Hochschule
für Technik
Stuttgart

Machine Learning for the Prediction of Bending Parameters

some hyped-up tagline

Philipp Kurrle

Supervised by Prof. Dr. Ulrike Pado

Co-supervised by Peter Lange

Computer Science

Software Technology Master

Hochschule für Technik

January, 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

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.

1.1 | Problem Statement

"There are two major types of supervised machine learning problems, called classification and regression." (**?**, p. 34)

"For regression tasks, the goal is to predict a continuous number, or a floating-point number in programming terms (or real number in mathematical terms). Predicting a person's annual income from their education, their age, and where they live is an example of a regression task. When predicting income, the predicted value is an amount, and can be any number in a given range. Another example of a regression task is predicting the yield of a corn farm given attributes such as previous yields, weather, and number of employees working on the farm. The yield again can be an arbitrary number." (Müller and Guido, p. 34)

2

Theoretical Foundations

2.1 | 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) One aspect which makes the the process of sheet metal forming complex is that it is characterized by highly non-linear behavior due to large deformations of the metal sheet and is therefore hard to predict. As a result, traditional methods often rely on trail and error approaches, (Dib et al., p. 1), such as creating 'technology tables' which contain bending parameters and resulting spring back data. These tables are created by performing a lot of different experiments with different bending angles and metal sheets. This process is time and cost intensive and therefore often not suitable for the production of high-volume and low-cost components.

2.1.0.1 | Sheet Metal

Sheet metal is any form of metal that has a relatively large length to thickness ratio, their thicknesses are typically from 0.4 mm to 6 mm. Above is considered to be plate and below is considered to be foil. Sheet metal is usually manufactured by flat rolling. Low-carbon steel is the most commonly used type of sheet metal. Its low cost and has a good formability as well as its sufficient strength for most applications. (Groover, p. 405) Therefore it is used in this work as well. Sheet metals play a vital role in many industries. A significant number of consumer and industrial products such as automobiles contain sheet metal parts. Most sheet metal forming operations are performed on machines called presses and these operations are known as pressworking. These presses usually use tooling called punch and die or stamping die . Three main processes are categorize the sheet metal forming processes: cutting, bending and drawing, where bending is the the most common and only relevant process for this thesis. Bending and drawing are used to shape sheet metal parts into their required forms. (Groover, p. 405)

2.1.0.2 | 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.

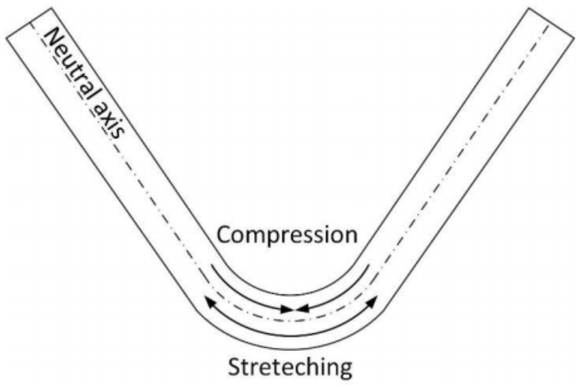

Figure 2.1: Bending plane, compression and stretching of sheet metal (Baig et al., 2021, p. 3)

2.1.0.3 | 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)

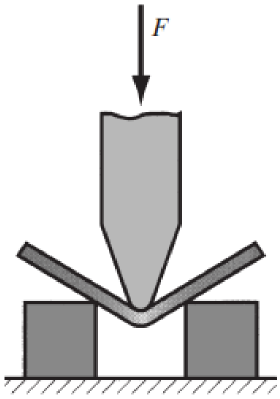

Figure 2.2: Air bending (Groover, p. 416)

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)

2.1.0.4 | 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.

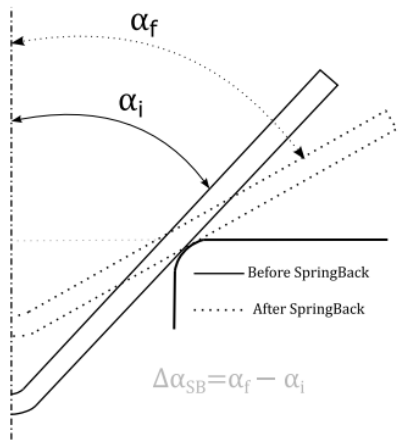

Figure 2.3: Spring back (Cruz et al., 2021, p. 5)

2.2 | 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)

2.2.1 | 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)

2.2.2 | 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.

2.2.3 | 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)

2.2.4 | 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)

2.2.5 | 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)

2.2.6 | Evaluations and Improvements of Models

2.2.6.1 | 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)

2.2.6.2 | 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)

2.3 | State of research

With the availability of data there has been an increased use of Machine Learning 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 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.

