Hochschule
für Technik
Stuttgart

Predicting Spring Back in Sheet Metal Forming

A Comparative Study of Machine Learning Models and Evaluation of Interpretability for Enhanced Process Insights

Philipp Kurrle

Supervised by Prof. Dr. Ulrike Pado

Co-supervised by M. Sc. Peter Lange

Faculty of Measurement, Computer Science and Mathematics

Master's Programme in Software Technology

Hochschule für Technik

May, 2023

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

Abstract

Sheet metal bending is a widely ued manufacturing process in various industries such as automotove, aerospace and construction. One of the challenges in this process is the accurate prediction and compensation of spring back, which occurs due the elastic recovery of the material after bending. Inaccurate predictions of spring back can lear do fitting issued, increased costs and reduced product quality. This study aims to research the potential of Machine Learning (ML) models to predict the sprin back in sheet metal bending using real-world air-bending data.

Three hypotheses were formulated and tested in this study. First, it was hypothesized that ML models can accurately predict spring back, outperforming traditional trial-and -error methods. Second, it was hypothesised that specific ML models or combinations of models would yield better performance based on six Design Principles. Third, it was hypothesized that ML models with high interpretability can provide valuable insights into the factors contributing to spring back.

The results of this study support all three hypotheses, demonstrating the effectiveness of ML models in predicting spring back in sheet metal bending. The Extra Trees and Support Vector Machine models achieved the best performance with a root mean squared error (RMSE) of 0.15 mm. The findings also highlight the importance of selecting suitable ML models ofr specific applications and the valuable insights that can be gained from models with high interpretability.

In conclusion, this study demonstrates the potential of ML models to accurately predict and compensate for spring back in sheet metal forming processes, leading to improved product quality and reduced manufacturing costs. The findings have implications for both the understanding of spring back phenomena and the development of more effective compensation strategies in the sheet metal forming industry.

Contents

List of Figures

List of Tables

List of Abbreviations

1

Introduction

The process of sheet metal forming has played a crucial role in the manufacturing sector for many years, resulting in a wide variety of products that serve various purposes. Sheet metal bending and stamping are the preferred techniques for creating complicated parts and components for everything from cars to airplanes (**?**, p. 1) (**?**, p.1). However, as the need for more efficiency and lower emissions develops, so does the need to optimize these processes. (**?**, p. 4).

1.1 | Problem Identification and Motivation

To meet the increased need for lightweight metal parts, substantial research on sheet metal forming methods has been done during the last several decades. One common challenge in the field is the phenomenon of spring back, which is the tendency of a sheet metal to return to its original shape after being bent. This distortion has the potential to drastically disrupt the entire production process, resulting in more try and error cycles. The complexity of sheet metal forming originates from the nonlinear behavior generated by metal sheet deformations, making spring back prediction particularly difficult (**?**, p. 1) (**?**, p. 1). Consequently, researchers have explored various compensation methods, including the use of Machine Learning (ML) models (**?**, p.1).

Traditional approaches often rely on trial and error techniques (**?**, p. 1). Expert consultations in the field of sheet metal forming revealed that a common trial and error approach involves the creation of "technology tables" which contain bending parameters and resulting spring back data (see appendix **??**). These tables are generated through numerous experiments with varying bending angles and metal sheets. However, this method is both time-consuming and costly, making it ill-suited for producing high-volume and low-cost components (personal communication, Dr. Wolfram Hochstrate and M. Sc. Peter Lange).

These challenges underscore the need for an accurate ML model to predict spring back. Current research predominantly focuses on applying ML to datasets generated via simulation using the Finite Element Method (FEM)and other bending techniques (see section **??** 'State of Research'). However, there has been limited attention given to the utilization of real-world data and the air-bending method. This study aims to fill these gaps by exclusively using air-bending to generate a comprehensive dataset, thus enhancing the relevance and motivation for applying ML to real-world scenarios.

The primary goal of this project is to accurately predict spring back and develop **ML!** (**ML!**) models that satisfy crucial quality attributes also called Design Principles (DPs). To evaluate the performance of the ML models, six key DPs have been identified: Correctness, relevance, robustness, stability, resource utilization, and interpretability (see **??**). The objective is to create models that excel in each of these areas, ensuring the predictions are as precise and reliable as possible.

This problem identification leads to the following research questions:

Hypothesis 1: Machine Learning (ML) models can accurately predict spring back in sheet metal forming using real -world air-bending data, outperforming traditional trial-and-error methods.
Hypothesis 2: Specific ML models or combinations of models (e.g., ensemble methods) will yield better performance in terms of the six Design Principles (DPs) compared to other models.
Hypothesis 3: ML models with high interpretability can provide valuable insights into the factors that contribute to spring back, potentially leading to better understanding of the underlying physical processes and more effective compensation strategies.

1.2 | Research Method and Structure

The research methodology employed in this thesis is Design Science Research (DSR), which is a research approach that emphasizes the development and assessment of innovative solutions to real- world problems. DSR aims to generate new knowledge by creating, implementing, and evaluating new artifacts or solutions.

It is worth taking a look at figure **??** up front to get a understanding on how this study is structured. Some changes have been made to the DSR process which are reasoned in the chapter **??**. The first activity 'Identify Problem & 'Motivation' is already covered in this introduction and the order of the activities 4 and 5 where changed to suit the structure of this study.

The thesis follows a two-phase process of “build” and “evaluate” within the DSR approach. The research methodology chapter (**??**) elaborates on how this approach is utilized throughout the thesis.

2

Theoretical Foundations

2.1 | Sheet Metal

Sheet metal is typically produced through the process of flat rolling various types of metals, resulting in a characteristic aspect ratio of surface area thickness that typically falls within the range of 0.4 mm to 6 mm. When the value exceeds this particular range, it is classified as a "plate", whereas if it falls below the range, it is classified as a "foil" (**?**, p. 405). Due to its accessibility, good formability, and sufficient strength for the majority of uses, low-carbon steel is a regularly used type of sheet metal (**?**, p. 405). Sheet metals play a vital role in many industries for example automotive and aerospace to reduce the weight of products (**?**, p. 1).

2.1.1 | Sheet Metal Manufacturing

The process of working with sheet metal is known as sheet-metal press-working and is often done with machine tools called presses. Utilizing so called "punch-and-die tooling" that is specifically made for these procedures, stamping presses are employed to complete these processes (**?**, p. 405). The punch is a shaped tool that is usually mounted as upper part on the press ram and applies pressure on the sheet metal to shape it. The die is a stationary tool that is mounted in the press bed and provides a support surface for the sheet metal and also defines the shape of the finished product **??** (**?**, p. 412). **??** shows the tooling setup together with a sheet metal being bend. While the term 'punch' is quite intuitive, the term "die" can cause confusion. The term 'die' is a typically term in metal working to refer to the lower or stationary part of a tooling setup.

Cutting, bending, and drawing are the three main groups of sheet metal forming processes. While bending and drawing processes are used to mold sheet metal into the desired forms necessary for making certain parts, cutting is used to separate huge sheets of material into smaller pieces. (**?**, p. 405). Drawing involves stretching the metal to create a convex or concave shape (**?**, p. 416), while bending is "defined as the straining of the metal around a straight axis" (**?**, p. 412). This study focuses on metal bending and the bending method used in this study (Air Bending) deploys a mixture of bending and drawing and will be explained in more detail in the next section (**?**, pp. 416).

2.2 | Sheet Metal Bending

Bending is a forming operation that is used to change the shape of a sheet metal by applying a load to it. It involves using force to shape the sheet metal into a desired form (**?**, p. 1). The metal is subjected to a load that is greater than its yield strength but less than its ultimate tensile strength , allowing the metal to be permanently deformed into a different shape. (**?**, p. 1). This means that a load below the yield strength will not deform the metal, while a load above the utlimate tensile strength will cause the metal to break.

Sheet metal bending is usually used to produce large quantities of components at low cost in various industries (**?**, p. 1). The metal sheet experiences plastic deformation during the bending process, when the outside fibers are stretched and the inside fibers are compressed. As a result, the sheet metal curves in the direction of the applied load. (**?**, pp. 1–3). The differently loaded cross-sectional areas are separated by a neutral plane in which theoretically no load occurs. If only the loaded cross-section is considered for simplification, the neutral plane is also referred to as the neutral axis or neutral line (**?**, pp. 67). **??** shows the neutral plane after the bending operation, it is visible, that it is closer to the inside of the bend than to the outside of the bend. This is because the inside of the bend is compressed and the outside is stretched, described before. This causes the metal to deform and stretch on the outside, while compressing and thickening on the inside. The arrows show where the metal was stretched and where it was compressed.

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.

2.2.1 | Air Bending

In the manufacturing industry, bending sheet metal is a critical process, allowing for the creation of various metal shapes and configurations. As can be seen in Groover (2020)

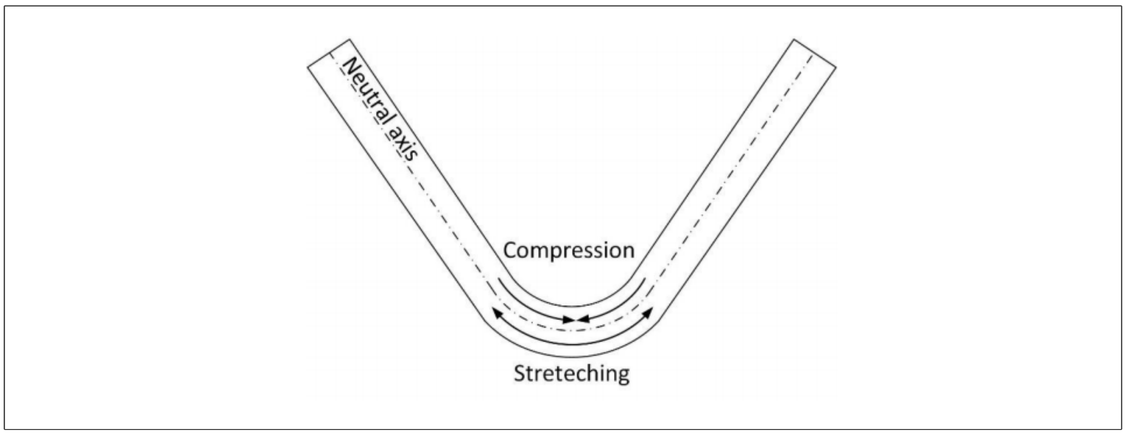


Figure 2.1: Neutral plane and the compression and stretching of a sheet metal (**?**, p. 3)

V-bending and its variant, Air Bending, are both techniques that utilize punch-and -die tooling (**?**, p. 416). The V-bending process, as shown in Figure **??**, entails pressing the metal into the shape of the die to form precise and complex bends. The die can take various shapes, including U-shaped, Z-shaped, or any other shape required to achieve the desired bend.

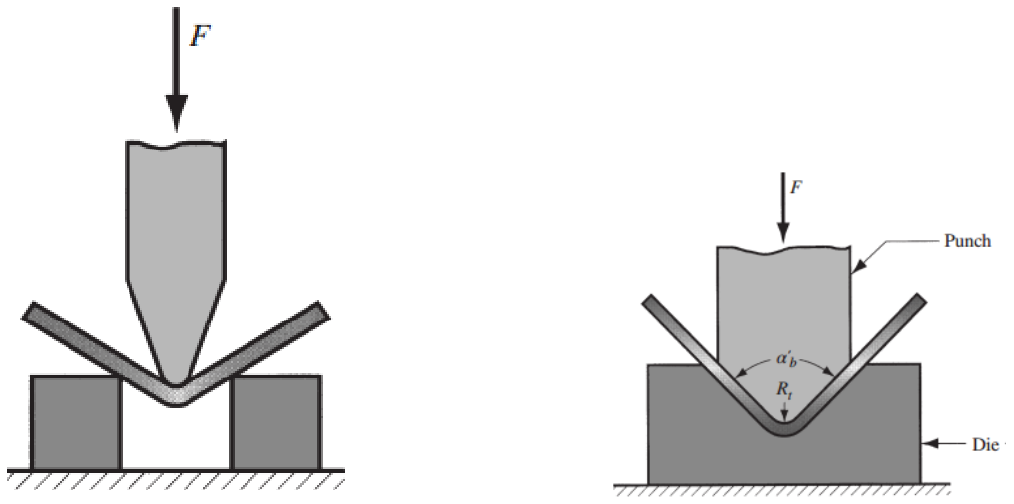
In contrast, air bending, as depicted in Figure **??**, utilizes an open die that only supports the metal on each side. This process permits a more extensive range of bend angles since the punch travel distance is not restricted by the die. Air bending can also achieve a tighter bend radius compared to V-bending, but it may result in a slightly rounded bend.

