Cloud Computing and Data Science
https://ojs.wiserpub.com/index.php/CCDS/

UNIVERSAL WISER PUBLISHER

Research Article

A Machine Learning Approach for Automated Cost Estimation of Plastic Injection Molding Parts

Florian Klocker[1], Reinhard Bernsteiner[2*], Christian Ploder[2], Martin Nocker[1]

[1]Digital Business & Software Engineering, MCI The Entrepreneurial School, Innsbruck, Austria
[2]Management, Communication & IT, MCI The Entrepreneurial School, Innsbruck, Austria
E-mail: Reinhard.bernsteiner@mci.edu

Received: 23 December 2022; **Revised:** 27 March 2023; **Accepted:** 3 April 2023

Abstract: Market competition leads to shorter cycle times for new or updated products. Therefore, flexibility in reacting to market changes, product development, and all related processes must be accelerated. In this regard, accurate cost estimation in the early stages of product development is critical for assessing the economic viability of a product. However, cost estimation requires data and expertise from several departments. Machine learning approaches could improve the accuracy and reduce the time needed for cost estimation. To investigate the eligibility of machine learning-based cost estimation, a case study was conducted on an industrial company that produces plastic molding parts as key components of its products. The study involved training various supervised machine learning algorithms on a dataset of plastic injection molding parts using three different cost calculation methods. The three methods differed in the extent to which they considered the different process steps involved in the production of the parts. Different tree-based machine learning regression models and neural network models were trained to identify the most suitable approach for cost estimation in the given context. The results showed that tree-based machine learning algorithms outperformed neural networks and that individually predicting manufacturing parameters for cost calculation of each manufacturing process step leads to the most accurate cost estimation. This paper demonstrates how machine learning can support cost estimation in the early stages of the product lifecycle, reducing development times and improving cost estimation accuracy.

***Keywords*:** cost estimation, neuronal network, machine learning, case study, plastic injection molding, industrial manufacturing, product life cycle

1. Introduction

The target cost estimation of a new product provides the first indications of its economic viability. Furthermore, some preliminary questions will be answered after the very first estimation. They include the determined target costs, the cost drivers, the minimum market price needed to be profitable, and thus also the market potential. Such cost estimations are done in several iterations defined by a continuous Product Innovation Process (PIP). The PIP involves several milestones, beginning with a market analysis and leading to a finished, saleable product. Each milestone is terminated with a design freeze, a target cost estimation, and an economic viability analysis.

This paper presents the cost estimation results early in the product development cycle. Due to legal restrictions and

DOI: https://doi.org/10.37256/ccds.4220232277

potential competitive advantages, it is not allowed to publish the company's name. The company has several branches and about 9,500 employees worldwide. The annual quantity of one plastic molding product ranges between 200,000 and 1,000,000 pieces per year.

The PIP implemented in the company can be described as follows: The development of a new product is usually initiated by the product management team located in the marketing department. After a profound market analysis, several data sources serve as a starting point to start the PIP with its predefined milestones. To ensure data consistency at each milestone, the design of the various mechanical parts is frozen and represented as Computer-Aided Design (CAD) models. For each design freeze, components are reviewed more precisely in terms of their feasibility by the engineering departments. Based on the provided data from the engineering departments, key cost figures can be delivered by the costing department. This cost estimation is performed by using logic-based systems and rules. The described iteration takes approximately one to two months to collect and provide all data, and cost estimation can be performed. During this time, the product development process continues. Consequently, technical cost modifications can only be addressed afterward.

The major shortcoming of the described cost estimation approach is that product development and costing processes are separated in time. One reason is the intermediate departments, which estimate the manufacturing parameters based on the product design, such as the number of cavities, labor ratio, cycle time, and tool costs. That leads to a delay of roughly one month, depending on the number of parts.

Thus, the cost estimation has to be accelerated, and the seamless cooperation between product development and costing needs to be promoted. In addition, early-stage cost calculations do not need to be as accurate as they should be when the product is launched. Cost estimations in the early stages should guide product development in the right direction and improve decision-making in cost engineering [1, 2].

One aspect of achieving closer integration between product development and cost estimation is the introduction of machine learning approaches, which aim to reduce the development time of new products. Since this is a new approach for the company, first tests and prototypes are applied to get first insights. The company has defined the following requirements: The estimated cost should be accurate, and the system must fit the already available data. Furthermore, employees from the related departments must be able to use the system in their daily routines. From an economic perspective, the system should directly estimate the total cost of a product. Therefore, several tree-based machine learning models and Neural Networks (NNs) have been trained and evaluated for cost estimation.

The paper is organized as follows. After the introduction, which overviews the research project and its requirements, Section 2 introduces the theoretical foundations in the fields of machine learning, cost calculation methods, and the product development process implemented in the company. Moreover, related work and the aim and research question of the study are presented. Section 3 outlines the methodology employed to address the research question and provides details on the data structures and the three cost calculation methods employed in the empirical part. Section 4 introduces the measurement metrics and presents the evaluation and interpretation of the results. Finally, limitations are stated in Section 5 before the paper is concluded in Section 6.