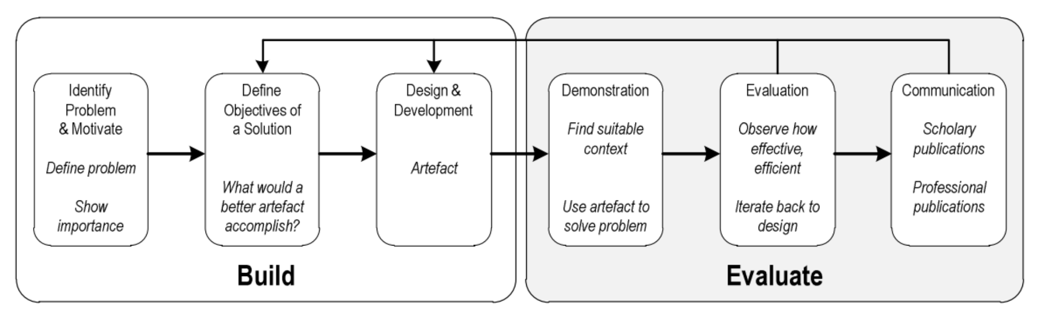

Figure 3.1: Design Science Research Approach according to Peffers et al.
Picture: (Sonnenberg and vom Brocke, 2012, p. 72)

Also, I'm not quite clear on the distinction between Demonstration and Evaluation in your context, since Demonstration would most likely be showing that your model predicts springback reliably, which refers back to evaluation?

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) not that they need to be adapted. They re-interpreted 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 artifacts in 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.

3.2.1 | 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.

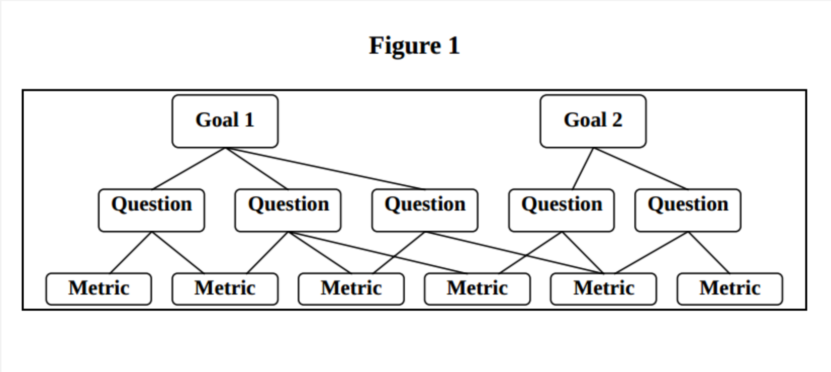

Figure 3.2: Goal-Question-Metric Ansatz (Basili et al., 2002, p. 3)

4

Build

4.1 | Design Principles

The following design principles is a selection of **?**'s quality parameters for ML models. In this Design Science Research (DSR) work the artifacts are ML models therefore these design principles are used to evaluate them.

Design Principle 1: Correctness

Does the artifact predict the spring back of a sheet metal with a high accuracy and correctness? With progression in manufacturing there is a growing demand for high-quality products, that means that the meta parts needs to be produced with high precision and accuracy. Here the sprin back is an undesired side effect which need to be mimimized. (Baig et al., 2021, p. 1) Sheet metal forming in manufacturing need a high level of quality and precision. Therefore, the spring back of a sheet metal is an important parameter to consider. (Cruz et al., 2021, p. 1) Predicting spring back is important to reduce the number of trial and error cycles in the manufacturing process. Also predicting spring back is complex because of many variables and parameters and often not all of them are known. Therefore, a machine learning model should predict the spring back of a sheet metal with a high accuracy and correctness. When using the ML model small errors in the prediction can cause fitting problems in the manufacturing process.

Design Principle 2: Appropriateness

Is the artifact appropriate for the given problem? While selecting a model it is important that it fits the problem/task and can deal with the given data. (**?**, p. 16)

Design Principle 3: Relevance

Does the artifact achieve a good bias-variance trade-off?

In addition to measure the correctness it is important to understand "why" the learner has this performance. This is important to understand the limitations of the model and to improve it. Therefore, it is important to understand the bias-variance trade-off. (Zhou, p. 50) Bias measures the differences between the learnesrs expected prediction and the ground-truth label. This results in the fitting ability of the learner. Variances measures the change of learning performance of the learner because of changes in the training set. This results in the impact of data disturbance on the results. (Zhou, p. 51)

Design Principle 4: Robustness

How well does the artifact handle outliers, noise and missing data? Using real-world data noice is a common problem and can have a negative impact on the performance of the learner. Therefore, it is important to measure how good the artifact performs when dealing with impact data. Sáez et al. proposed a new measure to establish the expected behavior of a learner with noisy data trying to minimize the problems: the Equalized Loss of Accuracy (ELA). (Sáez et al., p. 3)

Design Principle 5: Stability

Does the artifact generate repeatable results when trained on different datasets?

Design Principle 6: Interpretability

Is the artifact easy to understand and explain?

It should be noticed, that 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 model which allows conclusions how the results where generated is better.

Design Principle 7: Resource utilization

How many resources does the artifact need to train and predict?

