: 10.1142/S0219455420500911.

- [116] S. Moradi and H. V. Burton, "Response surface analysis and optimization of controlled rocking steel braced frames," *Bulletin of Earthquake Engineering*, vol. 16, no. 10, pp. 4861–4892, Oct. 2018, doi: 10.1007/s10518-018-0373-1.
- [117] J. Seo, L. Dueñas-Osorio, J. I. Craig, and B. J. Goodno, "Metamodel-based regional vulnerability estimate of irregular steel moment-frame structures subjected to earthquake events," *Eng Struct*, vol. 45, pp. 585–597, Dec. 2012, doi: 10.1016/j.engstruct.2012.07.003.
- [118] J. S. Jeon, A. Shafieezadeh, and R. Desroches, "Statistical models for shear strength of RC beam-column joints using machine-learning techniques," *Earthq Eng Struct Dyn*, vol. 43, no. 14, pp. 2075–2095, Nov. 2014, doi: 10.1002/eqe.2437.
- [119] N. D. Hoang, X. L. Tran, and H. Nguyen, "Predicting ultimate bond strength of corroded reinforcement and surrounding concrete using a metaheuristic optimized least squares support vector regression model," *Neural Comput Appl*, vol. 32, no. 11, pp. 7289–7309, Jun. 2020, doi: 10.1007/s00521-019-04258-x.
- [120] M. Azimi and G. Pekcan, "Structural health monitoring using extremely compressed data through deep learning," *Computer-Aided Civil and Infrastructure Engineering*, vol. 35, no. 6, pp. 597–614, 2020, doi: 10.1111/mice.12517.
- [121] A. Athanasiou, A. Ebrahimkhanlou, J. Zaborac, T. Hrynyk, and S. Salamone, "A machine learning approach based on multifractal features for crack assessment of reinforced concrete shells," *Computer-Aided Civil and Infrastructure Engineering*, vol. 35, no. 6, pp. 565–578, Jun. 2020, doi: 10.1111/MICE.12509.
- [122] Z. Zhang and C. Sun, "Structural damage identification via physics-guided machine learning: a methodology integrating pattern recognition with finite element model updating," *Struct Health Monit*, 2020, doi: 10.1177/1475921720927488.

- [123] H. Sun, H. Burton, and J. Wallace, "Reconstructing seismic response demands across multiple tall buildings using kernel-based machine learning methods," *Struct Control Health Monit*, vol. 26, no. 7, Jul. 2019, doi: 10.1002/stc.2359.
- [124] J. H. Friedman, "GREEDY FUNCTION APPROXIMATION: A GRADIENT BOOSTING MACHINE 1," 2001.
- [125] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin, "Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation," Sep. 2013, [Online]. Available: <http://arxiv.org/abs/1309.6392>
- [126] A. Karpatne *et al.*, "Theory-guided Data Science: A New Paradigm for Scientific Discovery from Data," Dec. 2016, doi: 10.1109/TKDE.2017.2720168.
- [127] A. Stoll and P. Benner, "Machine Learning for Material Characterization with an Application for Predicting Mechanical Properties," no. MI, 2020, [Online]. Available: <http://arxiv.org/abs/2010.06010>
- [128] N. Muralidhar, M. R. Islam, M. Marwah, A. Karpatne, and N. Ramakrishnan, "Incorporating Prior Domain Knowledge into Deep Neural Networks," in *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, Institute of Electrical and Electronics Engineers Inc., Jan. 2019, pp. 36–45. doi: 10.1109/BigData.2018.8621955.
- [129] S. Pawar, O. San, B. Aksoylu, A. Rasheed, and T. Kvamsdal, "Physics guided machine learning using simplified theories," *Physics of Fluids*, vol. 33, no. 1, p. 011701, Jan. 2021, doi: 10.1063/5.0038929.

- [130] J. Chen and Y. Liu, "Probabilistic physics-guided machine learning for fatigue data analysis," *Expert Syst Appl*, vol. 168, p. 114316, Apr. 2021, doi: 10.1016/J.ESWA.2020.114316.
- [131] O. Erge and E. van Oort, "Combining Physics-Based and Data-Driven Modeling for Pressure Prediction in Well Construction," Jun. 2020, doi: 10.5281/ZENODO.3906912.
- [132] Y. Bhuiyan, "Physics-Based Approaches For Structural Health Monitoring And Nondestructive Evaluation With Ultrasonic Guided Waves," 2018.
- [133] Y. Bhuiyan and V. Giurgiutiu, "Experimental and Computational Analysis of Acoustic Emission Waveforms for SHM Applications".
- [134] Y. Bhuiyan, J. Bao, B. Poddar, and V. Giurgiutiu, "Toward identifying crack-length-related resonances in acoustic emission waveforms for structural health monitoring applications," *Struct Health Monit*, vol. 17, no. 3, pp. 577–585, 2018, doi: 10.1177/1475921717707356.
- [135] A. Ebrahimkhanlou and S. Salamone, "A Deep Learning Approach for Single-sensor Acoustic Emission Source Localization in Plate-like Structures," *Structural Health Monitoring 2017*, vol. 0, no. shm, pp. 2139–2146, 2017, doi: 10.12783/SHM2017/14103.
- [136] H. Luo and S. G. Paal, "Machine Learning–Based Backbone Curve Model of Reinforced Concrete Columns Subjected to Cyclic Loading Reversals," *Journal of Computing in Civil Engineering*, vol. 32, no. 5, p. 04018042, Sep. 2018, doi: 10.1061/(asce)cp.1943-5487.0000787.
- [137] K. Morfidis and K. Kostinakis, "Seismic parameters' combinations for the optimum prediction of the damage state of R/C buildings using neural networks," *Advances in Engineering Software*, vol. 106, pp. 1–16, Apr. 2017, doi: 10.1016/j.advengsoft.2017.01.001.
- [138] H. V. Burton, S. Sreekumar, M. Sharma, and H. Sun, "Estimating aftershock collapse vulnerability using mainshock intensity, structural

- response and physical damage indicators,” *Structural Safety*, vol. 68, pp. 85–96, Sep. 2017, doi: 10.1016/j.strusafe.2017.05.009.
- [139] S. Mangalathu and J. S. Jeon, “Classification of failure mode and prediction of shear strength for reinforced concrete beam-column joints using machine learning techniques,” *Eng Struct*, vol. 160, pp. 85–94, Apr. 2018, doi: 10.1016/j.engstruct.2018.01.008.
- [140] C. Coleman *et al.*, “Analysis of DAWN Bench, a Time-to-Accuracy Machine Learning Performance Benchmark,” *ACM SIGOPS Operating Systems Review*, vol. 53, no. 1, pp. 14–25, Jul. 2019, doi: 10.1145/3352020.3352024.
- [141] Z. Liu *et al.*, “Accuracy analyses and model comparison of machine learning adopted in building energy consumption prediction,” *Energy Exploration and Exploitation*, vol. 37, no. 4, pp. 1426–1451, Jul. 2019, doi: 10.1177/0144598718822400/ASSET/IMAGES/LARGE/10.1177_0144598718822400-FIG5.JPEG.
- [142] A. Mosavi, M. Salimi, S. F. Ardabili, T. Rabczuk, S. Shamshirband, and A. R. Varkonyi-Koczy, “State of the Art of Machine Learning Models in Energy Systems, a Systematic Review,” *Energies* 2019, Vol. 12, Page 1301, vol. 12, no. 7, p. 1301, Apr. 2019, doi: 10.3390/EN12071301.
- [143] T. Y. Deo, A. D. Patange, S. S. Pardeshi, R. Jegadeeshwaran, A. N. Khairnar, and H. S. Khade, “A White-Box SVM Framework and its Swarm-Based Optimization for Supervision of Toothed Milling Cutter through Characterization of Spindle Vibrations,” Dec. 2021, doi: 10.48550/arxiv.2112.08421.
- [144] M. Jorayeva, A. Akbulut, C. Catal, and A. Mishra, “Machine Learning-Based Software Defect Prediction for Mobile Applications: A Systematic Literature Review,” *Sensors* 2022, Vol. 22, Page 2551, vol. 22, no. 7, p. 2551, Mar. 2022, doi: 10.3390/S22072551.

- [145] M. Tanveer, T. Rajani, R. Rastogi, Y. H. Shao, and M. A. Ganaie, "Comprehensive review on twin support vector machines," *Ann Oper Res*, pp. 1–46, Mar. 2022, doi: 10.1007/S10479-022-04575-W/TABLES/8.
- [146] Y. M. Bichu, I. Hansa, A. Y. Bichu, P. Premjani, C. Flores-Mir, and N. R. Vaid, "Applications of artificial intelligence and machine learning in orthodontics: a scoping review," *Prog Orthod*, vol. 22, no. 1, pp. 1–11, Dec. 2021, doi: 10.1186/S40510-021-00361-9/TABLES/4.
- [147] O. M. A. Ali, S. W. Kareem, and A. S. Mohammed, "Evaluation of Electrocardiogram Signals Classification Using CNN, SVM, and LSTM Algorithm: A review," *8th IEC 2022 - International Engineering Conference: Towards Engineering Innovations and Sustainability*, pp. 185–191, 2022, doi: 10.1109/IEC54822.2022.9807511.
- [148] A. Rodomanov and Y. Nesterov, "Rates of superlinear convergence for classical quasi-Newton methods," *Math Program*, vol. 194, no. 1–2, pp. 159–190, Jul. 2022, doi: 10.1007/S10107-021-01622-5/METRICS.
- [149] H. Xu, "Adaptive Stochastic Conjugate Gradient optimization for temporal medical image registration," *LSU Master's Theses*, Jan. 2013, doi: 10.31390/gradschool_theses.324.
- [150] V. Cheung, J.-B. Thomas, P. Rhodes, S. Tominaga, and H. Sakai, "Spectral Reflectance Estimation from Camera Responses Using Local Optimal Dataset," *Journal of Imaging 2023, Vol. 9, Page 47*, vol. 9, no. 2, p. 47, Feb. 2023, doi: 10.3390/JIMAGING9020047.
- [151] A. Tharwat, "Parameter investigation of support vector machine classifier with kernel functions," *Knowl Inf Syst*, vol. 61, no. 3, pp. 1269–1302, Dec. 2019, doi: 10.1007/S10115-019-01335-4/TABLES/12.

- [152] R. Li, H. Gu, B. Hu, and Z. She, "Multi-feature fusion and damage identification of large generator stator insulation based on lamb wave detection and SVM method," *Sensors (Switzerland)*, vol. 19, no. 17, 2019, doi: 10.3390/s19173733.
- [153] A. Tharwat and A. E. Hassanien, "Chaotic antlion algorithm for parameter optimization of support vector machine," *Appl Intell*, vol. 48, no. 3, pp. 670–686, Mar. 2018, doi: 10.1007/s10489-017-0994-0.
- [154] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, no. 2, pp. 197–227, Jun. 2016, doi: 10.1007/S11749-016-0481-7/FIGURES/4.
- [155] A. Sarica, A. Cerasa, and A. Quattrone, "Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: A systematic review," *Front Aging Neurosci*, vol. 9, no. OCT, p. 284242, Oct. 2017, doi: 10.3389/FNAGI.2017.00329/BIBTEX.
- [156] M. Sheykhmousa, M. Mahdianpari, H. Ghanbari, F. Mohammadimanesh, P. Ghamisi, and S. Homayouni, "Support Vector Machine Versus Random Forest for Remote Sensing Image Classification: A Meta-Analysis and Systematic Review," *IEEE J Sel Top Appl Earth Obs Remote Sens*, vol. 13, pp. 6308–6325, 2020, doi: 10.1109/JSTARS.2020.3026724.
- [157] M. Sipper and J. H. Moore, "AddGBoost: A gradient boosting-style algorithm based on strong learners," *Machine Learning with Applications*, vol. 7, p. 100243, Mar. 2022, doi: 10.1016/J.MLWA.2021.100243.
- [158] A. Mayr, H. Binder, O. Gefeller, and M. Schmid, "The evolution of boosting algorithms: From machine learning to statistical modelling," *Methods Inf Med*, vol. 53, no. 6, pp. 419–427, 2014, doi: 10.3414/ME13-01-0122/ID/JR0122-12/BIB.