2. Theoretical background

The following sections provide the theoretical background and related work for this paper. It covers machine learning algorithms and internal cost accounting approaches. Furthermore, cost calculation methods of plastic injection molding parts are addressed, and their manufacturing technology is briefly described.

2.1 *Machine learning algorithms*

Mitchel defined Machine Learning (ML) as "a computer program is said to learn from experience *E* concerning some class of tasks *T* and performance measure *P*, if its performance at tasks in *T*, as measured by *P*, improves with experience *E*" [3]. ML involves the development of algorithms and models that enable computers to automatically learn patterns and insights from data without being explicitly programmed. A solution should be derived from existing data.

Based on the task and the associated way of building experience, ML can be divided into three types: supervised learning, unsupervised learning, and reinforcement learning. Supervised learning aims to find a relationship between

input and output variables. The input and output variables are referred to as attributes or features. These features range from simple data structures like discrete or categorical data to more complex datasets such as images, texts, or audio signals. Supervised learning is predominantly used to make predictions for new data samples [4]. Unsupervised learning aims to find patterns and relationships in data that are not labeled, and therefore it is often used in pattern or knowledge discovery [4]. Reinforcement learning tries to train an agent to take actions in an environment towards a predefined goal. Each action is assessed with a positive or negative reward, with the purpose of maximizing the reward. This procedure results in trial-and-error learning.

Tree-based machine learning algorithms are effective and widely used techniques for addressing a wide range of prediction and classification problems. These algorithms construct decision trees or ensemble models of decision trees that recursively partition the data space into smaller regions, each corresponding to a specific outcome or class. Each node in a decision tree represents a decision rule based on a specific feature, which is used to split the data into smaller subsets. The terminal nodes, or leaves, of the tree represent the final predicted values. Decision trees are particularly useful for gaining insights into complex data sets due to their simplicity and interpretability, whereas ensemble models such as random forests and gradient boosting machines can achieve higher predictive accuracy by combining the outputs of multiple decision trees.

Artificial neural networks are formed from artificial neurons whose functioning is similar to the neurons of the neural system of a living being, intending to imitate it. In such systems, layers of neurons are connected to neighboring layers of neurons through synapses to transmit stimuli and trigger responses. Through conscious and unconscious learning, the connections between neurons are built and broken down, and their weight is changed. An artificial neuron replicates this principle. Each input value is individually weighted and summed with a bias. The result is transformed into the output value by an activation function. An artificial neural network consists of at least one input layer with at least two input neurons and one output layer with at least one output neuron. In between, there can be any number of hidden layers.

2.2 *Cost calculation schema in the company*

The costs of components, semi-finished parts, and finished goods of the company are calculated based on a surcharge cost calculation schema, as shown in Figure 1.

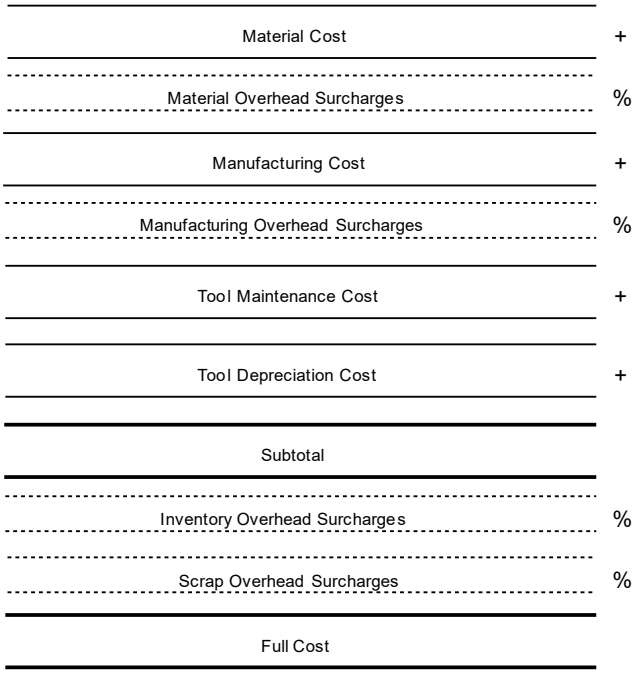

Figure 1. Surcharge costing schema

Material costs cover raw materials such as steel, zinc, or plastic granulate, as well as purchased parts. Manufacturing costs are only incurred in in-house production. For their estimation, a detailed routing structure is required, which includes a set of all manufacturing parameters combined with a complete Bill of Material (BOM). These include, e.g., machinery type, production time for a certain quantity, and the amount of labor during production. Depending on the manufacturing technology, different influencing factors determine the level of manufacturing costs. In addition, one of the major influencing factors is the annual target quantity, which determines the level of automation in production.