Conventional processes are often based on empirical trial and error approaches. (Dib et al., p. 1) A common approach is to experimentally create so named 'technology tables' which contain the bending parameters and the resulting spring back. (Quelle: Hochstrate?) This process is time and cost intensive and therefore often not suitable for the production of high-volume and low-cost components. Therefore, one of the benefits of using machine learning should be the reduction of the number of trial and error cycles in the manufacturing process. Furthermore, training the model should take not too much time and resources. As mentioned before often FEM-simulation are used to virtually try out metal forming processes. However, fully exploring the design space is computationally expensive and often not possible. (Dib et al., p. 3) The number of experiments can be reduced using a meta-model like ANN. (Dib et al., p. 3) A approach fully based on ML should perfor

4.2 | Dataset generation

For the dataset generation, bending experiments were performed on metal sheets with different thicknesses. The material used is cold rolled steel sheets of the norm DIN EN 10130. The thicknesses used were 0.5mm, 1mmm and 2mm. The material was used because it is commonly used in bending processes and its high availability. In previous tests, it was observed, that the spring back are well observable with this material. Using this material, 200 single bending pieces of the dimension 20×100 mm have been cut. Each piece was bend one time using a *Zwick* three-point-bending 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.

4.2.1 | 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 metal-sheet. 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 (in *N*), while the displacement sensor measures the displacement of the punch (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 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 between the the two points between the sheet metal is placed. The parameters y_p is the punch penetration, which is the distance the punch is moved into the sheet. The parameter t is the thickness of the metal sheet while α is the corresponding angle after the bending process.. Parameter r_p is the punch radius which is the radius of the tip of the punch, which was never replaced in all experiments 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)

In order to get consistent results, a number of constant and variable parameters were chosen. The parameters include the punch-and-die tooling made of steel where die punch had a radius (r_p) of 5 mm and and die radius (r_m) of 10 mm. The die opening V was varied between 10 and 50 mm and the punch penetration y_p was varied between 0 and 20 mm.

The machine was configured to move the punch with a constant speed of 100 mm/min until it measured a resistance of 1 N. That meant, that the punch reached the metal plate and the actual bending process can start. After a hold time of 1 second the punch was moved with a slower speed of 8 mm/min until the specified punch penetration was reached. The length and width of the metal sheet was 100 mm and 20 mm respectively.

Using the *ZwickRöll TestXpert* software, the following parameter where set for the experiment, which are summarized in Table **??**. The punch of the machine was never replaced and therefore the radius of the punch remained 5 mm. The width and length of metal sheets where 100 mm and 20 mm respectively. The punch speed was set too 80 mm/min and the hold time was set to 1 second, latter is the time the punch stays at the end of the metal sheet and has to be a least 1 second which is a limitation of the machine. After the bending process, the punch was moved back to the starting position with a speed of 8 mm/min. This speed was chosen significantly lower than the speed used for the bending process in order to accuratly measure the spring back to avoid any damage to the metal sheet. The punch force threshold was set to 1 N, which means, that the punch starts bending the metal a soon as it touches the sheet metal.

The following parameters where varied in the experiments, which are summarized in Table **??**. The die opening V was varied between 10 and 50 mm and the punch penetration y_p was varied between 0 and 20 mm. The thickness of the metal sheet t was varied between 0.5 and 2 mm. Also the punch penetration y_p was varied between 0 and 20

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

mm.

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

4.2.2 | Measuring The Spring Back

The output data contained different data points, which were used to calculate the spring back. Important parameters for the calculation are the force, punch penetration and testing time. As shown in Figure 4.2 at the y_p maximum the punch penetration and the force are maximized as well. The punch stays at that position for 1 second and then moves back with a slower speed. This hold time a limitation of the machine and can not be changed. After the punch is moved back, the force is reduced and the punch penetration is reduced as well, until the punch is at the initial position. 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.

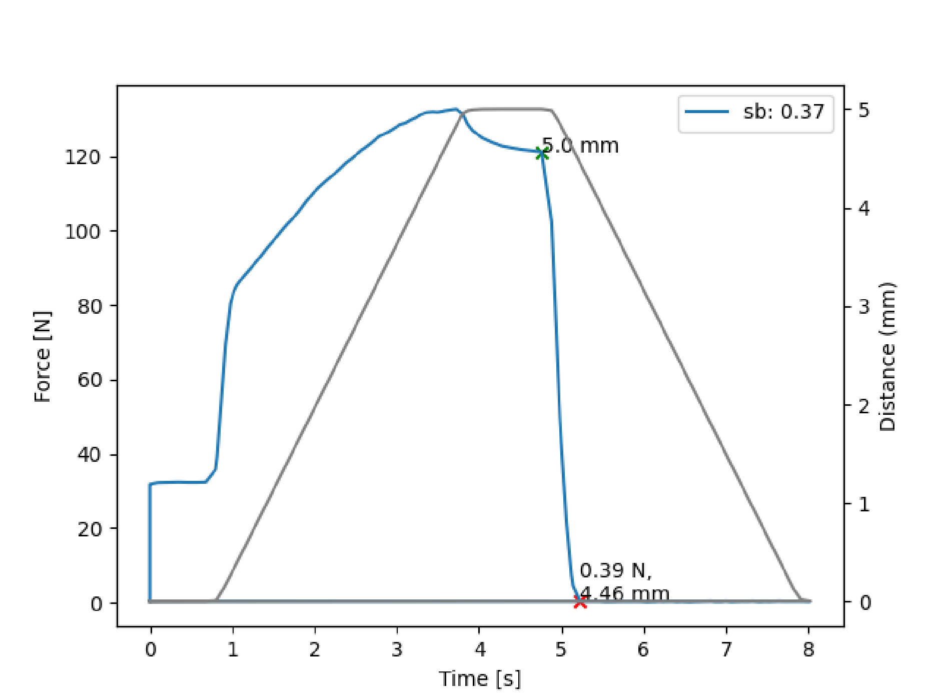

Figure 4.2: A steel metal sheet was bent with a punch penetration of 5 mm the spring back is 0 .37 mm. The blue line shows the force and the blue line shows the punch penetration.