In the automotive sector, air bending is frequently utilized to create sheet metal components (**?**, p. 342). Because it is flexible, it can achieve a variety of bending angles with the same punch-and-die tooling, it is usually the most popular bending technique (**?**, p. 3) (**?**, p. 1).

Modern press machines are frequently fitted with "computer numerical control" (CNC) systems, which automatically regulate the bending process and generate the desired shape. (**?**, p. 3) The process parameters are explained in **??**.

2.2.1.1 | Spring Back

After the deformation pressure is released, there is still some elastic energy remaining in the bent part. As a result, the bent part will partially return to its original shape, which



(a) Air bending (b) V-Bending

Figure 2.2: Differnt bending methods (**?**, pp. 416)

is known as spring back (**?**, p. 413–414).

?? shows the spring back of a metal plate after the punch was removed. The spring back is determined by the angle of the bent part after the punch is removed compared to the angle when the punch was still applied. The solid line shows the metal plate in its original form when the punch was still applied. The dashed line shows the metal plate after the punch was removed.

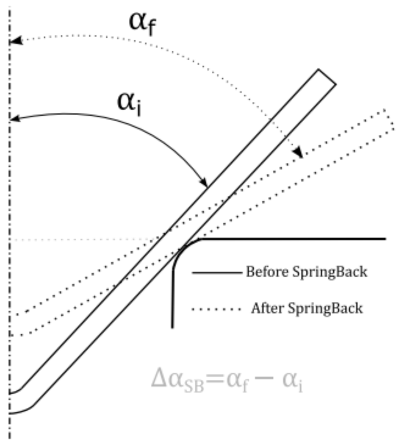


Figure 2.3: Spring back (**?**, p. 5)

The angle before the spring back is usually denoted as α_i and the angle after spring back as α_f. The spring back ($\Delta\alpha_{SB}$) is therefore the difference between α_f and α_i as show in equation **??** (**?**, p. 6).

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

To tackle this problem, there are various techniques to compensate for spring back. One commonly used approach is over bending, wherein the punch angle and radius are fabricated smaller than the specified angle. Prerequisite for all compensation methods is that the amount of spring back is known therefore the accurate prediction of the spring back play an important role in the manufacturing process (**?**, p. 114).

2.3 | Machine Learning

? describe machine learning as "Machine learning is about extracting knowledge from data." and "it is a research field at the intersection of statistics, artificial intelligence, and computer science and is also known as predictive analytics or statistical learning." (**?**, p. 1).

The purpose of the model is to make predictions or decisions without explicitly requiring programming to do so. **?** use programming terms to explain thad ML models rather than being "hard coded", are "soft coded", which means that they naturally alter their structure through repetition to increase their performance in performing the target task (**?**, pp. 4) (**?**, pp. 151–170). "This adaptive process is known as training, and it involves providing the algorithm with input data samples and the expected outputs, allowing the system to optimize itself to achieve the desired result not only with the training inputs but also with new, previously unseen data." (**?**, pp. 4). The training is the fundamental aspect of **ML!** and it can be continuous allowing the algorithm to learn from new data and its mistakes (**?**, pp. 4).

Machine learning algorithms have numerous applications in fields just as example agriculture (**?**), computer vision (**?**) or metal working (see section **??**).

2.3.1 | Terminology

This study tries to use common terminology for **ML!** to make it easier to understand: To carry out machine learning, a *dataset* ((**?**, p. 3)) is required as the initial foundation. A dataset is composed of *samples*, with each sample being a single row or observation. For example, if a dataset contains information about metal sheet bending, each row would represent a single bend.

A *feature* is an individual measurable property or characteristic of the data, and the associated value is called an *attribute*. A feature could be, for example, the thickness of a sheet metal component, and the attribute could be 3 mm. Features used to predict the outcome of a sample are called *independent features* or *input*, while the outcome feature is referred to as the *dependent feature* or *target*. In this study, the spring back is the dependent feature, while the other features which will be explained later are independent features. The dataset of this study will be explained in **??**.

The process of using machine learning algorithms to create a model is called *training* or *fitting* (**?**, p. 4), and the data used for that process is commonly referred to as *training data*. Usually, the training data is a subset of the whole dataset. The process of using the model to make predictions on new, unseen data is called *generalization* (**?**, pp. 26–27) (**?**, p. 4). The test-train split is described in more detail in section **??**.

The outcome of the learning process is referred to as a *model* that can be used to make predictions on new, unseen data.**ML!** algorithms are also often referred as *learners*.

2.3.2 | Supervised Learning and Unsupervised Learning

Learning tasks are classified into two types based on whether or not the outcome is already included in the training data. There are two types of learning: supervised learning and unsupervised learning. (**?**, p. 4).

Supervised learning is employed when not only the independent features but also the dependent feature is known. Since this is the case in the dataset used in this study, supervised learning is the preferred method (**?**, p. 2). To achieve this supervised **ML!** algorithms are trained with parts of the dataset, the training data. Building the training set usually requires human effort but once it is built supervised learning automates the task and makes it more efficient. The two main types of supervised learning are classification and regression (**?**, p. 25), which will be explained in the following section.

2.3.3 | Classification and Regression

Problems that are solved with machine learning can fall in one of two categories: Classifications and regression problems. Depending on the nature of the problems different algorithms are performance metrics are used (see **??**). For classifications the ML model needs to predict a class label from a predefined set of labels which are already present in the dataset (**?**, pp. 25–26). An example could be the prediction of the type of a flower based on its features such as petal length and width. The possible outcomes are the different types of flowers, which are the class labels.

On the other hand in regression problems the ML models tries to predict a continuous number (**?**, pp. 25–26). An example for a regression problem could be the prediction of the price of a flat based on its features such as location of rooms and square footage. The price in ths case is a continuous value, which means that it is not discrete

In this study the goal is to predict the spring back of sheet metal components, which is a continues number and therefore a regression problem. One usual problem of regression as well as classification ML models is that they tend to over-fit the training data which will be explained in the following section.

2.3.4 | Overfitting and Underfitting

The discrepancy between the output predicted by the learning algorithm and the actual output is referred to as the error. The error is usually determined on the new for the model unseen samples and therefore is also called generalization error. The error calculated on the training set is called empirical error. (**?**, p. 26).

One of the main goals of Machine Learning is to minimize the generalization error to create a model that performs well on new, unseen data. The difficulty is, that the details of new samples are unknown while training the learning algorithm, that means that only the empirical error can be minimized. This bears the risk of creating models that perform well on the training data but generalize poorly on new, unseen data. A usual reason is that the algorithm learns details of the training samples as general properties of the data which are not true for new samples. This phenomenon is called overfitting and is the main reason why the generalization error is usually higher than the empirical error. The opposite phenomenon is called underfitting and happens when the algorithm fails to learn geneal properties of the training samples (**?**, p. 26).

Overfitting models are too complex for the amount of data provided and hence do not generalize well, whereas underfitting models are too simplistic and perform poorly on the training set (**?**, p. 28).

? makes a good example for overfitting shown in **??**. The left plot shows a model that is too simple and therefore underfits the data while the most right plot shows a model that is too complex and therefore overfits the data. The middle plot shows a model that fits the data well and generalizes well to new samples.

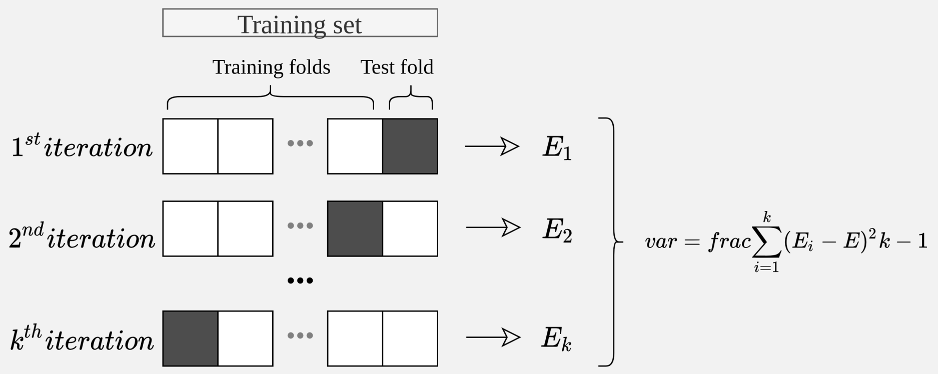


Figure 2.4: Variance of cross-validation, source: (**?**, p. 260), own illustration

The goal is to find the simplest model that captures the variability in the data (**?**, p. 35). Overfitting and underfitting can result from two different sources of error in the model: bias and variance, which will be explained in the following section.

2.3.5 | Bias-Variance Tradeoff

To explain the bias and variance first some terminology is needed. Accuracy refers usually to the closeness of the model's predictions to the actual values. Precision refers to the closeness of the model's predictions to each other, meaning how consistent the model's predictions are. Bias can be though of as the opposite of precision and refers to the systematic error in the model's predictions, meaning how far off the predictions are from the actual values. (**?**, p. 2). Applying this on the previous **??** high bias can result in underfitting, where the models is too simple to capture the complexity of the data. Variance on the other hand is the opposite of precision and refers to the variability of the model's predictions when trained on different subsets of the data (**?**, p. 2). High variance can result in overfitting, where the model learns the noise in the training data and therefore does not generalize well to new data (again **??**).

Variance and bias are conflicting because they represent two different sources of error. Error from bias is due to the model being too simple, while error from variance is due to the model being too complex. In order to have good generalization performance of the model a small bias and a small variance is favourable. **?** describe in their paper that the "price to pay for achieving low bias is high variance" (**?**, p. 14). Therefore, as the model becomes more complex, the bias decreases and the variance increases, and vice vers (**?**, p. 14). This is known as the bias-variance tradeoff and goal is to find a model that has a low bias and a low variance and minimized the overall error.

At this point common errors in ML models are explained and the goal of ML models is to minimize the generalization error. The following sections explain how to evaluate and improve ML models and how to find the best model for a given problem.

2.3.6 | Evaluation and Improvements of Models

There are several methods to evaluate and improve ML models. In the following sections provide explanations for the most important metrics and methods for this study.

2.3.6.1 | Regression Evaluation Metrics

Commonly used metrics for regression models are the **MAE!** (**MAE!**), Mean Squared Error **MSE!** (**MSE!**) and **RMSE!** (**RMSE!**). Their formal description is given in the equations **??** and **??** where e_i is the prediction error which is the difference between the predicted value of the model and the actual value. While y_i is the actual value and n is the number of samples in the testing data set.

The MAE is a measure of the average magnitude of errors in a set of predictions, without considering their direction. As it returns the same units as the data, it is easy to interpret. A lower MAE indicates better model performance (**?**, pp. 1248).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| \tag{2.2}$$

The MSE is computed like shown in **??** (**?**, p. 1248) (only the part inside the root) by determining the average of the squared differences between the predicted and actual values. If the predicted responses closely align with the actual responses, the MSE will be low, but if there is a significant difference between the predicted and actual responses , the MSE will be high (**?**, p. 30). The MSE is expressed in squared units of the data, making it difficult to interpret and therefore the RMSE is preferred.

