Hochschule
für Technik
Stuttgart

Utilizing Supervised Learning for the Prediction of Bending Parameters

some hyped-up tagline

Philipp Kurrle

Supervised by Prof. Dr. Ulrike Pado

Co-supervised by M. Sc. Peter Lange

Computer Science

Master's Programme in Software Technology

Hochschule für Technik

February, 2023

A dissertation submitted in partial fulfilment of the requirements for the degree of M.Sc. in Software Technology.

Abstract

This is the abstract. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Contents

List of Figures

List of Tables

List of Abbreviations

ML Machine Learning . . . 6

SVM Support Vector Machine . . . 9

SVR Support Vector Regression . . . viii

ANN Artificial Neural Network . . . 10

FEM Finite Element Method . . . 10

DSR Design Science Research . . . 19

DP Design Principles . . . 19

GQM Goal-Question-Metric . . . viii

RF Random forest . . . viii

DT Decision tree . . . viii

RMSE Root Mean Squared Error . . . 42

MSE mean squared error . . . 42

MAE Mean Absolute Error . . . 42

LOOCV Leave-one-out Cross Validation . . . 50

CV Cross-validation . . . 45

LPOCV Leave-P-Out-Cross-Validation . . . 48

LR Linear Regression . . . 30

DT Decision Tree . . . viii

ET Extra Trees . . . 36

MLP Multi-layer Perceptron . . . viii

GBT Gradient boosted trees . . . 34

1

Introduction

Sheet metal forming has been used for centuries in different manufacturing industries to create a wide range of products for different applications. Sheet metal bending and stamping can be considered as the most important variants in the forming industry. (Cruz et al., 2021, p. 1) Therefore these have been continuously improved in recent decades to meet the growing demand especially in automotive and aircraft industries with the goal to reduce energy efficiency and emissions. (Zheng et al., p. 4)

Sprinback is a common phenomenon in sheet metal forming processes. It is a deformation of the sheet metal that occurs when the sheet metal is bent. Therefore, predicting the spring back is important to reduce the number of trial and error cycles in the manufacturing process. (Cruz et al., 2021, p. 1) Sheet metal forming is a complex process that involves a large number of variables and parameters Therefore, it is difficult to predict the spring back accurately, which makes it an interesting case for machine learning.

In order to predict springback with minimium errors, this thesis build and evaluates different machine learning models to predict the springback of a sheet metal. The models are evaluated based on the mean absolute error (MAE) and the root mean squared error (RMSE). The best model is then used to predict the springback of a sheet metal with different parameters.

2

Theoretical Foundations

2.1 | Sheet Metal

Sheet metal is a type of metal usually manufacturey by flat rolling and characterized by its high length to thickness ratio, typically ranging from 0.4 mm to 6mm. Above this range, its is considered to be a plate and blow it, it is considered to be foil (Groover, p. 405). Low-carbon steel is the most commonly used type of sheet metal. Its low cost and has a good form ability as well as its sufficient strength for most applications. (Groover, p. 405) Sheet metals play a vital role in many industries. A significant number of consumer and industrial products such as automobiles contain sheet metal parts. The majority of sheet metal forming operations are conducted using presses and are known as press working. These presses employ tooling such as punch and die or stamping die. The sheet metal forming processes can be classified into three main categories: cutting, bending, and drawing, with bending being the most common and only relevant process for this thesis. Bending and drawing are employed to shape sheet metal parts into their required forms (Groover, p. 405).

2.1.1 | Bending

Bending is a forming operation that is used to change the shape of a sheet metal by apply a load to it. The load is applied in way that exceeds the yield strength of the metal, but is below its ultimate tensile strength, which allows the metal to bet permanently deformed into a new shape (Baig et al., 2021, p. 1).

During the bending process, the metal on the outside of the neural plane (the plane that is perpendicular to the bending axis) is stretched, while the metal, while the metal on the inside is compressed.

This results in a curvature of the sheet metal in the direction of the applied load. (Baig et al., 2021, p. 3) The amount of curvature that is achieved in the bending process is determined by the amount of load applied, the thickness and properties of the metal, and the location and length of the neutral plane. By controlling these factors, it is possible to achieve precise and consistent results in the bending of sheet metal.

Figure 2.1 shows the neutral plane after the bending operation, it is visible, that is is closer to the inside of the bend than to the outside of the bend. The arrows show where the metal was stretched and where it was compressed.

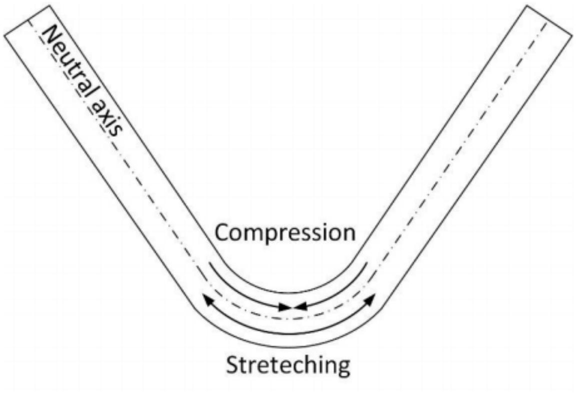


Figure 2.1: Bending plane, compression and stretching of sheet metal (Baig et al., 2021, p. 3)

2.2 | Sheet Metal Bending

The process of sheet metal bending involves using force to shape sheet metal into a desired form. It is usually used to produce large quantities of components at a low cost in various industries (Dib et al., p. 1).

2.2.0.1 | Air Bending

Air bending is a variant in the V-Bending process which is performed using puch-and-die tooling. (Groover, p. 416) As illustrated in Figure 2.2 the punch pushed the sheet metal which is placed in the die down. In other bending forms the die is also used to shape or form the metal in a specific way. This is where the term "die" comes from in this context. (Source missing) In other bending methods the die is fully closed and provides a form where the sheet metal is pressed into. In air bending this is not the case, but the term stays the same. (Source missing)

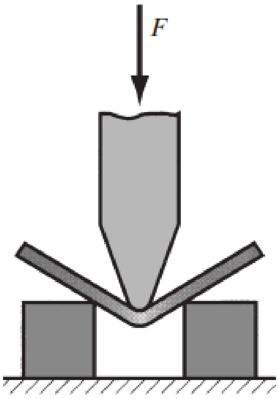


Figure 2.2: Air bending (Groover, p. 416)

Air bending is commonly used in automotive industry to manufacture sheet metal parts. (Kim et al., p. 342) In this process the punch sheet metal comes in contact of the outside edges of the die , as well as the punch tip, but it does not come in contact with the die surface. It is typically the preferred bending method because, its high flexibility because it is possible to achieve different bending angles using the same punch-and-die tooling. (Miranda et al., p. 3)(Cruz et al., 2021, p. 1)

Today sheet metal bending machines like air bending machines are usually equipped with "copmputer numeral control" (CNC) systems that can automatically control the bending process and produce the desired shape." (Miranda et al., p. 3) The air bending process is shows strong nonlinear behavior, considering its parameters and their interrelationships. (Miranda et al., p. 3)

2.2.0.2 | Spring Back

When the punch and therefore the bending pressure is removed at the end of the deformation operation, elastic energy remains in the bent part. This elastic energy is released, and the metal sheet partially returns to its original shape. (Groover, p. 113-114) In metal forming this process is called spring back and is illustrated in Figure 2.3. The solid line shows the metal plate in its original for when the punch was still applied. The dashed line shows the metal plate after the punch was removed. The metal plate is partially returned to its original shape. The angle before the spring back is usually denoted as α_f and the angle after spring back as α_i. The spring back ($\Delta\alpha_{SB}$) is therefore the difference between α_f and α_i as show in equation 2.1. (Cruz et al., 2021, p. 6)

$$\delta\alpha_{SB} = \alpha_i - \alpha_f \tag{2.1}$$

To address this issue there are several methods to compensate the spring back. For example one common method is over bending, which means that the punch angle and radius are fabricated smaller than the specified angle. (Groover, p. 114) Prerequisite for all compensation methods is that the spring back is known therefore the accurate prediction of the spring back play an important role in the manufacturing process.

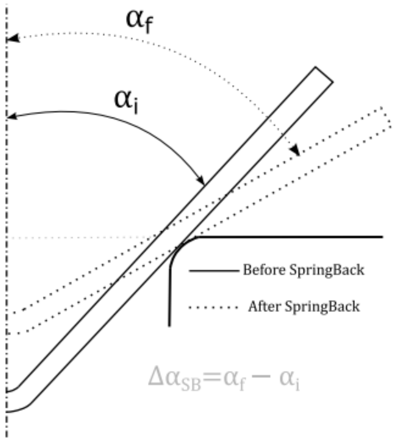


Figure 2.3: Spring back (Cruz et al., 2021, p. 5)

2.3 | Machine Learning

Machine learning also called Machine Learning (ML) is a field of study that involves using statistical and computational techniques to analyze and learn from data. (Müller and Guido, p. 1)

2.3.1 | Supervised Learning

Supervised learning is a type of machine learning where we have a set of input/output pairs that we use to train a model to predict an outcome for a given input. We use this model to make predictions on new, unseen data, the training set. It consists of the input/output pairs, needs to be created manually, but once the model is trained, it can automate and potentially improve upon tasks that would be time-consuming or difficult for humans to perform. (Müller and Guido, p. 25)

2.3.2 | Regression

There are two main types of supervised machine learning problems: classification and regression. For regression problems it is the goal to predict a continuous numerical value. This can be a real number in mathematical terms or a floating-point number in programming terms. (Müller and Guido, p. 226) Classification problems involve predicting a class label, which is a choice from a predefined list of possible options. (Müller and Guido, p. 25)

Predicting the spring back is a regression problem because the spring back is a continuous-valued output. Therefore, this work focuses on supervised learning for a regression problem and does not further consider classification problems.

2.3.3 | Overfitting and Underfitting

In supervised learning, the goal is to construct a model using training data that can accurately predict outcomes for new, unseen data that shares similar characteristics as the training data. The ability of a model to make accurate predictions on unseen data is referred to as generalization. The objective is to develop a model that generalizes as effectively as possible. (Müller and Guido, p. 35)

The effectiveness of an algorithm on new data is determined by its performance on a test set. Simple models tend to generalize better to new data. Overfitting occurs when a model is too complex for the amount of information available and does not generalize well, while underfitting occurs when a model is too simple and does not perform well on the training set. The goal is to find the simplest model that captures the variability in the data. (Müller and Guido, p. 35)

2.3.4 | Bias-Variance Tradeoff

Bias measures how well the central tendency of a learner's model approximates the actual function it is trying to learn. If the model consistently learns the true function accurately, it is unbiased. Otherwise, it is biased. (Neal, p. 7-8) In essence the bias is the differences between the predicted values by the model and the actual values. A high bias indiactes a model with a complex fit to the training data and therefore overfits (Neal, p. 20)

Variance measure how much the model's prediction vary, when trained on different subsets of the data. That means that a model with low variance generalizes one new data. A high variance indicates that the model as a complex fit to the data and therefore is overfitting the training data. This means it will not perform well on new data. (Neal, p. 7-8)

Both, high variance and high bias are undesirable properties for a model. The goal is, to find a model with low bias and low variance. This is called the bias-variance trade off. (Neal, p. 9)

Regularization

Regularization refers to a group of techniques that force a model to be simpler. this usually results in a slight increase in bias but a substantial decrease in variance (Burkov, 2019, p. 12).

The regularization method used in this thesis and in general the most popular are L1 and L2 regularization. In order to construct a regularized model, the objective function is altered by including a penalizing term that increases the value as the model becomes more complex (Burkov, 2019, p. 12)

2.3.5 | Ensemble Learning

Ensemble techniques in machine learning to involve the construction of multiple models , called "learners," for a given task. The primary goal of these methods is to improve the accuracy and performance of the model by combining the predictions of multiple learners. Ensemble techniques differ from single classifier methods by constructing multiple models and combining them using a voting strategy in order to highlight different aspects of the data. This can potentially lead to improved overall performance. (Shaik and Srinivasan, p. 253)

2.3.6 | Evaluations and Improvements of Models

2.3.6.1 | Cross Validation

Cross-validation is a method of evaluating the performance of a model by training multiple models on different subsets of the data and evaluating their performance. The most common version of cross-validation is k-fold cross-validation, in which the data is divided into k folds , and k models are trained, each using a different fold as the test set and the remaining folds as the training set. The accuracy of each model is then evaluated, and the average accuracy across all k models is used as an estimate of the generalization performance of the model. This process helps to reduce the variability in model performance due to the specific choice of training and test sets, and is therefore considered to be a more stable and reliable way to evaluate the performance of a model. (Müller and Guido, p. 252-260)

2.3.6.2 | Grid Search

Each machine learning method has a number of parameters that can be tuned to improve the performance of the model. Parameters are often not known in advance and must be tuned to the data. This is a common task in machine learning and therefore there are standard methods like grid search to find the best parameters. Grid search is a method of systematically working through multiple combinations of parameters (called grid), cross-validating as it goes to determine which tune gives the best performance. (Müller and Guido, p. 260-275)

2.4 | State of research

In recent years, there has been a growing adoption of ML in various applications related to sheet metal forming. With goal to achieve optimal manufacturing quality. Three main types of ML algorithms are used in sheet metal forming applications, namely supervised learning, unsupervised learning, and reinforcement learning (Cruz et al., 2021, p. 2).