Notes

- *Why is an initial force measured before the punch is moved?*
- *TODO: Add legend for grey line (punch penetration) and blue line (force))*
- *TODO: More dpi for the image*

4.2.3 | Dataset Exploration

4.2.3.1 | Features

The output data of the bending machine contained 26 features which can be found in the appendix. Out of these features only the standard power and the distance y_p and the force are relevant for calculating the spring back which was described in the last section 4.2.2. The final dataset therefore contained 3 features plus the added spring back. In total 396 data points where crated using the described approach. An example of the dataset is shown in Table **??**.

index	Distance	Spring Back	Thickness	Die Opening
1	5	0.6667	2.0	50
2	15	0.9164	2.0	50
3	10	0.6829	2.0	50
...	...	...	...	...
396	5	0.6667	3.0	10

Table 4.3: Varying parameters in the experimental setup

With 100 estimators in Random forest (RF) a mean squared error (MSE) of 0.15 and **RMSE!** (**RMSE!**) 0.39 was achieved. Figure 4.3 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.

4.2.4 | Visualizing The Data

Figure 4.4 shows the spring backs for the $V30$ dataset. In general, it can be seen that the spring backs is less for lower thicknesses and higher for higher thicknesses. Also, the spring backs for lower thickness tend to go up with increasing punch penetration, while the spring backs for higher thicknesses tend to go down with increasing punch penetration.

The factors for the behavior of the spring back can't be fully understood with the available data, which makes it a good case for machine learning.

Notes

- *TODO: Add better picture. Ordered by t, better colors, scienceplot, punch penetration instead of spring back*
- *Experimental setup methodology like Jonas.*

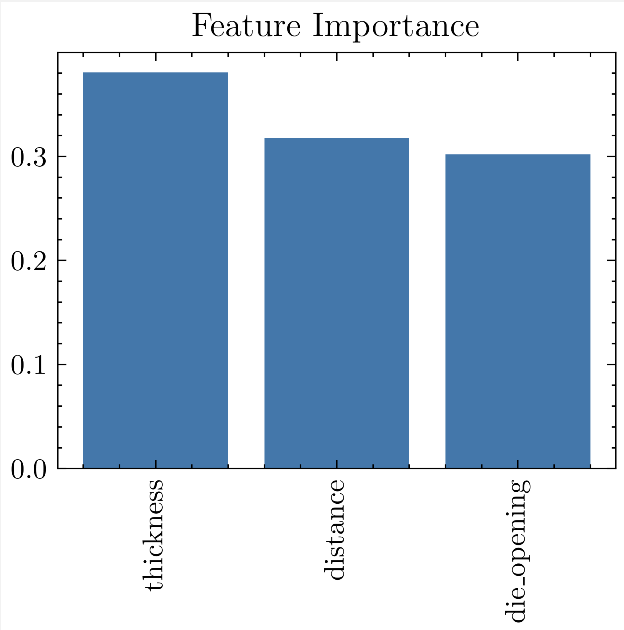

Figure 4.3: Relative feature importance

4.2.4.1 | Data Quality

The dataset was created in an experimental environment and the samples where carefully measured. Therefor the dataset does not contain many outliers and the data quality is high.

As shown Figure 4.6 data for all possible V and t combinations where collected. Also there the y_p values are evenly distributed and always range from 2.5 to 20 mm. Furthermore, the dataset was continuously extended with new data points throughout the project. During this process multiple outliers and wrong measurements where detected and removed.

It has to be noted that therefore the dataset does not model a real-world scenario, where the data quality is not as high as in the experimental environment. This has been considers in section 5.3 Robustness, where artificial noise is added to the dataset to test the robustness of the models.