The RMSE measures the average difference between the predicted and actual values and is calculated as the square root of the MSE. Its expression is in the same units as the response variable, making it the most popular metric for assessing the effectiveness of regression models. The RMSE and respectively the MSE are more sensitive to outliers than the MAE.

$$MSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \tag{2.3}$$

2.3.6.2 | Cross Validation

Cross-validation is a technique for testing a model's performance by training multiple models on different subsets of data and comparing their performance. The most frequent type of cross-validation is k-fold cross-validation, which divides the data into k folds and trains k models, each with one fold serving as the test set and the remainder serving as the training set. set (**?**, p. 252–260). The accuracy of each model is then assessed, and the average accuracy over all k models is used to quantify the model's generalization performance. 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 (**?**, p. 252–260).

Figure **??** shows how the cross-validation process and how the variance is calculated. It can be seen that the training data set was divided into multiple folds. The number of folds is denoted as k. One fold is used as test set (grey) while the remaining folds are used as training set (white). A model is trained on each iteration and the performance is evaluated with a metric denoted as E. Common metrics that can be used for evaluation are R^2, MSE, MAE and $RMSE$ which are explained in section **??** where they are used.

2.3.6.3 | 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 (**?**, p. 260–275).

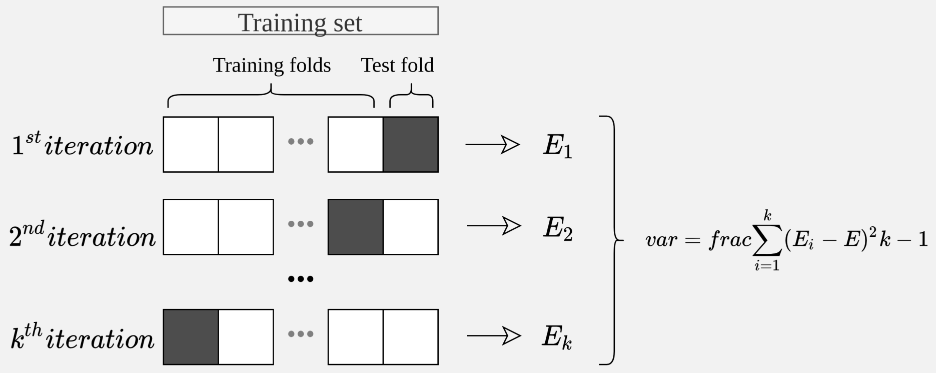


Figure 2.5: Variance of cross-validation, source: (?, p. 260), own illustration

2.4 | State of research

Sheet metals are prone to experience spring back after being bend since their elastic properties. Therefore, extensive research has already been conducted about spring back behavior in sheet metal forming processes. (?, p. 566). This chapter presents a non-exhaustive overview of the most important research for this study in the field of spring back prediction in sheet metal forming processes. The overview focuses on the application of **ML!** methods to predict the spring back.

In recent years, there has been a surge in the adoption of **ML!** for various applications related to sheet metal forming, aiming to achieve optimal manufacturing quality. Both supervised and unsupervised **ML!** methods have been employed in these applications (?, p. 2). Current research on mitigating spring back in sheet metal forming often use numerical simulations, most notably finite element analysis (FEA) to generate the data (?, p. 566). That means that the data used for training the model is generated by a simulation and not generated with real-life experiments.

? present two methods for compensating spring back in the deep drawing process, but conclude that more reliable simulations are needed for industrial applications. Using a Support Vector Machine, Liu et al. predicted spring back in a micro W-bending process. When compared to experimental results, they demonstrated high prediction accuracy and generalization performance (?, p. 1).

? investigated a variety of **ML!** algorithms for predicting spring back and maximum thinning in U-channel and square cup forming processes. The Multilayer Perceptron model was discovered to be the most effective (**?**). Similarly, **?** compared the performance of a quadratic response surface method (RSM) and two **SVM!** (**SVM!**) models for determining the best surrogate model for probabilistic sheet metal structure optimization. The **SVM!** models both outperformed the RSM model (**?**).

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. observe that virtual tryout implementations continue to rely substantially on human experience to make crucial decisions. Furthermore, the use of FEM does not guarantee the elimination of unforeseen faults caused by differences in material properties, tool geometry, and process parameters (**?**, p. 2).

? used a Support Vector Machine **SVM!** to predict the spring back of micro W-Bending operations (**?**). **?** 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 (**?**, 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 (**?**, p. 13).

Spring Back Prediction Using ANNs

Because of their great accuracy and generalization capabilities, **ANN!** (**ANN!**)s are used in sheet metal forming. (**?**, p. 2). **?** compared neural network and regression models to predict the spring back of steel sheet metal during the air bending process. They discovered that ANN could predict the spring back with greater accuracy (**?**). **?** created an ANN for the air bending process that predicts both spring back and punch travel in order to achieve the desired angle in a single stroke. **?**. **?** developed an ANN trained with FEM simulation data to predict the spring back for the wipe-bending process.

Because **ANN!**s 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 **ANN!**s trained with Finite Element Method **FEM!** (**FEM!**)simulation data.

3

Research methodology

The research methodology employed in this thesis adheres to the Design Science Research (DSR) approach, as detailed in (**?**, p. 17). DSR is a research paradigm where the researcher, acting as a designer, strives to develop artifacts that address specific problems or questions (**?**, p. 10). In the context of DSR, "design" is defined by Peffers et al. (2007) as the process of crafting an applicable solution for a given problem (**?**, p. 47).

As a methodological framework, the design-oriented research approach is particularly well-suited for addressing the research questions at hand. The prediction of spring back and bend deduction constitutes a significant issue in practical business applications, as previously discussed. Moreover, the development and implementation of machine learning models are inherently design-oriented activities. The term "artifact" is deliberately broad, encompassing various forms. In this study, the artifacts refer to **ML!** models that are applied to the problem of predicting spring back. Therefore the term artifact and machine learning model are used interchangeable in this study.

DSR can be executed through diverse means, with a notable example provided by **?**, as illustrated in Figure **??**. This approach consists of six steps, which are further divided into the overarching phases of "Build" and "Evaluate". This thesis adheres to these phases in its research methodology.

"Activity 1 - Problem identification and motivation" (?, p. 52): "Define the specific research problem and justify the value of a solution." (**?**, p. 52)

This study aims to improve the prediction of spring back in the air bending process using real world data with supervised learning. The value of the solution will be a reduction of trial and error cycles and therefore a reduction of time and cost in the air bending process. The problem as well as the motivation for this study was described in the **??** "Introduction" in more detail.

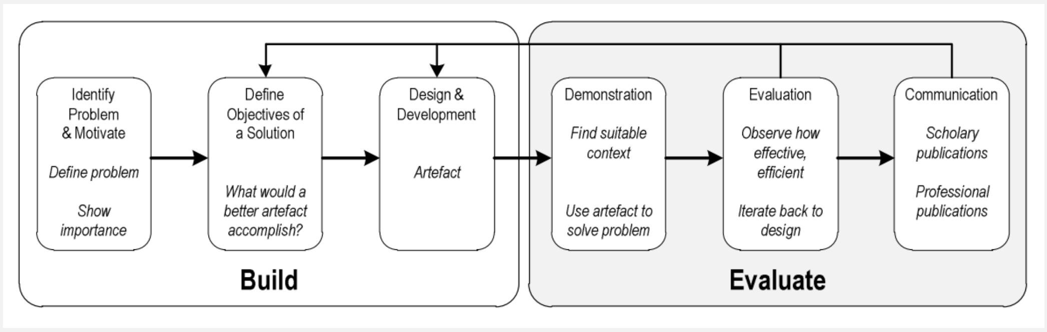

Identify Problem & Motivate
Define problem
Show importance
Define Objectives of a Solution
What would a better artefact accomplish?
Design & Development
Artefact
Demonstration
Find suitable context
Use artefact to solve problem
Evaluation
Observe how effective, efficient
Iterate back to design
Communication
Scholary publications
Professional publications
Build
Evaluate

Figure 3.1: Design Science Research Approach according to Peffers et al. Picture: (**?**, p. 72)

"Activity 2: Define the objectives for a solution" (?, p. 55): The goal of a solution comes from the definition of the problem, which takes into account what is possible and feasible. Objectives can be quantitative or qualitative, and they should be based on the problem specification that was established in the previous step. Knowledge about previous solutions and their effectiveness is required (**?**, p. 55).

The objectives for the solution, the different **ML!** models, will be to excell in all six Design Principles which will be introduced in the next chapter. The Design Principles are a set of criteria that are used to evaluate the performance of the **ML!** models.

"Activity 3: Design and development" (?, p. 55): In this phase, the development of the artifact takes place. An artifact, which may encompass models, methods, or constructs, serves as a critical element in addressing the research question. The process begins with outlining the functionality and architecture of the artifacts, followed by their subsequent creation (**?**, p. 55).

In this study the artifacts are Machine Learning models. This steps includes the model selection and the training of the models. The development of the artifacts is described in the **??** "Build".

"Activity 4: Evaluation" (?, p. 56): In this activity, the effectiveness of the developed artifact in addressing the defined problems from Activity 1 is examined and assessed. The evaluator is expected to have knowledge of relevant metrics and analytical techniques. The evaluation process may vary based on the nature of the problem at hand which is Machine Learning in this study (**?**, p. 56). A comparison between the functionality of the artifact and other existing solutions can be conducted. Additionally, quantifiable parameters can be employed to objectively measure the artifact's performance (**?**, p. 56). As already stated in this study the created artifacts are evaluated using six distinct design principles.

? note that researches are not obligated to follow the order of activities from 1 to 6 (**?**, p. 56). In this study the order fo the activities 4 and 5 swapped, in the original DSR approach the evaluation is done after the demonstration activity, this was changed because the evaluation activity is used to select the best performing artifacts. How the **ML!** models where evaluated is described in the **??**.

Activity 5: Demonstration The purpose of this activity is to demonstrate the effectiveness of the previously created artifact in solving one or more problems. Effective knowledge of the artifact is necessary for this activity (**?**, p. 55). Unlike the evaluation activity, which assesses the overall performance of the artifact, demonstration focuses on the performance of the artifact in specific scenarios. To achieve this, three scenarios, each comprising two examples (single spring back predictions), are utilized to compare the best-performing models. The scenarios are selected to cover the entire attribute space of the problem and are described in detail in **??**, *Evaluation*.

Activity 6 - Communication The last activity is ued to communicate significance of the problem and the usefulness of the artifacts to a wider audience (**?**, p. 56). In this study this is done in the **??** "Results and Discussion".

3.1 | Design Principles

Recent studies suggest that the desired result of design science endeavors should also be knowledge that takes the shape of **DP!** (**DP!**) **???**. They capture knowledge concerning the production of additional examples of objects that belong to the same category (**?**, p. 39). Other ML models are examples of the same category in the context of this study. This information is gathered as the researcher progresses from one instance of the artifact to a more general level (**?**, p. 37).

In this study **DP!**s should provide a set of guidelines for the creation of new artifacts.

3.2 | Evaluation of Machine Learning Models

The field of software engineering has established standards for determining the quality of software systems and their components. One such notable standard is the ISO/IEC 9126, which provides a quality model that is widely used in software engineering. These standards are not applicable for evaluating **ML!** models, as data-driven software components, such as **ML!** models, have functionalities that is not fully defined by the developer but rather is learnt from data. As a result, when compared to traditional software engineering, using ML poses unique issues. (**?**, p. 2). Therefore, as noted by **?**, adaptation is necessary. They have reinterpreted and expanded upon these existing quality models to make them applicable to the context of Machine Learning, as stated in their work (**?**, p. 1).