Furthermore, as automation increases, indirect factors have an additional impact on manufacturing costs. A higher degree of automation requires a more extensive molding tool and a giant production machine. A larger machine leads to higher operating requirements, directly influencing the labor needed to manage them.

Tool costs are generally divided into tool maintenance and tool depreciation costs. Such tool costs are highly related to the manufacturing technology, the complexity of the parts, and the target quantity. A tool is associated with a standard maintenance overhead, which has to be distributed over the annual quantity produced. In this context, if the amount is quite limited due to restrained purchasing behavior, the cost per piece increases. The cost that cannot be allocated directly to the product is distributed through overhead surcharge rates. This cost includes, e.g., administrative staff, unused production areas, the cost of scrap, or even inventory costs. An enormous administrative overhead would be required throughout the company to allocate such costs directly to the product. They have been bundled appropriately and added as a percentage surcharge according to their origin [5].

2.3 *Cost estimation process in product development of the company*

When developing a new product, a profound market analysis is conducted, which defines the target customer market, the customer profile, the potential sales volume, the minimum market selling price, and the desired target costs. These findings are bundled, and a new product development cycle is initiated according to predefined specifications. Fulfilling the product specifications and the target costs is essential to being economically successful and satisfying the customer's needs. These activities follow the predefined Product Innovation Process (PIP), which is structured into several milestones. Each milestone pursues a predefined outcome to achieve a finished product for market launch. Figure 2 provides an overview of the PIP.

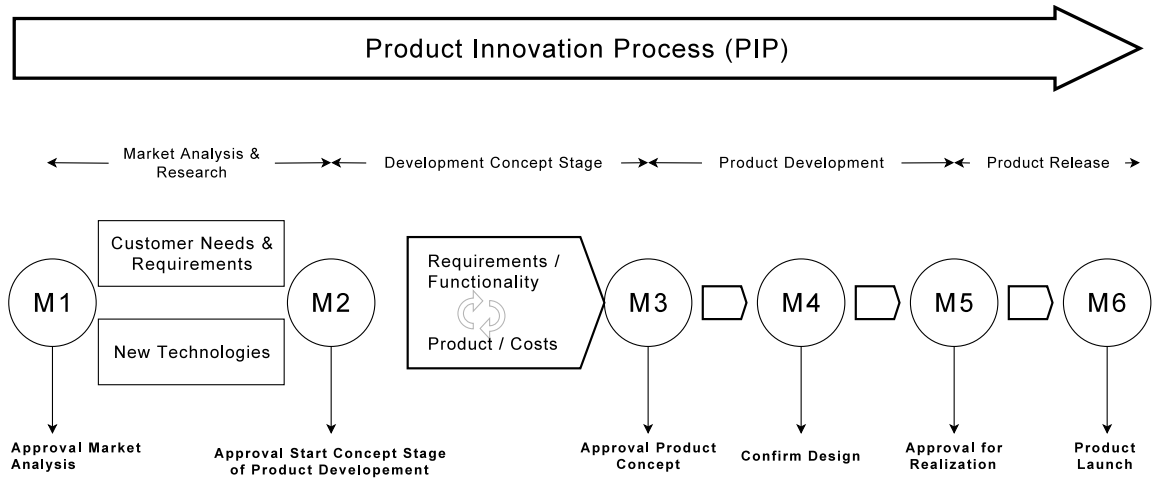

Figure 2. Product Innovation Process (PIP)

The initial three iterations of the PIP address the concept stage of development, dealing with different product ideas and setting the path for further activities. During this stage, it is necessary to make a reliable cost prediction to evaluate the economic viability of the new product. As product detail and complexity increase, modification costs of the development become increasingly expensive [6]. Milestone five (M5; Figure 2) serves to approve tool and equipment

procurement and start with an active marketing strategy.

The following provides a more detailed explanation of the target costing feedback loop, shown in Figure 3. To effectively perform cost estimation, a design freeze of the product concept, the estimated data from the engineering department, and expertise in costing and its structure are required. During the concept stage, the development department attempts to satisfy the requirements by designing several concepts using CAD software. Roughly two months before milestone completion, the CAD design is frozen and delivered to the engineering department. Depending on the required manufacturing technology, the parts are analyzed by different technical experts. In this context, the manufacturing parameters are estimated based on the provided data and forwarded to the costing department closely one month after the design freeze. After that, all these data are converted into calculable parameters and summed up into manufacturing costs. Finally, costs are analyzed, compared, and feedback is given to the development department just before the end of the milestone. Because the concept stage encompasses multiple milestones, this costing process might lead to long delays, making an optimal costing feedback loop challenging.

Furthermore, product development continues without cost information during the cost estimation process. Therefore, the probability of an ever-changing product during milestone completion is significantly high. That implies that potential findings from the cost analysis can no longer be included in the product's design without increased modification costs and additional time delays.