- [159] A. Mayr, H. Binder, O. Gefeller, and M. Schmid, "Extending statistical boosting: An overview of recent methodological developments," *Methods Inf Med*, vol. 53, no. 6, pp. 428–435, 2014, doi: 10.3414/ME13-01-0123/ID/JR0123-13/BIB.
- [160] W. Zhang *et al.*, "State-of-the-art review of soft computing applications in underground excavations," *Geoscience Frontiers*, vol. 11, no. 4, pp. 1095–1106, Jul. 2020, doi: 10.1016/J.GSF.2019.12.003.
- [161] J. Song, J. Zhao, F. Dong, J. Zhao, Z. Qian, and Q. Zhang, "A Novel Regression Modeling Method for PMSLM Structural Design Optimization Using a Distance-Weighted KNN Algorithm," *IEEE Trans Ind Appl*, vol. 54, no. 5, pp. 4198–4206, Sep. 2018, doi: 10.1109/TIA.2018.2836953.
- [162] H. Sarmadi and A. Karamodin, "A novel anomaly detection method based on adaptive Mahalanobis-squared distance and one-class kNN rule for structural health monitoring under environmental effects," *Mech Syst Signal Process*, vol. 140, p. 106495, 2020, doi: 10.1016/j.ymssp.2019.106495.
- [163] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A brief review of nearest neighbor algorithm for learning and classification," *2019 International Conference on Intelligent Computing and Control Systems, ICCS 2019*, pp. 1255–1260, May 2019, doi: 10.1109/ICCS45141.2019.9065747.
- [164] J. E. Miranda-Vega, M. Rivas-Lopez, and W. Flores-Fuentes, "k-Nearest Neighbor Classification for Pattern Recognition of a Reference Source Light for Machine Vision System," *IEEE Sens J*, vol. XX, no. XX, pp. 1–1, 2020, doi: 10.1109/jsen.2020.3024094.
- [165] M. J. Hasan and J. M. Kim, "Fault Detection of a Spherical Tank Using a Genetic Algorithm-Based Hybrid Feature Pool and k-Nearest

- Neighbor Algorithm,” *Energies* 2019, Vol. 12, Page 991, vol. 12, no. 6, p. 991, Mar. 2019, doi: 10.3390/EN12060991.
- [166] J. M. Pereira, M. Basto, and A. F. da Silva, “The Logistic Lasso and Ridge Regression in Predicting Corporate Failure,” *Procedia Economics and Finance*, vol. 39, pp. 634–641, 2016, doi: 10.1016/S2212-5671(16)30310-0.
- [167] R. K. AE Hoerl, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.
- [168] P. Buteneers, K. Caluwaerts, J. Dambre, D. Verstraeten, and B. Schrauwen, “Optimized parameter search for large datasets of the regularization parameter and feature selection for ridge regression,” *Neural Process Lett*, vol. 38, no. 3, pp. 403–416, Dec. 2013, doi: 10.1007/S11063-013-9279-8/TABLES/1.
- [169] S. Bhattacharya, K. Kalita, R. Čep, and S. Chakraborty, “A Comparative Analysis on Prediction Performance of Regression Models during Machining of Composite Materials,” *Materials* 2021, Vol. 14, Page 6689, vol. 14, no. 21, p. 6689, Nov. 2021, doi: 10.3390/MA14216689.
- [170] M. Rao, X. Yang, D. Wei, Y. Chen, L. Meng, and M. J. Zuo, “Structure fatigue crack length estimation and prediction using ultrasonic wave data based on ensemble linear regression and Paris’s law,” *Int J Progn Health Manag*, vol. 11, no. October, pp. 1–14, 2020.
- [171] E. J. Weyuker, T. J. Ostrand, and R. M. Bell, “Comparing the effectiveness of several modeling methods for fault prediction,” *Empir Softw Eng*, vol. 15, no. 3, pp. 277–295, Jun. 2010, doi: 10.1007/S10664-009-9111-2.

- [172] X. Yang and W. Wen, "Ridge and Lasso Regression Models for Cross-Version Defect Prediction," *IEEE Trans Reliab*, vol. 67, no. 3, pp. 885–896, Sep. 2018, doi: 10.1109/TR.2018.2847353.
- [173] A. K. Md. Ehsanes Saleh, B. M. Golam Kibria, and F. George, "Comparative Study of LASSO, Ridge Regression, Preliminary Test and Stein-type Estimators for the Sparse Gaussian Regression Model," *Statistics, Optimization & Information Computing*, vol. 7, no. 4, pp. 626–641, Dec. 2019, doi: 10.19139/SOIC-2310-5070-713.
- [174] Dahliyusmanto, T. Herawan, S. Yulina, and A. H. Abdullah, "A feature selection algorithm for anomaly detection in grid environment using k-fold cross validation technique," *Advances in Intelligent Systems and Computing*, vol. 549 AISC, pp. 619–630, 2017, doi: 10.1007/978-3-319-51281-5_62/TABLES/4.
- [175] B. K. F. G. AME Saleh, "Comparative study of lasso, ridge regression, preliminary test and stein-type estimators for the sparse gaussian regression model," *Stat Optim Inform Comput*, vol. 7, no. 4, pp. 626–641, 2019.
- [176] S. Ding, H. Li, C. Su, J. Yu, and F. Jin, "Evolutionary artificial neural networks: a review," *Artificial Intelligence Review 2011* 39:3, vol. 39, no. 3, pp. 251–260, Jun. 2011, doi: 10.1007/S10462-011-9270-6.
- [177] A. C. Neves, I. González, J. Leander, and R. Karoumi, "Structural health monitoring of bridges: a model-free ANN-based approach to damage detection," *J Civ Struct Health Monit*, vol. 7, no. 5, pp. 689–702, 2017, doi: 10.1007/s13349-017-0252-5.
- [178] S. Kekez and J. Kubica, "Connecting concrete technology and machine learning: Proposal for application of ANNs and CNT/concrete composites in structural health monitoring," *RSC Adv*, vol. 10, no. 39, pp. 23038–23048, 2020, doi: 10.1039/d0ra03450a.

- [179] O. I. Abiodun *et al.*, “Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition,” *IEEE Access*, vol. 7, pp. 158820–158846, 2019, doi: 10.1109/ACCESS.2019.2945545.
- [180] H. S. Sachdev, “Choosing number of Hidden Layers and number of hidden neurons in Neural Networks | LinkedIn,” 2020. Accessed: Dec. 07, 2022. [Online]. Available: <https://www.linkedin.com/pulse/choosing-number-hidden-layers-neurons-neural-networks-sachdev/>
- [181] X. Li, K. Cheng, T. Huang, Z. Qiu, and S. Tan, “Research on short term prediction method of thermal hydraulic transient operation parameters based on automated deep learning,” *Ann Nucl Energy*, vol. 165, p. 108777, Jan. 2022, doi: 10.1016/J.ANUCENE.2021.108777.
- [182] O. Abdeljaber, O. Avci, M. S. Kiranyaz, B. Boashash, H. Sodano, and D. J. Inman, “1-D CNNs for structural damage detection: Verification on a structural health monitoring benchmark data,” *Neurocomputing*, vol. 275, pp. 1308–1317, 2018, doi: 10.1016/j.neucom.2017.09.069.
- [183] X. Ma, X. He, and Z. C. Tu, “Prediction of fatigue–crack growth with neural network-based increment learning scheme,” *Eng Fract Mech*, vol. 241, p. 107402, Jan. 2021, doi: 10.1016/J.ENGFRACMECH.2020.107402.
- [184] H. Alqahtani, S. Bharadwaj, and A. Ray, “Classification of fatigue crack damage in polycrystalline alloy structures using convolutional neural networks,” *Eng Fail Anal*, vol. 119, no. September 2020, p. 104908, 2021, doi: 10.1016/j.engfailanal.2020.104908.
- [185] A. Shrestha, “Deep Learning-Based Real-Time Auto Classification,” pp. 1–21, 2020, doi: doi.org/10.3390/s20092710.
- [186] G. Chen, T. Li, Q. Chen, S. Ren, C. Wang, and S. Li, “Application of deep learning neural network to identify collision load conditions

- based on permanent plastic deformation of shell structures,” *Comput Mech*, vol. 64, no. 2, pp. 435–449, 2019, doi: 10.1007/s00466-019-01706-2.
- [187] X. Li, C. C. Roth, and D. Mohr, “Machine-learning based temperature- and rate-dependent plasticity model: Application to analysis of fracture experiments on DP steel,” *Int J Plast*, vol. 118, no. January, pp. 320–344, 2019, doi: 10.1016/j.ijplas.2019.02.012.
- [188] R. A. Rajagukguk, R. A. A. Ramadhan, and H. J. Lee, “A Review on Deep Learning Models for Forecasting Time Series Data of Solar Irradiance and Photovoltaic Power,” *Energies 2020, Vol. 13, Page 6623*, vol. 13, no. 24, p. 6623, Dec. 2020, doi: 10.3390/EN13246623.
- [189] A.-N. Sharkawy, “Principle of Neural Network and Its Main Types: Review,” *Journal of Advances in Applied & Computational Mathematics*, vol. 7, pp. 8–19, Aug. 2020, doi: 10.15377/2409-5761.2020.07.2.
- [190] K. O. Cho and H. J. Jang, “Comparison of different input modalities and network structures for deep learning-based seizure detection,” *Scientific Reports 2020 10:1*, vol. 10, no. 1, pp. 1–11, Jan. 2020, doi: 10.1038/s41598-019-56958-y.
- [191] L. Zhi, Y. Zhu, H. Wang, Z. Xu, and Z. Man, “A recurrent neural network for modeling crack growth of aluminium alloy,” *Neural Comput Appl*, vol. 27, no. 1, pp. 197–203, 2016, doi: 10.1007/s00521-014-1712-z.
- [192] J. Yang *et al.*, “A hierarchical deep convolutional neural network and gated recurrent unit framework for structural damage detection,” *Inf Sci (N Y)*, vol. 540, pp. 117–130, 2020, doi: 10.1016/j.ins.2020.05.090.
- [193] K. Kamei, M. A. Khan, and K. A. Khan, “Characterising Modal Behaviour of a Cantilever Beam at Different Heating Rates for

Isothermal Conditions,” *Applied Sciences* 2021, Vol. 11, Page 4375, vol. 11, no. 10, p. 4375, May 2021, doi: 10.3390/APP11104375.

- [194] F. He, M. Khan, and S. Aldosari, “Interdependencies between Dynamic Response and Crack Growth in a 3D-Printed Acrylonitrile Butadiene Styrene (ABS) Cantilever Beam under Thermo-Mechanical Loads,” *Polymers* 2022, Vol. 14, Page 982, vol. 14, no. 5, p. 982, Feb. 2022, doi: 10.3390/POLYM14050982.
- [195] H. H. Zedan Khalel, M. Khan, and A. Starr, “Dynamic response-based crack resistance analysis of fibre reinforced concrete specimens under different temperatures and crack depths,” *Journal of Building Engineering*, vol. 66, p. 105865, May 2023, doi: 10.1016/J.JOBE.2023.105865.
- [196] Y. L. A. Alshammari, F. He, and M. A. Khan, “Modelling and Investigation of Crack Growth for 3D-Printed Acrylonitrile Butadiene Styrene (ABS) with Various Printing Parameters and Ambient Temperatures,” *Polymers* 2021, Vol. 13, Page 3737, vol. 13, no. 21, p. 3737, Oct. 2021, doi: 10.3390/POLYM13213737.
- [197] K. Kamei, M. A. Khan, C. : Kamei, and K. ; Khan, “Investigating the Structural Dynamics and Crack Propagation Behavior under Uniform and Non-Uniform Temperature Conditions,” *Materials* 2021, Vol. 14, Page 7071, vol. 14, no. 22, p. 7071, Nov. 2021, doi: 10.3390/MA14227071.
- [198] F. He, H. Ning, and M. Khan, “Effect of 3D Printing Process Parameters on Damping Characteristic of Cantilever Beams Fabricated Using Material Extrusion,” *Polymers* 2023, Vol. 15, Page 257, vol. 15, no. 2, p. 257, Jan. 2023, doi: 10.3390/POLYM15020257.
- [199] S. Chatterjee, “A New Coefficient of Correlation,” *J Am Stat Assoc*, vol. 116, no. 536, pp. 2009–2022, 2021, doi:

10.1080/01621459.2020.1758115/SUPPL_FILE/UASA_A_1758115_SM2340.PDF.

- [200] C. Thirumalai, S. A. Chandhini, and M. Vaishnavi, "Analysing the concrete compressive strength using Pearson and Spearman," *Proceedings of the International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017*, vol. 2, pp. 215–218, Apr. 2017, doi: 10.1109/ICECA.2017.8212799.
- [201] T. W. MacFarland and J. M. Yates, "Spearman's Rank-Difference Coefficient of Correlation," *Introduction to Nonparametric Statistics for the Biological Sciences Using R*, pp. 249–297, Jan. 2016, doi: 10.1007/978-3-319-30634-6_8.
- [202] Y. Liu, "A Short Note on Spearman Correlation: Impact of Tied Observations," *Social Science Research Network*, Mar. 2017, doi: 10.2139/SSRN.2933193.
- [203] I. M. U. Multi-point, S. Acceleration, and M. Results, "Development of a Machine Learning-Based Damage," 2020.
- [204] Y. Bao, Z. Tang, and H. Li, "Compressive-sensing data reconstruction for structural health monitoring: a machine-learning approach," *Struct Health Monit*, vol. 19, no. 1, pp. 293–304, 2020, doi: 10.1177/1475921719844039.
- [205] M. H. Rafiei and H. Adeli, "A novel machine learning-based algorithm to detect damage in high-rise building structures," *Structural Design of Tall and Special Buildings*, vol. 26, no. 18, Dec. 2017, doi: 10.1002/tal.1400.
- [206] F. N. Catbas and M. Malekzadeh, "A machine learning-based algorithm for processing massive data collected from the mechanical components of movable bridges," *Autom Constr*, vol. 72, pp. 269–278, 2016, doi: 10.1016/j.autcon.2016.02.008.

