U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Master's Thesis

Machine Learning-assisted virtual certification of aerospace structural composite components

Luís Filipe Morais Fonseca Rodrigues

FEUP Supervisor:
Prof. Carolina Furtado

FEUP Co-Supervisor:
Prof. João Ferreira

INEGI Supervisor:
Dr. Igor Lopes

Porto, June 2024

Machine Learning-assisted virtual certification of aerospace structural composite components

Luís Filipe Morais Fonseca Rodrigues

Abstract

The generation of design allowables for composite laminates is of utmost importance for the design and certification of the composite structures used in the aerospace industry. The determination of these design allowables usually relies on expensive and time-consuming experimental test campaigns. With the increase of computational power, and the development of high-fidelity numerical models that accurately represent the response of composite materials, alternatives to generate design allowables based on finite element simulations have also been sought out to reduce the certification costs. However, these solutions are still computationally expensive. The recent advances on machine learning techniques open a new window of possibilities for the faster prediction of the structural response of materials, by allowing the definition of surrogate models that continuously and analytically describe the design space.

In this work, a open-hole high-fidelity model is employed to generate data that is then used to train four different types of machine learning algorithms. A framework for data treatment is proposed, including outlier detection, feature selection, data splitting, hyperparameter selection, data normalization and multi-output dealing approaches for better model performance. This resulted in surrogate models that are able to predict the notched strength, failure mode and stress-strain curves of several materials, stacking sequences and notched geometries.

The trained machine learning algorithms are evaluated by comparing them to numerical and real experimental results. It is shown that the models are able to make accurate predictions of open-hole multi-directional composite coupons in milliseconds, corresponding to a time-to-prediction speed-up of over 10k times. Furthermore, it is shown that the models can be used for the generation of design allowables, using a Monte Carlo approach, owing to their exceptional efficiency.

Keywords: Machine Learning, Surrogate Modelling, Composite Laminates, Open Hole Test, Aeronautical Certification, Design Allowables, Finite Element Analysis.

Machine Learning-assisted virtual certification of aerospace structural composite components

Luís Filipe Morais Fonseca Rodrigues

Resumo

A geração de valores de projeto para materiais compósitos é de extrema importância no *design* e certificação das estruturas compósitas utilizadas na indústria aeronáutica e aeroespacial. A determinação desses valores geralmente depende de campanhas de testes experimentais dispendiosas e demoradas. Com o aumento do poder computacional e o desenvolvimento de modelos numéricos de alta fidelidade que representam com precisão a resposta dos materiais compósitos, têm sido procuradas alternativas para gerar valores de projeto baseados em simulações de elementos finitos, visando reduzir os custos de certificação. No entanto, essas soluções ainda são computacionalmente exigentes. Os avanços recentes nas técnicas de *Machine Learning* abrem uma nova janela de possibilidades para a previsão mais rápida da resposta mecânica dos materiais, permitindo a definição de modelos de substituição que descrevem de forma contínua e analítica o espaço de projeto.

Neste trabalho, um modelo de alta fidelidade de *Open Hole Tension* é utilizado para gerar dados que são usados para treinar quatro tipos diferentes de algoritmos de *Machine Learning*. É proposta uma metodologia para o tratamento de dados, incluindo deteção de *outliers*, seleção de parâmetros de *input*, divisão de conjuntos de dados, seleção de hiperparâmetros, normalização de dados e abordagens para o tratamento de múltiplos *outputs*, visando melhorar o desempenho do modelo. Isto resultou em modelos capazes de prever a resistência dos laminados, o modo de falha e as curvas tensão-deformação, para várias propriedades de material, sequências de empilhamento e configurações geométricas.

Os algoritmos de *Machine Learning* treinados são avaliados comparando-os com resultados numéricos e experimentais reais. É demonstrado que os modelos são capazes de fazer previsões precisas de laminados compósitos multidirecionais com furo aberto em milissegundos, correspondendo a um aumento de velocidade de previsão superior a 10.000 vezes. Além disso, é evidenciado que os modelos podem ser utilizados para a geração de valores de projeto utilizando uma abordagem de Monte Carlo devido à sua eficiência computacional excepcional.

Acknowledgements

I would like to express my profound gratitude to my supervisor, Prof. Dr. Carolina Furtado, whose expertise, availability, and unwavering guidance enabled me to learn tremendously and go beyond the initial goals. Your insightful advice and consistent support have been vital throughout this journey.

A word of deep gratitude to Prof. Dr. João Ferreira, my co-supervisor. It has been a pleasure to learn from you over the last several years. Your mentorship has significantly shaped my university experience, making it more enjoyable, interesting and rewarding. You have shown me a whole new area of knowledge. This work would not have been possible without your initiatives and invaluable contributions.

To Dr. Igor Lopes, whose contributions, since the beginning, have undoubtedly enriched the outreach and impact of this work. Your detailed suggestions and corrections had a substantial influence on the quality of this thesis and the breadth of the results obtained. Your commitment to excellence has been instrumental.

To Nora Kovacs, for dedicating a significant amount of time to participating in our idea exchanges. Without your bright suggestions and input, it would have been impossible to reach results with this quality.

I treasure the love and support that have come from my parents throughout my life, in all the important moments, good or bad: my parents are the best parents in the world. My most profound word of gratitude to my grandmother, for always worrying, for all the prayers and candles, and for teaching me that there is no limit to kindness and having a loving heart. To my brothers, João, for always paving the way for me, and José, for always being here and pushing me more and more.

To my friends and family. I wouldn't dare name all of you nor enumerate the contributions you have had for this path, this page would be all too small. Thank you for always believing in me and for being there when I need you the most. To those who tell me this work is boring and to those who are overly excited about it, thank you. For the lunches, the coffees, the discussions and the nights out. It is a blessing to sincerely say that I have these true friendships.

To all the members of Porto Space Team, from the oldest (Vicente) to the youngest, you have been an integral part of my university experience. I have learned from you, been challenged by you, and I am grateful for the camaraderie and support we have shared.

Luís Rodrigues

“Não quero desculpas.”
~ I will not be accepting excuses.

Leonel Neves

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this introductory Chapter, a general context on the problem at hand will be provided, followed by the motivation behind this work and its main goals. An outline of the document's structure will also be presented.

1.1 Context

Carbon Fiber Reinforced Polymers (CFRPs) have emerged as a very important material in the aeronautical and aerospace industries, especially due to their exceptional strength-to-weight ratio. In the aeronautics industry, for instance, CFRPs have replaced traditional metallic alloys in many structural components, leading to enhanced performance and fuel efficiency of the aircrafts.

However, the certification of CFRP structures for these applications presents numerous challenges. The traditional certification process includes the determination of design allowables, that take into account material, geometric and manufacturing variability, complex factors in the case of CFRP materials. This process is done by performing extensive physical tests, which are both time-consuming and expensive. Given the complexity and variability of composite materials, there is a need for more efficient, although still reliable, certification methods. This has driven the exploration of alternative approaches, such as high-fidelity numerical simulation and Machine Learning-assisted methodologies (Cumbo et al. (2022)).

Numerical approaches offer a promising alternative. These techniques can replicate the physical behavior of materials under various conditions. In spite of this, these kind of simulations are still very costly, in computational terms. Machine Learning (ML) can analyze samples of data generated through these models, to create a surrogate model that is able to identify patterns and correlations in data, and in this way replicate the costly numerical approach in the rest of the domain with very "cheap" computations. Therefore, it provides an interesting approach to quickly obtain design allowables by performing predictions of the mechanical response for several geometric, layup and/or material variations. This topic is further explored in the current dissertation.

1.2 Motivation and Objectives

The motivation for the present work originates from several needs in the aeronautical and aerospace industries, related to the certification of CFRP structures.

As explained, one of the primary motivations is the high cost associated with traditional experimental certification processes. This is the case because these industries are highly regulated: design and manufacturing must comply with a wide range of standards. As a result, the certification of new structures and materials involves extensive testing under various conditions, starting from simple coupons up to the testing of full scale large components. By developing alternative certification methods that leverage ML capabilities, there is the potential of reducing the number of tests and numerical simulations required in such processes.

Unlike traditional materials, CFRPs can be tailored to specific applications by altering their layups and stacking sequences, for example. This increased design freedom can be an advantage in the one hand, but leads to vast design spaces, whose optimal solutions are therefore much harder to find. The use of ML models that can instantly predict material strength and behaviour would be of great advantage for an engineer.

Furthermore, calibration of material properties and analysis of sensitivity to the variation of certain parameters could be eased with such models. Instead of resorting to high fidelity simulations, that may take many days to complete, occupying computational resources, ML models have the potential of providing reliable results in seconds.

Lastly, the rapid growth of ML use in other sectors and industries, having demonstrated great effectiveness and impressive results, motivates its use in the area of certification of CFRP structures in particular.

The main objective of this dissertation is: **to develop ML models that predict the mechanical response of CFRP coupons under a certain type of testing scenario, for different stacking sequence configurations, geometric parameters and material properties, paving the road to virtual design and certification**. To achieve this, and more in particular, the following goals were set, in light of the presented motivating factors:

- To develop a methodology for data generation, data handling, ML model building, training and evaluation, based on state of the art approaches;
- To create reliable and accurate ML models for the prediction of Ultimate Strength, Failure Mode and Stress-Strain Curves for CFRP laminates under Open Hole Tension testing;
- To build models to solve these problems for a specific material only and then expand this by considering wider ranges of material properties that encompass many CFRP materials;
- To assess different types of ML algorithms in the proposed problems and compare their performance and usability with each other;
- To validate the proposed framework and models against experimental data and existing high-fidelity numerical models;
- To generate virtual Design Allowables using the ML models and evaluate the viability of using such approaches in structural certification;

- To demonstrate the potential of ML in reducing the time and cost associated with the certification of CFRPs.

1.3 Dissertation Outline

To accomplish the proposed goals, this document will start with a brief literature review on Composite Materials in Chapter 2, focusing on their certification process; on Open Hole Tension tests, which will be the coupon level tests considered throughout this work; on CFRP material properties; on laminate parameters that describe a given ply stacking sequence; and, finally, a review on failure modes and mechanisms will be made.

After this, in Chapter 3, focus will turn into best practices in Machine Learning. First, a more detailed overview on surrogate modelling will be presented, followed by the most important types of ML algorithms that will be used in this work, where the underlying procedures, their main advantages, disadvantages and parameters will be detailed. Then, outlier detection strategies, feature selection methods, data splitting techniques, hyperparameter selection approaches, data scaling procedures, and multi-output handling methodologies will be addressed. All of these will be important when developing and applying our methodology.

In Chapter 4, the methodology for generating data; pre-processing data; exploiting additional features in this data; building ML models; evaluating these models; and evaluating the sufficiency of data generated will be explained. Additionally, an illustrative database will be generated and used to train an ML model.

In Chapter 5, by following the presented methodology, a dataset will be generated to train models for a fixed material and coupons with variable geometry and stacking sequence. Models to predict material strength, failure mode and stress-strain curve will be built and evaluated by using them on never before seen data, by performing sensitivity analysis and by using them on data from experimental campaigns from the literature that used this material, whenever possible.

After this, in Chapter 6, the same analysis will be conducted, but this time a wide range of materials will be considered. This will allow for the comparison of ML model predictions with other experimental campaigns, that used different materials. Furthermore, as these models will be able to capture geometric and material variability, these will be used to generate virtual Design Allowables.

Finally, in Chapter 7, the main takeaways from this work will be summarized. Future work perspectives will also be addressed.

Terminology and nomenclature will be introduced throughout the document. Some data and further analysis is provided in the Appendices.

Chapter 2

Composite Materials' Certification

2.1 Background and terminology

2.1.1 Carbon Fiber Reinforced Polymers

The use of Carbon Fiber Reinforced Polymers (CFRPs) in aeronautic and aerospace structures has seen a very rapid growth since the 1990s. Today, these are a very important group of materials, sitting side-by-side with aluminum alloys as the most used materials in many applications, especially in the aeronautical industry (Mouritz (2012)). As is common in composites, the material consists of a stiff component, in this case the carbon fibers, and a softer component, in this case the polymeric matrix, most usually made of epoxy (Da Silva et al. (2013)). These are combined and a synergistic effect is observed, meaning that their combined properties and behaviour is much better than that of each component separately. The fibers provide strength and stiffness, while the polymer provides ductility (Mouritz (2012)).

CFRP laminates consist of multiple layers (also termed laminae or plies) of polymer matrix and fiber reinforcement, oriented at some direction, bonded together to form a solid material.

The main advantage of CFRPs is their specific strength and stiffness: these allow for the construction of lightweight yet safe structures. This is valuable in the aforementioned industries: the use of CFRPs in substitution of aircraft-grade aluminum alloys in structures, with careful design, may lead to fuel savings of up to 20% (Mouritz (2012)).

There are several other advantages, as stated by Mouritz (2012). CFRPs are better suited for integrated manufacture, leading to a lower number of components and reduced assembly costs; are less prone to suffering from fatigue; are not affected by corrosion; are better heat insulators and have a lower coefficient of thermal expansion.

Moreover, for design purposes, the intrinsically anisotropic nature of CFRPs is also an advantage, as it allows for tailoring of the laminates for specific solicitations. This increases their structural efficiency. Although this is true, the structural analysis on structures of this type is more difficult than that for classical isotropic materials, such as metals, for which the analysis tools, both analytical and computational, are well established and consolidated.

Other disadvantages of using CFRPs include slower manufacturing processes; often higher material, certification and production costs (although counteracted by the gained fuel efficiency); low through thickness mechanical properties (as the fibers are oriented in-plane); susceptibility for delamination; higher notch sensitivity than metals; their flammability and low

electric conductivity (Mouritz (2012)). All of these must be factored in when choosing to use CFRPs for structural components.

2.1.2 Layups and stacking sequences

As mentioned, laminates consist on a series of plies bonded together. Throughout this work, the orientation of the fibers in each ply of the laminate will have great importance. Most conventionally, laminates have plies oriented at angles of 0°, 45°, -45° and 90°, in relation to an established/principal direction, fixed for the whole laminate (this may be, for example, the direction of the solicitation).

A layup is defined solely by the number of plies at each direction. Each laminate has a layup: N_{0° layers at 0°; N_{45° layers at 45°; N_{-45° layers at -45°; N_{90° layers at 90°. A layup is therefore defined independently of the particular order of the plies.

On the other hand, a stacking sequence (or stacking order) takes into account the order at which the plies are stacked.

To illustrate these concepts, an example will be given. A given laminate may have a layup with $N_{0^\circ} = 4$; $N_{45^\circ} = 2$; $N_{-45^\circ} = 2$; $N_{90^\circ} = 2$. The laminate has a total of 10 plies. Its stacking sequence can be $[0, 45, -45, 0, 90]_S$. In this representation, the order of the plies is clearly shown. The subscript S indicates that the stacking sequence is symmetric (the five angles presented repeat themselves in the reverse order). It is clear that from a given stacking sequence the layup can be retrieved, but the opposite operation is not possible. When presenting stacking sequences, the angles will be given in degrees. This unit will be omitted for conciseness.

2.2 CFRP structures' certification

The certification process of composite structures, especially for aeronautical structures, is very different from the certification of metallic materials. Classical procedures used for metals cannot be directly transposed to CFRPs, and the reasons are many, as has been discussed in the industry for many years. As presented by Rouchon (1990) and Department of Defense (2002), composite materials have:

- higher sensitivity to environment, especially for matrix dominated properties;
- extreme sensitivity to delamination. This leads to damage susceptibility upon low velocity impacts;
- less sensitivity to fatigue than metals;
- mechanical properties are only defined during the manufacturing process;
- poor electrical conductivity;
- multiple possible failure modes and mechanisms.

The philosophy currently employed for composite material certification involves a building block approach.

2.2.1 Building Block approach

As stated by Rouchon (1990), the building block approach is a consistent way of showing that a structure made from composite materials is airworthy. It is an evidence based analysis that involves several stages of tests, beginning at elementary coupons and culminating at the testing of full sized structures, as illustrated in the pyramid seen in Figure 2.1. When certificating a component, it is common to go through all of the five presented levels: coupon, element, detail, sub-component and component.

It is crucial to note, as mentioned by Rouchon (1990), that "because of the large amount of testing which is needed to cover all critical design features taking into account the effects of adverse environment on failure modes and variability, structural substantiations with composites are costly in time and money". This is the main motivation for numerical modelling and surrogate modelling in the industry: the decrease in costs due to expensive experimental campaigns.

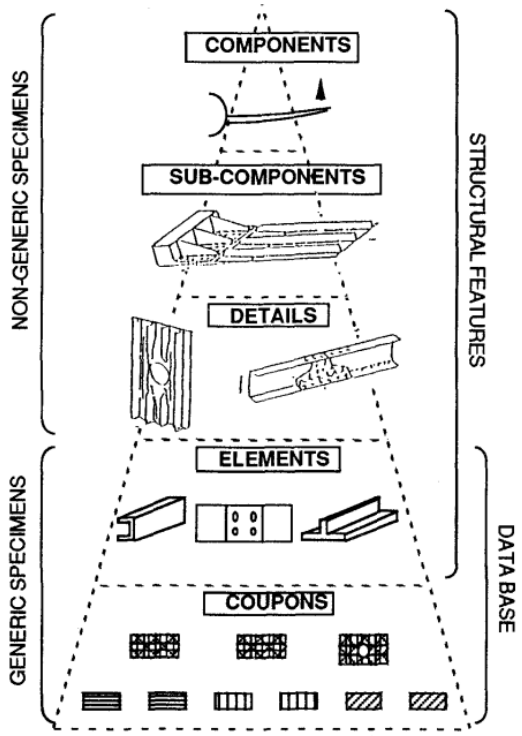


Figure 2.1: Pyramid of tests according to the building-block approach (Rouchon (1990)).

This building block approach covers five essential structural complexity levels (Department of Defense (2002)):

- Constituent level: evaluation of properties of the fiber and matrix;
- Lamina level: evaluation of the properties of fibers and matrix together;
- Laminate level: evaluation of the response of the material under a given layup/stacking sequence design;

- Structural element level: evaluation of the material's ability to tolerate common discontinuities;

- Structural sub-component (or higher): evaluation of behaviour and failure on complex structural assemblies.

This work will focus mainly on the laminate level, using the testing of coupons. The most common tests on coupons include tests on unnotched specimens; on open-hole specimens; on filled hole specimens; on bolted joints; on bonded joints; low velocity impact tests; edge impact tests and lightning strike tests. In particular, our approach will be centered in open-hole specimen tests, that will be further expanded on in Section 2.3.

2.2.2 Design Allowables

From the previously mentioned tests, design allowables are generated. These are values of the structural properties that can be used for design purposes. These allowables are a set of values generated through a statistical analysis that takes into account material and manufacturing variability. This variability may have several sources, such as environmental effects and wear-out of equipment during manufacturing; material variability from batch to batch; variability related to testing equipment and other inherent variability factors (Department of Defense (2002)).

The design allowables are therefore statistical measures, and may be classified as A-basis or B-basis, depending on the level of confidence on the given value. A and B basis design allowables can be used in different situations: an A-basis allowable is a material property with a 95% lower confidence bound on the first percentile of a given population of measurements, and is used on primary structural components. A B-basis allowable is a material propriety with a 95% lower confidence bound on the tenth percentile of a given population of measurements, and is used on secondary structural components. The difference between primary and secondary structures is that the failure of a primary structure leads to catastrophic failure of the whole system/vehicle, whereas the failure of a secondary structure doesn't: these are usually redundant structural components and, upon their failure, loads are safely distributed to other structural members (EASA (2023)).

Given the statistical nature of these design allowables, a large number of experimental tests must be performed. The recomended approach nowadays is the CMH-17 approach (Department of Defense (2002)). The method to apply depends on if data is structured (can be grouped in different material batches, and tests are done under different controlled environmental conditions) or unstructured. For example, as the material itself is a large source of variability, specimens can be grouped in different batches based on this. Depending on several factors, such as the structure of the data and the variability sources, CMH-17 proposes different statistical distributions to generate the allowables. Crucially, the recommended number of tests is on the order of dozens of tests.

To reduce this number of tests, approaches based on numerical simulations have been extensively proposed in the literature, showing promising results (Cumbo et al. (2022)). These, by considering the material variability, seek to use a reduced number of experimental data to support the numerical results.

2.3 Open Hole Tension test

The experimental procedure used to test specimens under Open Hole Tension (OHT) is standardized (ASTM (2018)). As is stated in this standard, the factors that influence the notched tensile strength of CFRP specimens include: "material, methods of material fabrication, accuracy of lay-up, laminate stacking sequence and overall thickness, specimen geometry (including hole diameter, diameter-to-thickness ratio, and width-to-diameter ratio), specimen preparation (especially of the hole), specimen conditioning, environment of testing, specimen alignment and gripping, speed of testing, time at temperature, void content, and volume percent reinforcement". These factors should therefore be controlled and reported.

Specimens are usually instrumented with strain gauges. This requires correct surface treatment. In Figure 2.2 the geometry and instrumentation of the specimens tested by Camanho et al. (2007) is presented.

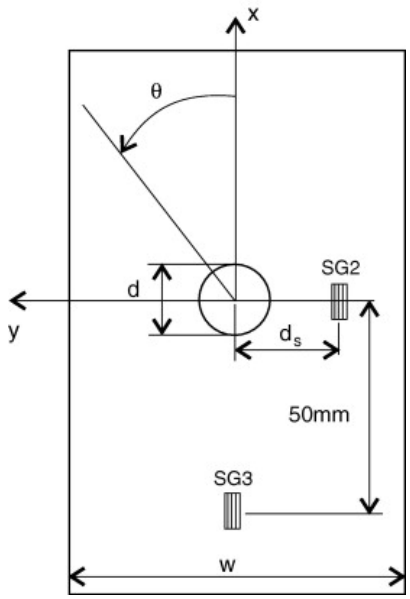


Figure 2.2: Geometry for OHT tests and strain-gauge (SG) placement in the test campaign performed by Camanho et al. (2007).

2.4 Material properties

Due to the anisotropic nature of the material under study, there are many important material properties that can influence OHT performance of CFRP laminates. Some of these are not easily measurable and usually require careful calibration. In this section, CFRP ply level material properties will be presented.

Figure 2.3 (Furtado et al. (2019)) displays the uniaxial response of composite materials under different solicitations. In this Figure, Furtado et al. (2019) present many of the material properties that will now be succinctly described, in this case to be use in a Finite Elements Method (FEM) model (l^* is a characteristic length of the finite elements).

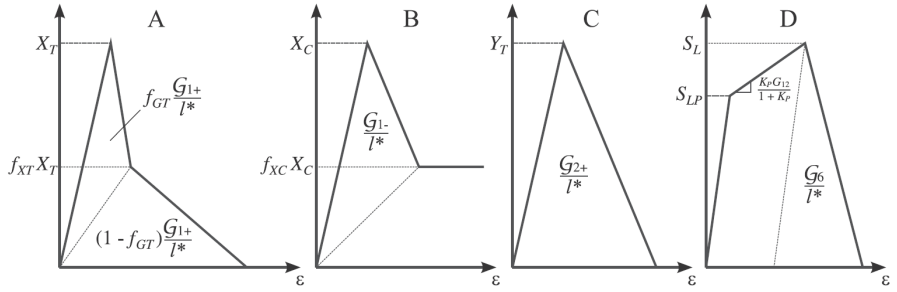


Figure 2.3: Uniaxial response of CFRP lamina in (A) longitudinal tension, (B) longitudinal compression, (C) transverse tension or compression and (D) in-plane shear (Furtado et al. (2019)).

In Chapter 5 all material properties and the ply thickness, t, will be kept constant when performing numerical simulations. These will be set for the known values of a particular material, as will be detailed.

In Chapter 6, simplifications and assumptions will be made regarding some of the material properties, to reduce the input space of allowed material variability.

2.4.1 Ply elastic properties

The relevant elastic properties, that must be given as inputs for FEM simulations, are:

- E_1: Longitudinal Tension Young's modulus.
- E_{1c}: Longitudinal Compression Young's modulus.
- E_2: Transverse Young's modulus.
- ν_{12}: In-plane poisson's coefficient.
- G_{12}: In-plane shear modulus.

2.4.2 Ply strength properties

Regarding ply strength properties, the ones most relevant for the FEM simulations are:

- X_T: Longitudinal Tensile strength.
- f_{XT}: Related to inflection point in longitudinal tension. $f_{XT}X_T$ is the longitudinal tensile stress at the inflection point.
- X_C: Longitudinal Compressive strength.
- f_{XC}: Related to inflection point in longitudinal compression. $f_{XC}X_C$ is the longitudinal compressive stress at the inflection point.
- Y_T: Transverse Tension strength.

- Y_C: Transverse Compression strength.
- S_L: In-plane shear strength.
- S_{LP}: Shear stress that activates plastic flow.
- K_P: Shear incremental stiffness under plastic flow.

2.4.3 Ply fracture toughness properties

Regarding ply toughness properties, the ones of interest for the numerical simulations are:

- G_{XT}: Intralaminar fracture toughness for longitudinal tension. (In Figure 2.3 this property is denoted alternatively as G_{1+}).
- f_{GT}: Fraction of G_{XT} dissipated up to the inflection point.
- G_{XC}: Intralaminar fracture toughness for longitudinal compression. (In Figure 2.3 this property is denoted alternatively as G_{1-}).
- G_{2+}: Mode I fracture toughness. (G_{IC} is the equivalent interlaminar Mode I fracture toughness).
- G_6: Mode II fracture toughness. (G_{IIC} is the equivalent interlaminar Mode II fracture toughness).

2.5 Lamination parameters

The most common way to represent a certain laminate is explicitly by its stacking sequence, as mentioned in Section 2.1.2. This is an array of angles (0°, ±45° and 90° angles). This representation is simple, but it has the drawback of having a variable dimension, depending on the number of layers of the laminate: each laminate stacking sequence may have a different number of angles. This makes the explicit representation not ideal for applications in Machine Learning (Furtado et al. (2021)).

Alternatively, a given stacking sequence may be represented in a compact way by its lamination parameters, ζ, proposed by Tsai and Pagano (1968). With these twelve parameters and a thickness variable, a laminate is uniquely defined (Furtado et al. (2021)). The in-plane parameters, $\zeta^A_{1,2,3,4}$, coupled parameters, $\zeta^B_{1,2,3,4}$, and out-of-plane parameters, $\zeta^D_{1,2,3,4}$ are defined in Equation (2.1):

$$\zeta^A_{\{1,2,3,4\}} = \frac{1}{h}\sum_{i=1}^{N}\begin{Bmatrix}\cos(2\theta_i)\\ \cos(4\theta_i)\\ \sin(2\theta_i)\\ \sin(4\theta_i)\end{Bmatrix}(z_i - z_{i-1}), \tag{2.1a}$$

$$\zeta^B_{\{1,2,3,4\}} = \frac{2}{h^2}\sum_{i=1}^{N}\begin{Bmatrix}\cos(2\theta_i)\\ \cos(4\theta_i)\\ \sin(2\theta_i)\\ \sin(4\theta_i)\end{Bmatrix}\left(z_i^2 - z_{i-1}^2\right), \tag{2.1b}$$

$$\zeta^D_{\{1,2,3,4\}} = \frac{4}{h^3}\sum_{i=1}^{N}\left\{\begin{array}{c}\cos(2\theta_i)\\ \cos(4\theta_i)\\ \sin(2\theta_i)\\ \sin(4\theta_i)\end{array}\right\}\left(z_i^3 - z_{i-1}^3\right), \tag{2.1c}$$

where h is he thickness of the laminate, θ_i denotes the orientation of the fibers on a certain ply i, at height $z \in [z_{i-1}, z_i]$, and N is the number of plies of the laminate.

These lamination parameters are interrelated: when the value of some of them is fixed, the others cannot take any arbitrary value. For the most general case, without any additional restrictions, all twelve lamination parameters are needed to define a laminate, and their feasible region is not mathematically known (Grenestedt and Gudmundson (1993)). This domain is known to be convex, but only known for certain combinations of lamination parameters (Setoodeh et al. (2006)).

In this work, laminates that are symmetric and balanced will be focused on. These restrictions have the implication that only five lamination parameters are needed to uniquely define such laminates.

A balanced laminate is such that it has the same number of 45° as -45°. More generally, Daniel et al. (1994) defines balanced laminates as those that consist "of pairs of layers with identical thickness and elastic properties but having +θ and -θ orientations of their principal material axes with respect to the laminate principal axes". (Of course for 0° and 90° oriented plies, the sign change does not produce any difference.) For these laminates, $\zeta^A_{3,4} = 0$.

A symmetric laminate is one in which the angles of the plies are symmetric in relation to the middle plane of the laminate. More precisely, Daniel et al. (1994) defines symmetric laminates as those in which for "each layer on one side of a reference plane (middle surface) there is a corresponding layer at an equal distance from the reference plane on the other side with identical thickness, orientation, and properties". In such laminates, $\zeta^B_{1,2,3,4} = 0$ and $\zeta^D_4 = 0$.

With these restrictions, a symmetric and balanced laminate is then defined by two in-plane parameters, $\zeta^A_{1,2}$, which are only layup dependent and therefore do not vary with stacking order, and three out-of-plane parameters, $\zeta^D_{1,2,3}$, that depend both on layup and stacking sequence.

It is important to note that from the two in-plane parameters, the fraction of the total number of plies for each direction is uniquely defined. For example, the layups $N_{0°} = 4$; $N_{45°} = 2$; $N_{-45°} = 2$; $N_{90°} = 2$ and $N_{0°} = 8$; $N_{45°} = 4$; $N_{-45°} = 4$; $N_{90°} = 4$ have a different total number of plies, but the fractions of plies at each direction is the same. Therefore, they will have the same values of $\zeta^A_{1,2}$ parameters. Moreover, these values of $\zeta^A_{1,2}$ will corresponding solely to layups with the same fractions of plies at each direction as those here shown.

2.6 Failure modes

There are several ways in which failure of CFRPs may occur. Firstly, we can distinguish failure within a given ply of the laminate (intralaminar failure) or failure by separation of contiguous plies (interlaminar failure) (Daniel et al. (1994)). There are many different in-plane failure mechanisms: several have been observed, depending on the solicitations the laminates are subjected to, their material and configuration.

When loaded in the fiber direction, under tension (longitudinal tension), the fibers typically fail first, as these are stiffer and have lower ultimate strain. This is a fiber dominated failure. At

the micro-scale, not all fibres break at the same time: single fiber breaks occur at the weakest points, resulting in non uniform stress states.

Under longitudinal (regarding fiber orientation) compression, different mechanisms are at play: failure is related to kinking of the fibers. For CFRPs, post-failure well defined fracture planes are usually observed. Under this solicitation, shear driven fiber compressive failure may also occur.

When tensile loading occurs in the in-plane direction perpendicular to the fibers (transverse tension), stress and strain concentrations appear in the matrix and in the fiber-matrix interface.

Under transverse compression, the most important stresses are interfacial, leading to matrix shear failure and interfacial debonding. Ultimately, the laminate fails in an overall shear dominated failure mode.

Similarly, under in-plane shear loading, stress concentrations appear in the matrix or at the interfaces, leading to a shear dominated failure in the matrix or to fiber-matrix debonding.

It is evident, after having analysed these failure mechanisms, that the mode in which a laminate fails will depend not only on the main type of solicitation (tension, compression or shear), and on the properties of its constituents, but also on the orientation of the plies in the laminate (that dictate if the solicitation is in the longitudinal or transverse direction).

Chapter 3

Machine Learning

3.1 Surrogate modelling in engineering

Over the last decades, computer simulation and numerical models, such as Finite Elements Method (FEM) based models, used to represent physical systems, have begun to play a crucial role in solving engineering problems. These are used in a variety of situations, such as optimization, design or structural analysis. Although this is true, these computer simulations have a tendency to be computationally expensive due to their excessively detailed representation of the real-world (Alizadeh et al. (2020)). For most industrial applications, there is no need for this high-fidelity modelling, as the main practical aim is not to understand the underlying physical phenomena, but instead to have accurate, concrete and low-cost estimates of certain variables such as, in the particular case of composite coupon testing, design allowables, expected modes of failure or shape of stress-strain curves. Computers are becoming faster and more powerful, but the computational complexity demand for the use of these models is still much higher than the available computational power.

The development of surrogate modelling is aimed at replacing these expensive numerical models when solving engineering problems. These surrogate models should provide the outputs wanted by the user, when provided with the same inputs, as the aforementioned simulations for a particular problem. This should be done, though, with much lower computational costs. Surrogate models are models of a model (Jiang et al. (2019)). They make educated guesses as to what an engineering function might look like, based on a few points in the input space. The models, that have some built-in assumptions, learn and become a very effective low cost replacement of the original numerical function for a wide variety of purposes, as stated by Forrester et al. (2008).

Multiple factors, including linearity/nonlinearity, the nature of the problem, the desired level of accuracy, the speed of the process, the required amount of information, among others, impact the appropriateness of a surrogate model (Alizadeh et al. (2020)).

The surrogate models here proposed aim for the augmentation of results coming from an expensive simulation framework. The basic idea is for the surrogate to act as a "curve fit" to the available data so that results may be predicted without resorting to the use of the original expensive numerical model. The approach is based on the assumption that, once built, the surrogate will be many orders of magnitude faster than the primary source while still being usefully accurate when predicting away from known data points (Forrester et al. (2008)).

Usually, for the surrogate model to approximate the corresponding numerical model, it must be trained with data generated from the latter. This is, of course, a computationally expensive part of the process, and therefore a good Design of Experiments (DoE) must be performed. This consists on choosing a sample of the input space to be simulated using the numerical model. The results from these simulations can then be used to build the surrogate model that should be able to accurately make predictions for the rest of the input space (Alizadeh et al. (2020)).

Essentially, the main advantage of surrogate modelling is its fast, computationally inexpensive and accurate enough predictive power. The drawback of these models is that, in general, they return the variables of interest without explaining how these were obtained or what is the underlying phenomena from which they were retrieved.

3.2 Background and terminology

In this Section, several key concepts and approaches will be presented, so that the information in subsequent Sections is more clear and concise. Terminology used throughout the work will also be shown.

3.2.1 Supervised learning

Machine Learning (ML) can be subdivided into different learning approaches: unsupervised learning, supervised learning and semi-supervised learning.

Unsupervised learning is used to analyze and cluster unlabeled datasets. These algorithms are termed unsupervised because they discover patterns in data without the need for any human intervention. These can be used for clustering, association and dimensionality reduction of data (Delua (2021)). They are used to find patterns in data and are not suitable for surrogate modelling.

In supervised learning, the approach that most interests us in this work, ML models are trained using a dataset that contains data-points with both the inputs and desired target outputs. The model learns the mapping between the input and response variables. Using these labeled inputs and outputs, the model can measure its accuracy and learn over time (Delua (2021)). All models that will be presented in this work are supervised learning models: they are termed "supervised" because they are provided both with the input and output values for each training data-point: there is the need for human intervention in the sense that the models need to be given the correct "answers" to the problem at hand during the learning phase.

Semi-supervised learning, according to Bergmann (2023), is an hybrid approach that combines supervised and unsupervised learning to train ML models for the same approaches as supervised learning is used in.

3.2.2 Features and labels

In this work, the input parameters that are given to the surrogate model will be termed features. These may be, for example, geometric features, such as the width of a specimen or a hole diameter; material properties, such as elastic properties, material's resistance or toughness; or

stacking sequence related parameters, such as laminate parameters or other metrics obtained from the layup or ply order.

Output variables will be termed labels, often referred to as target values or target classes, depending if the model is a classifier or a regressor, as will be detailed in Section 3.2.3. Examples of these will be the specimen's strength; deformation at failure; coefficients of a polynomial approximating the stress-strain curve; or a failure mode.

Both features and labels are variables that may have different natures. The variables that will be used can be discrete or even binary (for example, a discrete feature could be a variable that indicates whether the layup is symmetric or not; and a discrete label may be a variable that indicates the type of failure that the specimen suffers), or they can be continuous (for example the width of a specimen is a feature given by a variable consisting of float point numbers, and the same can be said for a label such as the ultimate strength of a specimen).

As discussed in Section 3.2.1, during training, the models are able to access both the features and the true labels of a given data-point. The labels corresponding to the "ground-truth", needed for training, will come from accurate numerical models. Later, to assess or test the performance of a model, it will be provided with only the features: the model will try to predict the labels and these predictions will then be compared to the true labels.

3.2.3 Classification and Regression

Supervised learning problems are usually categorized as classification or regression problems. The main difference between these lies in the nature of the label variables.

In a regression problem, models are trained to predict a continuous numeric value. On the other hand, in classification problems, the ML model is built to predict a discrete label, in these also frequently termed class. Regression algorithms are further divided into linear and non-linear algorithms. Classification algorithms are divided into binary classifiers and multi-class classifiers (Gültekin (2023)).

It is important to note that this distinction is very important when assessing model performance, as very distinct metrics are used for regression and classification models. For example, in classification, accuracy is a common metric for assessing performance, whereas in regression the error is estimated using other metrics, such as a Root Mean Square Error. These metrics and others will be further detailed in Section 4.5.

3.2.4 Single-Output and Multi-Output problems

Problems in ML can be further described, regardless of the fact that they are a classification or regression problem, as Single-Output or Multi-Output problems.

A Single-Output problem is one where the model tries to predict a single label (either a class or continuous value). These are the simplest types of problems. An example is the prediction of the ultimate strength of a specimen, or its failure mode.

A Multi-Output problem is one where the objective is to predict more than one value or class at once. For these more complex problems, many different approaches exist, which will be further detailed in Section 3.9. An example of such a problem is the prediction of several polynomial coefficients of a polynomial that best fits a stress-strain curve.

3.2.5 Under-fitting vs Over-fitting

Under and over-fitting are two common problems in supervised ML. These problems are here succinctly presented and illustrated in Figure 3.1.

Under-fitting happens when a model is not complex enough, hasn't been trained on sufficient data or well enough, in such a way that it is incapable of capturing the important patterns in data. It is said that the model has low variance (Kasirajan (2020)).

Over-fitting, on the other hand, occurs when a model is too complex and so fits the training data too well. This leads to good performance on the training set but poor performance on the test set (data not seen during training). It is said that the model has a high bias because of the training set (Kasirajan (2020)). A model like this has poor generalization performance: it would be desirable that the model could make good predictions on data that it has not seen during training.

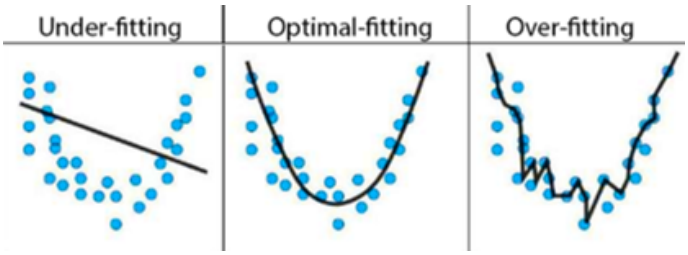


Figure 3.1: Illustration of under-fitting and over-fitting for a regression problem. Adapted from Bhande (2021).

3.2.6 Model interpretability

Model interpretability can be loosely defined as the degree to which a human can understand the cause of a decision of a model. The higher the interpretability of an ML model, the easier it is to comprehend why decisions or predictions have been made (Molnar (2022)).

For models with impact on safety and models of phenomena that are not scientifically understood enough, interpretability is very important, as it reassures regulatory entities of the reliability of the models' predictions and it provides researchers with important information to tackle similar problems.

Interpretability may be intrinsic to some models, if by their nature decisions have certain well-defined criteria (as will be seen for Decision Trees in Section 3.3.3, provided they are not too complex). On the other hand, others function more as "black-boxes": inputs are given and outputs received, but the internal workings of the model are not known or interpretable (an example of these are Neural Networks, that will be presented in Section 3.3.2).

Interpretability can also be acquired using *post-hoc* methods. These are used to interpret models after training and can be used both for intrinsically interpretable and black-box models.

An example of these interpretation tools are the SHAP (SHapley Additive exPlanations) values, proposed by Lundberg and Lee (2017). This is a method based on game theory. The SHAP value is the average marginal contribution of a feature value across all possible combinations of features. This tool can be employed for any model (it is model agnostic).

By using this technique, a score for each feature, that is a measure of its importance, is obtained. SHAP values can be useful not only to interpret a model but also to be used in Feature Selection, as will be alluded to in Section 3.5.

3.3 Types of Machine Learning Algorithms

In this Section, the types of ML algorithms that are most commonly used in these kinds of applications will be presented. The models that will be used throughout this work will be focused on.

3.3.1 Gaussian Processes

The so called kernel machines are a popular type of algorithms in ML applications. Gaussian Processes (GP) and Support Vector Machines (SVM) are examples of these kind of kernel machines.

Here the focus will be put on GPs, a principled, practical, probabilistic approach to learning in kernel machines (Williams and Rasmussen (2006)). For SVMs, refer to Platt et al. (1999) or Crammer and Singer (2001).

A GP is a non-parametric supervised learning technique: a generalization of the Gaussian probability distribution, so that instead of random variables, functions are dealt with, as is stated by Williams and Rasmussen (2006).

Non-parametric models differ from parametric statistical models, where a function class that includes a finite number of parameters is used. An example of these are Neural Networks (NNs), very flexible parametric functions, that will be presented in Section 3.3.2. Learning from observations means to modify the parameters in such a way that the error decreases. If the user is very well informed about the problem, physically motivated parametric functions may also be chosen, and its parameters fitted for a certain problem. But if many aspects of the phenomenon are unknown or hard to describe explicitly, non-parametric modelling can be more versatile and powerful (Seeger (2004)).

In GPs, what is assumed is the prior set of functions that will be used to model the problem. In this type of algorithm, the ensemble distribution of these functions is a Gaussian distribution (Duvenaud (2014)). An illustration of these prior functions for an univariate scenario is seen in Figure 3.2.

The GP is fully defined by its mean and covariance functions. It is common to assume that the mean is zero in all domain. Uncertainty about the mean can be specified as an extra term of the covariance function, and, therefore, by doing this, the GP can be fully defined by its covariance, also called kernel. The kernel determines how the model generalizes, or extrapolates to new data (Duvenaud (2014)). Different kernels may be chosen. The specific kernel, loosely speaking, defines which functions are more likely to approximate the final model, from all of the prior GP set of functions.

Given a set of observations (pairs of features and labels), only the functions that pass through those points are selected and kept in the model. This is illustrated in Figure 3.3. It is possible to choose the functions that pass exactly in those points or, with more freedom, those that pass close to those points. For this, a closeness threshold is defined using a parameter α. This accounts for noise in the data as can be seen in Figure 3.4.

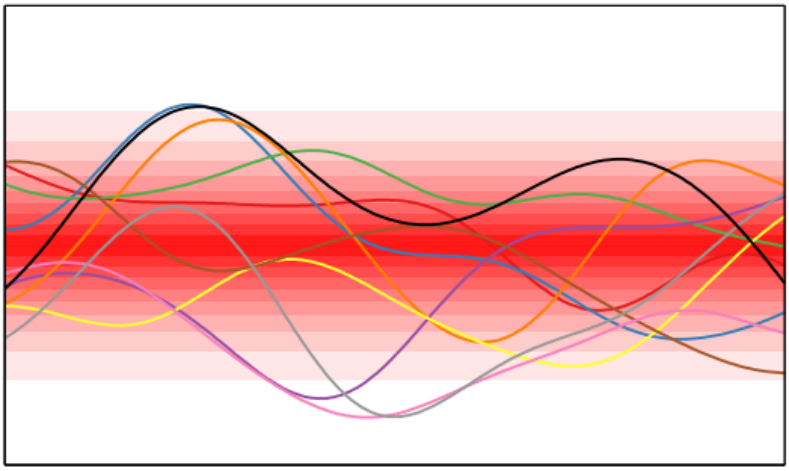


Figure 3.2: GP prior functions, for a univariate regression problem (Duvenaud (2014)).

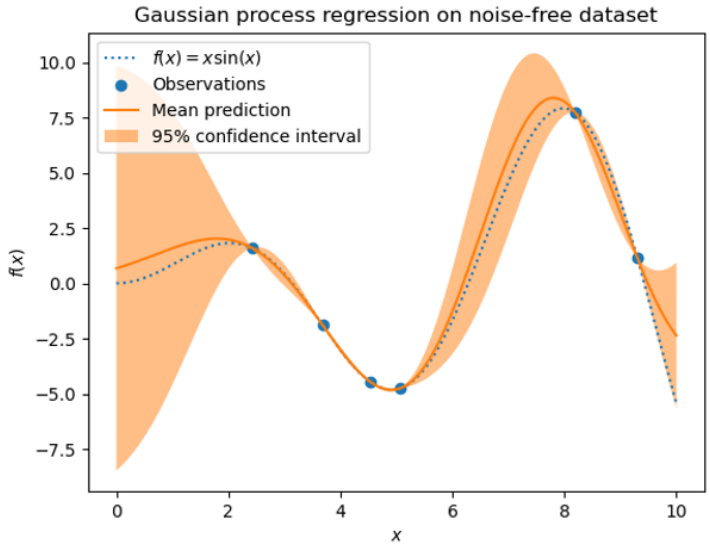


Figure 3.3: GP functions after fitting observations (Pedregosa et al. (2011)).

For classification problems, the GPs are implemented as probabilistic classifiers, predicting the probability of the data-point being classified as a given class. In these problems, the prior functions are Gaussian but the posterior functions are not, as these would not be suitable for probabilistic classification: Gaussian likelihood is inappropriate (Williams and Rasmussen (2006)).

The most popular types of kernels include Radial-Based Function (RBF) kernels, also known as "squared exponential" kernel, a stationary kernel (meaning that it is invariant upon translations of the data-points) that is parameterized by a length-scale parameter, l, that must be defined; and Mátern kernels, a generalization of the RBF kernels, with an additional parameter

ν, that controls the smoothness of the resulting function. Other kernels include constant kernels; rational quadratic kernels; exp-Sine-Squared kernels and dot-Product kernels (Pedregosa et al. (2011)).

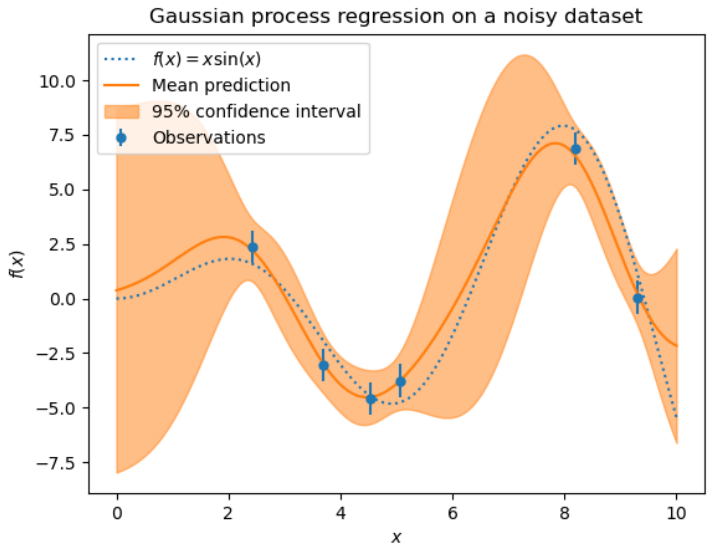


Figure 3.4: GP functions after fitting observations, accounting for noisy data (Pedregosa et al. (2011)).

Pedregosa et al. (2011) presents the foundation of the library `SKLearn`, in whose User Guide some advantages of GPs are shown.

These advantages include the fact that the models' predictions interpolate the observations; that the prediction is probabilistic and therefore allows for the computation of confidence intervals on the predictions; and the versatility of the approach, as different and even costum kernels can be applied.

On the other hand, the main disadvantage of GPs is the fact that they lose efficiency in high dimensional spaces, that is, for data with too many features.

The hyperparameters, that is, the algorithmic parameters that are not updated during training, that can be tuned to build a certain GP are:

- `alpha`: parameter α, that accounts for noisy data.
- `kernel`: type of kernel to be used.

3.3.2 Neural Networks

Neural Networks (NN) are perhaps the most popular and well known ML algorithms. As stated by Hong (2023), these networks are inspired by biological brains, that consist, in a very simplified way, of millions of interconnected neurons, that activate each other through electrochemical pulses. Analogously, Neural Networks consist on nodes (that emulate the neurons) and connections between them.

There are different types of Neural Network algorithms, namely Recurrent Neural Networks (RNNs); Convolutional Neural Networks (CNNs) and Fully Connected Neural Networks (FCNNs).

RNNs are a type of NN that uses sequential or time series data: these are commonly used on problems such as language translation, natural language processing and speech recognition. They are distinguished from other typer of NNs by their memory, as they are able to use information from previous inputs to influence the current input and output (IBM (2024b)).

CNNs use convolution layers, which by using filters, are able to perceive features and patterns from data with many dimensions. These are therefore distinguished from other NNs by their superior performance with image, speech or audio processing and recognition (IBM (2024a)).

FCNNs are the ones that will be used in this work. These are composed by a series of layers, each of them with a number of nodes. Each node at a given layer is connected, and therefore influenced by, all nodes from the previous layers, and is also connected, and therefore able to influence, all neurons on the following layer (hence the designation "Fully Connected Neural Network"). These structures have an input layer, consisting of the values of the features, an output layer, that gives the predicted value for the labels, and several hidden layers in between these. In Figure 3.5, a FCNN with three nodes (three features) in the input layer; two hidden layers with four nodes each; and a single output node is shown as an illustration.

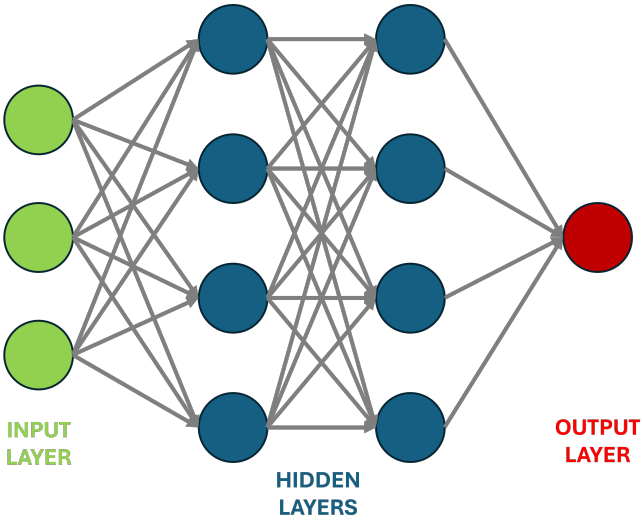


Figure 3.5: Illustration of a Fully Connected Neural Network.

A node in a given layer (with exception from the input layer) can be seen as a function that has as arguments N values x_i corresponding to the values of the nodes from the previous layer, that are connected to it, and gives as output a value z. To each x_i corresponds a weight w_i in the node function. Furthermore, each node has a bias parameter b. This function has the form shown in Equation (3.1),

$$z = f\left(\sum_{i}^{N}(w_i x_i) + b\right) \tag{3.1}$$

where $f(\bullet)$ is termed an activation function. The activation function is what allows the model to capture non-linear relations. If the activation function were a linear function (for example the identity function), the FCNN would only be able to learn linear patterns. The most common choices for activation functions are the $ReLU(\bullet)$ (Rectified Linear Unit) function, presented in Equation (3.2), that although apparently being a very simple function allows in general for the introduction of sufficient non-linearity; the hyperbolic tangent, $tanh(\bullet)$, as seen in Equation (3.3); and the sigmoid, $sig(\bullet)$ function, as seen in Equation (3.4) (Keras (2024), Brownlee (2024)).

$$ReLU(x) = max(0, x) \tag{3.2}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.3}$$

$$sig(x) = \frac{1}{1 + e^{-x}} \tag{3.4}$$

The identity function is also commonly used as the activation function for the output nodes in regression problems. In binary classification problems, the most commonly used activation function for output nodes is the sigmoid function, as it reduces the values to a range between 0 and 1, representing the predicted probability of one of the classes being the correct label (Brownlee (2024)).

The set of all w_i and b of all nodes are the trainable parameters of the Network. Their training consists essentially in an optimization problem: first, these trainable parameters are initialized arbitrarily; then the model receives data from the training set through the input layer, and the Network yields a prediction in its output layer; through a loss function, a measure of error is computed by comparing this prediction with the true label; by using an optimizer, the Network updates its weights and biases, with the objective of minimizing the loss function. The described process is termed an iteration, or epoch. Note that it is also a common approach to update the trainable parameters more than one time in an epoch, by feeding the Network multiple batches of data, and not the whole training dataset at once. The process is repeated until a stopping condition is met, such as reaching a limit number of iterations (a limit too low may lead to under-fitting and a limit to high may lead to over-fitting).

To prevent over-fitting, regularization strategies are employed. These include many possibilities, such as early stopping; L1 and L2 regularization, which are strategies that penalize large weight values; and dropout methods, which work by assigning a probability of ignoring ("dropping") certain nodes of the network (Bala (2023)).

The advantages of a FCNN model, henceforth termed NN model, are, as described by Pedregosa et al. (2011):

- Simplicity and flexibility to a broad spectrum of problems.
- Capability to learn non-linear models.

Some of its disadvantages, on the other hand, are:

- Different weight and bias initialization can lead to different validation scores.

- Large number of of hyperparameters.
- Sensitivity to feature scaling.

The most important tunable hyperparameters, when building a NN are then, following the terminology from `SKLearn`:

- `hidden_layers`: defines the number of hidden layers and the number of nodes in each of them. Represented by an array with with N integer values. N corresponds to the number of layers and each value to the number of nodes in a layer. For example, to a model with two hidden layers, one with 16 nodes and the other with 8 nodes would correspond an array (16,8).
- `alpha`: strength of an L2 regularization term.
- `batch_size`: size of the batches of data (number of data-points) fed to the FCNN iteratively during each epoch.
- `max_iter`: maximum number of epochs.
- `activation`: activation function.
- `optimizer`: optimizer used. Implemented optimizers in `SKLearn` are the 'adam' optimizer, the 'lbfgs' optimizer and the 'sgd' approach (Pedregosa et al. (2011)). A given optimizer[1] may have additional hyperparameters: for example for the 'adam' and 'sgd' optimizers, an initial learning rate (`learning_rate_init`) must be defined.

3.3.3 Decision Trees

Decision Trees (DT) are a supervised learning method used both for classification and for regression, that learn simple rules based on features inferred from the feature/label pairs on the training dataset to create a model that predicts a target class or a continuous value (Vermeulen (2020)). An illustration of a simple decision tree may be seen in Figure 3.6.

These are one of the most popular methods in ML when it comes to structured datasets. A DT consists of three types of elements, namely nodes; edges and leaf nodes (Alkhalifa (2022)).

A node consists in essence on a condition to which one or more features of the data-point will be subjected to. A node usually then splits in two edges (one for if the condition is met and the other if that isn't the case). These edges lead to other nodes, increasing the depth of the tree. If a node is terminal, it is called a leaf node. In classification problems, a specific class is assigned to each leaf node, and in regression problems to each node corresponds a value.