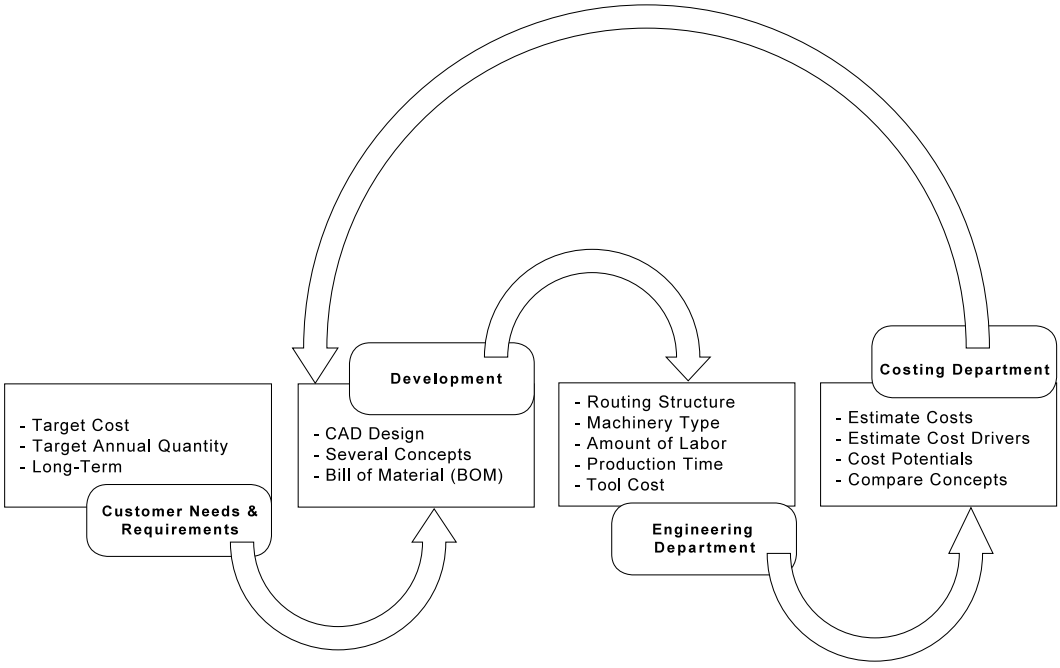

Figure 3. Costing feedback loop in product development

Ehrlenspiel et al. described a cost calculation method that can be committed with a relatively limited workload [7]. In this method, costing is not performed according to typical characteristics, such as cycle time or material input rate, since this data is not yet available at this early stage. To be able to perform such a short calculation, cost estimation is usually based on similarities between existing components. This approach partially overlaps with the company's cost estimation process in the early stages of product development.

A related approach was defined by Niazi et al. in which cost estimation is categorized into a qualitative and a quantitative method [8]. The qualitative method is mainly based on historical data derived from existing components and their costs, and this approach is recommended for products in an early phase of development. According to Niazi et al., costs estimated by neural networks or other ML models also fall into this category. Similar to the qualitative method

of Niazi et al. [8], Ben-Arieh et al. published an analogical approach to cost estimation, relying on data from existing components [9]. Furthermore, Ben-Arieh et al. describe intuitive cost estimation based purely on the experience of the cost engineer.

In this project, a quantitative approach was used with data derived from the production processes of each part. This approach is rather suitable for the parts under investigation.

2.4 *Cost factors of plastic injection molding parts*

Plastic injection molding is one of the most widespread technologies in the production market [10]. This manufacturing technology can produce highly complex product geometries combined with the ever-increasing number of new, more resilient plastic materials [9, 10]. That offers many advantages for product design and manufacturing. Moreover, plastic injection molds can be scaled to almost any required production volume. Injection molding follows a cyclical process, which consists of several stages.

The cost estimation of a plastic injection molding component is based on the calculation scheme shown in Figure 1. The material costs are mainly based on the part's weight and the material type, and some extra material is required for the runner system. In most cases, however, this proportion is of minor importance for cost estimation. Depending on part size, material type, and production volume, material costs can be compared to other cost components. The material costs can be calculated by multiplying the material cost by the product weight [11].

Manufacturing costs depend on various factors. The level of automation plays the most prominent role because it influences the number of cavities, cycle time, amount of labor, and the selected production machine. The higher the production volume, the higher the number of cavities inside the mold, to save machinery capacity and reduce manufacturing costs. The number of cavities usually doubles with an increasing level of automation due to a symmetrical filling of the mold via the runner system. The cooling time of the parts has a significant impact on manufacturing costs. Depending on the size and shape of the part, the cooling time is relatively high and thus increases the costs [10]. Furthermore, different materials have varying melting and cooling characteristics [12].

The machine has to be prepared by mounting the tool on the machine. Calibration and testing have to be performed to start a new manufacturing task. During this time, fixed costs are incurred, which have to be allocated to the produced components. Usually, these costs are referred to as setup costs. Consequently, components with a small lot size will incur high setup costs and, therefore, high manufacturing costs.