- [207] H. Liu, S. Liu, Z. Liu, N. Mrad, and A. S. Milani, "Data-Driven Approaches for Characterization of Delamination Damage in Composite Materials," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 3, pp. 2532–2542, Mar. 2021, doi: 10.1109/TIE.2020.2973877.
- [208] Y. Liu, K. Zhao, F. Hou, and X. Hao, "Crack Extension Life and Critical Crack Length Prediction Based on XGBoost," 2022, Accessed: May 29, 2023. [Online]. Available: <https://ceur-ws.org/Vol-3304/paper11.pdf>
- [209] Z. Wang *et al.*, "Acoustic emission characterization of natural fiber reinforced plastic composite machining using a random forest machine learning model," *Journal of Manufacturing Science and Engineering, Transactions of the ASME*, vol. 142, no. 3, Mar. 2020, doi: 10.1115/1.4045945/1072440.
- [210] Z. WANG, J. Chang, and Q. ZHU, "SVR Prediction Algorithm for Crack Propagation of Aviation Aluminum Alloy," *Authorea Preprints*, Jun. 2020, doi: 10.22541/AU.159196563.31132697.
- [211] L. Zhi, Z. Man, Z. Cao, M. Tao, and P. Wang, "Modeling crack growth of aluminum alloy under variable-Amplitude loading using dynamic neural network," *International Conference on Advanced Mechatronic Systems, ICAMechS*, vol. 0, pp. 453–458, 2016, doi: 10.1109/ICAMechS.2016.7813490.
- [212] P. Yasnii, O. Pastukh, Y. Pyndus, N. S. Lutsyk, and I. S. Didych, "Prediction of the Diagrams of Fatigue Fracture of D16T Aluminum Alloy by the Methods of Machine Learning," *Materials Science*, vol. 54, no. 3, pp. 333–338, 2018, doi: 10.1007/s11003-018-0189-9.
- [213] J. Zhang and T. Sato, "Support vector regression for structural identification," *Smart Structures and Materials 2005: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace*

Systems, vol. 5765, no. May 2005, p. 720, 2005, doi: 10.1117/12.598384.

- [214] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, “A comprehensive survey on support vector machine classification: Applications, challenges and trends,” *Neurocomputing*, 2020, doi: 10.1016/J.NEUCOM.2019.10.118.
- [215] T. Peltola, “Local Interpretable Model-agnostic Explanations of Bayesian Predictive Models via Kullback-Leibler Projections,” Oct. 2018, doi: 10.48550/arxiv.1810.02678.
- [216] K. Pekala, K. Woznica, and P. Biecek, “Triplot: model agnostic measures and visualisations for variable importance in predictive models that take into account the hierarchical correlation structure,” Apr. 2021, doi: 10.48550/arxiv.2104.03403.
- [217] J. Jiarpakdee, C. K. Tantithamthavorn, H. K. Dam, and J. Grundy, “An Empirical Study of Model-Agnostic Techniques for Defect Prediction Models,” *IEEE Transactions on Software Engineering*, vol. 48, no. 1, pp. 166–185, Jan. 2022, doi: 10.1109/TSE.2020.2982385.
- [218] R. Elshaw, M. Maher, and S. Sakr, “Automated Machine Learning: State-of-The-Art and Open Challenges,” Jun. 2019, Accessed: Jul. 07, 2023. [Online]. Available: <https://arxiv.org/abs/1906.02287v2>
- [219] F. Hutter, L. Kotthoff, and J. Vanschoren, “Automated Machine Learning,” p. 219, 2019, doi: 10.1007/978-3-030-05318-5.
- [220] Q. Yao *et al.*, “Taking Human out of Learning Applications: A Survey on Automated Machine Learning,” Oct. 2018, Accessed: Jul. 07, 2023. [Online]. Available: <https://arxiv.org/abs/1810.13306v4>
- [221] M. Bahri, F. Salutari, A. Putina, and M. Sozio, “AutoML: state of the art with a focus on anomaly detection, challenges, and research directions,” *Int J Data Sci Anal*, vol. 14, no. 2, pp. 113–126, Aug. 2022, doi: 10.1007/S41060-022-00309-0/TABLES/1.

- [222] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, and D. J. Inman, "A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications," *Mech Syst Signal Process*, vol. 147, p. 107077, 2021, doi: 10.1016/j.ymssp.2020.107077.
- [223] B. Jaishi and W.-X. Ren, "Structural Finite Element Model Updating Using Ambient Vibration Test Results," 2018, doi: 10.1061/ASCE0733-94452005131:4617.
- [224] J. E. Mottershead, M. Link, and M. I. Friswell, "The sensitivity method in finite element model updating: A tutorial," *Mech Syst Signal Process*, vol. 25, no. 7, pp. 2275–2296, 2011, doi: 10.1016/j.ymssp.2010.10.012.
- [225] P. E. Leser *et al.*, "Probabilistic fatigue damage prognosis using surrogate models trained via three-dimensional finite element analysis," *Struct Health Monit*, vol. 16, no. 3, pp. 291–308, 2017, doi: 10.1177/1475921716643298.
- [226] X. Q. Li, Q. J. Chen, and Z. De Ding, "Structural damage diagnosis and fine scale finite element intelligence simulation of long span cable stayed bridges," *Cluster Comput*, vol. 22, no. s2, pp. 4101–4107, 2019, doi: 10.1007/s10586-017-1515-y.
- [227] Q. Song *et al.*, "Micro-crack detection method of steel beam surface using stacked autoencoders on massive full-scale sensing strains," *Struct Health Monit*, vol. 19, no. 4, pp. 1175–1187, 2020, doi: 10.1177/1475921719879965.
- [228] A. J. Lew, C. H. Yu, Y. C. Hsu, and M. J. Buehler, "Deep learning model to predict fracture mechanisms of graphene," *npj 2D Materials and Applications* 2021 5:1, vol. 5, no. 1, pp. 1–8, Apr. 2021, doi: 10.1038/s41699-021-00228-x.

- [229] M. J. Buehler, "Modeling Atomistic Dynamic Fracture Mechanisms Using a Progressive Transformer Diffusion Model," *Journal of Applied Mechanics, Transactions ASME*, vol. 89, no. 12, Dec. 2022, doi: 10.1115/1.4055730/6918487/JAM-22-1281.PDF.
- [230] R. Perera, D. Guzzetti, and V. Agrawal, "Graph neural networks for simulating crack coalescence and propagation in brittle materials," *Comput Methods Appl Mech Eng*, vol. 395, p. 115021, May 2022, doi: 10.1016/J.CMA.2022.115021.
- [231] M. S. R. Elapolu, M. I. R. Shishir, and A. Tabarraei, "A novel approach for studying crack propagation in polycrystalline graphene using machine learning algorithms," *Comput Mater Sci*, vol. 201, p. 110878, Jan. 2022, doi: 10.1016/J.COMMATSCI.2021.110878.
- [232] Z. Chang, Z. Wan, Y. Xu, E. Schlangen, and B. Šavija, "Convolutional neural network for predicting crack pattern and stress-crack width curve of air-void structure in 3D printed concrete," *Eng Fract Mech*, vol. 271, p. 108624, Aug. 2022, doi: 10.1016/J.ENGFRACMECH.2022.108624.
- [233] T. Regan, C. Beale, and M. Inalpolat, "Wind Turbine Blade Damage Detection Using Supervised Machine Learning Algorithms," *Journal of Vibration and Acoustics, Transactions of the ASME*, vol. 139, no. 6, pp. 1–14, 2017, doi: 10.1115/1.4036951.
- [234] R. Miorelli, A. Kulakovskiy, O. Mesnil, and O. D'Almeida, "Automatic defect localization and characterization through machine learning based inversion for guided wave imaging in SHM," *AIP Conf Proc*, vol. 2102, 2019, doi: 10.1063/1.5099771.
- [235] K. Chauhan *et al.*, "Automated Machine Learning: The New Wave of Machine Learning," *2nd International Conference on Innovative Mechanisms for Industry Applications, ICIMIA 2020 - Conference Proceedings*, pp. 205–212, Mar. 2020, doi: 10.1109/ICIMIA48430.2020.9074859.

- [236] P. D. Yoo, M. H. Kim, and T. Jan, "Machine learning techniques and use of event information for stock market prediction: A survey and evaluation," *Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet*, vol. 2, pp. 835–841, 2005, doi: 10.1109/CIMCA.2005.1631572.
- [237] T. Siddique, D. Barua, Z. Ferdous, and A. Chakrabarty, "Automated farming prediction," *2017 Intelligent Systems Conference, IntelliSys 2017*, vol. 2018-January, pp. 757–763, Mar. 2018, doi: 10.1109/INTELLISYS.2017.8324214.
- [238] K. Y. Li et al., "Toward Automated Machine Learning-Based Hyperspectral Image Analysis in Crop Yield and Biomass Estimation," *Remote Sensing 2022, Vol. 14, Page 1114*, vol. 14, no. 5, p. 1114, Feb. 2022, doi: 10.3390/RS14051114.
- [239] P. Bhasha, J. Suresh Babu, M. N. Vadlamudi, K. Abraham, and S. K. Sarangi, "Automated Crop Yield Prediction System Using Machine Learning Algorithm," *J Algebr Stat*, vol. 13, no. 3, pp. 2512–2522, Jul. 2022, Accessed: Jul. 07, 2023. [Online]. Available: <https://www.publishoa.com/index.php/journal/article/view/908>
- [240] M. Bahri, F. Salutari, A. Putina, and M. Sozio, "AutoML: state of the art with a focus on anomaly detection, challenges, and research directions," *Int J Data Sci Anal*, vol. 14, no. 2, pp. 113–126, Aug. 2022, doi: 10.1007/S41060-022-00309-0/TABLES/1.
- [241] J. Błaszczyński, A. T. de Almeida Filho, A. Matuszyk, M. Szeląg, and R. Słowiński, "Auto loan fraud detection using dominance-based rough set approach versus machine learning methods," *Expert Syst Appl*, vol. 163, p. 113740, Jan. 2021, doi: 10.1016/J.ESWA.2020.113740.

- [242] A. Mustafa and M. Rahimi Azghadi, "Automated Machine Learning for Healthcare and Clinical Notes Analysis," *Computers 2021*, Vol. 10, Page 24, vol. 10, no. 2, p. 24, Feb. 2021, doi: 10.3390/COMPUTERS10020024.
- [243] J. Waring, C. Lindvall, and R. Umeton, "Automated machine learning: Review of the state-of-the-art and opportunities for healthcare," *Artif Intell Med*, vol. 104, p. 101822, Apr. 2020, doi: 10.1016/J.ARTMED.2020.101822.
- [244] R. Zeleznik *et al.*, "Deep convolutional neural networks to predict cardiovascular risk from computed tomography," *Nature Communications 2021 12:1*, vol. 12, no. 1, pp. 1–9, Jan. 2021, doi: 10.1038/s41467-021-20966-2.
- [245] M. Padmanabhan, P. Yuan, G. Chada, and H. Van Nguyen, "Physician-Friendly Machine Learning: A Case Study with Cardiovascular Disease Risk Prediction," *Journal of Clinical Medicine 2019*, Vol. 8, Page 1050, vol. 8, no. 7, p. 1050, Jul. 2019, doi: 10.3390/JCM8071050.
- [246] A. M. Alaa, T. Bolton, E. Di Angelantonio, J. H. F. Rudd, and M. van der Schaar, "Cardiovascular disease risk prediction using automated machine learning: A prospective study of 423,604 UK Biobank participants," *PLoS One*, vol. 14, no. 5, p. e0213653, May 2019, doi: 10.1371/JOURNAL.PONE.0213653.
- [247] A. Alsharef, K. Aggarwal, Sonia, M. Kumar, and A. Mishra, "Review of ML and AutoML Solutions to Forecast Time-Series Data," *Archives of Computational Methods in Engineering*, vol. 29, no. 7, pp. 5297–5311, Nov. 2022, doi: 10.1007/S11831-022-09765-0/TABLES/2.
- [248] L. Ferreira, A. Pilastri, C. M. Martins, P. M. Pires, and P. Cortez, "A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost," *Proceedings of the International Joint Conference on*

Neural Networks, vol. 2021-July, Jul. 2021, doi: 10.1109/IJCNN52387.2021.9534091.

- [249] "H2O Driverless AI." Accessed: Jul. 13, 2023. [Online]. Available: <https://h2o.ai/platform/ai-cloud/make/h2o-driverless-ai/>
- [250] G. Chen, W. Tang, S. Chen, S. Wang, and H. Cui, "Prediction of Self-Healing of Engineered Cementitious Composite Using Machine Learning Approaches," *Applied Sciences* 2022, Vol. 12, Page 3605, vol. 12, no. 7, p. 3605, Apr. 2022, doi: 10.3390/APP12073605.
- [251] S. Pfingstl, Y. N. Schoebel, and M. Zimmermann, "Reinforcement Learning for Structural Health Monitoring based on Inspection Data," *Materials Research Proceedings*, vol. 18, pp. 203–210, 2021, doi: 10.21741/9781644901311-24.

Appendices

Appendix A Ethical approval letter



17 August 2023

Dear Mrs Omar ,

Reference: CURES/20388/2023

Project ID: 23505

Title: Investigate Machine Learning approaches to model the interdependencies between applied load and crack propagation in Mechanical structure

Thank you for your application to the Cranfield University Research Ethics System (CURES).