A DT may be classified as a multi-variate decision tree, if at each split the imposed condition can depend on more than one feature (Kotsiantis (2013)).

An explanation on the application of DTs on classification problems will first be given, based on Breiman et al. (1984). For a classification problem with N classes, each node holds

[1] For more information on the 'adam' optimizer, refer to Kingma and Ba (2014). The 'sgd' optimizer is a classical Stochastic Gradient Descent. (Amari (1993)). 'lbfgs' is a Limited-memory Broyden-Fletcher-Goldfarb-Shanno implementation, an algorithm from the family of the quasi-newton methods (Liu and Nocedal (1989)).

the probability of the data-point under evaluation being labeled as a certain class. The first node, n_1, holds the probabilities $P_1 = [\frac{1}{N}, \frac{1}{N}, ..., \frac{1}{N}]$, where each element of P_1, termed P_1^i, $i = 1$ to N, is the probability of the data-point belonging to class i. A leaf node, n_k, should be one that contains a very high value of P_k^j for a certain class j, and very low values for the others. If this is the case, to n_k, class j will be attributed. Note that for a given node n_k, Equation (3.5) holds.

$$\sum_{i=1}^{N} P_k^i = 1 \tag{3.5}$$

Ideally, at each node, the most informative splitting condition will be used. Most informative may be defined in many ways. One, according to Breiman et al. (1984), is for the split to lead to the less impure subsequent nodes possible. To evaluate this, an impurity function is defined: it should have its maximum value for the first node, where all probabilities are the same, and be 0 for a node in which all $P_k^i = 0$ for all i except one (this is the purest possible node as it contains a probability of 1 for one class and 0 for all the others).

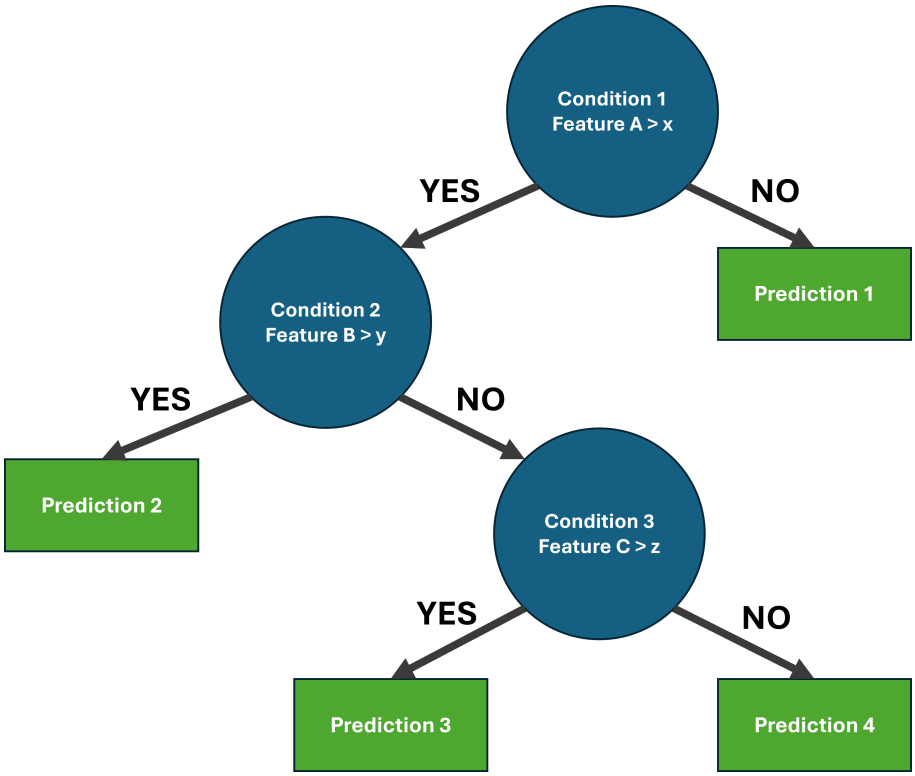


Figure 3.6: Generic Decision Tree illustration. Nodes are in blue; leaf nodes in green and edges in black.

To determine the conditions for each split (the features used in them and the decision threshold) and at which nodes to make these, the main condition is that the impurity of the consecutive nodes should decrease optimally.

Two measures for this decrease are the Information Gain (*IG*), based on entropy, and the

Gini index. These are implemented in the `SKLearn` library, and one of them must be chosen when building a model.

According to Alkhalifa (2022), *IG*, is the change in informational entropy, H. Informational entropy is usually defined as presented in Equation (3.6), classically proposed by Shannon (1948), and *IG*, for a given possible split in node n_k, is defined in Equation (3.7), following Alkhalifa (2022),

$$H_k = -\sum_{i=1}^{N} P_k^i \cdot log(P_k^i) \tag{3.6}$$

$$IG_k = H_k - H_{k_a} \tag{3.7}$$

where H_k is the entropy on the probabilities of the class variables before the split being evaluated and H_{k_a} after the split.

By using this measure, the model minimizes the information needed for data classification at each split.

The Gini index criterion, *GINI*, on the other hand, measures how often a randomly selected data-point would be incorrectly labeled. It is computed using Equation (3.8), by subtracting the sum of the squares of the probabilities for each class, for a given node n_k.

$$GINI_k = 1 - \sum_{i=1}^{N} (P_k^i)^2 \tag{3.8}$$

To terminate the growing of a tree, an heuristic rule may be applied. For example, it may stop when no further splits are found that significantly decrease node impurity (Breiman et al. (1984)), or when a limit depth, number of nodes or number of leaves of a tree is achieved (Kotsiantis (2013)). If trees are too small, they will not be able to capture the influence of features on the target class, and, on the other hand, if the trees are too large, over-fitting problems on the training data will appear, and generalization capabilities will be lost (Pedregosa et al. (2011)). Therefore, tree size should be a tunable parameter, depending on the data, and it governs the model's complexity, as stated by Hastie et al. (2009).

To decrease the DTs' complexity, it is common to perform the pruning of the tree. This process aims to generate a sub-tree from the original finalized DT, that has greater generalization capabilities (Kotsiantis (2013)).

DTs may also be used very similarly for regression problems, the only difference being that the models will now output floating point values. As is seen in Figure 3.7, the models are therefore not expected to have smooth or continuous outputs. The problem of over-fitting is also clearly seen in the Figure, as the larger tree, with maximum depth of 5, is fitting the training data at some points that do not follow the general trend, and therefore generalization capabilities are hindered.

Some of the main advantages of DTs, as presented by Pedregosa et al. (2011), include:

- Their simplicity and interpretability, as the logical rules may be explicitly accessed.
- Their requirement for little data preparation. Data normalization is not necessary and they even support missing values for some features.

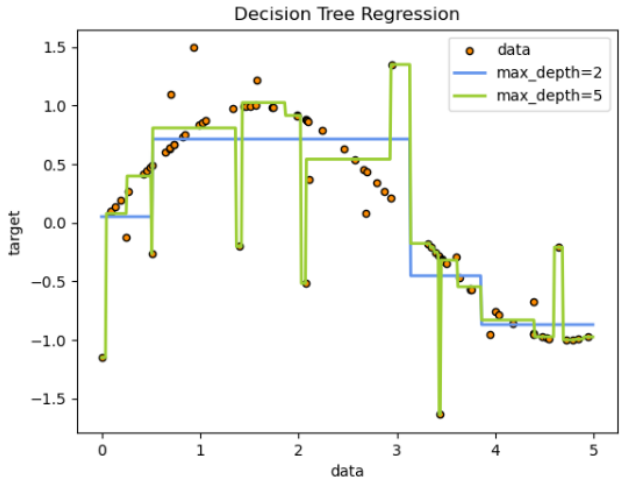


Figure 3.7: Decision Tree used in a regression problem (Alkhalifa (2022)).

- The computational cost of making a prediction is logarithmic with increased number of data points used to train the tree.
- Their ability to inherently handle Multi-Output problems.
- Their "white-box" nature, by which one can track all the decisions made by the tree to reach a prediction.

Some important disadvantages, though, are:

- Tendency for over-fitting for too large or complex DTs.
- Tendency for instability, that is, small variations on training data may lead to completely different DTs.
- Predictions of DTs are not continous nor smooth. Therefore they are not well suited for extrapolation.
- The problem of optimizing a DT has to be solved by heuristic methods, for it to be solved in a reasonable time-frame.
- DTs have difficulty learning some concepts, such as XOR, because it is not easily expressable by the nodal conditions.
- Tendency for creating biased trees when dealing with unbalanced datasets in classification problems.
- Tendency for over-fitting when dealing with data with too many features.

The most important tunable hyperparameters, when building a DT, are: the splitting criterion (Gini or Entropy based); maximum depth of the tree; minimum number of data samples used to create a split; maximum number of leaf nodes; minimum impurity decrease (to decide when to terminate the growing process).

3.3.4 Ensembles

To mitigate many of the disadvantages referred in Section 3.3.3, DTs are usually employed in ensembles, that is, collections of DTs that output a single value following a given algorithm. Most popular ensembles are Random Forests (RF); ensembles that use Bagging Methods (BM); and Gradient-Boosted Trees (GBT). The ensemble approach can be summarized, in general, as a combination of a large number of simple models to obtain a stronger final prediction (Natekin and Knoll (2013)).

A disadvantage of the ensemble approach is that interpretability is hindered, because although the logical rules are still accessible for each individual DT of the ensemble, there now may be hundreds of different DTs.

In these ensembles, the final prediction is obtained by using a simple combination of the weak decision tree learners, such as a voting system, in a classification problem as seen in Figure 3.8, or by averaging the weak predictions, in a regression problem, as stated by Natekin and Knoll (2013).

Random Forests

In RF algorithms, each tree is initialized randomly, and trained on different random subsets of data. In some implementations, each tree is even given only a subset of the features to use in the splits. These methods help in reducing over-fitting on the training data (Alkhalifa (2022)).

The RF algorithm has shown excellent performance in ML problems where the number of variables is much larger than the number of observations. Moreover, it is versatile enough to be applied to large-scale problems and returns measures of feature importance, as these can be retrieved from the individual decision trees, as stated by Biau and Scornet (2016).

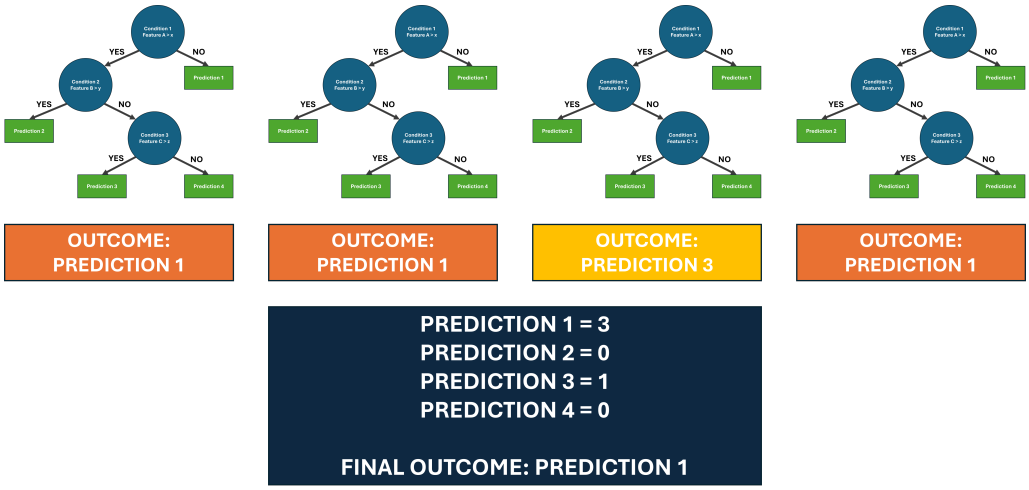


Figure 3.8: Ensemble of Decision Trees voting system illustration. Individual trees are different from each other, in terms of depth, splits, data with which they were trained, and therefore may yield different results on the same data-point.

Bagging Methods

Another ensemble method is the BM. It gets its name from "**b**ootstrap **agg**regat**ing**" (Breiman (1996)). Here, instead of growing random trees, the ensemble is composed of variations of the same tree, grown using different boot-strapped samples of the data, that is, samples taken from the original training set, with replacement (Breiman (1996), Hastie et al. (2009)).

First, a tree is grown, using the entirety of the training data. This tree is then pruned, by using the boot-straped samples of the training set, to create N sub-trees, that are used as the ensemble for estimation (Breiman (1996)).

Bagging has been shown to greatly improve performance of models. It is especially useful because of the instability of DTs (unstable individually because variations on training data may lead to dramatic changes in the splits chosen when growing the tree), as stated by Breiman (1996).

The final result of the ensemble is obtained similarly to what is done with RFs, by averaging or voting techniques.

- `n_estimators`: number of DTs in the ensemble.
- `max_samples`: number of samples to draw from the training set to train each DT, with replacement.
- `max_features`: fraction of features to draw from the total set of features to train each DT.

Gradient Boosted Trees

Another very popular ensemble method is GBT. This is one of the most powerful learning ideas introduced in the 1990s. Although it resembles other ensemble methods, such as RFs and BMs, it is fundamentally different (Hastie et al. (2009)). This is because, in this strategy, the weak learners are added to the ensemble sequentially, in a constructive way (Natekin and Knoll (2013)). The main idea behind this algorithm is to construct new learners that are maximally correlated with the negative gradient of the loss function from the previous model.

There are many strategies to avoid over-fitting in GBTs. Subsampling is a usually employed method, where at each iteration (that is, upon creating a new decision tree), only a sample of the training data is used for growing the DT; shrinkage, which is a type of regularization that gives incrementally less importance to each new iteration, with the objective of taking increasingly small steps to optimization, so that a potential erroneous learner, or one with over-fitting tendency, has as little impact as possible in the whole ensemble; and early stopping, which, by choosing a criterion, such as maximum number of iterations, stops the process of learning before over-fitting (Natekin and Knoll (2013)).

A very interesting implementation is the approach from Ke et al. (2017): the LightGBM.

The two main differentiating features that differentiate this implementation from other GBTs are the *Gradient-based One-Side Sampling*, a technique for choosing the more important data instances for information gain estimation, excluding data-points with low gradient contribution; and *Exclusive Feature Bundling*, a technique to reduce the number of features used within the algorithm, using a greedy algorithm (one that does not intend to find the best solution, but aims at finishing the task in a reasonable number of steps by making the locally optimal choice at each iteration). According to Ke et al. (2017), this approach speeds up the

learning process, when compared to other GBTs, and therefore this is a very efficient and scalable implementation. The tunable hyperparameters of LightGBM are the following:

- `colsample_bytree`: fraction of features to draw from the total set of features to train each DT.
- `min_child_samples`: minimum number of data needed to give a leaf node.
- `learning_rate`: gradient boosting learning rate.
- `max_depth`: maximum depth of the DTs. (-1 means unlimited maximum depth).
- `num_leaves`: maximum number of leaves in the DTs.
- `subsample`: number of samples to draw from the training set to train each DT, with replacement.
- `n_estimators`: number of DTs in the ensemble.

3.3.5 Summary

The types of models that will be used throughout this work, together with their more relevant hyperparameters, are summarized in Table 3.1. When building a model, the hyperparameters that are not here presented will be left always to their default value.

Table 3.1: Types of ML models.

Model Type Designation	Model Abbreviation	Tunable Hyperparameters
Gaussian Processes	GP	`kernel`[2]
Neural Networks	NN	`alpha`
		`batch_size`
		`hidden_layers`
		`activation`
		`learning_rate_init`[3]
		`max_iter`
		`optimizer`
Decision Tree ensemble: Gradient Boosting, LightGBM	LGBM	`colsample_bytree`
		`subsample`
		`min_child_samples`
		`learning_rate`
		`max_depth`
		`num_leaves`
		`n_estimators`
Decision Tree ensemble: Bagging Method	BM	`max_features`
		`max_samples`
		`n_estimators`

[2] Type and additional parameters. `alpha` will always be fixed at 0.01.
[3] Only for 'sgd' and 'adam' optimizers.

3.4 Outlier detection

An outlier is defined by Pedregosa et al. (2011), in simple terms, as an observation, or a data-point in a given dataset, that is far away from the others. The detection of these data points is of paramount importance in ML, because these extreme data-points can have a disproportionate influence on the performance of the models, as mentioned by Orr et al. (1991).

In some cases, outliers may contain important information, as is in the case of ML models for fraud detection or public health, but on other cases the outliers are merely unwanted noise in datasets or erroneous results from data generation. In whichever case, outlier detection is desirable (Yang et al. (2019)).

The most simple strategy for univariate outlier detection is defining a metric for closeness to the main distribution and then a threshold, a limit from which a data-point would be considered an outlier. The most widely used metrics are the Standard Deviation (SD), Median Absolute Deviation (MAD) and Inter-Quartile Range (IQR), as presented by Yang et al. (2019).

3.4.1 Standard Deviation metric

The use of the SD metric is the oldest of the approaches. It consists on assuming that the data follows a normal distribution: a mean value and a standard deviation can be computed from it. The SD is the metric for closeness, and the most common threshold is 3 SDs from the mean value. This implies an outlier level of 0.13%.

The metric can also be seen as the Z-Score, that is computed by using Equation (3.9). The expression transforms the normally distributed variable X to a standardized distributed variable Z, by using the mean, μ, and standard deviation, SD, of X. These can be computed, for a given sample of N data-points x_i, as seen in Equations (3.10) and (3.11).

Z distribution always has a mean value of 0 and a SD value of 1.

$$Z = \frac{X - \mu}{SD} \tag{3.9}$$

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{3.10}$$

$$SD = \sqrt{\frac{1}{N-1} \cdot \sum_{i=1}^{N} (x_i - \mu)^2} \tag{3.11}$$

Other authors, as Miller (1991), propose using even more aggressive thresholds, such as 2.5 or 2, implying outlier levels of 0.62% and 2.28%, respectively, as seen in Figure 3.9 for a standardized normal distribution. Leys et al. (2013) argue that the SD metric is not appropriate for outlier detection even in moderately complex situations. The three issues that the authors point to are the assumption of a normal distribution; the fact that mean and SD are themselves very significantly affected by outliers; and that the method is not capable of finding outliers in small samples.

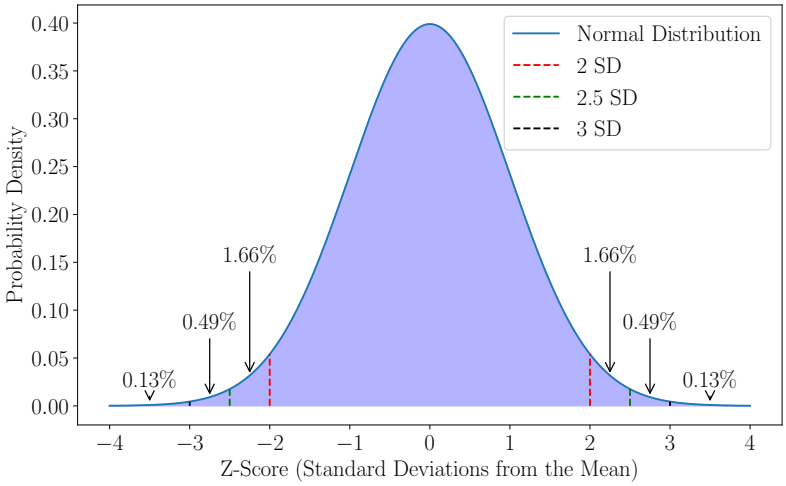


Figure 3.9: Fractions of data outside Standard Deviation limits.

3.4.2 Median Absolute Deviation metric

As an alternative, Leys et al. (2013) propose the use of the median and MAD. The median, as a metric of centrality, is not influenced by very far away outliers. Together with it, the MAD is used as the metric for closeness to the central distribution. MAD is defined as the median of the absolute deviations from the median, as seen in Equation (3.12),

$$MAD = b \cdot M_i(|x_i - M_j(x_j)|) \tag{3.12}$$

where x_j is the set of the N original observations and M_i is the median of the series. The value for the constant b is usually taken as $b = 1.4826$. This constant is linked to the assumption of normality of the data, disregarding the skewness induced by outliers (Rousseeuw and Croux (1993)).

This is a much more robust method, as the median and MAD are not easily affected by the outlier values, and Leys et al. (2013) propose a threshold of 2.5 as a reasonable choice.

3.4.3 Inter-Quartile Range metric

Another option is the use of the IQR metric (Yang et al. (2019)). IQR is computed as seen in Equation (3.13). Here the top (T_{max}) and bottom (T_{min}) thresholds are defined as seen in Equation (3.14),

$$\text{IQR} = \text{Q3} - \text{Q1} \tag{3.13}$$

$$\text{T}_{\text{min}} = \text{Q1} - c \cdot \text{IQR};\ \text{T}_{\text{max}} = \text{Q3} + c \cdot \text{IQR} \tag{3.14}$$

where $Q1$ and $Q3$ are the first and third quartiles, and c is defined by the user, usually set to 1.5. IQR can be affected by the outliers themselves, especially if the dataset has many outliers.

3.4.4 Other approaches

Yang et al. (2019) present other, more complex approaches. In the first, termed Clever SD technique, outliers are removed sequentially, and the mean and SD are recalculated as each of them is taken out. The method seems to work well, but tends to remove some normal points instead of only the outliers.

Another method is to perform a Two-Stage thresholding (2T). This technique resembles Clever SD, but it is not as extreme and here any closeness measure may be used. In the first step (cleaning step), any method may be used as previously presented (SD, MAD or IQR). Then, in the second step (threshold calculation), the threshold is recalculated with all data except for the data identified as outlier in the first step. Outliers are again identified and this iterative process can go on, although Yang et al. (2019) found that using only two stages provides good results and is computationally more efficient.

3.4.5 Summary

The main outlier detection methods presented in this Section have been summarized in Table 3.2.

Table 3.2: Types of outlier detection.

Method Designation	Method Abbreviation	Common Thresholds
Standard Deviation Method	SD	2; 2.5; 3
Median Absolute Deviation Method	MAD	2.5
Inter-Quartile Range Method	IQR	1.5
Clever Standard Deviation Method	Clever SD	2; 2.5; 3
Two-Stage Thresholding Method	2T	-[4]

3.5 Feature Selection Strategies

In the context of ML, Feature Selection (FS) is important, as it reduces data dimensionality and enhances model performance (Dhal and Azad (2022)). The fact that a model is high dimensional, meaning that it uses many features for prediction, significantly increases computational and storage complexity. Furthermore, not all features are essential, and some irrelevant or redundant features can downgrade the model's performance. Redundant features are those that exhibit close dependency to other features and are therefore not necessary, and irrelevant features are those that do not correlate or weakly correlate with the target variable.

[4] In this case, thresholds to be use depend on the metric. SD, MAD and IQR metrics can be used.

FS differs from Feature Extraction (FE) (Guyon and Elisseeff (2006)), that will be performed in Section 4.3: FS is the process of finding a better performing subset of features from the original set, whereas FE is the process of extracting a new feature subset from the original set. Although different, both aim at increasing model's performance and decreasing computational costs. FS and FE can and should be used together (Dhal and Azad (2022)).

There are many ways to perform FS. These are commonly classified in literature as Filter, Wrapper and Embedded methods (Dhal and Azad (2022), Venkatesh and Anuradha (2019), Vergara and Estévez (2014)). These will be presented here.

It is important to note that before these methods are used, it is crucial to perform the removal of features that do not vary at all or have very low variance (Pedregosa et al. (2011)). If a given feature is always the same, for all data-points, then it is irrelevant, as the model will not be able to extract any information from it.

3.5.1 Filter methods

Filter methods are the most simple and computationally efficient methods. These rely on attributing a simple correlation metric to each of the features, in relation to the label. A threshold for correlation is defined and features that do not surpass that threshold are eliminated (Vergara and Estévez (2014)). Common statistical metrics are the Pearson coefficient and the Mutual Information metric.

Filter methods assume independence between the ML process and the data, and therefore use a metric independent of the learning algorithm to assess feature importance. Vergara and Estévez (2014) state that filter methods are relatively robust against over-fitting, but may fail to select the best feature subset for the classification or regression model.

Pearson's coefficient, ρ, is a measure of correlation that only relates two quantities. Furthermore, it only detects linear relationships. This measure varies in the range [-1,1]: $\rho = -1$ means that the two quantities are perfectly linearly and negatively correlated; $\rho = 1$ means that they are perfectly linearly and positively correlated; and $\rho = 0$ means that no correlation is detected. The coefficient is computed as presented in Equation (3.15) (Venkatesh and Anuradha (2019)),

$$\rho(x,y) = \frac{\sum_i (x_i - \bar{x})\left(y_i - \bar{y}\right)}{\sqrt{\sum_i (x_i - \bar{x})^2 \left(y_i - \bar{y}\right)^2}}. \tag{3.15}$$

where x_i are the values of the independent variable (feature) and y_i are the values of the dependent variable (label) for each data-point. $\bar{x}$ and $\bar{y}$ are the mean values of x and y.

Mutual Information, *MI*, is another measure that may be used for FS. It is the measure of how two variables (X, Y) are mutually dependent (Venkatesh and Anuradha (2019)). It has two main properties: it can measure any kind of relationship between random variables, not only linear dependence; and is invariant under transformations in the feature space that are invertible and differentiable, as translations or rotations (Vergara and Estévez (2014)).

MI, for two variables X and Y, is computed as shown in Equation (3.16) for discrete variables, and as shown in Equation (3.17) for continuous variables, as shown by Venkatesh and Anuradha (2019),

$$MI(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{3.16}$$

$$MI(X,Y) = \int_X \int_Y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) dx\, dy. \tag{3.17}$$

where $p(x,y)$ is the joint probability function of X and Y, and $p(x)$ and $p(y)$ are the probability distribution functions of X and Y, respectively.

3.5.2 Wrapper methods

Wrappers, as stated by Vergara and Estévez (2014), use a learning algorithm as part of the function for evaluating feature importance. Different approaches may be followed, and either black-box or interpretable models may be used.

In the methods that use black-box models, different subsets of the original set of features are iteratively used to train a chosen ML model that is then evaluated by using a test set or Cross-Validation scores (Cross-Validation will be detailed in Section 3.6). These methods may start with all features and inductively choose the features to be removed based on model performance or, conversely, start with few features (a user defined minimum) and iteratively choose features to add.

Methods that use interpretable models work in a similar fashion but make use of importance metrics for each feature provided by the model itself instead of performance scores. For example, for DT models or DT ensembles, shown in Section 3.3, features that are used more times in splits or that achieve greater informational gain are given more importance and therefore are kept in the final set of features selected via FS.

A popular Wrapper model for FS is Recursive Feature Elimination (RFE). This method is implemented in `SKLearn` (Pedregosa et al. (2011)). The model starts with the original set of features and iteratively removes the ones that are considered less important by a given estimator that assigns importance to features. First this estimator is trained using all data and all features. The features that are considered less important are then removed and the model is trained again without them. The process repeats itself until a desired number of features is achieved.

It is also possible to use a Recursive Feature Elimination with Cross-Validation (RFECV). This method works by performing RFE in a loop of K-fold Cross-Validation (that will be described in Section 3.6.2) to find the optimal number of features. This method provides the user both with the ideal number of features and with the ideal features: the best after averaging all Cross-Validation folds. This approach is also implemented in `SKLearn` (Pedregosa et al. (2011)). In this case, the user defines the scoring function to evaluate model performance.

Yan and Zhang (2015) presented an application where RFE was used with SVMs in gas sensor data, and it is shown that it over-performs other common FS methods.

Awad and Fraihat (2023) presents an application of the RFECV method, with 10-fold Stratified Cross-Validation, in the area of intrusion detection systems, a classification problem. The selected features were used to train several ML models, including RFs and NNs, and it was shown that this FS method outperformed the models trained using the entirety of the initial set of features and features selected by other FS methods.

Wang et al. (2023) apply different FS methods in the area of Perovskite materials design and discovery. It is argued that SHAP values, presented in Section 3.2.6, can be used in conjunction with black-box models and wrapper methods, as they make the black-box models interpretable.

It is clear, then, that Wrapper methods could achieve higher performance and tend to have necessity of smaller dataset sizes, when compared to filter methods, but lead to more computational complexity. This is because the ML model chosen for evaluation may have to be fitted to data and evaluated several times (Cai et al. (2018)).

3.5.3 Embedded models

Embedded models are those that instead of performing FS at the pre-processing level, as the ones presented before, perform FS during the learning process. This approach aims to improve computational costs. For more details in embedded methods, refer to Lal et al. (2006).

Although interesting, these methods are computationally more costly than Filter methods and less accurate, in general, than Wrapper methods (Venkatesh and Anuradha (2019)).

3.5.4 Summary

In Table 3.3, the most popular FS strategies, implemented for the pre-processing of the ML models, are presented.

Table 3.3: Feature Selection strategies.

Feature Selection Method	Method Abbreviation	Type of Method
Pearson's Coefficient	PC	Filter
Mutual Info	MI	Filter
Recursive Feature Elimination	RFE	Wrapper
Recursive Feature Elimination with Cross-Validation	RFECV	Wrapper

3.6 Data splitting

As will be detailed, in an ML problem, where data is usually limited, the best practice is to split that which is available in different sets, to be used for different purposes. Not all data can be used for training because, to test generalization capabilities and to check for over and under fitting, the model has to be tested on data that it has not seen during training. Furthermore, it is common to make more partitions, for validation purposes, when choosing the best ML algorithm and its hyperparameters. All of these sets should be a representative sample of the entirety of the population.

3.6.1 Training, Validation and Test splits

If the situation is rich in data, a viable approach for splitting data is using three different sets (Hastie et al. (2009)):

- Training set: data that the models will use for learning.
- Validation set: data used to evaluate trained models and select the best hyperparameters.
- Test set: data used to evaluate the generalization capabilities of the model with the chosen hyperparameters.

It is important to note that the validation step is important because hyperparameter selection should be seen as an integral part of the learning process. If the hyperparameters are tuned by evaluating the models directly on the test set, there is the risk of over-fitting on the test set. This means that, as model selection would have been done on the test set, this set can no longer accurately be used to evaluate generalization capabilities of the model.

This approach is illustrated in Figure 3.10. The test set should be kept completely apart during the construction of the model and be brought out only at the end, for evaluation purposes (Hastie et al. (2009)).

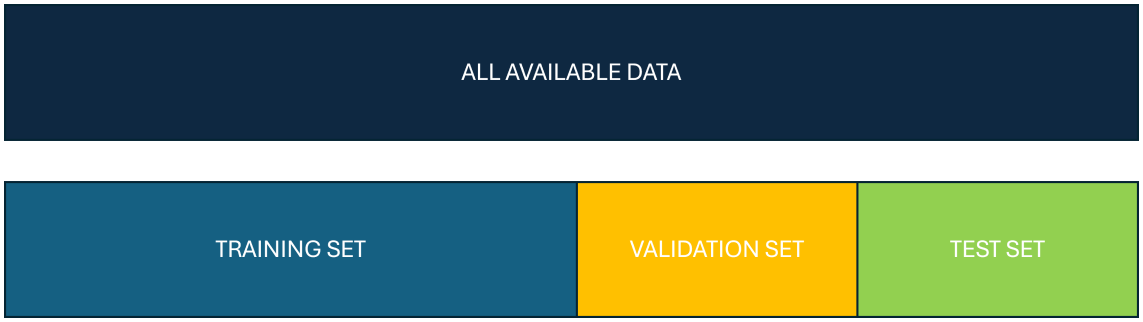


Figure 3.10: Train, Validation and Test sets illustration.

This method is usually not employed in situations similar to those of this work, as data is scarce and computationally expensive. There are other ways of splitting and using data, that make a more efficient use of it.

3.6.2 Cross-Validation

The most popular way of doing model selection is by employing K-fold Cross-Validation. In this method the available data is first shuffled to obtain a random order and avoid biases, and then split into a training/validation set (equivalent to the union of the training and validation sets seen in Figure 3.10) and a test set. The training/validation set is then divided into K roughly equally sized sets, or folds. After this, model training will be performed K times, following this procedure:

1. The K sets are numerated from 1 to K.
2. Set $i = 1$ is selected as the validation set.

3. Model is trained using all K-1 sets that were not selected as the validation set.

4. Validation set is used to evaluate the model using a relevant metric. This evaluation is stored.

5. Set $i+1$ is selected as the validation set. The process goes back to step 3 until all K sets have been used as validation sets.

6. Compute the average of all K evaluations. This is the value that should be used as a measure of model performance in hyperparameter selection.

This method is illustrated in Figure 3.11, for a 5-Fold scenario, where fold $i = 2$ is the currently chosen validation set.

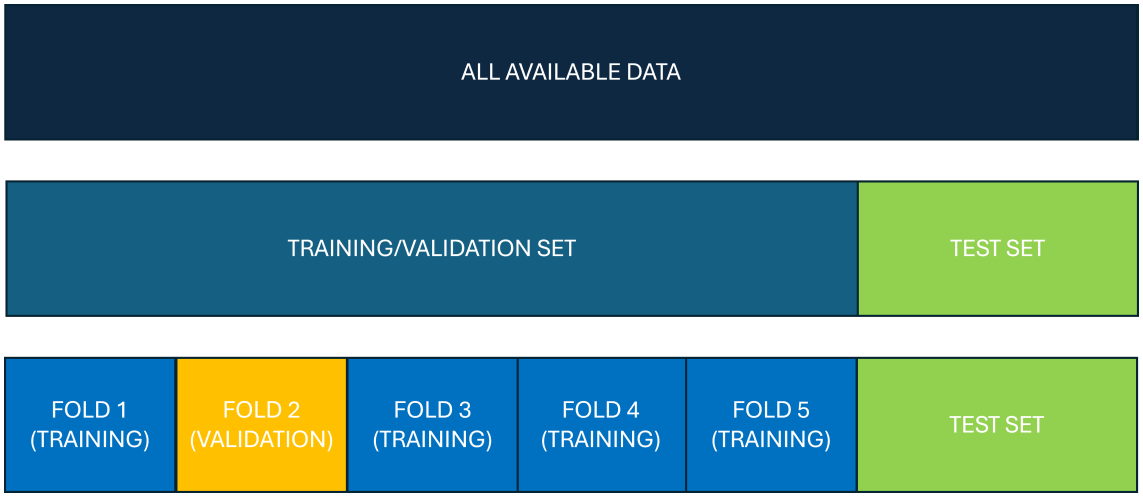


Figure 3.11: K-Fold Cross-Validation strategy illustration.

Typical choices of K are 5 or 10, according to Hastie et al. (2009). The edge case where K= N is termed the *leave-one-out* method. Here, for each fold, all data is used except for one of the data-points. This, of course, results in the largest possible number of fits for each potential model/hyperparameter combination, and is therefore a computationally expensive method.

For model evaluation, a scoring function is used. The scoring function is used for evaluating the performance of a model, and is not an integral part of the training algorithm (shouldn't be confused with the loss function of some ML algorithms).

After selecting the model that is expected to behave the best, all data in the training set should be used to train it, before it being evaluated on the test set.

There are other validation methods, as, for example, the Monte Carlo method, also called boot-strapping. Monte Carlo validation works similarly to K-fold Cross-Validation but here the folds are randomly chosen from the training set, with replacement. This may result in some samples being used for both training and testing data multiple times. Usually in this method, a large number of simulations is performed, and therefore it may be slow and computationally expensive (Ramezan et al. (2019)).

3.6.3 Sampling strategies

Independently of the validation strategy employed, it is necessary to choose the data-points that will be present in each split. There are different methods to choose the splits in the data that were previously described. The most simple way of splitting data is by performing random sampling, both for separating test and training data, and for choosing the folds in Cross-Validation. May et al. (2010) argue that this is the most common practice but only because ML practitioners put more focus on model architecture and learning algorithm, using arbitrary splits that may hinder model's generalization performance and even the capability of accurately evaluating the model. Poor data splitting can result in inaccurate and highly variable model performance.

A common approach to find better splits in the data is to use stratification techniques. Stratified train-test splits and stratified Cross-Validation techniques can be employed.

In classification problems, stratification ensures that all splits contain the same proportion of the target label. This avoids unbalanced sets of data: if the dataset has x% of points labeled as Class A (because, if data is generated or gathered in an ideal way, this is the real proportion of points in this class), it is beneficial that both the training and test set also have x% of the data points labeled as Class A. If this doesn't happen, the model's training will tend to be biased, and the test set will not allow us to accurately predict generalization capabilities of the model. The same reasoning may be used when finding the Cross-Validation splits (Prusty et al. (2022)).

On the other hand, in regression problems, stratification is usually done based on one or more of the features. For this, if the feature being used is a continuous variable, a number of bins to subdivide the data-points is chosen. For example, if 3 bins and Feature X are chosen, each data-point will be classified as having a small, medium or large value of feature X, according to limits set by the user. These labels can now be used for stratification purposes just as explained for classification problems: each split will contain the same proportion of data points with small, medium and large values of Feature X.

This strategy is used, for example, by Sweet et al. (2023) in a climate prediction problem, and it is found that it impacts hyperparameter selection and better performance is registered, depending on the feature that is selected, when compared to random splits.

3.6.4 Repeated Stratified Cross-Validation

It is possible to perform Repeated Cross-Validation and Repeated Stratified Cross-Validation. The difference here is that the fold selection is done multiple times, randomly, resulting in different splits of the data (accounting for the proportions of a given label or feature in the Stratified case). This method increases the robustness of the Cross-Validation, although it is more computationally expensive (Krstajic et al. (2014)).

3.6.5 Other splitting strategies and applications

May et al. (2010) propose a Self Organizing Map (SOM) based stratified sampling. SOMs can be used to perform data clustering, and therefore with this approach it is possible to stratify the data based on more than one of the features. It is shown, by using NNs, that these kinds of methods create more representative datasets and therefore better model performance with

less data for training. It is argued that these kind of methods are the best sampling choice for datasets with high dimensionality and skewed variable distributions.

May et al. (2010) further say that Random Sampling is not a suitable sampling technique for the generation of training, test and validation data samples, in particular, for skewed or non-uniformly distributed data, because it results in poor validation performance.

Merrillees and Du (2021) present some multi-label stratification possibilities for classification problems, in particular problems of Extreme Multi-label (in which data points may have hundreds or thousands of labels).

Farias et al. (2020) propose a complex train-test split strategy, based both on the input and output values by making use of similarity evaluation functions. Their approach outperformed classical random K-fold splitting strategies, using, among others, NN, SVM and RF algorithms, in 75% of the accessed examples. This shows the importance of having representative data splits, but it also shows that the choice of the complexity of the splitting strategy should depend on the data itself, and that in some cases the most simple approach is good enough.

Other methods, such as Stratified Nested Cross-Validation and Double Cross-Validation are also found in the literature (Krstajic et al. (2014)).

Ramezan et al. (2019) applied different sampling methods in an ML model in the area of classification of geographic objects and found that sampling strategies using stratification show statistically relevant improvements to model's performance.

In the biomedical area, Prusty et al. (2022) has applied Stratified K-fold Cross-Validation for the classification and detection of cervical cancer, using several models, including GBTs and RFs, achieving enhanced model performance.

Al-Abdaly et al. (2021) also used stratified splits to separate data into a test set and a training set, in a problem of prediction of steel fiber-reinforced concrete compressive strength using an RF model, achieving good performance.

3.6.6 Summary

In Table 3.4, the data splitting strategies that may be used in this work are presented as a summary. The other approaches, more complex, are not as of yet implemented.

Table 3.4: Data splitting strategies.

Splitting Strategy
Random Sampling
Univariate Stratification

It is worth reiterating one other point presented in this section, which is the strategy of Repeated Stratified Cross-Validation (RSCV), a strategy that further improves the robustness of Cross-Validation techniques, although at some higher computational cost.

3.7 Hyperparameter selection

As was previously alluded to, before a ML model is trained, one or more model parameters that will not be optimized during training, termed hyperparameters, must be set, as is stated by Luo (2016). This selection can greatly influence model performance. Lavesson and Davidsson (2006) even argue that hyperparameter tuning is more important than algorithm selection. In their work, additionally, they find that some algorithms are more sensitive to hyperparameter tuning than others.

The hyperparameters can be set manually, a process that requires user expertise, and is labour intensive, being finished when the user obtains a model with satisfactory accuracy, runs out of time or resources, or believes that the model's accuracy cannot be improved further. (This manual process is also referred to as "Trial and Error" or "Grad Student Descent" (GSD)(Yang and Shami (2020))). An alternative is the use of automatic hyperparameter search methods. Either way, this choice is based on validation (or Cross-Validation) scores or scoring metrics, as detailed in Section 3.6.

3.7.1 Exhaustive Grid Search

An automatic hyperparameter search method that is very popular is the Exhaustive Grid-Search (usually with Cross-Validation) method. Here the user defines an admissible hyperparameter space, based on the literature and on experience, which requires some expertise, that is exhaustively ran through: all hyperparameter combinations are used to build models that are then evaluated. This means that if the parameter space has N possible hyperparameter combinations, and if using K-fold Cross-Validation, the model will have to be fitted $N \cdot K$ times. (If using a repeated CV method with M repetitions, the model would have to be fitted $N \cdot K \cdot M$ times). This method ensures that the model with best validation score in the hyperparameter space will be used, but as it is an exhaustive search, it is computationally expensive (Pedregosa et al. (2011), Yang and Shami (2020)).

3.7.2 Random Grid Search

Another popular method is the Random Grid Search method. It works similarly to the Exhaustive Grid Search, in the sense that the user defines the parameter space, but in this case the space is not exhaustively tested: the user also defines a limit number of fits to the model. The parameter space is therefore sampled randomly and only some of the combinations are tested. The best scoring combination among these is chosen. This is, of course, a much less expensive search method, although the user is not guaranteed to have the best hyperparameters even withing the chosen parameter space (Yang and Shami (2020)).

3.7.3 Successive Halving

Jamieson and Talwalkar (2016) propose a search method based on successive halving. This approach can be seen as a tournament-with-elimination like competition among hyperparameter combination candidates, as illustrated by Pedregosa et al. (2011). It is an iterative process where at each iteration a limited amount of the data resources are used to evaluate all possible candidates. The method works by allocating a data resource budget to a set of hyperparameter

configurations, evaluating the performance of all combinations, and eliminating the worst half, as stated in the work of Li et al. (2018), where the method is further explored. The technique halves the number of possible candidates at each iteration, leading at the end to a single final winner. At each iteration, more data resources are allocated to evaluate the candidates. This approach is less expensive than the Exhaustive Grid Search method and leads to better results than the Random Grid Search method.

3.7.4 Other methods

Gradient-based optimization and bayesian optimization methods (Feurer and Hutter (2019)) may also be used, where classical optimization approaches are used to optimize the hyperparameters themselves, defining a loss function based on the score metric, that is used to approximate the best search direction in the hyperparameter space. Other less conventional optimization methods, as the Genetic Algorithm or Particle Swarm Optimization may also be used (Yang and Shami (2020)).

There are other possible methods found in the literature. Luo (2016) reviews optimization methods that take into account experience with different past ML problems and different datasets that are classified according to specific metrics to be comparable with the problem at hand (Meta-Learners). It also reviews sequential model-based optimization, where first only a few combinations of hyperparameters are used to build models, and the scores of these are used as labels to train a supervised ML model that tries to predict what would be the best hyperparameter combinations (this process is iteratively repeated).

3.7.5 Importance evaluation on hyperparameters

An important issue when defining a hyperparameter space is not only the choice of the ranges in which the hyperparameters may vary, but also choosing if a given hyperparameter is important to tune or if it can be left as a default value and therefore not come in the search process. This is discussed by Weerts et al. (2020). In this work, a methodology is presented to evaluate the performance loss that a model would have for not tuning a given hyperparameter. It is shown that for many models and particular problems, not tuning some hyperparameters leads to non-inferior results.

3.7.6 Summary

In Table 3.5, the most popular hyperparameter tuning methods are summarized.

Table 3.5: Methods for hyperparameter tuning.

Hyperparameter Search Method
Manual Tuning
Exhaustive Grid Search
Random Grid Search
Successive Halving

3.8 Data normalization

Data normalization is a fundamental part of data pre-processing for most ML models. This process's aim is to reduce the range in which a feature varies to a more useful range (such as ranges [-1,1] or [0,1]). The purpose of this is mainly for all features to vary in a common range. This is important for many ML algorithms because if it is not performed, features with greater values could have a larger impact on the predictions when compared with features ranging in intervals of lower magnitude, due only to this magnitude difference and not due to real relative feature importance. Normalization, therefore, eliminates this bias towards features with greater values, that is inherent to models such as NNs, SVMs and GPs (Singh and Singh (2020)). In some cases, normalization can be also applied to the output values (labels).

There are some types of models, such as those based on DTs in which data normalization does not have any impact because these do not have an inherent bias: splits are formulated for each feature taking into account its magnitude, but there is no greater importance given to features with greater values.

It is important to note that these processes should only occur after outlier removal, as described in Section 3.4, as is stated by Ali et al. (2014).

There are several alternatives for data normalization. Some can lead to better performance on a particular problem than others. Some of these alternatives, implemented in `SKLearn` will here be detailed. In all of these, data is centered and rescaled. Centering means translating all values such that their mean or median is null. Rescalling is a transformation such that the magnitude of the values is set to be in a particular range, or its dispersion is in another way controlled (for example by means of using Standard Deviations).

There are many other alternatives that can be found in literature, such as *tanh* based normalization; sigmoid based normalization; decimal scaling normalization or Pareto scaling. These are succinctly presented by Singh and Singh (2020).

3.8.1 Min-Max scaling

Minimum-Maximum scaling, or Min-Max scaling, is a simple transformation that uses only information from the maximum and minimum values of the feature, x_{max} and x_{min} respectively, to reduce data into a given interval. For example, to reduce data into range [0,1], one can use Equation (3.18). To reduce data into range [-1,1], one may use Equation (3.19) (Ali et al. (2014)). In essence, x_{min} is being used for centering, and the amplitude $x_{max} - x_{min}$ is being used for scaling purposes,

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.18}$$

$$x' = 2 \cdot \frac{x - x_{min}}{x_{max} - x_{min}} - 1 \tag{3.19}$$

where x is the original value of the feature, and x' the transformed value.

By inspecting the equations, it is evident that for $x = x_{min}$ and $x = x_{max}$, the limits of the pretended ranges are obtained.

3.8.2 Z-Score standardization

Z-Score standardization does not delimit a fixed range for data, but instead reduces it to a standardized normal distribution (assuming the original variable is normally distributed). The mean value will be 0 and its SD will be 1. For this, Equation (3.20) is used (equivalent to Equation (3.11)).

$$x' = \frac{x - \mu}{SD} \tag{3.20}$$

3.8.3 Robust scaling

Robust Scaling is a method for data rescaling that is more robust to the presence of outliers (even if these were supposed to already have been removed, as mentioned). First, the median (less affected by outliers) is used for centering, and then the first and third quartiles are used for the rescalling, as seen in Equation (3.21).

$$x' = \frac{x - Med}{IQR} \tag{3.21}$$

Where IQR is the Inter-Quartile Range, as defined in Equation (3.13), and Med is the Median value.

3.8.4 Summary

In summary, the normalization the methods for normalization that will be used within this worked are shown in Table 3.6

Table 3.6: Methods for data normalization.

Normalization Method	Value used for centering	Value used for rescalling
Min-Max scaling	x_{min}	$x_{max} - x_{min}$
Z-Score normalization	μ	SD
Robust scaling	Med	IQR

3.9 Multi-output problem approaches

Multi-output (MO) problems, as described in Section 3.2.4, are more complex than single-output (SO) problems, as they deal with multiple targets. Borchani et al. (2015) state that "the multivariate nature and the compound dependencies between the multiple feature/target variables" are some critical difficulties in MO problems.

There are multiple possible approaches for them. Some types of ML algorithms can inherently accommodate for more than one output, or can be easily adapted to do so. For example, an NN may have an output layer with more than one node, and a DT may have more than one label assigned to its leaf nodes. Other possibility is to perform transformations on data so that

the MO problem is transformed into one or more SO problems. There are other interesting possibilities, such as Chain Models, proposed by Read et al. (2011), that will be further detailed here.

3.9.1 Inherent multi-output handling

The advantages of using models that are inherently capable of handling multiple outputs is that, for a given MO problem, only one model needs to be trained, decreasing computational costs for this operation when compared to methods that subdivide the problem into many SO problems, and that, inherently, the ML models may be able to capture correlations between the labels (Pedregosa et al. (2011)).

3.9.2 Independent single-output models

When this is not possible, a popular method is the transformation of the MO problem with N target labels, into N independent SO problems. For each of them, a model predicting only one target is used, and the models do not share any information. This model has the problem of not modeling target correlations, and therefore information is lost (Read et al. (2011)).

3.9.3 Chained single-output models

The consensus view in the literature is that it is crucial to take into account label correlations during the prediction process. Read et al. (2011) propose a model similar to the aforementioned one, for classification problems, but based on a model chaining process. In this approach, N models, corresponding to the N labels to be predicted are built, but are fitted sequentially. The first model is trained using only the selected features; the second model is trained using these features plus the label of the first model; the third is trained using the features and the two labels previously predicted; and this process goes on until the Nth model, creating a Model Chain. This approach captures correlations between labels and keeps the main advantage of the approach of using N independent models, namely its simplicity and low time complexity when compared to other approaches, as Label-Combination or Pair-Wise classification methods (refer to Read et al. (2011)). The order of the models in the chain may have importance in this process, as stated by Pedregosa et al. (2011), who propose the training of several of these chains of models with random orders to find out the best performing configuration or to average out the predictions of all trained chains of models. Borchani et al. (2015) presents the possibility of using this approach for MO regression problems. This approach is implemented in `SKLearn` for both regression and classification.

3.9.4 Summary

The approaches that will be implemented for possible use in further sections are presented in Table 3.7 as a summary.

Table 3.7: Approaches on Multi-Output problems.

Multi-Output Approach
Inherently Multi-Output models
Independent Single-Output models
Chain Single-Output models

Chapter 4

Methodology

The purpose of the models created using the methodology presented here is for them to serve as surrogate models for Open Hole Tension (OHT) tests of CFRP laminates. The methodology for generating the data; extracting features and label values from data; preparing and organizing gathered data; building, training and evaluating ML models; and evaluating if the used amount of data is sufficient will here be presented. Additionally, an example database will be generated and used to illustrate part of the methodology.

4.1 Data Generation

Data for the training and testing of the ML models will come from FEM simulations based on a well validated model, following the methodology proposed by Furtado et al. (2019). The `Abaqus` FEM software will be used.

This methodology makes use of a validated continuum damage model to simulate intralaminar damage onset and propagation and a cohesive zone model to predict delamination behavior. It has been shown to accurately predict the tensile strength of specimens under Open Hole Tension, which was presented in Section 2.3, when compared to experimental results, for different materials, layups and hole diameters.

The model, whose geometry is seen in Figure 4.1, uses 8-node linear brick with reduced integration elements (C3D8R) for the plies, (one element in the thickness direction per ply thickness, t) and each layer is connected by COH3D8 user material cohesive elements with a thickness of 0.01 mm. The size of both the ply elements and cohesive elements is 0.5 x 0.5 mm^2. As all the layups considered are symmetric, a symmetry condition is applied at half the thickness, for enhanced computational efficiency. Other strategies to improve computational efficiency include the speed up of convergence and early stopping based on load drop and element deletion.

The relevant geometric features of the model may be seen in Figure 4.2, where the parameters W and D may vary from simulation to simulation. As is seen in the Figure, L = 2W.

Boundary conditions are shown in Figure 4.3: one of the ends is clamped, and a displacement is prescribed unidirectionally on the other end, so that the specimen is under tension.

The displacement is applied in `Abaqus` in a Dynamic, Explicit step. The displacement applied is such that the specimen fails, and outputs are registered until failure.

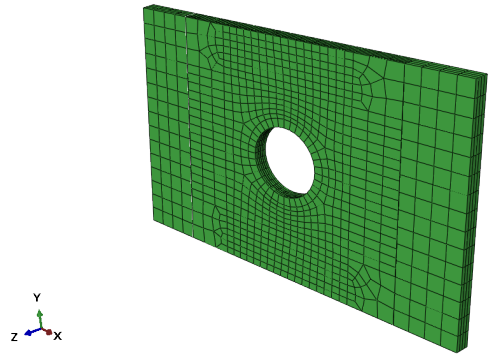


Figure 4.1: FEM model overview.

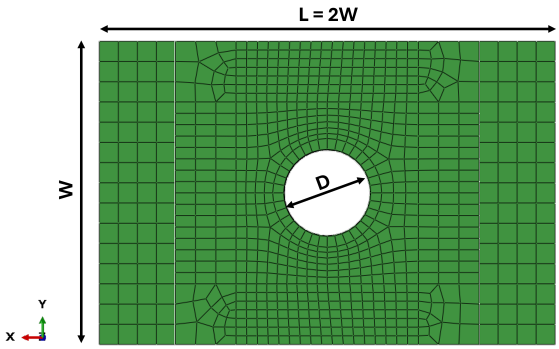


Figure 4.2: FEM model's mesh and geometry.

The surrogate models that will be developed in subsequent Sections will attempt to approximate these accurate numerical results.

4.1.1 Design of Experiments

As mentioned in Section 3.1, a good Design of Experiments (DoE) is important for surrogate modeling. The strategy used to select the inputs for the simulations that make up the databases for training and testing the ML models is similar to the one proposed by Furtado et al. (2021). It is important to remark that the approach here taken differs as the data from Furtado et al. (2021) was generated using an analytical model while here it was generated with high-fidelity numerical simulations, which provide more insights into the damage mechanics involved in sample failure as will be detailed later.

When generating a database, a sampling space is first chosen for each of the input values, as well as the number of samples, N_{points} that are desired. Then, a sampling method is applied to choose the values to be used for input in the FEM simulations. The sampling method here employed will be a Sobol Sequence sampling method (Saltelli et al. (2010), Sobol (2001)), implemented in the Python library SALib (Herman and Usher (2017)). This sampling method allows for a more uniform exploration of the sampling space than, for example, random sampling. It is important to recognise that this sampling method was designed for continuous values without

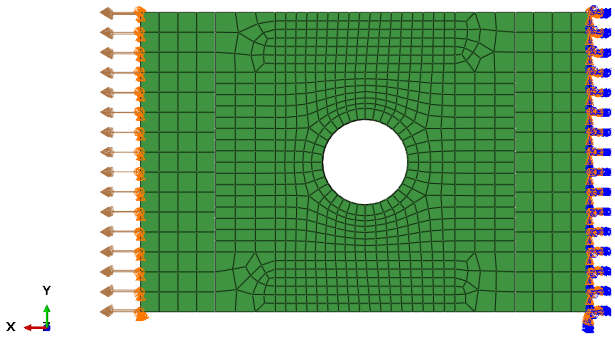


Figure 4.3: FEM model's boundary conditions and applied displacement.

inter-variable dependencies.

For the geometric parameters, D and W/D, a relevant to realistic applications range will be chosen. These ranges will be shown when presenting the particular generation of a database. Sobol sequencing can be applied without further care, as these variables are continuous and can be independently chosen.