Tooling costs can be split into tool depreciation and maintenance costs. Tool depreciation is the internal reduction of the purchase value of the tool. The cost per unit depends on the purchase value and the tool's maximum annual production quantity. The tooling investment costs are strongly affected by the number of cavities and the complexity of the components. Once the part reaches a critical size and complexity, the cavities cannot be further increased since the mold size would become too large for the machine. Increasing part complexity leads to higher tool investment costs and cooling times, negatively affecting manufacturing costs [13]. Maintenance costs depend on the tool's standard maintenance effort and factors that lead to additional maintenance, e.g., wearing out a stamp or a cavity in the mold.

Overhead surcharges are added to the direct costs, which correlate directly with the cost factors mentioned above to calculate the full cost of a part. In this paper, mainly full costs are addressed. In this context, full cost (also referred to as total cost) is the sum of variable and fixed cost components [11, 14].

2.5 *Related work*

Some studies have already demonstrated that accurate cost estimation with ML is feasible [15-18]. Wang et al. published a study on cost estimation of plastic injection molding parts using the backpropagation of an Automated Neural Network (ANN) [19] combined with the Particle Swarm Optimization method. This approach was defined by Eberhart and Kennedy [20], who used 2,100 groups of data, including 1,260 for training, 820 for testing, and 20 for validation. An ANN was used with one input layer, including eleven input units, two hidden layers, and one output layer representing the part cost. The training was performed over several cycles with Mean Squared Error (MSE), the Sum of Squared Error (SSE), Mean Absolute Error (MAE), and Root Means Squared Error (RMSE) converged to minimum and R^2 to the maximum [21]. R^2 converged to a maximum of 0.99785 on test data at 50,000 learning cycles. To determine the prediction accuracy of this validation set, Wang et al. used the Cost Percentage Error (CPE), following Gunaydin et

al. [22], which is defined as

$$\mathrm{CPE}(i) = \frac{E(i) - T(i)}{T(i)} \cdot 100\%, \tag{1}$$

where $E(i)$ represents the estimated costs for sample i, and $T(i)$ represents the actual costs for sample i. 75% of all validation data had a CPE of less than ±0.2%, which are excellent results. In their experiment, only numerical attributes were used, such as density, volume, surface area, part dimensions, and material information. The comparability might be limited since Wang's data set is relatively noiseless across the individual features. Furthermore, Wang's material data consist only of a limited number of material types, and most companies have widely varying prices within each material sub-type.

Brede et al. used a multi-view CNN to predict the mold quotation cost of plastic injection molding parts [23]. They collected and processed 700 samples, represented as CAD data. The data set was split into an 80:20 distribution for training and testing. For the 2D CNN, 20 image projections were derived of each sample from 20 different perspectives. These projections and metadata were inserted into the CNN using multiple convolutional layers with different strides. They were merged and further processed using multiple flattened layers. In contrast to Wang et al. [19], who addressed a regression problem, Brede et al. [23] focused on a classification problem with an output layer of 43 units. The cost data were mapped to the closest class and distributed into 5,000 EUR steps.

Ning et al. proposed an approach to cost estimation using 2D and 3D CNNs [15], primarily predicting the cost of metal parts. In both 2D CNN and 3D CNN, the data were provided from different projection perspectives. Thereby, more than 70,000 samples were generated in each case. Moreover, the voxel representations were supplied in three different resolutions. During the training, the loss of the 3D model converged easier and faster than the image-trained model. The models were compared using the Mean Absolute Percentage Error (MAPE). Their results show that the MAPE decreased with increasing resolution of the voxel representation. However, because of the increasing training effort at higher voxel resolution, another experiment was performed afterward, with a medium resolution (1,283) and more training samples (400,000).

A more recent scientific study addresses the prediction of metal-cutting part cost. Instead of numerical data or image projections, Yoo et al. followed an approach using 3D CNNs [24]. They generated various preprocessed CAD representations, such as mesh, point clouds, and voxel data. A 3D-CNN architecture was trained and compared with several other architectures, including Ning's approach [24]. According to Yoo et al.'s results of their proposed approach, a MAPE of 8.76% could be achieved. The main difference compared to Ning et al.'s approach is that Yoo et al. added additional numerical inputs to the model [24].

Loyer et al. conducted a study in the field of jet engine part manufacturing. They report that recent regression techniques such as GradientBoostingRegressor are up to two times more efficient than NN [25].

2.6 *Aim of the study and research question*

The central aim of this study is to explore if supervised ML for cost estimation is accurate enough to fulfill the company's requirements. That leads to the following research question: "Which supervised ML model is most accurate for cost estimation of plastic injection molding parts in this early stage of product development?"

The lot sizes for the products offer comprehensive data sets for ML approaches. A sample size of 4,036 from different data sources is collected, processed, and analyzed to ensure the quality of this study. A deviation or error rate of ±20% of the predicted cost compared to the cost calculation of the finished product is considered acceptable and accurate. The work by Wang et al. [19] and Loyer et al. [25] provides promising approaches to the objectives of this research project. The first question is whether NNs outperform tree-based supervised ML methods in this context. We trained various tree-based models, and different NN architectures with historical data, compared the models, and evaluated the results to answer this.