Sheet metals have a high ratio of yields strength to elastic modulus, which results in significant spring back, which can compromise the accuracy of the final product. As a result, extensive research has been carried to estimate the spring back behavior in sheet metal forming processes (Liu et al., 2021, p. 565).

Current research on mitigating the spring back in sheet metal forming is primarily focused on compensations methods through numerical simulations, most notably finite element analysis (FEA) (Liu et al., 2021, p. 565) Lingbeek et al. (2005) present two methods for compensating spring back in the deep drawing process, but more reliable simulations is needed of industrial applications.

Also traditional supervised learning methods where used in metal forming applications.

Liu et al. used a Support Vector Machine (SVM) to predict the spring back in a micro W-bending process, demonstratring high prediciotn accuracy and generalization performance in comparison to experimental results (Liu et al., b, p. 1) Dib et al. evaluated several ML algorithms for predicting the spring back in a maximum thinning in U-channel and square cup forming processes. The MLP found to be the best model Dib et al.. Similarly Abdessalem et al. compared the performance of a quadratic response surface metod (RSM) and two SVM models for determinig the best surrogate model for probalistic strucutre oprimization of hydrofomred sheet metals. Both SVM models outperformed the RSM model (Abdessalem and El-Hami, 2015).

It has to be noted that most of the research including ML relies on FEA to generate the data for training and testing the models. This is due to the fact that it is time consuming to obtain enough experimental data for sheet metal forming processes. Dib et al. not that the implementations of virtual tryout still heavily relies on human expertise to make critical decisions. Also the use of FEM does not necessarily eliminated unexpected defects due to variations in material properties, tool geometry and process parameters (Dib et al., p. 2).

With the availability of data there has been an increased use of Machine Learning **ML!** (**ML!**) in sheet metal forming with the goal to reduce costs and increase manufacturing quality. Bock et al. (Cao et al.) lesen The ML algorithms can be divided into the main categories supervised learning, unsupervised learning and reinforcement learning. (Liu et al., a) Supervised learning is generally used in classification problems and regression problems while unsupervised learning is used to find patterns in data (Cruz et al., 2021, p. 2).

Spring Back Prediction Using Unsupervised Learning

Artificial Neural Networks Artificial Neural Network (ANN)s are widely used in sheet metal forming because of their high accuracy and generalization performance. (Cruz et al., 2021, p. 2) (Narayanasamy and Padmanabhan) which compared regression and neural network modeling for predicting spring back of steel sheet metal during the air bending process. They observed that ANN was able to predict the spring back with higher accuracy. But they had a sample size of 25 and suggested further research. (Inamdar et al.) developed an ANN for the air bending process to predict spring back as well as the punch travel to achieve the desired angle in a single stroke. (Kazan et al.) developed an ANN trained with FEM simulation data to predict the spring back for the wipe-bending process.

Because ANNs need a large amount of data to train the model generating the data with real machines is a time-consuming process. Therefore, it is common to use ANNs trained with Finite Element Method Finite Element Method (FEM) simulation data. *Was* *sind die Nachteile von FEM? Warum nutze eich "echte" Experimente?*

Spring Back Prediction Using Supervised Learning

Liu et al. (2019) used a Support Vector Machine SVM to predict the spring back of micro W-Bending operations. Liu et al. (b) Dib et al. (2019) compared different ML techniques (logistic regression, SVM, KNN , ANN, random forest, decision tree, naive Bayes, MSP) to predict the spring back and the occurrence of defects in sheet metal. (Dib et al., p. 1) The authors conclude that the MLP and the SVM are the best performing algorithms and suggest further studies of ML regressions models and kriging regression models. (Dib et al., p. 13)

3

Research methodology

The research method used in this thesis follows the design science research (DSR) approach (von Rennenkampff et al., 2015, p. 17). DSR is a research paradigm in which the designer tries to create artifacts to answer questions for problems.

DSR is a research paradigm in which the designer creates artifacts and uses them to answer questions for problems and generate new scientific knowledge. The designed artifacts are both useful and fundamental to understanding the problem (Hevner and Chatterjee, 2010, p. 10)

Design, according to Peffers et al. (2007), is the creation of an applicable solution to a problem (Peffers et al., 2007, p.47)

According to Hevner et al. (2010) design" is both a process ("a set of activities") and a product ("artifactt"). (Hevner and Chatterjee, 2010, p .78) The design-oriented research approach as a methodological framework seems well suited to answer the research questions. Predicting spring back and bend deduction is a relevant problem in business practice. Also, the conception and implementation of machine learning models is a design activity.

The term artifacts is intentionally broad and can take on different forms. In this work, the artifact is different machine leaning models which are applied on the generated data. DSR can be implemented in various ways, a prominent example is provided by Peffers et al. and shown in Figure 3.1 The approach comprises six steps, which are dived into the superordinate phases "Build" and "Evaluate". This thesis follows these phases.

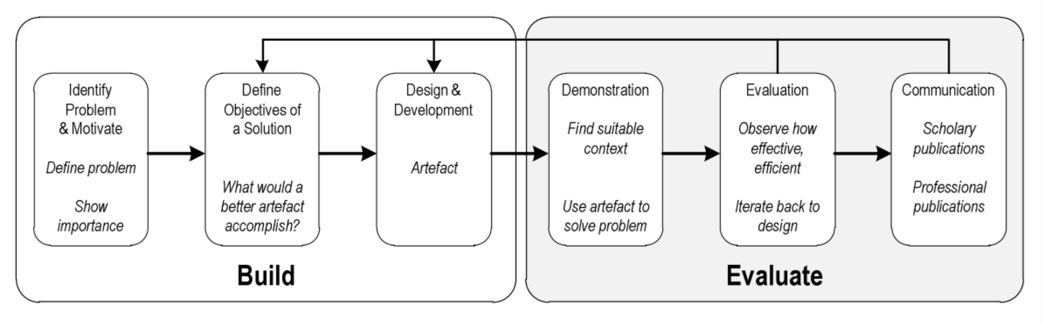


Figure 3.1: Design Science Research Approach according to Peffers et al.
Picture: (Sonnenberg and vom Brocke, 2012, p. 72)

Activity 1 - Problem identification and motivation This activity includes defining a specific research problem and the value of a potential solution. The problem is used for the development of the artifact. To reduce complexity, the problem should be divided into sub-problems. For problem-solving, explicit methods such as system requirements gathering or an implicit method such as programming and/or data analysis. (Peffers et al., 2007, p. 52)

Activity 2: Define the objectives for a solution The goals of a solution are derived from the problem definition. These are derived in the context of what is possible and feasible. Objectives can be quantitative or qualitative. Objectives should be derived from the problem specification and are thus based on the previous step. For knowledge about previous solutions and their effectiveness are required (Peffers et al., 2007, p. 55)

Activity 3: Design and development This step involves the creation of the artifact. An artifact can potentially contain models, methods or constructs, it can be anything that contributes to the solution of the research question. This step includes the definition of the functionality and architecture of the artifacts, followed by the creation of them. (Peffers et al., 2007, p. 55)

Activity 4: Demonstration The use of the previously created artifactt is demonstrated for one or more problems. This requires effective knowledge of the artifact. (Peffers et al., 2007, p. 55)

Activity 5 - Evaluation It is observed and evaluated how well the developed artifact provides a solution to the defined problems in activity 1. Knowledge of relevant metrics and methods of analysis is assumed. Depending on the nature of the problem, the evaluation can take different forms. A comparison of the functionality of the artifact and other solutions can be considered. Furthermore, quantified quantified parameters can be used to measure the performance of the artifacts (Peffers et al., 2007, p. 56) Hevner et al. suggest five different evaluation methods: Observational methods, analytical methods, experiments, testing of the artifact and descriptive methods (Hevner et al., 2004, p. 87)

Activity 6 - Communication The problem and the artifactt and its benefit are communicated externally (Peffers et al., 2007, p. 56) Hevner et al. describe in a conceptual framework guidelines for the

3.1 | Design Principles

Design Principles (DP) are seen as a central part of design-oriented research. (Gregor et al., 2013, p. 348) Design principles are characterized as "principles of form and function" as well as "principles of implementation" of an artifact. (Gregor and Jones, 2017, p.8) They are used to close the gap between researchers and user and allow prescriptive research on systems. They are used to capture knowledge about the artifact. (Sein et al., 2011, pp. 37-56). Koppenhagen et al. suggest generating design principles by grouping requirements for the solution and then creating core requirements, which can which can then be DPs. (Koppenhagen et al., 2012, p. 6)

3.2 | Evaluation of Machine Learning Models

Evaluating ML models is different from evaluating other software artifacts for several reasons. One of them is that data-driven software components, like ML models, have functionality that is not completely defined by the developer but is instead learned from data. Therefore, using machine learning presents new challenges compared to traditional software engineering (Siebert et al., 2022, p. 2).

In the field of software engineering, there are already standards that define the quality of software systems and its components. Most notable the ISO/IEC 9126 standards, which provide a quality model which is widely adapted in software engineering (Source). For above-mentioned reasons these standards are not suitable to evaluate machine learning models, therefore Siebert et al. (2022) note that they need to be adapted. They reinterpreted and extended these existing quality models to the ML context (Siebert et al., 2022, p. 1).

In order to define the Design Principles for the ML models this work, the considerations of (Siebert et al., 2022) are used. This enables a systematic process in order to assess the quality of the developed models.

In the work of Siebert et al. (2022), several quality attributes are considered, including Correctness, Relevance, Robustness, Stability, Interpretability and Resource Utilization. Alongside these attributes, the authors also provide a set of quality measures to evaluate the models, but these are nor complete and are only used as a starting point for the evaluation of the models in this work, further evaluation is described in Chapter 5.

It is important to note, that the mentioned quality attributes and measures should not be seen as a complete set of quality attributes and measures for ML models. Other studies, such as Zhang et al. (2020) have proposed additional sets of quality attributes. These additional perspectives will also be considered in the evaluation of the models, but only as a supplementary means of evaluating model performance.

Attributes that are the same as in Siebert et al. (2022) are correctness, relevance,

Correctness Model Relevance RobustnessSecurity Efficiency Fairness Interpretability

Attributes like Security, Fairness and Privacy (Zhang et al., 2020, p. 3) are not considered in this work, as they are difficult to measure and my not be relevant for the use case of this work.

3.3 | Goal Question Metric Approach

To make the defined quality attributes measurable, the “Goal-Question-Metric”-approach GQM was chosen in this work. It is one of the most common approaches in DSR and is divided into three levels. (Basili et al., 2002, p. 3)

1. Conceptual level (goal): "A goal is defined for an object, for a variety of reasons, with respect to various models of quality, from various points of view, relative to a particular environment." (Basili et al., 2002, p. 3)

2. operational level (question): "A set of questions is used to characterize the way the assessment/achievement of a specific goal is going to be performed based on some characterizing model. Questions try to characterize the object of measurement (product, process, resource) with respect to a selected quality issue and to determine its quality from the selected viewpoint." (Basili et al., 2002, p. 3)

3. quantitative level (metric): "A set of data is associated with every question in order to answer it in a quantitative way." (Basili et al., 2002, p. 3)

Objectives, questions and metrics can be presented in a hierarchical structure.

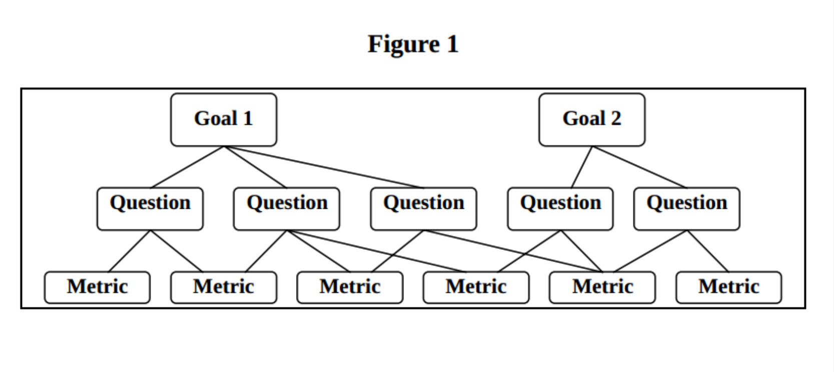


Figure 3.2: Goal-Question-Metric Ansatz (Basili et al., 2002, p. 3)

4

Build

This chapter outlines the steps of the Design Science Research (DSR) process, which are summarized under the "Build" phase, as described chapter 3 resarch methodology. The first step in this phase is to identify the problem that need to be addressed, this step leads to the formulation of Design Principles (DP), which are later used to evaluate the effectiveness of the artefact. The DP are derived from the requirements of the overall models and are based on the work of Siebert et al. (2022).