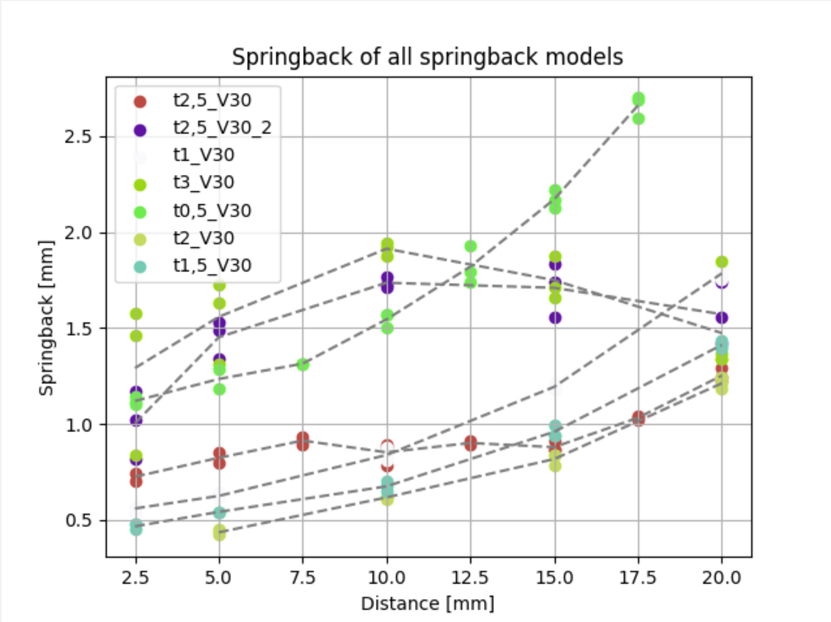

Figure 4.4: Springbacks for $V30$

Notes

- *The Figure feature importance was created using a random forest so it does not belong here. Find a*
- *different method to visualize the feature importance.*
- *Add example picture of spring back plots*
- *It is not yet decided if artificial noise will be added to the dataset.*
- *Add some data quality measure?*

4.2.5 | Data Preprocessing

The three independent features y_p, V and t as well as the dependend feature $spring\_back$ were normalized using the `MinMaxScaler`from the scikit-learn 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 min and max values of the test data would be used to scale the training data. How the data was split can be seen in Figure 4.6.

Because cross-validation is later used to tune and evaluate the models the data was split before the scaling. How the data was split can be seen in Figure 4.6.

Notes

- *TODO: Outlier handling is not done and mentioned. Is this necessary?*

- *TODO: Use more sophisticated scaling methods, not only min-max scaling. For example for V classfication*

4.2.6 | 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 **??**.

Library	Version
numpy	1.23.2
pandas	1.5.1
matplotlib	3.6.2

Table 4.4: Libraries used for the machine learning models.

Looking at the correlation matrix shown in Figure 4.5 it can be seen, that the distance and the spring back are more correlated than the other features. This is expected, because the punch penetration y_p is the main factor which influences the spring back. The other features are not correlated with each other, no multicollinearity is present which is good for the machine learning models.

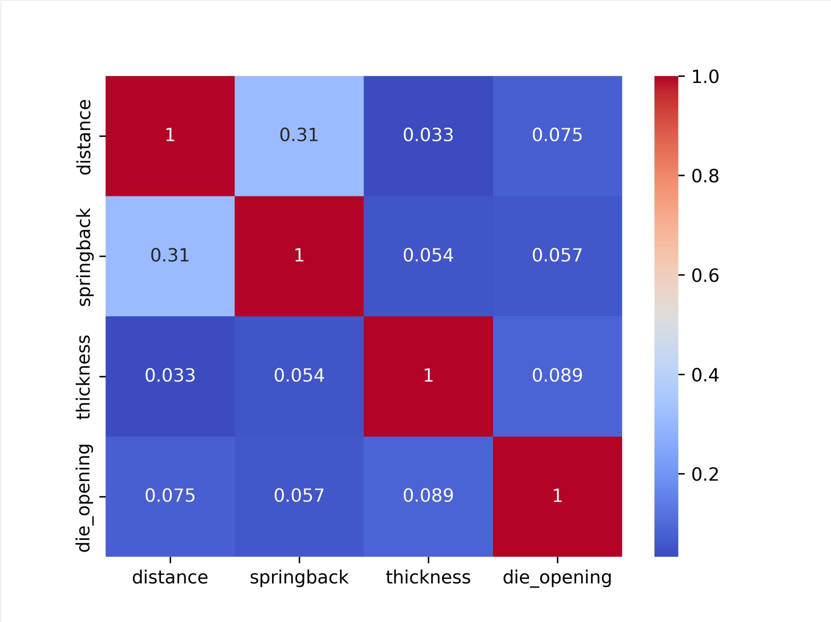

Figure 4.5: Correlation matrix

4.3 | Model Selection

4.3.1 | Support Vector Regression (SVR)

Support Vector Machines SVM are usually used for classification problems. The SVM algorithm is used to find a hyperplane in an N-dimensional space (N - the number of features) that distinctly classifies the data points. (Awad and Khanna, p. 42) Predicting the spring back is a regression problem, so the SVM algorithm need to be generalized of classification problems, where the model returns continuous values instead of finite set of values. This is done by using the Support Vector Regression (SVR) algorithm, which is inspired by SVM and uses the same principles. It fits a model to data using only residuals smaller in absolute value than a certain constant called ϵ-sensitivity

This is done by creating a "tube" of ϵ width around the data, with points inside the tube not being penalized and points outside the tube being penalized based on their distance from the predicted function. This is done similar to how SVMs penalize points in classification. Like SVM, SVR fins a well-fitting hyperplane to a kernel-induced feature space to achieve good generalization performance using the original features. (Montesinos López et al., p. 369)

Kernel Trick

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) In practise the mathematical details behind these kernels are not important, but it is important to know that they are used to transform the data into a higher dimensional space, where the data is linearly separable.