The same applies for the sampling on the material properties, in the cases where the generation of a database with variable material is intended, as will be the case in Chapter 6. For databases with constant material, as the one generated for Chapter 5, the material properties are fixed as their average value for the selected material.

For conventional laminates, made up of plies at discrete angles 0°, ±45° and 90°, which will be the case for all specimens addressed in this work, the sampling of stacking sequences must take a different approach, as these do not directly constitute a continuous variable.

To solve this problem, sampling on the $\zeta^A_{1,2}$ space will be performed. As was noted in Section 2.5, each pair of $\zeta^A_{1,2}$ parameters can be uniquely converted into a layup (the fraction of plies at each direction is uniquely defined and from these fractions a layup can be selected, for example the layup with the lowest number of plies). This conversion process is non-trivial (IJsselmuiden et al. (2009), Irisarri et al. (2011)), and usually involves time-consuming optimization algorithms. To avoid these problems, some restrictions will be imposed to the layups that will be considered admissible, and therefore the creation of a reasonably small database that directly relates $\zeta^A_{1,2}$ to layup will be possible. This will be done by computing the laminate parameters for all admissible layups, using Equation (2.1a). The imposed restrictions are as follows:

- Only 0°, -45°, 45° and 90° orientations are considered.
- The total number of plies range from 6 to 30.
- All laminates are balanced (same number of 45° and -45° plies).
- All laminates are symmetric.

These restrictions result in a number of admissible laminates that constitutes a reasonable size for a database relating them to laminate parameters.

It is important to recall, from Section 2.5, that these two parameters, ζ_1^A and ζ_2^A, are not independent, and therefore a further step must be taken when performing the sampling.

To take this into account, and once again following the work of Furtado et al. (2021), first a Sobol sequence sampling will be done on the whole square shaped space: both parameters ranging from -1 to 1. After this, for each point in this space, the closest admissible point (present in the admissible database that was created) is computed. Then, this original Sobol sampled point may be totally discarded if the computed distance is above a certain threshold, or it may be approximated to that closest point, if the distance is below the threshold. A threshold distance of 0.1 was used.

In Figure 4.4, an example of this sampling process is presented: the admissible points in grey; the points generated through the naive Sobol sampling in the rectangular domain in red; and finally, the chosen points, in blue.

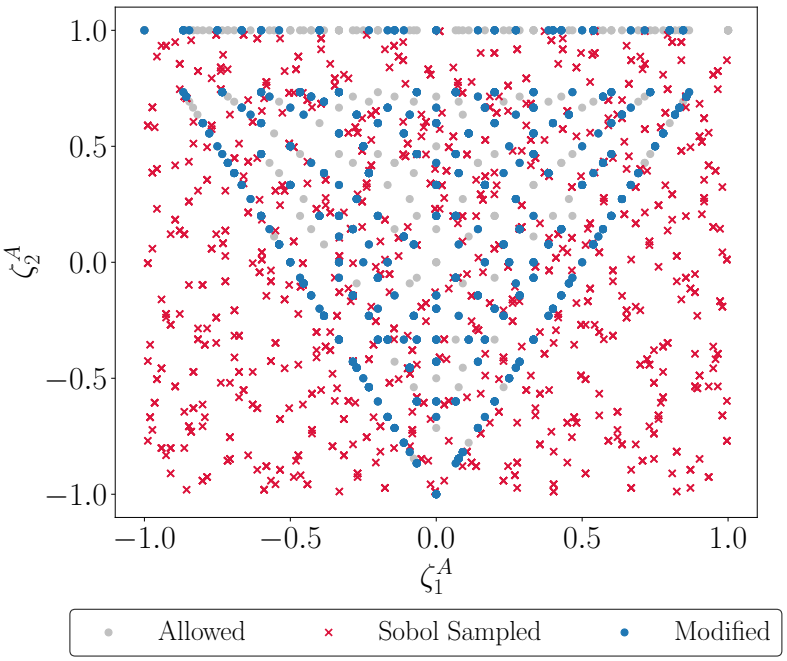


Figure 4.4: Sampling in the $\zeta_{1,2}^A$ space through Sobol sequencing.

By accessing the database of admissible layups, it is now possible to obtain the layups for each of the sampled $\zeta_{1,2}^A$ points. Repetition of layups is permitted, if coincident points exist, as the order of the plies is still to be defined.

To define the order of the plies for each selected layup, which influences the out-of-plane laminate parameters $\zeta_{1,2,3}^D$, further restrictions may be imposed. In particular, it is important to define a maximum number of consecutive plies in the same direction. This is because realistic laminates do not usually have large ply blocks of this kind. For the training and testing databases, a maximum ply block of 2 is imposed (no more than two consecutive plies in the same direction are admissible).

Having established this limitation, for each of the previously selected points, a random shuffle is performed until an admissible sequence is found. If no admissible stacking sequence

is found for a certain layup point, then it is removed from the selection. This may result in the elimination of layups with too many 0° or 90° plies, that are not layups with practical use. This is clearly seen in the example presented in Figure 4.5, by the absence of points in the top corners of the triangular domain. Repeated stacking sequences are also not allowed.

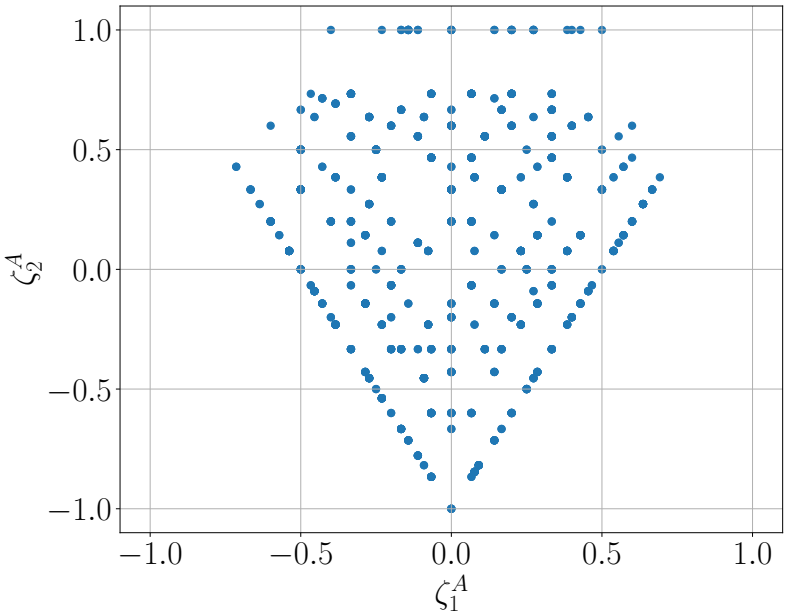


Figure 4.5: Sampling after stacking sequence random generation.

The $\zeta^D_{1,2,3}$ parameters can now be computed for each stacking sequence.

4.1.2 Running the FEM Simulations

After this DoE process, all input parameters for the simulations are selected:

- Geometric parameters: D and W/D;
- Material properties;
- Ply stacking sequences.

Using the methodology of Furtado et al. (2019), these can now be given as simulation input for FEM simulations in `Abaqus`, in the shape of *.inp* files.

To create these files, the selected points are given in two separate files: a *x_input.npy* file and a *layups.txt* file. The *x_input.npy* file contains the relevant geometric parameters: D and W/D; and laminate parameters $\zeta^A_{1,2}$ and $\zeta^D_{1,2,3}$. It also contains the material properties if these are variable. The *layups.txt* file contains a list whose elements are half of the stacking sequences of the symmetric laminates.

With the N_{points} files (in *.inp* format) generated, the simulations can be directly run in `Abaqus`.

The output values of interest are the stress-strain curves (and particularly the ultimate strength and the deformation at maximum stress) and the energies at failure. These can be obtained by post-processing the results from `Abaqus`.

4.2 Post-processing

After the simulations have finished running, a *.odb* results file is obtained for each simulation. Now a Python script is used that opens these results files and extracts the relevant results.

In this first post-processing step, for each simulation, the load-displacement curves are extracted, and stored in a *load_displacement.txt* file. The normalized sum of the energies of all elements, g_1, g_2, g_6, g_{coh}, in the longitudinal direction, transverse direction, related to shear and related to cohesion between plies, respectively, are also extracted and stored in a *energies.txt* file, for all pseudo-time steps.

From these files, present in each simulation folder, the values relevant to training may be extracted and stored in an organized manner.

4.2.1 Ultimate strength

From the load-displacement curve that was already extracted during the post-processing phase, the maximum load is obtained, F_{max} as well as the pseudo-time step in which it occurred, T_{max}.

This load value is then converted into the corresponding engineering stress, by using the geometric dimensions present in the input file. The width, W, is computed by multiplying the feature corresponding to the diameter, D, by the feature corresponding to the ratio between width and diameter, W/D. The total thickness of the specimen, t_{total} is computed by multiplying the ply thickness, t, considered a material propriety, by the number of plies, N, which can be counted in the *layup.txt* file.

With this information, Equation (4.1) is used to compute the maximum stress that the specimen is subjected to, or in other words, its notched Ultimate Strength, σ_u.

$$\sigma_u = \frac{F_{max}}{t_{total} \cdot W} \tag{4.1}$$

4.2.2 Deformation at maximum stress

Similarly to what was done in Section 4.2.1, the displacement value at the point of maximum load, D_{max}, which is the displacement value at T_{max}, will be converted into a deformation value, by using the geometry of the specimens.

As stated in Section 4.1, the length of the models is twice as large as their width. Therefore, using the previously extracted value for the width, W, it is possible to compute the initial length of the specimen, L = 2W.

With this information, Equation (4.2) is used to compute the deformation at maximum stress.

$$\varepsilon_{max} = \frac{D_{max}}{L} \tag{4.2}$$

4.2.3 Energies and Failure mode

Failure mode will be labeled automatically with a rule that takes the energies dissipated at failure into account.

The energies, g_1, g_2, g_6, g_{coh}, have been already extracted, for all pseudo-time steps, in the post-processing phase. Their absolute values at T_{max} are now stored, $g_{1_{max}}$, $g_{2_{max}}$, $g_{6_{max}}$, $g_{coh_{max}}$.

The total energy at failure is then computed, by adding the four components, as shown in Equation (4.3).

$$g_{total} = g_{1_{max}} + g_{2_{max}} + g_{6_{max}} + g_{coh_{max}} \tag{4.3}$$

The fraction of the total energy at failure, for each of the four components is what will be used in the failure labeling rule. These are computed as seen in Equation (4.4):

$$g_{1_{frac}} = \frac{g_{1_{max}}}{g_{total}} \tag{4.4a}$$

$$g_{2_{frac}} = \frac{g_{2_{max}}}{g_{total}} \tag{4.4b}$$

$$g_{6_{frac}} = \frac{g_{6_{max}}}{g_{total}} \tag{4.4c}$$

$$g_{coh_{frac}} = \frac{g_{coh_{max}}}{g_{total}} \tag{4.4d}$$

As presented in Section 2.6, there are several possible failure mechanisms and modes that may be present for CFRP laminates, that depend on the solicitation, material properties and fiber orientation. Here, the Failure Mode, FM, of the laminate will be classified as fiber dominated or as matrix dominated using a simple rule based on the energies dissipated at failure. Classifier models can be built to predict FM.

The rule is as follows: if $g_{1_{frac}} > g_{2_{frac}} + g_{6_{frac}} + g_{coh_{frac}}$ then the Failure Mode is fiber dominated, labeled as FM1; else, Failure Mode is matrix dominated, labeled as FM2.

4.2.4 Polynomial curve coefficients

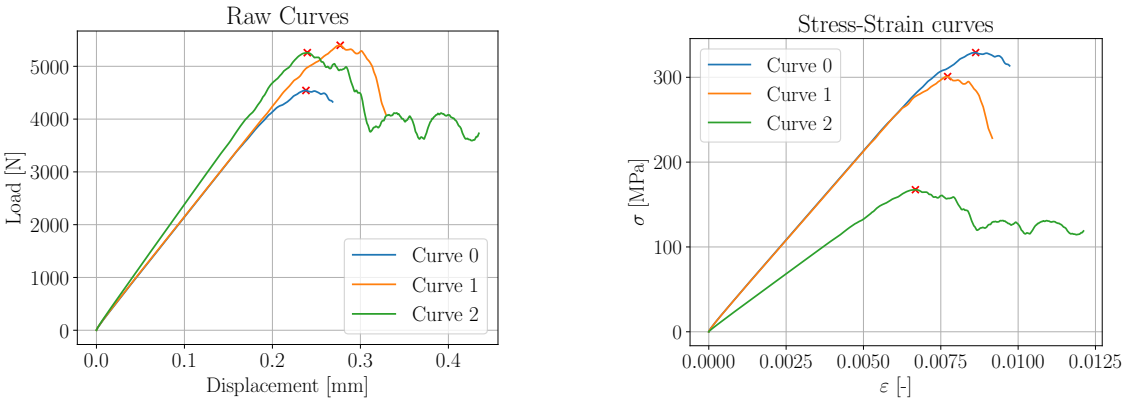
The stress-strain curves will be approximated by using third order polynomials. These will capture the general shape of the curve. Equation (4.5) shows a polynomial of this type, with four coefficients, a_0 to a_3.

$$\sigma = a_0\varepsilon^3 + a_1\varepsilon^2 + a_2\varepsilon + a_3 \tag{4.5}$$

All curves will be forced to pass through the origin, meaning that the independent coefficient of the polynomial will always be null: $a_3 = 0$.

To approximate stress-strain curves, the models will therefore only have to predict three coefficients: a_0, a_1 and a_2.

First, the load-displacement curves will be converted into stress-strain curves, by using the geometric parameters present in the input files, similarly to what was presented in Sections 4.2.1 and 4.2.2. For three illustrative examples, the true load-displacement curves are presented in Figure 4.6a, and the corresponding stress-strain curve in Figure 4.6b.



(a) Load-displacement curves.

(b) Stress-strain curves.

Figure 4.6: Transformation of load-displacement curves into stress-strain curves for three illustrative curves.

After this, the curves will be truncated at the point of maximum stress. This means that all points with deformation larger than the deformation at T_{max} are eliminated. This truncation can be seen in Figure 4.7 for each of the three example curves.

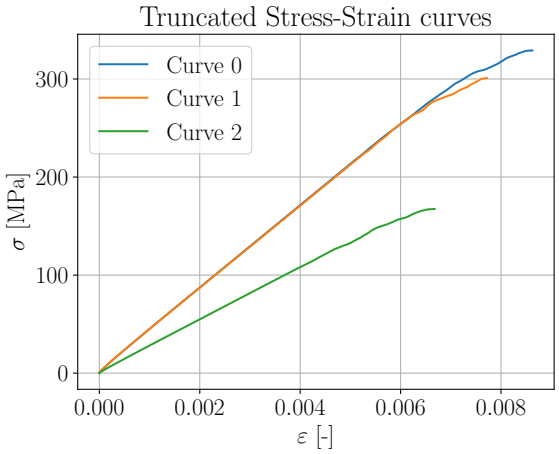
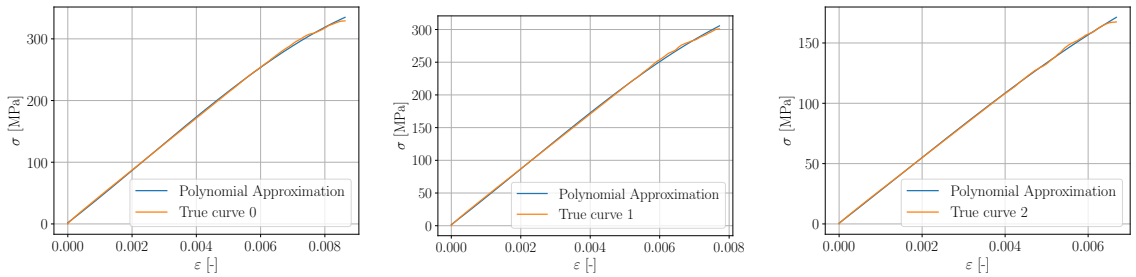


Figure 4.7: Three example stress-strain curves after the truncation process.

The `scipy` library (Virtanen et al. (2020)) will be used to fit the polynomials to these truncated stress-strain curves. For this, a function $\sigma(\varepsilon, a_i)$, $i = 0, 1, 2$, is defined, as shown in Equation (4.5). The function *curve_fit* from the `scipy` library is then used, providing us the a_i coefficients for each curve, that can now be stored. The third order polynomial fittings were deemed sufficiently accurate. For the three examples at hand, these can be seen in Figure 4.8.



(a) (b) (c)

Figure 4.8: Curve fitting with third order polynomials for the three example curves.

4.3 Feature Engineering

As stated in Chapter 1, the objective of this work is the development of surrogate models, able to reproduce, with acceptable error, the results from well calibrated FEM models, but with much lower computational costs.

For this purpose, a possible approach would be to provide the ML models the same inputs as are given to the FEM model, and expect them to predict the same outputs. The issue with this approach lies with the stacking sequence input. This input for the FEM simulations comes in the form of a list of angles, which is of variable length, as it depends on the number of layers of the laminate. This is not a very friendly nor easily interpretable input for ML models, as was alluded to in Section 2.5. To get around this problem, the ζ laminate parameters will be used.

Although this solution works very well for the in-plane $\zeta^A_{1,2}$ parameters, that is not the case for the out-of-plane $\zeta^D_{1,2,3}$ parameters. This is because these are parameters more related to flexural stiffness, and therefore do not correlate so well with in-plane strengths. The order of the plies, which only affects $\zeta^D_{1,2,3}$ and not $\zeta^A_{1,2}$, does impact the ultimate strength of the laminate, as was shown by Esteves (2023).

To capture this influence then, other features are derived from the stacking sequence. Below, these will be shown. Other features, not relating to the stacking sequence order, namely ζ^A distance and Number of Plies, are also presented, and will be used, as they are useful.

Each of the stacking sequences in Table 4.1 will be used as examples to illustrate the derivation of the features. Note that Example ID 3 wouldn't be admissible in the training and testing databases because it has 4 consecutive plies oriented at 90°, but it will here be used for illustration purposes.

4.3.1 Maximum Ply Block

For a given stacking sequence, it is possible to group plies into blocks of consecutive plies with fibres in the same direction. These blocks in practice may be seen as plies with greater thickness. In a stacking sequence, ply blocks of plies at 0°; at -45°; at 45° and at 90° may be distinguished.

The plies in each of these ply blocks can be counted, and these counts can be organized into an array of Ply blocks, as seen in Table 4.2. The Maximum Ply Block (MPB) of a given stacking sequence is the maximum value in this array. It depends, of course, on the order of the plies.

Table 4.1: Example layups and stacking sequences for feature derivation. The subscript $_S$ indicates that the stacking sequences are symmetric.

Example ID	Layup: Number of plies at each direction				Stacking Sequence
	0°	-45°	45°	90°	
1	4	2	2	2	$[-45,0,0,45,90]_S$
2	4	2	2	2	$[0,-45,45,90,0]_S$
3	2	2	2	6	$[0,90,45,-45,90,90]_S$
4	2	2	2	6	$[90,90,45,-45,0,90]_S$

This feature does not seem to correlate strongly with the strength of a specimen. Additionally, it may not have a lot of variance, in practical terms, as stacking sequences with MPBs greater than 2 are not frequent. This means that almost all laminates of interest have a MPB of 2 (for symmetric laminates there is always at least a ply block of 2 in the symmetry plane of the laminate). Therefore, it is probably not a very informative feature for the ML models.

4.3.2 Average Ply Block

A more useful feature, as proposed by Esteves (2023), is the Average Ply Block (APB). To derive this metric, the number of plies for each ply block is again counted. In this case, the ply block counts are assembled in three groups: those corresponding to plies at 0°; those corresponding to plies at -45° or 45°; and those corresponding to plies at 90°. These groups are averaged out to extract the arrays of Average Ply Block Per Type, as seen in Table 4.2. The Average of these three Average Ply Block Per Type values is the APB.

For Example ID 2, for illustration, there are three ply blocks of plies at 0°, two with a ply count of 1 and the other (at the symmetry plane) with a ply count of 2. This results in an Average Ply Block Per 0° direction of $(2\cdot1+2)/3 \approx 1.33$. There are four ply blocks with directions ±45°, all with a ply count of 1, what results in Average Ply Block Per ±45° of $(4\cdot1)/4 = 1$. There are two ply blocks with direction 90°, both with a ply count of 1, what results in Average Ply Block Per ±45° of $(2\cdot1)/2 = 1$. APB is then the sum of the three Average Ply Block Per Type divided by three.

This seems to be a feature carrying more information about the stacking sequence, for a given order of plies.

4.3.3 Average Mismatch Angle

The mismatch between successive angles in a stacking sequence is known to have an impact on the material's behaviour. This is another metric proposed by Esteves (2023), shown to correlate with specimen strength.

A stacking sequence is a sequence of angles θ_i, $i = 1$ to N. The mismatch between two successive plies is then $\Delta\theta_{i,i+1} = \theta_{i+1} - \theta_i$.

From the stacking sequence, the average Mismatch Angle (MA) between plies is extracted as shown in Equation (4.6).

$$MA = \frac{\sum_{i=1}^{N-1}(\Delta\theta_{i,i+1})}{N-1} \tag{4.6}$$

Examples of the derivation of this feature, including a list of the $N-1$ values of $\Delta\theta_{i,i+1}$ and the resulting MA, for each of the four example stacking sequences, are presented in Table 4.2.

This feature also seems to carry relevant information on the stacking sequence. In general, the larger the mismatch angle, the higher the stresses (Esteves (2023)).

Table 4.2: Maximum Ply Block, Average Ply Block and average Mismatch Angle derived for four examples.

ID	Ply blocks	MPB	Average Ply Block Per Type [0° , ±45° , 90°]	APB	Mismatch Angles	MA
1	[2,1,1,2,1,1,2]	2	[2.00,1.00,2.00]	1.67	[45,0,45,45,0,45,45,0,45]	30.0
2	[1,1,1,1,2,1,1,1,1]	2	[1.33,1.00,1.00]	1.11	[45,90,45,90,0,90,45,90,45]	60.0
3	[1,1,1,1,4,1,1,1,1]	4	[1.00,1.00,2.00]	1.33	[90,45,90,45,0,0,0,45,90,45,90]	49.1
4	[2,1,1,1,2,1,1,1,2]	2	[1.00,1.00,2.00]	1.33	[0,45,90,45,90,0,90,45,90,45,0]	49.1

4.3.4 ζ^D distance

ζ^D distance, $\zeta^{\bar{D}}$, is a simple combination of the $\zeta^D_{1,2,3}$ parameters, defined in Equation (4.7) (Esteves (2023)). It measures the distance, in the $\zeta^D_{1,2,3}$ space, of a point to the origin.

$$\zeta^{\bar{D}} = \sqrt{\zeta_1^{D^2} + \zeta_2^{D^2} + \zeta_3^{D^2}} \tag{4.7}$$

4.3.5 ζ^A distance

ζ^A distance, $\zeta^{\bar{A}}$, is defined similarly to ζ^D distance, but with the $\zeta^A_{1,2}$ parameters, as shown in Equation (4.8). This metric in essence gives information about how far the laminate is from a Quasi-Isotropic laminate.

$$\zeta^{\bar{A}} = \sqrt{\zeta_1^{A^2} + \zeta_2^{A^2}} \tag{4.8}$$

This feature does not relate to the stacking sequence, and is only layup dependent.

4.3.6 Number of plies

The Number of Plies, NP, is a metric that may reasonably explain some variation on the results of the OHTs. It is simply a count of the total number of plies of the layup.

This metric does not relate to the stacking sequence, and is only layup dependent.

The NP; laminate parameters (computed by using Equation (2.1)); and laminate parameter distances (computed using Equations (4.7) and (4.8)) features, for each of the example stacking sequences, are presented in Table 4.3. Note that the ζ^A_1, ζ^A_2 and $\zeta^{\bar{A}}$ are the same for Examples

ID 1 and ID 2, because these have the same layup (and the same can be said for the pair ID 3 and ID 4). The $\zeta_1^D, \zeta_2^D, \zeta_3^D$ and $\zeta^{\bar{D}}$ features change within these pairs, because they are ply order dependent.

Table 4.3: Number of plies, laminate parameters, and laminate parameter norms derived for four examples.

ID	NP	ζ_1^A	ζ_2^A	$\zeta^{\bar{A}}$	ζ_1^D	ζ_2^D	ζ_3^D	$\zeta^{\bar{D}}$
1	10	0.2	0.2	0.2828	0.440	-0.088	-0.432	0.6229
2	10	0.2	0.2	0.2828	0.440	0.104	-0.144	0.4745
3	12	-0.333	0.333	0.4714	0.102	0.481	0.083	0.4991
4	12	-0.333	0.333	0.4714	-0.676	0.481	0.083	0.8340

4.3.7 Feature combinations

The previously presented features were also combined in four different ways, to create metrics with increased correlation with the simulation outputs, C_1 to C_4. These are shown in Equations (4.9) to (4.12), and computed for the four examples in Table 4.4. The objective of the creation of these artificial parameters was to try to, in a synergistic way, combine other features and operate on them to have values that are more strongly, and if possible more linearly, correlated with Ultimate Strength.

$$C_1 = \sqrt[3]{\frac{MA}{APB}} \tag{4.9}$$

$$C_2 = log_{10}\left(\sqrt[3]{\frac{\zeta_{\bar{D}}}{APB}}\right) \tag{4.10}$$

$$C_3 = log_{10}\left(\zeta_{\bar{D}} \cdot APB\right) \tag{4.11}$$

$$C_4 = \sqrt[3]{\zeta_{\bar{D}} \cdot \zeta_{\bar{A}}} \tag{4.12}$$

Table 4.4: Feature combinations derived for four examples.

ID	C1	C2	C3	C4
1	2.6207	-0.3280	0.0374	0.5606
2	3.7798	-0.2836	-0.6401	0.5120
3	3.3268	-0.3275	-0.4072	0.6174
4	3.3268	-0.1564	0.1062	0.7326

4.3.8 Stacking Sequence features summary

In summary, there are a total of 15 features derived from the layup and the order of the plies (stacking sequence). These are summarized in Table 4.5. Note that the feature MPB will seldom be used throughout this work, due to its low variance for realistic laminates. This will be referred in later Chapters.

Table 4.5: Summary of features derived from the layups and stacking sequences.

Derived Features	Abbreviation	Related to Stacking Sequence Order
Number of Plies	NP	No
1st in-plane parameter	ζ_1^A	No
2nd in-plane parameter	ζ_2^A	No
1st out-of-plane parameter	ζ_1^D	Yes
2nd out-of-plane parameter	ζ_2^D	Yes
3rd out-of-plane parameter	ζ_3^D	Yes
ζ^A Distance	$\zeta^{\bar{A}}$	No
ζ^D Distance	$\zeta^{\bar{D}}$	Yes
Maximum Ply Block	MPB	Yes
Average Ply Block	APB	Yes
Mismatch Angle	MA	Yes
Combination 1	C_1	Yes
Combination 2	C_2	Yes
Combination 3	C_3	Yes
Combination 4	C_4	Yes

4.4 Data Pipeline

In this Section, the general pipeline by which data goes through, from raw state to the training of the models, will be outlined. Each type of model may have its specifications, and these will be detailed in subsequent sections. All databases, although different (fixed material or variable material, for example) will go through this pipeline. The number of simulations performed for a specific database will be denoted as N_{points}.

4.4.1 Data organization

Data is first in a raw state, coming from its generation. The raw output files, containing the values extracted as described in Section 4.2, are stored in *.npy* files. These files can be directly and easily converted into `Numpy` (Harris et al. (2020)) *ndarrays* with N_{points} rows. These are in total four output files:

- *y_database.npy* - contains a `Numpy` *ndarray* of dimensions (N_{points},1). Each row contains the value of Ultimate Strength of the specimen, extracted as described in Section 4.2.1.

- *deformation_database.npy* - contains a `Numpy` *ndarray* of dimensions (N_{points},1). Each row contains solely the value of the deformation at the point T_{max}, as seen in Section 4.2.2.
- *energies_database.npy* - contains a `Numpy` *ndarray* of dimensions (N_{points},). Each row contains four float values, corresponding to $g_{1_{frac}}$, $g_{2_{frac}}$, $g_{6_{frac}}$ and $g_{coh_{frac}}$, in this order, extracted as described in Section 4.2.3.
- *coeffs_database.npy* - contains a `Numpy` *ndarray* of dimensions (N_{points}, 3). Each row contains four float values, corresponding to the polynomial curve coefficients, a_0, a_1 and a_2, in this order, as seen in Section 4.2.4.

Together with these four files, in the raw database, the files that were given as input to create the *.inp* files are also stored. These are the *layups.txt* and *x_quad.npy* files, as explained in Section 4.1. It is important to note that *layups.txt* contains a list with N_{points} elements, each of which corresponds to the rows of the raw output files. Similarly, each of the N_{points} rows of the *x_quad.npy* correspond to the row with the same index in the raw output files.

This raw state of data is not convenient. First, it is not easy to create the features that need to be derived from the stacking sequence, as described in Section 4.3, and these would have to be stored in an additional file. Furthermore, data analysis is more difficult, and a risk of mismatching indexes would always exist during the pipeline before and after training, because data must be shuffled, cleaned (by removing outliers for example), and split. It is important, therefore, that all data is combined into a single file. For this, the `Pandas` library (Wes McKinney (2010)) will be used, in particular its class *DataFrames*. These will allow us to store data in a single table and store it in a *.pkl* file. An intermediate *DataFrame* is then created, that puts together all the referred raw data, with clearly labeled columns:

- One column with the half layups, from *layups.txt*.
- Two columns for the geometric parameters D and W/D, from *x_quad.npy*.
- A column for each of the material properties, from *x_quad.npy* or explicitly specified if the material is fixed.
- A column for an optional label of the class/type of the laminate. This can be useful for data handling (it can specify if the laminate is hard or soft, or it can specify the name of the material from which it is made, for example).
- A column for each of the values of the raw output files: one for the Ultimate Strength (σ_u); one for Failure Deformation (ε_{max}); four for the energies dissipated at failure ($g_{1_{frac}}$, $g_{2_{frac}}$, $g_{6_{frac}}$, $g_{coh_{frac}}$) and three for the polynomial coefficients (a_0, a_1, a_2).

It is important to note that each row now has an identifying index, i. An example of such table can be seen in Table 4.6, for three random selections of stacking sequences and geometry parameters. Material properties presented are fixed for these, but could vary from row to row, depending on the purpose of the database. After this, operations are performed on the intermediate *DataFrame*, such that a fully processed *Dataframe* is obtained. These operations will generate all of the features defined in Section 4.3, and a column will be created for each of them. The Failure Mode is also derived from the failure energies, as described in Section 4.2.3.

An example of the fully processed *DataFrame* is shown in Table 4.7. The column with "..." is the place where the columns from "t" to "Giic", already shown in Table 4.6, would be.

Table 4.6: Example of intermediate *DataFrame*.

i	Ultimate Strength	Stacking Sequence	t	D	W/D	$g_{1_{frac}}$	$g_{2_{frac}}$	$g_{s_{frac}}$	$g_{coh_{frac}}$	a_0
743	444.600	[45,45,0,−45,0,90,90,0,90,90,−45,90,0,0,90]	0.125	2.703	3.479	88.96	3.51	7.15	0.38	−3.3E8
656	273.585	[90,45,−45,45,90,45,−45,45,−45,90,90,−45,0]	0.125	3.328	5.119	80.72	6.76	11.79	0.73	−1.7E8
704	503.082	[0,−45,0,0,−45,45,90,0,−45,45,45,0]	0.125	5.172	3.537	93.48	1.44	3.98	1.11	−3.4E8

a_1	a_2	Failure Deformation	Class	E1	G12	XT	XC	SL	GXT	GXC	Giic
2.0E6	6.1E4	0.00834	OHT IM7	171420	5290	2323.5	1200.1	92.3	133.3	61	0.79
1.0E6	3.7E4	0.00831	OHT IM7	171420	5290	2323.5	1200.1	92.3	133.3	61	0.79
11.6E6	7.8E4	0.00706	OHT IM7	171420	5290	2323.5	1200.1	92.3	133.3	61	0.79

Table 4.7: Example of fully processed *DataFrame*.

i	Ultimate Strength	Stacking Sequence	...	Failure Mode	NP	ζ_1^A	ζ_2^A	ζ_1^D	ζ_2^D
743	444.600	[45,45,0,−45,0,90,90,0,90,90,−45,90,0,0,90]	...	FM1	30	-0.0667	0.467	0.0744	0.0305
656	273.585	[90,45,−45,45,90,45,−45,45,−45,90,90,−45,0]	...	FM1	26	-0.231	-0.231	33728	-0.323
704	503.082	[0,−45,0,0,−45,45,90,0,−45,45,45,0]	...	FM1	24	0.333	0.000	0.495	0.201

ζ_3^D	$\zeta^{\bar{D}}$	$\zeta^{\bar{A}}$	MPB	APB	MA	C_1	C_2	C_3	C_4
0.213	0.228	0.471	2	1.433	45.0	3.191	-0.613	-1.120	0.475
0.183	0.502	0.326	2	1.444	57.6	3.416	-0.352	-0.321	0.547
-0.222	0.579	0.333	2	1.210	47.0	3.386	-0.246	-0.356	0.578

4.4.2 Data preparation and model training

The datasets are now organized and ready to be used. Before fitting the models, though, some considerations have to be made.

Data is first imported and shuffled, such that there aren't any artificial biases in the splits that will be done further down the line, originating from the data generation process.

After this, for any model that will be trained, the following items should be specified:

1. **Type** of models: one of the types of models specified in Section 3.3. These will be selected from Table 3.1.
2. **Class** of models: regression or classification; single output or multi-output. These classes were detailed in Section 3.2. In the case of a multi-output scenario, an approach from Table 3.7 will be selected.
3. **Target**: values or classes that will be predicted.
4. **Outlier detection**: the metric and threshold to be used, from Table 3.2.
5. **Feature Selection** that is done and the features that are given as input to the model. This will be chosen from Table 3.3.
6. **Train-test split**: fraction of data points used for testing and type of split from Table 3.4.
7. **Cross-Validation strategy**: number of folds; stratification rule and scoring function.
8. **Data normalization** or scaling that was used, selected from Table 3.6.
9. **Hyperparameter** selection; type of grid search, chosen from Table 3.5.

When presenting the models built for the various purposes, in Sections to follow, all of these nine items will be detailed. In addition, the dataset being used and its generation process will be made clear beforehand. With this information, one may follow the pipeline below, that uses the now organized data to train predictors.

According to the **class** and **type** of predictors to be used, predictors are imported from the `SKLearn` library.

An **outlier detection** is done as shown in Section 3.4. Outliers are identified in the dataset and removed from it. For multi-output models, this detection is done in a similar fashion: the metric is computed for all targets, and only the data-points that do not surpass the threshold for any of them are kept in the database.

A **train-test split** is then done. As explained in Section 3.6.1, a fraction of the total data points will not be seen by the models during training, and therefore are will allow us to test and evaluate model performance in an unbiased way. This fraction must be specified when preparing data for model training. Values between 10% and 20% of the total dataset are common.

There are different strategies to make this split. These were outlined in Section 3.6.1. When preparing data, a plain Shuffle Split or a Stratified split may be chosen. Depending on the strategy chosen, the number of bins and stratification label ought to be specified.

The **target** value(s) or class column(s) is removed from the train and test set data and stored in a different array. This results in four arrays:

- X_{train}, an array with all the input features for the data points selected for training;
- X_{test}, an array with all the input features for the data points selected for testing;
- Y_{train}, an array with the target values for the data points selected for training;
- Y_{test}, an array with the target values for the data points selected for testing.

Feature Selection may be performed, using techniques described in Section 3.5. The names of the features to be used will be stored in an array. Features with null variance will always be removed, as they do not bring useful information. Then, feature selection may be performed by using Pearson coefficients, mutual info regression coefficients, with a specified threshold, or by applying a Recursive Feature Selection (with or without Cross-Validation), with a specified base model.

A **Cross-Validation strategy** is defined for model training. The type of cross-validation should be one of the many described in Section 3.6.2: a plain K-Fold cross-validation strategy or a stratified K-Fold cross-validation strategy. The number of folds; bins and stratification labels must be specified, if applicable. If a repeated K-Fold validation is employed, the number of repetitions will also be specified.

Data normalization is then performed, using one of the scalers presented in Section 3.8: Min-Max scaling; Z-Score normalization or Robust scaling may be chosen.

Finally, a **hyperparameter** grid is selected, so that the best hyperparameters possible are found by comparing cross-validation scores when performing a type of grid-search, as shown in Section 3.7.

Models then train for various values in the hyperparameter grid, and when the best hyperparameters are found, all of the training set is used to train a final model that can then be evaluated.

Models are stored in a *.joblib* file, together with the features that were used for training; the training set and the test set (features and target labels in both cases).

4.5 Model evaluation

4.5.1 Ultimate strength regressors

A strength regressor is a single output regressor. These models will directly predict the Ultimate Strength, σ_u, of the specimens under OHT, from the geometric, laminate and material features given as input. These values are obtained from the FEM simulations as shown in Section 4.2.1

To evaluate the strength regressors, two main metrics will be used: Root Mean Squared Error, RMSE, and Relative Error, RE. These metrics are errors computed between the values obtained as outputs of the FEM simulations, y_i^{true}, stored in an array y^{true} and the predictions of the ML models, y_i^{pred}, stored in y^{pred}.

The RMSE is the squared root of the mean squared error between each of these points. It gives a single value for each model/test set.

RE is computed for each of the points, RE_i, and will not be averaged. It will instead be used for histogram plotting purposes and to evaluate the fraction of predictions that are under a certain RE threshold. Occasionally, the absolute value of the RE will be used.

So, for n points in a testing set, the metrics are defined as shown in Equations (4.13) and (4.14).

$$\mathrm{RMSE}(y^{true}, y^{pred}) = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}\left(y_i^{true} - y_i^{pred}\right)^2} \tag{4.13}$$

$$\mathrm{RE_i}(y_i^{true}, y_i^{pred}) = \frac{y_i^{pred} - y_i^{true}}{y_i^{true}} \tag{4.14}$$

Furthermore, the values of y^{true} and of y^{pred} will usually be plotted against each other, in a scatter plot, for better visualisation. In a perfect model, $y^{pred} = y^{true}$, and so this plot would have all points aligned with a line that bisects the odd quadrants.

The models will be often compared to *DUMMY Models*, which will serve as baseline. In the case of a regressor, these will always predict the mean value of the training set. This is a good reference to see how much better are the ML models than just naive guessing.

Sensitivity analysis

To explore if the trained models capture the main trends in laminate behavior, smaller databases will be generated (with points not seen during training), with the objective of performing sensitivity analysis.

First, a base laminate is chosen, with input values close to the average values in the range for which the models were trained. Then, the other laminates are chosen in such a way that only one of the variables in the input values vary in relation to the base laminate. These include the geometric variables; the laminate parameters; and the material properties, when the models are trained for variable material.

Comparison with experimental results

Whenever possible, the models' predictions will be compared to real experimental results from the literature.

Although the models are supposed to approximate the numerical models, and, therefore, the errors of the ML models when put in comparison to experimental results will depend both on their approximation to numerical models and on the accuracy of the numerical models themselves, it is important to try to understand how reasonable it would be to use the ML models in real applications.

4.5.2 Failure classifiers

A failure classifier is a single output binary classifier. These models will predict the Failure mode, from the two possible classes (fiber dominated or matrix dominated), of the specimens under OHT. From the geometric, laminate and material features given as input.

To evaluate the failure classifier, two main metrics will be used: Accuracy, *Acc*, and Balanced Accuracy, *BAcc*.

Accuracy is defined simply as the quotient between correct predictions, and incorrect predictions.

Balanced accuracy is a metric that is useful when datasets are unbalanced. For example, if in the test set 80% of the data-points have fiber dominated failure and 20% have matrix dominated failure, the accuracy of a *DUMMY* predictor that always predicts fiber dominated failure would be of 80%. Balanced accuracy is defined in such a way that the *BAcc* is 50% for a *DUMMY* predictor as described, for any unbalance on a dataset. It is defined as seen in Equation (4.15),

$$BAcc = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) \tag{4.15}$$

where TP is the number of True Positives; TN the number of True Negatives; FP the number of False Positives; and FN the number of False Negatives. These are defined, considering FM1 as the positive class in the binary classification problem as follows:

- TP: Data-point with true label FM1 that is correctly predicted;
- TN: Data-point with true label FM2 that is correctly predicted;
- FP: Data-point with true label FM1 that is incorrectly predicted;
- FN: Data-point with true label FM2 that is incorrectly predicted.

Furthermore, confusion matrices will be presented. In these, the predictions on the test set can more directly be scrutinized. These matrices present the TP, TN, FP and FN numbers, in terms of their percentages.

The classifiers will be compared to a *DUMMY* classifier, that will serve as baseline. The *DUMMY* classifier always gives the class that was the most seen in the training set as its prediction.

4.5.3 Curve predictors

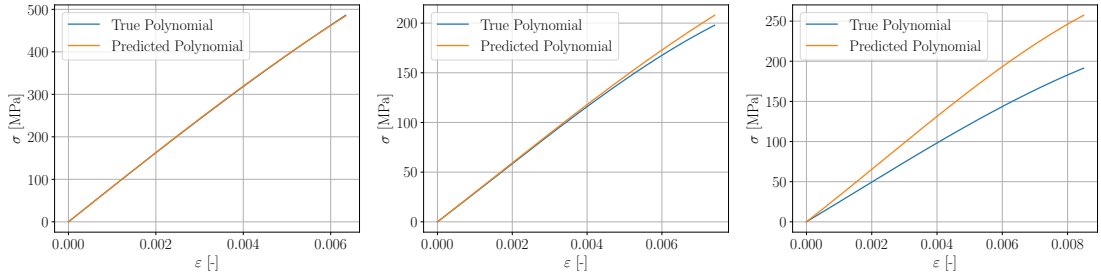
The curve predictors are multi-output regressors. They predict the coefficients of a polynomial that approximates the real stress-strain curve, extracted as shown in Section 4.2.4. Furthermore, they predict the deformation at maximum stress, extracted as shown in Section 4.2.2. These predictors will be evaluated using multiple metrics.

First, for each polynomial coefficient, a RMSE value will be computed. These will have different orders of magnitude, depending on the orders of magnitude of each of the targets' values.

The predicted coefficients will then be used to compute a predicted polynomial curve. This curve will be compared with the real polynomial curve using a Relative Area Error, RAE, defined in Equation (4.16),

$$\text{RAE} = \frac{ABC}{ATC} \tag{4.16}$$

where ABC is the sum of the absolute values of the Areas Between true and predicted Curves, and ATC is the total Area below the True Curve. These areas are computed until the point of true deformation. This means that this metric will not evaluate the predictions of Failure Deformation.



(a) $RAE = 0.67\%$. (b) $RAE = 7.74\%$. (c) $RAE = 34.85\%$.

Figure 4.9: Examples of true/predicted polynomial curve pairs and respective RAE.

To illustrate this metric, in Figure 4.9, three pairs of true and predicted curves, with different RAEs are presented. It is seen that RAE under 10% seem to be reasonably accurate, whereas errors above 30% seem to be unacceptable.

All four target predictions, combined, yield the full stress-strain curve, truncated at the point of Failure Deformation. The stress that derives from the intersection of the predicted curve with the predicted deformation at failure can be computed. This is an indirect way of computing the Ultimate Strength of the laminates. The prediction performance of this Ultimate Strength can then be evaluated in the same way that the Ultimate Strength predicted directly by the Strength Regressors was evaluated. Inclusively, sensitivity analysis and comparison with experimental results can be made.

4.6 Learning Curves

Generating data by using FEM simulations is computationally expensive. Usually, a ML model's performance improves with increased number of training points, but after a given threshold, a plateau is reached, meaning that no significant performance gains are achieved with increased number of training points.

To understand if the size of the generated databases is appropriate, that is, that the plateau has already been reached for the number of points in the database, it is common to make use of learning curves.

These curves plot a performance metric as a function of the number of points used for training.

To make these curves, cross-validation scores will here be used:

1. The full database, with N_{points} is used to train the models using a 5-Fold Cross-Validation strategy.
2. Training scores for each fold are recorded. For each fold, 80% of the N_{points} are used for training, and the remaining 20% for validation.
3. The average value and standard deviation of the 5 recorded values is computed.
4. The process is repeated for different fractions of the database: the 5-Fold CV will be used for training of a model, but using only 10%; 32.5%; 55% and 77.5% of the N_{points}. Points for these fractions are chosen randomly.

Model's have fixed hyperparameters, chosen from a grid search performed on the model trained with the full dataset.

By using this method, five average values and five standard deviations are obtained, corresponding to CV approaches on each fraction of the database. With these, a learning curve can be plotted that accounts for the average of the performance metric, the score, as well as the dispersion of the results, when training on some number of points.

4.7 Importance of the derived features

4.7.1 Database generation

First, to analyse the relevance of the stacking sequence on the simulation results, and therefore the pertinence of using the $\zeta^D_{1,2,3}$ parameters and stacking sequence derived features, a database with fixed geometry was used. This will also serve for demonstration purposes to illustrate the pipeline shown in Section 4.4.2.

It is noted, though, that this database generation does not follow the methodology presented in Section 4.1. Instead, in this database, only six layups were used (chosen manually). These are presented in Table 4.8. The database is composed of 600 points, 200 for each laminate class, termed soft, quasi-isotropic or hard (in each laminate class, half of the laminates have a given NP and the other half a different NP). This terminology is related to the stiffness of the laminates. Each of these data-points corresponds to a stacking sequence that is a permutation of the respective layup. This database shall be termed *DATABASE1*. For demonstration purposes, only the Ultimate Strength under OHT was extracted, as shown in Section 4.2.1.

Within a layup class, the $\zeta^A_{1,2}$ parameters are constant, (even for different NP, because the proportion of plies at each direction is the same), but each permutation will have different $\zeta^D_{1,2,3}$ parameters and different stacking sequence derived features.

No restriction on the permutations was applied, and therefore there exist stacking sequences with maximum ply blocks greater than 2. In more detail, the maximum MPB numbers may be seen in Table 4.9.

In Chapters 5 and 6, to create databases and train models capable of practical use, all the methodology presented in this Chapter will be applied, so that it will be possible to have models with practical usability.

Table 4.8: *DATABASE1* layups summary.

Laminate Class	Total number of Plies	Plies at 0°	Plies at ±45°	Plies at 90°	Number of Stacking Sequence Permutations
Soft	16	2	8	6	100
Soft	32	4	16	12	100
Quasi-Isotropic	16	4	8	4	100
Quasi-Isotropic	24	6	12	6	100
Hard	16	6	8	2	100
Hard	32	12	16	4	100

Table 4.9: *DATABASE1* maximum MPB.

Laminate Class	Total number of Plies	Maximum MPB
Soft	16	6
Soft	32	8
Quasi-Isotropic	16	4
Quasi-Isotropic	24	6
Hard	16	6
Hard	32	8

FEM simulations were ran, with the geometry shown in Table 4.10. The material properties correspond to those of the material IM7/8552. The most relevant, as presented in Section 2.4, are shown in Tables 4.11 to 4.13 (Furtado et al. (2019)). Ply thickness used was t=0.125mm The Ultimate Strength was extracted. The minimum and maximum values of these results for *DATABASE1*, can be seen in Table 4.14.

Having extracted the results, the database was stored as a `Pandas` *DataFrame*, with 600 rows, a column for the layup class, a column for the stacking sequence and a column for each of the features presented in Section 4.3, derived from the stacking sequence.

Table 4.10: *DATABASE1* fixed geometry.

D [mm]	W/D [-]
6	6

Table 4.11: Elastic properties of the IM7/8552 material.

E_1 [MPa]	E_{1c} [MPa]	E_2 [MPa]	ν_{12} [-]	G_{12} [MPa]
171420	150000	9080	0.32	5290

Table 4.12: Strength properties of the IM7/8552 material.

X_T [MPa]	f_{XT} [-]	X_C [MPa]	f_{XC} [-]	Y_T [MPa]	Y_C [MPa]	S_L [MPa]	S_{LP} [MPa]	K_P [-]
2323.5	0.4	1200.1	0.2	62.3	253.7	92.3	66.9	0.05

Table 4.13: Toughness properties of the IM7/8552 material.

G_{XT} [kJ/m^2]	f_{GT} [-]	G_{XC} [kJ/m^2]	G_{IC} [kJ/m^2]	G_{IIC} [kJ/m^2]
133.3	0.3	61	0.79	0.802

Table 4.14: *DATABASE1* results summary.

Laminate Class	Total number of Plies [-]	Minimum Ultimate Strength [MPa]	Maximum Ultimate Strength [MPa]
Soft	16	283.6	355.1
Soft	32	278.2	342.5
Quasi-Isotropic	12	399.2	466.4
Quasi-Isotropic	24	387.2	469.9
Hard	16	495.2	560.4
Hard	32	490.3	551.3

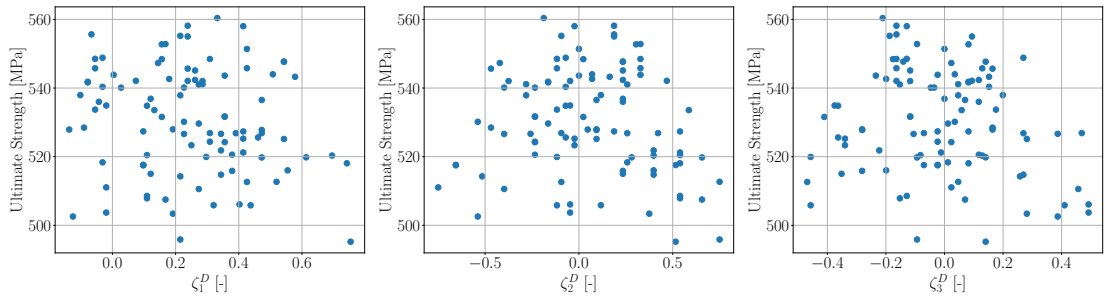
As the number of plies does not make a significant difference in the results, the laminates will from now on be grouped solely as Hard, Quasi-Isotropic and Soft, each of them with 200 points.

It is now possible to plot the ultimate strength as a function of each of the derived features, for each laminate class. In Figures 4.10 to 4.12, some of these plots are presented, in this case for the Hard laminates. Similar trends are found for quasi-isotropic and Soft laminates of *DATABASE1*.

First, in Figure 4.10, it is noticeable the fact that, at least individually, there are no clear trends between any of the $\zeta^D_{1,2,3}$ laminate parameters and its ultimate strength, as was to be expected, as these parameters are usually more related to bending stiffness. It is known though, that combined they carry all the necessary information to identify a particular stacking sequence. This means that a sufficiently complex ML model, with sufficient data to train on should be able to accurately distinguish between stacking sequences with only these parameters. As previously mentioned, feature engineering should help in reducing the need for model complexity and size of training data.

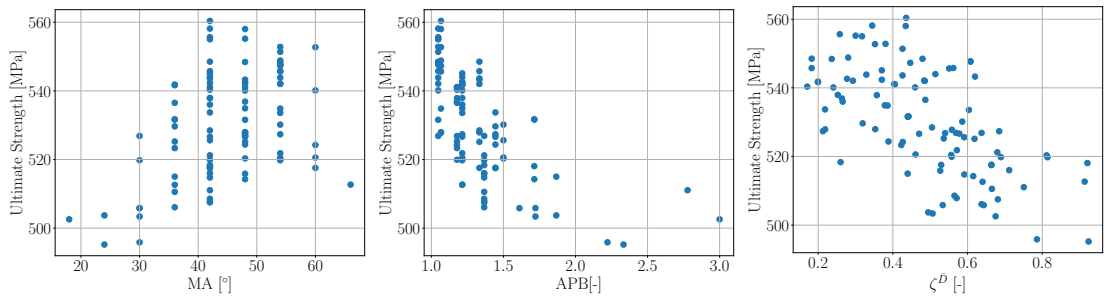
In Figure 4.11, the same type of plots are presented, now for other stacking sequence derived features. Trends are now much more noticeable: Ultimate Strength increases with increasing MA, and decreases with increasing APB and $\zeta^{\bar{D}}$. These clear trends should provide much more useful information when training the ML models.

In Figure 4.12, the trends are further exploited: Ultimate Strength increases with increasing C_1 and decreases with C_3 and C_4. These combinations offer slightly more concentrated clusters of points (more than MA or $\zeta^{\bar{D}}$), arranged more linearly (than for example what happens with APB) and in a continuous fashion (unlike what was seen for MA in Figure 4.11a). These were designed so that model learning was eased even further. C_2 does not show such a clear visible trend, so it shouldn't be more helpful than the $\zeta^D_{1,2,3}$ parameters. It will be kept as a possible feature, tough, as it may be useful to test the feature selection schemes, for example.



(a) (b) (c)

Figure 4.10: Ultimate strength as a function of $\zeta^D_{1,2,3}$, for Hard laminates, *DATABASE1*.



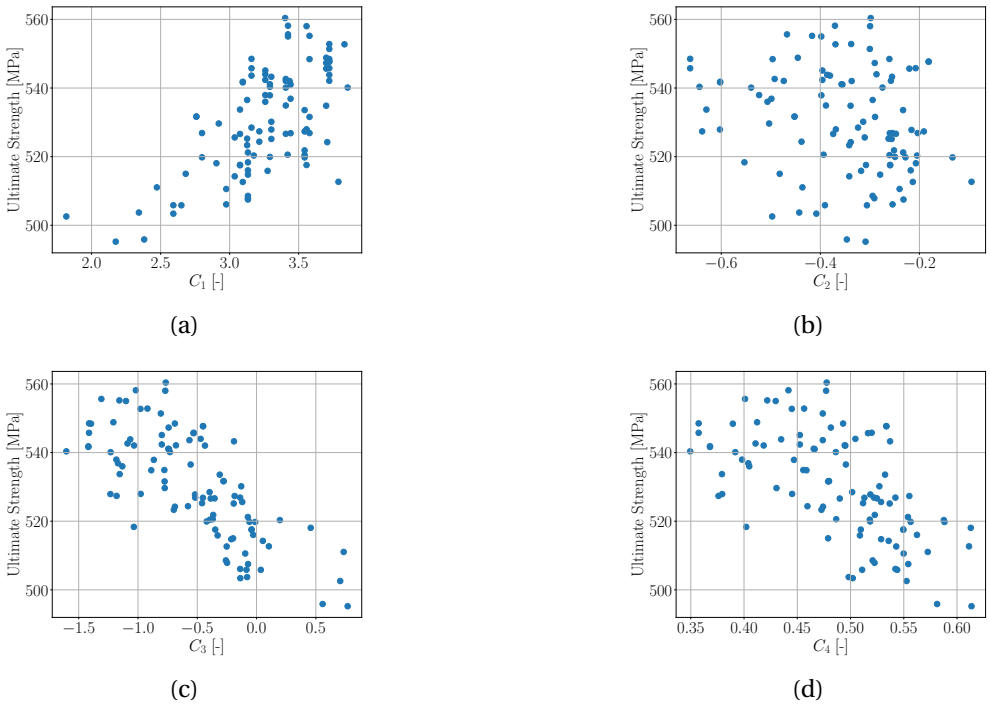
(a) (b) (c)

Figure 4.11: Ultimate strength as a function of MA, APB and $\zeta^{\bar{D}}$ for Hard laminates, *DATABASE1*.