We are pleased to inform you your CURES application, reference CURES/20388/2023 has been reviewed. You may now proceed with the research activities you have sought approval for.

If you have any queries, please contact CURES Support.

We wish you every success with your project.

Regards,

CURES Team

Page 1 of 1

Figure_Apx A-1 Ethical Approval Letter

Appendix B Python code of machine learning models

B.1 Code of ABS machine learning model

```
1  # Import necessary libraries
2  import pandas as pd
3  import numpy as np
4  from sklearn.model_selection import train_test_split, ShuffleSplit, cross_val_score
5  from sklearn.linear_model import LinearRegression
6  from sklearn.tree import DecisionTreeRegressor
7  from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
8  from sklearn.svm import SVR
9  from sklearn.neural_network import MLPRegressor
10 from sklearn.linear_model import Lasso, Ridge
11 from sklearn.metrics import mean_squared_error, r2_score
12 import matplotlib.pyplot as plt
13 from sklearn.impute import SimpleImputer
14 from sklearn.preprocessing import StandardScaler
15 from sklearn.neighbors import KNeighborsRegressor
16 from scipy.stats import pearsonr
17 import seaborn as sns
18 # Load the training and validation datasets
19 df = pd.read_csv("ABS_Val_dataset.csv")
20 # check for missing values
21 missing = df.isnull().sum()
22 print(missing)
23 # handle missing values
24 imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
25 X = df.iloc[:, :-1]
26 X = imputer.fit_transform(X)
27 # Plot the correlation matrix between features in the training dataset
28 sns.heatmap(df.corr(), annot=True)
29 plt.show()
30 # Calculate the correlation coefficient for each feature-target pair
31 correlations = {}
32 for feature in df.columns[:-1]:
33     corr, _ = pearsonr(df[feature], df["Crack_depth"])
34     correlations[feature] = corr
35 # Print the correlation coefficients
36 print(correlations)
37 # Create a scatter matrix of the data
38 for feature in df.columns[:-1]:
39     corr, _ = pearsonr(df[feature], df["Crack_depth"])
40     plt.figure()
41     plt.scatter(df[feature], df["Crack_depth"])
42     plt.title(f"{feature} vs Crack_depth\n r={corr:.2f}")
43     plt.xlabel(feature)
44     plt.ylabel("Crack_depth")
45     plt.show()
46
```

Figure_Apx B-1 Code of ABS Model with all Features

```

1 from scipy import stats
2 # assuming df is your DataFrame and it has columns: feature_1, feature_2, feature_3, feature_4, and
3 # Calculate Spearman's correlation coefficient for each feature
4 for feature in df.columns[:-1]:
5     spearman_corr, p_value = stats.spearmanr(df[feature], df["Crack_depth"])
6     print(f"Spearman's correlation coefficient between {feature} and predicted crack depth: {spearman_corr}")
7 from sklearn.preprocessing import StandardScaler
8 from sklearn.model_selection import ShuffleSplit, cross_val_score
9 from sklearn.neighbors import KNeighborsRegressor
10 from sklearn.linear_model import Lasso, Ridge
11 from sklearn.metrics import mean_squared_error
12 import numpy as np
13 import matplotlib.pyplot as plt
14 # Prepare the data
15 X = df[['NF', 'AMP']].values
16 y = df.iloc[:, -1].values
17 scaler = StandardScaler()
18 X = scaler.fit_transform(X)
19 # Shuffle the data and define the cross-validation iterator
20 cv = ShuffleSplit(n_splits=5, test_size=0.3)
21 # Define the list of algorithms
22 algorithms = [KNeighborsRegressor(n_neighbors=3), Lasso(alpha=0.01, max_iter=10), Ridge(alpha=0.01)]
23 # Train and evaluate each algorithm using cross-validation
24 for algorithm in algorithms:
25     # Compute the cross-validation scores
26     cv_scores = cross_val_score(algorithm, X, y, cv=cv, scoring='neg_mean_squared_error')
27     # Compute the mean of the scores
28     mean_score = -np.mean(cv_scores)
29     # Fit the model on the full data
30     algorithm.fit(X, y)
31     print(f"{algorithm.__class__.__name__} - Training MSE: {mean_score:.3f}")
32     # Evaluate the model on the testing data
33     y_pred = algorithm.predict(X)
34     mse = mean_squared_error(y, y_pred)
35     print(f"{algorithm.__class__.__name__} - Test MSE: {mse:.3f}")
36     # Compute the MSE of the baseline model
37     y_mean = y.mean()
38     y_pred_baseline = [y_mean] * len(y)
39     mse_baseline = mean_squared_error(y, y_pred_baseline)
40     print(f"MSE of baseline: {mse_baseline:.3f}")
41     # Compare the MSE of the model to the baseline
42     if mse < mse_baseline:
43         print("Model outperforms baseline")
44     else:
45         print("Baseline outperforms model")
46     # Plot the predicted values and the actual values
47     fig, ax = plt.subplots()
48     ax.plot(np.arange(len(y)), y)
49     ax.plot(np.arange(len(y_pred)), y_pred)
50     ax.set_xlabel('Actual Crack Depth mm')
51     ax.set_ylabel('Predicted Crack Depth mm')
52     ax.set_title(f'{algorithm.__class__.__name__}')
53     ax.legend(["Actual", "Predicted"])
54     plt.show()

```

Figure_Apx B-2 Code of ABS Model with Selected Features

B.2 Code of Aluminum machine learning model

```

1 # Import necessary libraries
2 import pandas as pd
3 import numpy as np
4 from sklearn.model_selection import train_test_split, cross_val_score, ShuffleSplit
5 from sklearn.linear_model import LinearRegression
6 from sklearn.tree import DecisionTreeRegressor
7 from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
8 from sklearn.svm import SVR
9 from sklearn.neural_network import MLPRegressor
10 from sklearn.linear_model import Lasso, Ridge
11 from sklearn.metrics import mean_squared_error, r2_score
12 import matplotlib.pyplot as plt
13 from sklearn.impute import SimpleImputer
14 from sklearn.preprocessing import StandardScaler
15 from sklearn.neighbors import KNeighborsRegressor
16 from scipy.stats import pearsonr
17 import seaborn as sns
18
19 # Load the training and validation datasets
20 df = pd.read_csv("aluminum_val_dataset.csv")
21 # Check for missing values
22 missing = df.isnull().sum()
23 print(missing)
24 # Handle missing values
25 imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
26 X = df.iloc[:, :-1]
27 X = imputer.fit_transform(X)
28 # Plot the correlation matrix between features in the training dataset
29 sns.heatmap(df.corr(), annot=True)
30 plt.show()
31
32 # Calculate the correlation coefficient for each feature-target pair
33 correlations = {}
34 for feature in df.columns[:-1]:
35     corr, _ = pearsonr(df[feature], df["Crack_depth"])
36     correlations[feature] = corr
37
38 # Print the correlation coefficients
39 print(correlations)
40
41 # Create a scatter matrix of the data
42 for feature in df.columns[:-1]:
43     corr, _ = pearsonr(df[feature], df["Crack_depth"])
44     plt.figure()
45     plt.scatter(df[feature], df["Crack_depth"])
46     plt.title(f'{feature} vs Crack_depth\n r={corr:.2f}')
47     plt.xlabel(feature)
48     plt.ylabel("Crack_depth")
49     plt.show()
50 from scipy import stats
51
52 # assuming df is your DataFrame and it has columns: feature_1, feature_2, feature_3, feature_4, and crack_depth_pred
53
54 # Calculate Spearman's correlation coefficient for each feature
55 for feature in df.columns[:-1]:
56     spearman_corr, p_value = stats.spearmanr(df[feature], df["Crack_depth"])
57     print(f'Spearman's correlation coefficient between {feature} and predicted crack depth: {spearman_corr}')
58 # Prepare the data
59 X = df.iloc[:, :-1].values
60 y = df.iloc[:, -1].values
61 X = pd.DataFrame(X, columns=df.iloc[:, :-1].columns)
62 scaler = StandardScaler()
63 X = scaler.fit_transform(X)
64
65 # Split the data into training and test sets
66 X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7)
67 # Define the list of algorithms
68 algorithms = [KNeighborsRegressor(n_neighbors=5, weights='distance', metric='manhattan'), Lasso(alpha=0.01), Ridge(alpha=0.01)]
69
70 # Train and evaluate each algorithm
71 mse_scores = []
72 algorithm_names = []
73 for algorithm in algorithms:
74     algorithm_names.append(algorithm.__class__.__name__)
75     # Compute the cross validation scores
76     cv_scores = cross_val_score(algorithm, X, y, cv=5, scoring='neg_mean_squared_error')
77     # Compute the mean and standard deviation of the scores
78     mean_score = -np.mean(cv_scores)
79     # Training
80     algorithm.fit(X_train, y_train)
81     print(f'{algorithm.__class__.__name__} - Training MSE score: {mean_score:.3f}')
82     # Prediction
83     y_pred = algorithm.predict(X_test)
84     mse = mean_squared_error(y_test, y_pred)
85     print(f'MSE of {algorithm.__class__.__name__} Validation MSE score: {mean_squared_error(y_test, y_pred)}')
86     mse_scores.append(mse)
87     # Compute the MSE of the baseline model
88     y_mean = y_train.mean()
89     y_pred_baseline = [y_mean] * len(y_test)
90     mse_baseline = mean_squared_error(y_test, y_pred_baseline)
91     print(f'MSE of baseline: {mse_baseline}')
92     # Compare the MSE of the model to the baseline
93     if mse < mse_baseline:
94         print("Model outperforms baseline")
95     else:
96         print("Baseline outperforms model")
97     # Plot the predicted values and the actual values on the test set
98     fig, ax = plt.subplots()
99     ax.plot(np.arange(len(y_test)), y_test)
100    ax.plot(np.arange(len(y_pred)), y_pred)
101    ax.set_xlabel('Actual Crack Depth mm')
102    ax.set_ylabel('Predicted Crack Depth mm')
103    ax.set_title(f'{algorithm.__class__.__name__}')
104    ax.legend()
105    plt.show()
106
107 # Plot the MSE scores for each algorithm
108 sns.barplot(x=algorithm_names, y=mse_scores)
109 plt.title('Mean Squared Error (MSE) for each algorithm')
110 plt.xlabel('Algorithm')
111 plt.ylabel('MSE score')
112 plt.show()

```

Figure_Apx B-3 Code of Aluminum Model with all Features

B.3 Code of Concrete machine learning model

```
1 # Import necessary Libraries
2 import pandas as pd
3 import numpy as np
4 from sklearn.model_selection import train_test_split, ShuffleSplit, cross_val_score
5 from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
6 from sklearn.svm import SVR
7 from sklearn.metrics import mean_squared_error, r2_score
8 import matplotlib.pyplot as plt
9 from sklearn.impute import SimpleImputer
10 from sklearn.preprocessing import StandardScaler
11 from scipy.stats import pearsonr
12 import seaborn as sns
13 # Load the training and validation datasets
14 df = pd.read_csv("Concrete_FConcrete_Val_dataset.csv")
15
16 # check for missing values
17 missing = df.isnull().sum()
18 print(missing)
19
20 # handle missing values
21 imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
22 X = df.iloc[:, :-1]
23 X = imputer.fit_transform(X)
24 # Plot the correlation matrix between features in the training dataset
25 sns.heatmap(df.corr(), annot=True)
26 plt.show()
27
28 # Calculate the correlation coefficient for each feature-target pair
29 correlations = {}
30 for feature in df.columns[:-1]:
31     corr, _ = pearsonr(df[feature], df["Crack_depth"])
32     correlations[feature] = corr
33
34 # Print the correlation coefficients
35 print(correlations)
36
37 # Create a scatter matrix of the data
38 for feature in df.columns[:-1]:
39     corr, _ = pearsonr(df[feature], df["Crack_depth"])
40     plt.figure()
41     plt.scatter(df[feature], df["Crack_depth"])
42     plt.title(f'{feature} vs Crack_depth\n r={corr:.2f}')
43     plt.xlabel(feature)
44     plt.ylabel("Crack_depth")
45     plt.show()
46 from scipy import stats
47
48 # assuming df is your DataFrame and it has columns: feature_1, feature_2, feature_3, feature_4, and crack_depth_pred
49
50 # Calculate Spearman's correlation coefficient for each feature
51 for feature in df.columns[:-1]:
52     spearman_corr, p_value = stats.spearmanr(df[feature], df["Crack_depth"])
53     print(f'Spearman's correlation coefficient between {feature} and predicted crack depth: {spearman_corr}')
54 # Prepare the data
55 X = df.iloc[:, :-1].values
56 y = df.iloc[:, -1].values
57 X = pd.DataFrame(X, columns=df.iloc[:, :-1].columns)
58 scaler = StandardScaler()
59 X = scaler.fit_transform(X)
60
61 # Shuffle the data and define the cross-validation iterator
62 cv = ShuffleSplit(n_splits=6, test_size=0.3)
63
64 # Define the list of algorithms
65 algorithms = [RandomForestRegressor(), SVR(), GradientBoostingRegressor()]
66
67 # Train and evaluate each algorithm using cross-validation
68 for algorithm in algorithms:
69     # Compute the cross-validation scores
70     cv_scores = cross_val_score(algorithm, X, y, cv=cv, scoring='neg_mean_squared_error')
71     # Compute the mean and standard deviation of the scores
72     mean_score = -np.mean(cv_scores)
73     # Fit the model on the full data
74     algorithm.fit(X, y)
75     print(f'{algorithm.__class__.__name__} - Training MSE: {mean_score:.3f}')
76     # Evaluate the model on the testing data
77     y_pred = algorithm.predict(X)
78     mse = mean_squared_error(y, y_pred)
79     print(f'{algorithm.__class__.__name__} - Test MSE: {mse:.3f}')
80     print(f'{algorithm.__class__.__name__} hyperparameters:')
81     print(algorithm.get_params())
82     # Compute the MSE of the baseline model
83     y_mean = y.mean()
84     y_pred_baseline = [y_mean] * len(y)
85     mse_baseline = mean_squared_error(y, y_pred_baseline)
86     print(f'MSE of baseline: {mse_baseline:.3f}')
87     # Compare the MSE of the model to the baseline
88     if mse < mse_baseline:
89         print("Model outperforms baseline")
90     else:
91         print("Baseline outperforms model")
92     # Plot the predicted values and the actual values on the full data
93     fig, ax = plt.subplots()
94     ax.plot(np.arange(len(y)), y)
95     ax.plot(np.arange(len(y_pred)), y_pred)
96     ax.set_xlabel('Actual Crack Depth mm')
97     ax.set_ylabel('Predicted Crack Depth mm')
98     ax.set_title(f'{algorithm.__class__.__name__}')
99     ax.legend()
100     plt.show()
```