4.3.2 | Multi Linear Regression

4.3.3 | Polynomial Regression

4.3.4 | Decision Tree Regression

4.3.5 | Random Forest Regression

A commonly used method in machine learning. The goal is to solve classification or regression problems by predicting the value of a output variable by one or multiple input variables. (Shaik and Srinivasan, p. 253) To build a Decision tree (DT) the source dataset represents the root node of the tree this data set is split into leafs (children) by using a set of spitting rules until each leaf in the DT is "pure" and only contains one target value. Depending on the use cases this is a single class or a single regression value. (Müller and Guido, p. 70-72) The main drawbback of DTs is the tendency to overfit and poor generalization performance, what makes them not paticaly for most use cases. Therefore usualy ensemble methods are used instead of a single DT. (Müller and Guido, p. 78) (Liu et al., c, p. 251) Random forest (Breiman) is a type of ensemble learning algorithm in which multiple decision trees, which are "weak learners," are trained and combined to produce a more accurate and stable prediction, known as a "strong learner." (Awad and Khanna, p. 24) The risk of overfitting is mitigated by subset and feature randomization. Each root node uses a unique subset of the data and each leaf is split using a random 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) In this supervised learning method, a "divide and conquer" approach is used. This involves dividing the data into smaller samples, incrementally building a randomized tree predictor for each sample, and then combining (aggregating) these predictors together. This approach has proven to be effective. Because not only one but multiple classifiers are used the random forest learning is known as ensemble model. (Shaik and Srinivasan, p. 254)

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. This 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)

4.3.5.1 | 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 random forest algorithm gradient boosting combines multiple weak learners to create a strong learner. The difference to a random forest 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.

Results

4.4 | Model Training

4.4.1 | Training-Test Split

Figure 4.6 shows the data set and which parts of it is used for training and testing the used models. Samples with a die opening of 30 are used to test the performance and the remaining part is used for training. A different approach would be to us a random test and train split, this would lead to a better performance of the models but would not evaluate their ability to predict new data of a different die opening. The die opening 30 was chosen because it is in the middle of the selected data set and therefore the models should be able to predict the data of this die opening. All models are trained with the same data set and the same parameters. The only difference is the used algorithm.

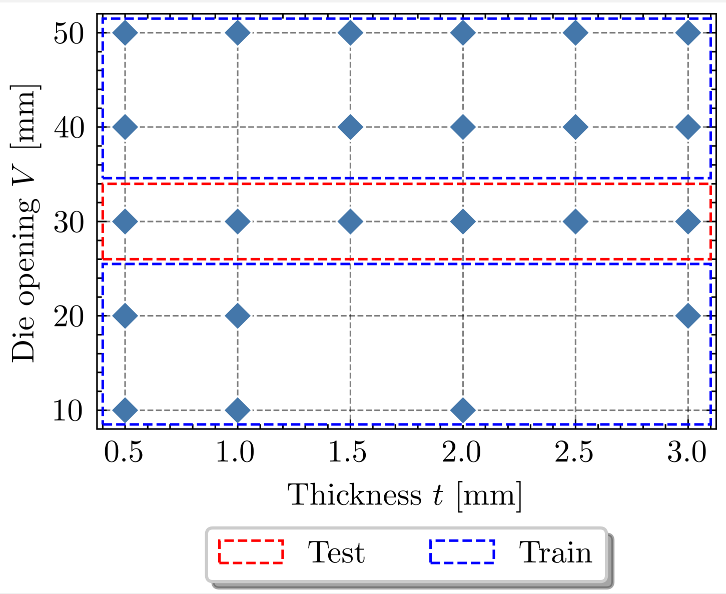

Figure 4.6: The training and test dataset. (Data especially for V40 is still missing)

Additional to the data split shown in Figure 4.6 a random test split was used to train the models. The random 80/20 split was also used to evaluate the performance of the models and to compare them to each other.

The reason for choosing to different methods for the data split is that the first one is used to evaluate the models' ability to generalize on new data. The second one is used to evaluate the models' performance on the data set. Also in real-world applications is possible that there is already data for all needed $V-t$-combinations and models could be trained on that data.

It is expected that the models perform better on the random data but the information gain about the models' ability to generalize on new data is less because the random split will most certainly contain data of all $V - t$-combinations.

Notes

- ***Dataset not yet complete**, data for all missing Vt combinations will be added*
- *V10 does differ very much and it might be not good to include in the data set. Maybe add V60 instead*

4.4.2 | Random Forest

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 4.5.

The *criterion* hyperparameter was set to *absolute_error* because the absolute error is the metric used to evaluate the model.

The *n_estimators* where set to 10 using grid search cross validation, because the model should not be too complex and the number of trees should not be too high.

The *max_depth* was set to 10 using grid search cross validation. The default unpruned model did overfit the training data and was not able to generalize on the new data well. This is expected behavior of decision trees which is descriped by (Müller and Guido, p. 133-136) amongst others.

The *min_samples_split* was set to 2 using grid search cross validation. The default value of 2 was chosen because it is the default value of the random forest model in Scikit-Learn.

The *min_samples_leaf* was set to 1 using grid search cross validation. The default value of 1 was chosen because it is the default value of the random forest model in Scikit-Learn.

The *max_features* was set to *auto* and therefor the models does use all avialble featues. As described in Figure 4.3 only limited featues are avaialbe and all of them are important for the dependend variable spring back

Table 4.5: Hyperparameters of the random forest model.