4.7.2 Model's tuning

To train a ML model on *DATABASE1*, for demonstration purposes, the pipeline presented in Section 4.4.2 was used, with the following specifications:

1. Type of algorithm: GP.
2. Class of models: Regression, single output.
3. Target: Ultimate Strength.
4. Outlier detection: SD metric approach, with a threshold of 3.
5. Feature Selection: Performed manually for demonstration purposes. A model was trained using only the ζ_1^A and ζ_2^A parameters; another using these plus the ζ_1^D, ζ_2^D and ζ_3^D parameters; and then another with all the 15 features presented in Table 4.5.
6. Train-Test split: plain split. Test set containing 10% of the data.
7. Cross-Validation strategy: Repeated 5-Fold cross validation, with stratification for feature ζ_1^A, with 3 bins. 3 repetitions.
8. Data normalization: Robust Scaler.



(a) (b) (c) (d)

Figure 4.12: Ultimate strength as a function of $C_{1,2,3,4}$, for Hard laminates, *DATABASE1*.

9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

There were no outliers identified in the database, and therefore all 600 points were used.

In Table 4.15 the hyperparameters chosen by the Exhaustive Grid Search methodology are presented for the trained model.

Table 4.15: Best hyperparameters found through Exhaustive Grid Search for the Ultimate Strength regressor trained on *DATABASE1*.

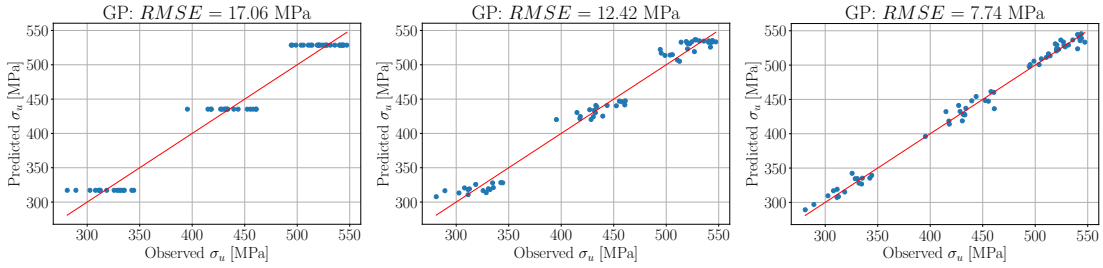
Model	Hyperparameter	Best Choice
GP	kernel type	Mátern
Regressor	kernel parameters	`nu` = 2.5

4.7.3 Test Results

To evaluate the models, the RMSE was used and the "Observed" Ultimate Strength, obtained in the FEM simulations, was plotted against the values "Predicted" by the ML models. Also, the relative error bounds were registered.

These results of model evaluation, with different features given as input, are shown in Figure 4.13 and Table 4.16.

For the model that used only the in-plane laminate parameter, as seen in Figure 4.13a, a RMSE of 17.06 MPa was registered. It is clear that the model cannot distinguish between stacking sequences (they are, of course, only aware of the $\zeta^A_{1,2}$ parameters, solely layup dependent),



(a) Training with only ζ^A parameters.

(b) Training with ζ^A and ζ^D parameters.

(c) Training with all derived features.

Figure 4.13: Results of the trained model on the test set of *DATABASE1*.

Table 4.16: *DATABASE1* Relative Error results.

	Fraction of Data below the RE Limit for each Model			
RE Limit	$\zeta^A_{1,2}$	$\zeta^A_{1,2}$ and $\zeta^D_{1,2,3}$	All	Dummy model
2%	0.400	0.467	0.717	0.133
5%	0.800	0.933	0.967	0.200
10%	0.967	1.000	1.000	0.300
20%	1.000	1.000	1.000	0.583

and so can only distinguish the hard, quasi-isotropic and soft laminate types. The model has learned to predict a value close to the mean of each layup type's ultimate strength. This is the best strategy to minimize the error, with the available information. It is a better strategy than the one used by the *DUMMY* regressor, that always predicts the mean value of the overall training set. There are three distinct values being predicted: one for the soft layup, of 317 MPa; one for the quasi-isotropic layup, of 435 MPa; and another for the hard layup, of 528 MPa. The relative errors are below 20% for all points in the test set, and 80% of the predicted values have a relative error below 5%. Clearly, though, the stacking sequence does impact the ultimate strength observed in the FEM simulation results and, therefore, distinguishing between stacking sequences would be advantageous. For this, when training the models, more input features must be given.

In Figure 4.13b, better results are registered, with RMSE now equal to 12.42 MPa, and all points below a relative error of 10%. 93.3% of the points have a relative error below 5%. In this case, the ζ_D parameters were provided as features and, as these carry information about the stacking sequence of the laminates, the models can now distinguish within a given layup type. Even though this is true, given only these parameters, the models can't always make the best predictions. This could perhaps be improved with larger datasets, as the ζ_D parameters, in theory and combined, carry all the information needed to identify a given stacking sequence. Feature engineering, as was previously explained, can also improve these predictions.

The effect of better suited features can be seen in Figure 4.13c. By using all features presented in Table 4.5, predictions on never before seen data are considerably better. RMSE equal

to 7.74 MPa was achieved, as well as relative errors below 5% for 96.7% of the points. These are very satisfactory results, that showcase the importance of the feature engineering effort that was made.

4.7.4 Automatic Feature Selection

DATABASE1 will here be used to test and demonstrate the use of the Recursive Feature Elimination with Cross-Validation (RFECV), presented in Section 3.5, as a means of automatically selecting the features to be used in training.

To use this scheme, a model that provides feature importance after being fitted with training data is needed. A Random Forest (RF) model will be used, with the default hyperparameters from `SKLearn`.

It can be observed in Figure 4.14 that, upon decreasing the number of training features from 15 to 9, performance, evaluated through the CV Score does not decrease significantly. After that point, the error and the dispersion on results from different folds start to increase very significantly. As the absolute maximum score was obtained for 15 features, though, the RFECV method seems to indicate that the model performs best with all features being used. Some features may be redundant, but seem to not hinder model performance, even slightly helping its capabilities.

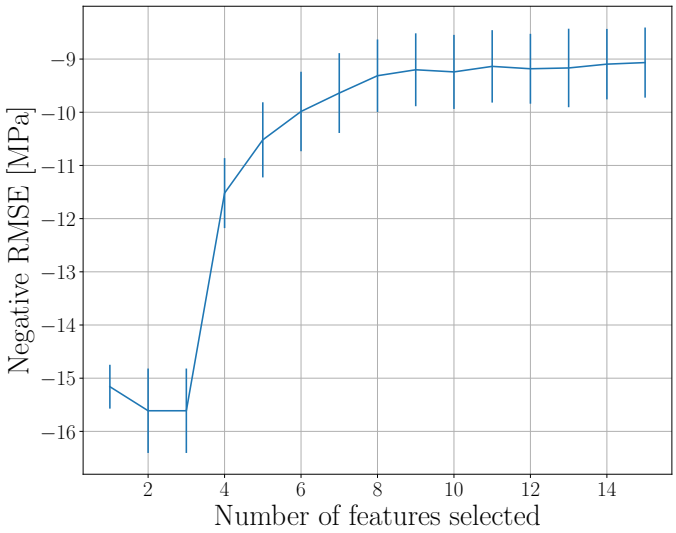


Figure 4.14: Recursive Feature Elimination with Cross-Validation for *DATABASE1*. Variation of RMSE as less important features are eliminated.

4.8 Summary

In this Chapter, the methodology that will be used in subsequent Chapters was presented. The method for Data Generation and Database building; the post-processing of simulations, including the extraction of the relevant output values; the derivation of more informative features from the original input parameters, especially from the layup and stacking sequence;

the pipeline that data will follow before model training, how data is organized and the specifications that should be provided for the definition of a model; how models will be evaluated; and how the size of the generated databases will be evaluated as sufficient or insufficient were explained and detailed. Additionally, a demonstration database, *DATABASE1* was generated and used to show the importance of the engineered features; to illustrate how models will be specified and how regressors are evaluated; and finally the feature selection algorithm through RFECV was tested.

It is important to remember that, although these are useful results, in *DATABASE1* the geometric and material features of the specimens under OHT did not vary, and only three layup types were manually selected.

The objective of the development of the models to follow is for them to have a greater generalization power, for a wider range of geometric, layup and possibly material parameters. Furthermore, models will be trained so that they predict not only the Ultimate Strength, but also the Failure Mode and stress-strain curves under OHT.

Chapter 5

Surrogate models for an open-hole coupon of a constant material system

In this chapter, a database with fixed material properties, variable geometry and variable stacking sequence will be generated and used to train ML models to predict ultimate strength, failure mode and stress-strain curves for specimens under OHT. Therefore, the models are specialized for the chosen material, not being able to make predictions for different properties, but are expected to be accurate when used on different geometric and stacking sequence inputs.

5.1 Dataset generation

5.1.1 Design of Experiments

The generation of this database followed the procedure outlined in Section 4.1.

The geometric parameters were sampled in the ranges shown in Table 5.1. The laminate parameters were sampled as explained in Section 4.1.1: $\zeta^A_{1,2}$ sampled in range [-1,1] but forced to the admissible laminate space; $\zeta^D_{1,2,3}$ resulting from the shuffling of the selected layups to result in different stacking sequences, with a limit *MPB* of 2, for more realistic laminates. This means that all laminates will have $MPB = 2$ (as they are symmetric there is always a ply block of two in the symmetry plane), and therefore *MPB* will not be a useful feature at all (it has in this database a variance of 0).

The material properties of IM7/8552 material, shown in Tables 4.11 to 4.13, are considered.

The Design of Experiments resulted in a total of $N_{points} = 1051$.

Table 5.1: Range of sampling of the geometric features in *DATABASE2*.

	D [mm]	W/D [-]
Min. Value	2	3
Max. Value	10	8

This database will be referred to as *DATABASE2*.

5.1.2 FEM Simulations

FEM simulations were ran as described in Section 4.1.2. Details on the computational costs for the generation of this database may be seen in Table 5.2[5]. This resulted in total on more than 11 days to generate the database.

Table 5.2: Computational costs of running the numerical simulations.

Number of Simulations	Average running time [min]	Average post-processing time [min]
1051	14.04	1.43

One data-point had a null value of ultimate strength. The existence of this result is attributed to a numerical issue in data generation, and it was therefore removed from the dataset. This resulted in a total of 1050 useful data-points in *DATABASE2*.

Results were post-processed as presented in Section 4.2. All features were derived as shown in Section 4.3, and the post processed results were organized in a processed *DataFrame*, as presented in Section 4.4.1.

A summary of these outputs is provided in Tables 5.3 to 5.5. We notice in Table 5.4, where FM1 represents fiber dominated Failure Mode and FM2 means matrix dominated Failure Mode, as mentioned in Section 4.2.3, that the dataset is very unbalanced when it comes to failure mode, with fiber dominated failure being much more frequent. This expected and is an accurate representation of the population. It is also evident from Figure 5.1 that matrix dominated failure occurs on laminates with more 90° and ±45° laminates (that have lower values of ζ_1^A).

Table 5.3: Ultimate Strength results in *DATABASE2*.

Minimum Ultimate Strength [MPa]	Maximum Ultimate Strength [MPa]	Average Ultimate Strength [MPa]	Standard Deviation of Ultimate Strength [MPa]
91.90	819.34	399.49	163.11

Table 5.4: Failure Mode results in *DATABASE2*.

	Fiber dominated failure (FM1)	Matrix dominated failure (FM2)
Number of points	934	116
Percentage	88.95%	11.05%

[5]Average time was computed by recording duration time of the first 500 simulations, for which it was possible to control the used computational resources. 24 CPUs were used to run the FEM simulations. A single CPU was used for post-processing.

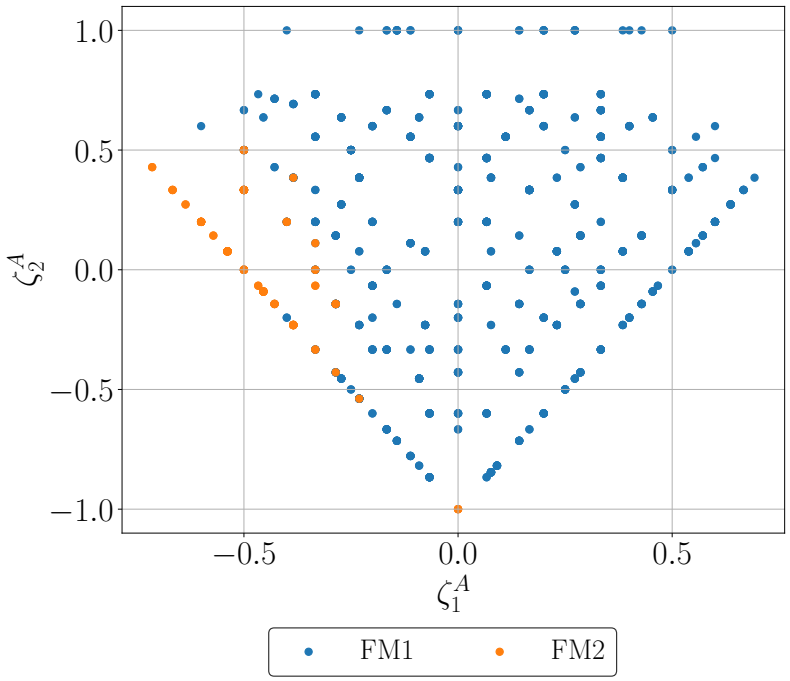


Figure 5.1: Failure Mode of points presented in the $\zeta_{1,2}^A$ space for *DATABASE2*.

Table 5.5: Failure Deformation results in *DATABASE2*.

Minimum Failure Deformation [-]	Maximum Failure Deformation [-]	Average Failure Deformation [-]	Standard Deviation of Failure Deformation [-]
4.86E-03	18.73E-03	7.34E-03	1.16E-03

In each of the following Sections, a class of ML model will be specified, built, trained and evaluated: strength regressors, failure mode classifiers and curve predictors.

5.2 Strength Regressor

Four models, each using a different ML algorithm will be trained to predict a single value: the Ultimate Strength of the OHT specimen. Model preparation will follow the outline presented in Section 4.4.2.

5.2.1 Models' tuning

In particular for these models, the following details have been used:

1. Type of models trained: NN; LGBM; GP; BM.
2. Class of models: Regression, single output.
3. Target: Ultimate Strength, σ_u.
4. Outlier detection: MAD metric approach, with a threshold of 2.5.
5. Feature Selection: RFECV, with a base RF regressor (default hyperparameters). CV Score: RMSE.
6. Train-Test split: stratified split. Stratification for feature "W/D" with 3 bins. Test set containing 10% of the data.
7. Cross-Validation strategy: 5-Fold cross validation, with stratification for feature "W/D", with 3 bins.
8. Data normalization: Robust Scaler, for the GP; NN regressors. No normalization for the LGBM and BM regressors.
9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

The outlier detection method identified 38 outliers, all with a MAD score above 2.5. This decreases the total number of available data-points for the training and testing of this regressor from 1050 to 1012. These eliminated laminates are considered to be not frequent enough in the population. This process is presented in Figure 5.2, where the points to be used are at the left hand side of the red line. Despite this slight decrease in available data, model performance should be enhanced because of this processing phase. It is interesting to note that, in the case of an SD method having been applied with a threshold of 3.0 or even 2.5, no outlier would have been identified.

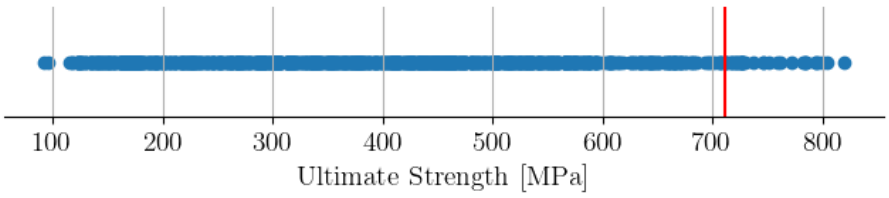
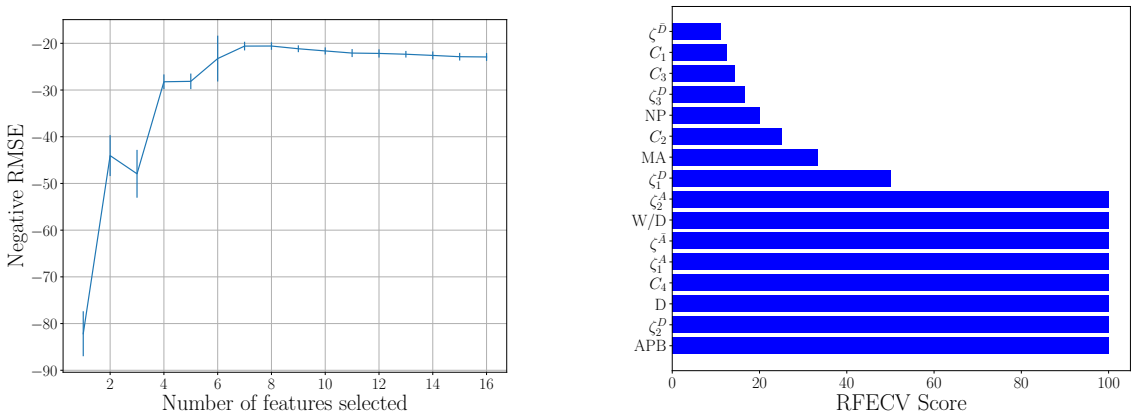


Figure 5.2: Outlier detection using MAD method, threshold = 2.5, on the Ultimate Strength feature in *DATABASE2*.

For the Feature Selection method by RFECV, an RF regressor was used. This choice was mainly based on its simplicity, and associated low costs when fitting; its easy interpretability; and it was also chosen in order to avoid bias towards one of the four models that will be trained (for example, the FS could be biased towards the LGBM model had we used a LGBM model in FS). It was deemed that the method using this simple ML model, and without hyperparameter selection (because there is no interest in great performance of the model, but only on the relative feature importance on its predictions), was adequate for FS in this problem.

Figure 5.3a clearly shows that performance (evaluated through the RMSE CV score), slightly increases when reducing the number of used features from 16 to 8, as redundant and irrelevant features are eliminated. Below 7 features, performance is rapidly lost: the model doesn't have enough information. It can also be seen that for a number of features above 7, the dispersion in results for the five folds is very low, allowing for the inference that results of this type of regression should be consistent for different data splits.

Figure 5.3b shows the 8 most important features, those with the highest RFECV score. The model has identified the geometric features (D and W/D); the in-plane laminate parameters ($\zeta^A_{1,2}$, $\zeta^{\bar{A}}$); and some features related to ply order (APB; ζ^D_1, C_4) as the essential ones. All other 8 features were considered redundant or irrelevant and therefore will not be used as inputs for prediction.



(a) Variation of RMSE as less important features are eliminated.

(b) Best features chosen by using the RFECV method.

Figure 5.3: Recursive Feature Elimination with Cross-Validation for the Ultimate Strength regression problem in *DATABASE2*.

The train-test split with a ratio of 10% results in 101 data-points in the test set and 911 in the training set.

The stratification feature W/D was chosen for both the train-test split and the 5-fold CV split. Further analysis suggests that this decision does not influence the results significantly. As presented in Appendix C.1, single feature stratified folds do not seem to improve performance regardless of the chosen feature, number of bins, number of repetitions, even when compared to a plain split, without stratification. A more complex approach, considering multiple features, as referenced in Section 3.6.1, could provide better results. This is yet to be implemented.

Regarding data normalization, as mentioned in Section 3.3, such pre-processing is not nec-

essary for the tree based models, as it does not impact the models' performance in any way. For the GP and NN models, on the other hand, this is a very important operation. The Robust Scaler was chosen based on its robustness to possible outliers, even though outlier detection had already been performed.

In Table 5.6, the hyperparameters chosen by the Exhaustive Grid Search methodology are presented for each of the four trained models. All other hyperparameters of the models were left to their default values, as attributed in the `SKLearn` library.

Table 5.6: Best hyperparameters found through Exhaustive Grid Search for the Ultimate Strength regressors trained on *DATABASE2*.

Model	Hyperparameter	Best Choice
GP Regressor	kernel type	Mátern
	kernel parameters	nu = 1.5
NN Regressor	`alpha`	0.1
	`batch_size`	25
	`hidden_layers`	(32,32)
	`activation`	'relu'
	`learning_rate_init`	-
	`max_iter`	1000
	`optimizer`	'lbfgs'
LGBM Regressor	`colsample_bytree`	0.8
	`subsample`	1.0
	`min_child_samples`	5
	`learning_rate`	0.1
	`max_depth`	-1
	`num_leaves`	5
	`n_estimators`	800
BM Regressor	`max_features`	1.0
	`max_samples`	1.0
	`n_estimators`	800

5.2.2 Models' evaluation

After hyperparameter selection and model training, the four models are evaluated on never before seen data, using the test set.

In Figure 5.4, the predicted and numerical FEM simulation results of the Ultimate Strength, σ_u, for the laminates in the test set are plotted, and the RMSE for each of the four models is presented. The red lines represent what would be a perfect regressor, where predicted and observed values would match in 100% of the cases.

In Figure 5.5, the relative errors are presented for the four models, in the shape of an histogram. These results are also reported in Table 5.7: in this table, the fraction of the test set with a registered absolute value of relative error below the thresholds of 20%, 10%, 5% and 2% are presented for the four models and also for a *DUMMY* model (that always predicts the mean Ultimate Strength of the data-points in the training set), for comparison purposes.

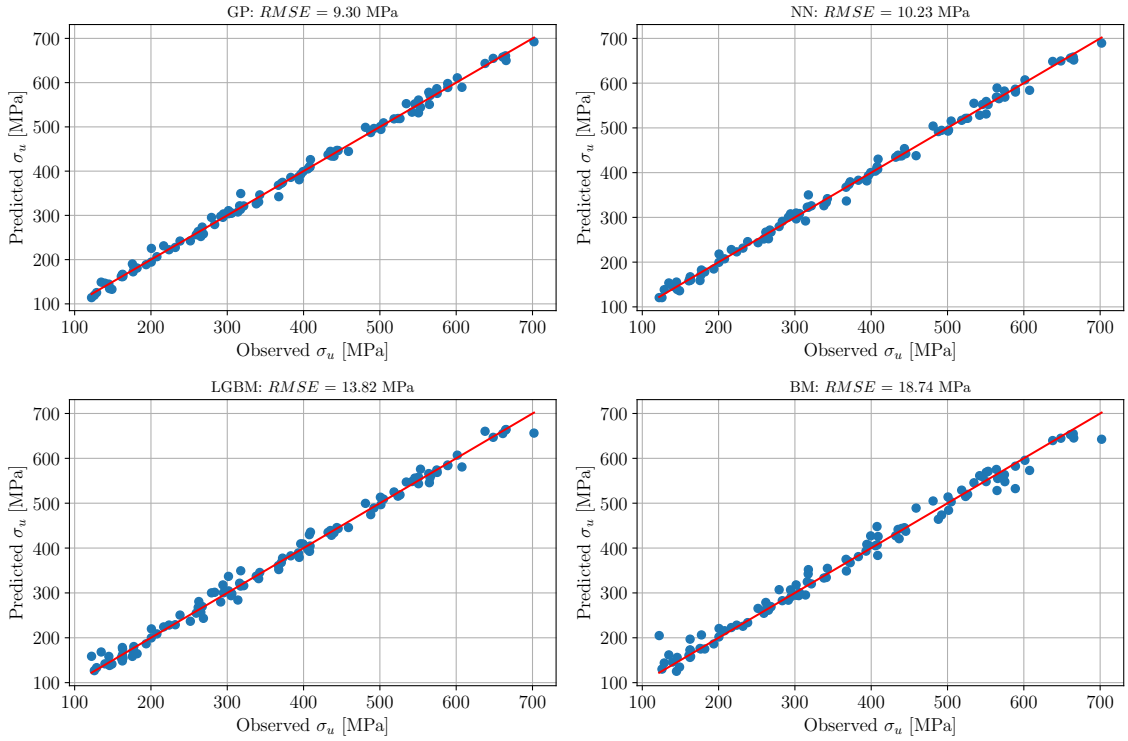


Figure 5.4: Observed (from numerical simulations) and predicted Ultimate Strength for the four ML Ultimate Strength regressors trained on *DATABASE2*.

Table 5.7: Relative Error (RE) fractions below thresholds of 2%, 5%, 10% and 20% for the four ML Ultimate Strength regressors trained on *DATABASE2*.

	Fraction of Data below the RE Limit for each Model				
RE Limit	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor	*DUMMY* Regressor
2%	0.618	0.559	0.490	0.412	0.029
5%	0.882	0.892	0.755	0.735	0.108
10%	0.971	0.980	0.971	0.922	0.137
20%	1.000	1.000	0.980	0.971	0.245

When evaluated on the test set, the regressors registered low RMSEs: the GP model being the one that obtained the best test score (with RMSE below 10MPa), followed by the NN model, as seen in Figure 5.4. As seen in Figure 5.5 and Table 5.7, both models had RE lower than 20% for all points and RE lower than 10% for over 97% of the test data-points. The NN model outperforms the GP model when analysing the 5% and 10% RE thresholds, but the GP model has 61.8% of the test points predicted with RE lower than 2%, whereas for the NN model the percentage of points that verify this criterion is of 55.9%. This is what ultimately results in a lower RMSE for the GP model. The tree based models have worse test scores. It is interesting to compare these results with those of the *DUMMY* model: if one were to naively guess the strength of the specimens, an error below 20% would be achieved only 24.5% of the time, not remotely close to the performance of the ML models.

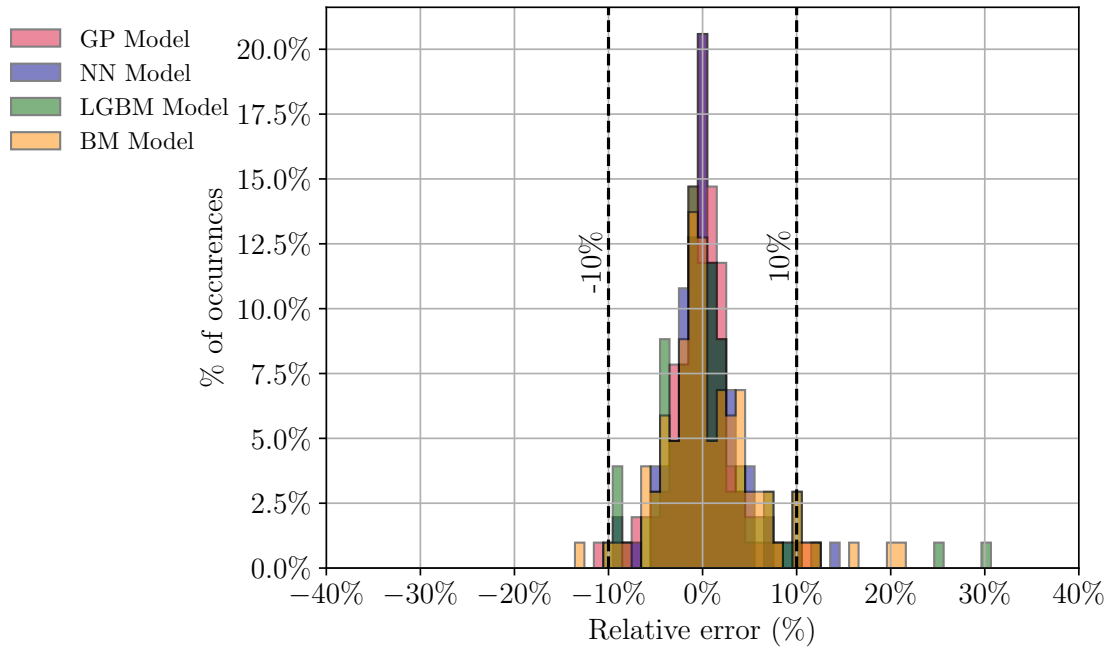


Figure 5.5: Histogram of Relative Errors for the four ML Ultimate Strength regressors trained on *DATABASE2*.

5.2.3 Learning curves

As detailed in Section 4.6, a way of assessing if the database has enough data-points is by means of plotting learning curves. Here this was done using RMSE as the scoring metric.

The initial dataset from which the N_{train} data-points are sampled had already been cleaned of outliers, using the same MAD method with a threshold of 2.5. For this analysis, all 16 features were used, and not only the ones selected by FS. This was done because absolute model performance isn't the priority. The hyperparameters of the models were fixed, and are the ones that were presented in Table 5.6. For the CV splits, a StratifiedKFold approach was used, with 5 folds; the stratification feature being W/D, with 3 bins.

The learning curves are presented in Figure 5.6, for the four model types.

It was shown that the amount of data in *DATABASE2* is sufficient for the GP model: this already obtained a very low CV RMSE score when trained with less than 500 data-points. On the other hand, the NN, LGBM and BM models were still seeing significant improvements on their performance even after being trained with more than 800 data-points, as seen by the slope of their learning curves at that point.

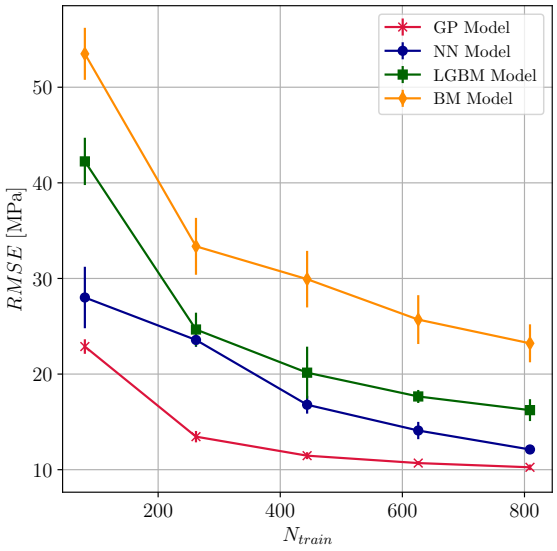


Figure 5.6: Learning curves for Ultimate Strength regressors trained on *DATABASE2*, using RMSE score.

5.2.4 Sensitivity analysis

To evaluate the ability of the models to capture the trends in the strength of the OHT specimens, a sensitivity analysis was performed.

Here the values of the features D, W/D, $\zeta^A_{1,2}$, Number of Plies and $\zeta^D_{1,2,3}$ varied within their respective ranges, one at each time. While varying one parameter, all the others were fixed. When varying the ζ^A parameters, it was inevitable to also vary the NP and ζ^D parameters, but these variations were kept to the minimum within what is possible. The inputs for this sensitivity analysis may be seen in Appendix B. Results are presented in Figures 5.7 and 5.8.

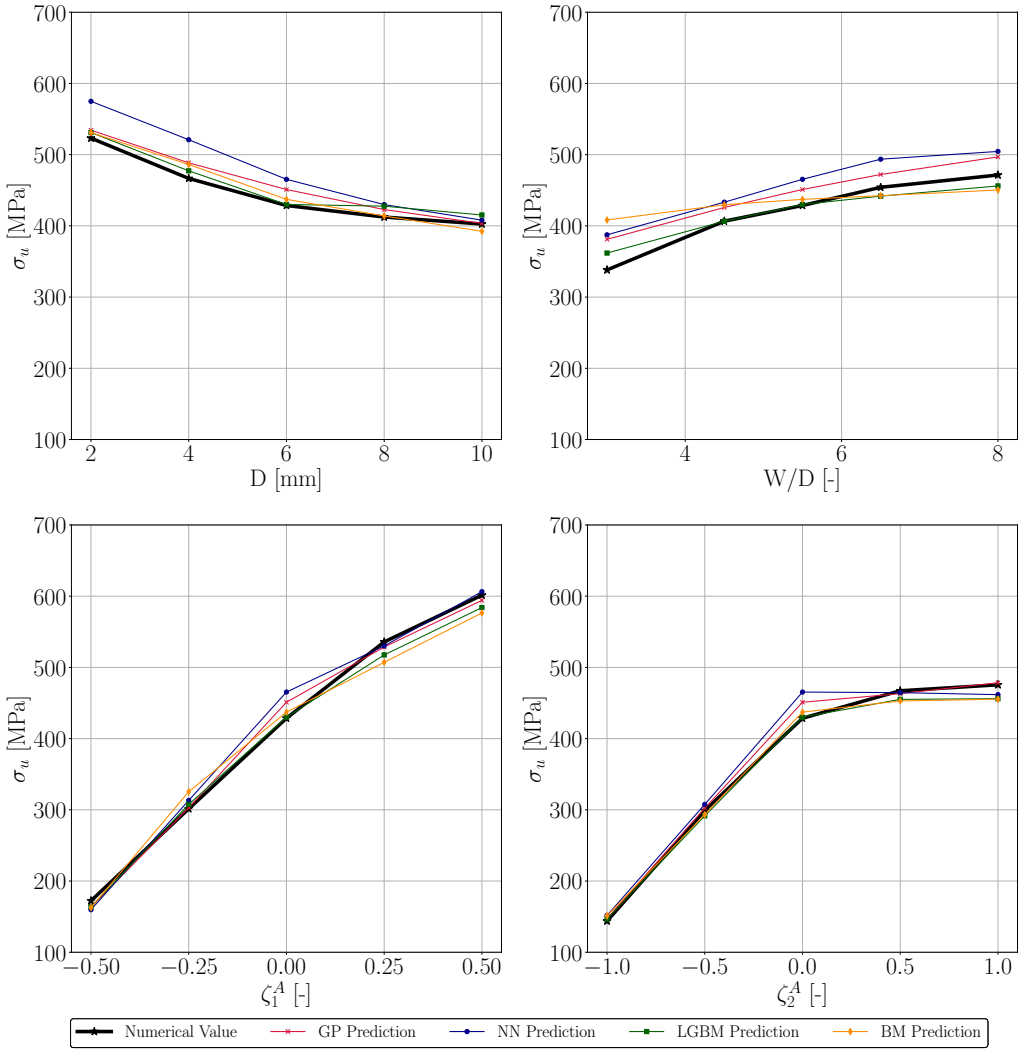


Figure 5.7: Results of the sensitivity analysis of Ultimate Strength regressors trained on *DATABASE2* for geometric features and in-plane laminate parameters.

This sensitivity analysis first provided some insight on the influence that each feature has on the real strength of the specimens: ζ^A_1, when varying from -1 to 1, changes the Ultimate Strength from less than 200MPa to over 600MPa, as observed by analysing the numerical re-

sults. ζ_2^A, when varying from -1 to 0 changes the Ultimate Strength from less than 150MPa to over 400MPa, but when varying from 0 to 1 a plateau is seen: Ultimate Strength still increases, but only by approximately 50MPa. Regarding the geometric features, their variation on the analysed ranges corresponds to changes of over 100MPa in Ultimate Strength: strength decreases with increasing hole diameter, and increases with increasing W/D ratio, as expected.

The analysis shows that all models were able to capture these tendencies in the results. In this case, it is interesting to note, though, that the DT based models (LGBM and BM), outperform the other two models in the error of their predictions.

Regarding the sensitivity of the models in relation to NP and $\zeta_{1,2,3}^D$ parameters, it must be said that trends between these and the Ultimate Strength are not as straightforward as the previous ones, and the ML models were not able to capture them, in general, as seen in Figure 5.8.

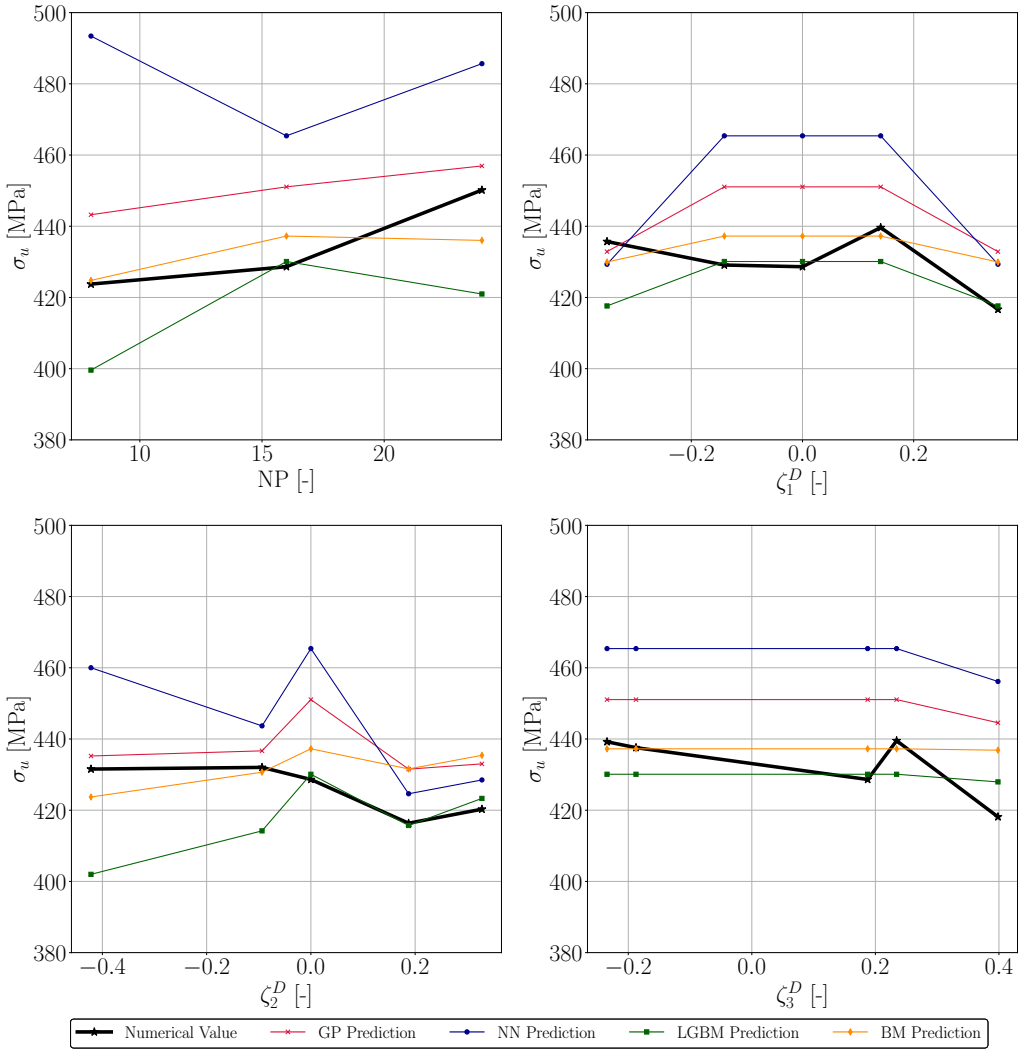


Figure 5.8: Results of the sensitivity analysis of Ultimate Strength regressors trained on *DATABASE2* for number of plies and out-of-plane laminate parameters.

5.2.5 Validation with experimental results

Camanho et al. (2007) presents results for experimental OHT tests on five specimens, all using the IM7/8552 material and a stacking sequence [90, 0, 45, -45]$_{3S}$.[6] The difference between the five specimens is their diameter D, that varies linearly from 2 to 10. W/D = 6 is kept constant. The results of this experimental campaign, numerical FEM simulations using the methodology of Furtado et al. (2019), and the predictions from the four ML models are presented in Figure 5.9.

Relative Errors (RE) of the predictions in relation to the numerical and experimental results are reported in Table 5.8.

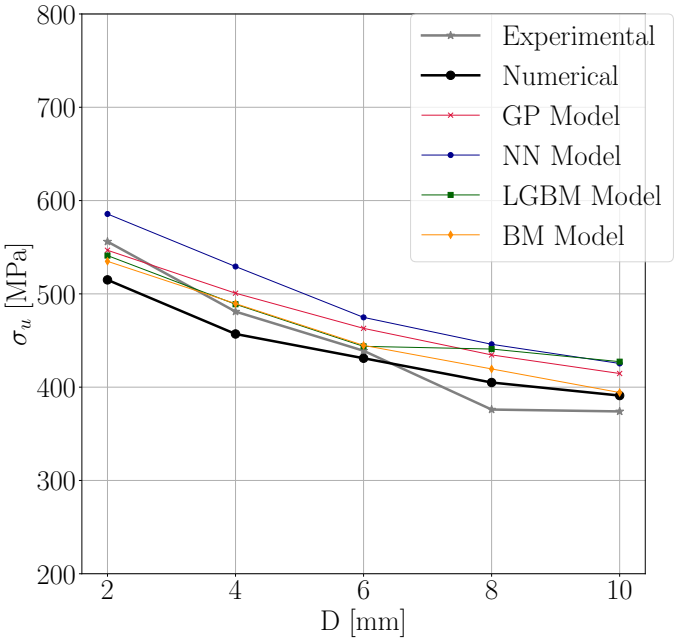


Figure 5.9: Comparison of the experimental results from Camanho et al. (2007) with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE2*, as a function of hole diameter.

Xu et al. (2016) also presents an interesting and similar study, for seven IM7/8552 laminates. Diameter varied from 0.5 mm to 50.8 mm. Although some of these diameters are outside of the range with which the ML models were trained, it is still interesting to analyse how the models behave in these regions: their extrapolation capabilities.

A layup [45,90,-45,0]$_{4S}$ was kept constant. The relation W/D was kept constant and equal to 5, except for the laminates with smaller diameters: W/D = 32.26 and W/D = 16.13 were used for the specimens with D = 0.5 mm and D = 1.0 mm, respectively. These last two W/D ratios are also outside the training range of the ML models.

The results for all seven cases are presented in Figure 5.10: the experimental results, the results from numerical simulations using the methodology from Furtado et al. (2019) and the

[6] A subscript "$_{NS}$" indicates that the angles presented repeat themselves N times and to the resulting stacking sequence a symmetric repetition should be applied to obtain the complete stacking sequence.

Table 5.8: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE2)* relative to numerical and experimental (in parenthesis) results on the specimens from Camanho et al. (2007).

D [mm]	σ_u (Exp.) [MPa]	σ_u Num. [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
2	515	556	6.2 (-1.7)	13.7 (5.3)	5.1 (-2.7)	3.9 (-3.8)
4	457	481	9.6 (4.1)	15.8 (10.0)	7.0 (1.6)	7.1 (1.8)
6	431	439	7.4 (5.5)	10.2 (8.2)	3.0 (1.1)	3.2 (1.3)
8	409	376	6.3 (15.6)	9.0 (18.6)	7.8 (17.3)	2.6 (11.6)
10	391	374	6.1 (10.9)	8.8 (13.7)	9.3 (14.3)	0.8 (5.4)

predictions from the four ML models.

Relative Errors (RE) of the predictions in relation to the numerical and experimental results are reported in Table 5.9 for all seven cases.

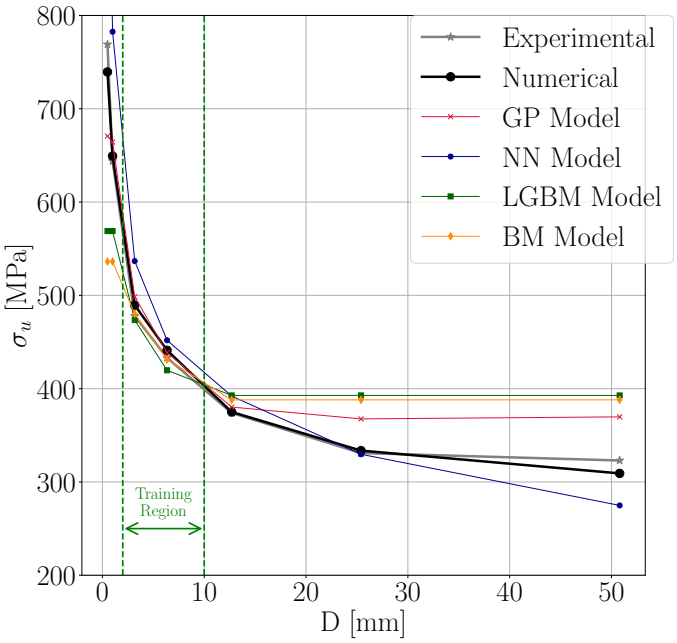


Figure 5.10: Comparison of the experimental results from Xu et al. (2016) with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE2*, as a function of hole diameter.

Clarkson (2012) presents results for three examples, in which the geometric features were kept constant, but the stacking sequences vary. The material is the IM7/8552; geometric features are D = 6.35 mm and W/D = 6. The stacking sequences used were the following:

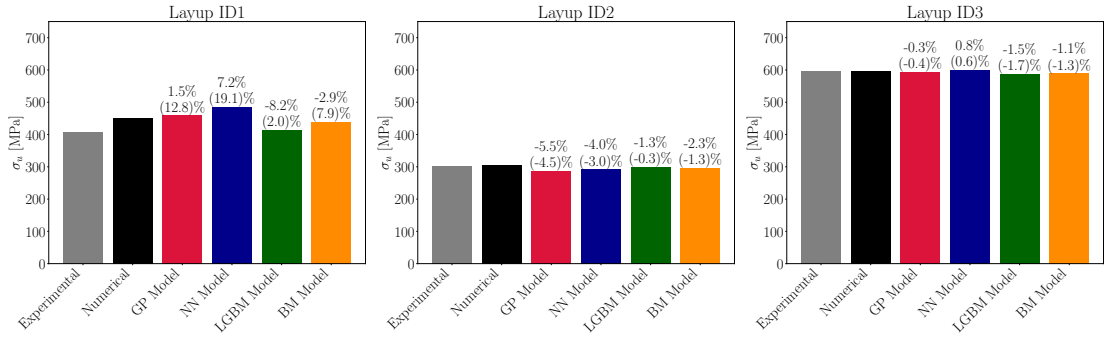
- Layup ID1: $[45, 0, -45, 90]_{2S}$;
- Layup ID2: $[45,-45, 0, 45, -45, 90, 45, -45, 45, -45]_S$;

- Layup ID3: $[0, 45, 0, 90, 0, -45, 0, 45, 0, -45]_S$.

Table 5.9: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE2)* relative to numerical results on the specimens from Xu et al. (2016).

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
0.50*	739	769	-9.3 (-12.8)	42.1 (36.6)	-23.1 (-26.0)	-27.5 (-30.3)
1.00*	649	644	2.3 (3.2)	20.5 (21.5)	-12.4 (-11.7)	-17.4 (-16.7)
3.18	490	478	1.8 (4.3)	-9.6 (12.3)	-3.3 (-0.9)	-1.9 (0.5)
6.35	441	433	-0.6 (1.3)	2.4 (4.4)	-4.9 (-3.1)	-2.2 (-0.4)
12.70*	375	374	1.4 (1.7)	4.5 (4.8)	4.7 (5.0)	3.5 (3.7)
25.40*	334	331	10.2 (11.1)	1.2 (-0.4)	17.7 (18.7)	16.3 (17.2)
50.80*	309	323	19.6 (14.5)	-11.1 (-14.9)	27.0 (21.6)	25.5 (20.1)

Results are presented in Figure 5.11, where the numerical results were obtained using the methodology from Furtado et al. (2019). On each model's bar, the RE in relation to the numerical and experimental results are reported.



(a) (b) (c)

Figure 5.11: Comparison of the experimental results from Clarkson (2012) with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE2*, for three different layups. Relative Error presented in relation to numerical and experimental (in parenthesis) results.

The ML models were put to the test on real experimental results. Their predictions were compared to these results and also with numerical predictions for the same specimens. Overall, the ML based predictions were accurate, especially when compared to numerical results.

Regarding the data from Camanho et al. (2007), the ML models captured once again the tendency for a decrease in strength for increasing hole diameter, as seen in Figure 5.9 (this is a different stacking sequence from the one seen in the sensitivity analysis). Here, the GP and BM models were the best performing, with RE lower than 10% for all points when compared to

*Outside of training range. Training range is D∈[2, 10] mm.

the numerical results, and RE larger than 10% when compared to experimental tests only when the numerical simulations deviated substantially from the experimental results, as in the point with a diameter of 8mm, as was shown in Table 5.8.

Data from Xu et al. (2016) is useful not only for a similar analysis as the one done for Camanho et al. (2007) data, but also to analyse the extrapolation potential of the ML models. In Figure 5.10, it is very interesting to note the capability of the NN model to predict for inputs well outside of its training range, as are the diameters of 25.40 mm and 50.80 mm, for which it obtained RE of 1.2% and 11.1% (when compared to numerical results), well below the prediction errors committed by the other models. This is due to its nature: in the case of the DT based models, predictions are constant outside the training range, as the trees aren't capable of imposing splitting conditions for values outside of those with which they were trained. This explains the flat lines for the LGBM and BM models for the diameters below 2 mm and above 6 mm. The GP model does not present a completely flat line outside its training range, but errors committed are very large in the extrapolation zone. It is observed, though, that, although all models capture a correct trend in data, the NN model's predictions are worse than those of the other models within the interpolation range.

Data from Clarkson (2012) gives confidence in the ML model's ability to predict Ultimate Strength for different layups. Errors below 10% were obtained for all models, when compared to the numerical results. Here the GP model was once again the best performing model, as seen in Figure 5.11.

It is important to note that the errors of model predictions when compared to experimental results were tendentiously larger than the same but taking the numerical results as reference and often RE were above 10%. This was to expect: the fact that the error between numerical and experimental results is not negligible would always mean that the ML predictions, trained on numerical simulations, would be further away from experimental results than what would be ideal. Were the models trained on numerical simulations with greater accuracy, this error would be diminished. The overarching objective should be to have numerical simulations as close as possible to the real results (which is outside the scope of this work) and have ML predictions as close as possible to these numerical results (which is the purpose of this work), so that eventually the ML predictions accurately approximate the real experimental results. Occasionally, the RE relative to experimental results is lower than the RE relative to numerical results, but this happens only when the numerical model is overestimating strength and the ML model underestimates this prediction (or vice-versa).

5.3 Failure Classifier

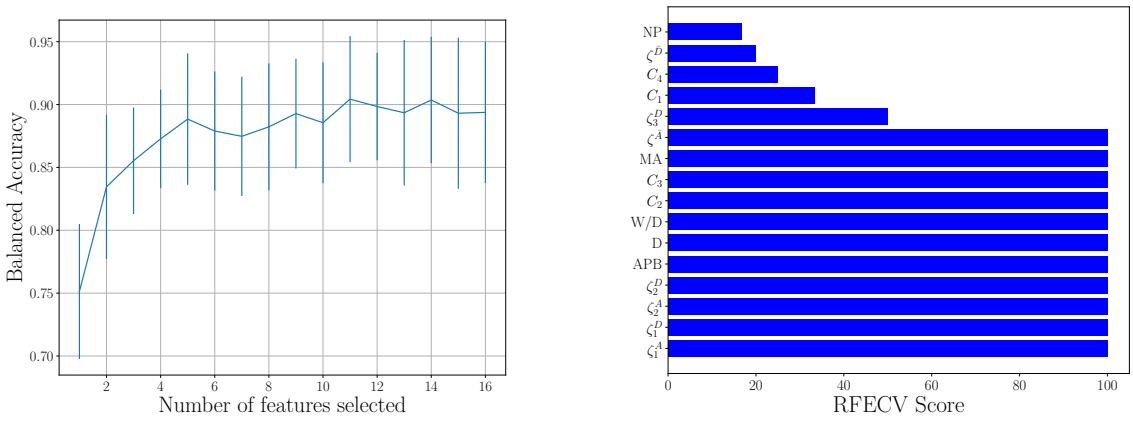
The models trained in this Section will use the available features as inputs, to predict if the failure mode of the laminates under OHT is fiber or matrix dominated. Failure mode is classified as seen in Section 4.2.3.

5.3.1 Models' tuning

1. Type of models trained: GP; NN; LGBM; BM
2. Class of models: Binary Classification, single output.
3. Target: Failure Mode.
4. No outlier detection was used.
5. Feature Selection: RFECV, with a base RF regressor (default hyperparameters). CV Score: Balanced Accuracy.
6. Train-Test split: Stratified split: stratification for the target Failure Mode. Test set containing 20% of the data.
7. Cross-Validation strategy: 5-Fold cross validation, with stratification for the target Failure Mode.
8. Data normalization: Robust Scaler, for the GP; NN regressors. No normalization for the LGBM and BM regressors.
9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

The reason for the use of no outlier detection method for this problem is its binary classification nature and the fact that the methods implemented, as presented in Section 3.4, are based solely on the output values.

By using the RFECV as the FS method, the ideal number of features to be used was 11, as seen in Figure 5.12. It is worth noting, by analysing Figure 5.12a, that in this classification problem the models performance has a lot more variance from fold to fold, regardless of the number of features being used. In the previously presented regression problem, as seen in Figure 5.3a, it was evident that this dispersion only occurred when using not enough features. Nevertheless, the 11 features with a score of 100, presented in Figure 5.12b were selected, and these comprise both geometric features, all in-plane laminate parameters and some features related to ply order.



(a) Variation of Balanced Accuracy as less important features are eliminated.

(b) Best features chosen by using the RFECV method.

Figure 5.12: Recursive Feature Elimination with Cross-Validation for the Failure Mode classification problem in *DATABASE2*.

Table 5.10: Best hyperparameters found through Exhaustive Grid Search for the Failure Mode classifiers trained on *DATABASE2*.

Model	Hyperparameter	Best Choice
GP Classifier	kernel type	Mátern
	kernel parameters	nu = 1.5
NN Classifier	alpha	0.001
	batch_size	25
	hidden_layers	(64,32,16)
	activation	'tanh'
	learning_rate_init	0.1
	max_iter	1000
	optimizer	'adam'
LGBM Classifier	colsample_bytree	1.0
	subsample	0.1
	min_child_samples	1
	learning_rate	0.1
	max_depth	10
	num_leaves	20
	n_estimators	100
BM Classifier	max_features	0.5
	max_samples	1.0
	n_estimators	600

The stratification rule based on the target is here essential, both for the test-train split and for the cross-validation folds, as the unbalanced population must be correctly represented in

the splitted data.

In Table 5.10 the hyperparameters chosen by the Exhaustive Grid Search methodology are presented for each of the four trained models. All other hyperparameters of the models were left to their default values, as attributed in the `SKLearn` library.

5.3.2 Models' evaluation

The four ML classifiers will now be evaluated on never before seen data, stored in the test set. As mentioned, the classifiers predict if the laminates Failure Mode is fiber dominated (FM1) or if it is not (FM2).

In Figure 5.13, the confusion matrices are presented for all four models. These show the percentages of TP, TN, FP and FN for the predictions on the test set, as explained in Section 4.5.2.

In Figure 5.14, two metrics are presented for the classifiers: both their accuracy and balanced accuracy, measured on the test set.