The Build phase involves the creation of machine learning models, which serve as artifacts. By implementing these steps, the DSR process establishes a rigourous and methodical approach for creating new artifacts.

4.1 | Problem Identification and Motivation

For several decades, researchers have been conducting extensive research on sheet metal forming technologies in response to the growing demand for lightweight metal components. One of the challenges is the phenomenon of spring back, as discuess in **??**. As a result, various efforts have been made to devise methods for compensation (Liu et al., c, p.1).

Sheet metal forming is a complex process due to non-linear behavior caused by deformations of the metal sheet, making the spring back difficult to predict. Consequently, conventional methods often employ trail and error approaches (Dib et al., p. 1). Through speaking with experts of the field of sheet metal forming, it was found that a usual trial and error approach is the creation of so called"technology tables" that contain the bending parameters and resulting spring back data. These tables are produced by conducting numerous experiments with varying bending angles and metal sheets. However, this process is time-consuming and costly, making it unsuitable for the production of high-volume and low-cost components.

SOURCE: Peter and Dr. Hochstrate but only while talking to them. Find a proper source

4.2 | Objectives of a Solution

The design principles used to evaluate the machine learning (ML) models in DSR process and form the objectives of the solutoin. work are based on a selection of quality parameters proposed by Siebert et al. Siebert et al. (2022). These quality parameters have been tailored specifically for the evaluation of ML models and are thus well-suited to the evaluation of the models developed in this thesis.

Design Principle 1: Correctness

"Does the artifact predict the spring back of a sheet metal with a high accuracy and correctness?" (Siebert et al., 2022, p. 16)

The demand for high-quality products is a driving force in the manufacturing industry, as consumers and businesses alike demand products that meet stringent standard of precision and accuracy. This is particularly true in the case of sheet metal forming, where even small deviations from the desired dimensions can cause significant implications for the final product. Therefore, metal parts need to be produced with high precision and accuracy and spring back is an undesired side effect which needs to be minimized (Cruz et al., 2021, p.1).

ML has emerged as a powerful tool for predicting the spring back in other bending methods as shown in section 2.4 state of research. A ML model should predict the spring back with a high degree of accuracy and correctness, as even small errors in the prediction will cause fitting problem in the manufacturing process.

Design Principle 2: Relevance

In addition to measure the correctness it is important to understand “why” the learner has this performance. A model is considered relevant when it is able to accurately predict the outcomes of new unseen data, that were not used during training. A relevant model should have a low error rate on both the training data and the test data and achieve a good bias-variance trade-off (Siebert et al., 2022, p. 16).

Understanding the bias-variance trade-off is crucial in developing ML models that generalize well to unseen data (Zhou, 2021, p. 49–51). In machine learning (ML), achieving a good bias-variance balance is essential. This means that a model can accurately capture the underlying patterns in the data without overfitting or underfitting.

A good bias-variance is essential a ML it means that a model can accurately capture the underlying patterns without overfitting or underfitting to the data (Zhou, 2021, p. 49–51).

Design Principle 3: Robustness

When working with real-world data, quality issues such as outliers, missing data, and noise are common problems. These issues can have a negative impact on the performance of the model, making it challenging to produce accurate predictions. To overcome these challenges, a robust model must be able to handle data quality issues and still produce accurate predictions (Siebert et al., 2022, p. 16).

Sáez et al. proposed a new measure called Equalized Loss of Accuracy to evaluate classifications models for robustness. Because regressions algorithms are used in this new metrics have to be found (Sáez et al., p. 3).

Design Principle 4: Stability

The stability of a model is an important quality parameter, as it is essential that the model produces the same results when trained on different datasets. This is particularly important in the case of ML models, as the training data is often limited and the model is trained on a small subset of the available data. Therefore, it is important that the model is able to generalize well to unseen data and produce repeatable results when trained on different datasets (Siebert et al., 2022, p. 16).

Design Principle 5: Interpretability

Interpretability refers to the ease with which humans can understand and make sense of the decisions made by a trained machine learning model (Molnar, 2020, p. 13). Miller (2019) define interpretability as the “degree to which a human can understand the cause of a decisio” (Miller, 2019, p. 1). Good interpretability is important because it allows users to trust and rely on the model, and it can also help with debugging and improving the model (*Source*).

Interpretable models will also deliver more insights for this project, as the goal is to understand the relationship between the features and the target outcome.

As mentioned in chapter 2 there are many parameters and variables involved in the sheet metal forming process. That makes the process design quite complex, particularly in the production of components which require several stages, and thus more than one set of tools (Dib et al., p. 1). A interpretable model which allows conclusions how the results where generated is better.

Design Principle 6: Resource utilization

One of the main objectives of using machine learning to predict spring back is to minimize the number of trial-and-error cycles in the manufacturing process and save resources (see Section **??**). However, it is important to consider that creating an ML model also requires resources.

Therefore, it is crucial to take into account the resources needed to train and make predictions with the model (Siebert et al., 2022, p. 16).

4.3 | Dataset generation

To generate the dataset, bending experiments on metal sheets of varying thicknesses where conducted. Specifically, cold rolled steel sheets that conform to the DIN EN 10130 with thicknesses raning from 0.5 mm to 3 mm where used. The material was selected because it is commonly used in bending processes and widely available. To create the dataset, pieces of the material were cut into 20×100 mm pieced each piece was bent using a Zwick three-point-bending machine. It was paid attention to ensure that each metal sheet was bent in the same rolling direction as preliminary tests showed that the spring back differed depending on the rolling direction.

The dataset consisted of 384 `.tra` files which is a proprietary format used by the machine. Python script where developed to covert the output data format from the machine to CSV files.

The following describes the experimental setup used for the experiments performed. The data collection was done outside of the thesis period and originally consited of 384 samples. `X` samples where added later to the dataset to fill gaps present in the dataset.

4.3.1 | Experimental setup

The experimental setup comprises of a three-point bending machine, consisting of a punch and die, with the latter lacking a bottom, which allows only air bending.

The material testing machine utilized is the *Zwick MX 25A*, which is equipped with a load cell and a displacement sensor. The load cell measures the force applied to the sheet measured in *N*), while the displacement sensor measures the displacement of the punch, denoted as y_p. The punch is mounted on the top of the machine and is stationary, while the die is mounted on the bottom and is the part which can be moved. The die opening of the machine is adjustable with from 10mm to 100 mm. The machine is operated via a computer and the *ZwickRöll TestXpert* software, which is used for both machine control and data collection.

The experimental setup and the process parameters are shown in Figure 4.1 where V is the die opening which is the opening of the two contact points on which the sheet metal is placed. The parameter y_p is the punch penetration, which is the distance the punch is moved into the sheet. The metal sheet thickness is donated as t and the bending angle as α. The parameter r_p is the radius which of the tip of the punch, which was never replaced in the experimental setup and therefore remains constant.

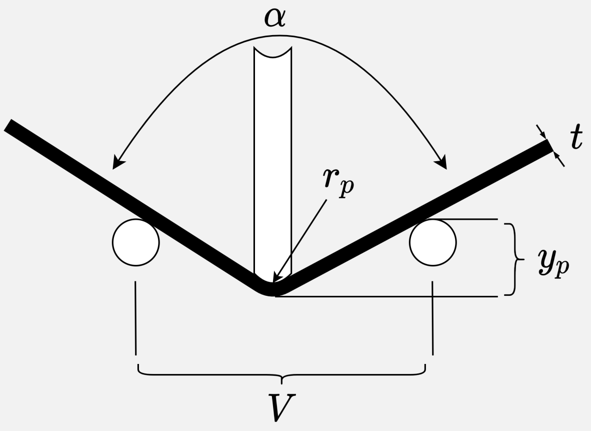


Figure 4.1: **Process parameters:** Sheet bending angle (α), sheet thickness (t), punch penetration (y_p), die opening (V) and punch radius (r_p)

To ensure consistent results, a set of constant and variable parameters were selected. The constant parameters consisted of the punch-and-die tooling made of steel, which included the die punch radius (r$_p$)*, aswellasthelengthandwidthofthemetalsheet.Additionally, thepunchtrav*

All metal sheets used in the setup were standardized with a length of 80 mm and a width of 20 mm. The hold time, which refers to the duration that the punch remains stationary after reaching the maximum displacement ($y_{p\max}$), was set to a minimum of

1 second. The punch force threshold was set to 1 N, meaning that the punch was initially moved at a higher speed until the force reached 1 N, and then moved at a slower speed of 80 mm /min until the $y_{p\max}$ was reached.

Parameter	**Values**	**Unit**
Punch radius	5	mm
Sheet width	20	mm
Sheet length	100	mm
Punch speed	80	mm/min
Punch speed up (after bend)	8	mm/min
Hold time	1	s
Punch force threshold	1	N

Table 4.1: Constant parameters in th eperimental setup

The experiment involved varying three parameters: the die opening (V), the maximum punch penetration (y_p), and the thickness of the metal sheet (t). The die opening was varied from 10 mm to 50 mm, while the maximum punch penetration was varied from 2 .5 mm to 20 mm. The thickness of the metal sheet was also varied, with values ranging from 0.5 mm to 2 mm. These parameters and their values can be seen in Table **??**.

Parameter	**Values**	**Unit**
Punch penetration y_p	2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20	mm
Die opening V	10, 20, 30, 40, 50	mm
Thickness t	0,5, 1, 1.5, 2, 2.5, 3	mm

Table 4.2: Varying parameters in the experimental setup

4.3.2 | Measuring The Spring Back

The output data included various data points that were used to calculate the spring back. Key parameters for this calculation were the force (“Standardkraft”), punch penetration (“Standardweg”), and time (“Testzeit”). Figure 4.2 illustrates these three parameters, with the blue line representing the force and the gray line representing the punch travel.

In Figure 4.2 it can be seen that the force jumps to around 10 N as soon as the punch touches the metal. This is the case because the travel speed of the punch is 80 mm/min which relatively fast. It was set to that speed in order to reduce the time of the experiments.

The wait time at of 1 second y_{pmax} is a limitation of the machine and can not be changed and therefore is always a part of the experimental setup.

Explain the dip

For a short time after the lift, the load cell still measures a force. That is because the metal sheet springs back and the punch is still in contact with the sheet. This was measured using a python script,the green and the yellow point represent the resulting spring back distance.

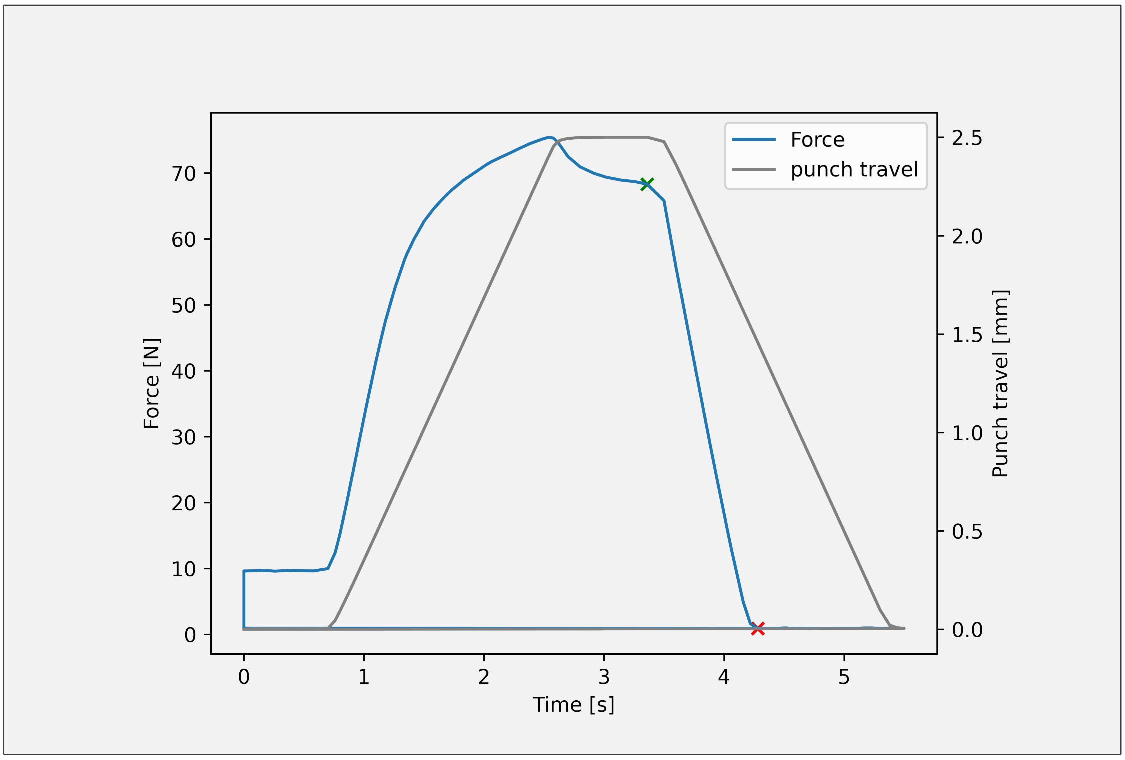


Figure 4.2: A steel metal sheet was bent with a punch penetration of 5 mm the spring back is 0 .37 mm. The blue line shows the force and the blue line shows the punch penetration.