In this study, in order to define the Design Principles for the **ML!** models in this work, the considerations of (**?**) are used. This allows for a systematic process to assess the quality of the developed models. **?** consider several quality attributes, 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 not complete and are only used as a starting point for the evaluation of the models in this work, further evaluation is described in **??**.

3.3 | Goal Question Metric Approach

To make the defined quality attributes measurable, the Goal-Question-Metric (**GQM!** (**GQM!**))-approach was chosen in this study. It is one of the most common approaches in DSR and is divided into three levels (**?**, 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.” (**?**, p. 3)

Specific goals are established for a software product (in this case a **ML!** model). These goals consider various quality aspects, perspectives, and contexts. Clearly defined goals help to guide the direction of the development (**?**, p. 3). In this study the goals are the Design Principles that are to be established in the next **??**.

2. Operational level (question): To achieve the set goals, one or multiple questions are formulated to assess and characterize the objectives. These questions address the quality attributes and help to understand the object of measurement from the selected viewpoint. The questions guide the measurement process by identifying the relevant aspects that need to be analyzed (**?**, p. 3).

3. Quantitative level (metric): For each question, a set of metrics is defined to provide quantitative data and answers. These metrics are used to evaluate the object of measurement, offering precise and objective insights into the quality aspects being assessed (**?**, p. 3).

In this study only one question is formulated for each goal. The goals, questions and metrics are presented in the next chapter.

4

Build

This chapter outlines the activities of the **DSR!** (**DSR!**) process, which are summarized under the “Build” phase, except “Identify Problem & Motivation” which was already done in the **??** “Introduction”. The first activity in the DSR process is to identify the problem that needs to be addressed, this step leads to the formulation of **DP!**, which are later used to evaluate the effectiveness of the artefact.

4.1 | Objectives of a Solution

The objectives of a solution are essential to evaluate the effectiveness of ML models in the DSR process. Therefore six Design Principles are used based on the quality parameters proposed by **?**. These quality parameters are specifically tailored for evaluating ML models and are therefore well-suited for this study.

In the following sections six DPs will be described which will provide insights into the strengths and weaknesses of the ML models and guide the decision making process in the DSR process. The chosen **DP!** are: Correctness, relevance, robustness, stability, resource utilization and interpretability. By leveraging these principles, researches can optimize their models to achieve the desired outcomes and improve the quality of their research. In Chapter **??** “Evaluation” uses the DPs from this chapter and combines them with relevant questions and metrics to assess the effectiveness of the ML models. This approach will ensure a thorough evaluation of the ML models based on the quality parameters that are important for the DSR process.

This study aims to get all information to answer three research questions, which are formulated in the

Design Principle 1: Correctness

The demand for lightweight products is a driving force in the manufacturing industry, as consumers and businesses alike demand products that meet stringent standard of precision and accuracy (**?**, p. 2). 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 it is desirable to minimize the spring back (**?**, p.1).

ML! has emerged as a powerful tool for predicting the spring back in other bending methods as shown in **??** "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 model 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 (**?**, p. 16).

Understanding the bias-variance trade-off is crucial in developing **ML!** models that generalize well to unseen data (**?**, p. 49–51). It suggests, that a model can accurately capture the underlying patterns in the data without overfitting or underfitting (**?**, p. 49–51).

Design Principle 3: Robustness

Robustness is described in the IEEE standard glossary of software engineering terminology as: "The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" (**?**, p. 64). **?** 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" (**?**, p. 2). **?** also specifically mention data quality issues like outliers or noise (**?**, p. 16).

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 model should be robust, to be able to handle data quality issues and still produce accurate predictions (**?**, p. 16).

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 (**?**, p. 16). This is particularly important in this study, since the training data is limited and the model is trained on a smaller 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 (**?**, p. 16). Stability can be linked back to the discussion of variance in **??**. Variance refers to the models precautions vary for different datasets, if a model is not stable it will generate results that differ for different datasets, which might results in poor overall generalization performance.

Design Principle 5: Resource utilization

One of the primary objectives of utilizing machine learning for predicting spring back is to reduce the number of trial-and-error cycles during the manufacturing process, thus saving valuable resources such as material and labor (refer to **??**).

For example the technology sighted in **??** requires 11 unique bends to obtain the data for only one type of metal with a certain thickness. There are multiple problems with this approach: (personal communication, Dipl. Ing. Michael Philipeit).

- Only data for one type of metal with a certain thickness can be obtained with on technology table. If new data is needed for a different type of metal or thickness, a new technology table has to be created.
- The metal sheets cannot be repurposed due to their specific angles resulting in wasted materials and manual labor.
- The machines used for bending the metal are often occupied around the clock, further adding to the inefficiency of the bending process.

Machine learning (**ML!**) models have the potential to accurately predict spring back for various types of metals, thicknesses, and angles, without the need for new technology tables. This approach offers several advantages,including faster processing times and reduced manual labor requirements. Additionally, it eliminates the need for new metal sheets, resulting in less waste. By leveraging the power of **ML!** algorithms, manufacturers can optimize their metal bending processes and reduce costs associated with material waste and manual labor.

Nevertheless, it is crucial to acknowledge that developing an **ML!** model also entails resource consumption. Training **ML!** models can be a labor-intensive process, as it often requires significant computing power and the generation of large amounts of training data, depending on the algorithm used. Resources used up by **ML!** are for example memory space and also time needed to train the model and make predictions.

Therefore, it is crucial to carefully consider the resources needed to train and use the model effectively (**?**, p. 16). What resources are used and evaluated is described in more detail in **??**.

The initial five **DP!**s primarily assess the performance of **ML!** models across various criteria. However, the final **DP!** takes a divergent approach by focusing on the human aspect of **ML!** models. This is crucial because a model's accuracy alone is insufficient; it must also be comprehensible to the user.

Design Principle 6: Interpretability

? defines interpretability as "as the ability to explain or to present in understandable terms to a human." (**?**, p. w). **?** add that it is also important to understand "why" the model has this performance (**?**, p. 1). Good interpretability is crucial because it fosters trust in the model and enables users to rely on it. Additionally, interpretability can aid in debugging and enhancing the model's performance.

In this study it is the goal to comprehend the decision making process of the trained models as good as possible. As discussed in **??**, the sheet metal forming process involves numerous variables, making the process design intricate, particularly when producing components that necessitate several processing stages and multiple sets of tools (**?**, p. 1). Therefore, and interpretable model is deemed a better model and enables the user to make informed decision based on the models results.

4.2 | Dataset generation

In this study a dataset was generated by conducting a series of bending experiments on metal sheets with varying thicknesses. Cold-rolled steel conforming to the DIN EN 10130 standard was used as the material, with thicknesses ranging from 0.5 mm to 3 mm.

This material was chosen due to its widespread use in bending processes and its wide availability. To create the dataset, 20 x 80 mm sheets were utilized along with a stamping press from the company ZwickRoell GmbH & Co. KG The metal sheets were bent in the rolling direction, as it could be observered in previous experiments that the spring back effect varied depending on the rolling direction. The original dataset comprised 384 `.tra` files, which is a proprietary format used by the stamping press. Python scripts were developed to convert the output data format from the machine to more accessible CSV files.

The following section outlines the experimental setup used for the experiments performed. Data collection occurred outside of the thesis period and initially consisted of 384 samples. An additional 20 samples were later added to the dataset to fill any gaps present in the dataset.

Many other studies use finite element method (FEM) simulations to generate their datasets as shown in **??**). However, the FEM is a computationally expensive method and requires a significant amount of time to generate a dataset. Additionally, FIE simulations due to the simulated nature are not able to capture all the effects of the sheet metal forming process. In contrast, data from real-world experiments is more representative of the actual behavior of the sheet metal since experiments can provide a more comprehensive understanding of the material properties, behavior, and interactions that are taking place during the forming process. This can include factors such as the properties of the metal being formed, the tooling used, and the specific process conditions. Real-world data can also provide insights into unexpected or unpredictable variations that may occur during the forming process, which can be difficult to capture in a simulated environment. Additionally, experiments can be used to validate and improve the accuracy of simulation models, ensuring that the simulation results are more representative of the actual behavior of the metal. Therefore, it is expected that the models trained on the generated dataset will be more accurate and represent the real world better.

The experiments were carried out in a controlled environment, allowing for the development of a large dataset with good repeatability.

4.2.1 | Experimental Setup

The stamping press utilized for the experimental setup is the *ZwickRoell GmbH & Co. Z100*, which is equipped with a load cell and a displacement sensor. The press is capable of exerting a maximum force of 100 kN and has a maximum stroke of 100 mm. Metal sheets with dimensions of 20 x 80 mm were used for the experiments. These measure the force (F) applied to the sheet in Newton (N) and the displacement of the punch, also called punch penetration (y_p in mm), respectively. The force is measured by the an "Xforce" load cell, while the displacement is measured by the displacement sensor **?**. The stationary die is mounted on the bottom , while the movable punch is mounted (see**??** for more details). The die opening of the machine is adjustable from 10 mm to 100 mm. The machine is operated via a computer and the *ZwickRoell TestXpert* software, which is used for both machine control and data collection.

The process parameters and experimental setup are illustrated in Figure **??**. The die opening, V, represents the distance between the two contact points of the die, where the sheet metal is placed. The punch penetration, y_p is the distance the punch moves into the sheet and the thickness of the metal sheet is denoted by t and ranges from 0.5 to 3 mm in this study. The radius of the punch tip, denoted as r_p, remains constant throughout the experiments as it was never replaced. The bending angle, α, is provided for completeness but will not be used in the subsequent analysis.

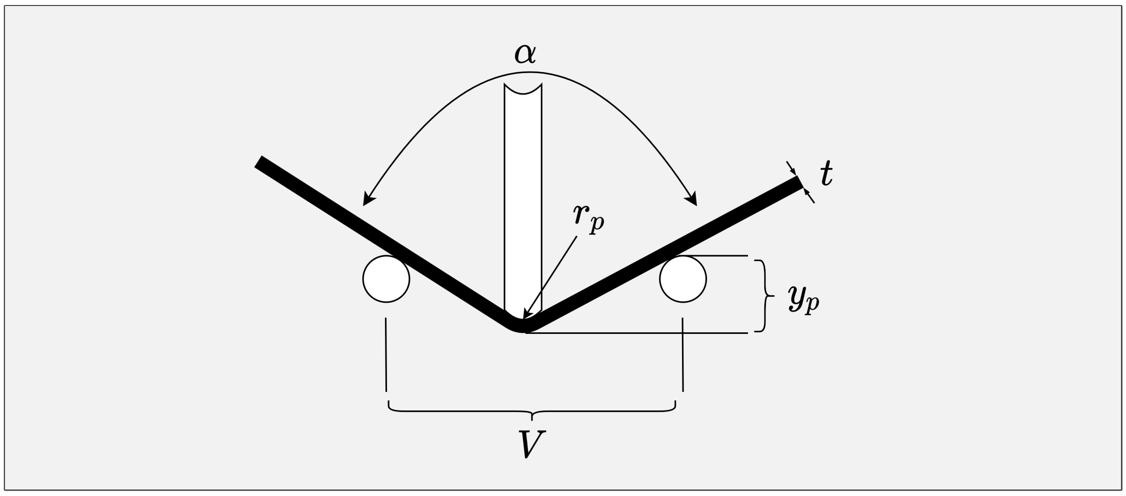


Figure 4.1: **Process parameters:** Sheet bending angle (α), sheet thickness (t), punch penetration (y_p), die opening (V) and punch radius (r_p)

To ensure consistent results, a set of constant and variable parameters were selected. The punch was never changed and therefore the punch radius (r_p) is a constant parameter. Also the length and the width of the used metal sheets were standardized to 20 mm and 80 mm, respectively.