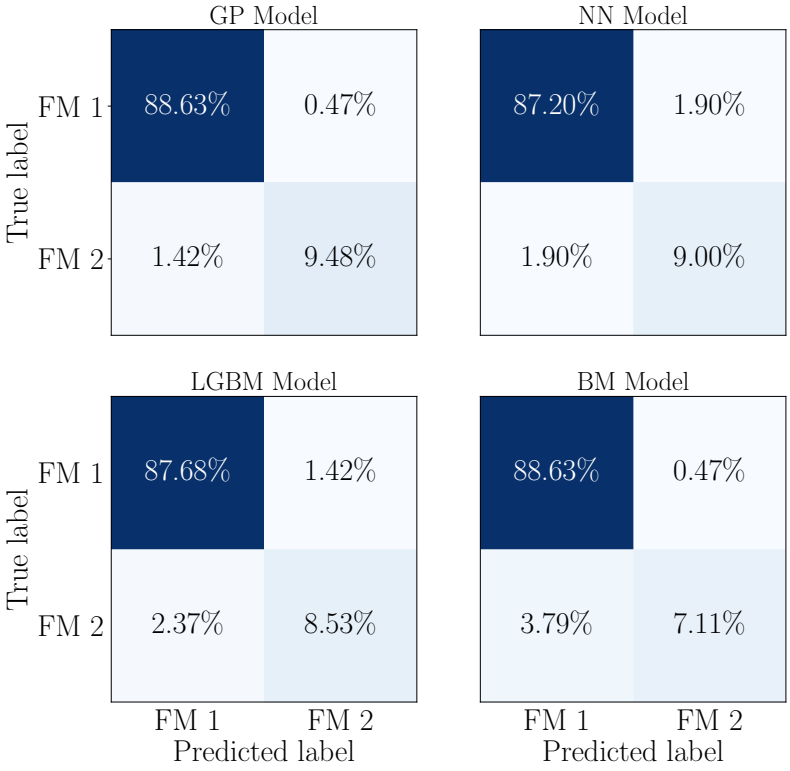
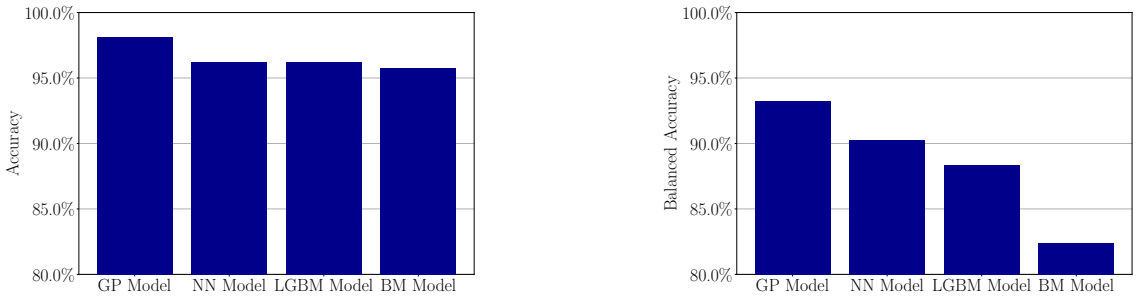


Figure 5.13: Confusion Matrices for the four models trained on *DATABASE2* for the Failure Mode classification problem.

It is shown that the models only miss their predictions in less than 5% of the test cases. The most accurate model was the GP classifier, with an accuracy on the test set of over 98%, as evident in Figure 5.14a.

As the databases are unbalanced, due to the tendency of specimens to have fiber dominated failure, the Balanced Accuracy was deemed a very useful metric, even being used for model scoring during Cross-Validation. It is important to reiterate that a *DUMMY* model, that always

predicts FM1, would be correct in 88.95% of the cases (corresponding to the unbalance in data). Such model would have a balanced accuracy of 50%.



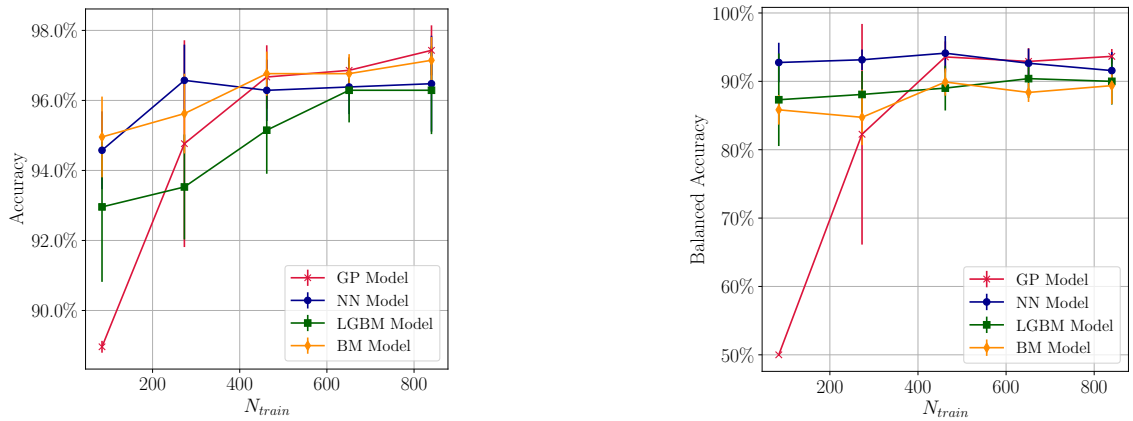
(a) (b)

Figure 5.14: Accuracy (a) and Balanced Accuracy (b) scores for the Failure Mode classifiers trained on *DATABASE2*.

All ML classifiers obtained a balanced accuracy test score of over 80%, as seen in Figure 5.14b. The GP model was once again the best performing, reaching a Balanced Accuracy of 93.21%. This is because it missed 13.03% of the data-points that are truly labeled as having FM2, and only 0.53% of the data-points truly labeled as FM1. (Note that ((100%-13.03%) + (100%-0.53%))/2 = 93.21%). The worst performing model, BM, misses 34.8% of the data-points truly labeled as FM2. Taking these conclusions would be possible by inspecting solely the Confusion Matrices (Figure 5.13), but these are more easily inferred by the Balanced Accuracy scores.

5.3.3 Learning curves

The sufficiency of the amount of data in the databases was again evaluated using the two relevant scores: Accuracy and Balanced Accuracy. These results are presented in Figure 5.15.



(a) (b)

Figure 5.15: Learning Curves for the Failure Mode classifiers trained on *DATABASE2* based on (a) Accuracy and (b) Balanced Accuracy scores.

Regarding these, it is seen that the Balanced Accuracy score in general improves only slightly when increasing the number of data in training, whereas this increase is more noticeable for the Accuracy score. For both cases, it is evident that the GP classifier is the one that suffers more from scarcity of training data. In general, from the slope of the curves at the point with the most training data, it can be said that the amount of data used is sufficient. More data could improve the accuracy especially of the BM and the GP classifiers, though.

5.4 Curve Predictor

The models here trained will use the available features to predict polynomial approximations to the stress-strain curve (by predicting its polynomial coefficients) and the corresponding deformation at failure.

5.4.1 Models' tuning

1. Type of models trained: GP; NN; LGBM; BM.
2. Class of models: Regression, multi-output - Chain regression strategy.
3. Target: Polynomial coefficients a_0, a_1, a_2 and Failure Deformation (ε_{max}).
4. Outlier detection: MAD method, with threshold of 2.5. This MAD score was computed for all targets.
5. Feature Selection: RFECV, with a base RF regressor using inherent multi-output capabilities (and default hyperparameters). CV Score: Custom Score that computes the maximum stress for both the true curve and the predicted curve for all predictions and then a RMSE between these two lists of values.
6. Train-Test split: Stratified split: stratification for feature W/D with 3 bins. Test set containing 10% of the data.
7. Cross-Validation strategy: 5-Fold cross validation, with stratification for feature "W/D", with 3 bins.
8. Data normalization: Robust Scaler, for the GP; NN regressors. No normalization for the LGBM and BM regressors.
9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

From the four types of models that were chosen, only the LGBM implementation does not support inherently the prediction of multiple outputs, although in theory this should be possible for Gradient Boosted Trees. Furthermore, upon experimentation, due to the very different orders of magnitude of the four targets, the NN model required further processing of data, in this case the normalization of the output values, as the a_2 coefficient's contribution to the loss function was disproportionately large. The GP and BM models could be trained using this inherent approach, without further care, and good results were possible. However, the chain regressors approach yielded the best performance for the tested case seen in Appendix C.2,

and therefore it was decided to use it for all regressors. The order of the chain is: a_0; a_1; a_2; "Failure Deformation". Other orders were tested, as shown in Appendix C.3. Some difference in performance was noted, and this was the best performing order, from the ones tested.

The disadvantage of the decision to use the chain approach is the increment in complexity, as for each prediction there are effectively four regressors, and increased training time, as four regressors must be trained for each of the four model types, which does take some time, especially using the Exhaustive Grid Search and when many combinations of hyperparameters are possible.

It was decided to use the same MAD criterion with a threshold of 2.5 for outlier detection. In this case, the MAD score was computed for all four targets, and then data-points that surpassed the threshold for any of them were deemed outliers and removed. As seen in Figure 5.16, only the points between the red lines in all the four cases were kept (to the left of the red line in the a_2 case). This resulted in a reduction of the available number of points for the training and test sets from 1050 to 835.

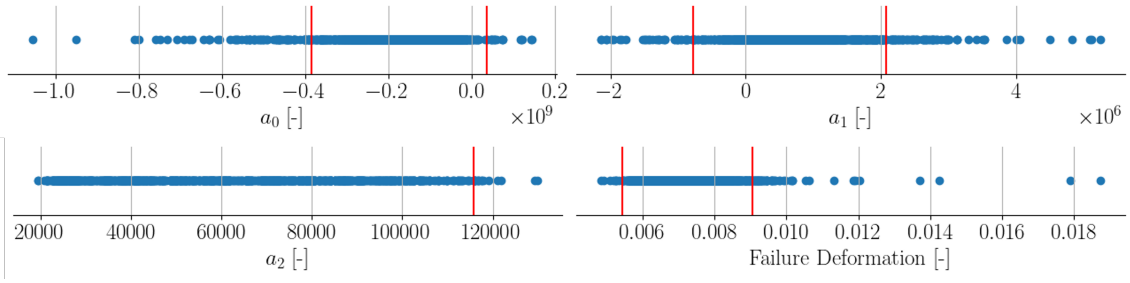
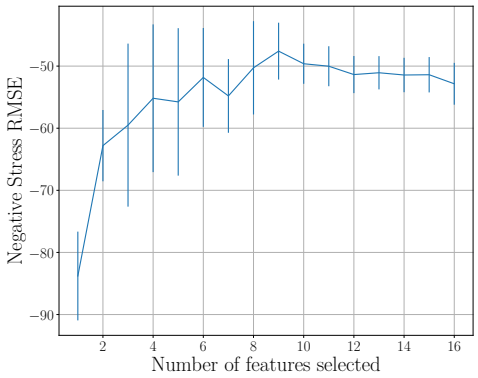


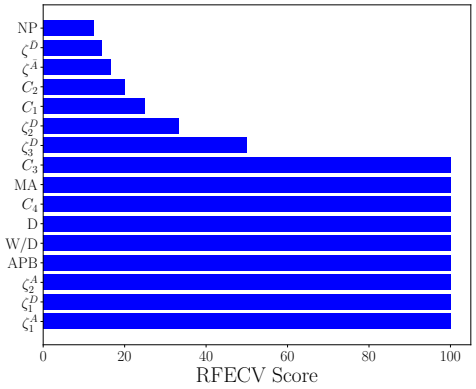
Figure 5.16: Outlier detection using MAD method, threshold = 2.5, on the Curve Predictors' target labels in *DATABASE2*.

For the FS process, it was decided once again to use a RF regressor, because it is different from the models that were trained in reality, because its training time is low and because it has inherent multioutput capabilities (meaning it is possible to use a single regressor to predict the four target values). There were some possible approaches for the scoring method: the average RMSE of the four predictions could be used, an option that was considered not good enough because the contribution of the larger a_0 values would bias the score; a metric related to RAE, as its mean or median value, an option that was not considered ideal as it would only use three of the four targets; or an error related to the indirectly predicted Ultimate Strength, extracted from the stress-strain curve, which was the chosen option, as it fairly encompasses the contribution of all targets. It was decided to compute an RMSE for these indirect stress predictions.

As seen in Figure 5.17a, the ideal number of features is 9, according to the chosen method. The negative indirect Stress RMSE decreases below this number of features. The scores of the points that use higher number of features do not have such a high dispersion as those seen for the classifier in Section 5.3 but not as low as the ones seen for the direct stress regressor, Section 5.2. The selected features include, as seen in Figure 5.17b, once again both geometric features, the in-plane laminate parameters and some of the ply order related features, as was to be expected.



(a) Variation of indirect Ultimate Strength related RMSE as less important features are eliminated.



(b) Best features chosen by using the RFECV method.

Figure 5.17: Recursive Feature Elimination with Cross-Validation for the stress-strain curve prediction problem in *DATABASE2*.

Regarding the hyperparameter selection process, it is worth highlighting that the grid search was done individually for every regressor. The chosen hyperparameters are presented in Table 5.11 (totalling 16 grid searches).

Table 5.11: Best hyperparameters found through Exhaustive Grid Search for the Curve Prediction regressors trained on *DATABASE2*.

Model	Hyperparameter	Best Choice a_0 regressor	Best Choice a_1 regressor	Best Choice a_2 regressor	Best Choice ε_{max} regressor
GP Regressor	kernel type	Mátern	Mátern	Mátern	Mátern
	kernel parameters	nu = 0.5	nu = 1.5	nu = 4.5	nu = 1.5
NN Regressor	alpha	0.1	0.1	0.1	1.0
	batch_size	25	25	25	25
	hidden_layers	(32,32,32)	(32,32)	(64,32,16)	(64,32,16)
	activation	'relu'	'relu'	'relu'	'tanh'
	learning_rate_init	0.01	-	-	-
	max_iter	1000	1000	1000	1000
	optimizer	'adam'	'lbfgs'	'lbfgs'	'lbfgs'
LGBM Regressor	colsample_bytree	1.0	1.0	1.0	1.0
	subsample	1.0	1.0	1.0	1.0
	min_child_samples	1	5	10	5
	learning_rate	0.01	0.1	0.1	0.1
	max_depth	-1	-1	-1	-1
	num_leaves	20	5	5	10
	n_estimators	600	800	800	800
BM Regressor	max_features	1.0	1.0	1.0	1.0
	max_samples	0.75	1.0	1.0	1.0
	n_estimators	400	600	400	800

5.4.2 Models' evaluation

The trained models will be evaluated on the test set. The errors of the prediction of each single target will be registered. Furthermore, the errors of polynomial prediction will be evaluated by using the RAE metric.

In Figure 5.18, the observed-predicted scatter plots are presented for all four targets, only for the best performing model, the GP model. The RMSE values for these four targets are presented in Table 5.12 for the four models.

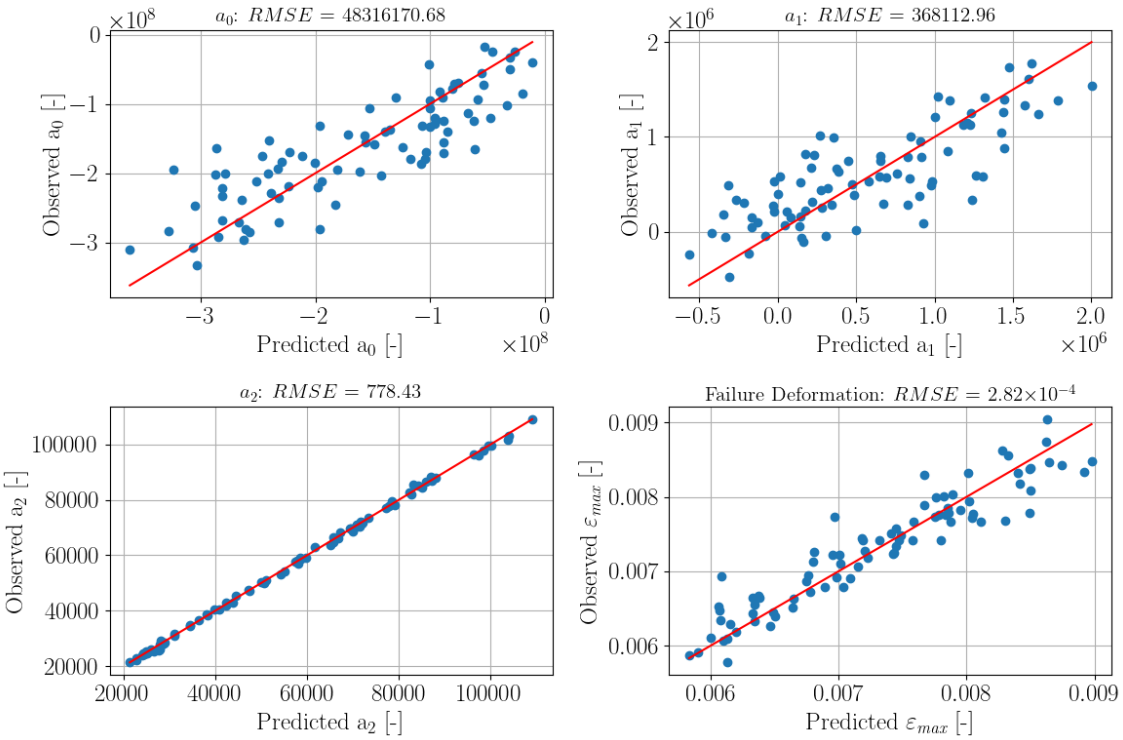


Figure 5.18: Errors in polynomial coefficient and failure deformation predictions for the Gaussian Processes (GP) based Regressors trained on *DATABASE2*.

Table 5.12: RMSE values for each of the four targets of the four stress-strain curve predictors trained on *DATABASE2*.

	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor
a_0 [-]	4.83E+07	6.16E+07	5.67E+07	5.64E+07
a_1 [-]	3.68E+05	4.85E+05	4.33E+05	4.61E+05
a_2 [-]	7.78E+02	8.99E+02	21.94E+02	28.94E+02
ε_{max} [-]	2.82E-04	3.85E-04	3.26E-04	3.63E-04

In Figure 5.19, the RAE is presented in the form of histograms, one for each of the four models. Furthermore, in Table 5.13, the mean and median RAE for each model is presented. This

RAE was computed, as explained in Section 4.5.3, between the predicted and true polynomial approximations up to the true failure deformation (meaning that this metric does not use the predicted Failure Deformation).

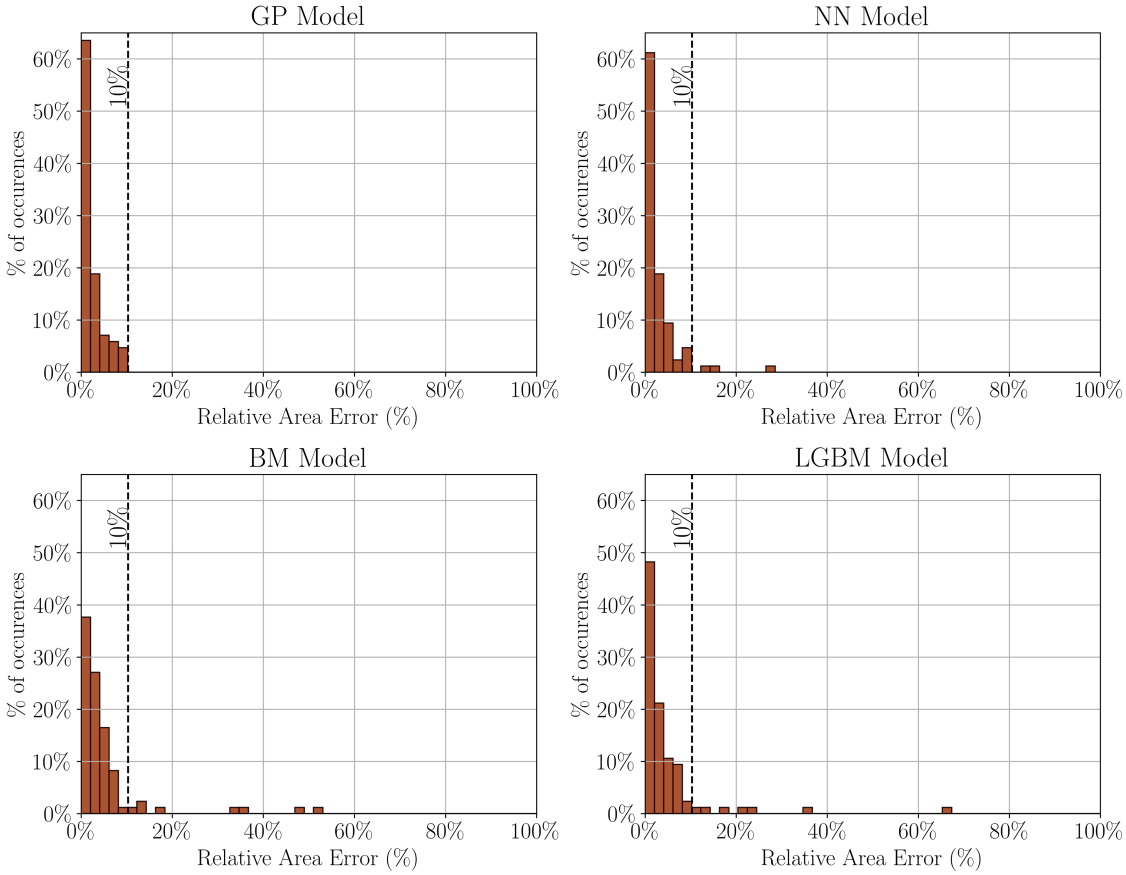
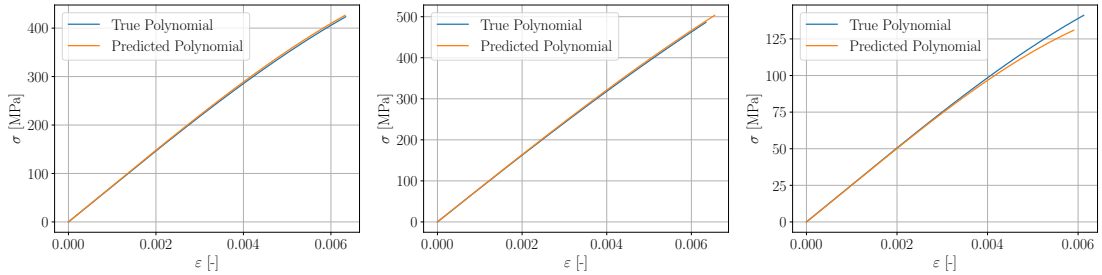


Figure 5.19: Relative area error histograms for the four Curve Predictor models trained on *DATABASE2*.

Table 5.13: Mean and Median RAE for the four Curve Predictor models trained on *DATABASE2*.

	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor
Mean RAE [%]	2.34	2.86	4.78	5.36
Median RAE [%]	1.46	1.68	2.17	2.75

By analysing Figure 5.18, some conclusions can be taken straight away. First of all, it is noticeable that the models can very accurately correlate the input features with the a_2 target. This coefficient corresponds to the linear part of the polynomial. This evidences the fact that the model can easily predict the slope of the stress-strain curves in the early loading stages. The dispersion of the a_0 and a_1 predictions is much greater. RMSEs are, of course, not comparable between targets, due to the disparity in their magnitude. These two coefficients correlate to the



(a) (b) (c)

Figure 5.20: Examples of predicted stress- strain curves from the stress-strain curve predictors trained on *DATABASE2*.

non-linear part of the curves. Regarding the Failure Deformation target, it can be said that the model is capable of performing reasonable predictions in this regard.

By analysing Table 5.12, it is easily concluded that the GP model is the best performing, in all four targets. The NN model seems to have more difficulty capturing the trends in data for targets a_0 and a_1. This is attributed to the magnitude of these targets, that can lead to numerical problems when minimizing the loss function of the networks. The tree based models present large errors when predicting the linear part of the curves, when compared to the other two models. Between the two, though, the Gradient Boosted method based model (LGBM) seems to perform better.

In Figure 5.19, it is possible to see that most of the models' polynomial curve predictions are close to the true polynomial approximations, below the acceptable threshold of 10%. Remarkably, for this test set, all predictions of the GP model have an RAE below 10%. The NN model only has some outlier errors, never surpassing 30% (note that RAE does not include the Failure Deformation target), whereas the LGBM and BM models make curve predictions that reach RAEs of over 50%. With these conclusions, together with an analysis of Table 5.13, it is easy to conclude, as would be to predict from the analysis of the individual targets, that the GP model is the best performing, followed by the NN model and then the DT based models when predicting the polynomial approximations of the curves.

In Figure 5.20, three examples of curves predicted by the ML models were shown. In Figure 5.20a, a very good prediction is seen; in Figure 5.20b a curve where the polynomial coefficients were accurately predicted but the failure deformation has some error is presented; and in Figure 5.20c, on the other hand, a curve for which the failure deformation was reasonably well predicted, but there is some greater error on the polynomial coefficients is seen.

5.4.3 Learning curves

To evaluate if the amount of data used is sufficient, learning curves were plotted for each of the four targets, using their individual RMSE score.

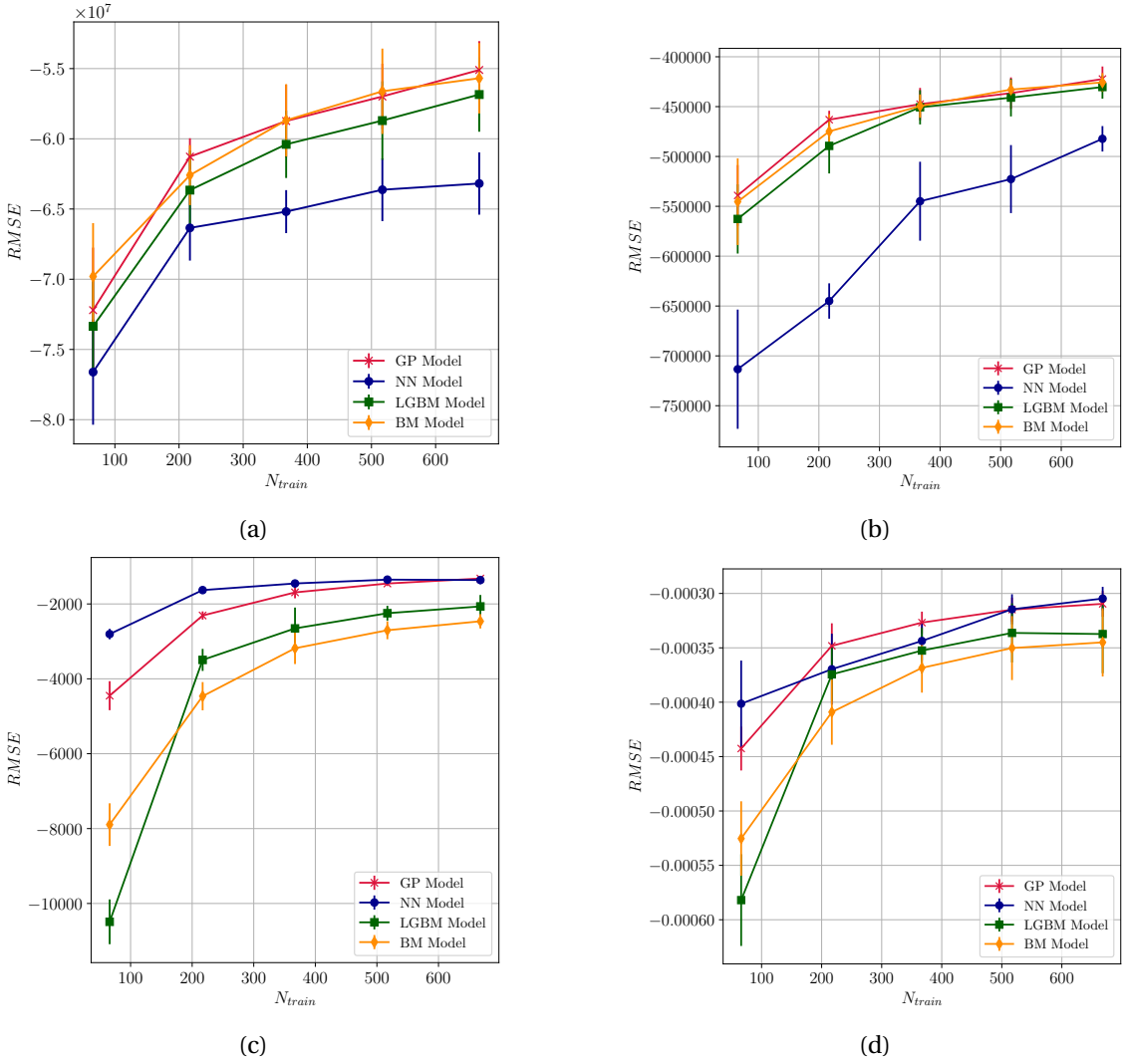


Figure 5.21: Learning curves for stress-strain Curve Predictors trained on *DATABASE2*, using RMSE score for each of the four targets: (a) a_0; (b) a_1; (c) a_2; (d) ε_{max}.

From Figure 5.21 it can be concluded that, in general, data is sufficient. It can be noted that the NN regressor has even more difficulty when predicting a_0 and a_1 with small values of N_{train}.

5.4.4 Strength Predictions

All four targets of the models can, combined, yield the polynomial curve prediction truncated at the predicted Failure Deformation, as seen. From this stress-strain curve, the maximum stress can be predicted. This is an indirect way of obtaining the Ultimate Strength.

From this metric, all the analysis made in Section 5.2 can be reproduced. This is shown in Appendix D.1. From this analysis, it can be concluded that the indirect Ultimate Strength regressors exhibit, curiously, good performance. Their predictions are, in general, worse than those of the direct Ultimate Strength regressors. The indirect models seem to have a tendency to under-estimate the strength values, which is related to the chosen order of the polynomials.

5.5 Concluding remarks

It was shown in this Chapter that it is possible to build and train data-driven surrogate models capable of predicting Ultimate Strength; Failure Mode and even the general shape of whole stress-strain curves, for a given material, in this case the IM7/8552 CFRP. Results were very satisfactory in this regard.

Overall, for all three different objectives, the best performing model was the one based on Gaussian Processes. This may be explained by the fact that that these types of models are very simple, without too many hyperparameters to optimize, thus facilitating the model building and hyperparameter grid choice phases. It can be argued that more complex Neural Networks (for example with more layers and nodes or with regularization layers) or Decision Tree ensembles than the ones here built could match the results of the Gaussian Processes based model. Although this is recognized, the problem at hand seems to have simple enough tendencies (as shown by the ability of the GP models to easily capture them), and therefore the addition of much more model complexity was deemed unnecessary. It was shown that computationally cheap (predictions made in fractions of a second) and very simple ML models can yield excellent results.

In the following Chapter, this analysis will be expanded. Variability in material properties will be included, resulting in a significant increase in the feature space's dimensionality, and in the difficulty of the tasks. Extraction of Design Allowables based on the trained ML models will be performed.

Chapter 6

Surrogate models for an open-hole coupon including material variability

In this Chapter, a database with variable material properties, geometry and stacking sequences will be generated and used to train ML models to predict ultimate strength, failure mode and stress-strain curves for specimens under OHT. The models are expected to be accurate for a wide range of materials, geometries and stacking sequences. As these models take into account all these parameters, which are sources of variability in the test results, they may be used to generate virtual Design Allowables.

6.1 Dataset generation

6.1.1 Material property assumptions

With the objective of reducing the dimensionality of the sampling space, some assumptions and simplifications regarding material properties will be taken. For most CFRP materials, some properties can be estimated based on others, by making some simplifying assumptions. Other properties variation does not influence the OHT results much (Sasikumar et al. (2023)), and therefore it is reasonable to keep them to a common value when their value is not known for the material at hand. All the assumptions made in this work to simplify the input space required for numerical simulations will now be detailed.

The assumptions made regarding the elastic properties, presented in Section 2.4.1, are the following:

- E_{1c} assumed to be 80% of E_1 (Furtado et al. (2019)).
- E_2 estimated based on master ply and Tsai's modulus (Arteiro et al. (2020)).
- ν_{12} estimated based on master ply and Tsai's modulus (Arteiro et al. (2020)).

This means that, with the mentioned assumptions, only the E_1 and G_{12} elastic properties must be specified (and given as input for numerical simulations).

Regarding the ply strength properties, detailed in Section 2.4.2, the values that will be assumed for the properties assumed constant are the average property values of several different carbon fiber reinforced epoxy matrix composite materials. The assumptions are the following:

- f_{XT}: Assumed as constant and equal to 0.27.
- f_{XC}: Assumed as constant and equal to 0.4125.
- Y_T: Assumed as constant and equal to 56.7MPa.
- Y_C: Assumed as constant and equal to 222.74MPa.
- S_{LP}: Best fitting from the in-plane shear stress-strain curve of the ply. Calculated based on the other shear properties and simplifications on the shape of the curve (Furtado et al. (2019)).
- K_P: Best fitting from the in-plane shear stress-strain curve of the ply. Calculated based on the other shear properties and simplifications on the shape of the curve (Furtado et al. (2019)).

This means that, with the mentioned assumptions, only the X_T, X_C and S_L strength properties must be specified as inputs.

The assumptions made regarding the toughness properties, presented in Section 2.4.3, are the following (once again the constant values are averages from different materials):

- f_{GT}: Assumed as constant and equal to 0.444.
- G_{IC}: Assumed as constant and equal to 0.3132 kJ/m^2.

This means that, with the mentioned assumptions, only the G_{XT}, G_{XC} and G_{IIC} toughness properties must be specified as inputs.

In summary, the relevant material properties inputs for the numerical simulations that will be performed, having in consideration the presented assumptions, are a total of eight properties: E_1, G_{12}, X_T, X_C, S_L, G_{XT}, G_{XC} and G_{IIC}. Together with these, the ply thickness, t, will also be variable. Note that the ply thickness affects the Y_T, Y_C and S_L properties, due to the in-situ effect.

6.1.2 Design of Experiments

The generation of this database followed the procedure outlined in Section 4.1.1.

The geometric parameters were sampled from the ranges shown in Table 6.1. Material properties were sampled in the same way, from ranges shown in Table 6.2, and the remaining properties were determined as explained in Section 6.1.1. The laminate parameters were sampled as explained in Section 4.1: $\zeta^A_{1,2}$ in range [-1,1] but forced to the admissible laminate space; $\zeta^D_{1,2,3}$ resulting from the shuffling of the selected layups to result in different stacking sequences, with a limiting *MPB* of 2, for more realistic laminates. This implies that all laminates will have $MPB = 2$, and therefore *MPB* will not be a useful feature (it will have a variance of 0).

The Design of Experiments resulted in a total of $N_{points} = 2648$. The database here generated will be referred to as *DATABASE3*.

Table 6.1: Range of sampling of the geometric features in *DATABASE3*.

	D [mm]	W/D [-]
Min. Value	2	3
Max. Value	10	8

Table 6.2: Range of sampling of the material properties in *DATABASE3*.

	E_1 [MPa]	G_{12} [MPa]	X_T [MPa]	X_C [MPa]	S_L [MPa]	G_{XT} [kJ/m^2]	G_{XC} [kJ/m^2]	G_{IIC} [kJ/m^2]	t [mm]
Min. Value	140000	1800	2000	900	60	50	40	0.5	0.125
Max. Value	200000	6000	3500	1800	100	400	150	4.0	0.300

6.1.3 FEM Simulations

From the 2648 input files generated in the DoE phase, 2382 simulations obtained non-null results for the coupon strength. Furthermore, 7 of these points obtained ultimate strengths below 30MPa and were removed from the database, having been considered not valid. These problems are attributed to numerical errors in the simulations. This resulted in a total of 2375 points that entered the data pipeline.

Computational costs for the generation of this database are similar to those observed in *DATABASE2*, as presented in Table 5.2 (the model is the same, only the inputs and number of points to be generated vary). This resulted in (estimated) 26 days of running and post-processing time, with the available computational resources[7].

Results were post-processed as presented in Section 4.2. All features were derived as shown in Section 4.3, and the post processed results were organized in a processed *DataFrame*, as presented in Section 4.4.1.

Summary of these outputs may be seen in Tables 6.3 to 6.5. It is possible to notice in Table 6.4 that the dataset is very unbalanced when it comes to failure mode, similarly to DATABASE2, with fiber dominated failure being much more frequent. In Figure 6.1 the data-point distribution in the $\zeta^A_{1,2}$ space is presented: it is evident that in this case, laminates may have matrix dominated failure modes in a wider part of the space, because FM also depends on material properties.

Table 6.3: Ultimate Strength results in *DATABASE3*.

Minimum Ultimate Strength [MPa]	Maximum Ultimate Strength [MPa]	Average Ultimate Strength [MPa]	Standard Deviation of Ultimate Strength [MPa]
104.13	1400.39	472.03	212.90

[7] 24 CPUs were used for running the simulations, and 1 CPU was used for post-processing.

Table 6.4: Failure Mode results in *DATABASE3*.

	Fiber dominated failure (FM1)	Matrix dominated failure (FM2)
Number of points	2094	281
Percentage	88.16%	11.83%

Table 6.5: Failure Deformation results in *DATABASE3*.

Minimum Failure Deformation [-]	Maximum Failure Deformation [-]	Average Failure Deformation [-]	Standard Deviation of Failure Deformation [-]
4.30E-03	28.65E-03	9.22E-03	2.42E-03

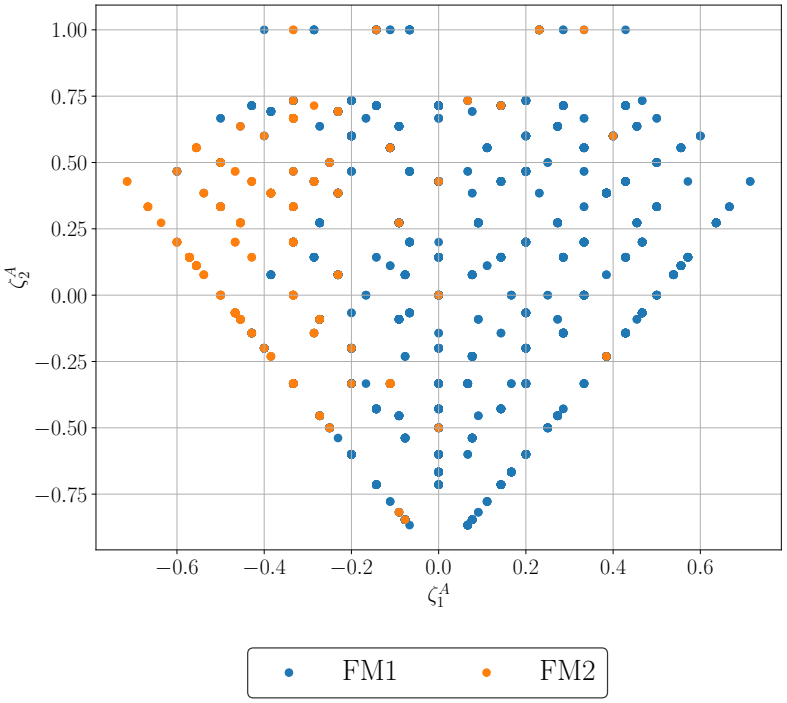


Figure 6.1: Failure mode of points presented in the $\zeta_{1,2}^A$ space for *DATABASE3*.

In each of the following Sections, a class of model will be specified, built, trained and evaluated: Ultimate Strength regressors, Failure Mode classifiers and stress-strain Curve Predictors. After this, the best performing Ultimate Strength regressor will be used to generate Design Allowables.

6.2 Strength Regressor

Four models, each corresponding to a different type of ML algorithm, will be trained to use the available features as inputs, and predict a single output: the Ultimate Strength, σ_u, of the laminate under OHT.

6.2.1 Models' tuning

Model preparation follows the outline presented in Section 4.4. In particular, we have used the following details:

1. Type of models trained: NN; LGBM; GP; BM.
2. Class of models: Regression, single output.
3. Target: Ultimate Strength, σ_u.
4. Outlier detection: MAD metric approach, with threshold of 2.5.
5. Feature Selection: RFECV, with a base RF regressor (default hyperparameters). CV Score: RMSE.
6. Train-Test split: Stratified split. Stratification for feature "W/D" with 3 bins. Test set containing 10% of the data.
7. Cross-Validation strategy: 5-Fold cross validation, with stratification for feature "W/D", with 3 bins.
8. Data normalization: Robust Scaler, for the GP and NN regressors. No normalization for the LGBM and BM regressors.
9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

The outlier detection method that was used identified 133 outliers, all with a MAD score above 2.5. No data-point had a MAD score below 2.5 (note that the 7 data-points that were preemptively removed after data generation would have been detected in this stage). This decreases our total number of data-points fit for training and evaluation from 2375 to 2242. This is presented in Figure 6.2, where the points to be used are to the left hand side of the red line.

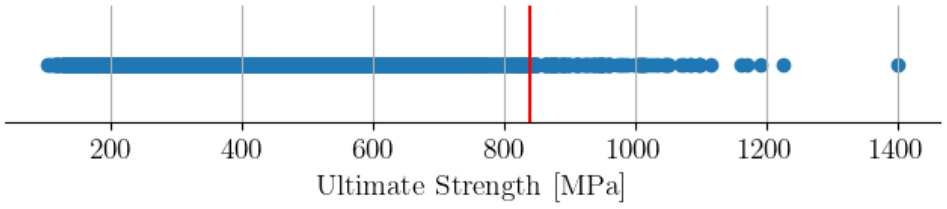


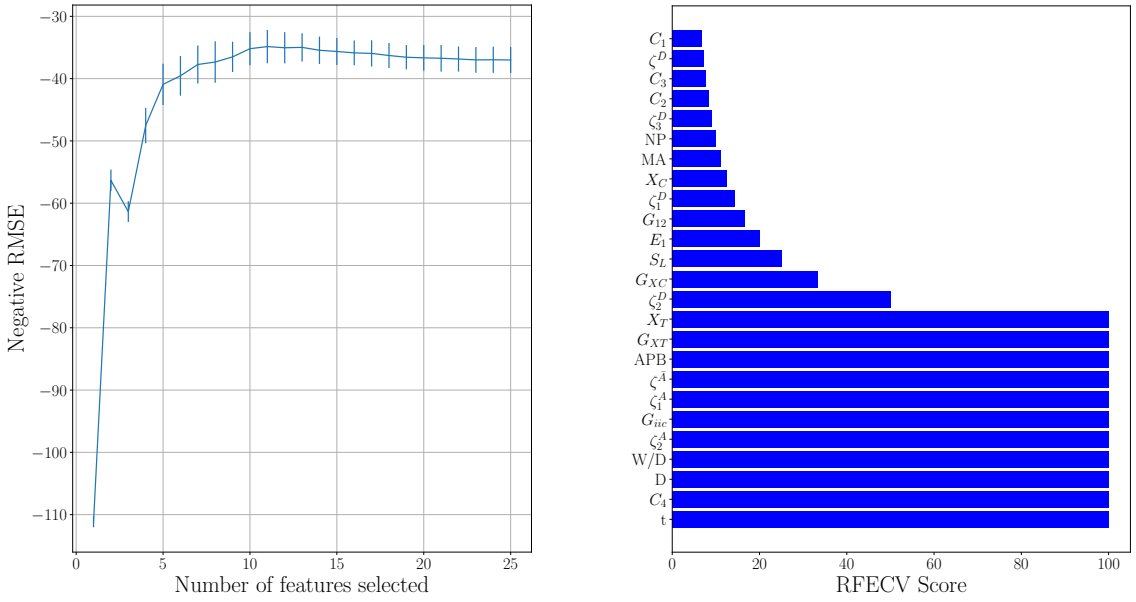
Figure 6.2: Outlier detection using MAD method, threshold = 2.5, on the Ultimate Strength feature in *DATABASE3*.

Despite this slight decrease in available data, model performance should be enhanced because of this pre-processing phase: there are not enough data-points in the areas from which the outliers were removed to provide good enough training. As a side-note, had we used a SD outlier detection method, with a threshold of 3.0, only 84 outliers would have been identified, which would have led to a decrease in available data-points from 2375 to 2291.

For the Feature Selection method by RFECV, an RF regressor was used. This choice was once again based on its simplicity, and associated low costs when fitting, and also so that there wouldn't be any bias towards one of the four models that are being trained.

Figure 6.3a shows clearly that performance (evaluated through the RMSE CV score), slightly increases when reducing the number of used features from 25 to 11, as redundant and irrelevant features are being eliminated. Below this number of features, performance is rapidly lost.

Figure 6.3b shows the 11 most important features, those with the highest RFECV Score. The model has identified the geometric features (D and W/D); the in-plane laminate parameters and derived in-plane parameter distance ($\zeta^A_{1,2}$, $\zeta^{\bar{A}}$); and some features related to ply order (APB; ζ^D_2, C_4). Regarding material properties, the RFECV method considered ply thickness; G_{IIC}; X_T and G_{XT} as relevant. As it is known that E_1 also has a significant impact on laminate strength (as will be showcased in Section 6.2.4), this propriety will also be considered as an input for the models, based on this physical argument. This results in a total of 12 used features, from the available 25 features. All other material properties will be disregarded.



(a) Variation of RMSE as less important features are eliminated.

(b) Best features chosen by using the RFECV method.

Figure 6.3: Recursive Feature Elimination with Cross-Validation for the Ultimate Strength regression problem in *DATABASE3*.

The train-test split results in 224 points in the test set and 2018 points in the training/validation set.

In Table 6.6, the hyperparameters chosen by the Exhaustive Grid Search methodology are

presented for each of the four trained models. All other hyperparameters of the models were left to their default values, as attributed in the `SKLearn` library.

Table 6.6: Best hyperparameters found through Exhaustive Grid Search for the Ultimate Strength regressors trained on *DATABASE3*.

Model	Hyperparameter	Best Choice
GP Regressor	kernel type	Mátern
	kernel parameters	nu = 1.5
NN Regressor	alpha	0.1
	batch_size	50
	hidden_layers	(32,32)
	activation	'relu'
	learning_rate_init	0.001
	max_iter	1000
	optimizer	'adam'
LGBM Regressor	colsample_bytree	1.0
	subsample	1.0
	min_child_samples	10
	learning_rate	0.1
	max_depth	-1
	num_leaves	5
	n_estimators	800
BM Regressor	max_features	1.0
	max_samples	0.75
	n_estimators	400

6.2.2 Models' evaluation

After hyperparameter selection and model training, the four models are evaluated on never before seen data, using the test set.

In Figure 6.4, the predicted and ground truth values of the ultimate strength for the laminates in the test set are plotted, and the RMSE for each of the four models is presented. Red lines represent what would be a perfect regressor.

In Figure 6.5, the relative errors are presented for the four models, in the shape of an histogram. These results are also reported in Table 6.7: in this table, the fraction of the test set with a registered error below the thresholds of 20%, 10%, 5% and 2% are presented for the four models and also for a *DUMMY* model, for comparison purposes.

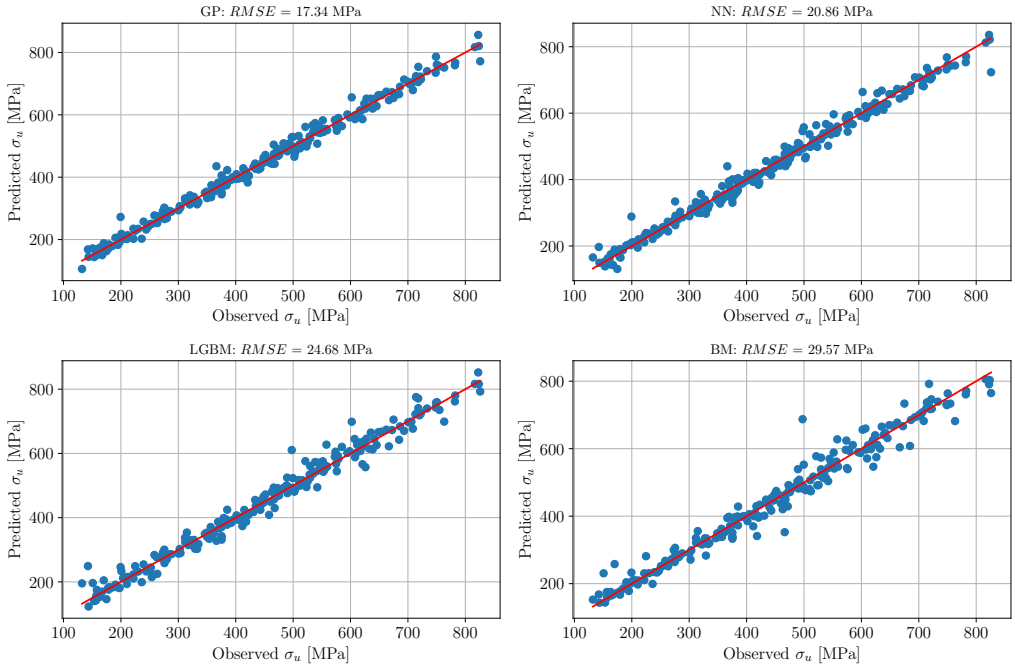


Figure 6.4: Observed (from numerical simulations) and predicted Ultimate Strength for the four ML Ultimate Strength regressors trained on *DATABASE3*.

Table 6.7: Relative Error (RE) fractions below thresholds of 2%, 5%, 10% and 20% for the four ML Ultimate Strength regressors trained on *DATABASE3*.

	Fraction of Data below the RE Limit for each Model				
RE Limit	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor	*DUMMY* Regressor
2%	0.458	0.427	0.409	0.369	0.031
5%	0.787	0.742	0.716	0.702	0.071
10%	0.964	0.938	0.898	0.880	0.169
20%	0.991	0.973	0.973	0.978	0.369

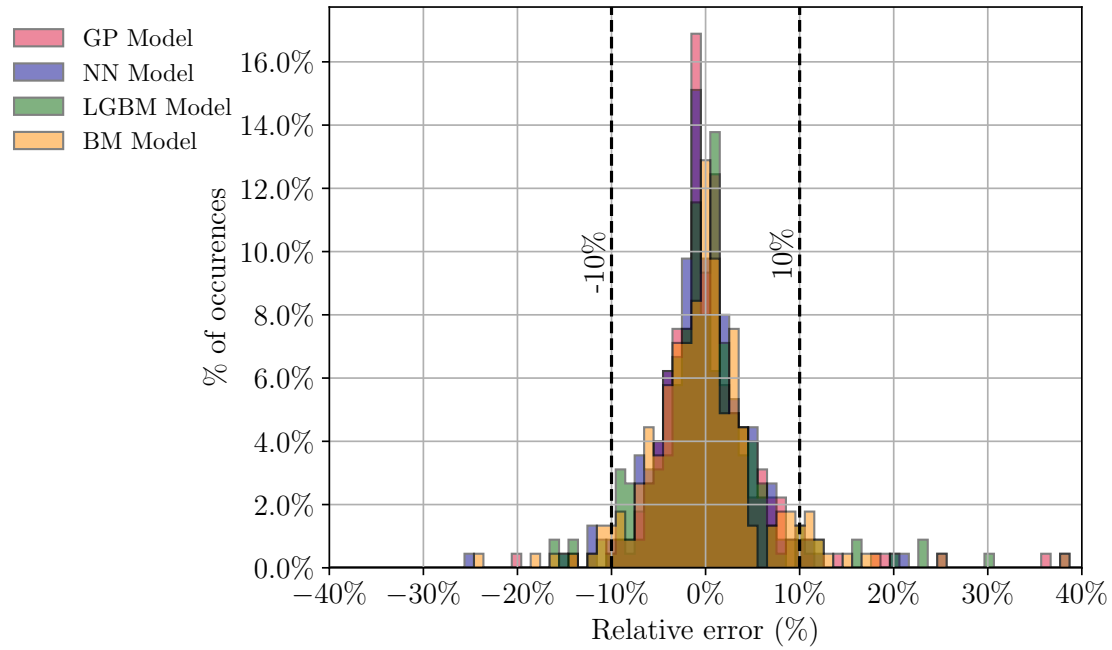


Figure 6.5: Histogram of Relative Errors for the four ML Ultimate Strength regressors trained on *DATABASE3*.

Here the four regressors were evaluated on never before seen data. The best performing model was once again the GP regressor, with an RMSE of 17.34MPa, followed by the NN regressor, the LGBM regressor, and finally, the worst performing, BM regressor, with an RMSE of 29.57MPa.

In terms of relative errors, the great performance capabilities of the GP regressor are once again highlighted, having predicted 96.4% of the test set data-points with a relative error lower than 10%. All others perform reasonably well, predicting a fraction of above 88% of the test set above this threshold.

These results are satisfactory, although, as expected, the models here trained are worse performing than those trained for the fixed material. This is evidently justified by the fact that the task at hand is now more difficult: the input dimension is higher; models are capable of making predictions for a wide range of materials.

6.2.3 Learning curves

As was detailed in Section 4.6, a way of assessing if the database has enough data-points is by means of plotting learning curves. Here this was done using RMSE as the scoring metric.

The initial dataset from which the N_{train} data-points are sampled had already been cleaned of outliers, using the same MAD method with a threshold of 2.5. For this analysis, all 25 possible features were used. The hyperparameters of the models were fixed, and are the ones that were presented in Table 6.6. For the CV splits, a StratifiedKFold approach was used, with 5 folds; the stratification feature being W/D, with 3 bins.

The learning curves are presented in Figure 6.6, for the four model types.

This analysis showed that the number of data-points in the database is sufficient. For all models, the slope of the learning curves is very small at the last point of the curves, with over 1700 training points.

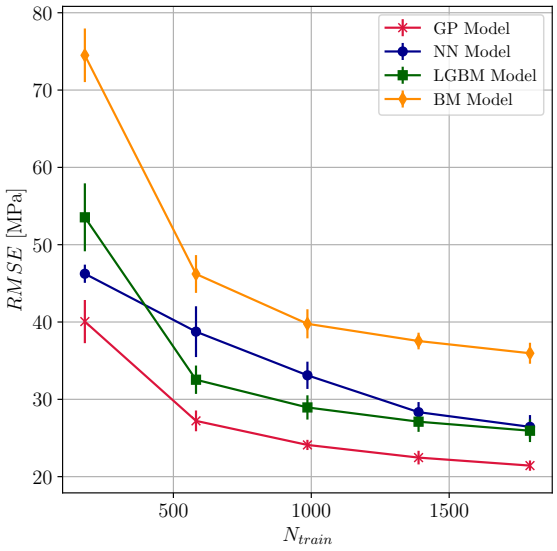


Figure 6.6: Learning curves for Ultimate Strength regressors trained on *DATABASE3*, using RMSE score.

6.2.4 Sensitivity analysis

To evaluate the ability of the models to capture the trends in the strength of the OHT specimens, a sensitivity analysis was performed.