3. Methodology

This section describes the data structures and the three cost calculation methods used to train tree-based ML models and multiple NNs.

3.1 *Data collection and preparation*

The more data and the higher the quality of the data used to train ML models, the better the performance of the final model [26]. A set of raw data, such as numerical manufacturing data, including bill of material and routing structure of plastic injection molding parts, are collected from the company's manufacturing unit. Exemplarily, some samples are shown in Table 1. These data serve as the input for calculating the associated costs. Additional data is extracted from the 3D CAD models to better understand the part's properties. The associated CAD model metrics are extracted with Python version 3.8, using the Python numpy-stl library version 2.16.3. Once the data are reviewed and cleaned, a data set of 4,036 samples is obtained.

Table 1. Sample table of routing structure data

No.	Weight [g]	Material Type	Cavity No. [pcs]	Cycle Time [s]	Labor Time [h/1,000 pcs]	Tool Cost [C]	Lot Size [pcs]
1	6.77	PA6 GF	8	17	0.081	127,587	116,480
2	2.42	PA6 GF	8	18	0.082	113,691	160,000
3	3.62	POM	16	19	0.101	301,333	140,000
4	38.48	PA6 GF	8	32	0.111	180,366	172,800
5	1.20	ABS	16	16	0.071	51,970	256,000
6	0,06	PP	16	12	0.061	106,200	320,000
7	1.58	TPE	2	27	0.04	16,568	32,000
...	...	...	...	...	...	...	...
4,036	0.82	TPU	16	23	0.081	105,067	352,000

Exploratory data analysis is performed to identify individual correlations and data scattering across specific features. Data is uploaded to the Microsoft Azure data lake and accessed using a Databricks notebook. The raw data evaluation and the results are performed with Python version 3.8, Pandas version 1.2.4 for data table visualization, and Seaborn version 0.11.1 for plotting, based on the Matplotlib library. To further improve data quality and the overall quality of the models, samples with very low frequencies and specific outliers are removed, resulting in a more balanced data set with 3,554 samples for training and testing.

3.2 *Design of the case study*

A quantitative analysis is conducted to identify the most accurate method for predicting the costs of plastic injection molding parts using supervised ML models. Due to several influencing factors on costs, three fundamentally cost calculation methods are applied, as shown in Figure 4-6.