The bending process was carried out as follows:

1. The metal sheet was placed on the die
2. The punch was moved downwards until it reached the sheet (which is the case when the load cell measures force of at least 1 N)
3. The punch was moved downwards at a constant speed of 80 mm/min until the maximum displacement ($y_{p\max}$) was reached
4. The punch was held at the maximum displacement ($y_{p\max}$) for a minimum of 1 second
5. The punch was moved upwards at a constant speed of 80 mm/min until it reached the initial position
6. The metal sheet was removed from the die

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 punch penetration (y_p), and the thickness of the metal sheet (t). The die opneing was varies from 10 mm to 50 mm, the maximum punch penetration was varied from 2.5 mm to 20 mm and the metal sheet was varied from 0.5 mm to 2 mm. These parameters and their values are displayed in **??**.

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 included various data points that were used to calculate the spring back. Key parameters for this calculation were the force, punch travel, and time which are illustrated in **??**. The blue line is representing the force while the gray line is representing the punch travel. The punch movement is measured as soon as the displacement sensor is activated with when it notices a difference of 1 N which is the case when the punch touches the metal sheet.

It can be seen that before the punch touches the metal sheet there is already a force measured of about 10 N, this is the case because the punch is moving at a speed of 80 mm/min which is relatively fast and so the displacement sensor is activated before the punch touches the metal sheet. 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.

At the point where the punch is moved fully down, known as y_{pmax}, there is a noticeable decrease in force, as observed in **??**. This phenomenon is most likely attributed to the released friction (personal communication with Prof. Dr.-Ing. Marco Schneider).

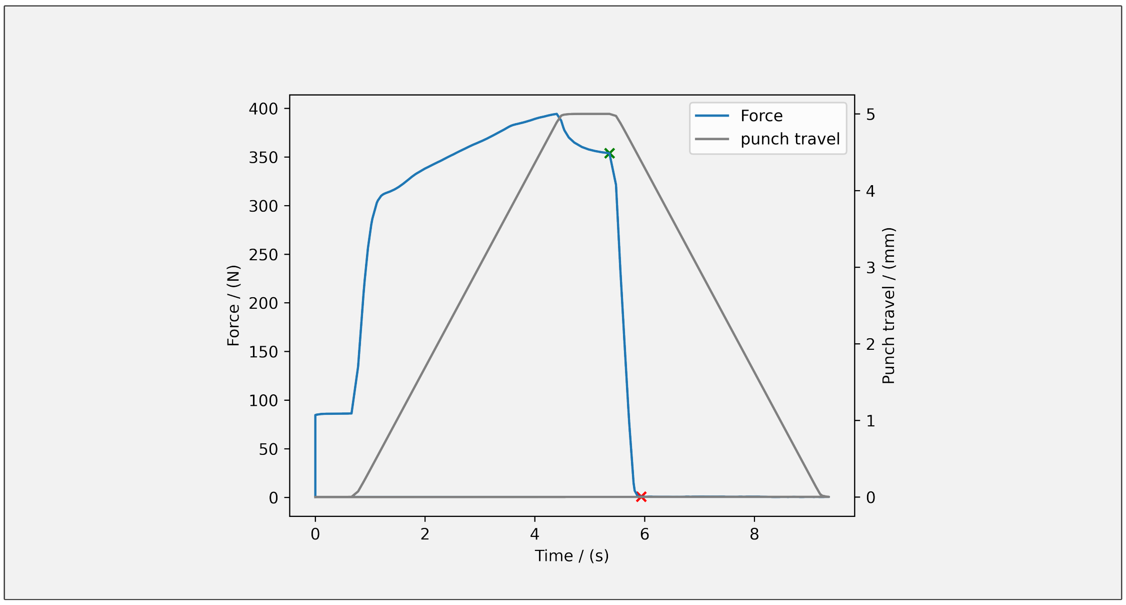


Figure 4.2: **Spring back measured:** A steel metal sheet was bent with a punch penetration of 5 mm the spring back is 0 .37 mm. The distance between the green and the red point repesents the spring back distance.

4.2.3 | Dataset Exploration

The dataset was explored using the *pandas* **?** and the *matplotlib* **?** python 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.

4.2.3.1 | 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 **??**. These three features were combined with the calculated spring back to form the final dataset, which contained a total of 404 data points generated. An example of the dataset is presented in **??**.

?? depicts the spring backs that were observed in a specific subset of the data where the die opening is 30 mm. The horizontal axis, labeled as the x-axis, represents the punch penetration distance, while the vertical axis, labeled as the y-axis, represents the spring back. In order to clearly illustrate all the data points, the mean of the spring backs has been plotted as a line.

index	Punch Penetration	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: Excerpt from the dataset with 404 data points used in this study.

Upon examining the figure, two general trends become evident. Firstly, a lower thickness of the metal sheet leads to a higher degree of spring back. Secondly, a higher punch penetration distance leads to a higher spring back as well. It is also apparent that there is a non-linear relationship between the punch penetration and the spring back. These trends are also observed for other die openings and therefore apply for the whole dataset.

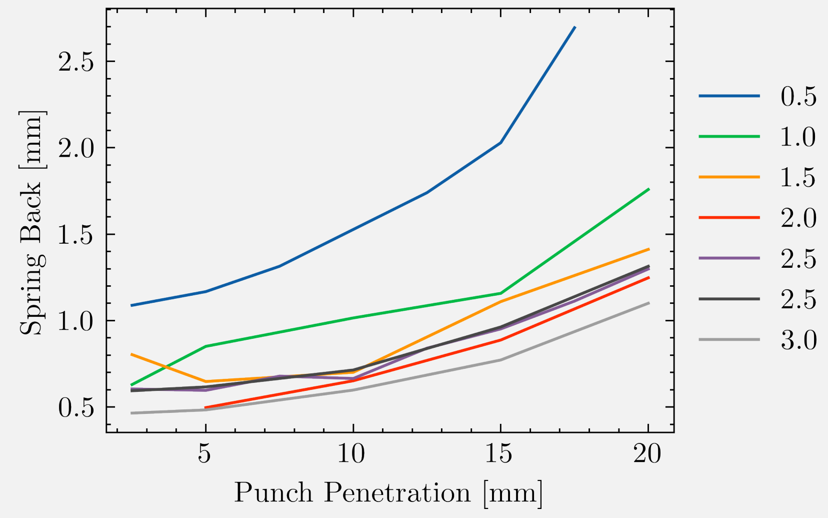


Figure 4.3: All measured springbacks for the die opening $V30$ are shown.

4.2.3.2 | Data Quality

The dataset was created careful and in a controlled experimental environment to ensure precise and accurate measurements of the samples. As a result, the dataset exhibits a minimal number of outliers and high data quality, which is a crucial prerequisite for developing reliable machine learning models. As demonstrated in **??**, the stability of the models was tested, and it was observed that the addition of noise or outliers to the dataset resulted in a significant decrease in the performance of the models. This finding is indicative of the high quality of the dataset, as it implies that the dataset does not contain many outliers that could negatively impact model performance.

?? illustrates that the dataset encompasses nearly all possible combinations of the process parameters, namely V, t, and y_p. The y_p values in the dataset are uniformly distributed and consistently ranged between 2.5 and 20 mm. Furthermore, the dataset was continuously expanded with new data points throughout the project to enhance its size and diversity. The figure also show the test-train split which will be explained in **??**.

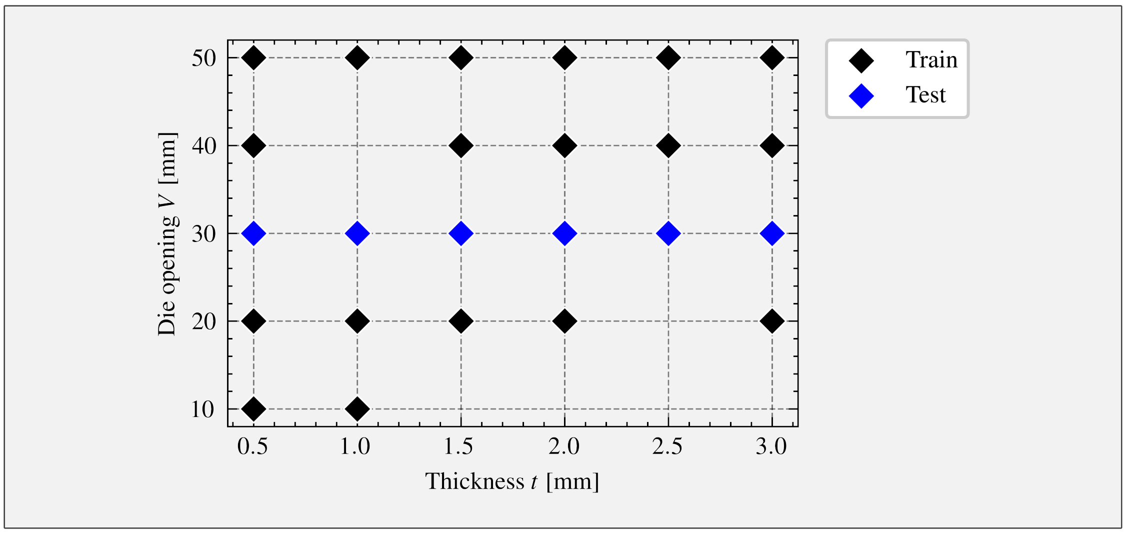


Figure 4.4: The training and test dataset.

The following gaps can be seen in the dataset: Metal sheet thickness with $V10$ after 1 mm could not be measured because the die opening was too small to fit the metal sheet. Also data for V 20 and t 2,5 as well as V 40 and t 1 are missing, for time related reasons.

Throughout the data collection and expansion process, any outliers and incorrect measurements were identified and removed from the dataset to maintain its high quality. This will provide insights into how well the machine learning models perform in more realistic scenarios and aid in developing more effective models for practical applications.

4.2.4 | Data Preprocessing

To normalize the independent features y_p, V, and t, as well as the dependent feature *spring_back*, the `MinMaxScaler` or, in some cases, the `StandardScaler` from the `scikit-learn` **?** library where employed.

The `MinMaxScaler` scales the data between 0 and 1 and the scaler was fit on the training data and subsequently used to transform the test data. It is important to note that scaling was only performed on the training data, as cross-validation was later employed to tune and evaluate the models. Scaling the entire dataset before the split could result in data leakage, as the minimum and maximum values of the test data would be used to scale the training data.

The data split is depicted in **??**.

4.2.5 | Computational Setup

A ThinkPad X1 Carbon 2019 with an Intel Core i7-10610U CPU @ 1.80GHz and 16 GB RAM was utilized to train the machine learning models. The OS used was Ubuntu 20.04.2 LTS on which the code was written in Python 3.8.5 with the PyCharm IDE. The code can be found on GitHub and on the the CD accompanying this thesis. The libraries used are listed in **??**.

Library	Version	Author Use
Sci-Kit Learn	1.2.2	**?**
numpy	1.23.2	**?**
pandas	1.5.1	**?**
matplotlib	3.6.2	**?**
scienceplots	2.0.1	**?**
seaborn	0.12.2	**?**

Table 4.4: Libraries used for the machine learning models.

A note to the citations of the libraries. All of the above libraries are well documented on their website. Libraries used for the machine learning models.

Upon examining the correlation matrix depicted in **??**, the correlation is between 0 and 1, the higher the value the stronger the correlation. 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. The other features are not connected with each other, demonstrating the lack of multicollinearity, which is a desirable trait for machine learning models.

The correlation matrix was generated using the `heatmap` function from the *seaborn* library ?.