Here the values of the features D, W/D, $\zeta^A_{1,2}$, $\zeta^D_{1,2,3}$, Number of Plies and of the nine used material properties, even the ones not selected by the FS method, were varied within their respective ranges, one at each time, while all the other parameters were fixed. When varying the ζ^A parameters, it was inevitable to also vary the ζ^D parameters and NP, but these variations were kept to the minimum within what is possible. The inputs for this sensitivity analysis may be seen in Appendix B. Results are presented in Figures 6.7 to 6.9.

Figure 6.7 shows that the models are capable of capturing the general trends when varying geometry and layup. The tree based models seem to have some difficulty when presented with very large or very low values of ζ^A_2.

Figure 6.8 shows, once again, that the models have difficulty in capturing trends when individually varying the ζ^D out-of-plane parameters and Number of Plies.

Figure 6.9 shows a very interesting analysis on material property variations. First, we note that the properties G_{XT} and t are the ones that influence Ultimate Strength the most: an increase in G_{XT} from 50kJ/m^2 to 140kJ/m^2 leads to an increase in σ_u of more than 150MPa, and an increase in ply thickness from 0.08mm to 0.27mm leads to a decrease in strength of about 100MPa. Features E_1, X_T and G_{IIC} also influence the strength of the specimens in more than 50MPa when varying them in the training ranges. The other four material properties don't seem to affect the Ultimate Strength of the specimens significantly, as seen by their almost flat nu-

merical results lines. The reason these properties were selected in the DoE phase was that with this decision, the same points selected for simulation in this work can be used for a OHC (Open Hole Compression) study (in that case these properties, G_{12}, X_C, S_L and G_{XC}, should be more important).

Regarding the ML models' predictions, it is observed that both the GP and NN models capture the trends in data very well. The tree based regressors' prediction lines are further away from the ground truth simulation results. The four properties that do not influence significantly the specimens' strength are the ones that were not selected during the FS phase, which explains the completely flat prediction lines in the figures corresponding to them. This testifies that FS was well performed: these features were correctly deemed not relevant and therefore were eliminated.

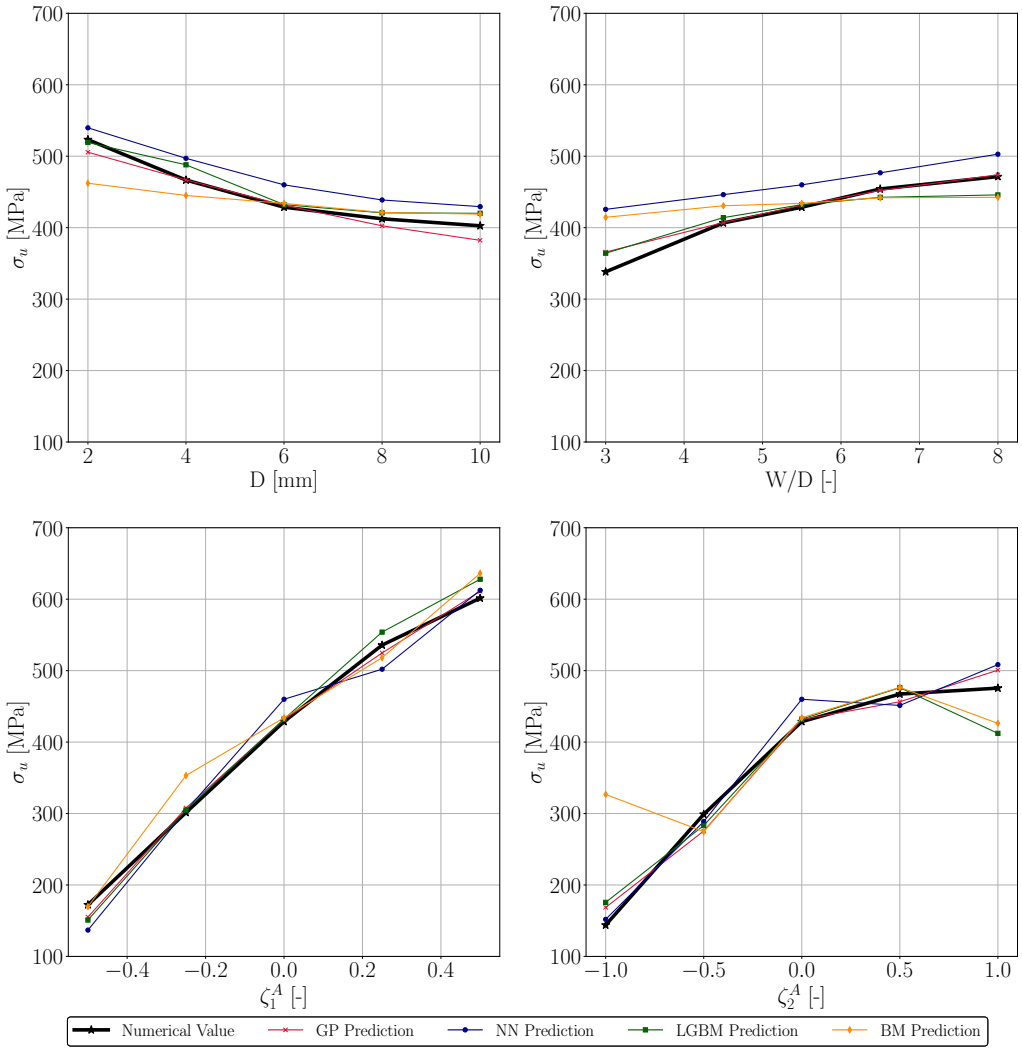


Figure 6.7: Results of the sensitivity analysis of Ultimate Strength regressors trained on *DATABASE3* for geometric features and in-plane laminate parameters.

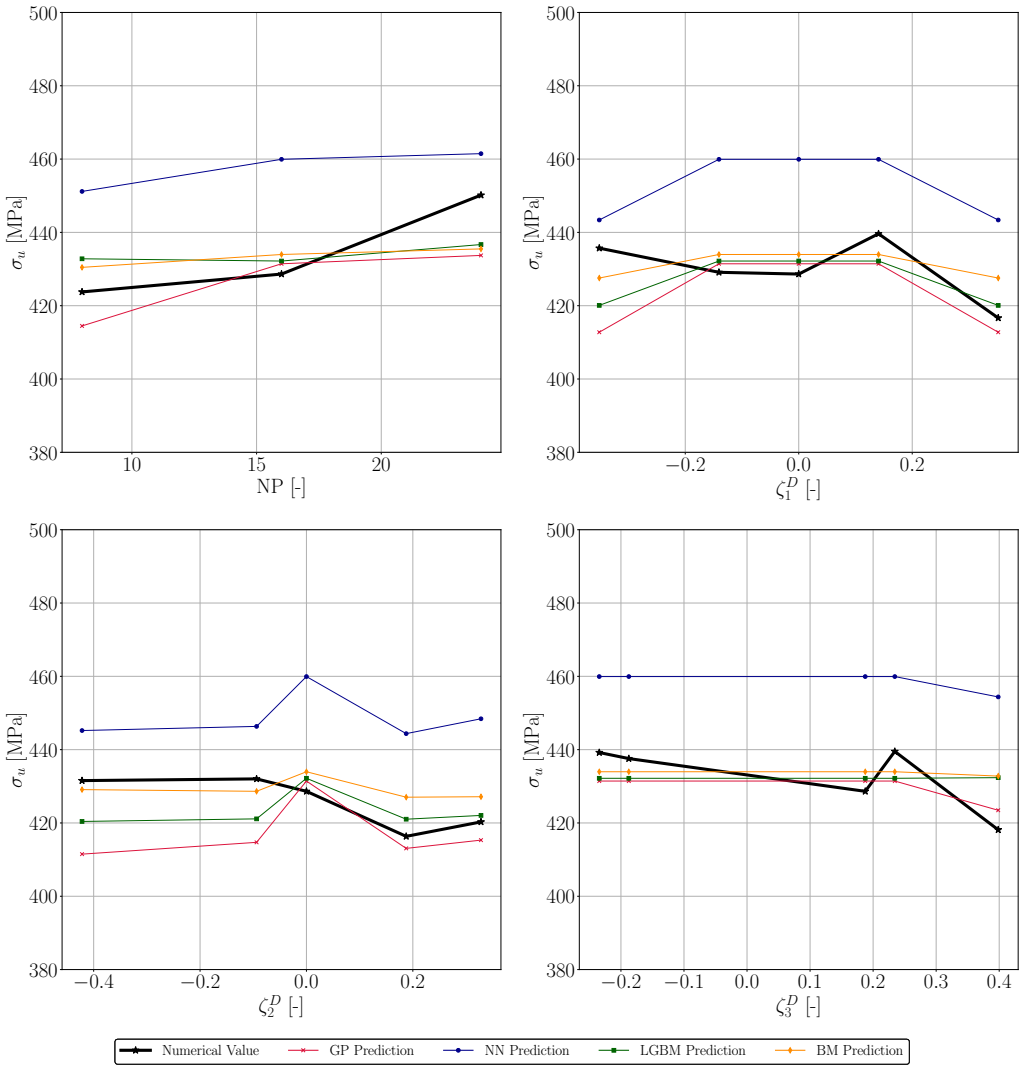


Figure 6.8: Results of the sensitivity analysis of Ultimate Strength regressors trained on *DATABASE3* for number of plies and out-of-plane laminate parameters.

6.2.5 Validation with experimental results

The models will be compared first to the same experimental results that were used and already presented in Section 5.2.5, for the IM7/8225 material. Then other experiments, with different materials, will be used for evaluation purposes.

The results of the experimental campaign of Camanho et al. (2007), numerical FEM simulations using the methodology of Furtado et al. (2019) for that data, and the predictions from the four ML models are presented in Figure 6.10.

Relative Errors (RE) of the predictions in relation to the numerical FEM simulations and experimental results are reported in Table 6.8.

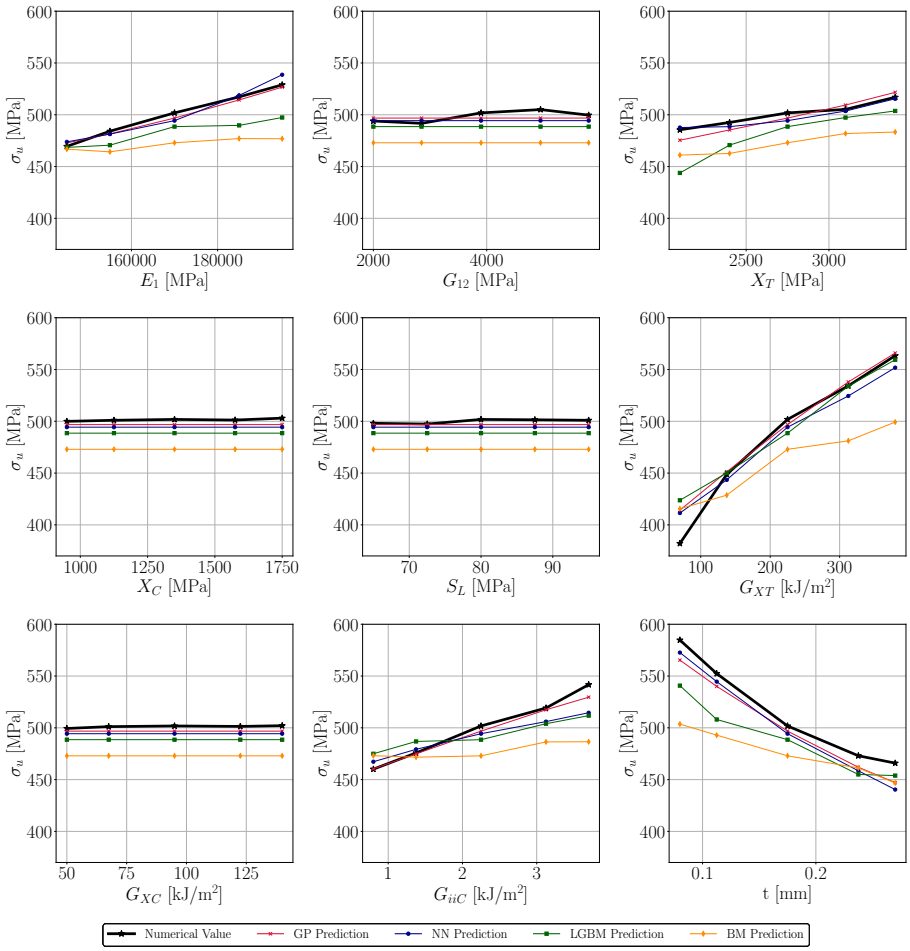


Figure 6.9: Results of the sensitivity analysis of Ultimate Strength regressors trained on *DATABASE3* for material properties.

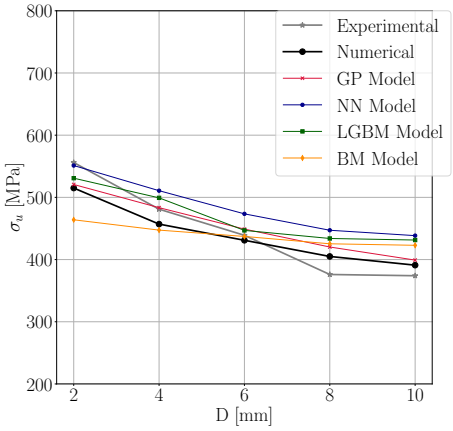


Figure 6.10: Comparison of the experimental results from Camanho et al. (2007) with numerical results and predictions from the four Strength regressors trained on *DATABASE3*.

Table 6.8: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE3)* relative to numerical and experimental (in parenthesis) results on the specimens from Camanho et al. (2007).

D [mm]	σ_u (Exp.) [MPa]	σ_u Num. [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
2	515	556	1.1 (-6.4)	7.0 (-0.9)	3.1 (-4.5)	-9.9 (-16.6)
4	457	481	5.8 (0.5)	11.8 (6.2)	9.2 (3.8)	-2.1 (-7.00)
6	431	439	4.2 (2.3)	9.9 (7.9)	3.7 (1.1)	1.4 (-0.4)
8	409	376	2.7 (11.8)	9.3 (18.9)	6.1 (15.4)	4.0 (13.1)
10	391	374	2.1 (6.7)	12.1 (17.2)	10.3 (15.4)	8.2 (13.1)

The results for all seven cases tested by Xu et al. (2016) are presented in Figure 6.11: the experimental results, the results from numerical simulations using the methodology from Furtado et al. (2019) and the predictions from the four ML models. The training range of feature D is marked in the Figure.

Relative Errors (RE) of the predictions in relation to the numerical FEM simulations and experimental results are reported in Table 6.9.

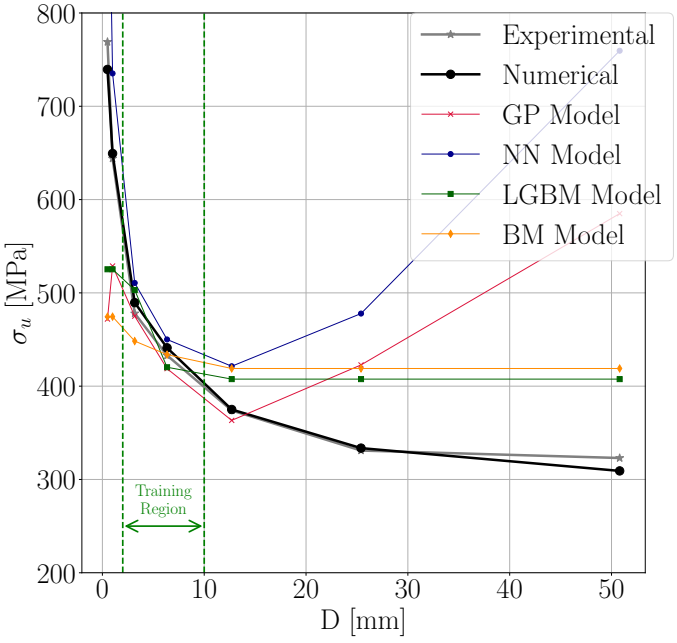
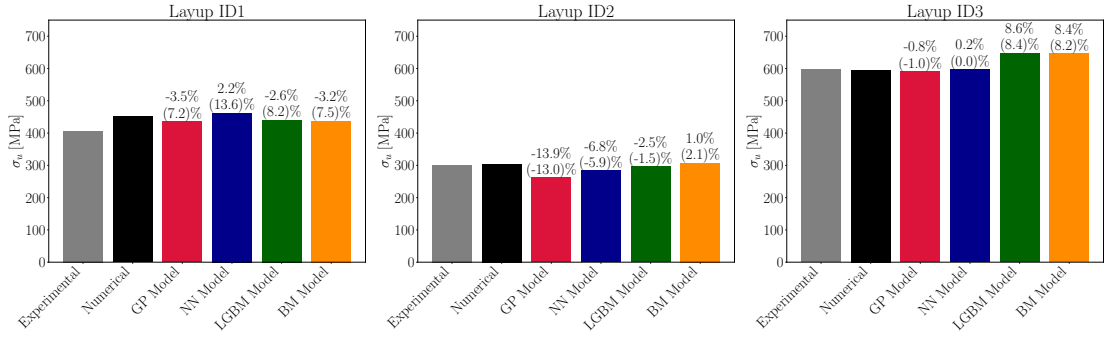


Figure 6.11: Comparison of the experimental results from Xu et al. (2016) with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE3*, as a function of hole diameter.

Table 6.9: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE3)* relative to numerical results on the specimens from Xu et al. (2016).

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
0.50*	739	769	-36.1 (-38.6)	43.0 (37.4)	-28.9 (-31.7)	-35.8 (-38.3)
1.00*	649	644	-18.5 (-17.9)	13.3 (14.2)	-19.1 (-18.4)	-26.9 (-26.3)
3.18	490	478	-3.1 (-0.6)	4.2 (6.8)	2.7 (5.3)	-8.5 (-6.2)
6.35	441	433	-5.0 (-3.3)	2.1 (3.9)	-4.7 (-2.9)	-1.6 (0.2)
12.70*	375	374	-3.1 (-2.9)	12.4 (12.7)	8.7 (9.00)	11.7 (12.0)
25.40*	334	331	26.6 (27.7)	43.1 (44.4)	22.0 (23.1)	25.4 (26.6)
50.80*	309	323	89.3 (81.1)	145.8 (135.2)	31.9 (26.2)	35.6 (29.7)

Results regarding the three different layups with fixed geometry from Clarkson (2012) are presented in Figure 6.12, where the numerical results were obtained using the methodology from Furtado et al. (2019). Above each model's bar, the RE in relation to the numerical and experimental results is reported.



(a) (b) (c)

Figure 6.12: Comparison of the experimental results from Clarkson (2012) that are near or within model training range with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE3*, for three different layups. Relative Error presented in relation to numerical and experimental (in parenthesis) results.

Erçin et al. (2013) presents an experimental study for a different material, the T800/M21. The material properties of this material are presented in Tables 6.10 to 6.12. The material properties sampled when generating *DATABASE3* are in bold in those Tables. Ply thickness used was $t = 0.125$ mm. The study uses two different layups:

- Laminate 1: $[90/45/0/\text{-}45]_{3S}$.
- Laminate 2: $[90_2/0_2/45_2/\text{-}45_2/90/0/45/\text{-}45]_S$

*Outside of training range. Training range is D∈[2, 10] mm.

For each of these laminates, three different geometries are tested: specimens with hole diameter of 3mm, 5mm and 7mm. W/D is kept constant and equal to 4.

Experimental and numerical results, as well as the ML model predictions are presented as a function of hole diameter in Figure 6.13, for both laminates. The relative errors for Laminate 1 and Laminate 2 are presented in Table 6.13 and Table 6.14, respectively. The numerical simulations used all of the presented material properties. The ML models, of course, only use the five chosen material features: E_1, X_T, G_{XT}, G_{IIC} and t.

Table 6.10: Elastic properties of the T800/M21 material.

$\boldsymbol{E_1}$ [**MPa**]	E_{1c} [MPa]	E_2 [MPa]	ν_{12} [-]	$\boldsymbol{G_{12}}$ [**MPa**]
172000	137600	8900	0.32	**5000**

Table 6.11: Strength properties of the T800/M21 material.

$\boldsymbol{X_T}$ [**MPa**]	f_{XT} [-]	$\boldsymbol{X_C}$ [**MPa**]	f_{XC} [-]	Y_T [MPa]	Y_C [MPa]	$\boldsymbol{S_L}$ [**MPa**]	S_{LP} [MPa]	K_P [-]
3039	0.4	**1051**	0.2	75	250	**95**	66.9	0.09

Table 6.12: Toughness properties of the T800/M21 material.

$\boldsymbol{G_{1+}}$ [**kJ/m^2**]	f_{GT} [-]	$\boldsymbol{G_{1-}}$ [**kJ/m^2**]	G_{2+} [kJ/m^2]	$\boldsymbol{G_6}$ [**kJ/m^2**]
340	0.52	**60**	0.228	**0.652**

The four models were tested on another different material, from another test campaign. In this case, geometric properties were kept constant: D = 6 mm and W/D = 5.04. Six different ply stacking sequences were used.

For confidentiality reasons, stacking sequences, material properties and absolute experimental/numerical results cannot be presented. Material properties are within the training ranges.

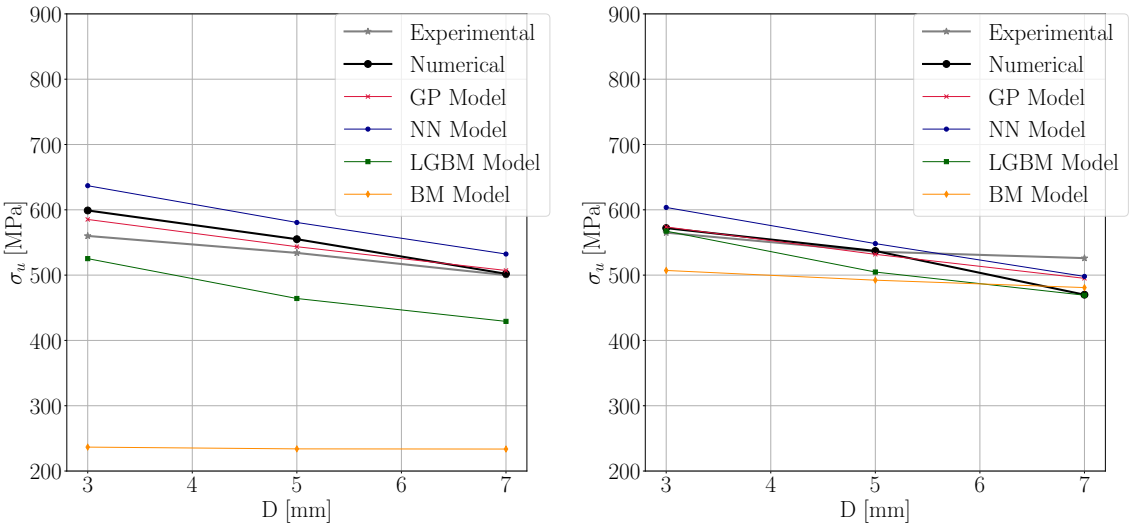
Results for the six layups can be seen in Figure 6.14, in relative terms.

Relative errors of the four ML models in relation to the experimental and numerical results are presented in Table 6.15. Numerical results were obtained with FEM simulations using the model proposed by Furtado et al. (2019).

In this Section, the models trained on *DATABASE3* were compared to results from the literature. The GP regressor was once again the best performing model overall.

When compared to numerical results obtained on data from Camanho et al. (2007), the GP regressor's predictions obtained errors lower than 6% for all points. The NN model overpredicted all strength values and the tree based models achieved reasonable results, with errors below 10% in almost all cases.

Regarding data from Xu et al. (2016), all models achieved good results inside the training range for feature D. Outside of this range, though, that is, when extrapolating, no model



(a) Laminate 1.

(b) Laminate 2.

Figure 6.13: Comparison of the experimental results from Erçin et al. (2013) with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE3*, as a function of hole diameter, for both Laminates.

Table 6.13: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE3)* relative to numerical results on the specimens from Erçin et al. (2013), for Laminate 1.

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
3	599	560	-2.3 (4.5)	6.3 (13.7)	-12.3 (-6.2)	-60.5 (-57.7)
5	555	534	-2.1 (1.8)	4.6 (8.7)	-16.4 (-13.1)	-57.9 (-56.2)
7	502	500	1.0 (1.4)	6.0 (6.5)	-14.5 (-14.2)	-53.5 (-53.3)

achieved good results. Even the NN model, that was the best extrapolate for the fixed material case, couldn't here capture any global trend. This is again attributed to the more difficult task of here including impact of material properties in predictions.

When compared to results from Clarkson (2012), all models were accurate on the three layups, with relative errors below 10%. The exception was the GP model's prediction for Layup ID2, where an underestimation of 13.9% was registered. A possible explanation for this is the fact that the presented numerical simulations were computed having as input all known features of the IM7/8552 material, and not with the simplifications presented in Section 6.1.1. Some of the properties that were assumed constant for not having a significant impact on specimen strength seem to have some larger influence on softer layups, as is the case of Layup ID2, explaining the larger errors.

When used on data for material T800/8552, from Erçin et al. (2013), the models behave very well, with relative errors below 6% being achieved once again for all predictions of the GP

Table 6.14: Relative error of the four Ultimate Strength regressors' predictions (trained on *DATABASE3)* relative to numerical results on the specimens from Erçin et al. (2013), for Laminate 2.

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
3	572	565	0.3 (1.6)	5.5 (6.8)	-0.8 (0.4)	-11.3 (-10.2)
5	537	536	-1.0 (-0.8)	2.1 (2.3)	-6.0 (-5.8)	-8.3 (-8.2)
7	470	526	5.3 (-5.9)	6.0 (-5.3)	-0.2 (-10.8)	-2.3 (-8.6)

Table 6.15: Relative error of ML predictions for results with a different material relative to numerical results and experimental test results (in parenthesis) for the four Ultimate Strength regressors trained on *DATABASE3*.

Layup ID	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
1	77.9 (44.9)	52.0 (23.8)	62.4 (32.3)	172.7 (122.1)
2	-14.7 (-21.5)	-18.0 (-24.6)	-3.1 (-10.9)	-6.7 (-14.2)
3	-3.8 (-6.7)	1.3 (-1.7)	-7.8 (-10.5)	-10.0 (-12.6)
4	-2.8 (-15.5)	-4.5 (-17.1)	-4.6 (-17.1)	-8.2 (-20.3
5	-4.6 (-21.9)	-6.1 (-23.1)	-11.9 (-27.9)	-17.1 (-32.2)
6	-14.9 (-17.7)	-13.9 (-16.6)	-19.3 (-21.9)	-17.4 (-20.0)

model. It must be noted that the BM model grossly under estimates results for Laminate 1, as seen in Figure 6.13. The LGBM also underestimates these values. The GP and NN models are able to capture the not steep decrease in strength with increasing diameter. The GP model achieves errors below 3% in all points except for the point with diameter 7mm of Laminate 2, where an error of 5.34% was registered.

When analysing Table 6.15, it is evident that larger errors were registered when comparing the ML predictions with numerical results for softer Layups, as Layup ID1, Layup ID2 and Layup ID6. These poor results are attributed to the same reason as previously mentioned for Clarkson (2012) soft layup, the material assumptions made. For the other three Layups, ID3, ID4 and ID5, errors below 7% were registered for the predictions of the NN and GP models, whereas the LGBM and BM registered larger errors generally. Figure 6.14 seems to suggest, that in general the models do seem to capture the trends in varying layup composition.

In all cases, when evaluated directly against the experimental results, the models evidently behave better on the points for which the numerical result was more accurate. This error can only be sustainably reduced by having more accurate numerical simulations, from which more accurate surrogate models could be generated.

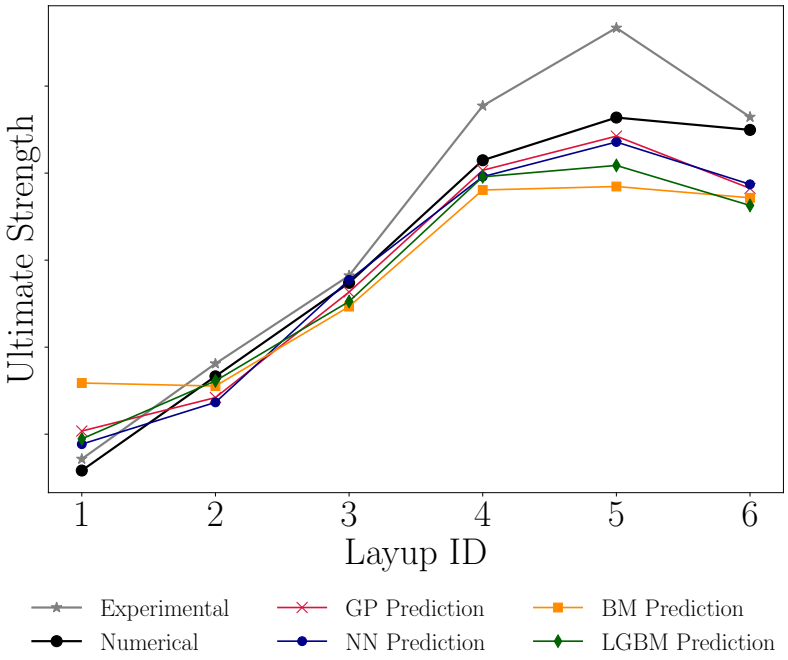


Figure 6.14: Comparison of the experimental results with a different material with numerical results and predictions from the four Ultimate Strength regressors trained on *DATABASE3*, for six different layups.

6.3 Failure Classifier

Four ML models will now be trained on *DATABASE3* and used in the classification problem: predicting the failure mode of the specimens.

6.3.1 Models' tuning

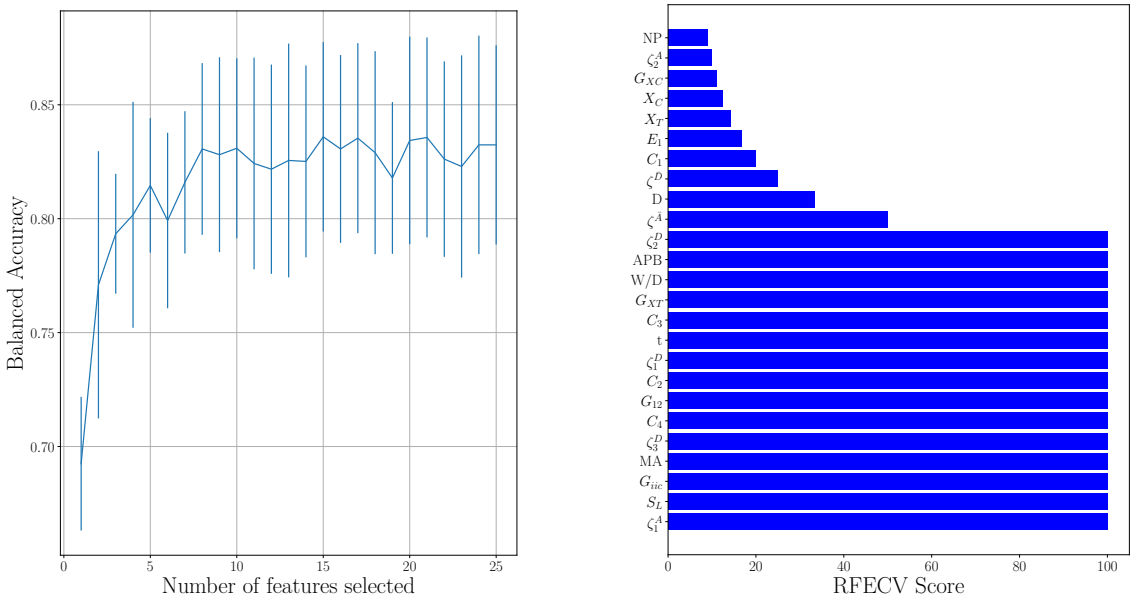
Model preparation will follow the outline presented in Section 4.4.2. In particular:

1. Type of models trained: GP; NN; LGBM; BM.
2. Class of models: Binary Classification, single output.
3. Target: Failure Mode.
4. No outlier detection was used.
5. Feature Selection: RFECV, with a base RF regressor (default hyperparameters). CV Score: Balanced Accuracy.
6. Train-Test split: Stratified split: stratification for the target Failure Mode. Test set containing 20% of the data.
7. Cross-Validation strategy: 5-Fold cross validation, with stratification for the target Failure Mode.

8. Data normalization: Robust Scaler, for the GP; NN classifiers. No normalization for the LGBM and BM classifiers.

9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

According to the RFECV FS method, the ideal number of features to be used is 15, as seen in Figure 6.15a. These features include, as presented in Figure 6.15b, the ratio W/D; the in-plane parameter ζ_1^A; all out-of-plane parameters $\zeta_{1,2,3}^D$; the features related to stacking sequence order APB, MA, C_2, C_3, C_4; and the material properties G_{XT}, S_L, G_{12}, G_{IIC} and t. All other features were considered not relevant for the prediction of failure mode.



(a) Variation of Balanced Accuracy as less important features are eliminated.

(b) Best features chosen by using the RFECV method.

Figure 6.15: Recursive Feature Elimination with Cross-Validation for the Failure Mode classification problem in *DATABASE3*.

The best hyperparameters chosen by the Exhaustive Grid Search performed are presented in Table 6.16.

6.3.2 Models' evaluation

The four ML classifiers will now be evaluated on never before seen data, stored in the test set. As was mentioned, the classifiers predict if the laminates Failure Mode is fiber dominated (FM1) or if it is not (FM2).

In Figure 6.16, the confusion matrices are presented for all four models. These show the percentages of TP, TN, FP and FN for the predictions on the test set.

Table 6.16: Best hyperparameters found through Exhaustive Grid Search for the Failure Mode classifiers trained on *DATABASE3*.

Model	Hyperparameter	Best Choice
GP Classifier	kernel type	Mátern
	kernel parameters	nu = 2.5
NN Classifier	alpha	0.1
	batch_size	25
	hidden_layers	(32,32)
	activation	'relu'
	learning_rate_init	0.001
	max_iter	1000
	optimizer	'adam'
LGBM Classifier	colsample_bytree	0.5
	subsample	0.1
	min_child_samples	10
	learning_rate	0.1
	max_depth	10
	num_leaves	10
	n_estimators	100
BM Classifier	max_features	1.0
	max_samples	1.0
	n_estimators	600

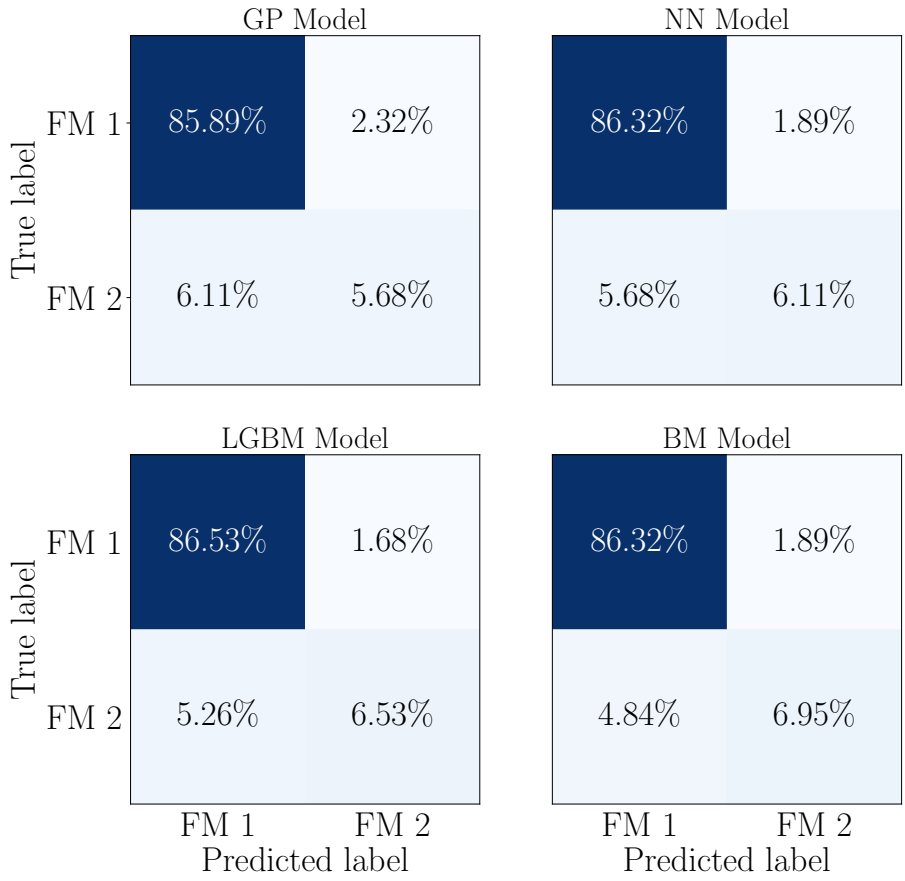
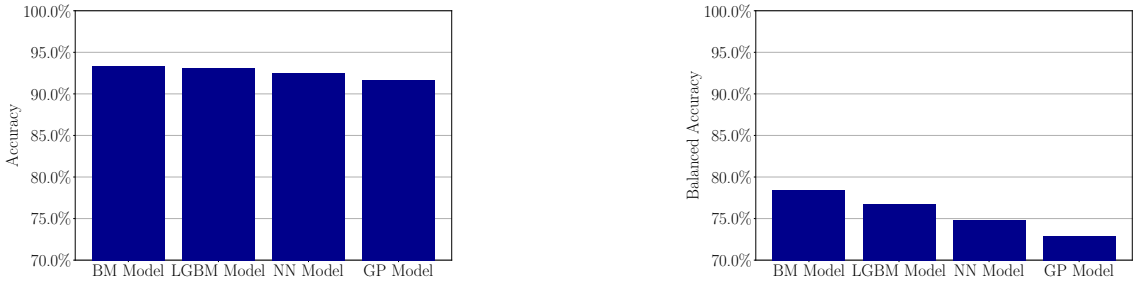


Figure 6.16: Confusion Matrices for the four models trained on *DATABASE3* for the Failure Mode classification problem.



(a) (b)

Figure 6.17: Accuracy (a) and Balanced Accuracy (b) scores for the Failure Mode classifiers trained on *DATABASE2*.

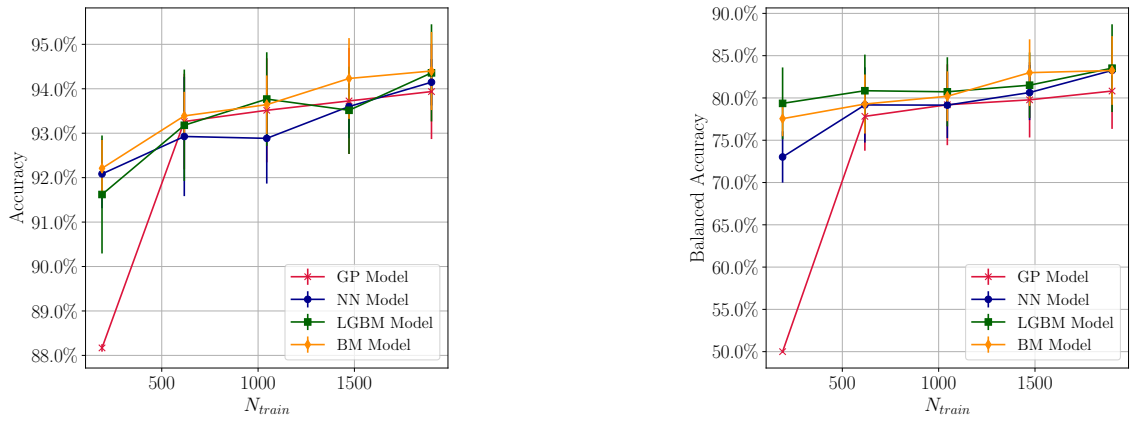
All of the models achieve accuracy scores above 91.5%, but it must be kept in mind that there is an unbalance of 88.16% in the population.

The tree based algorithms were here the ones with better performance. The BM classifier achieved an accuracy score of 93.6%. In terms of balanced accuracy, a score of 78.9% was achieved (a *DUMMY* classifier, always predicting FM1, would obtain a balanced accuracy of 50%).

Different approaches to this problem could be envisioned in the future, such as the development of a multi-output regressor that would try to directly predict the dissipated energies at failure, instead of classifying the Failure Mode.

6.3.3 Learning curves

The sufficiency of the amount of data in the databases was again evaluated using two relevant scores: accuracy and balanced accuracy. These results are presented in Figure 6.18.



(a) (b)

Figure 6.18: Learning Curves for the Failure Mode classifiers trained on *DATABASE3* based on (a) Accuracy and (b) Balanced Accuracy scores.

The addition of variation on material properties seems to bring a large increase in predic-

tion difficulty. In Section 6.3.3, some space for improvement with increasing the number of data-points can be foreseen.

6.4 Curve Predictor

The models that will be trained in this Section will use the available features to predict polynomial approximations to the stress-strain curve and the corresponding deformation at failure.

6.4.1 Models' tuning

1. Type of models trained: GP; NN; LGBM; BM.

2. Class of models: Regression, multi-output. GP and BM used their inherent multioutput capability. NN and LGBM used a chain regression strategy.

3. Target: Polynomial coefficients a_0, a_1, a_2 and Failure Deformation, ε_{max}.

4. Outlier detection: MAD method, with threshold of 2.5. This MAD score was computed for all targets.

5. Feature Selection: RFECV, with a base RF regressor using inherent multi-output capabilities (and default hyperparameters). CV Score: Custom Score that computes the maximum stress for both the true curve and the predicted curve for all predictions and then a RMSE between these two lists of values.

6. Train-Test split: Stratified split: stratification for feature W/D with 3 bins. Test set containing 10% of the data.

7. Cross-Validation strategy: 5-Fold cross validation, with stratification for feature "W/D", with 3 bins.

8. Data normalization: Robust Scaler, for the GP; NN regressors. No normalization for the LGBM and BM regressors.

9. Hyperparameter Selection: Exhaustive grid search. Grid shown in Table A.1.

As explained, the MAD metric was computed for all targets and only the points that met the criteria for all of them were kept. In Figure 6.19 this is shown: only points between the red lines were kept (to the left of the red line in the a_2 case).

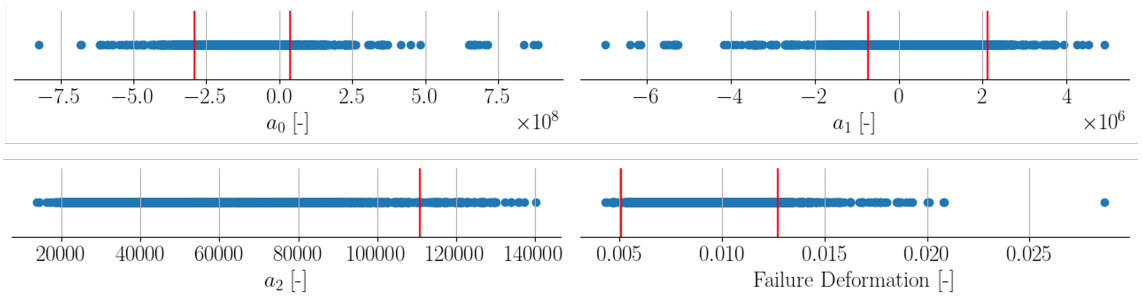
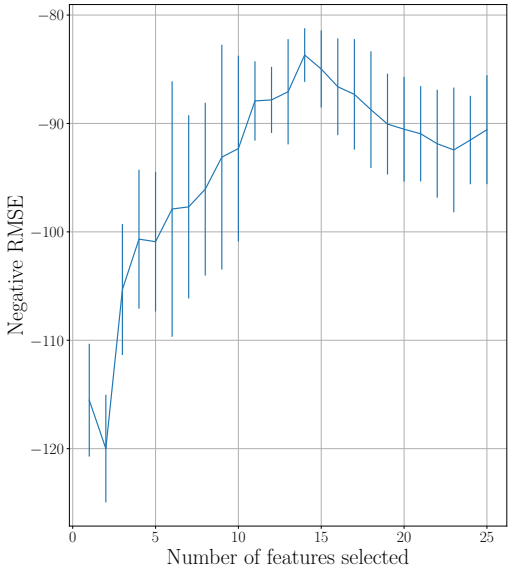
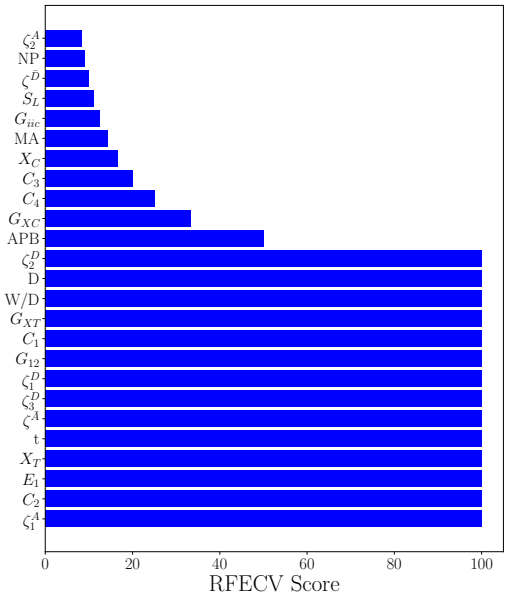


Figure 6.19: Outlier detection using MAD method, threshold = 2.5, on the Curve Predictors' target labels in *DATABASE3*.

The RFECV identified 14 essential properties, as seen in Figure 6.20a. These are: both geometric properties, D and W/D; the in-plane parameter ζ_1^A and the in-plane parameter norm, $\zeta^{\bar{A}}$; all out-of-plane parameters $\zeta_{1,2,3}^D$; C_1 and C_2, related to ply order; and the material properties G_{XT}, G_{12}, t, X_T and E_1, as seen in Figure 6.20b. Although not identified by the FS method, features ζ_2^A and G_{IIC} were forced into the group of used features, based on physical reasoning, resulting in a total of 16 used features.



(a) Variation of indirect Ultimate Strength related RMSE as less important features are eliminated.



(b) Best features chosen by using the RFECV method.

Figure 6.20: Recursive Feature Elimination with Cross-Validation for the stress-strain curve prediction problem in *DATABASE3*.

The hyperparameter selection via Exhaustive Grid Search resulted in the hyperparameters presented in Table 6.17 being used for each of the four chained regressors, in each of the four models.

Table 6.17: Best hyperparameters found through Exhaustive Grid Search for the Curve Prediction regressors trained on *DATABASE3*.

Model	Hyperparameter	Best Choice a_0 regressor	Best Choice a_1 regressor	Best Choice a_2 regressor	Best Choice ε_{max} regressor
GP Regressor	kernel type	Mátern	Mátern	Mátern	Mátern
	kernel parameters	nu = 0.5	nu = 1.5	nu = 4.5	nu = 1.5
NN Regressor	alpha	1.0	0.1	0.1	0.1
	batch_size	25	50	25	100
	hidden_layers	(64, 32, 16)	(64, 32, 16)	(32,32)	(32,32)
	activation	'relu'	'relu'	'relu'	'relu'
	learning_rate_init	0.01	0.001	0.001	0.01
	max_iter	1000	2000	2000	1000
	optimizer	'adam'	'adam'	'adam'	'adam'
LGBM Regressor	colsample_bytree	0.6	1.0	1.0	1.0
	subsample	1.0	1.0	1.0	0.6
	min_child_samples	10	10	10	5
	learning_rate	0.01	0.1	0.1	0.1
	max_depth	10	-1	-1	-1
	num_leaves	20	5	5	10
	n_estimators	600	800	800	800
BM Regressor	max_features	1.0	1.0	1.0	0.75
	max_samples	0.75	1.0	1.0	1.0
	n_estimators	600	1000	400	800

6.4.2 Models' evaluation

The trained models can now be evaluated on the test set. In Figure 6.21, the predictions for the four targets against ground truth values are presented, solely for the best performing model, the GP regressors. The RMSE values for this and the remaining models are presented in Table 6.18.

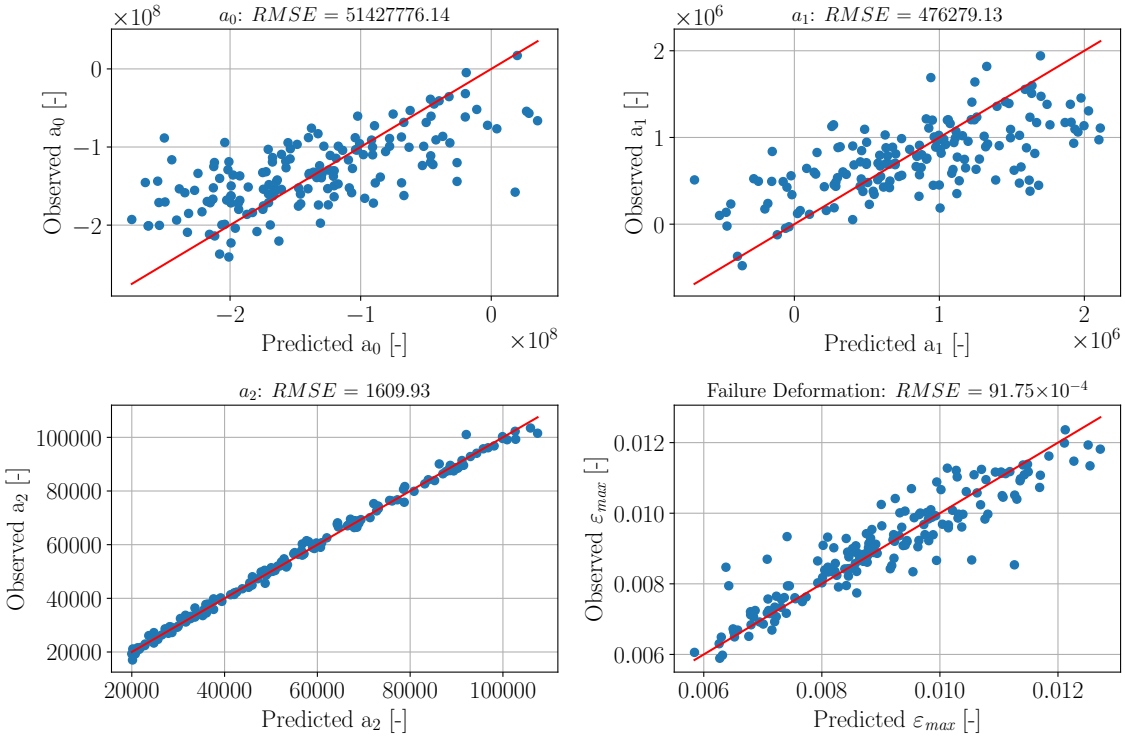


Figure 6.21: Errors in polynomial coefficient and failure deformation predictions for the Gaussian Processes (GP) based Regressors trained on *DATABASE3*.

Table 6.18: RMSE values for each of the four targets of the four stress-strain curve predictors trained on *DATABASE3*.

	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor
a_0 [-]	5.14E+07	5.56E+07	4.70E+07	4.64E+07
a_1 [-]	4.76E+05	5.32E+05	4.40E+05	4.32E+05
a_2 [-]	16.09E+02	16.59E+02	21.63E+02	29.95E+02
ε_{max} [-]	61.42E-05	113.64E-05	60.97E-05	67.9E-05

The polynomial curve predictions (not accounting for the Failure Deformation predictions) are evaluated using the RAE metric. This is shown in Figure 6.22 and Table 6.19.

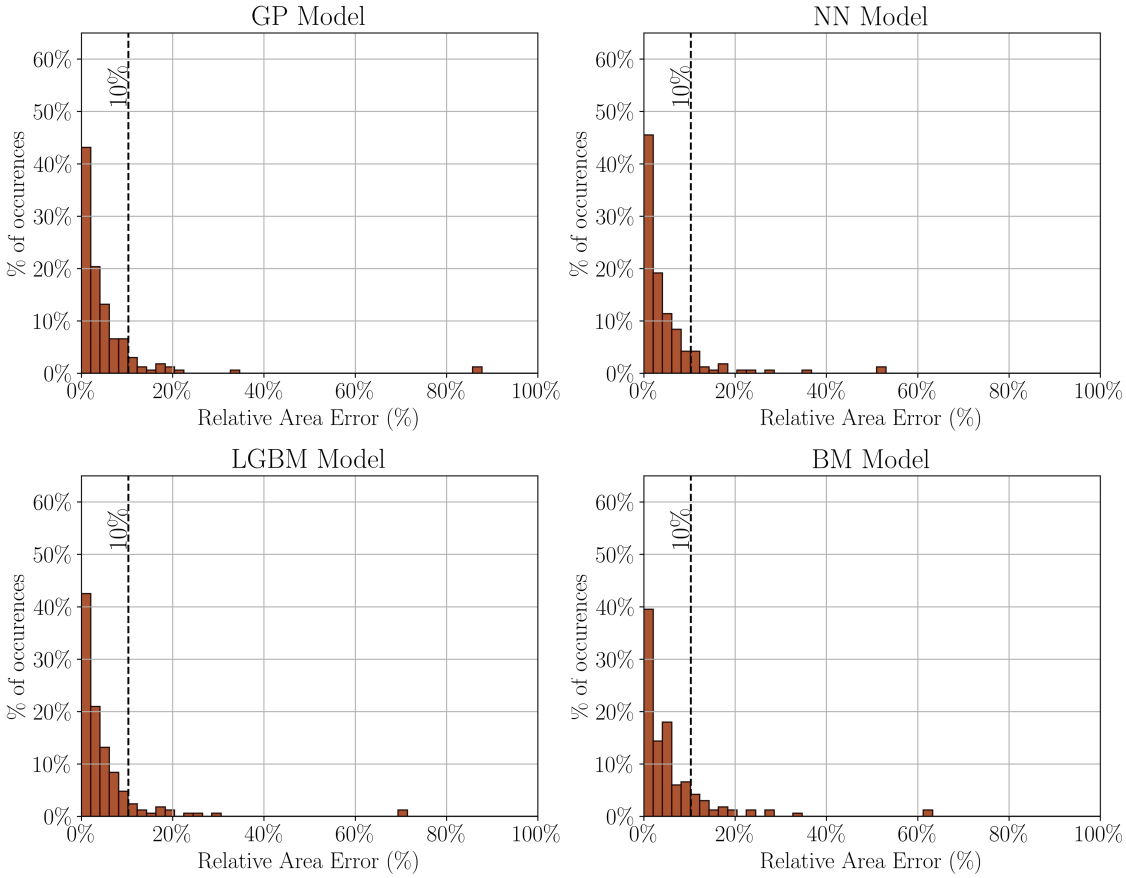


Figure 6.22: Relative area error histograms for the four Curve Predictor models trained on *DATABASE3*.

Table 6.19: Mean and Median RAE for the four Curve Predictor models trained on *DATABASE3*.

	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor
Mean RAE [%]	5.13	4.81	5.08	5.86
Median RAE [%]	2.66	2.61	2.92	3.64

Once again, as seen in Section 6.4.2, the models are much better at predicting a_2, the coefficient related to the linear part, and Failure Deformation than they are at predicting a_0 and a_1.