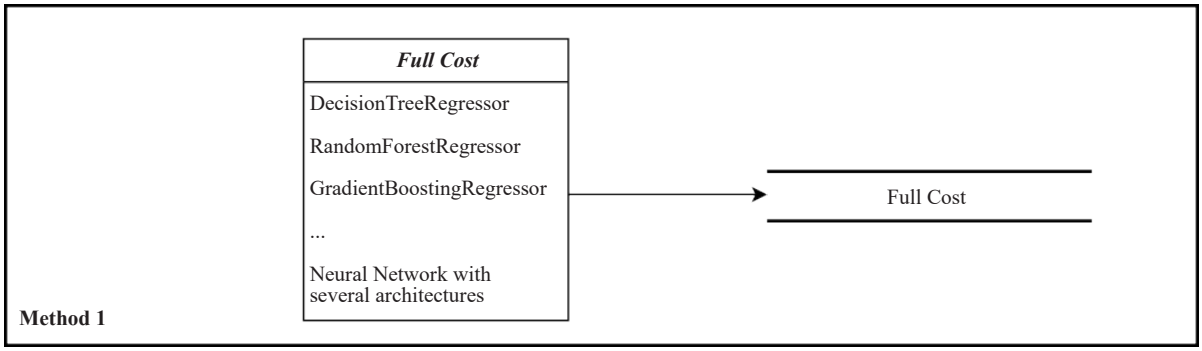

Figure 4. Experiment design of Method 1

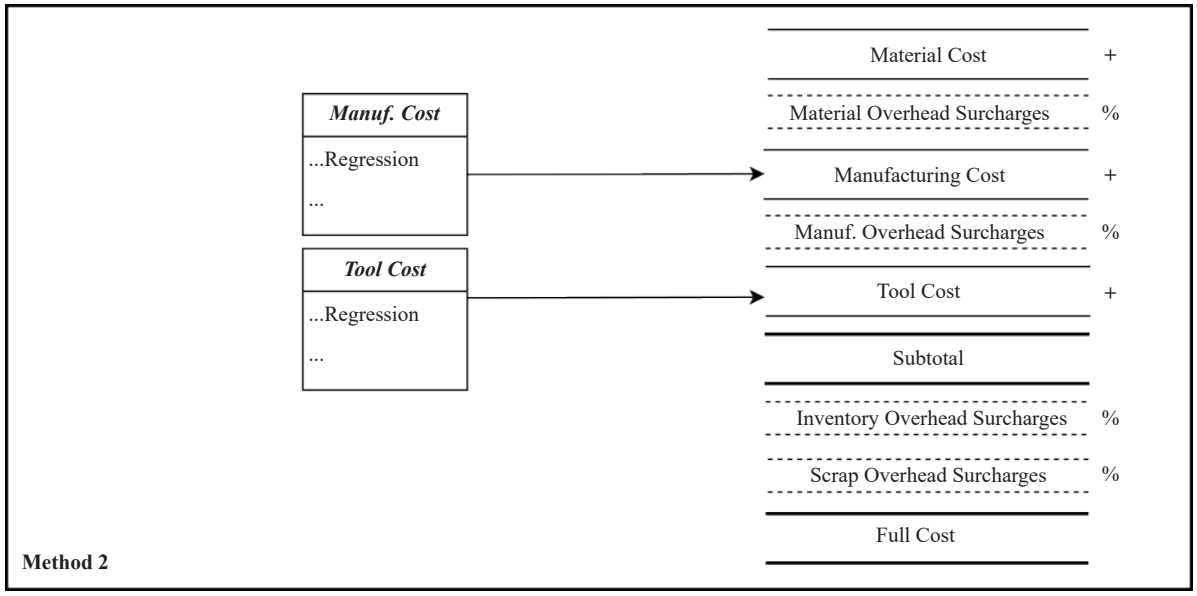

Figure 5. Experiment design of Method 2

The first method (Method 1, Figure 4) is related to predicting the full costs of the plastic injection molding part. Different tree-based regression models and NN architectures are trained with the same data set, and calculating the full cost directly is the preferred method by the company. The second method (Method 2, Figure 5) predicts the individual cost components, manufacturing, and tool costs using different regression models. In this method, material costs are calculated directly from the given part weight and material type, which are provided by the CAD data of the product. The tool cost component includes tool depreciation as well as tool maintenance costs. Consequently, the predicted cost components are increased with the individual overhead surcharges and summed up to the full cost. The third method (Method 3, Figure 6) represents the most fine-grained approach. The cost components for each process step of the routing structure are estimated. The regression models and classifiers are trained and tested with the same data distribution used for the other methods. Full costs are calculated based on the costing scheme as depicted in Figure 4-6.

We primarily use models from the sci-kit-learn ML Python library, version 0.24.1. The NN models are programmed and trained using the Keras deep learning Application Programming Interface (API) version 2.8.0. To train the ML models, the preprocessed data is split into an 80:20 distribution for training and testing. Overall, 14 samples from current product development projects are collected for this case study. Various parts with varying parameters are chosen.