4.3.3 | Dataset Exploration

The dataset was explored using the python library *pandas* McKinney (2010) and the *matplotlib* Hunter (2007) library. The goal of this section is to give the reader an overview of the dataset and to show the relationship between the features and the dependent variable.

4.3.3.1 | Features

The output data of the bending machine consisted of 26 features, which are listed in the appendix. Only three features - force, distance y_p, and time - are relevant for calculating the spring back, as described in the previous section (Section 4.3.2). These three features were combined with the calculated spring back to form the final dataset, which contained a total of 396 data points generated using the method described earlier. An example of the dataset is presented in Table 4.3.

index	Distance	Spring Back	Thickness	Die Opening
1	5.0	0.6667	2.0	50
2	15.0	0.9164	2.0	50
3	10.0	0.6829	2.0	50
...	...	...	...	...
396	5.0	0.6667	3.0	10

Table 4.3: Varying parameters in the experimental setup

Figure 4.3 illustrates the spring backs observed in the $V30$ dataset. Two general trends are evident. First, a lower thickness of the metal sheet results in a higher spring back. Second, a higher punch penetration results in a higher spring back. Also it non-linear relationship between the punch penetration and the spring back can be observed.

In the chosen example it chan be observed that the spring back of the metal sheets with $t = 0.5$ are significantly higher then the other thicknesses. Despite the availability of data, the factors underlying the spring back behavior remain incompletely understood. Because too many factors are involved in the bending process.

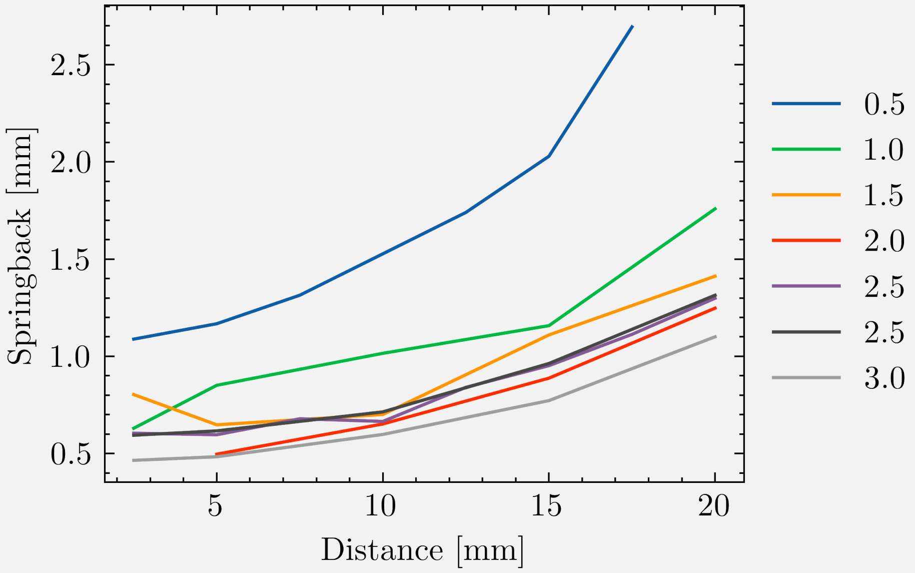


Figure 4.3: Springbacks for $V30$

4.3.3.2 | Data Quality

The dataset was created in a controlled experimental environment, where was given to ensure that the samples were measured with precision and accuracy. This results in a dataset that contains a minimal number of outliers and a high level of data quality, which is an essential requirement for reliable machine learning models.

As depicted in Figure 4.5, the dataset covers all possible combinations of the process parameters, V, t and y_p. The y_p values in the dataset are uniformly distributed and always range between 2.5 and 20 mm. The dataset was continuously expanded with new data points throughout the project to increase its size and diversity.

During the data collection and expansion process, several outliers and incorrect measurements were identified and removed from the dataset to ensure its high quality. Although this dataset has been developed in a controlled environment, it may not accurately represent real-world scenarios, where the data quality is often affected by a range of factors such as measurement errors, noise, and bias. This will be taken into account in later sections, where data quality issues will be added in order to evaluate the robustness of the models.

4.3.4 | Data Preprocessing

The three independent features y_p, V and t as well as the dependent feature $spring\_back$ were normalized using the `MinMaxScaler` or in some cases the `StandardScaler` from the `scikit-learn`Pedregosa et al. (2011) library. The`MinMaxScaler` scales the data between 0 and 1. The scaler was fitted on the training data and then used to transform the test data. The scaler was saved to be used for the prediction of the spring back of the real world data.

Scaling is only done on the training data, because cross-validation is later used to tune and evaluate the models. Scaling the whole data set before the split would lead to data leakage because the minimum and maximum values of the test data would be used to scale the training data. How the data was split can be seen in Figure 4.5.

4.3.5 | Computational Setup

For training the machine learning models a ThinkPad X1 Carbon 2019 with an Intel Core i7-10610U CPU @ 1.80GHz and 16 GB RAM was used. The operating system used is Ubuntu 20.04.2 LTS. The code for the model is written in Python 3.8.5 using the IDE PyCharm. The libraries used are mentioned in Table 4.4.

Library	**Version**
numpy Harris et al. (2020)	1.23.2
pandas McKinney (2010)	1.5.1
matplotlib Hunter (2007)	3.6.2
scienceplots Garrett (2021)	2.0.1

Table 4.4: Libraries used for the machine learning models.

Upon examining the correlation matrix depicted in Figure 4.4, it is evident that the distance and spring back features exhibit a stronger correlation than the other features. This correlation is to be expected since punch penetration y_p is the primary factor that influences the amount of spring back. It is noteworthy that the other features are not correlated with each other, indicating the absence of multicollinearity, which is a desirable trait for machine learning models.

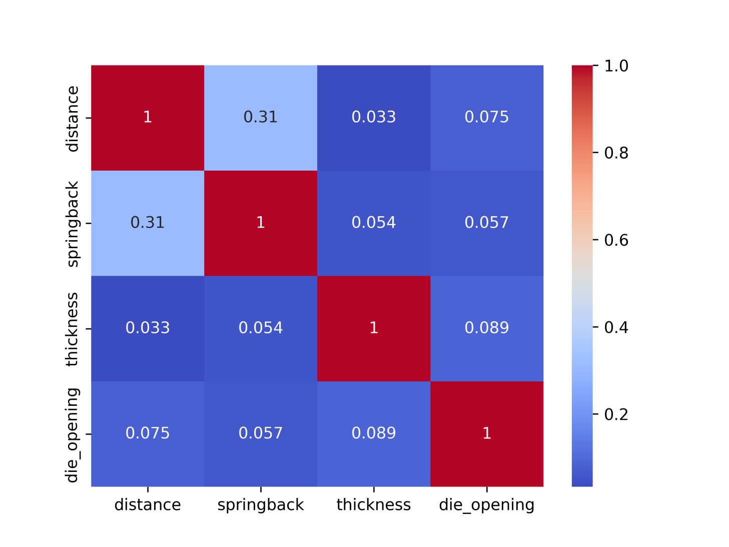


Figure 4.4: Correlation matrix

4.4 | Model Selection

This thesis focuses on the utilization of supervised machine learning models to predict spring back. Therefore, a selection of the most common machine learning models where used, based on the systematic literature review of Dridi (2021). The author of the paper divided supervised learning approaches into five categories: Logic-based learning, instance-based learning, static-based learning, Deep learning and Support vector machines (Dridi, 2021, p. 8).

4.4.1 | Baseline Model

A linear regression model is used as baseline model in this thesis. Linear regression is a simple and widely used model that is used to predict a continuous value. The model is based on the assumption that the relationship between the independent variables and the dependent variable is linear. The model is trained by minimizing the sum of squared errors between the predicted values and the actual values (Müller and Guido, p. 47–68).

Since in the section 4.3.3 it has already been stated that the relationship between the independent variables and the dependent variable is not linear it is not expected that the linear regression model will perform well. The linear regression model is used to compare the performance of the other models.

4.4.1.1 | Linear Regression

A liner regression model uses the feature inputs to make prediction about the target by creating a linear relationship which is easy to understand and interpret (Molnar, 2020, p. 37).

Linear models can be used to understand the relationship between a target variable y and one or more input features x. The general formula for predicting in a linear model in regression is shown in 4.1 (Müller and Guido, p. 45). In this context, $x[0]$ to $x[p]$ represent the attributes (with p being the number of attributes) of a specific data point. The parameters w and b, which are learned by the model, and the predicted output $\hat{y}$, are also included (Müller and Guido, p. 45).

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + ... + w[p] * x[p] + b \tag{4.1}$$

Linear Regression (LR) models estimate the target value as linear combinations of the feature inputs. The linearity of the relationship between the inputs and the target makes the interpretation of the model straightforward (Molnar, 2020, p. 37).

Linear regression has no hyperparameters to tune and therefore no hyperparameter tuning was performed.

4.4.2 | Statistics Based Learning

The approach that relies on statistics and uses distributive statistics to make the problem more manageable. It predicts based on the distribution structure. Learning through statistics involves utilizing the Naïve Bayes Algorithm (Dridi, 2021, p. 8).

4.4.2.1 | Naive Bayes

...

4.4.3 | Support Vector Regression (SVR)

Support Vector Machines (SVM) are a popular approach for solving classification problems. The algorithm seeks to identify a hyperplane in an N-dimensional space, where N represents the number of features, to effectively separate and classify data points (Awad and Khanna, p. 42).

However, for regression problems such as predicting spring back, the SVM algorithm needs to be adapted to provide continuous values instead of a finite set of values. To address this, the SVR algorithm was developed, which draws inspiration from the SVM algorithm and leverages similar principles. SVR fits a model to data by only considering residuals smaller in absolute value than a designated constant known as ϵ-sensitivity. By doing so, SVR can accurately model and predict continuous values, making it a suitable approach for regression problems.

To enable the SVR algorithm to generate continuous predictions, it creates a "tube" while the points outside the tube are penalized based on their distance from the predicted function. This approach is similar to how SVMs penalize points in classification problems (Montesinos López et al., p. 369). The kernel trick is a method to transform the data into a higher dimensional space, where the data is linearly separable. This is done by using a kernel function, which is a function that maps the data into a higher dimensional space. Two methods are usually used for SVMs, the polynomial kernel and the radial basis function,also known as gaussian kernel (Müller and Guido, p. 97-98).

Like SVM, the SVR algorithm finds a well-fitting hyperplane to a kernel- induced feature space to achieve good generalization performance using the original features. This is detailed in (Montesinos López et al., p. 369).

The hyperparameters of the SVR algorithm are listed in table **??**.

Hyperparameter Description	**Value**
kernel	rbf
degree	1
gamma	0.1
C	4000
epsilon	0.001

Table 4.5: Hyperparameters of the SVR model

4.4.4 | Logic-based learning

Logic-based algorithms solve problems by applying logical functions sequentially or incrementally. An example of such an algorithm is a decision tree, which is commonly used as a classification and regression model (Dridi, 2021, p. 10).

4.4.4.1 | Decision Trees

Decision Trees (DTs) are a widely used type of model for both classification and regression problems. These models construct a hierarchy of rules to predict outcomes based on the data (Müller and Guido, p. 70) (Shaik and Srinivasan, p. 253). In DT models, the data is split into multiple segments based on specific feature values. This division creates various subsets of the dataset, with each sample belonging to one subset. The final subsets are called terminal or leaf nodes, while the intermediate subsets are known as internal or split nodes. To predict the outcome in a specific leaf node, the mean outcome of the training data in that leaf node is used (Müller and Guido, p. 70-72).

Decision trees are models are useful when the relationship between features and outcome is or when features interact with one another. These models split the data multiple times based on certrain cutoff values, resulting in different subset of the dataset. The final subsets, called leaf nodes are used to predict the outocme by taking the average of the training data in that subset (Molnar, 2020, p. 76).

Grid search cross validation was used to find the best hyperparameters for the decision tree. All hyperparameters are summarized in Table **??**.

Hyperparameter **Description**	**Value**
criterion	absolute_error
max_depth	30
min_samples_split	4
min_weight_fraction_leaf	0.
splitter	random
ccp$_{a}lpha$	0.0

Table 4.6: Hyperparameters of DT model.

4.4.4.2 | Random Forest

The main problem with DT is their tendency to overfit and have poor generalization performance, making them unsuitable for most use cases. To address this, ensemble methods are typically used instead of a single DT (Müller and Guido, p. 78) (Liu et al., c, p. 251). One popular ensemble learning algorithm is the RF (Breiman), which combines multiple decision trees (weak learners) to create a more accurate and stable prediction (strong learner) (Awad and Khanna, p. 24). Because the data is divided into smaller subsets, and a randomized tree predictor is build for each subset this aproach is also called "divide and conquer" approach. The risk of overfitting is mitigated by using subset and feature randomization. Each root node uses a unique subset of the data, and each leaf is split using a random set of features. This ensures that no single tree sees all of the data, allowing the model to focus on general patterns rather than being sensitive to noise (Liu et al., c, p. 251).