Interestingly, the analysis of Table 6.18 seems to show that the tree based regressors are better at predicting the non-linear coefficients, a_0 and a_1; the GP regressor is the best at predicting a_2 (closely followed by the NN regressor); and the GP regressor the best at predicting Failure Deformation, being the NN regressor much worse than its counterparts. It would be possible to envision a scenario where some of the regressors in a regressor chain are based on one ML algorithm and other regressors in the chain based on other ML algorithms.

When analysing Figure 6.22 and Table 6.19, the NN regressor seems to be the better performing, with the lowest mean and median values. Following it are the GP and LGBM models, and finally, as the worst performing, the BM predictor.

6.4.3 Learning curves

To evaluate the sufficiency of data, learning curves are plotted in Figure 6.23, for all models and each of the four targets. In this analysis, models are not chained, each of them uses only the original features.

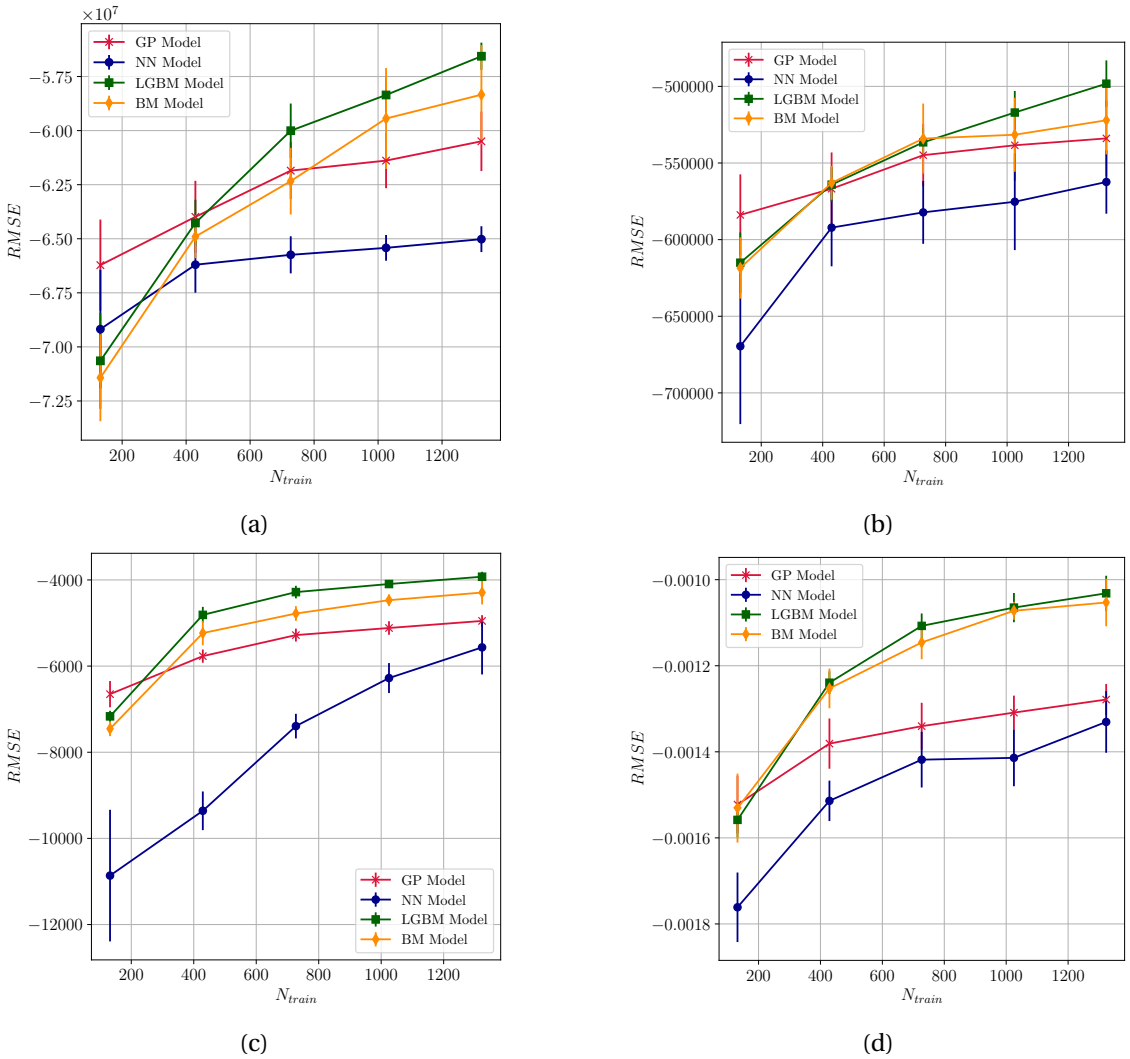


Figure 6.23: Learning curves for stress-strain Curve Predictors trained on *DATABASE3*, using RMSE score for each of the four targets: (a) a_0; (b) a_1; (c) a_2; (d) ε_{max}.

The analysis here presented seems to show that an increase in number of training datapoints could lead to better performance for all targets and most of the models. Note that the NN predictor being the worst performing in Figure 6.23 could be explained by its greater sensitivity

to hyperparameters, that are fixed for this analysis. Furthermore, these scores are validation scores, that although being measures of predicted capability of the models are not the same as evaluations on a test set. In addition, note that in this analysis, regressors are not chained, as was explained.

6.4.4 Strength Predictions

All four targets of the models, when combined, yield the polynomial curve prediction of the stress-strain curve, truncated at the predicted Failure Deformation. From this, the maximum stress can be predicted. This is an indirect way of obtaining the Ultimate Strength. All the analysis made in Section 5.2 can be reproduced. This is shown in Appendix D.2. Once again, models seem to show good performance in general: worse than that of the direct Ultimate Strength regressors, and with a tendency to under-estimate strength values, that is attributed to the chosen order of the fitted polynomials.

6.5 Generation of Design Allowables

Up to this point, the predictions performed for a given specimen took into account only its average material properties and geometric dimensions. In this Section, the variability of these parameters will also be considered. This will allow for the generation of virtual Design Allowables (see Section 2.2.2).

To generate Design Allowables, only the best performing model will be used, that is, the Gaussian Processes (GP) based Ultimate Strength regressor. Variability of the input parameters will be taken from the literature or will be assumed. By generating a sufficient number of points around the mean values of these features, sampled according to their variability by assuming a normal or uniform distribution, a distribution of the notched strength of the specimens will be obtained. From this, a B-basis Design Allowables will be generated by computing the 10th percentile of the distribution. This is a Monte Carlo based approach (Vallmajó et al. (2019)). 10000 points will be generated and their strength computed to create the distribution, an approach similar to the one performed by Furtado et al. (2021), which should ensure sufficient confidence in the 10th percentile.

Angle variability will not be considered here, as layups with orientations different from 0°, ±45 degree and 90° would yield inadmissible laminate features to input in the ML model. The variability of material properties for material IM7/8552 is presented in Table 6.20, and for all of these normal distributions are assumed. Material variability was based on the work of Camanho et al. (2007). The variability on geometric features is presented in Table 6.21, and, for these, uniform distributions are used (Furtado et al. (2021)).

Table 6.20: Variability of material properties of the IM7/8552.

	E_1 [MPa]	G_{12} [MPa]	X_T [MPa]	X_C [MPa]	S_L [MPa]	G_{XT} [kJ/m^2]	G_{XC} [kJ/m^2]	G_{iic} [kJ/m^2]
Mean Value	171420	5290	2357.32	1200.07	92.34	133.3	61	0.7879
Standard Deviation	2380	6.877	150.26	145.68	0.57	13.3	6.1	0.0803

Table 6.21: Variability of geometric features.

	D [mm]	W [mm]	t [mm]
Mean Value	6.35	38.1	0.125
Variation Range	[6.223, 6.477]	[37.338, 38.862]	[0.1225, 0.1275]

First, Design Allowables will be generated for a data point already presented: Layup ID3 from Xu et al. (2016). For this data-point, experimental results exist for 19 specimens. Furthermore, 61 numerical simulations were ran using the model from Furtado et al. (2019). In these simulations, fiber variability was considered: fibers were allowed to vary ±3°.

It was not feasible to run a number of FEM simulations close to the 10000 that were used to (almost instantly) generate the distribution by the ML model, as this would be computationally too expensive. With the estimate of 14 minutes[8] per simulation, 10000 cases would need more than 97 days for results to be available. For result comparison purposes, the 61 simulations that were ran were deemed sufficient.

In Figure 6.24, the experimental, numerical and ML based Empirical Cumulative Distribution Functions (ECDF) are presented. Vertical lines show the mean values and 10th percentile for all distributions.

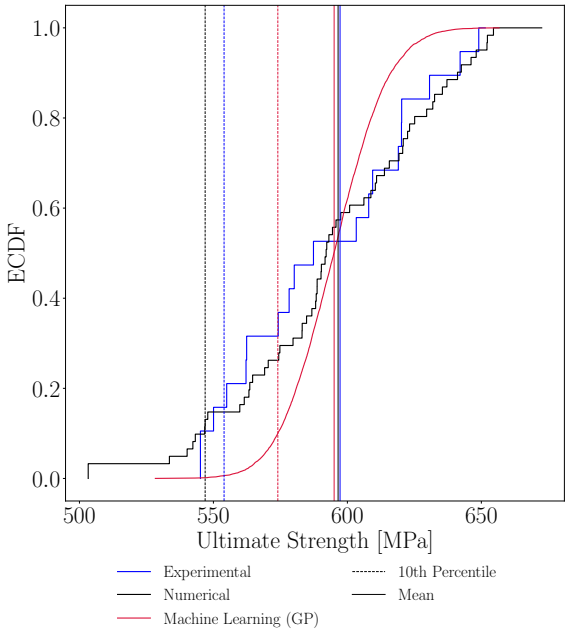


Figure 6.24: Experimental, numerical and ML generated design allowables for Clarkson (2012), Layup ID3.

Design allowables will be computed using the same strategy for two data points presented in earlier Sections: from Camanho et al. (2007) data, the specimen with D = 4mm; and from Xu

[8] Considering once again the use of 24 CPUs.

et al. (2016), the specimen with D = 3.18mm. For these, only experimentally obtained mean values and standard deviations are available. For comparison purposes, we will assume a normal distribution of results with those parameters.

The same material variability as presented in Table 6.20 will be used, assuming normal distributions (the material is for all examples the IM7/8552). For D and W geometric parameters, a variability of 2% of their mean values was considered, and once again an uniform distribution used. Again, ply angle variability will not be considered.

Results are presented, in the shape of ECDF curves, in Figure 6.25 and Figure 6.26, for the data-point from Camanho et al. (2007) and Xu et al. (2016), respectively.

Relative Errors registered for the mean value predictions are of 0.7% and 0.5% for the data-point from Camanho et al. (2007) and Xu et al. (2016), respectively. Relative errors for the 10th percentile registered are of 4.0% and 0.7% for the data-point from Camanho et al. (2007) and Xu et al. (2016), respectively.

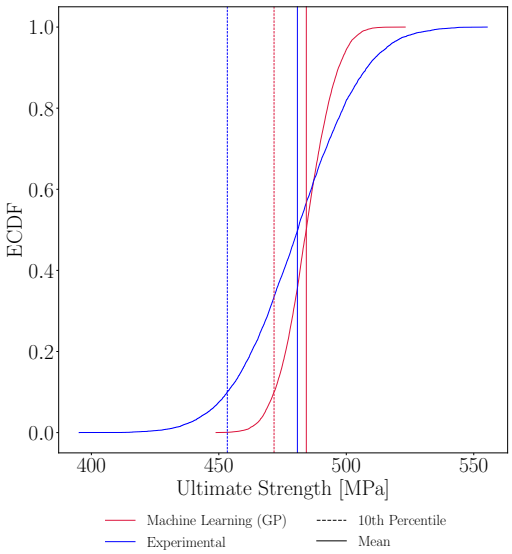


Figure 6.25: Experimental and ML generated design allowables for Camanho et al. (2007), D = 4mm.

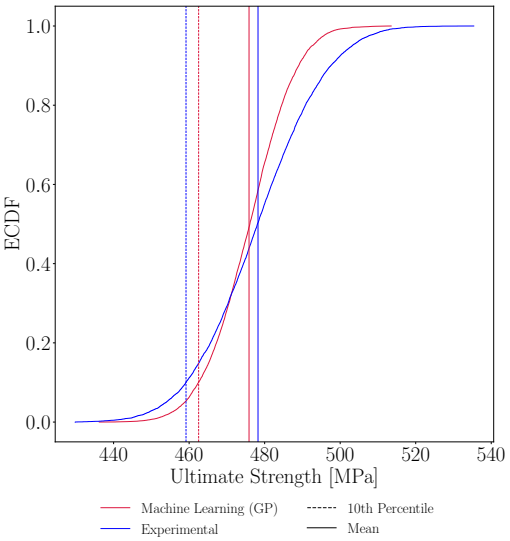


Figure 6.26: Experimental and ML generated design allowables for Xu et al. (2016), D = 3.18mm.

The ML model will now be tested for a different material, for which numerical results that include material variability are available. These results are available for 18 different stacking sequences, all corresponding to the same quasi-isotropic layup. This presents itself as a challenging task, as not only should the ML model capture material variability, but it must capture well the variation of average Ultimate Strength when varying solely the ply order. 101 numerical simulations were performed for each stacking sequence.

Material properties, stacking sequences and results in absolute value will not be presented due to confidentiality reasons.

In Figure 6.27 an overview of the results is presented. Here the mean value extracted from the 101 simulations and predicted by the ML models on those same data points is presented. The standard deviation of these results is also shown in the form of the error bars.

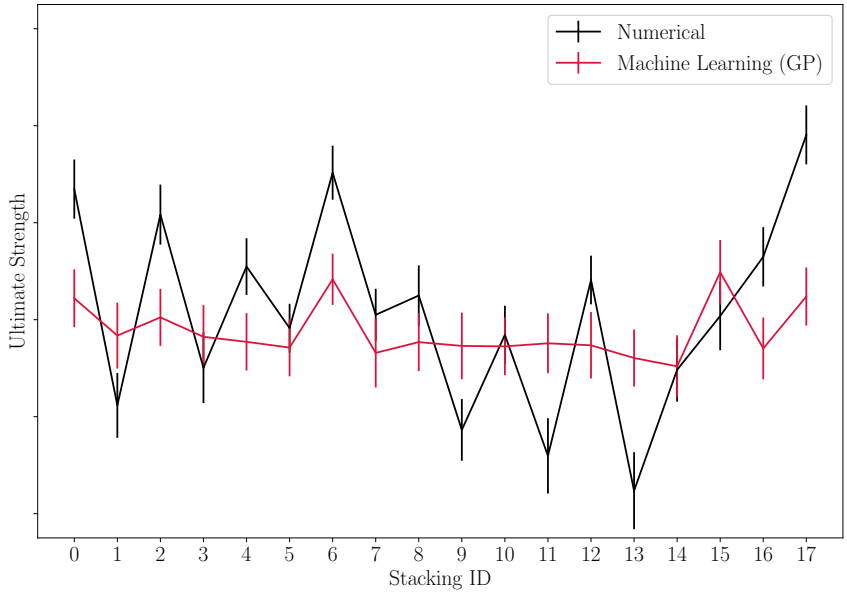


Figure 6.27: Predictions for 18 different stacking sequences, with the same material, geometry and layup, using the trained model after Feature Selection (12 selected features).

In Figure 6.28 the same numerical results are presented. The ML model here used was one in which no FS was performed, it uses all 25 features, with the objective of better capturing ply order variation trends. It is the model presented in Appendix C.4, where it is seen that, on the test set, performance differences were insignificant.

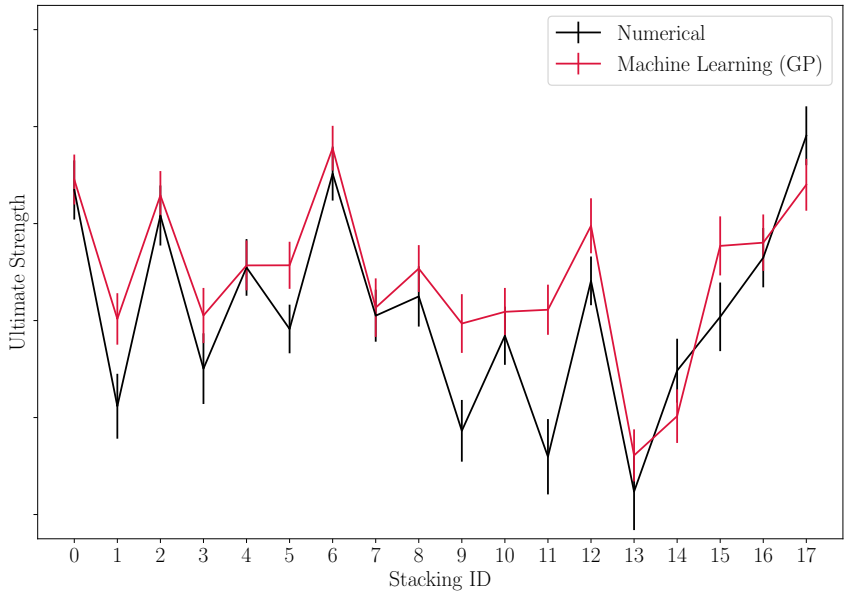


Figure 6.28: Predictions for 18 different stacking sequences, with the same material, geometry and layup, using the trained model without Feature Selection (all 25 features are used).

Ultimate strength distributions were generated using the GP regressor for all 18 stacking sequences, again using 10000 points. Only material variability was considered, as this was also the only variability considered in the numerical simulations. For this, the model that used all features was chosen. The mean and 10th percentile results taken from these distributions will be compared with the same values from the distributions coming from the numerical results.

Absolute value of Relative Errors are presented in Table 6.22. Additionally, in Figure 6.29, three examples are shown: the first stacking sequence; the stacking sequence for which the results were the best; and the stacking sequence for each the results were worst, in the shape of ECDF curves. These were generating by employing the model that uses all 25 features as input.

6.5.1 Discussion

In this section, the potential to use data-driven ML models to generate Design Allowables was showcased. Their ability to make predictions on many thousands of points in mere seconds is for sure their main advantage, allowing for the use of the Monte Carlo approach.

In Figure 6.24, it is clearly seen that the model was able to predict the mean value with great accuracy (relative error of 0.4% and 0.2% when compared to the experimental and numerical campaigns' mean strength values, respectively), as had already been established in previous Sections. The ML generated distribution of results seems to show less variability. This resulted in relative errors on the 10th percentile (that would be the B-basis allowable) of 3.6% and 5.0%, relative to experimental and numerical results, respectively. This can be at least partially attributed to the fact that the ML models here developed are not capable of taking into account variability on ply orientation. This variability was taken into account in the numerical simulations (and is inherently present in the experimental tests campaign, evidently).

The results presented in Figures 6.25 and 6.26 show a similar trend: a very accurate mean value prediction is registered but the model's results seem to present lower variance. Again, this can be attributed to the lack of capability of the models in using fiber orientation variability.

The results shown in Figures 6.27 and 6.28 are very interesting. The model that used only 12 features, selected by the RFECV FS method seems to not be able to capture trends when changing stacking sequence, maintaining all other parameters fixed (including layup), despite it using some features solely related and engineered for that purpose (APB, ζ_2^D, C_4), as seen by the mostly flat line throughout most stacking sequences. These seem to not be sufficient to capture the importance of ply order. The model that used all features (further including the ply order related features MA, ζ_1^D, ζ_3^D, $\zeta^{\bar{D}}$, C_1, C_2, C_3) can capture these trends in a much more accurate way. Furthermore, as seen in Table 6.22, the models registered relative errors in the 10th percentile similar to those registered for the mean value, which indicates that the ML models found a distribution with similar variability to that found by the numerical models, as can also be seen in Figure 6.29. It is interesting to note that this ability to capture trends in ply order was not captured by the evaluation made on the test set (Appendix C.4).

This is a very positive result: this is a fairer comparison than the ones previously presented as it is the only case where the ML models were able to include variability in the same parameters as the ones of the reference case (in this case the numerical model). Both models varied only the material properties. We see then that when the mean value prediction matches the numerical result, as in Stacking ID4, the allowable values would also match. Of course, when the mean value prediction is less accurate, as in Stacking ID11 (with a RE lower than 8%, though),

the 10th percentile shows a similar error (hence the distributions are shifted from each other).

6.6 Concluding Remarks

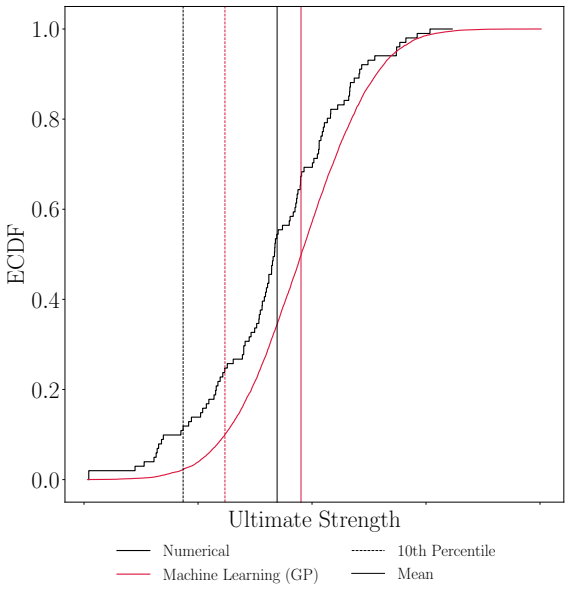
In this Chapter it was shown that it is possible to train accurate and very efficient Surrogate Models for specimens under OHT, for a wide range of material properties, as well as with geometric and stacking sequence variability.

The Ultimate Strength Regressors and stress-strain Curve Predictors displayed outstanding results. In particular, the best performing model, based on Gaussian Processes, was able to capture the important patterns in data, as shown in the sensitivity analysis, and presented good performance when compared to real data from more than one material. It was shown that the model can achieve these performance when trained with a reasonable number of data-points. The model classifier displayed reasonable performance, although still not very satisfactory. It could potentially improve with more training data.

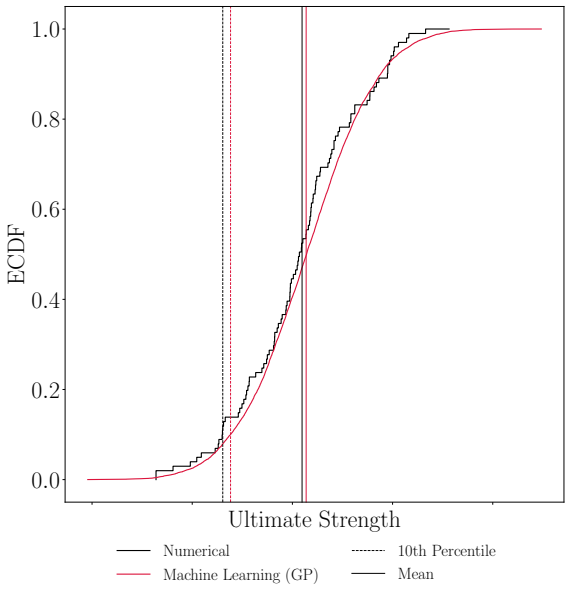
In the final part of the Chapter, variability in material and geometric parameters were included, for Design Allowable generation. It was concluded that including ply angle variability should be an important to futurely have included. Despite this, it was shown that good results can be obtained. This was evident on the final analysis, where the same variability was used for both a high-fidelity numerical model and the trained ML model. Furthermore, it was shown that provided with the relevant engineered features, the models can much more accurately capture the influence of ply order on ultimate strength. In terms of computational costs, the order of magnitude of prediction time by the ML models, in the order of fractions of a second, is much lower than that of the time it takes to run a FEM simulation, which is on average over 14 minutes.

Table 6.22: Absolute value of relative errors for mean value and 10th percentile for the 18 different stacking sequences.

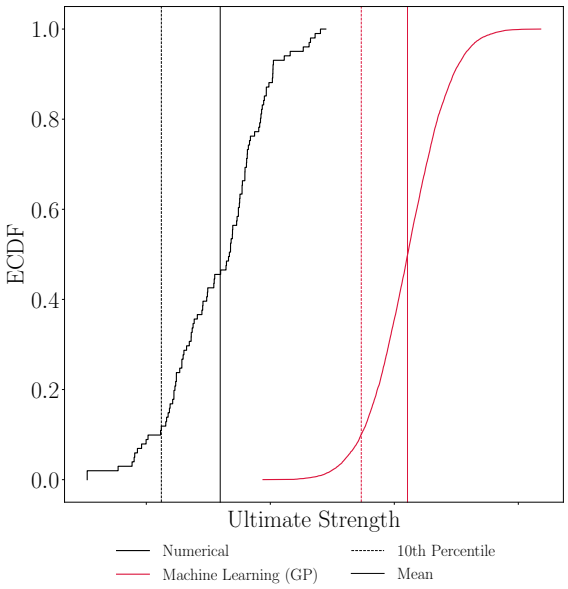
Stacking ID	Absolute RE Mean Value of Ultimate Strength	Absolute RE 10th Percentile of Ultimate Strength
0	0.49%	0.85%
1	4.48%	5.25%
2	0.93%	1.21%
3	2.67%	3.08%
4	0.08%	0.18%
5	3.12%	3.10%
6	1.17%	1.35%
7	0.40%	0.17%
8	1.35%	1.68%
9	5.57%	6.17%
10	1.17%	1.05%
11	7.73%	8.48%
12	2.64%	2.24%
13	1.95%	2.70%
14	2.28%	1.97%
15	3.45%	3.95%
16	0.72%	0.79%
17	2.21%	1.93%



(a) Stacking ID0.



(b) Stacking ID4.



(c) Stacking ID11.

Figure 6.29: Numerical and ML generated design allowables for three example stacking sequences.

Chapter 7

Conclusions

Within this work, a methodology for building fast and accurate Machine Learning surrogate models based on high-fidelity numerical simulations was used to successfully train models that predict strength, failure and general shape of stress-strain curves for CFRP laminates under OHT. It was shown that these models can be trained with a reasonable amount of data-points, that are computationally expensive to generate. This is achieved with a well thought-out Design of Experiments. Predictions made with the models here developed have negligible computational costs, as results are obtained in fractions of a second.

7.1 Fixed material properties models

The models built for a fixed material displayed excellent results. From the four trained Ultimate Strength regressors, the Gaussian Processes based model showed the overall best performance. It was capable of making predictions on never before seen data, be it the test set data-points, the points from the sensitivity analysis or the examples from experimental test campaigns taken from the literature. The Neural Network regressor displayed some potential to be used for extrapolation.

The Failure Mode classifier was able to make accurate predictions on never before seen data, even accounting for the fact that the fiber dominated Failure Mode is much more prevalent on the population of specimens. Once again the Gaussian Processes based model was the best classifier.

Regarding the Curve Predictors the Gaussian model was again the best performing model. It evidenced the ability to accurately predict the linear part of the stress-strain curves, and also to predict the deformation at failure. Capturing the non-linear parts of the curves was a more difficult task, although the model was able to do this with sufficient accuracy, such that the defined curve evaluation metrics were very satisfactory. These models were used to indirectly predict Ultimate Strength of the specimens, showing very interesting results, although, as expected, worse than that of the direct Ultimate Strength regressors. The models tend to underestimate the strength of the specimens, which is attributed to the order of the polynomials being fitted to the true stress-strain curves. This problem should be mitigated by increasing polynomial order.

7.2 Variable material properties models

When expanding the model building to variations of material properties, the ML models maintained great performance, in general. The strength regressors were able to be accurate on the test set and to capture trends, as was showcased in the performed sensitivity analysis. When used on real examples, small enough errors were obtained, especially between the model's predictions and numerical results from FEM simulations. In general, larger errors were obtained when comparing the model's predictions with experimental results. This is because in these case there is another source of inaccuracy, related to the performance of the high-fidelity models themselves. Their performance on the IM7/8552 material was, as expected, slightly worse than that of the models specialized in this material. The best performing model was the Gaussian Processes based regressor, that showed the ability to make accurate predictions on other materials. The models presented difficulty in predicting strength values for softer laminates, which is attributed to the material property assumptions that were made.

The Failure Classifiers' performance, although having high levels of accuracy, cannot be said to be able to make good predictions when taking into account the unbalance in the dataset. The best performing model was the decision tree based Bagging Method model. The number of true negatives was only slightly larger than the number of false negatives on the test set, and therefore balanced accuracy scores did not surpass 80%. These models could improve by being trained on more data; by performing outlier detection based on their features or by rethinking the approach to this problem. The problem is more difficult than in the fixed material case because the failure mode can depend on several material properties.

Regarding the stress-strain Curve Predictors, these were able to capture the general shape of the curves. Once again, the best model was the Gaussian Processes based model. The ML models were able to capture the linear part of the curves and the failure deformation. When evaluated using the area based metrics, the models perform was to be expected, than those trained for a fixed material, as the task at hand in this case is more difficult. The models displayed interesting results when indirectly predicting the Ultimate Strength of the specimens, even for different materials, although still suffering from problems when estimating the strength of soft laminates, as was the case with the direct Ultimate Strength regressors.

7.3 Design Allowable generation

The Ultimate Strength regressors were used to generate Design Allowables, as it was now possible to include material variability, in addition to the geometric variability. It was concluded, in general, that the ML models' resulting strength distributions have lower variance than that of experimental results and numerical simulations. This is attributed to the lack of the capability of the models to include variability in ply orientation direction. When the models were compared to numerical results that included solely material variability, a fairer comparison, the ML generated distributions showed very similar variances. This analysis showed that the ML models, by having the ability to produce thousands of predictions instantaneously, have great potential for generation of Design Allowables. For this to be feasible and acceptable by regulatory entities, though, all relevant sources of variability (including fiber direction) should be included, and the models should be sufficiently accurate when predicting variance and mean value (which is achieved by simultaneously improving numerical model accuracy in

relation to real tests, and ML model performance in relation to numerical models).

7.4 Future work

There is still much to explore in the applicability of the models here developed and in the space of similar models.

Regarding the Machine Learning framework here proposed, it could be very interesting to experiment other more complex stratification strategies: although literature shows that it is important to have attention to the way in which splits in data are made, the single feature strategy here employed may be too simple to make significant differences. Furthermore, more complex outlier detection strategies, that account for more than one dimension at each time, could be very beneficial, especially for multi-output models, and could help in making a more efficient use of the generated data. Additionally, more efficient data generation could potentially be achieved by performing an online Design of Experiments, that is, for example choosing the following data-point to simulate based on previous points, possibly using ML models for these choices.

To improve the performance of the failure classifier models, besides enlarging the training dataset, other approaches could be tested, such as the development of multi-output models that predict the energies at failure directly, instead of trying to classify the failure mode.

The models here built can easily be expanded to be used, for example for other types of laminates, such as Double-Double ply arrangements. For this case, the features engineered here would probably not be as useful to capture ply order impact on results, but ply angles could be directly used. Furthermore, other coupon tests could be experimented with. Open Hole Compression simulations could be trivially experimented with, even using the same Design of Experiments here used. Filled hole, low velocity impact, bolted joint, edge impact cases could be dealt with in a similar fashion, provided accurate numerical models are available.

To generate more accurate Design Allowables, in the future, it would be important to be able to include ply angle variability in the input of the models. Furthermore, it should be interesting to try to train models without the material assumptions that were here made for the case of variable material. This could lead to the necessity of generating larger datasets. The design allowable generation could also potentially be done, for a single material, by training models with data-points that are inside the variability range of the chosen material. This would be an interesting approach in which material variability would be used by the models but they would still be specialized in a single material.

Appendix A

Hyperparameter Grid

Table A.1: Hyperparameter Grids for the four ML models.

Model	Hyperparameter	Best Value
Gaussian Regressor	`kernel`	Mátern(nu = 0.5, 1.5, 2.5, 4.5) RBF
Neural Regressor	`alpha`	1.0, 0.1, 0.001
	`batch_size`	25, 50, 100
	`hidden_layers`	(64, 32, 16), (32,32, 32), (32,32)
	`activation`	'relu', 'tanh'
	`learning_rate_init`	0.1, 0.01, 0.001
	`max_iter`	1000, 2000
	`optimizer`	'adam', 'lbfgs'
LightGBM Regressor	`colsample_bytree`	0.6, 0.8, 1.0
	`subsample`	0.1, 0.5, 1.0
	`min_child_samples`	1, 5, 10
	`learning_rate`	0.01, 0.1, 0.5
	`max_depth`	10, 20, -1
	`num_leaves`	5, 10, 20
	`n_estimators`	100, 400, 600, 800
Bagging Regressor	`max_features`	0.5, 0.75, 1.0
	`max_samples`	0.5, 0.75, 1.0
	`n_estimators`	400, 600, 800, 1000

Appendix B

Sensitivity Analysis

Material properties are kept constant and correspond to those of the IM7/8522 material, for all cases in Tables B.1 and B.2.

For all cases in Table B.3, the geometry and laminate parameters used were those of case ID0, seen in Tables B.1 and B.2.

Table B.1: Input for the FEM simulations with the objective of performing a sensitivity analysis on the geometric features and in-plane laminate parameters.

ID	D [mm]	W/D [-]	ζ_1^A [-]	ζ_2^A [-]	ζ_1^D [-]	ζ_2^D [-]	ζ_3^D [-]	NP [-]
0	6	5.5	0.00	0.00	0.000	0.000	0.188	16
1	**2.0**	5.5	0.00	0.00	0.000	0.000	0.188	16
2	**4.0**	5.5	0.00	0.00	0.000	0.000	0.188	16
3	**8.0**	5.5	0.00	0.00	0.000	0.000	0.188	16
4	**10.0**	5.5	0.00	0.00	0.000	0.000	0.188	16
5	6.0	**3.0**	0.00	0.00	0.000	0.000	0.188	16
6	6.0	**4.5**	0.00	0.00	0.000	0.000	0.188	16
7	6.0	**6.5**	0.00	0.00	0.000	0.000	0.188	16
8	6.0	**8.0**	0.00	0.00	0.000	0.000	0.188	16
9	6.0	5.5	**0.25**	0.00	0.004	0.000	0.188	16
10	6.0	5.5	**0.50**	0.00	0.313	-0.375	0.469	8
11	6.0	5.5	**-0.25**	0.00	-0.004	0.000	0.188	16
12	6.0	5.5	**-0.50**	0.00	-0.500	0.000	0.177	24
13	6.0	5.5	0.00	**0.50**	0.000	0.195	0.258	16
14	6.0	5.5	0.00	**1.00**	0.000	1.000	0.000	16
15	6.0	5.5	0.00	**-0.50**	0.012	-0.969	0.176	16
16	6.0	5.5	0.00	**-1.00**	0.000	-1.000	-0.188	16

Table B.2: Input for the FEM simulations with the objective of performing a sensitivity analysis on the Number of Plies and out-of-plane laminate parameters.

ID	D [mm]	W/D [-]	ζ_1^A [-]	ζ_2^A [-]	ζ_1^D [-]	ζ_2^D [-]	ζ_3^D [-]	NP [-]
0	6.0	5.5	0.0	0.0	0	0	0.188	16
17	6.0	5.5	0.0	0.0	0.094	-0.750	-0.281	**8**
18	6.0	5.5	0.0	0.0	0.052	-0.250	-0.073	**24**
19	6.0	5.5	0.0	0.0	**0.141**	0.000	0.188	16
20	6.0	5.5	0.0	0.0	**0.352**	0.000	0.188	16
21	6.0	5.5	0.0	0.0	**-0.141**	0.000	0.188	16
22	6.0	5.5	0.0	0.0	**-0.352**	0.000	0.188	16
23	6.0	5.5	0.0	0.0	-0.094	**0.188**	0.188	16
24	6.0	5.5	0.0	0.0	-0.070	**0.328**	0.188	16
25	6.0	5.5	0.0	0.0	-0.070	**-0.094**	0.188	16
26	6.0	5.5	0.0	0.0	0.070	**-0.422**	0.188	16
27	6.0	5.5	0.0	0.0	0.000	0.000	**0.234**	16
28	6.0	5.5	0.0	0.0	0.000	0.000	**0.398**	16
29	6.0	5.5	0.0	0.0	0.000	0.000	**-0.188**	16
30	6.0	5.5	0.0	0.0	0.000	0.000	**-0.234**	16
31	6.0	5.5	0.0	0.0	0.000	0.000	**-0.398**	16

Table B.3: Input for the FEM simulations with the objective of performing a sensitivity analysis on the material property features.

ID	E_1 [MPa]	G_{12} [MPa]	X_T [MPa]	X_C [MPa]	S_L [MPa]	G_{XT} [kJ/m^2]	G_{XC} [kJ/m^2]	G_{IIC} [kJ/m^2]	t [mm]
00	170000	3900	2750	1350	80.0	225.0	95.0	2.250	0.1750
32	**145000**	3900	2750	1350	80.0	225.0	95.0	2.250	0.1750
33	**155000**	3900	2750	1350	80.0	225.0	95.0	2.250	0.1750
34	**185000**	3900	2750	1350	80.0	225.0	95.0	2.250	0.1750
35	**195000**	3900	2750	1350	80.0	225.0	95.0	2.250	0.1750
36	170000	**2000**	2750	1350	80.0	225.0	95.0	2.250	0.1750
37	170000	**2850**	2750	1350	80.0	225.0	95.0	2.250	0.1750
38	170000	**4950**	2750	1350	80.0	225.0	95.0	2.250	0.1750
39	170000	**5800**	2750	1350	80.0	225.0	95.0	2.250	0.1750
40	170000	3900	**2100**	1350	80.0	225.0	95.0	2.250	0.1750
41	170000	3900	**2400**	1350	80.0	225.0	95.0	2.250	0.1750
42	170000	3900	**3100**	1350	80.0	225.0	95.0	2.250	0.1750
43	170000	3900	**3400**	1350	80.0	225.0	95.0	2.250	0.1750
44	170000	3900	2750	**950**	80.0	225.0	95.0	2.250	0.1750
45	170000	3900	2750	**1125**	80.0	225.0	95.0	2.250	0.1750
46	170000	3900	2750	**1575**	80.0	225.0	95.0	2.250	0.1750
47	170000	3900	2750	**1750**	80.0	225.0	95.0	2.250	0.1750
48	170000	3900	2750	1350	**65.0**	225.0	95.0	2.250	0.1750
49	170000	3900	2750	1350	**72.5**	225.0	95.0	2.250	0.1750
50	170000	3900	2750	1350	**87.5**	225.0	95.0	2.250	0.1750
51	170000	3900	2750	1350	**95.0**	225.0	95.0	2.250	0.1750
52	170000	3900	2750	1350	80.0	**70.0**	95.0	2.250	0.1750
53	170000	3900	2750	1350	80.0	**137.5**	95.0	2.250	0.1750
54	170000	3900	2750	1350	80.0	**312.5**	95.0	2.250	0.1750
55	170000	3900	2750	1350	80.0	**380.0**	95.0	2.250	0.1750
56	170000	3900	2750	1350	80.0	225.0	**50.0**	2.250	0.1750
57	170000	3900	2750	1350	80.0	225.0	**67.5**	2.250	0.1750
58	170000	3900	2750	1350	80.0	225.0	**122.5**	2.250	0.1750
59	170000	3900	2750	1350	80.0	225.0	**140.0**	2.250	0.1750
60	170000	3900	2750	1350	80.0	225.0	95.0	**0.800**	0.1750
61	170000	3900	2750	1350	80.0	225.0	95.0	**1.375**	0.1750
62	170000	3900	2750	1350	80.0	225.0	95.0	**3.125**	0.1750
63	170000	3900	2750	1350	80.0	225.0	95.0	**3.700**	0.1750
64	170000	3900	2750	1350	80.0	225.0	95.0	2.250	**0.0800**
65	170000	3900	2750	1350	80.0	225.0	95.0	2.250	**0.1125**
66	170000	3900	2750	1350	80.0	225.0	95.0	2.250	**0.2375**
67	170000	3900	2750	1350	80.0	225.0	95.0	2.250	**0.2700**

Appendix C

Analysis on Machine Learning framework parameters

C.1 Cross-Validation parameters

The purpose of this analysis is to evaluate the influence of parameters in the Cross-Validation (CV) process as number of repetitions; number of stratification bins and stratification variable on model performance. Analysis was done only for the GP and LGBM models.

For this, a test set was split from *DATABASE2*, without any stratification (plain random split). In all cases, a 5-fold Cross-Validation was performed. The first case serves as term of comparison, here CV was performed without stratification and only 1 repetition was done.

Results are presented in Table C.1. Both the training score (from the CV) and the test score (performance on the test set) were presented.

Table C.1: Train and test scores when varying CV parameters for Ultimate Strength regressors trained on *DATABASE2*.

			GP		LGBM	
Stratification Variable	Number of Repeats	Number of Bins	Train RMSE [MPa]	Test RMSE [MPa]	Train RMSE [MPa]	Test RMSE [MPa]
None	1	-	11.80	12.77	14.81	11.93
W/D	3	3	11.80	12.82	14.62	11.84
W/D	1	5	12.06	13.09	15.36	12.20
W/D	1	3	11.80	12.64	14.79	12.47
D	1	3	11.80	12.91	15.02	12.13
ζ_1^A	1	3	11.80	13.31	14.92	11.84
ζ_2^A	1	3	11.80	13.31	14.82	12.20
ζ_1^D	1	3	11.80	12.81	14.60	11.73
ζ_2^D	1	3	11.80	13.11	15.26	12.43
ζ_3^D	1	3	11.80	12.40	14.60	12.43

The GP model chose the same hyperparameters almost every time. The LGBM chose different hyperparameter combinations (it has a wider range of options), but no trend in per-

formance improvement was detected. It was concluded that these factors do not influence significantly the model performance.

C.2 Comparison on multi-output regression approaches

The purpose of this analysis is to evaluate the influence of the multi-output approach. The two approaches here evaluated are taking advantage of the inherent possibility to handle multi-outputs of some regressors against an approach in which four single-output regressors are chained together.

For this, a test set was split from *DATABASE2*, the same as the one used in Section 5.4. The analysis will be done only for the GP regressor, on the basis of it being the best performing model, its low training time, and its inherent multi-output capabilities. It is also possible to perform inherent multi-output regression for the NN and BM regressors, but not for the LGBM, at least on its current implementation.

Results are presented in Table C.2. RMSE for each parameter, the median Relative Area Error and the indirectly predicted stress RMSE are reported.

Table C.2: Inherent and Chain regressors approach comparison for stress-strain Curve Predictors trained on *DATABASE2*.

Order	RMSE a0	RMSE a1	RMSE a2	RMSE def	median REA	Stress RMSE [MPa]
Inherent	4.83E+07	3.68E+05	7.78E+02	2.82E-04	1.46	13.53
Chain	4.89E+07	3.78E+05	12.62E+02	2.79E-04	1.50	15.36

It was concluded that the best performing approach is the chain regressors approach, although not by a large margin. In the trained models, the decision to use chain regressors has the additional advantage of enabling a fairer comparison between the GP, NN, LGBM and BM models, as the same approach will be used for all.

C.3 Order of chain regressors

The purpose of this analysis is to evaluate the influence of regressor order in the performance of chain regressors that make up the stress-strain curve predictors.

For this, a test set was split from *DATABASE2*, the same as the one used in Section 5.4. There are four regressors, each predicting one label: a_0, a_1, a_2, ε_{max}. 4 different orders will be tested (from the 24 possible permutations).

The analysis will be done only for the GP regressor, on the basis of it being the best performing model and on its low training time.

Results are presented in Table C.3. RMSE for each parameter, the median Relative Area Error and the indirectly predicted stress RMSE are reported.

Table C.3: Evaluation of four different regressor orders for GP stress-strain Curve Predictors trained on *DATABASE2*.

Order	RMSE a0	RMSE a1	RMSE a2	RMSE def	median REA	Stress RMSE [MPa]
(a_0, a_1, a_2, ε_{max})	4.83E+07	3.68E+05	7.78E+02	2.82E-04	1.46	13.53
(ε_{max}, a_2, a_1, a_0)	5.70E+07	4.26E+05	8.04E+02	2.93E-04	1.57	15.31
(a_1, a_0, a_2, ε_{max})	4.88E+07	3.71E+05	8.52E+02	2.81E-04	1.45	13.90
(ε_{max}, a_2, a_1, a_0)	4.84E+07	3.72E+05	7.90E+02	2.93E-04	1.49	15.37

It was concluded that regressor order does have some impact on overall model performance. The best of the four orders is (a_0, a_1, a_2, ε_{max}).

C.4 Evaluation of Feature Selection

In this Section, Ultimate Strength regressors that were trained using all available features (16 for those trained on *DATABASE2* and 25 for those trained on *DATABASE3*) are compared to those trained after a Feature Selection process, as presented in Chapter 5 and Chapter 6. All of the other steps for building the models are unchanged.

Results are presented in Tables C.4 and C.5, for the models trained on *DATABASE2* and *DATABASE3*, respectively. These are in form of test set RMSE score (the same test set is used).

Table C.4: Comparison between models built using Feature Selection and models that use all available features. Models trained on *DATABASE2*.

Features used by the GP model	GP RMSE [MPa]	NN RMSE [MPa]	LGBM RMSE [MPa]	BM RMSE [MPa]
8 features	9.30	10.23	13.82	18.74
All 16 features	9.37	11.01	13.46	19.81

Table C.5: Comparison between models built using Feature Selection and models that use all available features. Models trained on *DATABASE3*.

Features used by the GP model	GP RMSE [MPa]	NN RMSE [MPa]	LGBM RMSE [MPa]	BM RMSE [MPa]
12 features	17.34	20.86	24.68	29.57
All 25 features	17.63	22.11	24.62	30.67

It was concluded that the used FS process improves, in general, the performance of the models when evaluated on the test set, even though only slightly. In the case of the LGBM model, performance registered a (not significant) decrease.

Appendix D

Indirect Ultimate Strength regression by stress-strain Curve predictors

D.1 Surrogate models for an open-hole coupon of a constant material system

In Figure D.1, the observed-predicted scatter plots are presented for all four trained models. Additionally, in Table D.1, the fraction of points below RE thresholds of 2%, 5%, 10% and 20% are presented.

These indirect Ultimate Strength models were used once again for a sensitivity analysis, analogously to what was done in Section 5.2.4. This is shown in Figures D.2 and D.3.

The four ML models were used in examples that have real experimental results from the literature, as was done with the direct Ultimate Strength predictors in Section 5.2.5. In Figure D.4 and Table D.2 the models are used on data from Camanho et al. (2007) and results are presented. In Figure D.5 and Table D.3 the results of the models' predictions on data from Xu et al. (2016) are presented. Data from the three layups used by Clarkson (2012) was once again used. Results are presented in Figure D.6.

These indirect strength predictions may be compared to what was discussed in Section 5.2. It is important to note, when analysing Figure D.1 that the RMSEs are not directly comparable to those of Figure 5.4, because it is not the same test set (couldn't be because the starting dataset is different after the different outlier detection that was performed). Even so, it is evident that the predictions are very reasonable: by analysing Table D.1, we see that the models although worse performing than the direct strength predictors, do yield interesting errors. From Figure D.1, a trend for underestimating specimen strength is observed. This is attributed to the fact that even the true polynomial approximations of the real numerical curves are still only third order polynomial approximations, that themselves tend to underestimate this strength. This is easily seen in Figure D.7, where the observed ultimate strength, from the numerical simulations, is plotted against the maximum stress of the third order polynomials that were fitted on the numerical stress-strain curves. Higher order polynomials could reduce this bias, although this would have the disadvantage of adding more targets to the curve predictors.

Regarding the strength predictions, the GP model is the best performing one, once again, closely followed by the NN model.

It was shown that the models are capable of capturing the trends in strength with changing geometric features and also in-plane laminate parameters. It should be noted that the model's performance in this sensitivity analysis was only marginally worse than that of the direct strength regressors. Once again, models cannot capture the trends when varying Number of Plies and out-of-plane laminate parameters.

When compared to examples with real experimental results, the models are overall capable of good performance, even surpassing the direct regressors in some cases. It is particularly interesting to analyse Figure D.5: these curve predictors' extrapolation capabilities are not remotely close to those displayed by their direct counterparts, especially regarding the NN models. Here the results deviate from both numerical and experimental results very rapidly outside the training ranges. Besides this fact, it is possible to highlight the good performance of the models on most of the example cases (when interpolating), even though the models do not directly predict this quantity.

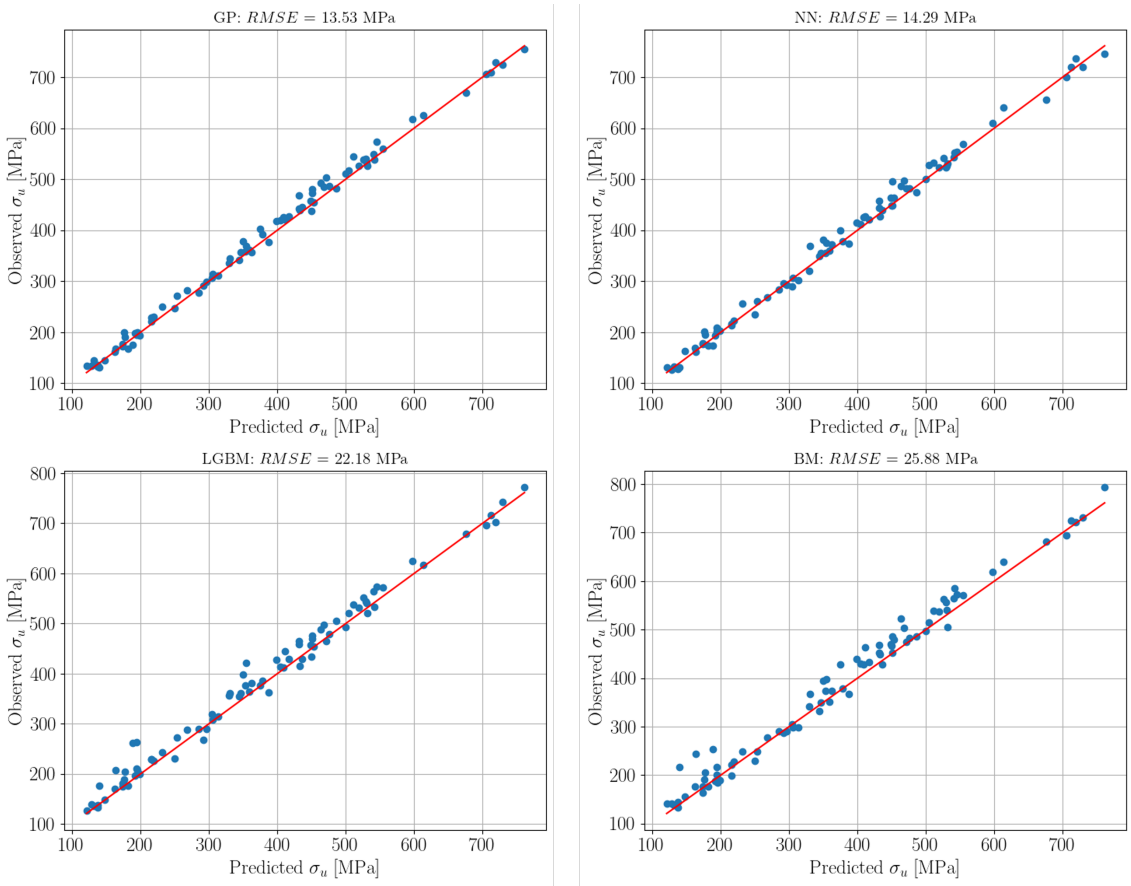


Figure D.1: Observed (from numerical simulations) and indirectly predicted Ultimate Strength for the four ML stress-strain curve predictors trained on *DATABASE2*.

Table D.1: Relative Error (RE) fractions below thresholds of 2%, 5%, 10% and 20% for the indirectly predicted Ultimate Strength by the four ML stress-strain curve predictors trained on *DATABASE2*.

	Fraction of Data below the RE Limit for each Model				
RE Limit	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor	DUMMY Regressor
2%	0.393	0.417	0.310	0.226	0.024
5%	0.774	0.773	0.619	0.583	0.071
10%	0.964	0.952	0.917	0.833	0.167
20%	1.000	1.000	0.952	0.964	0.369

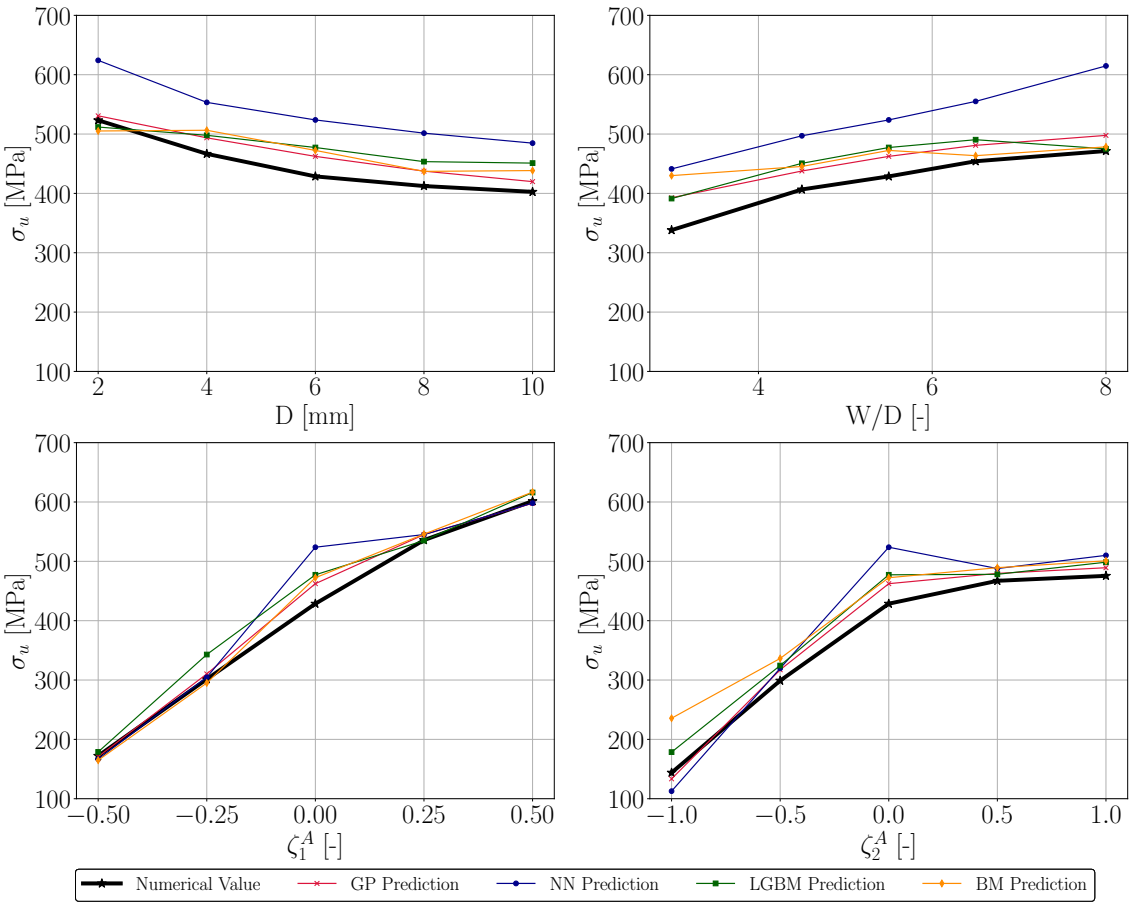


Figure D.2: Results of the sensitivity analysis of the indirectly predicted Ultimate Strength by the stress-strain curve predictors trained on *DATABASE2* for geometric features and in-plane laminate parameters.