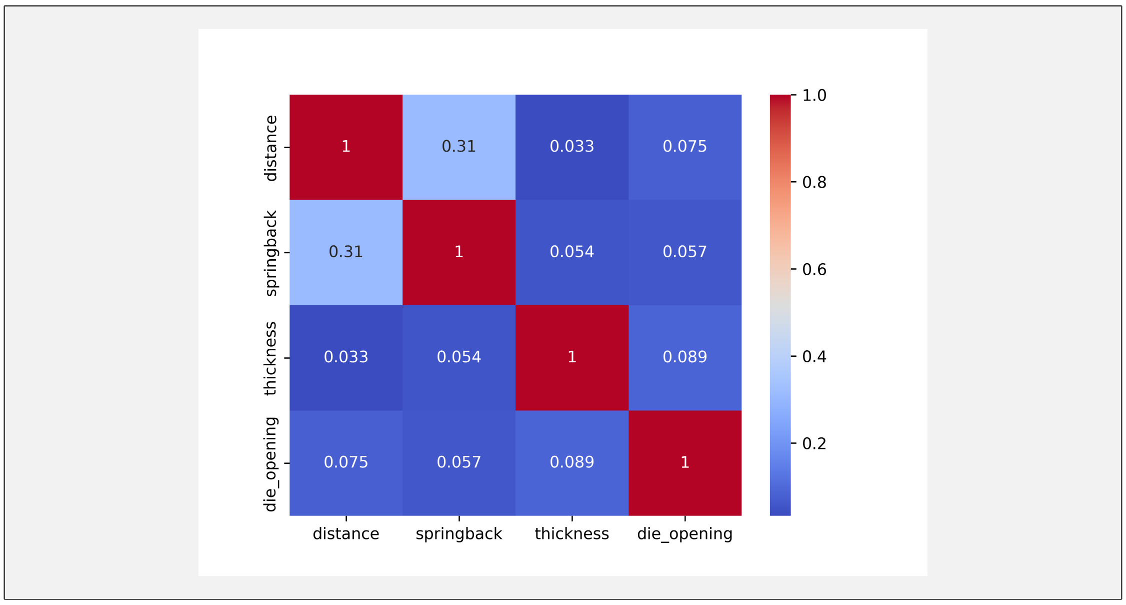


Figure 4.5: Correlation matrix generated with a Random Forest Regressor model

4.3 | 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 ?. ? defines eight widely used supervised machine learning models in the paper. These models are Logistic Regression, Support Vector Machines, Decision Trees, Random Forests, AdaBoost, Naive Bayes , and K-Nearest Neighbors.

However, Naive Bayes and K-Nearest Neighbors are not used in this thesis since they are typically utilized for classification problems. Furthermore, in order to compare the performance of the other models, a Linear Regression model and a Multi-Layer-Perceptron were also employed. The models can be classified into five categories: Logic-based learning, instance-based learning, Deep learning, and Support Vector Machines (**?**, p. 8).

4.3.1 | The Baseline Model

In this study, a Linear Regression (LR) model is employed as the baseline model. Linear regression a widely used method to predict a continuous output (dependent variables) based on a set of input features (independent variables). The goal of a **LR!** (**LR!**) model is to find the best-fitting straight to explain the relationship between the dependent (spring back) and independent (die opening, thickness, punch penetration) variables (**?**, pp. 45–46). This so called regression line can be express as shown in eq:linear-regression, which is taken from Müller and Guido (2016) (**?**, p. 45).

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

The linear regression model is trained by minimizing the sum of squared errors between predicted and actual values. The values $x[0]$ to $x[p]$ indicate the properties of a single data instance, with p being the number of features. The learned parameters of the model are w and b, and $\hat{y}$ is the prediction generated by the model.

The Mean Squared Error is calculated by adding together the squared discrepancies between the predicted and true values (**?**, p. 47–68). This model presumes a linear relationship between the independent and dependent variables. The dataset exploration in **??**, on the other hand, revealed a nonlinear relationship , implying that the linear regression model may not deliver reliable predictions. Despite this disadvantage, the model can be used as a baseline against which other models can be compared.

4.3.2 | Support Vector Regression (SVR)

Support Vector Machines (SVMs) are a popular approach for solving classification problems. To understand how the SVM algorithm works, it can be helpful to approach the problem from a geometric perspective. For example, a two-dimensional space as shown in **??** can be considered, where each data point has a class assigned, in this case, either red or blue. The SVM algorithm seeks to identify a line (hyperplane) that separates the data points into different classes in the best way possible which means that there is the largest possible distance between the line and the nearest points of each class (**?**, pp. 92–96).

The points that lie closest to the hyperplane are called support vectors, these define the margin which is tried to be maximized by the algorithm and therefore are important for determining the optimal position of the hyperplane (**?**, p. 42). The hyperplane serves as a decision boundary that separates the data points into different classes (**?**, p. 11) , in the example the points on the one side of the hyperplane are classified with the blue class while the points on the other side are classified with the red class

Since each feature in the dataset is formulated as one dimension in the geometrical space, visualizing the hyperplane is usually not as straightforward as in the example depicted in Figure **??**. In the case of the spring back dataset, for instance, the hyperplane would be a three-dimensional plane, but it is not uncommon to have spaces with even higher dimensions. The algorithm seeks a hyperplane in an N-dimensional space (where N is the number of features) in order to effectively separate and classify data points (**?**).

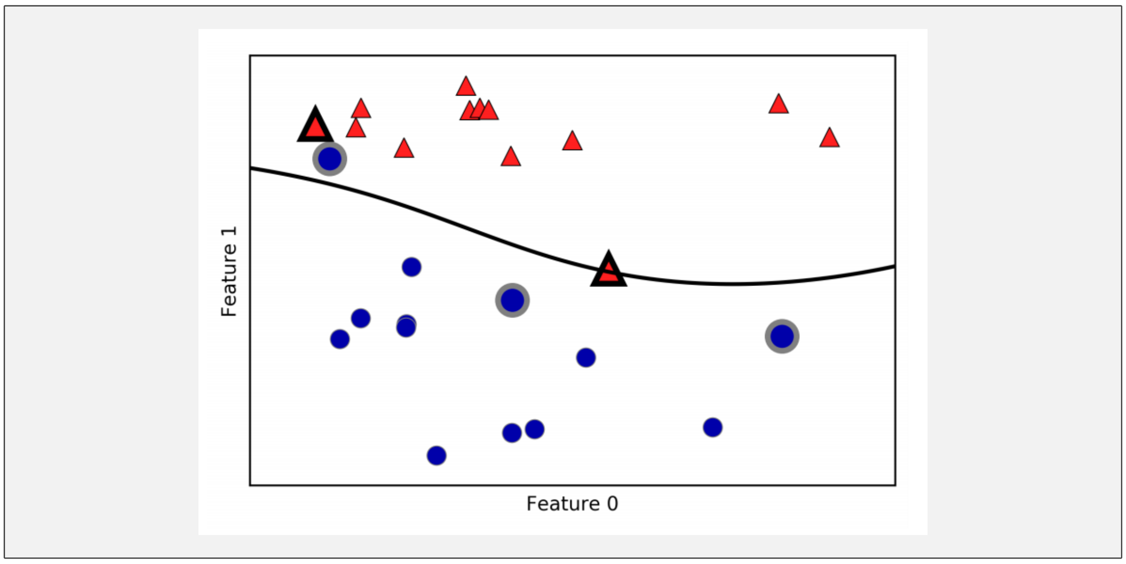


Figure 4.6: Simple **SVM!** example (**?**, p. 94)

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 Support Vector Regression (SVR) algorithm was developed, which draws inspiration from the **SVM!** algorithm and leverages similar principles (**?**, p. 92).

To enable the **SVR!** (**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 **SVM!**s penalize points in classification problems (**?**, p. 369) (**?**, pp. 67–68).

To transfer data into a higher-dimensional space, the kernel trick is used. Two methods are usually used for **SVM!**s, the polynomial kernel and the radial basis function, also known as gaussian kernel (**?**, p. 97–98). Similar to the **SVM!** algorithm, 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 (**?**, p. 369).

4.3.3 | 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 (**?**, p. 10).

4.3.3.1 | Decision Trees

Decision Trees (DT) construct a hierarchy of rules to predict outcomes based on the data (**?**, p. 70) (**?**, p. 253). Based on certain feature values, the algorithm divides the data into various segments and creates a variety of subsets of the dataset, with each sample belonging to one subset. The intermediate subsets are called decision nodes, and the final subsets are called terminal or leaf nodes (**?**, p. 358). The average outcome of the training data of a particular leaf node is used to predict its outcome (**?**, p. 70–72).

?? shows a reduced real-life example of a decision tree taken out of a Random Forest (see **??**that was trained for this study. At the core of the decision tree lies the root node, which serves as the starting point for the analysis. In this particular dataset, the root node contains all 403 data points, representing the complete set of observations in the sample. From the root node, the decision tree branches out into two or more decision nodes, each representing a split in the data based on a decision rule. These decision rules are listed on the first line of each node, and provide a criterion for dividing the data into subsets based on specific feature thresholds. For example, in the line `x[1] <= 0.75`, the decision rule is based on the second independent variable, $x[1]$, and the threshold value of 0 .75. In **??** the green nodes represents the root, the blue nodes are the decision nodes, and the red nodes are the leaf nodes.

The independent variables in this dataset are indicated as $x[i]$, where $x[0]$ indicates punch penetration, $x[1]$ represents metal sheet thickness, and $x[2]$ represents die opening.

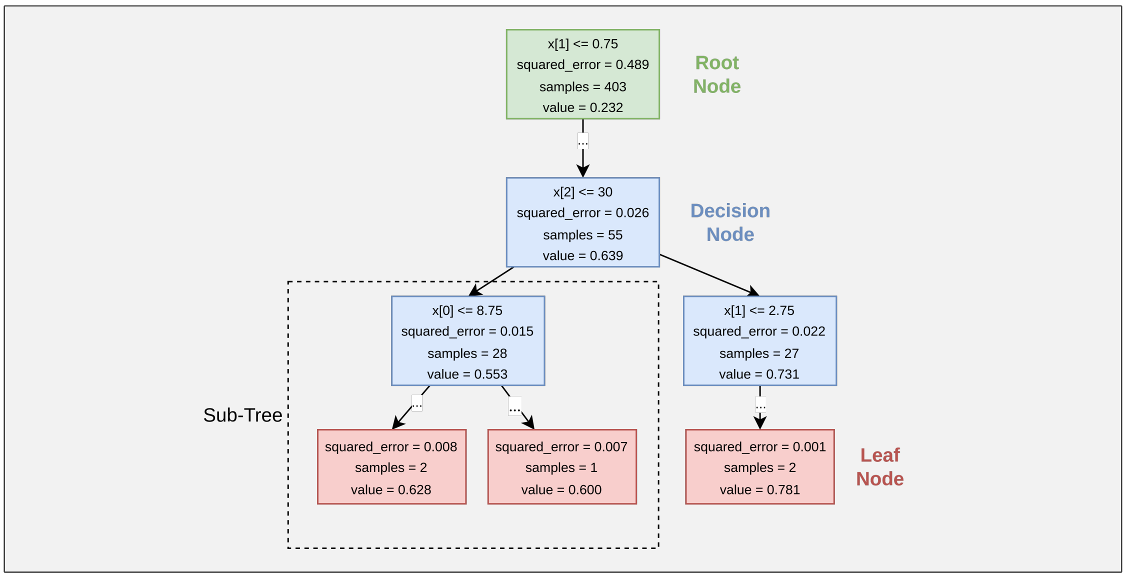


Figure 4.7: Decision tree example made with `sklearn.tree.plot_tree` **?**.

These variables are utilized to make judgments at each node and to forecast the value of the dependent variable.

The mean squared error, also known as MSE, related to the predictions made by each decision node, is a crucial metric for assessing the performance of the decision tree. This metric indicates how well the decision node separates the data into different subsets based on the selected feature threshold. A lower value indicates a better separation, and suggests that the decision rule is more effective in predicting the value of the dependent variable. Performance metrics like the **MSE!** are further explained in **??**. The line labeled "samples" in each node denotes the number of samples that reach that particular decision node. This metric provides insight into the distribution of data within the decision tree, and can be used to identify areas of high or low density within the sample.