This mechanism is flexible enough to handle classifications and regression problems, this is one of the reasons that random forests count to the most successful ML methods (Biau and Scornet, p. 3-4) (Breiman, p. 25).

Random forests are a type of machine learning algorithm that uses bagging and the random selection of features to produce accurate results. They are effective at handling noise and can work with both continuous and categorical variables. This combination of techniques helps improve the performance of the algorithm (Liu et al., c, p. 259). Decision trees have a limitation in their ability to overfit, which is a disadvantage. his is mitigated by the use of subset and feature randomization. Specifically, each base model uses a unique subset of the data, and each node in the decision tree is split using a random set of features. This ensures that no single tree sees all of the data, allowing the model to focus on general patterns rather than being sensitive to noise (Liu et al., c, p. 259).

Hyperparameter Tuning

Grid search cross validation was used to find the best hyperparameters for the random forest model using Scikit-Learn's default `GridSearchCV` function. All hyperparameters are summarized in Table 4.7.

The *max_features* was set to *auto* and therefor the models does use all available features. As described in Figure 5.6 only limited features are available and all of them are important for the dependent variable spring back The default unpruned model did overfit the training data and therefore was not able to generlaize well on new data (as decvitbed by (Müller and Guido, p. 133-136)). There it was set to 10 using `GridSearchCV`.

Hyperparameter Description	Value
n_estimators The number of trees in the forest.	10
criterion	absolute_error
max_depth	30
min_samples_split	4
min_samples_leaf	2
max_features	auto
max_leaf_nodes	X
max_leaf_nodes	X

Table 4.7: Hyperparameters of the RF model.

4.4.4.3 | Gradient Boosting Regression Tree

A gradient boosting regression is a type of ensemble learning algorithm in which multiple decision trees are combines to produce a more accurate and stable prediction. Similar to the RF gradient boosting combines multiple weak learners to create a strong learner. The difference to a RF is, that the trees are trained in a serial manner and each tree corrects the errors of the previous tree (Müller and Guido, p. 88–89).

Gradient boosted tree use strong pre-pruning and therefore produce shallow trees with a depth of one to five. This brings the advantage of a smaller model which uses less memory and also results in a faster prediction. Usually generating more trees improves the overall performance of the model (Müller and Guido, p. 88–89). Also the algorithm performs well without scaling the dataset and can handle a mixture of binary and continuous features (Müller and Guido, p. 88–89). Like other tree-based models it does not perform well on high-dimensional data.

Gradient boosted trees (GBT) are more sen "Gradient boosted trees are frequently the winning entries in machine learning competitions, and are widely used in industry. They are generally a bit more sensitive to parameter settings than random forests, but can provide better accuracy if the parameters are set correctly." (Müller and Guido, p. 88-89)

"Apart from the pre-pruning and the number of trees in the ensemble, another important parameter of gradient boosting is the learning rate, which controls how strongly each tree tries to correct the mistakes of the previous trees. A higher learning rate means each tree can make stronger corrections, allowing for more complex models. Adding more trees to the ensemble, which can be accomplished by increasing n estimators, also increases the model complexity, as the model has more chances to correct mistakes on the training set." (Müller and Guido, p. 88-89)

"The main parameters of gradient boosted tree models are the number of trees, n estimators, and the learning rate, which controls the degree to which each tree is allowed to correct the mistakes of the previous trees. These two parameters are highly interconnected, as a lower learning rate means that more trees are needed to build a model of similar complexity. In contrast to random forests, where a higher n estimators value is always better, increasing n estimators in gradient boosting leads to a more complex model, which may lead to overfitting. A common practice is to fit n estimators depending on the time and memory budget, and then search over different learning rates." (Müller and Guido, p. 88-89)

Hyperparameter Tuning

The *kernel*, *degree*, gamma and *epsilon* where set using grid search cross validation. Gamma controls the width of the gaussian kernel, it determines when points are close or far away. The C parameter controls the importance of each point.

The features of the dataset have different order of magnitude, this is already a problem for other models, but big effects on the kernel SVM. To resolve this problem the data was scaled using the *MinMaxScaler* between 0 and 1. The model trained on the scaled data performed better than the model trained on the unscaled data.

4.4.4.4 | Extra Trees

When developing a tree within a random forest, only a portion of the features are assessed for splitting at each node, which was previously explained in section **??**. To further increase the randomness of the trees, it is possible to utilize random thresholds for each feature rather than determining the most optimal thresholds like traditional DT do (Géron, 2022, p. 351).

An ensemble of this kind highly arbitrary trees is known as an extra-trees (or extremely randomized trees) ensemble. This approach exchanges more bias for a lower variance. Furthermore, the use of Extra Trees (ET) classifiers makes the training process faster compared to conventional random forests since one of the most time-intensive aspects of tree development is determining the optimal threshold for each feature at each node (Géron, 2022, p. 351).

4.4.5 | Neural Networks

Another method for supervised learning involves the utilization of Neural Netowrks to perform classification and regression tasks. Neural Network models consist of multiple layers, and it is trained through a layer-by- layer approach (Dridi, 2021, p. 11).

While this thesis does not delve into the use of Deep Learning for predicting spring back, it does employ a simple multi-layer perceptron as a baseline model. Although not the main focus, the results from this model can provide insights into the potential of neural networks for this task and for guiding future research.

4.4.5.1 | Multi-layer Perceptron

The multi-layer perceptron (MLP), which is the most widely used type of artificial neural network, has to be found producing generalizable models in various fields (Taud and Mas, 2018, p.451).

The MLP is a feedforward neural network that is organized into layers, with information moving in one direction from the input layer tough the hidden layers to the output layer Bishop et al. (1995). Each connection between neurons has a weight and perceptrons within the same layer share the same activations function which is typically a sigmoid function in hidden layers. The backpropagation algorithm is typically used to train MLP models, by propagating errors backwards from the output layer to the input layer in order to adjust the weights (Taud and Mas, 2018, p. 454).

Hyperparameter **Description**	**Value**
random_state	1
solver	lbfgs
max_iter	5000
alpha	0.01
hidden_layer_sizes	(100, 100)
learning_rate	constant

Table 4.8: Hyperparameters of the MLP model.

4.5 | Model Training

4.5.1 | Training-Test Split

Figure 4.5 displays the dataset partitioned into training and testing sets for the models used in this study. Specifically, the samples with a die opening of 30 were reserved for testing, while the remaining data was used for training . While a random train-test split can often yield better model performance, this approach may not evaluate the model's ability to predict new data with a different die opening.

The decision to use a die opening of 30 was based on its central location within the selected dataset, allowing the models to predict this value accurately. Furthermore,removing data from the middle of the parameter space during training is expected to improve model performance and promote generalization to new data.

All models were trained with the same dataset and parameters, differing only in the chosen algorithm.

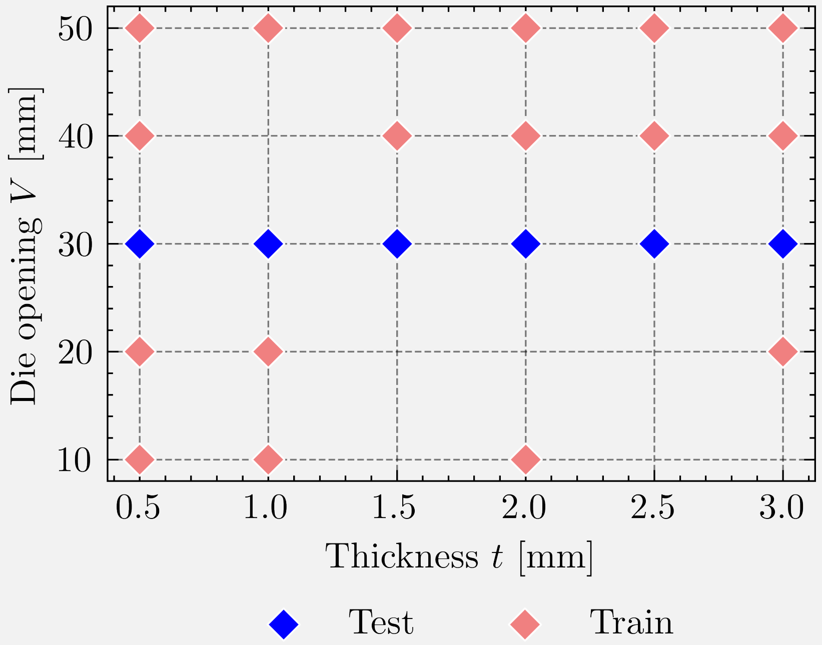


Figure 4.5: The training and test dataset. (Data especially for V40 is still missing)

In addition to the data split shown in Figure 4.5, a random 80 /20 split was used to train and evaluate the models. This split was employed to assess model performance and compare results.

Using two different methods for data splitting serves two distinct purposes. The first method evaluates the models' ability to generalize on new data. The second method assesses the models' performance on the dataset. In real-world applications, it may be possible to train models on pre-existing data that encompasses all necessary $V - t$-combinations.

While the models are expected to perform better on the random data, less information can be gained about their ability to generalize to new data because the random split will most likely contain data from all $V - t$-combinations.

4.6 | Structure of The Code

The code is available on GitHub: `https://github.com/Philipp0205/Master-Thesis-Code`

The dataset can be found under the `data/dataset` folder. All the used models are in the `learner` folder. Each learner class has a `main` function which can be used to run the model. Also the reports-array can be used to generate a specific report for the model. The implementation of all reports can be found in the `reports` folder.

5

Evaluation

This chapter conducts a comprehensive analysis of the ML models implemented in the preceding chapter. An overview of the quality metrics used for the evaluation is presented in Table 5.1. The quality metrics have been structures based on the GQMapproach mentioned in subsection 3.3.

The quality metrics are aligned with the quality model from Siebert et al. (2022), which in turn is based on the ISO/IEC 9126 standard for software quality evaluation. The standard has been adapted by Siebert et al. (2022) to meet the specific requirements of machine learning models, as described in section **??**.

As Siebert et al. (2022) mostly focuses on the evaluation of classification models, the quality metrics have been adapted to fit the requirements of the regression models implemented in this thesis. This is elaborated in each individual quality metric section.

Table 5.1 shows the quality metrics used for the evaluation of the ML models. The GQM approach is used to structure the evaluation of the ML models. To each goal a question or one or more metrics where assigned. The following subsections follow this structure and describe the quality metrics in more detail.

Goal	Question	Metric
Correctness	Ability of the model to perform the current task measured on the development dataset and the runtime dataset (Siebert et al., 2022, p. 16)	MAE, MSE, RMSE
Relevance	Does the model achieve a good bias-variance tradeoff? Which means neither overfitting or unterfitting the data. (Siebert et al., 2022, p. 16)	Variance of CV, R^2
Robustness	Ability of the model to outliers, noise and other data quality issues (Siebert et al., 2022, p. 16)	Loss of Accuracy, Average Loss
Stability	Does the artifact generate repeatable results when trained on different data? (Siebert et al., 2022, p. 16)	LOOCV stability
Interpretability	How well can the model be explained? (Siebert et al., 2022, p. 16)	Linearity, monotonicity, interaction
Resource utilization	How much resources are required to train and run the model? (Siebert et al., 2022, p. 16)	Training time, runtime, storage space

Table 5.1: Overview of the goals, questions and metrics for the evaluation of artifacts following the GQM approach.

5.1 | DP1: Correctness

The model must be able to perform well on the selected task. Siebert et al. (2022) used the classification metrics precision, recall and F- score to evaluate the correctness models. As the regression models implemented in this thesis are not able to predict the class of a sample, these metrics are not applicable.

Commonly used metrics for regression models are the mean absolute error Mean Absolute Error (MAE), mean squared error (MSE) and Root Mean Squared Error (RMSE). Their formal description is given in 5.1, 5.2 and 5.3 where e_i being the prediction error which is the difference between the predicted value by the model and the acutal value. y_i is the actual value and n is the number of samples in the testing data set.

For the evaluatoins the `sci-kit learn` (Pedregosa et al. (2011)) implementation of these metric was used.

Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| \tag{5.1}$$

The MSE is the average of the squared differences between the predicted and the actual values. The MSE is more sensitive to outliers than the MAE.

Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}e^2 \tag{5.2}$$

The RMSE is the square root of the MSE. The RMSE is the most popular metric for evaluating the performance of regression models. The RMSE is interpretable in the same units as the response variable. The RMSE is more sensitive to outliers than the MAE. The MAE is the average of the absolute difference between the predicted and actual values.

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{MSE} \tag{5.3}$$

For a full overview about the performance all three metrics are sued for the evaluation.

5.1.1 | Results

The Table 5.2 shows the results of the evaluation of the models for the correctness.