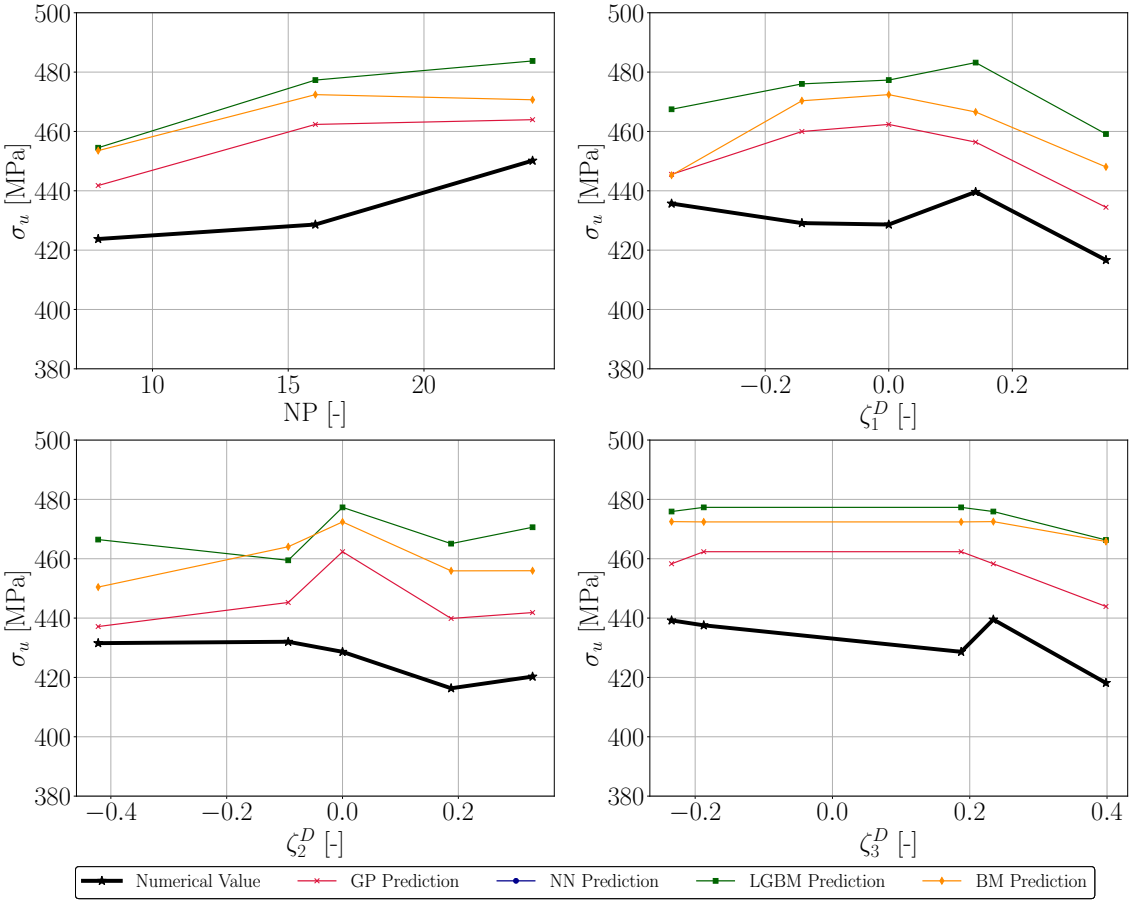


Figure D.3: Results of the sensitivity analysis of the indirectly predicted Ultimate Strength by the stress-strain curve predictors trained on *DATABASE2* for number of plies and out-of-plane laminate parameters.

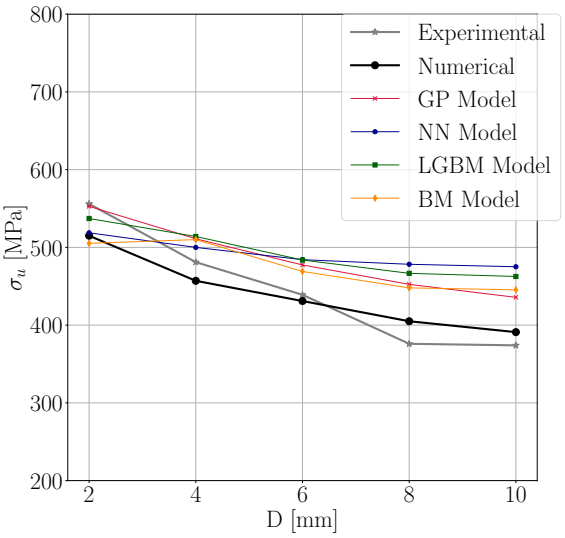


Figure D.4: Comparison of the experimental results from Camanho et al. (2007) with numerical results and indirect predictions from the four stress-strain curve predictors trained on *DATABASE2*, as a function of hole diameter.

Table D.2: Relative error of the indirect predictions from the four stress-strain curve predictors trained on *DATABASE2* relative to numerical and experimental (in parenthesis) results on the specimens from Camanho et al. (2007).

D [mm]	σ_u (Exp.) [MPa]	σ_u Num. [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
2	515	556	7.34 (-0.58)	0.73 (-6.70)	4.30 (-3.39)	-1.90 (-9.13)
4	457	481	11.83 (6.25)	9.41 (3.95)	12.48 (6.86)	6.03 (6.03)
6	431	439	10.77 (8.75)	12.34 (10.29)	12.24 (10.20)	8.83 (6.85)
8	409	376	10.62 (20.33)	16.94 (27.20)	14.09 (24.11)	9.52 (19.13)
10	391	374	6.05 (11.44)	8.80 (21.51)	9.29 (18.31)	0.83 (13.91)

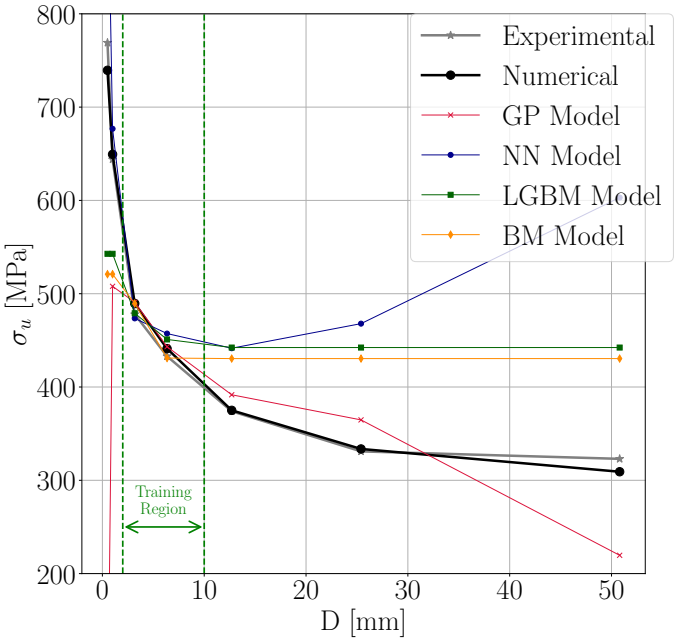
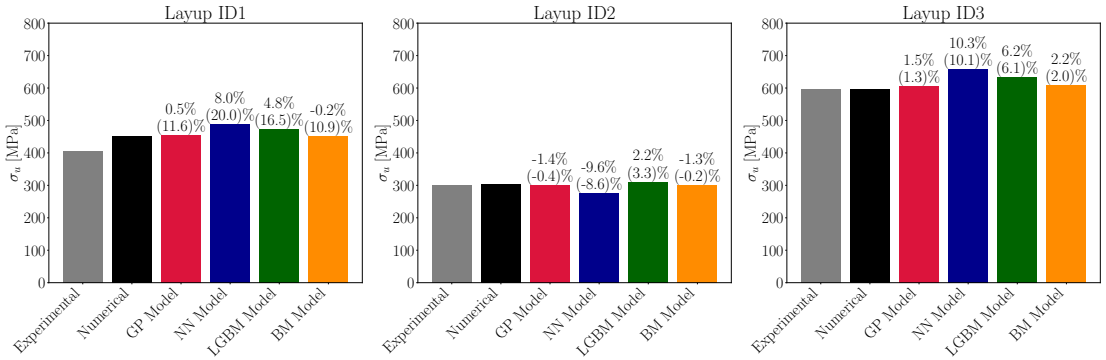


Figure D.5: Comparison of the experimental results from Xu et al. (2016) with numerical results and predictions from the four stress-strain curve predictors trained on *DATABASE2*, as a function of hole diameter.

Table D.3: Relative error of the indirect predictions from the four stress-strain curve predictors trained on *DATABASE2* relative to numerical results on the specimens from Xu et al. (2016).

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
0.50*	739	769	-100.00 (-100.00)	32.40 (27.23)	-26.55 (-29.42)	-29.51 (-32.26)
1.00*	649	644	2.30 (3.16)	20.51 (21.52)	-12.40 (-11.66)	-17.42 (-16.73)
3.18	490	478	-21.74 (-21.13)	4.28 (5.09)	-16.37 (-15.72)	-19.74 (-19.11)
6.35	441	433	0.17 (2.69)	-3.36 (-0.93)	-2.24 (0.22)	-0.09 (2.42)
12.70*	375	374	0.50 (2.36)	3.69 (5.61)	2.26 (4.15)	-2.26 (-0.45)
25.40*	334	331	9.21 (10.20)	40.10 (41.37)	32.43 (33.63)	28.88 (30.05)
50.80*	309	323	-28.93 (-32.01)	95.17 (86.71)	43.14 (36.93)	39.31 (33.27)

*Outside of training range. Training range is D∈[2, 10] mm.



(a) (b) (c)

Figure D.6: Comparison of the numerical and experimental (in parenthesis) results on data from Clarkson (2012) with the indirect strength predictions from the four stress-strain curve predictors trained on *DATABASE2*.

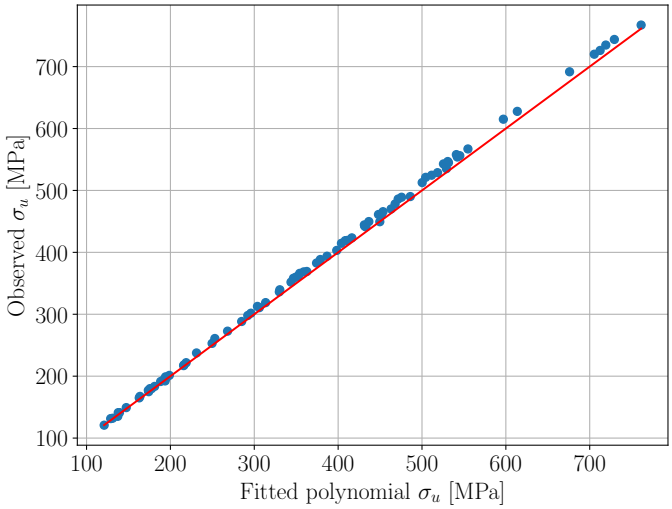


Figure D.7: Outlier detection using MAD method, threshold = 2.5, on the Ultimate Strength feature in *DATABASE2*.

D.2 Surrogate models for an open-hole coupon including material variability

The polynomial curve predictions are now combined with the Failure Deformation prediction, to yield an indirectly predicted Ultimate Strength. Thus, the analysis made with the direct Ultimate Strength regressors can be reproduced.

In Figure D.8, the RMSE between these indirect predictions and the Ultimate Strength observed in the numerical simulations is presented. In Table D.4, the fractions of the test set below thresholds of relative error of 2%. 5%, 10% and 20% are shown.

The same sensitivity analysis was performed. In Figure D.9 the geometric features and in-plane laminate parameters were varied; in Figure D.10 the same was done for Number of Plies and out-of-plane parameters; and in Figure D.11 for the nine material properties.

The same data from literature that was used in Section 6.2.5 was here used for the indirect strength predictors. First, results for the Camanho et al. (2007) experimental campaign are presented in Figure D.12 and Table D.5. The result comparison with data from Xu et al. (2016) is presented as before, in Figure D.13 and Table D.6. Results for the three different stacking sequences from Clarkson (2012) is presented in Figure D.14. For the material T800/8552, data from Erçin et al. (2013), results and errors are shown in Figure D.15 and Tables D.7 and D.8. Results for the six laminates made from another material are presented in Figure D.16.

Regarding these indirect Ultimate Strength predictions, reasonable performance was registered. The GP and LGBM were the best performing models on the test set, with RMSEs of 24.35MPa and 27.13MPa, respectively. In terms of Relative Errors, the GP regressor was able to predict 92.8% of the points in the test set with an error lower than 10%.

When it comes to the sensitivity analysis, these models seem to behave, as expected, worse than the direct strength regressors, although good performance is achieved overall, especially for the GP predictor when varying geometric features and in-plane laminate parameters and the NN regressor when varying material properties.

When dealing with the examples of data coming from literature, the indirect models behaved well, although worse than their direct counterparts.

The models do not possess extrapolation abilities: often, outside the training range, the GP predictor predicted null strength values, and the tree based models, as previously explained, give a constant value in these regions. The NN model also grossly over or underestimates the material's strength, as seen in Table D.6.

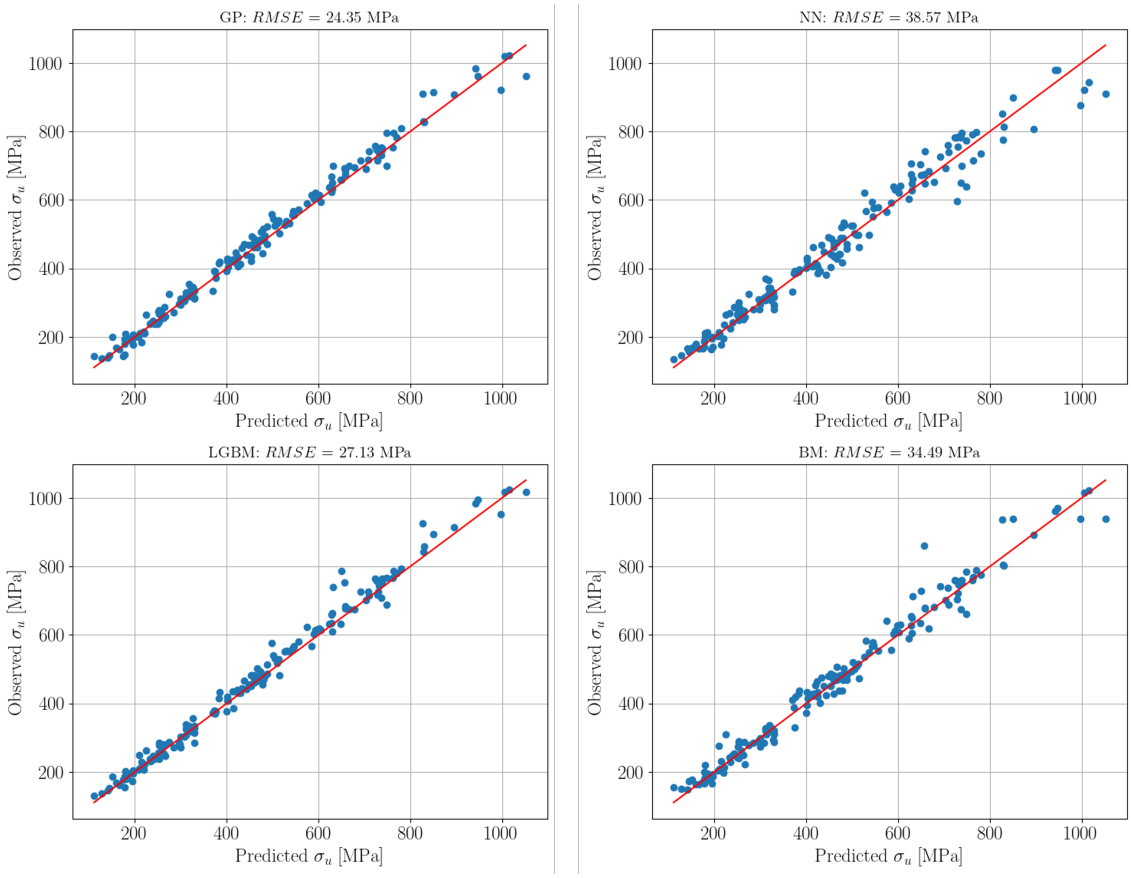


Figure D.8: Observed (from numerical simulations) and indirectly predicted Ultimate Strength for the four ML stress-strain curve predictors trained on *DATABASE3*.

Table D.4: Relative Error (RE) fractions below thresholds of 2%, 5%, 10% and 20% for the indirectly predicted Ultimate Strength by the four ML stress-strain curve predictors trained on *DATABASE3*.

	Fraction of Data below the RE Limit for each Model				
RE Limit	GP Regressor	NN Regressor	LGBM Regressor	BM Regressor	DUMMY Regressor
2%	0.325	0.181	0.337	0.265	0.060
5%	0.681	0.482	0.699	0.602	0.120
10%	0.928	0.765	0.898	0.819	0.199
20%	0.988	0.994	0.988	0.964	0.301

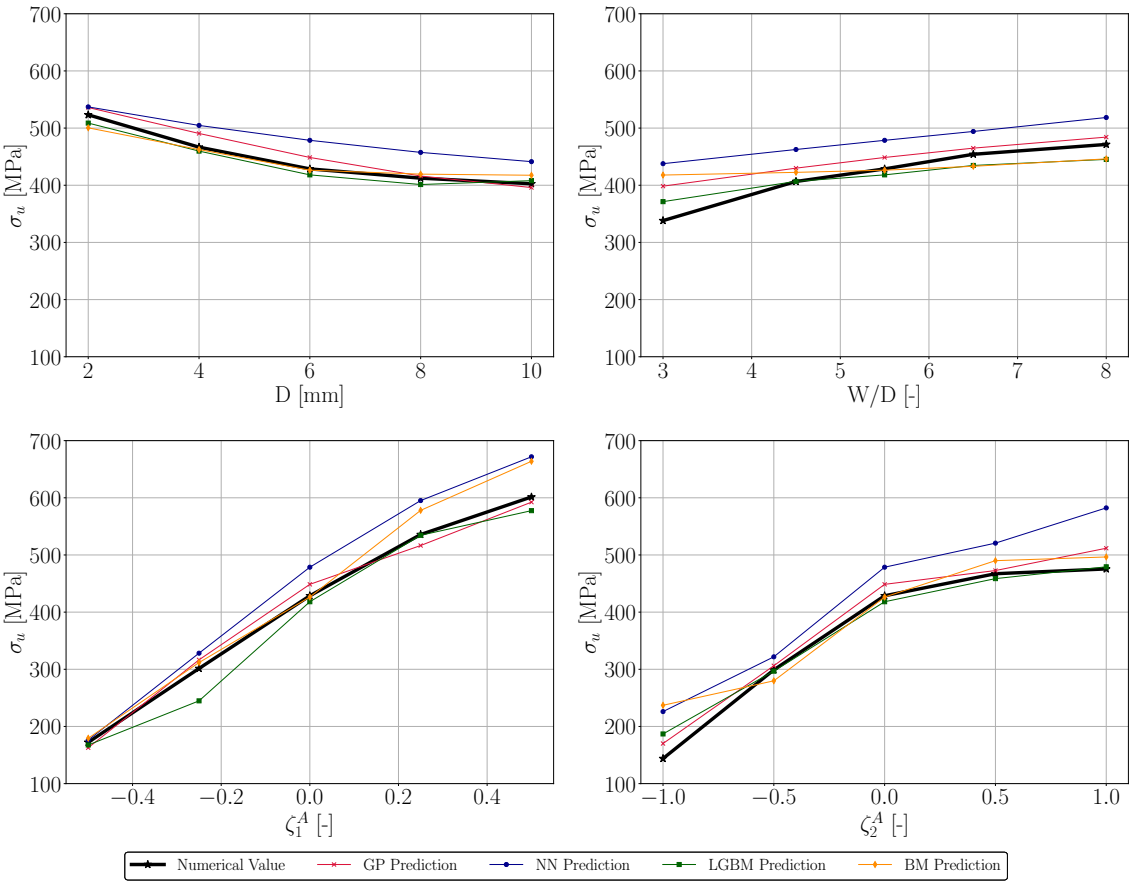


Figure D.9: Results of the sensitivity analysis of the indirectly predicted Ultimate Strength by the stress-strain curve predictors trained on *DATABASE3* for geometric features and in-plane laminate parameters.

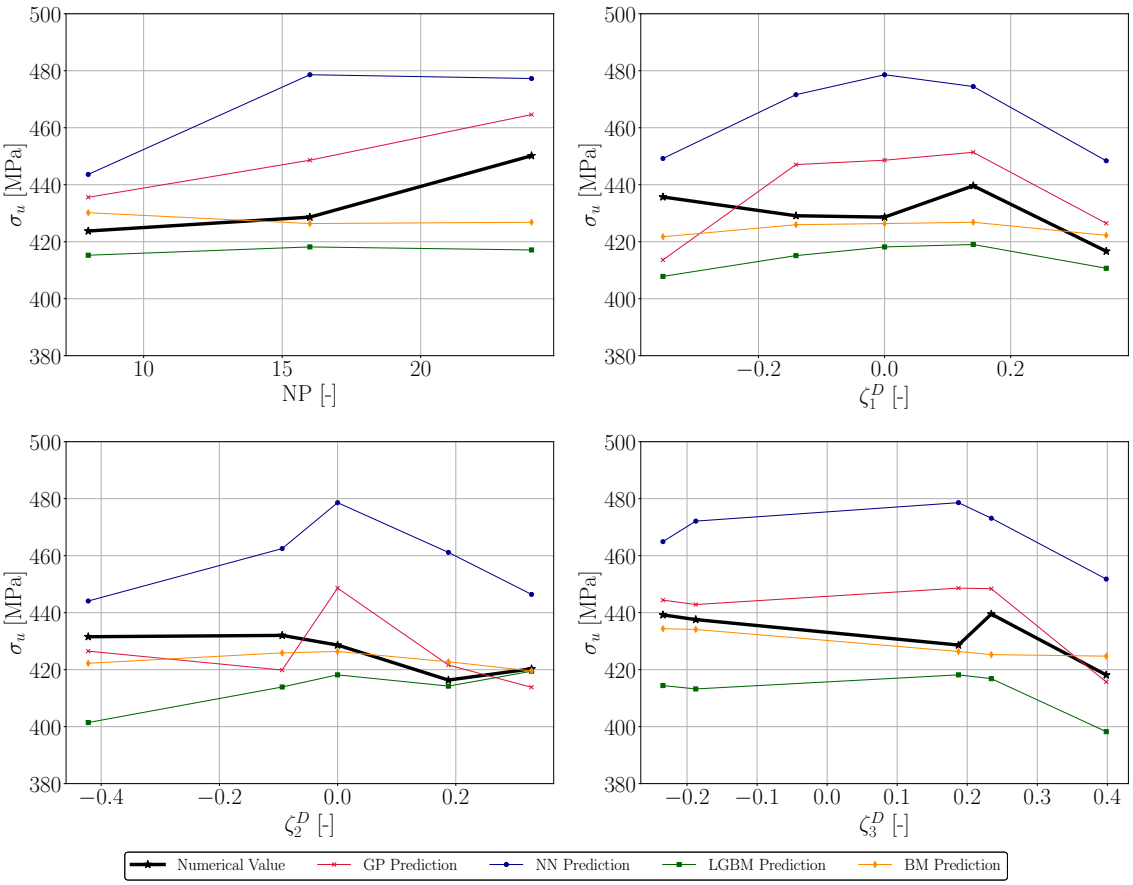


Figure D.10: Results of the sensitivity analysis of the indirectly predicted Ultimate Strength by the stress-strain curve predictors trained on *DATABASE2* for number of plies and out-of-plane laminate parameters.

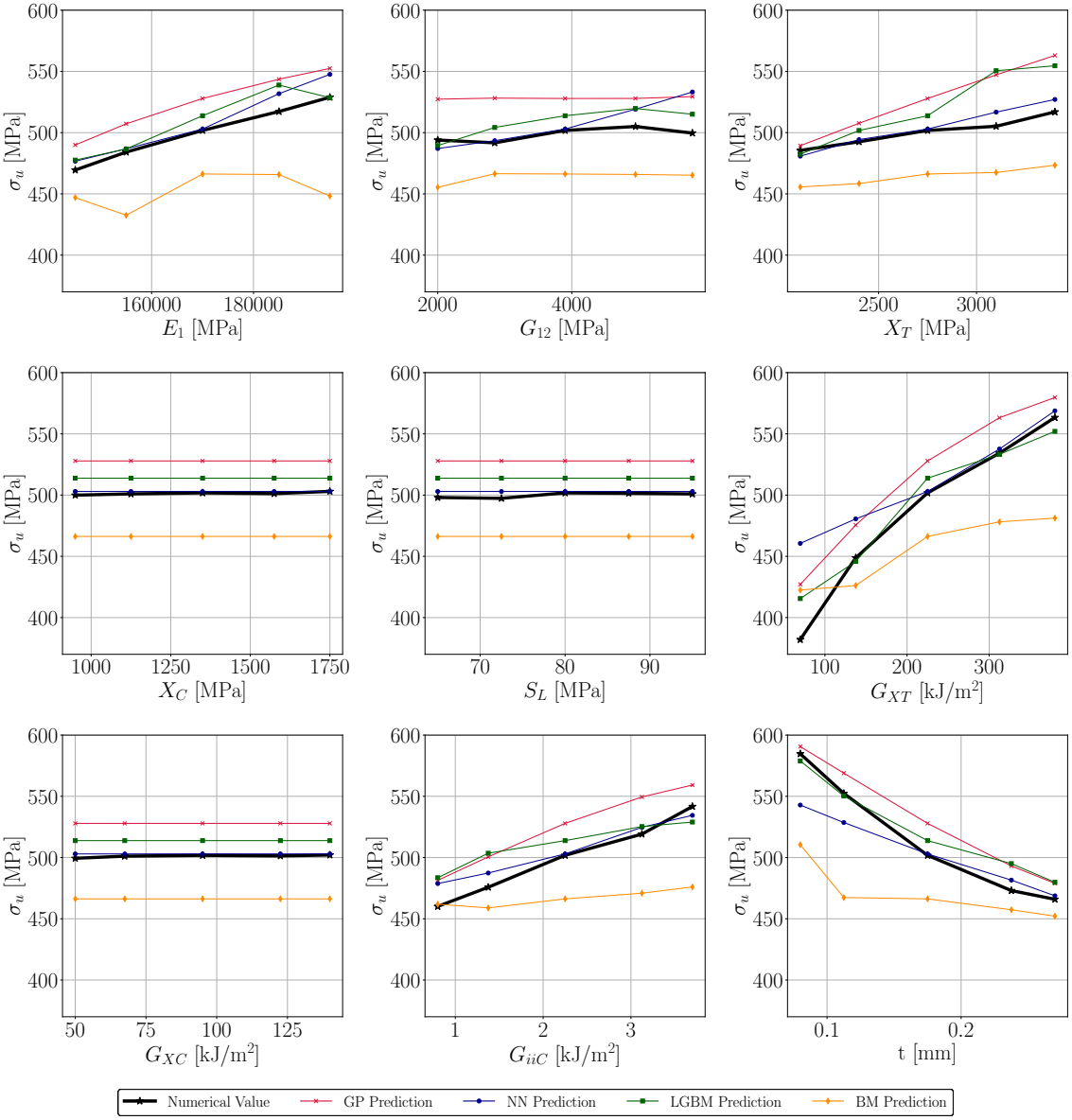


Figure D.11: Results of the sensitivity analysis of the indirectly predicted Ultimate Strength by the stress-strain curve predictors trained on *DATABASE2* for material properties.

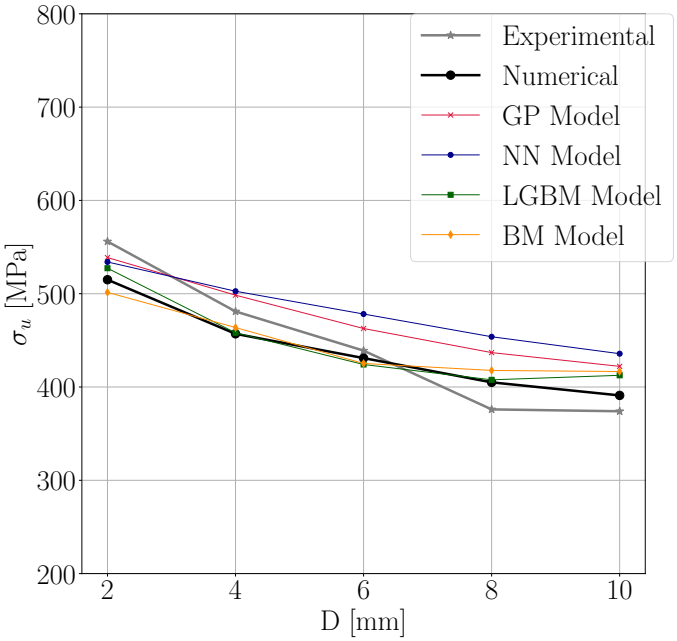


Figure D.12: Comparison of the experimental results from Camanho et al. (2007) with numerical results and indirect predictions from the four stress-strain curve predictors trained on *DATABASE3*, as a function of hole diameter.

Table D.5: Relative error of the indirect predictions from the four stress-strain curve predictors trained on *DATABASE3* relative to numerical and experimental (in parenthesis) results on the specimens from Camanho et al. (2007).

D [mm]	σ_u (Exp.) [MPa]	σ_u Num. [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
2	515	556	4.61 (-3.10)	3.70 (-3.95)	2.39 (-5.16)	-2.63 (-9.81)
4	457	481	9.07 (3.63)	9.99 (4.51)	0.25 (-4.75)	1.48 (-3.59)
6	431	439	7.36 (5.40)	10.96 (8.94)	-1.59 (-3.38)	-1.34 (-3.13)
8	409	376	7.87 (16.19)	12.07 (20.71)	0.67 (8.43)	3.15 (11.11)
10	391	374	7.97 (12.88)	11.44 (16.50)	5.51 (10.30)	6.55 (11.39)

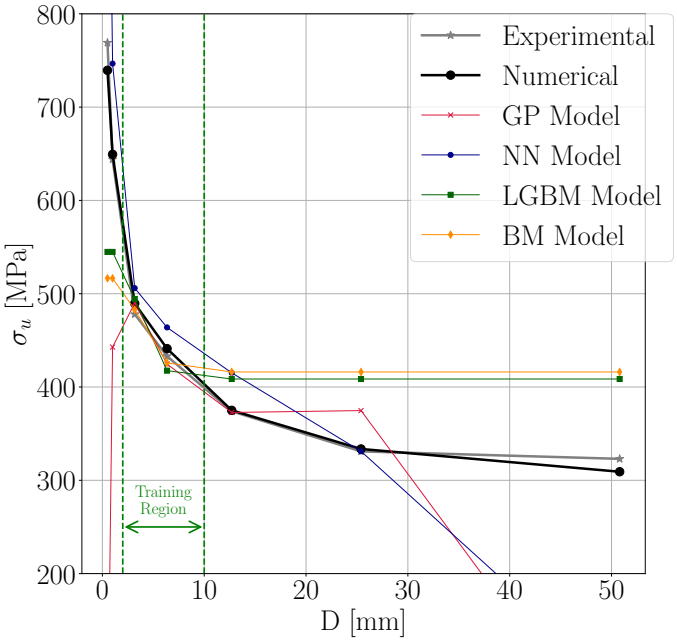
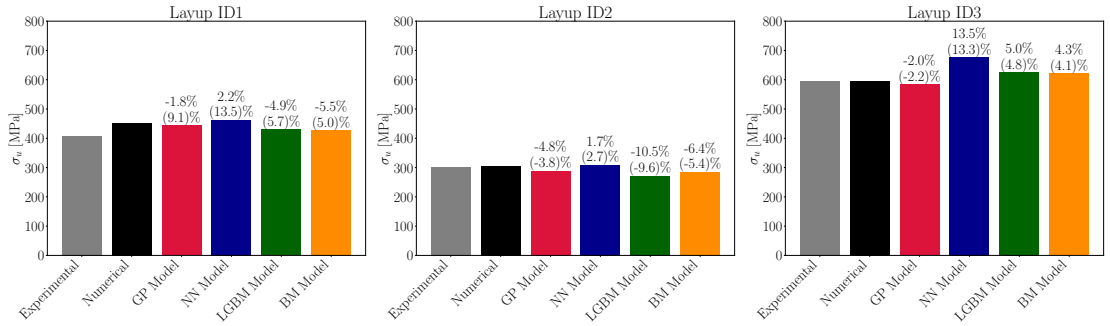


Figure D.13: Comparison of the experimental results from Xu et al. (2016) with numerical results and predictions from the four stress-strain curve predictors trained on *DATABASE3*, as a function of hole diameter.

Table D.6: Relative error of the indirect predictions from the four stress-strain curve predictors trained on *DATABASE3* relative to numerical results on the specimens from Xu et al. (2016).

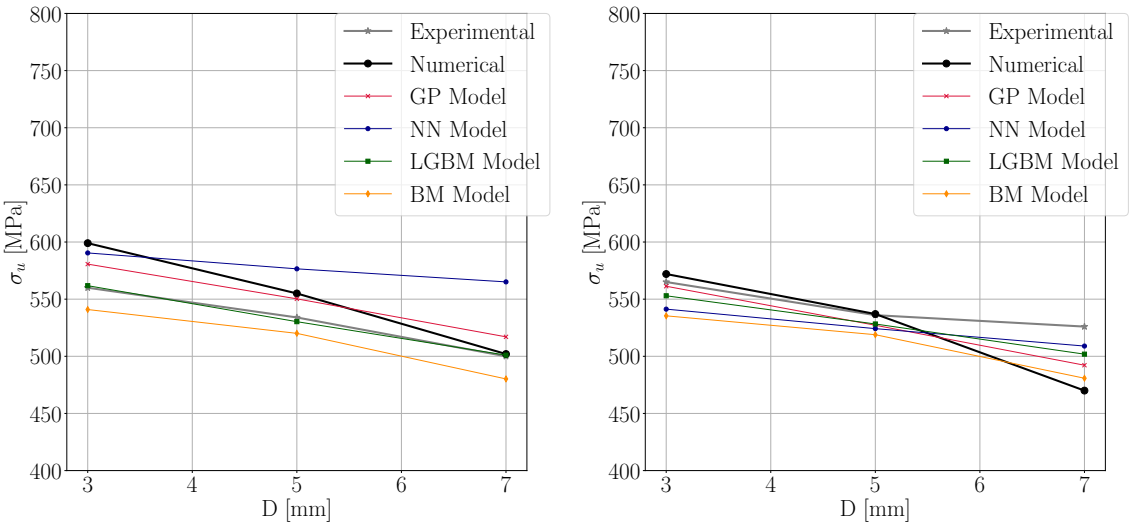
D [mm]	σ_u (Exp.) [MPa]	σ_u Num. [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
0.50†	739	769	-100.00 (-100.00)	56.59 (56.46)	-26.32 (-29.15)	-30.15 (-32.83)
1.00†	649	644	-31.82 (-31.25)	14.98 (15.94)	-16.10 (-15.40)	-20.46 (-19.80)
3.18	490	478	0.12 (2.53)	3.36 (5.86)	0.99 (3.43)	-1.34 (1.04)
6.35	441	433	-3.76 (-1.95)	5.16 (7.13)	-5.35 (-3.57)	-3.45 (-1.64)
12.70†	375	374	-0.61 (-0.35)	10.72 (11.02)	8.96 (9.25)	10.97 (11.27)
25.40†	334	331	12.33 (13.21)	-0.74 (0.04)	22.48 (23.44)	24.74 (25.72)
50.80†	309	323	-100.00 (-100.00)	-74.18 (-75.28)	32.15 (26.50)	34.59 (28.83)

†Outside of training range. Training range is D∈[2, 10] mm.



(a) (b) (c)

Figure D.14: Comparison of the numerical and experimental (in parenthesis) results on data from Clarkson (2012) with the indirect strength predictions from the four stress-strain curve predictors trained on *DATABASE3*.



(a) Laminate 1. (b) Laminate 2.

Figure D.15: Comparison of the experimental results from Erçin et al. (2013) with numerical results and indirect strength predictions from the four stress-strain curve predictors trained on *DATABASE3*, as a function of hole diameter, for both Laminates.

Table D.7: Relative error of indirect strength predictions from the four stress-strain curve predictors (trained on *DATABASE3)* relative to numerical results on the specimens from Erçin et al. (2013), for Laminate 1.

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
3	599	560	-3.06 (3.69)	-1.43 (5.44)	-6.21 (0.32)	-9.70 (-3.41)
5	555	534	-0.85 (3.05)	3.88 (7.97)	-4.45 (-0.69)	-6.29 (-2.61)
7	502	500	3.00 (3.41)	12.57 (13.02)	-0.19 (0.21)	-4.43 (-3.96)

Table D.8: Relative error of indirect strength predictions from the four stress-strain curve predictors (trained on *DATABASE3)* relative to numerical results on the specimens from Erçin et al. (2013), for Laminate 2.

D [mm]	σ_u Num. [MPa]	σ_u (Exp.) [MPa]	RE GP Regressor [%]	RE NN Regressor [%]	RE LGBM Regressor [%]	RE BM Regressor [%]
3	572	565	-1.87 (-0.65)	-5.37 (-4.20)	-3.33 (-2.13)	-6.40 (-5.24)
5	537	536	-1.83 (-1.65)	-2.38 (-2.20)	-1.62 (-1.44)	-3.37 (-3.19)
7	470	526	4.72 (-6.43)	8.29 (-3.24)	6.78 (-4.59)	2.30 (-8.59)

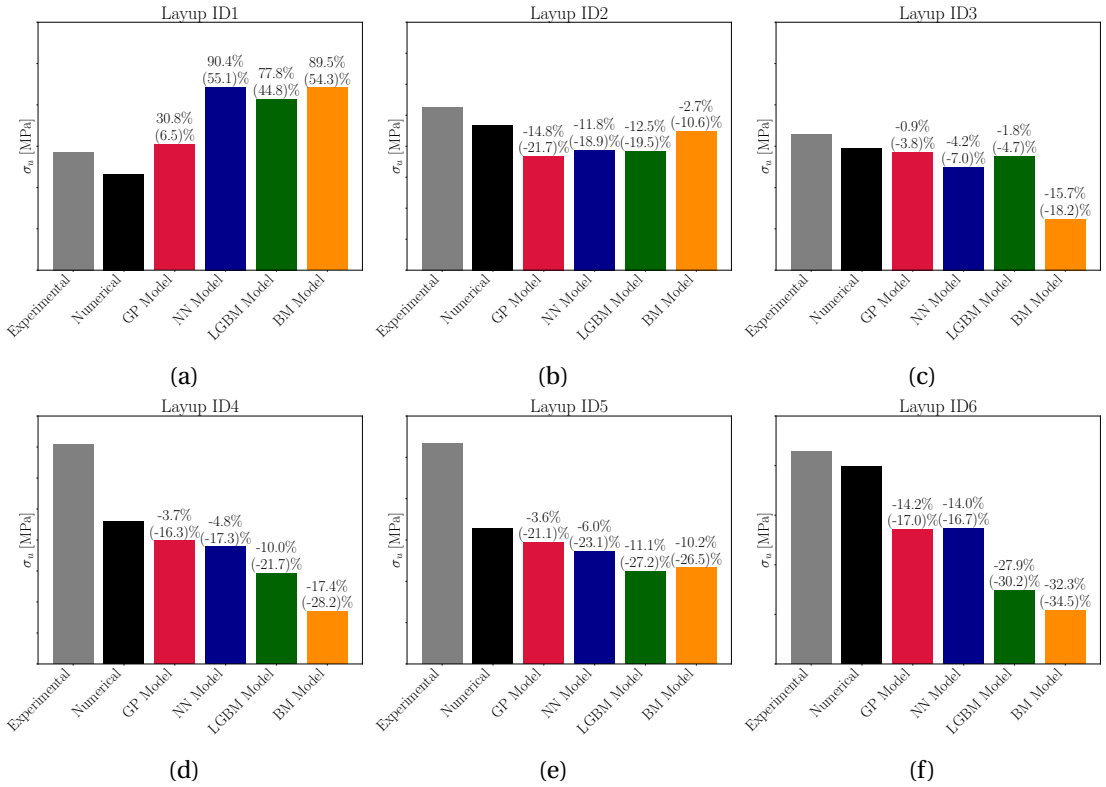


Figure D.16: Comparison of the experimental results with a different material with numerical results and indirect strength predictions from the four stress-strain curve predictors trained on *DATABASE3*, for six different layups. Relative Error presented in relation to numerical and experimental (in parenthesis) results.

Bibliography

Nadia Moneem Al-Abdaly, Salwa R Al-Taai, Hamza Imran, and Majed Ibrahim. Development of prediction model of steel fiber-reinforced concrete compressive strength using random forest algorithm combined with hyperparameter tuning and k-fold cross-validation. *Eastern-European Journal of Enterprise Technologies*, 5(7):113, 2021.

Peshawa Jamal Muhammad Ali, Rezhna Hassan Faraj, and Erbil Koya. Data normalization and standardization: a technical report. *Mach Learn Tech Rep*, 1(1):1–6, 2014.

Reza Alizadeh, Janet K Allen, and Farrokh Mistree. Managing computational complexity using surrogate models: a critical review. *Research in Engineering Design*, 31(3):275–298, 2020.

Saleh Alkhalifa. 7.2.2 measuring success with regressors, 2022. URL `https://app.knovel.com/hotlink/khtml/id:kt0134L341/machine-learning-in-biotechnology/measuring--success-with`.

Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993.

Albertino Arteiro, Naresh Sharma, Jose Daniel D Melo, Sung Kyu Ha, Antonio Miravete, Yasushi Miyano, Thierry Massard, Pranav D Shah, Surajit Roy, Robert Rainsberger, et al. A case for tsai's modulus, an invariant-based approach to stiffness. *Composite Structures*, 252:112683, 2020.

ASTM. ASTM D5766/D5766M-11 standard test method for open-hole tensile strength of polymer matrix composite laminates, 2018.

Mohammed Awad and Salam Fraihat. Recursive feature elimination with cross-validation with decision tree: Feature selection method for machine learning-based intrusion detection systems. *Journal of Sensor and Actuator Networks*, 12(5):67, 2023.

Priya C Bala. Regularization in neural networks, 2023. URL `https://www.pinecone.io/learn/regularization-in-neural-networks/`. Accessed: 2024-06-15.

Dave Bergmann. What is semi-supervised learning? IBM, December 2023. URL `https://www.ibm.com/topics/semi-supervised-learning`. Accessed: 2024-05-27.

Anup Bhande. Techniques for handling underfitting and overfitting in machine learning, 2021. URL `https://towardsdatascience.com/techniques-for-handling-underfitting-and-overfitting-in-machine-learning-348daa2380b9`. Accessed: 2024-06-15.

Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25:197–227, 2016.

Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larranaga. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5 (5):216–233, 2015.

L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, 1st edition, 1984. doi: 10.1201/9781315139470.

Leo Breiman. Bagging predictors. *Machine learning*, 24:123–140, 1996.

Jason Brownlee. How to choose an activation function for deep learning, 2024. URL `https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/`. Accessed: 2024-06-15.

Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79, 2018.

Pedro Ponces Camanho, Pere Maimí, and CG Dávila. Prediction of size effects in notched laminates using continuum damage mechanics. *Composites science and technology*, 67(13):2715–2727, 2007.

Elizabeth Clarkson. Hexcel 8552 im7 unidirectional prepreg 190 gsm & 35% rc qualification statistical analysis report. *National Institute for Aviation Research*, 2012.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of machine learning research*, 2(Dec):265–292, 2001.

Roberta Cumbo, Antonio Baroni, Alfredo Ricciardi, and Stefano Corvaglia. Design allowables of composite laminates: A review. *Journal of Composite Materials*, 56(23):3617–3634, 2022.

Lucas F Martins Da Silva, Fernando J Lino Alves, and António T Marques. Materiais de construção. *Publindústria, edições Técnicas, Porto, Portugal*, 2013.

Isaac M Daniel, Ori Ishai, Issac M Daniel, and Ishai Daniel. *Engineering mechanics of composite materials*, volume 3. Oxford university press New York, 1994.

Julianna Delua. Supervised versus unsupervised learning: What's the difference?, March 2021. URL `https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning`. Accessed: 2024-05-27.

Washington Department of Defense. Composite materials handbook. volume 1. polymer matrix composites guidelines for characterization of structural materials, 2002.

Pradip Dhal and Chandrashekhar Azad. A comprehensive survey on feature selection in the various fields of machine learning. *Applied Intelligence*, 52(4):4543–4581, 2022.

David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Toronto, 2014.

EASA. CS-25: Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes - Amendment 27. Technical report, European Union Aviation Safety Agency, 2023.

GH Erçin, PP Camanho, J Xavier, G Catalanotti, S Mahdi, and P Linde. Size effects on the tensile and compressive failure of notched composite laminates. *Composite Structures*, 96:736–744, 2013.

João Manuel Pires Esteves. Sensitivity analysis towards the virtual generation of design allowables for composite materials through machine learning models, June 2023. Bachelor's final project report.

Felipe Farias, Teresa Ludermir, and Carmelo Bastos-Filho. Similarity based stratified splitting: an approach to train better classifiers. *arXiv preprint arXiv:2010.06099*, 2020.

Matthias Feurer and Frank Hutter. Hyperparameter optimization. *Automated machine learning: Methods, systems, challenges*, pages 3–33, 2019.

Alexander Forrester, Andras Sobester, and Andy Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

C Furtado, G Catalanotti, A Arteiro, PJ Gray, BL Wardle, and PP Camanho. Simulation of failure in laminated polymer composites: Building-block validation. *Composite Structures*, 226: 111168, 2019.

Carolina Furtado, LF Pereira, Rodrigo Paiva Tavares, M Salgado, F Otero, Guiuseppe Catalanotti, Albertino Arteiro, Miguel A Bessa, and Pedro P Camanho. A methodology to generate design allowables of composite laminates using machine learning. *International Journal of Solids and Structures*, 233:111095, 2021.

JL Grenestedt and Peter Gudmundson. Layup optimization of composite material structures. *Optimal design with advanced materials*, pages 311–336, 1993.

Hazal Gültekin. Classification vs regression. Medium, December 2023. URL `https://medium.com/@hazallgultekin/classification-vs-regression-e15c619604f7`. Accessed: 2024-05-27.

Isabelle Guyon and André Elisseeff. An introduction to feature extraction. In *Feature extraction: foundations and applications*, pages 1–25. Springer, 2006.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL `https://doi.org/10.1038/s41586-020-2649-2`.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Jon Herman and Will Usher. Salib: An open-source python library for sensitivity analysis. *Journal of Open Source Software*, 2(9):97, 2017.

Won-Kee Hong. *Artificial Intelligence-Based Design of Reinforced Concrete Structures - Artificial Neural Networks for Engineering Applications*. Elsevier, 2023. ISBN 978-0-443-15252-8. URL https://app.knovel.com/hotlink/toc/id:kpAIBDRCS2/artificial-intelligence/artificial-intelligence.

IBM. What are convolutional neural networks?, 2024a. URL https://www.ibm.com/topics/convolutional-neural-networks. Accessed: 2024-06-15.

IBM. What are recurrent neural networks?, 2024b. URL https://www.ibm.com/topics/recurrent-neural-networks. Accessed: 2024-06-15.

Samuel T IJsselmuiden, Mostafa M Abdalla, Omprakash Seresta, and Zafer Gürdal. Multi-step blended stacking sequence design of panel assemblies with buckling constraints. *Composites part b: engineering*, 40(4):329–336, 2009.

Francois-Xavier Irisarri, Mostafa M Abdalla, and Zafer Gürdal. Improved shepard's method for the optimization of composite structures. *AIAA journal*, 49(12):2726–2736, 2011.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*, pages 240–248. PMLR, 2016.

P. Jiang, Q. Zhou, and X. Shao. *Surrogate Model-Based Engineering Design and Optimization*. Springer Tracts in Mechanical Engineering. Springer Nature Singapore, 2019. ISBN 9789811507311. URL https://books.google.pt/books?id=3ZO7DwAAQBAJ.

Aarthi Kasirajan. Underfitting and overfitting in machine learning, June 8 2020. URL https://www.analyticsvidhya.com/blog/2020/02/underfitting-overfitting-best-fitting-machine-learning/.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Keras. Layer activation functions, 2024. URL https://keras.io/api/layers/activations/. Accessed: 2024-06-15.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Sotiris B Kotsiantis. Decision trees: a recent overview. *Artificial Intelligence Review*, 39:261–283, 2013.

Damjan Krstajic, Ljubomir J Buturovic, David E Leahy, and Simon Thomas. Cross-validation pitfalls when selecting and assessing regression and classification models. *Journal of cheminformatics*, 6:1–15, 2014.

Thomas Navin Lal, Olivier Chapelle, Jason Weston, and André Elisseeff. Embedded methods. In *Feature Extraction*, 2006. URL https://api.semanticscholar.org/CorpusID:1421150.

Niklas Lavesson and Paul Davidsson. Quantifying the impact of learning algorithm parameter tuning. In *AAAI*, volume 6, pages 395–400, 2006.

Christophe Leys, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, 49(4):764–766, 2013. ISSN 0022-1031. doi: https://doi.org/10.1016/j.jesp.2013.03.013.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5:1–16, 2016.

Robert J May, Holger R Maier, and Graeme C Dandy. Data splitting for artificial neural networks using som-based stratified sampling. *Neural Networks*, 23(2):283–294, 2010.

Maximillian Merrillees and Lan Du. Stratified sampling for extreme multi-label data. In Kamal Karlapalem, Hong Cheng, Naren Ramakrishnan, R. K. Agrawal, P. Krishna Reddy, Jaideep Srivastava, and Tanmoy Chakraborty, editors, *Advances in Knowledge Discovery and Data Mining*, pages 334–345, Cham, 2021. Springer International Publishing. ISBN 978-3-030-75765-6.

Jeff Miller. Reaction time analysis with outlier exclusion: Bias varies with sample size. *The Quarterly Journal of Experimental Psychology Section A*, 43(4):907–912, 1991.

Christoph Molnar. *Interpretable Machine Learning: A Guide For Making Black Box Models Explainable*. Lulu Press, Inc., 2022.

Adrian P Mouritz. *Introduction to aerospace materials*. Elsevier, 2012.

Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013.

John M Orr, Paul R Sackett, and Cathy LZ Dubois. Outlier detection and treatment in i/o psychology: A survey of researcher beliefs and an empirical illustration. *Personnel Psychology*, 44(3):473–486, 1991.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

Sashikanta Prusty, Srikanta Patnaik, and Sujit Kumar Dash. Skcv: Stratified k-fold cross-validation on ml classifiers for predicting cervical cancer. *Frontiers in Nanotechnology*, 4: 972421, 2022.

Christopher A. Ramezan, Timothy A. Warner, and Aaron E. Maxwell. Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. *Remote Sensing*, 11(2), 2019. ISSN 2072-4292. doi: 10.3390/rs11020185. URL `https://www.mdpi.com/2072-4292/11/2/185`.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Machine learning*, 85:333–359, 2011.

Jean Rouchon. Certification of large airplane composite structures. In *ICAS Congress Proceedings*, volume 2, pages 1439–1447, 1990.

Peter J Rousseeuw and Christophe Croux. Alternatives to the median absolute deviation. *Journal of the American Statistical association*, 88(424):1273–1283, 1993.

Andrea Saltelli, Paola Annoni, Ivano Azzini, Francesca Campolongo, Marco Ratto, and Stefano Tarantola. Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index. *Computer physics communications*, 181(2):259–270, 2010.

Aravind Sasikumar, Albert Turon, Ivan R Cózar, Oriol Vallmajó, Jorge Camacho Casero, Matthias De Lozzo, and Said Abdel-Monsef. Sensitivity analysis methodology to identify the critical material properties that affect the open hole strength of composites. *Journal of Composite Materials*, 57(10):1791–1805, 2023.

Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

Shahriar Setoodeh, Mostafa Abdalla, and Zafer Gurdal. Approximate feasible regions for lamination parameters. In *11th AIAA/ISSMO multidisciplinary analysis and optimization conference*, page 6973, 2006.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Dalwinder Singh and Birmohan Singh. Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, 97:105524, 2020.

Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.

Lily-belle Sweet, Christoph Müller, Mohit Anand, and Jakob Zscheischler. Cross-validation strategy impacts the performance and interpretation of machine learning models. *Artificial Intelligence for the Earth Systems*, 2(4), 2023.

Stephen W Tsai and Nicholas J Pagano. *Invariant properties of composite materials*. Technomic Publishing Company, 1968.

O Vallmajó, IR Cózar, C Furtado, R Tavares, A Arteiro, A Turon, and PP Camanho. Virtual calculation of the b-value allowables of notched composite laminates. *Composite Structures*, 212: 11–21, 2019.

B Venkatesh and J Anuradha. A review of feature selection and its methods. *Cybernetics and information technologies*, 19(1):3–26, 2019.

Jorge R Vergara and Pablo A Estévez. A review of feature selection methods based on mutual information. *Neural computing and applications*, 24:175–186, 2014.

Andreas François Vermeulen. *Industrial Machine Learning - Using Artificial Intelligence as a Transformational Disruptor*. Apress, an imprint of Springer Nature, 2020. ISBN 978-1-4842-5315-1. URL `https://app.knovel.com/hotlink/toc/id:kpIMLUAIT1/industrial-machine-learning/industrial-machine-learning`.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Junya Wang, Pengcheng Xu, Xiaobo Ji, Minjie Li, and Wencong Lu. Feature selection in machine learning for perovskite materials design and discovery. *Materials*, 16(8):3134, 2023.

Hilde JP Weerts, Andreas C Mueller, and Joaquin Vanschoren. Importance of tuning hyperparameters of machine learning algorithms. *arXiv preprint*, 2020.

Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Xiaodong Xu, Michael R Wisnom, Kuan Chang, and Stephen R Hallett. Unification of strength scaling between unidirectional, quasi-isotropic, and notched carbon/epoxy laminates. *Composites Part A: Applied Science and Manufacturing*, 90:296–305, 2016.

Ke Yan and David Zhang. Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B: Chemical*, 212:353–363, 2015. ISSN 0925-4005. doi: https://doi.org/10.1016/j.snb.2015.02.025.

Jiawei Yang, Susanto Rahardja, and Pasi Fränti. Outlier detection: how to threshold outlier scores? In *Proceedings of the international conference on artificial intelligence, information processing and cloud computing*, pages 1–6, 2019.

Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.