Finally, the "value" line in each node represents the predicted spring back for the samples that reach that particular decision node. This value provides an estimate of the dependent variable and can be used to make predictions based on the input data.

4.3.3.2 | Random Forest

The primary issue with DT is their propensity to overfit, leading to poor generalization performance and rendering them unsuitable for most use cases. To tackle this problem, ensemble methods are often employed instead of a single DT (**?**, p. 83) (**?**, p. 251).

In ensemble methods, multiple models such as decision trees are combined to form a single model. To make new predictions, the predictions of each individual model are aggregated, for example by averaging the predictions (**?**, p. 222).

The Random Forest (RF) algorithm (**?**) which combines multiple decision trees (weak learners) to produce a more precise and stable prediction (strong learner) (**?**, p. 24) (**?**, pp. 340). This approach, known as the "divide and conquer" method, involves dividing the data into smaller randomized subsets and constructing a randomized tree predictor for each subset (**?**, p. 251).

The risk of overfitting is mitigated through subset and feature randomization, also known as bagging. Bagging involves randomly selecting a subset of features or variables at each decision node in the decision tree. This process generates multiple, slightly varying copies of the dataset (**?**, p. 341). Overfitting occurs when the algorithm learns individual data points instead of the general pattern (see **??**). Each root node utilizes a unique subset of the data, and each leaf is split using a random set of features. This approach ensures that no single tree is exposed to all the data, enabling the model to concentrate on general patterns rather than being susceptible to noise (**?**, p. 83). Therefore bagging aids in diminishing the impact of individual data points or outliers that may have a significant influence on a single decision tree.

The algorithm is visualized in **??** the green decision nodes are the root nodes, blue decision nodes are the decision nodes, and red nodes are the leaf nodes. The red arrows indicate the decisions made by each decision tree for one specific sample. Instead of a single decision tree, N decision trees are built in the format described in **??**. The bagging was not visualized but every single decision tree is trained on a variation of the dataset for the before described reasons. The red arrows indicate the decisions made by each decision tree for one specific sample. As a result, each decision tree independently arrives at one prediction which in this case would be a spring back value. Averaging the predictions of all trees makes the final prediction (**?**, p. 9).

The Random Forest mechanism is versatile enough to address both classification and regression problems, making it one of the most successful **ML!** methods (**?**, p. 3–4) (**?**, p. 25).

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 (**?**, p. 3–4) (**?**, p. 25).

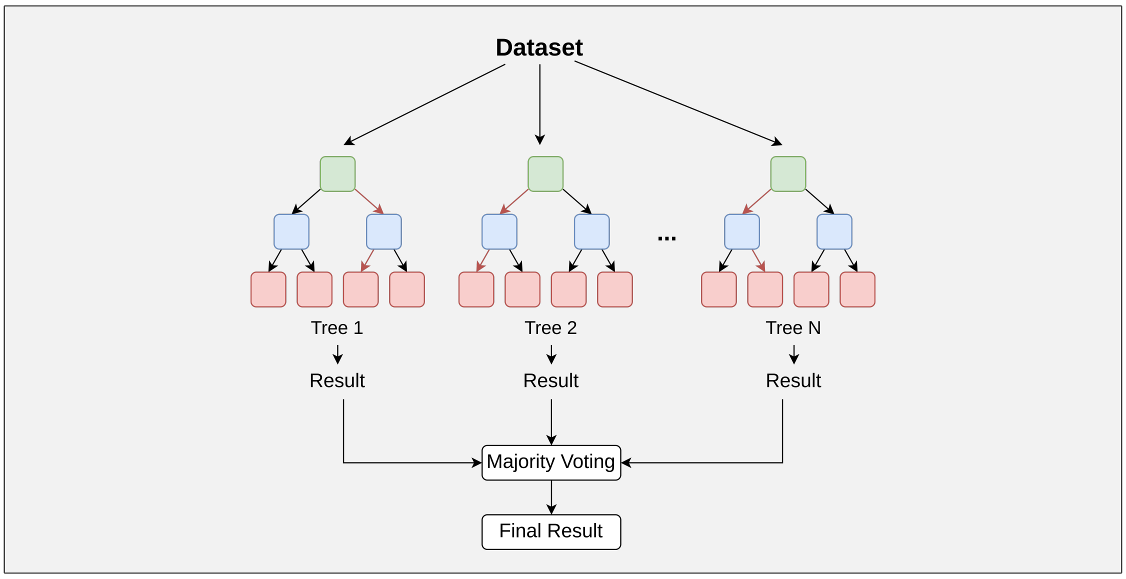


Figure 4.8: Visualization of the Random Forest algorithm as defined by (**?**, p.1) (Own representation).

4.3.3.3 | Gradient Boosting Regression Tree

A **GBT!** (**GBT!**) is an ensemble learning algorithm that combines multiple Decision Trees to yield more accurate and stable predictions. The main difference between this algorithm and the Random Forest algorithm is that the trees are trained sequentially rather than simultaneously, with each tree correcting the errors of the previous tree. (**?**, p. 88–89). This is done with training each new tree with the residual errors of the previous tree, which is the difference between the predicted value and can also be called the loss. One example for a metric that can be used as loss is the Mean Squared Error (MSE) (more details in section **??**). To train a tree on the residual errors, the process is similar to train a decision tree on any other regression task. Instead of using the original target variable (spring back) as the label for the training data instead the residual errors are used. This process sequentially boost the performance of the mode with new tree added to the ensemble, hence the name Gradient *Boosting* (**?**, p. 222)

GBMs form an ensemble of shallow trees built sequentially, whereas Random Forests (discussed in **??**) create an ensemble of independent trees with a deep structure. Every tree in the group learns from and enhances the previous tree (**?**, p. 221). The shallow trees are achieved through a process called pre-pruning, which means that the growth of a tree is halted at a specific depth. This provides the advantage of a smaller model that consumes less memory and facilitates faster prediction. Generally, generating more trees enhances the overall performance of the model (**?**, pp. 74, 88–89).

Furthermore, the algorithm performs well without scaling the dataset and can handle a combination of binary and continuous features (**?**, p. 91)

In **??**, we illustrate the Gradient Boosted Regression Trees (GBRT) algorithm using a sample dataset. The first tree is trained on the original dataset with the target variable being the spring back. After the construction of the initial decision tree, each sample in the dataset will have a predicted value which is likely different from the actual spring back. For each sample, the residual, which is the difference between the predicted and actual spring back values, is computed. The second tree in the sequence is then trained using these residuals as the target variable instead of the actual spring back values. The GBRT algorithm continues to build trees sequentially, with each tree focusing on the residuals of the preceding tree as its target variable. his process continues until a predefined stopping criterion is satisfied. Such criteria may include a maximum number of trees or a threshold for minimal improvement in the final outcome (**?**, p. 227).

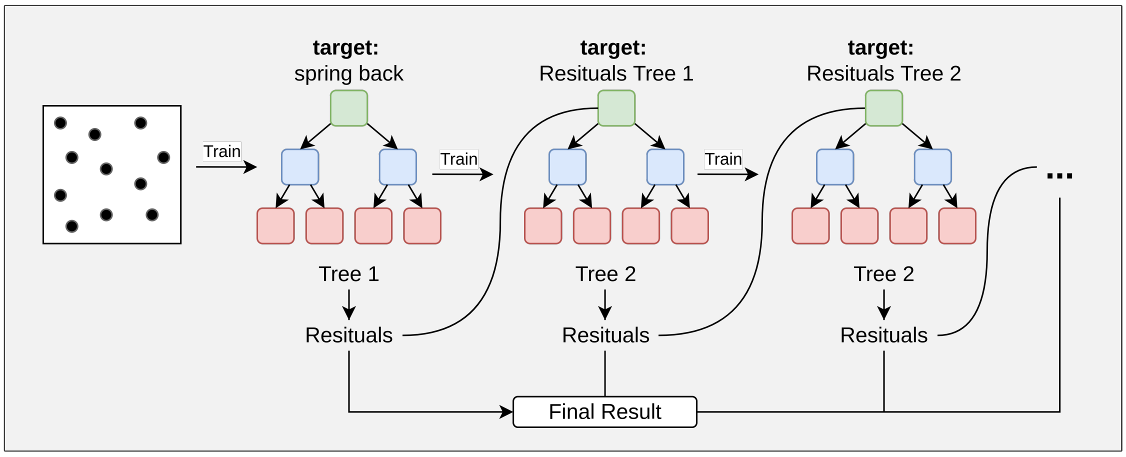


Figure 4.9: Visualization of the **GBR!** (**GBR!**) algorithm adapted from (**?**, p. 222)

4.3.3.4 | Extra Trees

In the construction of a Random Forest, Decision Trees are developed using a randomly selected subset of features for each node (see Section **??**). To further enhance the randomness of the trees, random thresholds can be employed for decision rules in addition to the random selection of features (**?**, p. 351).

An ensemble consisting of such highly arbitrary trees is referred to as an Extra-Trees (or Extremely Randomized Trees) ensemble. This approach generates more diverse and less correlated trees, leading to reduced variance and increased bias (**?**, p. 351). Furthermore, using ET classifiers speeds up the training process as compared to standard Random Forests, as establishing the correct threshold for each feature at each node is one of the most time-consuming components of tree growth (**?**, p. 351).

4.3.4 | Neural Networks

Another method for supervised learning involves the utilization of Neural Networks to perform classification and regression tasks.

While this study does not go into detail about the usage of Artificial Neural Networks for spring back prediction because that would be outside the scope of this paper. Instead a simple Multi-Layer Perceptron (MLP) is used for the comparison with the other models. The goal is to evaluate if the employment of Neural Networks can be a future research area.

The perceptron is inspired by the biological neuron in the brain and is the basic processing unit of a neural network. Each perceptron has one or more inputs from the previous perceptions or the environment and computes a weight sum of these inputs. The weight determines if a perceptron triggers a certain activation function and 'fires' or not. The activation function is often a simple sigmoid function which determines a threshold for the perceptron to activate (**?**, pp. 271–273).

The MLP is a feedforward neural network which means that is organized into layers, with information moving in one direction from the input layer through the hidden layers to the output layer. These layers are the input layer, one or more hidden layers and an output layer (**?**, pp. 279–280). The input layer is the network's first layer and is in charge of receiving input data. It contains the same number of nodes as the number of features in the dataset (**?**, p. 105). In this example, the neural network's input layer would be made up of three perceptrons, each of which corresponded to one of the independent features, namely thickness, die opening, and punch penetration. The input layer does not perform any computation and only passes the input data to the next layer, which is **??** is a hidden layer.

In a fully connected neural network like the MLP each neuron in the input layer is connected to each neuron in the first hidden layer. While doing this each neuron multiplies the input values with the weight value associated with the connections denoted as w_i The output layer of the neural network does produce the final output of the network.

The output is calculated based on the weighted output produced by the last hidden layer (**?**, p. 106).

Setting the weights to a specific values is the task of the training process and is the actual 'learning' part of the MLP. During training the algorithm adjusts the weights of the connections between the neurons in order to minimize the error between the predicted and actual output. A backpropagation algorithm is typically used to train **MLP!** (**MLP!**) models, by propagating errors backwards from the output layer to the input layer in order to adjust the weights (**?**, p. 454).