Looking at the RMSE all models except the LR and **DC!** (**DC!**) models show a relatively equal performance. The MLP has a slight advantage over the other models, with an RMSE of 0.201. Therefore it could be interesting to investigate the performance of neural networks / deep learning in the future.

It's worth mentioning that the SVM model has a slightly better MAE, while still maintaining a comparable MSE compared to the other models. This could be due to the fact that the errors made by the SVM are evenly distributed across instances, or that the errors made by the other models are large for a few instances but smaller for the rest.

The algorithms LR does perform worse than the other models. A likely reason is, that the relationship between the independent and dependent variables is not linear. This cna result in an inadequate fit and reduced predictive power (Source).

DT on the other hand, can perform poorly when they overfit the training data. Also, they are pron to instability, both problems are mitigates by using ensemble methods like RF and GBT which can overcome the limitations of individual decision trees.

Because the simple DT as well as the LR are not able to predict the spring back correctly, they are not considered for the next steps of the evaluation.

Model Name	rs	MAE	MSE	RMSE
LR	-	0.199	0.092	0.303
LR	x	0.261	0.127	0.356
DT	-	0.193	0.076	0.266
DT	x	0.212	0.119	0.346
RF	-	0.183	0.060	0.244
RF	x	0.134	0.041	0.202
ET	-	0.126	0.040	0.201
ET	x	0.134	0.041	0.202
GBT	-	0.189	0.058	0.242
GBT	x	0.143	0.085	0.292
SVM	-	0.09	0.04	0.20
SVM	x	0.09	0.04	0.20
MLP	-	0.114	0.041	0.201
MLP	x	0.121	0.040	0.199

Table 5.2: Overview of the used machine learning models and their metrics.

5.2 | DP2: Relevance

A model is considered relevant when it achieves a balance between bias and variance, avoiding both overfitting and underfitting of the training data. The relevance of the model can be quantified through the *variance of cross-validation*, which proves insight into how the model performs when trained and evaluated on different subsets of data and how generalizes.

A low variance indicates that the model's performance is consistent across different folds,suggesting that the model is not overfitting the training data. Conversely, a high variance implies that the performance can vary significantly depending on the specific data points used the test set, indicating a potential overfitting problem.

Figure 5.1 shows how the variance of cross-validation was calculated. The parameter E denotes to the estimator which is used to calculate the Cross-validation (CV) score. The parameter k denotes to the number of folds, which was set to 5 in this context. As estimator for the trained models R^2 was used. The R^2 is a statistical measure of how close the data are to the fitted regression line.

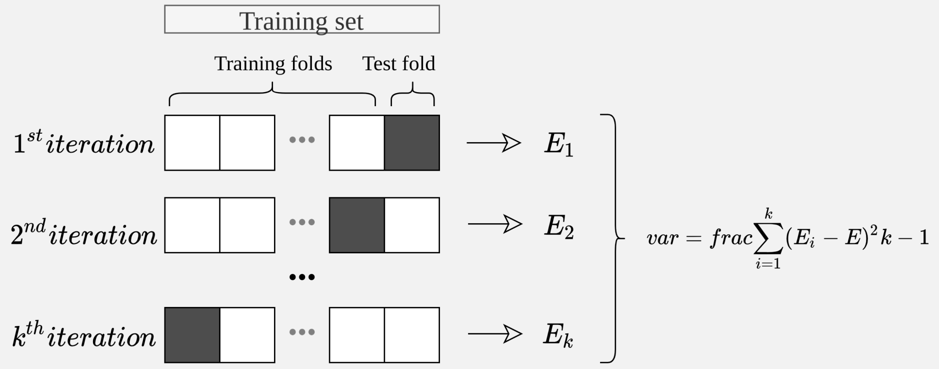


Figure 5.1: Variance of cross-validation

$$R^2 = \frac{Explained\_variance}{Total\_variance\_targert\_variable} \tag{5.4}$$

The value of the R^2 is the percentage of the response variable variation that is explained by a linear model. It returns a score between 0 and 1, where 0 means that the model does not explain any of the variance in the response variable around its mean, and 1 means that the model explains all the variance in the response variable around its mean (Müller and Guido, p. 43). Special in sci-kit learns implementations is that the score can be negative if the model performs poorly. If a model always predicts the expected value of y, regardless of the input features, it would receive an R$^2 scoreof 0.0$?.

Therefore, a high R^2 indicated a good model fir and good bias-variance tradeoff(Müller and Guido, p. 43).

5.2.1 | Results

Table 5.3 shows the variance of cross-validation and the R^2 for all used machine learning models. To calculate the variance of cross-validation the variance of Scikit-Learn's `cross_val_score` was calculated. Five-fold cross-validation was used to calculate the variance of cross-validation. The R^2 was calculated with the formula 5.4.

It can be observed that most of the models have relatively low variance, indicating consistent performance across different folds, however, two models stand out - GBT and acMLP.

For the GBT model, the variance is high, which implies that its performance fluctuates significantly across different folds. This inconsistency is a typical sign of overfitting, where the model is overly complex and has learned the noise in the data rather than the underlying pattern. The individual cross-validation scores for the GBT model also confirm this assumption, with a negative score on the first fold and relatively high scores on the subsequent folds. Given this poor performance, the GBT model cannot be considered relevant for the given task.

Similarly, the MLP model has a relatively high variance score, which suggests that its performance also fluctuates across different folds. The individual scores for the MLP model show that it performs poorly on a few folds.

This performance inconsistency could be due to either overfitting or high variability in the data.

Overall, it's important to consider the cross-validation scores in conjunction with the other other metrics to get a more complete picture of how well your model is performing. This will be done in later sections.

Model Name	rs	Variance of CV	R^2
RF	-	0.068	0.8
RF	x	0.071	0.805
ET	-	0.064	0.832
ET	x	0.352	0.82
GBT	-	0.425	0.775
GBT	x	0.426	0.866
SVM	-	0.061	0.838
SVM	x	0.061	0.798
MLP	-	0.146	0.839
MLP	x	0.146	0.871

Table 5.3: Performance of the used machine learning models according relevance.

5.3 | DP3: Robustness

The IEEE standard glossary of software engineering terminology describe robustness as "The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" (Terminology, 1990, p. 64). Sáez et al. (2016) extend this definition to fit machine learning models and describe it as the "capability of an algorithm to build models that are insensitive to data corruptions and suffer less from the impact of noise" (Sáez et al., p. 2). Siebert et al. (2022) specifically mention data quality issued like outliers or noise (Siebert et al., 2022, p. 16).

Unlike DP Correctness(5.1), robustness is a non-functional characteristic of a ML model. A way to evaluate the robustness is to check the correctness of the model with added noise to the data (Zhou, 2021, p. 18), if the model is still able to predict the correct values, it is considered robust.

As mentioned in section 4.3.3 the data set was generated in a controlled environment does not contain many data quality issues and noise. This offers the opportunity to test the robustness of the models by manipulating the data set to introduce the data quality issues missing values and noise.

5.3.1 | Missing Data

When applying the model to real-world data, it is possible that some values are missing. Looking at the available dataset two scenarios or missing data can occur: Missing Vt-pairings and missing values.

In the first scenario, there may be no data available for a specific die opening, which can be due to the die opening was never being used before or a lack of recorded information. In the second scenario, while data may exist for the desired Vt combination, certain values may be missing or the dataset may not be complete.

To address these scenarios, the following tests were conducted:

5.3.1.1 | Missing values

Initially, the use of the Leave-P-Out-Cross-Validation (LPOCV) method was considered. However, this approach was omitted due to computational constraints, as it generates all possible training and test sets by removing $p$samples from the complete dataset, resulting in a large number of overlapping test sets and a high computational cost.

As an alternative, regular CV was applied, with an increasing number of folds. The minimum number of folds was 2 and the maximum was equal to the total number of samples in the dataset. However, this experiment did not provide distinguishable results and was therefore not included in the results.

The third and final experiment utilized random sampling to create different train- test data compositions, allowing for the evaluation of the model's handling of missing data. Split rations from 90:10 to 10:90 were used, with a step size of 10. This approach was inspired by a similar method used in a previous study (Liu et al., 2021, p. 570–574) which used thee different split ratios to evaluate the performance of a deep learning model.

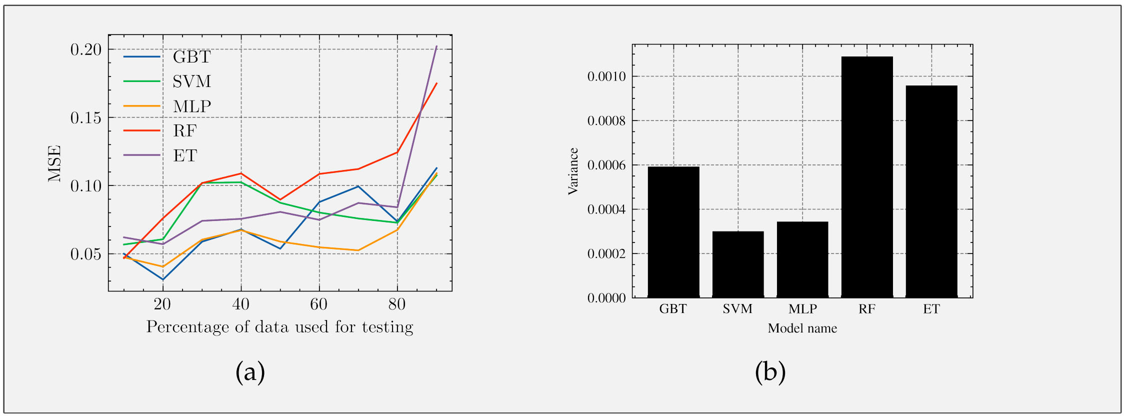
The results of this experiment provided valuable insights into the model's performance under different data compositions, particularly when dealing with missing data, and were used in conjunction with other metrics to gain a comprehensive understanding of the model's overall performance.

Figure **??** shows how the models performed when trained on continuously less data.

It can be seen that all models perform well when trained with on 90% of the data. However, when trained on 50% of the data, the MLP and GBT models start to perform better than the other models. When trained on 20% of the data and less all models significantly underperform. Also the ET model stays relatively stable when trained on less data.

Looking at the variances the the SVM and MLP models perform the and seem to be the most robust models when trained on less data. It has to be noted, that as seen in sub-figure (a) the most variances comes from training on below 20% of the data.

Overall the MLP model seems to be the most robust model, as it performs well on all data compositions and has the lowest variance.



(a)

(b)

Figure 5.2: Figure (a) shows the comparison of performance on less data. Figure (b) shows the variance of the models when trained on less data.

5.3.2 | Noise

A common practise in ML is to add noise to the training data to improve the model's generalization abilities. In the following section Gaussian Noise will be added to the data set to evaluate the robustness of the model.

There is no standard approach to adding noise to the data. In this thesis, the noise was added to the training data only.

To add the noise, *numpy.random.normal* function was used, which generate random numbers with a Gaussian distribution Pedregosa et al. (2011). The mean and standard deviation of each feature were calculated based on the original data, and the function was used to generate noisy values for each feature.

To study the model's reaction to increasing amounts of noise, the noise added to the training data was gradually increased starting from 1% to 50%, which means that the last iteration contains as many noisy as "clean" samples.

The difference between all RMSE between iterations was defined as loss. The total loss was calculated by averaging the difference of these values as shown in Equation 5.5.

$$\text{Average Loss} = \frac{1}{N}\sum_{i=1}^{N}|\text{RMSE}i - \text{RMSE}avg| \tag{5.5}$$

- *Does the calculation of the loss make sense?*

5.3.3 | Results

Model Name	rs	Noise	
		loa	
LR	-	0.070	0.251
.071	0.224		
Random Forest	x	0.241	
SVM	-	0.251	
MLP	-	0.451	

Table 5.4: Results of used machine learning models regarding the design principle robustness.

5.4 | DP4: Stability

Stability is defined as the ability of the model to generate repeatable results when trained on different data (Siebert et al., 2022, p. 16).

One appropriate way to measure the stability of a model is to use Leave-one-out Cross Validation (LOOCV), which is a a form of CV where one sample is used for validation and the remaining data is used for training (Gareth et al., 2013, p. 200–201).

In order to evaluate the model the LOOCV was repeated for all samples in the dataset resulting in a total of n iterations. The stability of the model was determined by calculating the average prediction error across all iterations, using the equation provided in Equation 5.6 which is taken from (Gareth et al., 2013, p. 201).

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i \tag{5.6}$$

The approach provides an unbiased estimate of the model's generalization error but this estimate is poor because only one samples is used for validation (Gareth et al., 2013, p. 201). Therefore, it can not be used to make a statement about the generalization error of the model. Since the goal is to measure repeatable results, the LOOCV is a suitable metric to measure the stability of the model.

A low $CV_{(n)}$ value indicates a high stability of the model, because the model is able to generate repeatable results when trained on different data. Note: The method can be very time-consuming, especially for large datasets, but it may provide more accurate estimates for small datasets.