Hyperparameter	Value	Description
n_estimators	10	The number of trees in the forest.
criterion	absolute_error	The function to measure the quality of a split.
max_depth	30	The maximum depth of the tree.
min_samples_split	4	The minimum number of samples required to split an internal node.
min_samples_leaf	2	The minimum number of samples required to be at a leaf node.
max_features	auto	The number of features to consider when looking for the best split.
max_leaf_nodes	X	Grow trees with max_leaf_nodes in best-first fashion.
max_leaf_nodes	X	Grow trees with max_leaf_nodes in best-first fashion.

Notes

- ~~With a test test split (V30) the models does not achieve a good R2 score. That means that the model is not able to generalize well to new data (0.0something).~~
- ~~When using a random test train split the R2 score is much better (0.9something).~~
- ~~Look again at spring back calculation~~

4.4.3 | Gradient Boosted Regression Trees

Hyperparameter Tuning

"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)

4.4.4 | Support Vector Machine

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.

Notes

- *Paraphrase parameters better*

Hyperparameter	Value	Description
kernel	rbf	Kernel type used in the algorithm.
degree	1	Degree of polynomial kernel function.
gamma	0.1	Kernel coefficient for rbf, poly and sigmoid kernels.
C	4000	"Regularization parameter. The strength of the regularization is inversely proportional to C"
epsilon	0.001	"Epsilon in the epsilon-SVR model."

Table 4.6: Hyperparameters of the Suport Vector Regressor.

4.5 | Structure of The Code

- *The code is available on GitHub:* `www.github.com/...`
- *TODO: Short explanation what to do to reproduce the results.*

5

Evaluation

This chapter critically examines the machine learning models conceived and partially implemented in the previous chapter. Table **??** shows an overview of the evaluated criteria. These are structured according to the Goal-Question-Metric approach.

The quality metrics are oriented on the quality model from (Siebert et al., 2022), which is based on the ISO/IEC 9126 standard for the evaluation of software quality. The standard was changed to fit the requirements of machine learning models Siebert et al. (2022).

The metrics of the goal *Robustness* where original only defined for classification models and therefore needed to be changed. *TODO: Explain what has been changed*

Goal	Question	Metric
Appropriateness	How well does the model type fit the current task?	Prerequisites for model type
Correctness	Ability of the model to perform the current task measured on the development dataset and the runtime dataset	Precision, Recall, F-score
Relevance	Does the model achieve a good bias-variance tradeoff? Which means neither overfitting or unterfitting the data.	Variance of cross-validation and fit
Robustness	Ability of the model to outliers, noise and other data quality issues	Equalized Loss of Accuracy (ELA)
Stability	Does the artifact generate repeatable results when trained on different data?	Leave-one-out-cross validation stability
Interpretability	How well can the model be explained?	Complexity measures (e.g., no. of parameters, depth)
Resource utilization	How much resources are required to train and run the model?	Training time, runtime, storage space

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

Notes

- *TODO: Improve layout of table*

5.1 | DP1: Appropriateness

5.2 | DP2: Correctness

The model must be able to perform well on the selected task. To measure the correctness of the model, the metrics MAE, MSE and RMSE are used. In the formulas 5.1, 5.2 and 5.3 the variable e_i is the prediction error which is the difference between the predicted value by the model the actual value. y_i is the actual value and n is the number of samples in the testing data set.

The mean absolute error (MAE) and mean squared error (MSE) are the most commonly used metrics for evaluating the performance of regression models.

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. Additionally the R^2 was added to the overview. The R^2 is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. The value of the R^2 is the percentage of the response variable variation that is explained by a linear model.

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.

5.2.1 | Results

Model Name	MAE	MSE	RMSE
Random Forest	0.16	0.04	0.21
Random Forest (rand.split)	0.14	0.05	0.22
Support Vector Machine	0.09	0.04	0.20
SVM (rand. split)	0.09	0.04	0.20

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

Explanation of results...

5.3 | 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 perfoms 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 because this is usually the metric used to evaluate the regression models.

Figure 5.1: Variance of cross-validation

For completeness the R^2 score is also shown int the table. The R^2 is a statistical measure of how close the data are to the fitted regression line. 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. Therefore, a high R^2 indicated a good model fir and good bias-variance tradeoff. (Müller and Guido, p. 43)

R^2

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

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.

Model Name	Variance of CV	R^2
Random Forest	0.034	0.771
Random Forest (rand. split)	0.034	0.771
Support Vector Machine	0.008	0.851
SVM (rand. split)	0.008	0.851

Table 5.3: Overview of the used machine learning models and their metrics.

5.4 | Robustness

According to Sáez et al. robustness "is 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). In this context the metric Equalized Loss of Accuracy (Equalized Loss of Accuracy (ELA)) was established to measure the robustness of different machine learning models. But the ELA is only usable for classification models, therefore a different metric has to be found.

There are a number of ways to evaluate the performance of regression models, including mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE). You can use these measures to compare the performance of different regression models on a given task. Also the R^2 can be used to measure the robustness of the model.