Figure_Apx B-4 Code of Concrete Model with all Features

```

1  # Prepare the data
2  X = df[['NF']]
3  y = df.iloc[:, -1]
4
5  # Shuffle the data and define the cross-validation iterator
6  cv = ShuffleSplit(n_splits=6, test_size=0.3)
7
8  # Define the list of algorithms
9  algorithms = [RandomForestRegressor(), SVR(), GradientBoostingRegressor()]
10
11 # Train and evaluate each algorithm using cross-validation
12 for algorithm in algorithms:
13     # Compute the cross-validation scores
14     cv_scores = cross_val_score(algorithm, X, y, cv=cv, scoring='neg_mean_squared_error')
15     # Compute the mean and standard deviation of the scores
16     mean_score = -np.mean(cv_scores)
17     # Fit the model on the full data
18     algorithm.fit(X, y)
19     print(f"{algorithm.__class__.__name__} - Training MSE: {mean_score:.3f}")
20     # Evaluate the model on the testing data
21     y_pred = algorithm.predict(X)
22     mse = mean_squared_error(y, y_pred)
23     print(f"{algorithm.__class__.__name__} - Test MSE: {mse:.3f}")
24     print(f"{algorithm.__class__.__name__} hyperparameters:")
25     print(algorithm.get_params())
26     # Compute the MSE of the baseline model
27     y_mean = y.mean()
28     y_pred_baseline = [y_mean] * len(y)
29     mse_baseline = mean_squared_error(y, y_pred_baseline)
30     print(f"MSE of baseline: {mse_baseline:.3f}")
31     # Compare the MSE of the model to the baseline
32     if mse < mse_baseline:
33         print("Model outperforms baseline")
34     else:
35         print("Baseline outperforms model")
36     # Plot the predicted values and the actual values on the full data
37     fig, ax = plt.subplots()
38     ax.plot(np.arange(len(y)), y)
39     ax.plot(np.arange(len(y_pred)), y_pred)
40     ax.set_xlabel('Actual Crack Depth mm')
41     ax.set_ylabel('Predicted Crack Depth mm')
42     ax.set_title(f'{algorithm.__class__.__name__}')
43     ax.legend()
44     plt.show()
45
46

```

Figure_Apx B-5 Code of Aluminum Model with Selected Features

B.4 Code of automated machine learning model

```
1 # Import required Libraries
2 import h2o
3 from h2o.automl import H2OAutoML
4 import os
5 import tkinter as tk
6 from tkinter import filedialog, Tk
7 import matplotlib.pyplot as plt
8 import seaborn as sns
9 # Initialize H2O cluster
10 h2o.init()
11 # Create root window for file dialog
12 # Create visible root window for file dialog
13 root = tk.Tk()
14 root.attributes("-topmost", True)
15 root.withdraw()
16
17 # Get file name using file dialog
18 file_name = filedialog.askopenfilename(initialdir=r"C:\Users\s318075\Code",
19                                       title="Open file",
20                                       filetypes=(("CSV files", "*.csv"),
21                                                  ("all files", "*.*")))
22 # Import data from file
23 h2o.init()
24 data = h2o.import_file(file_name)
25 # Assuming data is already loaded
26 splits = data.split_frame(ratios=[0.7], seed=1)
27 train = splits[0]
28 test = splits[1]
29 # Define target column
30 target = train.columns[-1]
31 print("Target column:", target)
32 # Perform AutoML
33 aml = H2OAutoML(max_models=3,
34                 seed=1,
35                 stopping_metric='AUTO',
36                 sort_metric='AUTO')
37 aml.train(y=target, training_frame=train)
38 # Get Leaderboard
39 leaderboard = aml.leaderboard.as_data_frame()
40 # Plot actual vs predicted for each model
41 actual = test[target].as_data_frame().values
42 for model in aml.leaderboard['model_id'].as_data_frame().values:
43     mdl = h2o.get_model(model[0])
44     predictions = mdl.predict(test).as_data_frame().values
45     # Sort values for curve line
46     sorted_indices = actual.argsort(axis=0)
47     sorted_actual = actual[sorted_indices].flatten()
48     sorted_predictions = predictions[sorted_indices].flatten()
49     plt.figure(figsize=(10, 6))
50     plt.plot(sorted_actual, sorted_predictions, label="Predicted", color='blue')
51     plt.plot(sorted_actual, sorted_actual, color='red', linestyle="--", label="Actual")
52     plt.title(f"Actual vs Predicted for {model[0]}")
53     plt.xlabel("Actual")
54     plt.ylabel("Predicted")
55     plt.legend()
56     plt.show()
57 # Get Leaderboard
58 leaderboard = aml.leaderboard.as_data_frame()
59 lb = aml.leaderboard
60 lb.head()
61 # Retrieve feature importance
62 importance = aml.varimp(use_pandas=True)
63 # Print feature importance
64 print(importance)
65 # Get model ids for all models in the AutoML Leaderboard
66 model_ids = list(aml.leaderboard['model_id'].as_data_frame().iloc[:,0])
67 model_ids
68 # Get the best metric
69 best_metric = leaderboard.sort_values(by='rmse', ascending=False)['model_id'].iloc[0]
70
71 # Plot performance of each model
72 plt.barh(leaderboard['model_id'], leaderboard['rmse'])
73 plt.title('Model Performance')
74 plt.xlabel('rmse')
75 plt.ylabel('Model ID')
76 plt.show()
77 # Get the best model
78 best_model = h2o.get_model(best_metric)
79 # Evaluate model performance
80 perf = best_model.model_performance(data)
81 # Print evaluation metrics
82 print("rmse:", perf.rmse())
83 print(best_model.params)
84 # Deploy model for prediction
85 model_path = h2o.save_model(model=best_model, path="./model", force=True)
86 loaded_model = h2o.load_model(model_path)
87 prediction = loaded_model.predict(data)
```

Figure_Apx B-6 Code of Automated Machine learning

Appendix C Collected data of ABS to train and validate selected models.

Crack Location	Temp	NF	AMP	Crack depth
5	50	25.35	33.64	0.46
5	50	24.9	22.49	0.68
5	50	23.94	21.9	0.81
5	50	21.94	17.16	1.47
5	50	16.53	22.02	2
5	50	12.48	22.63	2.46
5	50	25.92	34.47	0.32
5	50	25.88	34.15	0.96
5	50	24.5	27.85	1.28
5	50	22.47	22.82	1.63
5	50	17.52	20.03	2.03
5	50	26.61	34.95	0.33
5	50	24.18	26.84	1.65
5	50	13.5	13.28	2.4
5	50	19.15	18.58	2.34
5	50	12.5	10.82	2.51
5	60	25.31	37.33	0.32
5	60	24.64	30.04	0.69
5	60	20.39	33.75	1.7

5	60	19.53	28.97	2.06
5	60	12.2	9.43	2.4
5	60	25.65	33.96	0.33
5	60	25.72	30.33	1.04
5	60	23.37	26.63	1.57
5	60	21.11	21	1.92
5	60	14.09	13.91	2.37
5	60	12.04	8.84	2.54
5	70	24.85	30.86	0.4
5	70	20.39	17.75	1.87
5	70	17.2	17.44	2.3
5	70	10.94	8.98	2.68
5	70	23.75	29.59	1.61
10	50	25.33	36.64	0.32
10	50	24.98	36.49	0.71
10	50	24.37	34.95	1.08
10	50	21.45	32.41	1.8
10	50	18.03	22.88	2.12
10	50	15.97	27.68	2.52
10	50	9.61	8.14	2.69
10	50	22.86	26.4	1.53
10	60	25.18	35.1	0.29

10	60	24.82	33.99	0.71
10	60	23.8	34.83	1.19
10	60	22.77	26.92	1.61
10	60	19.3	20.01	2
10	60	17.43	23.09	2.28
10	60	10.35	7.53	2.58
10	70	24.37	31.17	0.41
10	70	24.14	26.95	1.04
10	70	21.08	28.16	1.63
10	70	15.85	12.43	2.14
10	70	12.83	6.79	2.5
15	50	25.72	37.02	0.32
15	50	25.68	25.71	0.71
15	50	22.51	34.65	1.38
15	50	17.81	24.73	2.2
15	50	16.22	7.87	2.45
15	60	25.2	40.63	0.32
15	60	24.27	30.1	1.23
15	60	21.89	33.14	1.59
15	60	20.48	23.45	2.17
15	60	14.31	11.29	2.33
15	60	11.43	9.06	2.61

15	70	25.55	31.3	0.34
15	70	23.41	32.12	1.38
15	70	19.87	29.44	2.1
15	70	14.2	6.96	2.6
25	50	26.47	32.78	0.51
25	50	25.93	30.56	0.84
25	50	21.82	20.5	1.74
25	50	15.32	17.32	2.29
25	50	7.79	7.76	2.63
25	60	25.79	28.41	0.24
25	60	25.63	29.48	0.63
25	60	23.45	30.6	0.83
25	60	19.64	22.08	1.46
25	60	17.97	15.68	2.36
25	60	9.28	8.03	2.66
25	70	25.97	31.21	0.26
25	70	25.51	32.65	0.93
25	70	24.02	27.96	1.39
25	70	22.24	25.55	1.78
25	70	20.84	28.93	1.99
25	70	19.81	21.62	2.17
25	70	13.42	8.48	2.57

5	50	23.27	4.49	0.46
5	50	20.39	3.82	1.13
5	50	25.32	2.94	0.67
5	50	25.27	3.39	0.85
5	50	20.51	4.62	1.69
5	50	24.71	1.94	0.29
5	50	23.98	3.95	1.17
5	50	18.73	2.37	2.02
5	50	23.68	4.61	0.88
5	50	22.21	3.35	1.01
5	50	16.03	2.02	2.23
5	50	22.67	3.96	0.77
5	50	21.85	4.1	0.84
5	50	17.98	2.06	1.35
5	50	24.22	3.95	0.61
5	50	19.72	3.71	1.6
5	50	17.2	2.97	2.25
5	50	23.89	4.28	0.4
5	50	23.06	4.41	0.84
5	50	21.28	0.89	1.6
5	50	25.11	3.98	0.72
5	50	23.27	4	2.08

5	50	20.91	2.89	2.55
5	50	25.18	3.67	0.37
5	50	24.37	3.48	1.66
5	50	17.68	2.37	2.22
5	50	25.2	3.59	0.57
5	50	22.98	3.28	1.41
5	50	16.24	2.01	2.21
5	50	24.35	3.49	0.27
5	50	24.55	2.47	1.03
5	50	23.13	3.2	1.89
5	50	25.33	3.73	0.66
5	50	23.76	3.64	1.49
5	50	19.79	2.99	2.28
5	50	24.55	3.27	0.63
5	50	22.34	3.32	1.67
5	50	13.71	3.29	2.28
5	50	24.62	3.44	0.44
5	50	23.06	3.33	1.23
5	50	19.16	3.52	1.9
5	50	24.51	3.7	0.66
5	50	21.68	3.52	1.58
5	50	19.08	3.05	1.91

5	50	24.04	4.12	0.48
5	50	22.19	4.13	1.48
5	50	19.48	4.17	2.21
5	50	24.38	3.38	0.49
5	50	23.05	3.08	1.38
5	50	18.25	3.24	1.86
5	50	25.73	2.39	2.22
5	50	23.13	2.3	2.64
5	50	18.8	1.83	2.78
5	50	25.5	2.72	0.59
5	50	24.65	2.65	1.82
5	50	19.78	3.78	2.47
5	50	23.2	3.28	0.83
5	50	22.41	3.53	1.37
5	50	20.38	2.42	1.73
5	50	24.25	3.64	0.8
5	50	23.35	2.24	1.59
5	50	19.48	2.21	2.03
5	50	24.88	3.01	0.79
5	50	21.85	3.08	1.2
5	50	14.47	1.79	2.12
5	50	23.97	3.35	0.87