5.4.1 | Results

Model Name	$CV_{(n)}$
Random Forest	0.053
Random Forest (rand. split)	0.053
Decision Tree	0.086
Decision Tree (rand. split)	0.062
Support Vector Machine	0.051
SVM (rand. split)	0.000

Table 5.5: Overview of the used machine learning models and their metrics.

Explanation of the results

5.5 | DP4: Interpretability

When following definitions of interpretability used in section **??**, it is clear that it is not possible to measure interpretability in a quantitative way. No mathematical formula can be used to measure interpretability, but other methods can be used to measure the interpretability of a model. Interpretable models allow the usage of global model-agnostic evaluation methods which will be used later in the evaluation of the models (see Section **??**).

In a first steps it has to be defined what makes a model interpretable.

Interpretable Models

According to Molnar (2020) one way to make a model interpretable is to limit the choice of algorithms to those that produce interpretable results. Example of such algorithms include linear regression, logistic regression and decision trees (Molnar, 2020, p. 35).

Molnar (2020) defines three properties of interpretable models:

Linearity A linear model is one in which the relationship between features and the target outcome is represented as a linear equation (Molnar, 2020).

Monotonicity A model with monotonicity constraints ensures that there is a consistent relationship between a feature and the target out come across the entire range of the feature. This can make it easier to understand the relationship.

Interactions Some models can automatically include interactions between features to improve prediction, while other require manual creation of interaction features. However, too many or complex interactions can make the model more difficult to interpret.

In the model selection for this thesis the focus is on interpretable models, as they are easier to understand and explain, therefore mostly models are chosen which fulfill all three properties.

Later in the thesis, the interpretability of the models is used to explain the results.

Number of Parameters

More complex models tend to have more parameters and are therefore more difficult to interpret. The number of parameters is calculated using the the `model .get_params` function in scikit-learn.

Note: The Depth of the model could be used as well for the evaluation, but not all models have a depth therefore this metric is not suitable. The depths is representing the number of layers in the model. A model with many layers, is generally more complex than a shallow network with fewer layers.

Notes

- *Space complecity -> How much memeory is needed?*

Linear Regression

Linear regression is a linear model as it models the relationship between the features and the target as a linear equation.

This linearity usually does mean that the relationship between the feature and the target goes in the same direction over the entire range of the feature. Therefore LR is monotonic.

LR models do not include interaction features by default. To capture interaction effects in LR, interaction features need to be created and added to the model manually. This was not done in this work.

5.5.1 | Results

Model Name	Linear	Monotone	Interaction	Param.
Linear Regression	Yes	Yes	No	4
Decision Tree	No	Some	Yes	11
Random Forest	No	Some	Yes	17
Logistic Regression	No	Yes	No	XX
SVM	X	X	XX	11

Table 5.6: Overview of the used machine learning models and their metrics.

Notes

- *Reasoning for results in table.*

Explanation of the results

5.6 | DP5: Resource utilization

To measure the resource utilization of the model, the following metrics are used:

Training time Measured in seconds. Refers to the time it takes to train the model. Training a model requires resources such as memory, CPU, and GPU, therefore the longer it takes to train a model, the more resources are required. According to resources utilization a shorter training time is desirable.

The training time is measured using the `time.time` function in python . The function returns the time in seconds since the epoch. The time is measured before and after the model is fitted. The difference between the two times is the training time.

Inference time Measured in milliseconds. It refers to the time it takes to make a prediction on data once it has been trained. It is an important measure not only for real-world application but also a faster runtime uses less resources and is therefore more efficient.

Inference time Measured in milliseconds. Time it takes to make 100 predictions. A model that takes longer to make predictions may be more complex than a model that is able to make predictions more quickly. Measured using the `time.time` function in python. 100 valuees are picked out of the teset set and the time is measured before and after the prediciton is made.

Memory space Measured in Mb. It refers to the amount of memeory required to run the model. The more storage space required to store the model, the more resources are required to store it. Therefore, a smaller storage space is desirable. To measure the memory usage the `memory_usage` function from the `memory_profiler` package is used.

Model Name	rs	Training time ms	Inference time ms	Memory Usage kb
LR	-	3.14	0.982	269.887
LR	x	3.558	1.157	269.887
DT	-	2.743	1.136	174.0
DT	-	3.455	1.212	174.008
RF	-	25.205	2.886	173.219
RF	x	26.821	2.269	173.266
ET	-	19.62	1.725	173.121
ET	x	20.854	1.887	173.156
SVM	-	309.721	2.789	195.734
SVM	x	310.018	2.793	195.746
MLP	-	19100.697	11.663	180.949
MLP	x	-	19100.697	11.663
180.949				

Table 5.7: Overview of the used machine learning models and their metrics.

As can be seen in Table 5.7, the ET is faster than to th conventional RF. This expected since the most-time intensive part of the RF is the splitting of the nodes with determining the optimal threshold for each feature at each node (see section 4.4.4.4). It has been observed, that this advantage is lost when the `n_estimators` parameter is set above 10.

The MLP is the slowest model to train and to make predictions. Most certainly this is because an MLP typically has more parameters to learn and has a deeper architecture, which means that it requires mor computations to train and predict. Also the training algorithm used is backpropagation, wich can be more computationally intensive. Also an MLP trains all data at once and therefore cant parallelize the process. It has to be noted, that tuning the hyperparameters is a time-consuming steps and the chosen parameters might not be the optimal ones.

5.7 | Results

Using the evaluation metrics described in the previous sections, the following results were achieved...

What are cruz et al. doing?

- They have two different data devision methods, DD1 and DD2.
- Also they use a validation set and a test set.
-

The results in Figure **??** show that the random forest

5.7.1 | Overall Results

This section will showcase the outcomes generated by the trained models in more detail and will also discuss the results in relation to the research questions.

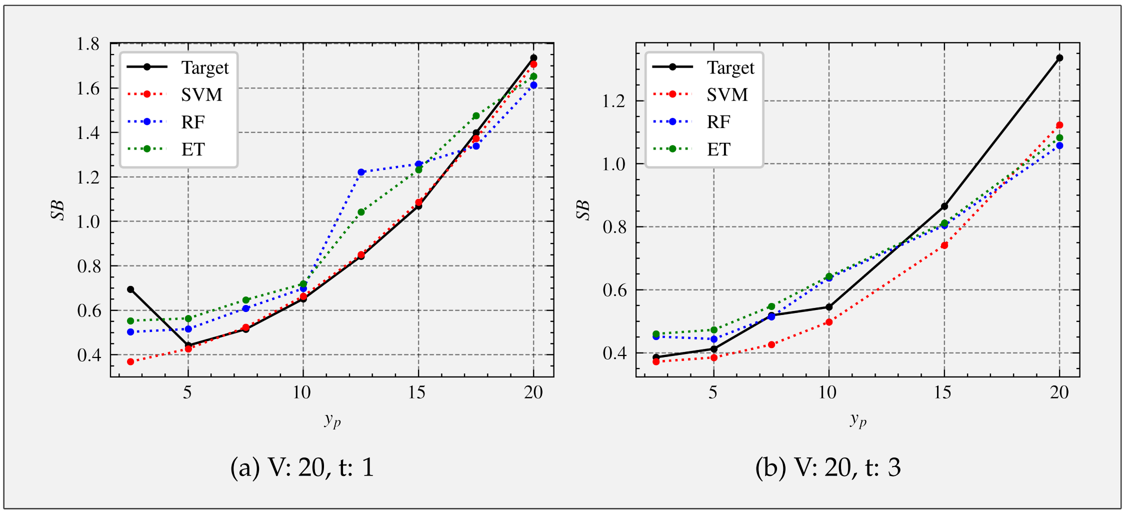
In Section **??** the overall correctness of the model was evaluated using different metrics. In order to evaluate the results of this problem in more detail, three test cases are used in this analysis. The V/t ratio is used to determine the bending conditions of the test cases. In the dataset used in ranged from 3.3 (V50and t3)to 100 (V50 t 0.5)and the test cases are chosen to represent the different bending conditions.

Recommended V/t ratios in industrial practices are between 6 and 10 (Cruz et al., 2021, p.7). Bending operations performed outside of this range of recommended ratios may result in high spring back.

Case A is chosen, so that is is below the recommended ration, case B is chosen to be within the recommended ratios and case C is chosen to be above the recommended ratios. As can be seen, case A features a significant spring back while the other cases do not.

In order to evalute the model's performance without bias, each test case where removed from the training data set so that the model has not seen the test case before. This results in a small but notable performance decrease in the test cases.

Case 5.3a with a V/t-ratio of 20 demonstrates the models' ability to predict the spring with high accuracy. However, the performance decreases in case b)5.3b with a V/t-ratio of 6.6, particularly when predicting the spring backs. It is likely that there is a measurement error at V = 7.5 in case b) 5.3b, causing some models to potentially predict the correct spring back.

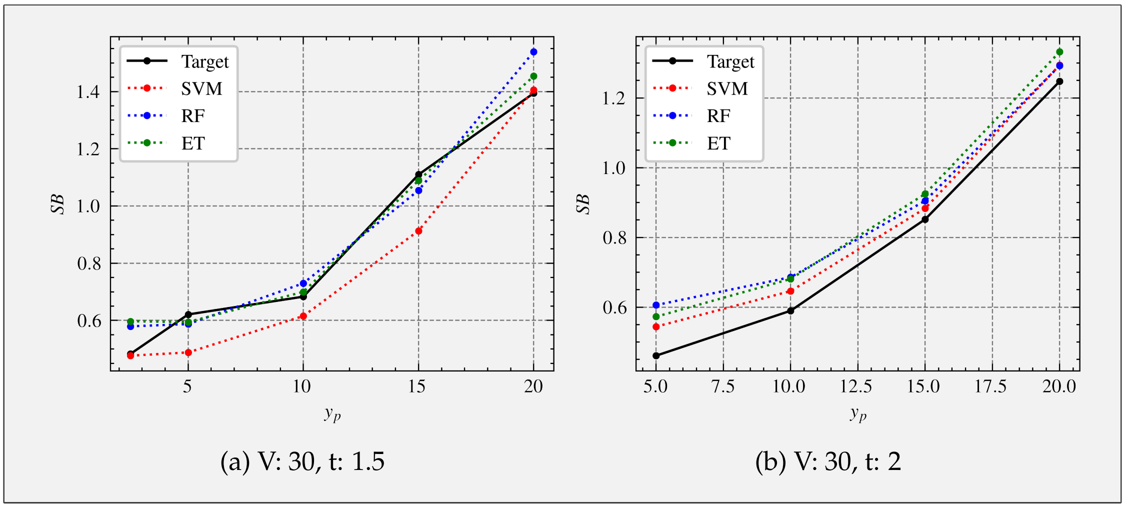


(a) V: 20, t: 1

(b) V: 20, t: 3

Figure 5.3: Performance plots for case A

As seen in Figure 5.4, the performance in case B is similar. The models exhibit high accuracy in predicting the spring back in case a) **??** with a V/t-ratio of 15. Additionally, the spring backs at the higher end are also predicted with high accuracy.



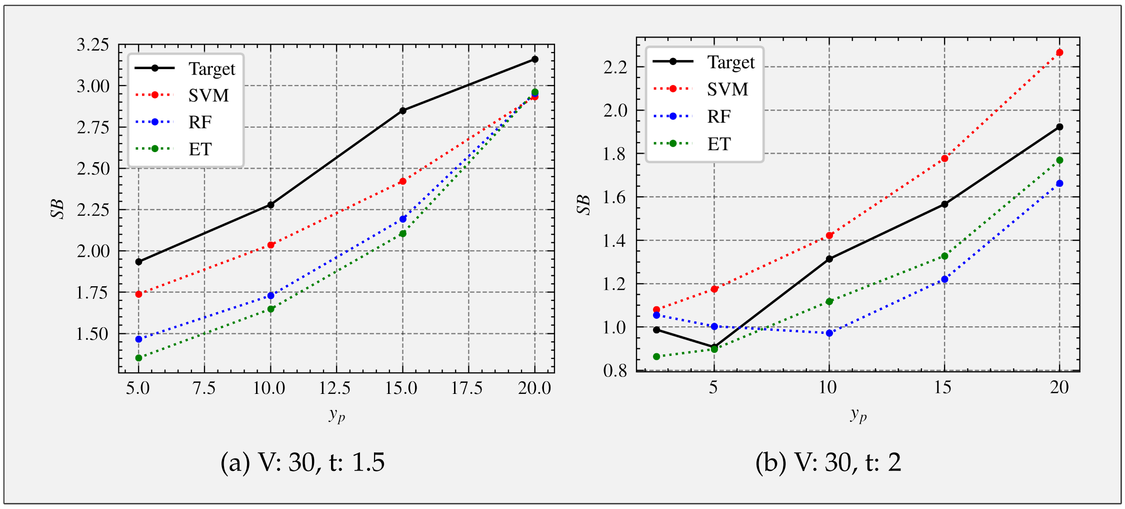
(a) V: 30, t: 1.5

(b) V: 30, t: 2

Figure 5.4: Performance plots for case B

In case C as seen in Figure 5.5 the models performance is getting worse.