This is done in multiple steps, **?** make and example with a network to recognize handwritten digits, this example is simplified and applied on the MLP model used in this study ((**?**, p. 12–24)):

1. The input data is passed through the network, layer by layer, until it reaches the output layer and a result is calculated. This process is known as the forward pass.
2. The error is then calculated by comparing the predicted output with the actual output. For example, if the projected spring back is 0.5 and the actual spring back is 0.6, the difference between these values is the error which can also be negative.
3. The error is propagated backwards through the network, from the output layer to the input layer, to calculate the gradient of the error with respect to the network's weights and biases. To reduce the error, the weights are then updated in the direction of the negative gradient using an optimization algorithm such as gradient descent.
4. This process is repeated iteratively for all samples in the training set, updating the weights and bias steps after each batch, until the error function reaches a minimum value or a stopping criterion is met.

In summary, backpropagation is a process that uses the error between the predicted and actual output to update the weights and biases of a neural network. By iteratively updating the weights and biases, the network can learn to make more accurate predictions (**?**, p. 53–57).

Usually a weight x_0 is included in the perceptron model as an intercept value to increase its generality. It is typically represented as the weight coming from an additional bias unti that always has a value of +1. **??** show a simplified visual representation of an MLP used in this study. It uses the tree input V, t, y_p from the used dataset, the perceptrons in the input layer are denoted as x_i, the perceptrons in the hidden layer as a_i and the perceptron in the output layer as $f(X)$. The weights of the connections between the perceptrons are denoted as w_i.

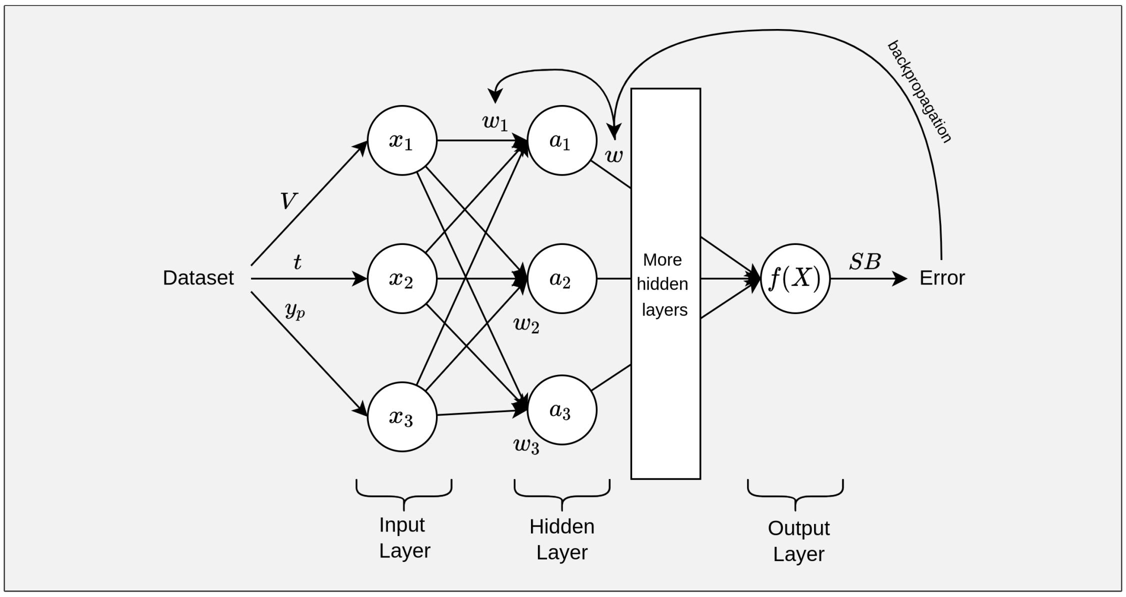


Figure 4.10: Visual representation of a Multi-layer Perceptron. Modeled after decription of **?** and applied on the scenario of this study (own repsentation).

4.4 | Model Training

Supervised learning involves training machine learning algorithms a dataset which contains the inputs features as well as target feature in this study spring back. In order to evaluate how well the model generalizes to new, unseen data, it is necessary to split the dataset into training and testing sets. This involves taking some samples from the original dataset and using them exclusively to test the trained models.

4.4.1 | Training-Test Split

?? 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, but this 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 on the same dataset. For algorithms with hyper-parameters, Grid-Search Cross-validation was used to set the them accordingly. The full hyper-parameters for each model are listed in .

4.4.2 | Setting the Hyper-Parameters

Prior to training a machine learning model, users of machine learning implementations like the ons found in `sci-kit learn` generally need to manually choose a Machine Learning algorithm and configure one or more model parameters, known as hyper-parameters (**?**, pp. 1). The chosen hyper-paramters significantly influence the performance of the resulting model and difference a lot from use case to use case (**?**, pp. 1). As a result, it is critical to set the proper parameters.

A few examples of hyper-parameters for a Random-Gorest modelcitescikit-learn include the number of trees, the number of features to consider when looking for the best split, and the maximum depth of the tree **?**. The use of a grid-search cross-validation is a popular method to discover the best hyper-parameters. This is the approach used in this study and was described in **??** 'Grid Search'. The hyper-parameters utilized for each model are listed in **??** to allow the reader to duplicate the outcomes of this study. This data can also be seen in the study's source code.

4.5 | Result of the Build-Phase

The design and development of the machine learning models have been completed, marking the end of the 'Build' phase in this study. All models were implemented using the libraries listed in **??** and the source code is publicly available on GitHub at:

`https://github.com/Philipp0205/Master-Thesis-Code`

Each model has its own folder under the sub-directory `learner`, containing the model and any additional files associated with the learner. To run the model, simply execute the `main` function. The dataset used in **??** "Dataset Generation" and subsequent sections can be found in the sub-directory *data/dataset*. The dataset is structured into different folders for each die opening used to generate the data. Additional files for the dataset, such as a visualization of how the spring back was calculated for every single bend, can also be found in that folder.

5 Evaluation

This chapter represents the evaluation phase, which includes the evaluation, demonstration and communication activities outlined in the **DSR!** process in **??**.

The evaluation activity provides a comprehensive analysis of the Machine Learning models built in the previous chapter. This phase evaluates the effectiveness of the developed models in solving the defined research questions. During the demonstration phase, different scenarios are chosen to illustrate the potential real-world applications of the models and their performance in these contexts. Finally, during the communication phase, the results of this study will be presented with the aim of effectively communicate the results and their interpretation.

?? provides an overview of the quality metrics employed for the evaluation, structured based on the **GQM!** approach discussed in **??**. The quality metrics align with the quality model from **?**, which in turn is based on the ISO/IEC 9126 standard for software quality evaluation. This standard has been adapted by **?** to address the specific requirements of machine learning models, as detailed in Section **??**. Since **?** primarily focus on evaluating classification models, the quality metrics have been tailored to suit the needs of the regression models implemented in this study. Each individual quality metric section provides further elaboration, on what was changed. **??** also displays the quality metrics used for evaluating the **ML!** models.

Goal	Question	Metric
Correctness	How is the ability of the model to perform the current task measured on the development dataset and the runtime dataset (?, p. 16)?	MAE, MSE, RMSE
Relevance	Does the bias-variance tradeoff work well for the model? Which means that the data should not be overfit or underfit. (?, p. 16)?	Variance of CV, R^2
Robustness	Ability of the model to outliers, noise and other data quality issues (?, p. 16)	Loss of Accuracy, Average Loss
Stability	Does the artifact generate repeatable results when trained on different data? (?, p. 16)	LOOCV stability
Interpret-ability	How well can the model be explained? (?, p. 16)	Different interpretability methods
Resource utilization	How much resources are required to train and run the model? (?, 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. **?** used the classification metrics precision, recall and F- score to evaluate the correctness. Since predicting the spring back is a regression task, these metrics are not applicable. For this study metrics are needed to measure how well the predicted spring back for a specific observation matches the actual spring back (**?**, p. 29). These metrics used in this study are the MAE, MSE and RMSE and where presented in **??** For the evaluation the *sci-kit learn* (**?**) implementations of these metrics were used.

There is an ongoing debate regarding the best metric to use for evaluating regression models. For instance **?** contend that the RMSE is not an appropriate measure for determining the average performance of a model and suggest using the MAE instead, while **?** argue in favor of the RMSE. The MAE is commonly used when a dataset contains outliers. However, as the dataset reviewed in **??** is likely to not contain many outliers, the RMSE will be used as the primary metric in this study (**?**, p. 1249). The MSE and MAE will be used as well but only when they make sense, for example when noise is added to the dataset and therefore more outliers (**??**) are present or if the non squared error yields more meaningful results.

Both measures will be included in the analysis, to give the reader a better understanding of the performance of the models. The MSE was not included in the analysis, as it is not expressed in the same unit as the data and is therefore more difficult to interpret and might be misleading.

How much error is acceptable?

An important question to ask when evaluating the performance of a model is how much error is acceptable. This is a domain specific question and therefore tolerances in the domain of sheet metal forming are used. The DIN 2768 norm provides different mechanical precision levels and tolerances, Table three describes the tolerances for angular dimensions formed metal components. The norm does also differentiate between different levels of precision, the highest precision level is f (fine) followed by m (medium), c (coarse) and v (very coarse). The norm does differentiate between parts of different side length, since the metal sheets used in this study are 80 mm long and are bent in the middle the side length is 40 mm each. For metal parts between 10 mm and 50 mm side length the fine and medium tolerance is ±0°30′minutes and degrees This translates to a ±0.5° tolerance of the opening angle.

That means the **ML!** models should be able to predict the spring back accurate enough so that the bended angle of the metal part is within the tolerance of ±0.5°. Since the spring back is measured in millimeters in the dataset in a first step it has to be converted to degrees to apply the tolerance.

5.1.1 | Results

?? presents the results of the evaluation of the Machine Learning models for correctness, sorted by the RMSE in ascending order. The results where calculated using 5-fold cross validation and the mean of the RMSE and MAE is presented. It can be can be concluded that the best models have an RMSE of less than 0.2 millimetres, indicating that they can predict the spring back with acceptable accuracy for many applications.

The Linear Regression and Decision Tree models have poor predictive performance compared to the other models. For the **LR!** model this is expected since the relationship between the independent and dependent variables is not linear. In contrast, Decision Trees can perform poorly when overfitting the training data and they are prone to instability. These problems are mitigated by using ensemble methods like Random Forest and Gradient Boosted Trees. As the simple Decision Tree and Linear Regression models are not able to predict the spring back accurately enough, they are not considered for the next steps of the evaluation. As there are only two algorithms that are deemed interpretable with explanations specific to their respective models , it follows that alternative techniques must be identified for the interpretability assessment to be conducted later. This matter will be deliberated in **??**.

The Ada Boost, Random Forest, and Gradient Boosted Tree models demonstrate moderate predictive performance while the Multi-Layer-Perceptron, Extra Trees, and Support Vector Machine models have relatively low MAE and RMSE values, indicating good predictive performance. It is worth noting that the SVM model has a slightly better MAE while still maintaining a comparable RMSE compared to the other models. This could be because the errors made by the SVM are evenly distributed across instances or because the errors made by the other models are large for a few instances but smaller for the rest.

Model Name	MAE	RMSE	Relative Error
Decision Tree	0.275	0.401	
Linear Regression	0.228	0.283	
Ada Boost	0.250	0.293	
Gradient Boosted Trees	0.193	0.261	
Random Forest	0.220	0.277	
Extra Trees	0.216	0.261	
Multi-Layered-Perceptron	0.161	0.205	
Support Vector Machine	0.128	0.170	

Table 5.2: **Results for the Design Principle 1 Correctness:** The results are sorted by RMSE in ascending order.

As already noted most models have a RMSE of 0.2 which means that the error for a prediction is 0.2 mm. For example if the spring back is 0.5 mm the model predicts could predict the spring back between 0.3 mm and 0.7 mm. To evaluate the the models the question still remains if this is an acceptable error or not. To answer this question the tolerance from the DIN 2768 which is 0.5°. In **??** multiple sample from the dataset where chosen and their tolerance was calculated. The examples chosen are the same as in **??**