5	50	21.83	3.29	1.58
5	50	18.4	2.26	1.93
5	50	24.16	2.73	0.87
5	50	20.75	1.49	1.53
5	50	19.38	1.77	1.9
5	50	26.88	3.4	0.46
5	50	25.6	3.26	1.48
5	50	21.63	1.73	2.01
5	50	24.4	3.68	0.73
5	50	20.99	2.93	1.24
5	50	17.58	2.07	1.5
5	50	25.95	3.27	0.55
5	50	25.03	3.05	1.19
5	50	19.32	1.84	1.94
5	50	23.77	3.59	0.63
5	50	22.56	2.97	1.67
5	50	17.85	2.16	2.14
5	50	21.96	3.65	0.7
5	50	20.86	1.78	1.56
5	50	19.42	2.62	2.51
5	50	24.45	3.66	0.74
5	50	20.22	1.63	1.2

5	50	13.1	1.32	1.81
5	50	25.87	3.24	0.53
5	50	23.26	3.06	1.45
5	50	20.35	2.2	2.04
5	50	25.22	3.87	0.51
5	50	23.58	1.78	1.68
5	50	19.37	2.98	2.2
5	50	24.01	3.67	0.75
5	50	19.9	2.55	1.49
5	50	15.32	0.85	2.02
5	50	24.51	3.72	0.77
5	50	20.7	1.17	1.68
5	50	17.09	1.12	2.23
5	50	24.51	4.03	0.72
5	50	21.9	1.83	1.4
5	50	18.03	1.29	1.84
5	50	23.15	3.42	0.59
5	50	21.44	2.73	1.27
5	50	17.83	2.44	1.95
5	50	25.45	3.01	0.72
5	50	23.64	2.68	1.65
5	50	18.29	2.49	2.31

5	50	26.31	2.5	0.55
5	50	24.91	3.81	1.57
5	50	21.76	1.55	2.05
5	50	24.26	5.58	0.55
5	50	23.52	3.21	1.13
5	50	18.57	1.86	2.13
5	50	24.17	3.38	0.47
5	50	21.68	2.95	1.47
5	50	18.54	2.82	1.97
5	50	24.37	3.68	0.6
5	50	23.16	3.33	1.63
5	50	19.13	2.45	2.16
5	50	23.44	4.4	0.58
5	50	21.97	2.34	1.3
5	50	17.49	3.21	1.96
5	50	26.19	3.33	1.05
5	50	24.74	3.57	1.81
5	50	20.38	3.09	2.93
5	50	23.5	4.52	0.53
5	50	22.44	2.53	1.23
5	50	19.33	1.6	1.99
5	50	24.56	4.33	0.69

5	50	23.92	3.82	1.25
5	50	22.1	2.65	2.02
5	50	25.04	3.83	0.44
5	50	24.34	3.57	1.05
5	50	22.95	3.49	1.47
5	50	24.77	3.25	0.68
5	50	23.95	3.37	1.02
5	50	18.19	2.98	1.82
5	50	24.51	4.01	0.23
5	50	24.09	3.57	1.11
5	50	21.51	2.7	1.72
5	50	25.41	4.04	0.44
5	50	24.79	3.73	0.97
5	50	23.32	4.28	1.67
5	50	23.83	2.82	0.4
5	50	23.45	2.84	0.95
5	50	22.16	2.32	1.66
5	50	24.39	4.2	0.44
5	50	24.07	3.83	1.05
5	50	22.79	3.06	1.93
5	50	24.55	3.68	0.29
5	50	23.46	3.29	1.09

5	50	20.92	2.84	1.54
5	50	25.02	2.85	0.51
5	50	24.4	2.46	1.31
5	50	22.27	1.94	1.94
5	50	24.39	3.93	0.67
5	50	23.99	3.24	1.08
5	50	22.26	2.16	2.11
5	60	24.24	2.26	0.5
5	60	20.9	1.67	0.78
5	60	25.07	4.34	0.31
5	60	23.16	4.12	1.29
5	60	19.06	2.16	1.82
5	60	24.83	3.93	0.53
5	60	23.84	4.61	1.01
5	60	19.19	3.25	1.79
5	60	22.47	3.7	0.72
5	60	20.79	3.98	0.93
5	60	18.26	2.59	2.03
5	60	21.34	3.61	0.54
5	60	20.34	4.05	0.68
5	60	17.49	3.67	1.75
5	60	23.56	4.37	0.45

5	60	23.31	3.48	0.67
5	60	20.05	3.72	1.81
5	60	23.4	4.81	0.47
5	60	21.46	3.85	0.88
5	60	18.24	4.01	1.74
5	60	25.22	3.03	0.45
5	60	23.92	3.87	1.23
5	60	19.7	3.69	2.01
5	60	25.32	3.31	0.58
5	60	24.59	3.77	0.61
5	60	18.64	2.67	2.18
5	60	22.29	3.99	0.62
5	60	21.09	3.06	0.77
5	60	13.92	1.08	1.75
5	60	23.1	4.2	0.52
5	60	22.98	3.37	1.15
5	60	20.47	3.38	1.85
5	60	25	3.97	0.51
5	60	23.77	3.7	1.33
5	60	21.6	3.57	1.95
5	60	23.97	3.99	0.71
5	60	19.01	1.03	1.69

5	60	15.48	1.58	2.19
5	60	23.7	3.22	0.76
5	60	22.93	2.53	1.27
5	60	18.76	3.56	2.33
5	60	24.98	3.68	0.39
5	60	24	2.56	1.09
5	60	17.95	1.57	1.87
5	60	23.14	3.92	0.46
5	60	21.79	3.05	1.34
5	60	18.5	3.42	1.98
5	60	24.69	3.46	0.65
5	60	21.13	1.65	1.38
5	60	14.29	0.84	2.05
5	60	24.93	1.97	0.44
5	60	25.86	1.81	0.63
5	60	23.26	2.14	2.03
5	60	18.64	0.62	2.36
5	60	23.43	2.03	0.66
5	60	21.38	3.17	1.45
5	60	16.76	1.96	2.51
5	60	22.22	1.98	1.01
5	60	21.82	2.42	1.65

5	60	20.98	2.14	2.48
5	60	23.83	4.03	0.82
5	60	21.88	2.65	1.67
5	60	18.68	3.35	2.19
5	60	24.29	2.65	0.49
5	60	20.04	0.92	1.94
5	60	15.14	0.86	2.55
5	60	23.47	2.64	0.83
5	60	21.23	2.11	1.46
5	60	12.02	0.29	1.96
5	60	24.84	2.43	0.52
5	60	21.24	1.88	1.37
5	60	17.32	0.56	1.89
5	60	27.11	3	0.67
5	60	25.78	2.78	1.44
5	60	21.77	1.4	1.99
5	60	24.45	2.78	0.49
5	60	21.38	1.78	0.9
5	60	19.29	1.17	2.13
5	60	25	3.46	0.43
5	60	22.16	1.74	1.83
5	60	18.41	1.57	2.17

5	60	22.67	3.57	0.52
5	60	18.36	1	1.91
5	60	16.28	2.3	2.22
5	60	24.77	2.96	0.76
5	60	21.92	2.05	1.65
5	60	17.29	1.22	2.32
5	60	23.81	3.11	0.79
5	60	20.32	1.72	1.32
5	60	16.9	1.93	2.05
5	60	17.37	3.16	0.51
5	60	24.99	1.42	2.03
5	60	14.02	0.76	2.43
5	60	24.67	3.34	0.66
5	60	22.99	2.44	1.72
5	60	17.03	0.79	2.37
5	60	24.37	3.26	0.95
5	60	22.04	1.77	1.41
5	60	24.12	3.34	0.67
5	60	20.31	2.02	1.34
5	60	17.11	2.62	2.04
5	60	23.26	3.61	0.78
5	60	24.39	3.92	0.62

5	60	20.92	1.77	1.39
5	60	13.81	0.63	2.22
5	60	22.97	4.26	0.5
5	60	18.49	1.65	1.91
5	60	15.87	0.3	2.3
5	60	25.9	2.61	0.48
5	60	24.33	3.09	1.31
5	60	20.2	2.79	2.07
5	60	23.6	4.16	0.82
5	60	21.51	1.95	1.7
5	60	17.86	2.89	2.29
5	60	25.28	3.9	0.62
5	60	24.29	3.93	1.13
5	60	20.02	3.97	1.91
5	60	24.73	3.27	0.62
5	60	23.79	2.8	1.02
5	60	21.94	0.47	2.03
5	60	24.34	3.75	0.52
5	60	23.97	3.36	1.35
5	60	21.79	1.11	2.19
5	60	24.77	4.09	0.74
5	60	24.03	3.37	1.53

5	60	21.43	3.47	2.14
5	60	23.89	3.89	0.48
5	60	22.12	3.27	1.14
5	60	19.8	2.2	1.88
5	60	24.38	3.83	0.5
5	60	24.11	3.5	1.06
5	60	22.7	2.91	1.75
5	60	25.13	3.9	0.48
5	60	24.01	3.59	1.48
5	60	20.49	0.98	2.06
5	60	21.94	2.2	0.42
5	60	21.21	2.6	1
5	60	19.65	2.27	1.79
5	60	22.49	3.35	0.46
5	60	21.97	2.73	1.15
5	60	21.16	2.53	2.02
5	60	24.35	3.51	0.49
5	60	23.26	2.57	1.48
5	60	21.33	2.72	1.9
5	60	24.78	3.11	0.62
5	60	24.02	2.37	1.01
5	60	22.68	1.72	1.78

5	60	24.23	2.69	0.59
5	60	23.54	2.16	0.93
5	60	21.02	1.78	1.79
5	70	22.96	25.24	0.41
5	70	20.41	27.83	0.78
5	70	16.6	16.45	1.69
5	70	24.03	23.17	0.57
5	70	23.44	20.46	0.86
5	70	21.88	26.08	1.62
5	70	21.74	25.48	0.46
5	70	20.24	27.99	0.97
5	70	18.6	18.65	1.76
5	70	21.89	28.05	0.54
5	70	22.47	26.02	0.67
5	70	21.69	22.89	1.47
5	70	21.71	25.32	0.78
5	70	17.35	26.36	0.93
5	70	21	23.05	0.76
5	70	18.54	27.42	1.38
5	70	23.33	8.53	0.42
5	70	23.48	20.33	0.64
5	70	22.56	3.15	0.63

5	70	17.29	2.81	1.2
5	70	23.96	2.83	0.46
5	70	22.37	3.04	1.37
5	70	17.93	2.06	2.13
5	70	24.6	3.77	0.31
5	70	24.21	3.27	1.11
5	70	20.6	0.64	1.76
5	70	23.22	2.6	0.58
5	70	20.2	3.14	1.18
5	70	16.64	2.15	1.99
5	70	23.89	2.29	0.52
5	70	23.58	2.66	1.15
5	70	15.93	2.77	2.19
5	70	22.66	1.69	0.53
5	70	23.6	3.07	1.41
5	70	21.74	3.75	2.28
5	70	21.39	2.58	0.55
5	70	20.91	3.28	1.33
5	70	9.68	0.29	2.22
5	70	24.05	2.56	0.86
5	70	21.19	3.3	1.49
5	70	17.75	3.39	1.77

5	70	24.63	1.7	0.73
5	70	25.58	1.14	1.01
5	70	21.63	2.19	1.99
5	70	23.47	1.7	0.92
5	70	24.2	1.74	1.05
5	70	23.94	1.48	1.37
5	70	20.2	2.36	2.44
5	70	21.9	2.21	0.53
5	70	21.01	1.66	0.92
5	70	14.54	1.77	1.7
5	70	24.12	1.88	1.16
5	70	21.76	2.89	1.53
5	70	16.91	2.63	1.83
5	70	22.67	2.89	0.67
5	70	19.83	1.29	1.7
5	70	18.86	2.6	1.94
5	70	24.47	1.73	0.98
5	70	20.49	1.55	1.72
5	70	11.96	0.29	2.23
5	70	24.4	1.64	1.06
5	70	21.92	2.32	1.46
5	70	17.54	0.43	1.94

5	70	22.96	2.75	0.84
5	70	18.68	0.97	1.75
5	70	13.44	1.93	2.21
5	70	25.72	2.78	0.63
5	70	24.14	2.99	1.7
5	70	20.22	1.56	2.16
5	70	23.19	1.99	0.92
5	70	21.79	2.21	1.24
5	70	18.49	1.96	1.65
5	70	25.66	2.17	0.55
5	70	23.91	1.95	1.52
5	70	19.4	1.75	2.02
5	70	23.14	2.65	0.72
5	70	20.81	2.19	1.68
5	70	17.94	3.12	2.12
5	70	23.62	2.96	0.84
5	70	20.21	1.92	1.55
5	70	14.95	0.94	2.56
5	70	25.49	2.42	0.78
5	70	21.83	2.01	1.8
5	70	17.47	0.43	2.39
5	70	25.23	2.67	0.61