Reasons for that? ...



(a) V: 30, t: 1.5

(b) V: 30, t: 2

Figure 5.5: Performance plots for case C

The best performing model is the SVM model with an RMSE of 0.2. Therefore the cases A, B and C are predicted using the SVM model.

As seen in Figure **??**, the model performs consistent across all thee scenarios, indicating that it can effectively handle a broad range of V/t ratios also outside the recommended industry guidelines range.

5.8 | Intepretation

5.8.1 | Model Specific Interpretability Methods

To attain interpretability, the simplest approach is to select algorithms that generate interpretable models. Some of these models like linear regression and decision trees where used in this work.

The following section looks at the generated and uses model specific interpretability methods to interpret the results.

5.8.1.1 | Feature Importance

5.8.1.2 | Linear Regression

The linearity of the relationship between the inputs make the mode interpretable as elaborated in section 4.4.1.1.

In a LR model, the significance of a feature can be quantified using the absolut value of its t-statistic. The t-statistic represents the scaling of the estimated weight with its standard error (Molnar, 2020, p. 40).

With 100 estimators in RF a MSE of 0.15 and RMSE 0.39 was achieved. Figure 5.6 compares and visualizes the relative importance of the features used for training the model. As shown, the thickness is the most important feature followed by distance and die open. The results show, that all three featured are relevant for the outcome and so no feature can be removed from the dataset to get a better performance of the model.

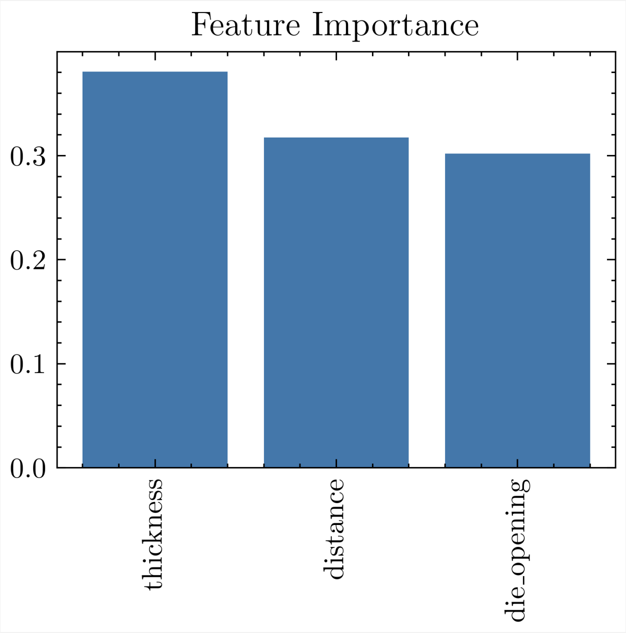


Figure 5.6: Relative feature importance

Also it is possible to create visual interpretations of the model using a weight plot or effect plot.

Weight and effect plot here

And also Lasso

Lasso plot here

5.8.1.3 | Decision Trees

Feature Importance Plot possible as well

OneR algorithm

Baysan Rule List

Global Model-Agnostic Methods

As already described in section 5.5, only algorithms that are interpretable where used for the evaluation.

Molnar (2020) differentiates between model-specific and model-agnostic interpretability methods. Model-specific methods are methods that are specific to a certain model type, for example, decision trees. Model-agnostic methods are methods that can be used with any model type, for example, permutation importance.

Also Molnar (2020) differentiates between global and local interpretability. Global interpretability refers to the ability to understand the overall behavior of the model, while local interpretability refers to the ability to understand the behavior of the model for a specific instance.

For the purpose of this thesis, the focus is on global model-agnostic methods, as they can be used with any model type and provide an overview of the model's behavior.

Partial Dependence Plots

5.8.2 | Local Model-Specific Methods

5.9 | Discussion

- *Data set had only limited features, but there are many more factors influencing the spring back*

6

Conclusions

6.1 | Limitations

- Only limited amount of process parameters.

6.2 | Bezug auf Stand der Vorschung

- Neues Qualitäsmodell für die Evaluatoin von Prozessdaten
- Echte Daten statt FEA
- Drei punkt biege verfahren

6.3 | Revisiting the Aims and Objectives

6.4 | Critique and Limitations

6.5 | Future Work

As seen in the results the MLP was one of the best performing models. This means further research could focus on deep learning models for the prediction of spring back.

- Use deep learning and more data
- Ensemble models could be used (dib et al.)
- More features like rolling direction of metal sheet and Spannung using drawing test

- More bending parameters like bend dedcuction

6.6 | Final Remarks

6.7 | TODO

- Stability section refinement
- Read Siebert paper and add important points to the thesis
- —
- Robustness refinement

References

Abdessalem, A. B. and El-Hami, A. (2015). A probabilistic approach for optimising hydroformed structures using local surrogate models to control failures. International Journal of Mechanical Sciences, 96:143–162.

Awad, M. and Khanna, R. Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers. The Expert's Voice in Machine Learning. Apress Open.

Baig, S. U. R., Wasif, M., Fatima, A., Baig, M. M. A., and Iqbal, S. A. (2021). Machine Learning for the Prediction of Springback in High Tensile Strength Steels after V-Bending Process Using Tree-Based Learning. Preprint, In Review.

Basili, V. R., Caldiera, G., and Rombach, H. D. (2002). THE GOAL QUESTION METRIC APPROACH. NA, page 10.

Biau, G. and Scornet, E. A random forest guided tour. 25(2):197–227.

Bishop, C. M. et al. (1995). Neural networks for pattern recognition. Oxford university press.

Bock, F. E., Aydin, R. C., Cyron, C. J., Huber, N., Kalidindi, S. R., and Klusemann, B. A Review of the Application of Machine Learning and Data Mining Approaches in Continuum Materials Mechanics. 6:110.

Breiman, L. Random Forests. 45(1):5–32.

Burkov, A. (2019). The hundred-page machine learning book, volume 1. Andriy Burkov Quebec City, QC, Canada.

Cao, J., Brinksmeier, E., Fu, M., Gao, R. X., Liang, B., Merklein, M., Schmidt, M., and Yanagimoto, J. Manufacturing of advanced smart tooling for metal forming. 68(2):605–628.

Cruz, D. J., Barbosa, M. R., Santos, A. D., Miranda, S. S., and Amaral, R. L. (2021). Application of Machine Learning to Bending Processes and Material Identification. Metals, 11(9):1418.

Dib, M. A., Oliveira, N. J., Marques, A. E., Oliveira, M. C., Fernandes, J. V., Ribeiro, B. M., and Prates, P. A. Single and ensemble classifiers for defect prediction in sheet metal forming under variability. 32(16):12335–12349.

Dridi, S. (2021). Supervised learning-a systematic literature review.

Gareth, J., Daniela, W., Trevor, H., and Robert, T. (2013). An introduction to statistical learning: with applications in R. Spinger.

Garrett, J. D. (2021). garrettj403/SciencePlots.

Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc.".

Gregor, S., Hevner, A. R., and University of South Florida (2013). Positioning and Presenting Design Science Research for Maximum Impact. MIS Quarterly, 37(2):337–355.

Gregor, S. and Jones, D. (2017). THE ANATOMY OF A DESIGN THEORY. College of Business and Economics The Australian National University, page 60.

Groover, M. P. Fundamentals of Modern Manufacturing: Materials, Processes, and Systems. John Wiley & Sons, Inc, seventh edition edition.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825):357–362.

Hevner, March, Park, and Ram (2004). Design Science in Information Systems Research. MIS Quarterly, 28(1):75.

Hevner, A. and Chatterjee, S. (2010). Design Science Research in Information Systems, volume 22, pages 9–22. Springer US, Boston, MA.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. Computing in Science & Engineering, 9(3):90–95.

Inamdar, M., Date, P. P., Narasimhan, K., Maiti, S. K., and Singh, U. P. Development of an Artificial Neural Network to Predict Springback in Air Vee Bending. 16(5):376–381.

Kazan, R., Fırat, M., and Tiryaki, A. E. Prediction of springback in wipe-bending process of sheet metal using neural network. page 6.

Kim, H., Nargundkar, N., and Altan, T. Prediction of Bend Allowance and Springback in Air Bending. 129(2):342–351.

Koppenhagen, N., Gaß, O., and Müller, B. (2012). Design Science Research in Action - Anatomy of Success Critical Activities for Rigor and Relevance. NA.

Lingbeek, R., Huetink, J., Ohnimus, S., Petzoldt, M., and Weiher, J. (2005). The development of a finite elements based springback compensation tool for sheet metal products. Journal of Materials Processing Technology, 169(1):115–125.

Liu, S., Shi, Z., Lin, J., and Li, Z. Reinforcement learning in free-form stamping of sheet-metals. 50:444–449.

Liu, S., Xia, Y., Shi, Z., Yu, H., Li, Z., and Lin, J. (2021). Deep learning in sheet metal bending with a novel theory-guided deep neural network. IEEE/CAA Journal of Automatica Sinica, 8(3):565–581.

Liu, X., Du, Y., Lu, X., and Zhao, S. Springback Prediction and Forming Accuracy Control of Micro W-bending Using Support Vector Machine. In 2019 6th International Conference on Frontiers of Industrial Engineering (ICFIE), pages 23–27. IEEE.

Liu, Y., Wang, Y., and Zhang, J. New Machine Learning Algorithm: Random Forest. In Liu, B., Ma, M., and Chang, J., editors, Information Computing and Applications, volume 7473 of Lecture Notes in Computer Science, pages 246–252. Springer Berlin Heidelberg.

McKinney, W. (2010). Data structures for statistical computing in python. In van der Walt, S. and Millman, J., editors, Proceedings of the 9th Python in Science Conference, pages 51 – 56.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. Artificial intelligence, 267:1–38.

Miranda, S. S., Barbosa, M. R., Santos, A. D., Pacheco, J. B., and Amaral, R. L. Forming and springback prediction in press brake air bending combining finite element analysis and neural networks. 53(8):584–601.

Molnar, C. (2020). Interpretable machine learning. Lulu. com.

Montesinos López, O. A., Montesinos López, A., and Crossa, J. Support Vector Machines and Support Vector Regression, pages 337–378. Springer International Publishing.

Müller, A. C. and Guido, S. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc, first edition edition.

Narayanasamy, R. and Padmanabhan, P. Comparison of regression and artificial neural network model for the prediction of springback during air bending process of interstitial free steel sheet. page 8.

Neal, B. On the Bias-Variance Tradeoff: Textbooks Need an Update.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3):45–77.

Sáez, J. A., Luengo, J., and Herrera, F. (2016). Evaluating the classifier behavior with noisy data considering performance and robustness: The equalized loss of accuracy measure. Neurocomputing, 176:26–35.

Sein, Henfridsson, Purao, Rossi, and Lindgren (2011). Action Design Research. MIS Quarterly, 35(1):37.

Shaik, A. B. and Srinivasan, S. A Brief Survey on Random Forest Ensembles in Classification Model. In Bhattacharyya, S., Hassanien, A. E., Gupta, D., Khanna, A., and Pan, I., editors, International Conference on Innovative Computing and Communications, volume 56 of Lecture Notes in Networks and Systems, pages 253–260. Springer Singapore.

Siebert, J., Joeckel, L., Heidrich, J., Trendowicz, A., Nakamichi, K., Ohashi, K., Namba, I., Yamamoto, R., and Aoyama, M. (2022). Construction of a quality model for machine learning systems. Software Quality Journal, 30(2):307–335.

Sonnenberg, C. and vom Brocke, J. (2012). Evaluation Patterns for Design Science Research Artefacts. In Helfert, M. and Donnellan, B., editors, Practical Aspects of Design Science, volume 286, pages 71–83. Springer Berlin Heidelberg, Berlin, Heidelberg.

Sáez, J. A., Luengo, J., and Herrera, F. Evaluating the classifier behavior with noisy data considering performance and robustness: The Equalized Loss of Accuracy measure. 176:26–35.

Taud, H. and Mas, J. (2018). Multilayer perceptron (mlp). Geomatic approaches for modeling land change scenarios, pages 451–455.

Terminology, I. (1990). Ieee standard glossary of software engineering terminology. IEEE Std 610.12-1990, pages 1–84.

von Rennenkampff, A., Nissen, V., and Stelzer, D. (2015). Management von IT-Agilität: Entwicklung eines Kennzahlensystems zur Messung Number Band 2 in Ilmenauer Schriften zur Wirtschaftsinformatik. Univ.- Verl. Ilmenau, Ilmenau.

Zhang, J. M., Harman, M., Ma, L., and Liu, Y. (2020). Machine learning testing: Survey, landscapes and horizons. IEEE Transactions on Software Engineering, 48(1):1–36.

Zheng, K., Politis, D. J., Wang, L., and Lin, J. A review on forming techniques for manufacturing lightweight complex—shaped aluminium panel components. 1(2):55–80.

Zhou, Z.-H. (2021). Machine Learning. Springer Singapore.