As mentioned in section 4.2.2 dataset exploration the dataset has a high quality with no outliers and no missing values. Therefore, these data quality issues will be added to the dataset to measure the robustness of the model.

5.4.1 | Missing values

When applying the model to real-world data, there is a chance that some values are missing. Two possible scenarios are possible:

1. **Missing Vt-combinations**: There is no data available about a metal with a certain thickness and die opening. Reasons could be that the metal is not used in the industry or the data was never measured.
2. **Missing values**: There is data for the Vt combination but some values are missing because of general data quality issues.
3. **Outliers**: There are values that are not in the range of the dataset. Reasons could be that the data was measured incorrectly.

To measure all three of these scenarios, three different tests where created. The first test measures the robustness of the model.

1. Missing Vt-combinations

As can be seen Figure 4.6 of the test and training dataset, all possible Vt combinations in the set parameter range where measured. This is not the case in real-world data. Therefore, the robustness of the model has to be measured with missing Vt combinations.

Therefore, certain Vt combinations were removed from the training dataset. The removed Vt combinations were chosen randomly. The number of removed Vt combinations was 10% of the total number of Vt combinations in the training dataset. Subsequently the model was trained on the training dataset with the removed data and evaluated on the test dataset. The performance of the mode is measures against the performance of the model trained with all data.

Calculation of the loss of accuracy

The process described above was done 100 times, to mitigate the effect of the random selection of the Vt combinations. The calculation of the loss of accuracy is follows Equation 5.5, the metric is called Mean Loss Of Accuracy (Mean Loss of Accuracy (MLA)).

Mean Loss Of Accuracy

$$\text{MLA} = \frac{1}{100}\sum_{i=1}^{100}(\text{MSE}_{all} - \text{MSE}_{missing}) \tag{5.5}$$

To calculate the loss off accuracy the models trained with missing Vt combinations were compared to the model trained with all data. The difference between the MSE of the models was calculated. The total loss of accuracy was calculated by averaging the difference of the MSE of the models.

2. Missing values

3. Outliers and wrong data

To measure the robustness of the model, outliers were added to the data. The outliers were added to the data in the following way:

Outliers where added to the training dataset. The model was trained on the training dataset. The model was evaluated on the test dataset. The outliers where removed from the training dataset. The process was repeated 100 times.

Noise

Not sure if I want to measure that.

Note: This process of measuring the robustness of the model differs from the approach used by **?** and Sáez et al..

5.4.2 | Results

Model Name	Missing Vt-combinations loa	Missing Values loa	Outliers loa
Random Forest	0.034	0.000	0.771
Support Vector Machine	0.000	0.000	0.000

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

Notes

- *An alternative metric instead of ELA could be the approach of Scher and Trügler . But it would be necessary to implement the metric in Python.*

- *Decide if I want to evaluate noise as well or if the rest is enough.*

5.5 | 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).

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

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.

5.5.1 | Results

Model Name	$CV_{(n)}$
Random Forest	0.053
Random Forest (rand. split)	0.053
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

5.6 | Interpretability

Interpretability refers to the ease with which humans can understand and make sense of the decisions made by a trained machine learning model (Siebert et al., 2022, p. 16). Miller (2019) define interpretability as the "degree to which a human can understand the cause of a decision" (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.

When following the above definitions of interpretability, 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.

Big-O Notation

Time complexity is a common way of measuring how fast or slow an algorith will perform for the input size (Source). Usually the complextiy is measured with the big-O notation. ...

The term complexity is an overloaded term, normally complexity is measured with the big-O notation and measures the scale in time as the number if inputs increases (Source). However, in the context of machine learning, complexity is often used to refer to the number of parameters in a model.

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?*
- *Does using the big O notation make sense?*

5.6.1 | Results

Model Name	Linear	Monotone	Interaction	big-O	Parameters
Decision Trees	No	Some	Yes	XX	XX
Logistic Regression	No	Yes	No	XX	XX
SVM	X	X	XX	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

5.7 | 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.

Runtime 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.

Runtime = Inference time?

Inference time Measured in milliseconds. Time it takes to make a prediction. A model that takes longer to make predictions may be more complex than a model that is able to make predictions more quickly.

The runtime is measured using the `time.time` function in python. On value is picked out of the test set and the time is measured before and after the prediction is made. The difference between the two times is the runtime.

Storage space Measured in kilobytes. It refers to the amount of storage space required to store 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.

Model Name	Training time s	Runtime ms	Storage space kb
RF	0.0222	2.0275	80.2
LR	0.000	0.000	0.000
SVM	0.000	0.000	0.000

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

5.8 | Interpretation Of Results

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

Global Model-Agnostic Methods

As already described in section 5.6, 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

Evaluation SVM (Only Notes)

(Müller and Guido, p. 104)

6

Conclusions

6.1 | Revisiting the Aims and Objectives

6.2 | Critique and Limitations

6.3 | Future Work

6.4 | Final Remarks

6.5 | TODO

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

References

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.

Bauldry, S. (2015). Structural equation modeling.

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

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.

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.

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

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.

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.

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.

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

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.

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.

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.

Schwarz, G. (1978). Estimating the dimension of a model. The annals of statistics, pages 461–464.

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.

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

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. Machine Learning. Springer Singapore.