5	70	22.43	1.41	1.67
5	70	17.18	0.53	2.33
5	70	23.5	3.19	0.64
5	70	20.22	1.69	1.27
5	70	15.37	0.8	2.04
5	70	23.66	3.47	0.58
5	70	21.51	2.2	1.46
5	70	18.01	0.51	2.11
5	70	24.56	2.18	0.5
5	70	22.39	2.11	1.46
5	70	19.27	0.65	2.11
5	70	22.24	3.41	0.46
5	70	22.77	2.47	0.8
5	70	16.84	1.15	1.79
5	70	24.81	2.55	0.51
5	70	21.22	3.46	1.83
5	70	18.72	3.41	2.12
5	70	24.21	2.31	0.59
5	70	23.26	2.78	1.2
5	70	21.4	1.88	2.19
5	70	24.69	2.93	0.62
5	70	23.87	4.1	1.18

5	70	22.38	4.24	1.79
5	70	25.86	2.19	0.56
5	70	24.52	3.19	1
5	70	20.48	1.11	2.02
5	70	24.55	2.72	0.69
5	70	23.2	2.83	1.15
5	70	21.6	1.38	2.07
5	70	24.66	2.72	0.39
5	70	23.35	3	1.45
5	70	21.09	3.6	1.77
5	70	22.12	2.96	0.57
5	70	21.53	3	0.83
5	70	20.03	2.44	1.79
5	70	23.01	2.76	0.45
5	70	23.61	3.18	0.83
5	70	21.01	2.38	1.67
5	70	24.48	2.84	0.33
5	70	23.44	2.03	1.24
5	70	21.44	2.11	1.6
5	70	24.89	2.22	0.59
5	70	23.82	2.13	0.99
5	70	22.38	1.75	1.89

5	70	25.57	2.19	0.64
5	70	23.98	1.57	1.13
5	70	21.25	1.87	1.85
5	70	23.58	3	0.85
5	70	22.3	2.23	1.61
5	70	19.96	1.27	2.19
5	20	19.02	20.16	1.55
5	20	19.36	28.18	1.57
5	20	19.33	25.05	1.39
5	20	19.9	34.26	1.14
5	20	20.02	32.11	0.79
5	20	18.82	29.49	1.64
5	20	18.46	35.73	1.62
5	20	18.49	32.22	1.36
5	20	19.54	32.24	1.08
5	20	19.73	31.95	0.81
5	20	18.44	29.64	1.59
5	20	18.53	32.09	1.5
5	20	18.59	33.13	1.36
5	20	19.83	31.01	1.13
5	20	19.84	34.55	0.77
5	20	19.53	27.63	1.24

5	20	19.59	34.4	1.19
5	20	19.51	39.71	1.15
5	20	19.48	40.71	0.89
5	20	20.56	31.7	0.63
5	20	19.7	36.83	1.24
5	20	19.63	35.29	0.99
5	20	19.97	34.51	0.86
5	20	20.11	39.47	0.71
5	20	20.24	40.11	0.49
5	20	19.79	35.8	1.07
5	20	20.3	31.88	0.95
5	20	19.71	34	0.88
5	20	19.7	37.87	0.71
5	20	19.93	35.78	0.52
5	50	23.27	4.49	0.46
5	50	20.39	3.82	1.13
5	50	25.32	2.94	0.67
5	50	25.27	3.39	0.85
5	50	20.51	4.62	1.69
5	50	24.71	1.94	0.29
5	50	23.98	3.95	1.17
5	50	18.73	2.37	2.02

5	50	23.68	4.61	0.88
5	50	22.21	3.35	1.01
5	50	16.03	2.02	2.23
5	50	22.67	3.96	0.77
5	50	21.85	4.1	0.84
5	50	17.98	2.06	1.35
5	50	24.22	3.95	0.61
5	50	19.72	3.71	1.6
5	50	17.2	2.97	2.25
5	50	23.89	4.28	0.4
5	50	23.06	4.41	0.84
10	60	25.18	35.1	0.29
10	60	24.82	33.99	0.71
10	60	23.8	34.83	1.19
10	60	22.77	26.92	1.61
10	60	19.3	20.01	2
10	60	17.43	23.09	2.28
10	60	10.35	7.53	2.58
15	70	23.41	32.12	1.38
15	70	19.87	29.44	2.1
15	70	14.2	6.96	2.6
25	50	26.47	32.78	0.51

25	50	25.93	30.56	0.84
25	50	21.82	20.5	1.74
25	50	15.32	17.32	2.29
25	50	7.79	7.76	2.63
5	70	21.49	1.91	1.66
5	70	16.33	0.51	2.2
5	70	23.27	2.02	1.05
5	70	19.57	1.79	1.35
5	70	16.56	1.71	2.29
5	70	22.75	2.84	0.71
5	70	20.92	2	1.41
5	70	17.82	2.3	1.92
5	70	25.35	2.25	0.7
5	70	23.25	2.6	1.67
5	70	16.46	0.45	2.18
5	70	24.39	2.96	0.55
5	70	23.42	2.7	1.82
5	70	16.9	1.07	2.31
5	70	23.49	5.66	0.47
5	70	18.69	1.4	1.62
5	70	13.15	2.53	2.32
5	70	24.63	3.54	0.51

5	70	22.67	3.25	1.45
5	70	19.38	1.72	2
5	70	22.04	2.81	0.5
5	60	18.82	3.07	1.62
5	60	11.67	0.71	2.26
5	60	24.61	3.03	0.56
5	60	21.86	2.18	1.47
5	60	17.79	2.4	1.99
5	60	26.05	2.54	0.65
5	60	24.1	2.92	1.64
5	60	18.54	3.62	2.3
5	60	26.23	2.53	0.66
5	60	25.11	2.75	1.37
5	60	17.42	1.29	2.23
5	70	22.47	3.44	1.62
5	70	23.31	2.81	0.52
5	70	22.34	2.76	0.83

Appendix D Collected data of Aluminum to train and validate selected Models.

Crack location	Temp	NF	AMP	Crack depth
4.5	25	57.24	4.44	0.5
4.5	25	57.22	4.43	0.5
4.5	25	57.04	3.72	1
4.5	25	57.03	3.73	1
4.5	25	57.54	4.91	0.25
4.5	25	57.57	4.93	0.25
4.5	25	57.25	4.45	0.5
4.5	25	57.26	4.44	0.5
4.5	25	57.08	3.79	1
4.5	25	57.09	3.8	1
4.5	50	57.93	5.01666 7	0
4.5	50	57.94333	5.02	0
4.5	50	57.32	4.74333 3	0.25
4.5	50	57.35333	4.70666 7	0.25
4.5	50	57.13	4.25666 7	0.5
4.5	50	57.12	4.25	0.5

Crack location	Temp	NF	AMP	Crack depth
4.5	50	57.14	4.26	0.5
4.5	50	56.75	3.55	1
4.5	50	56.74	3.54	1
4.5	50	57.35	4.81	0.25
4.5	50	57.39	4.9	0.25
4.5	50	57.15	4.32	0.5
4.5	50	57.17	4.31	0.5
4.5	50	57.18	4.4	0.5
4.5	50	56.83	3.68	1
4.5	50	56.84	3.67	1
4.5	50	56.92	3.69	1
4.5	100	57.33333	4.28	0
4.5	100	57.41333	4.38333 3	0
4.5	100	56.95333	4.26333 3	0.25
4.5	100	56.96	4.28666 7	0.25
4.5	100	56.85333	4.11666 7	0.5
4.5	100	56.85	4.12	0.5
4.5	100	56.86	3.36	0.5

Crack location	Temp	NF	AMP	Crack depth
4.5	100	56.22	3.18	1
4.5	100	56.21	3.17	1
4.5	100	57.11	4.52	0.25
4.5	100	57.13	4.43	0.25
4.5	100	56.89	4.13	0.5
4.5	100	56.97	4.27	0.5
4.5	100	56.98	4.28	0.5
4.5	150	56.5	3.45	0.25
4.5	150	56.51	3.7	0.25
4.5	150	56.29	3.36	0.5
4.5	150	56.28	3.34	0.5
4.5	150	55.52	2.91	1
4.5	150	55.53	2.92	1
4.5	150	56.78	3.89	0.25
4.5	150	56.73	3.87	0.25
4.5	150	56.42	3.61	0.5
4.5	150	56.44	3.64	0.5
4.5	150	56.61	3.87	0.5
4.5	150	56.62	3.86	0.5
4.5	150	55.74	3.21	1
4.5	150	55.73	3.22	1

Crack location	Temp	NF	AMP	Crack depth
4.5	150	55.92	3.38	1
4.5	200	56.82333	3.32	0
4.5	200	56.87667	3.37333 3	0
4.5	200	56.24333	3.41666 7	0.25
4.5	200	56.35667	3.52333 3	0.25
4.5	200	55.75	3.056	0.5
4.5	200	56.26	3.23	0.25
4.5	200	56.23	3.51	0.25
4.5	200	55.75	3	0.5
4.5	200	55.76	3.09	0.5
4.5	200	55.12	2.65	1
4.5	200	55.14	2.66	1
4.5	200	56.57	3.71	0.25
4.5	200	56.69	3.78	0.25
4.5	200	55.97	3.23	0.5
4.5	200	55.99	3.21	0.5
4.5	200	56.11	3.53	0.5
4.5	200	55.36	2.91	1
4.5	200	55.37	2.93	1

Crack location	Temp	NF	AMP	Crack depth
4.5	200	55.56	3.12	1
5	20	37.88	100.48	0.5
5	20	38.06	92.38	0.5
5	20	38.95	94.95	0.5
5	20	14.99	255.38	1.89
5	20	14.04	109.04	1.93
5	20	15.01	207.7	1.44
5	20	11.95	164.55	1.86
5	20	36.87	102.52	1
5	20	37.88	101.61	1
5	20	9.97	77.04	2.52
5	20	11.95	77.21	2.66
5	20	36.06	106.53	1.5
5	20	33.98	117.37	1.5
5	20	36.06	108.56	1.5
5	20	36.06	109.12	1.5
5	20	36.06	109.08	1.5
5	20	32.01	76.7	2.17
5	20	32.9	99.35	1.74
5	20	35.14	112.92	1.68
5	20	33.98	113.08	1.93

Crack location	Temp	NF	AMP	Crack depth
5	20	33.84	110.39	1.78
5	20	29.92	94.39	2.04
5	20	28.18	37.23	2.84
5	20	28.98	13.82	2.58
5	20	29.81	35.95	2.44
5	20	29.92	19.51	2.96
5	20	14.49	52.24	2.79
5	20	18.6	92.91	2.69
5	20	30.81	35.42	2.31
10	20	38.95	90.86	0.5
10	20	37.03	81.58	1
10	20	37.03	92.9	1
10	20	37.03	99.14	1
10	20	37.03	94.13	1
10	20	37.03	96.86	1
10	20	35.14	107	1.5
10	20	35.9	100.24	1.5
10	20	36.06	102.96	1.5
10	20	34.99	111.8	1.5
10	20	33.98	109.71	1.5
10	20	15.01	223.5	1.19

Crack location	Temp	NF	AMP	Crack depth
10	20	12.9	171.91	2.3
10	20	12.92	79.82	2.3
10	20	32.9	116.3	1.86
10	20	33.98	116.27	1.86
10	20	33.98	118.31	1.73
10	20	32.01	117.88	1.74
10	20	13.93	115.61	2.13
10	20	11.99	54.76	2.86
10	20	12.98	55.15	2.6
10	20	11.99	55.79	2.88
10	20	11.95	77.25	2.65
10	20	11.95	69.85	2.72
10	20	31.76	86.04	2.19
10	20	30.93	80.44	2.06
10	20	29.81	51.45	2.31
10	20	29.92	38.42	2.21
10	20	29.92	102.14	2.1
10	20	29.92	30.58	2.7
10	20	28.98	42.56	2.38
10	20	11.68	78.93	2.71
10	20	27.99	14.71	2.62

Crack location	Temp	NF	AMP	Crack depth
10	20	27.99	51.06	2.43
4.5	25	57.52	4.913	0.25
4.5	25	57.56	4.93	0.25
4.5	25	57.23	4.43	0.5
4.5	25	57.23	4.44	0.5
4.5	100	56.34	3.41	1
4.5	100	56.53	3.52	1
4.5	100	56.54	3.54	1
4.5	150	57	3.78	0
4.5	150	57.08	3.88	0
4.5	150	56.50	3.62	0.25
4.5	150	56.63	3.71	0.25
4.5	150	56.28	3.35	0.5
5	20	38.06	98.96	1
5	20	37.03	103.88	1
5	20	34.84	99.95	1.45
5	20	14.02	85.52	1.8
5	20	14.99	190.16	1.4
5	20	15.01	195.77	1.64
5	20	30.81	98.57	1.84
10	20	12.86	120.52	2.43

Crack location	Temp	NF	AMP	Crack depth
10	20	12.86	139.94	2.15
10	20	14	197.2	2.05
10	20	33.03	109.46	1.94

Appendix E Collected data of Concrete to train and validate selected Models.

NF	AMP	Temp	Crack depth
258.41	0.01287	20	0
210.7354	0.006944	20	0
268.0628	0.004695	20	0
286.0335	0.010165	20	0
255.6423	0.011804	20	2
245.2107	0.011357	20	2
244.9761	0.007652	20	2
248.6097	0.010271	20	2
206.4516	0.007643	20	2
247.5568	0.010471	20	4
234.1536	0.009463	20	6
244.2748	0.006431	20	6
213.5834	0.006671	20	6
235.5106	0.012479	20	6
237.037	0.012065	20	6
258.41	0.01287	20	0
255.6423	0.011804	20	2
247.5568	0.01	20	4
234.1536	0.009	20	6

NF	AMP	Temp	Crack depth
250.5377	0.014623	40	0
268.6254	0.005234	40	0
237.4768	0.013899	40	0
235.5106	0.006617	40	0
247.8544	0.01345	40	2
257.0281	0.009119	40	2
250.9804	0.005955	40	2
226.9504	0.018753	40	2
230.2158	0.00762	40	2
240.0151	0.011678	40	4
227.02	0.010446	40	6
241.7375	0.01467	40	6
207.7922	0.006253	40	6
253.2146	0.01522	60	0
245.6814	0.10384	60	0
291.24	0.006849	60	0
274.0899	0.007181	60	0
255.489	0.006661	60	2
229.3907	0.018143	60	2
231.8841	0.008586	60	2
236.5989	0.017772	60	2

NF	AMP	Temp	Crack depth
232.7273	0.008835	60	2
232.2288	0.010496	60	4
219.6553	0.009089	60	6
230.009	0.016896	60	6
229.8025	0.008012	60	6
242.41	0.015938	60	0
232.4837	0.013099	20	0
225.38	0.012525	20	2
216.7569	0.010414	20	4
210.7461	0.008893	20	6
247.5822	0.012349	20	0
249.2697	0.026458	20	0
189.6296	0.006394	20	2
204.6363	0.022207	20	2
206.7851	0.012536	20	2
229.1853	0.01001	20	2
177.9013	0.010894	20	2
232.939	0.011628	20	4
234.4322	0.00844	20	4
252.4655	0.01051	20	4
254.9801	0.007331	20	4

NF	AMP	Temp	Crack depth
258.5859	0.017374	20	6
255.7443	0.010782	20	6
237.9181	0.016251	20	6
246.866	0.009529	20	6
242.0425	0.015342	40	0
239.4501	0.015598	40	2
231.8767	0.01561	40	4
219.3224	0.010501	40	6
247.8219	0.016827	40	0
218.0579	0.023371	40	0
187.1345	0.012526	40	2
186.453	0.010598	40	2
202.3715	0.025229	40	2
197.9892	0.013809	40	2
214.4054	0.007617	40	2
232.5159	0.005812	40	2
207.7922	0.011901	40	4
231.4647	0.007061	40	4
197.3786	0.021478	40	4
161.0063	0.011128	40	4
223.3857	0.014644	40	6

NF	AMP	Temp	Crack depth
222.6895	0.010521	40	6
219.3659	0.009755	40	6
224.1681	0.000101	40	6
234.1903	0.015176	60	0
231.6821	0.014567	60	2
224.3543	0.013665	60	4
212.2071	0.013052	60	6
243.346	0.017259	60	0
241.2818	0.021095	60	0
188.5125	0.012929	60	2
186.3173	0.00784	60	2
204.4728	0.032218	60	2
199.2218	0.014655	60	2
197.2265	0.01045	60	4
229.3907	0.006793	60	4
158.9075	0.026825	60	4
159.8002	0.012068	60	4
218.43	0.009668	60	6
218.9906	0.007438	60	6
289.9207	0.006532	20	0
264.4628	0.0085	20	0

NF	AMP	Temp	Crack depth
268.0628	0.007338	20	0
243.1149	0.00762	20	2
251.9685	0.005607	20	2
205.9533	0.010435	20	2
251.7207	0.017444	40	0
252.9644	0.011411	40	0
272.051	0.01118	40	0
229.5964	0.008561	40	2
232.7273	0.012478	40	2
234.8624	0.015746	60	0
232.939	0.007081	60	0
239.8137	0.01347	60	2
257.2864	0.00786	60	2
235.9447	0.020474	60	6
203.6895	0.009215	60	6
268.9076	0.013474	20	0
268.3438	0.005918	20	0
246.866	0.008097	20	2
249.0272	0.0062	20	2
246.3908	0.007271	20	2
235.0781	0.007064	20	4

NF	AMP	Temp	Crack depth
235.7274	0.006743	20	4
242.654	0.013982	20	6
230.2158	0.007623	20	6
244.0419	0.014599	40	0
249.7561	0.007902	40	0
218.9906	0.009038	40	2
212.0961	0.009166	40	2
233.1512	0.008325	40	4
232.2941	0.007146	40	4
222.6087	0.013439	40	6
217.3175	0.007053	40	6
248.7852	0.014134	60	0
242.654	0.006462	60	0
207.7922	0.011143	60	2
206.7851	0.009263	60	2
227.758	0.009106	60	4
226.9504	0.007265	60	4
212.6246	0.010315	60	6
212.6246	0.006084	60	6
233.79	0.015835	40	6
231.0469	0.009892	40	6

Appendix F Collected ABS data to train and validate Automated model.

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	70	22.78243	33.11981	0.455
5	70	22.35746	36.19763	0.682
5	70	21.46691	33.57629	0.813
5	70	19.93939	18.30464	1.472
5	70	15.11809	18.11099	2.022
5	70	11.76824	16.42704	2.458
5	70	22.413	30.45114	0.402
5	60	22.83948	36.49431	0.32
5	60	22.27576	30.75316	0.693
5	60	18.74042	26.18383	1.704
5	60	17.94616	22.12791	2.064
5	60	11.38252	9.228056	2.402
5	50	23.39039	34.03542	0.317
5	50	23.25653	34.18937	0.963
5	50	21.88097	27.90206	1.279
5	50	19.87893	19.7793	1.632
5	50	15.98832	17.67307	2.026
5	50	12.96294	10.9044	2.396
15	70	22.17543	32.68593	0.412

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
15	70	21.70294	24.8782	1.04
15	70	19.13601	23.67163	1.627
15	70	14.53109	11.70494	2.14
15	70	12.59202	7.028587	2.497
15	60	22.71055	34.88892	0.29
15	60	22.40178	34.22035	0.708
15	60	21.32328	31.47028	1.19
15	60	20.35937	26.47412	1.614
15	60	17.44476	21.0032	2.002
15	60	16.32051	18.61725	2.282
15	60	9.848929	7.699687	2.584
15	50	22.91443	35.92867	0.32
15	50	22.44826	36.64494	0.714
15	50	22.00954	34.60997	1.084
15	50	20.53203	25.63245	1.53
15	50	19.80834	28.26882	1.803
15	50	16.34401	18.92221	2.124
15	50	14.97822	19.44611	2.516
15	50	8.332	8.435812	2.691
25	70	23.38329	30.89194	0.256

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
25	70	22.9	32.11676	0.929
25	70	21.52959	23.30044	1.386
25	70	20.05541	21.05523	1.778
25	70	19.3968	22.8171	1.99
25	70	18.86939	21.44065	2.168
25	70	12.67789	8.170579	2.573
25	60	23.34706	32.53443	0.24
25	60	23.25302	29.59519	0.63
25	60	21.21907	30.80661	0.83
25	60	17.93744	17.99147	1.46
25	60	17.2	13.01688	2.364
25	60	8.7	7.946623	2.657
25	50	23.88209	32.37764	0.511
25	50	23.39181	30.9816	0.84
25	50	19.57039	18.51245	1.738
25	50	14.05418	13.25636	2.288
25	50	7.055554	7.40463	2.634
5	50	24.55	3.27	0.63
5	50	22.34	3.32	1.67
5	50	13.71	3.29	2.28

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	24.62	3.44	0.44
5	50	23.06	3.33	1.23
5	50	19.16	3.52	1.9
5	50	24.51	3.7	0.66
5	50	21.68	3.52	1.58
5	50	19.08	3.05	1.91
5	50	24.04	4.12	0.48
5	50	22.19	4.13	1.48
5	50	19.48	4.17	2.21
5	50	24.38	3.38	0.49
5	50	23.05	3.08	1.38
5	50	18.25	3.24	1.86
5	50	25.73	2.39	2.22
5	50	23.13	2.3	2.64
5	50	18.8	1.83	2.78
5	50	25.5	2.72	0.59
5	50	24.65	2.65	1.82
5	50	19.78	3.78	2.47
5	50	23.2	3.28	0.83
5	50	22.41	3.53	1.37

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	20.38	2.42	1.73
5	50	24.25	3.64	0.8
5	50	23.35	2.24	1.59
5	50	19.48	2.21	2.03
5	50	24.88	3.01	0.79
5	50	21.85	3.08	1.2
5	50	14.47	1.79	2.12
5	50	23.97	3.35	0.87
5	50	21.83	3.29	1.58
5	50	18.4	2.26	1.93
5	50	24.16	2.73	0.87
5	50	20.75	1.49	1.53
5	50	19.38	1.77	1.9
5	50	26.88	3.4	0.46
5	50	25.6	3.26	1.48
5	50	21.63	1.73	2.01
5	50	24.4	3.68	0.73
5	50	20.99	2.93	1.24
5	50	17.58	2.07	1.5
5	50	25.95	3.27	0.55

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	25.03	3.05	1.19
5	50	19.32	1.84	1.94
5	50	23.77	3.59	0.63
5	50	22.56	2.97	1.67
5	50	17.85	2.16	2.14
5	50	21.96	3.65	0.7
5	50	20.86	1.78	1.56
5	50	19.42	2.62	2.51
5	50	24.45	3.66	0.74
5	50	20.22	1.63	1.2
5	50	13.1	1.32	1.81
5	50	25.87	3.24	0.53
5	50	23.26	3.06	1.45
5	50	20.35	2.2	2.04
5	50	25.22	3.87	0.51
5	50	23.58	1.78	1.68
5	50	19.37	2.98	2.2
5	50	24.01	3.67	0.75
5	50	19.9	2.55	1.49
5	50	15.32	0.85	2.02

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	24.51	3.72	0.77
5	50	20.7	1.17	1.68
5	50	17.09	1.12	2.23
5	50	24.51	4.03	0.72
5	50	21.9	1.83	1.4
5	50	18.03	1.29	1.84
5	50	23.15	3.42	0.59
5	50	21.44	2.73	1.27
5	50	17.83	2.44	1.95
5	50	25.45	3.01	0.72
5	50	23.64	2.68	1.65
5	50	18.29	2.49	2.31
5	50	26.31	2.5	0.55
5	50	24.91	3.81	1.57
5	50	21.76	1.55	2.05
5	50	24.26	5.58	0.55
5	50	23.52	3.21	1.13
5	50	18.57	1.86	2.13
5	50	24.17	3.38	0.47
5	50	21.68	2.95	1.47

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	18.54	2.82	1.97
5	50	24.37	3.68	0.6
5	50	23.16	3.33	1.63
5	50	19.13	2.45	2.16
5	50	23.44	4.4	0.58
5	50	21.97	2.34	1.3
5	50	17.49	3.21	1.96
5	50	26.19	3.33	1.05
5	50	24.74	3.57	1.81
5	50	20.38	3.09	2.93
5	50	23.5	4.52	0.53
5	50	22.44	2.53	1.23
5	50	19.33	1.6	1.99
5	50	24.56	4.33	0.69
5	50	23.92	3.82	1.25
5	50	22.1	2.65	2.02
5	50	25.04	3.83	0.44
5	50	24.34	3.57	1.05
5	50	22.95	3.49	1.47
5	50	24.77	3.25	0.68

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	23.95	3.37	1.02
5	50	18.19	2.98	1.82
5	50	24.51	4.01	0.23
5	50	24.09	3.57	1.11
5	50	21.51	2.7	1.72
5	50	25.41	4.04	0.44
5	50	24.79	3.73	0.97
5	50	23.32	4.28	1.67
5	50	23.83	2.82	0.4
5	50	23.45	2.84	0.95
5	50	22.16	2.32	1.66
5	50	24.39	4.2	0.44
5	50	24.07	3.83	1.05
5	50	22.79	3.06	1.93
5	50	24.55	3.68	0.29
5	50	23.46	3.29	1.09
5	50	20.92	2.84	1.54
5	50	25.02	2.85	0.51
5	50	24.4	2.46	1.31
5	50	22.27	1.94	1.94

Crack location	Temperature	Natural Frequency	Amplitude	Crack Depth
5	50	24.39	3.93	0.67
5	50	23.99	3.24	1.08
5	50	22.26	2.16	2.11
5	60	24.24	2.26	0.5
5	60	20.9	1.67	0.78
5	60	25.07	4.34	0.31
5	60	23.16	4.12	1.29
5	60	19.06	2.16	1.82
5	60	24.83	3.93	0.53
5	60	23.84	4.61	1.01
5	60	19.19	3.25	1.79
25	60	19.64	22.08	1.46
25	60	17.97	15.68	2.36
25	60	9.28	8.03	2.66
25	70	25.97	31.21	0.26
25	70	25.51	32.65	0.93
25	70	24.02	27.96	1.39
25	70	22.24	25.55	1.78
25	70	20.84	28.93	1.99