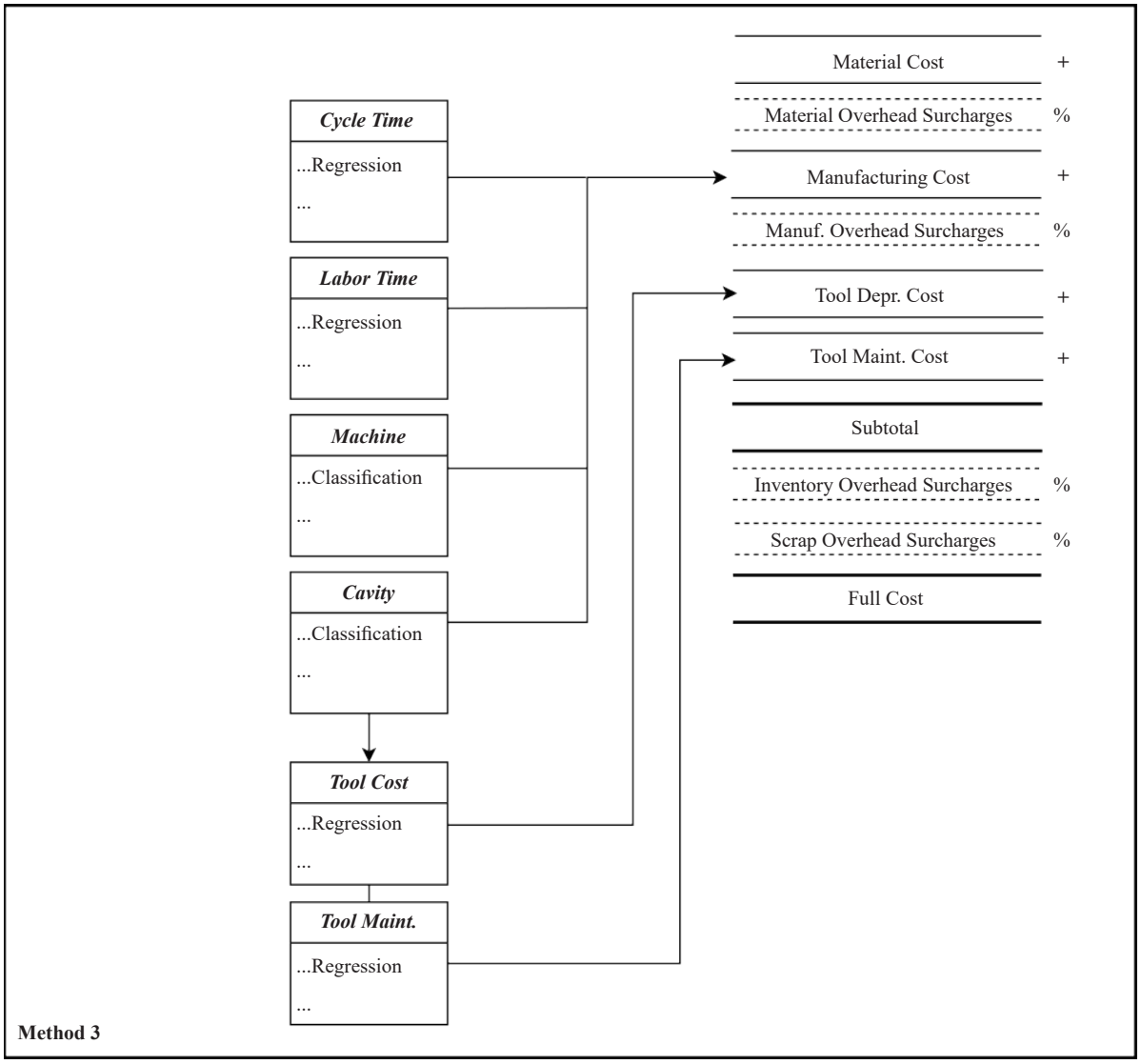

Figure 6. Experiment design of Method 3

Tree-based models are mainly used because these models are versatile ML algorithms and well-suited for more complex data sets with less data preparation [27, 28]. Furthermore, these models do not require any explicit feature selection analysis. The models come from the sci-kit-learn ML library based on the Classification and Regression Tree algorithm (CART) [26, 29].

The Decision Tree regression provides the foundation for the various ensemble models. Different boosting algorithms were applied, e.g., Adaptive Boosting, because it is one of the most potent boosting methods [26, 30]. The fundamental concept of boosting is to train multiple models sequentially, where each successor model tries to improve its predecessor. The model parameters are optimized using Hyperopt [31]. Hyperopt is more efficient than an exhaustive search, such as a random or grid search, due to its optimized search strategy.

Different NN architectures are trained and tested to verify whether NN leads to better results. These architectures differ in terms of the number of hidden layers and the number of neurons in these layers. Since there are different methods to find the optimal architecture, Auto-Keras is used to determine the best network architecture using an optimized search technique [32]. For optimization, the Adam-Optimizer is used.

Different performance metrics are calculated to evaluate the quality and accuracy of the models:

• R^2 (squared) describes the goodness-of-fit and is "the proportion of the variation in the dependent variable that is

predictable from the independent variable(s)" [21].

• Mean Absolute Error (MAE) "represents the average of the absolute difference between the actual and predicted values in the dataset, and measures the average of the residuals in the dataset" [21].

• Mean Squared Error (MSE) "represents the average of the squared difference between the original and predicted values in the data set and measures the variance of the residuals" [21].

• Root Mean Squared Error (RMSE) is "the square root of Mean Squared error and measures the standard deviation of residuals" [21].

• Mean Absolute Percentage Error (MAPE) [33].

Finally, the respective full cost of the three methods is compared and evaluated.

4. Results

First, we answer the question if NNs outperform tree-based ML methods in the given context. Then we present the results of the three calculation methods. Based on these results, the research question can be answered.

4.1 *Method 1a: Full cost prediction with supervised machine learning*

Method 1 directly predicts the full cost. As mentioned, we trained tree-based algorithms with different adaptations. Each model is trained and tested with the same input features to ensure comparability. Table 2 presents the individual models with their results.

Table 2. Evaluation metrics of regressions on the test set (Method 1). Metrics in €/1,000 pcs

Model	Metrics				
	R^2	MAE	MSE	RMSE	MAPE
DecisionTreeRegressor	0.970	25.350	4,039.162	63.554	0.192
RandomForestRegressor	0.964	23.081	4,864.697	69.747	0.146
GradientBoostingRegressor	0.977	24.910	3,126.408	55.914	0.229
AdaBoostRegressor	0.974	19.287	3,407.967	58.378	0.144
XGBRegressor	0.980	18.125	2,683.371	51.801	0.134
LGBMRegressor	0.957	23.739	5,781.792	76.038	0.216
CatBoostRegressor	0.949	28.617	6,842.087	82.717	0.252

The Extreme Gradient Boosting model (XGBRegressor) shows the lowest MAE of 18.125 and MAPE of 0.134 when applied to the test set. The R^2 of 0.980 differs slightly from the Adaptive Boosting model (AdaBoostRegressor) with an R^2 of 0.974 and the Gradient Boosting model (GradientBoostingRegressor) with an R^2 of 0.977. AdaBoostRegressor has a relatively low MAE of 19.287, close to the MAE of the XGBRegressor model, but a significantly higher MSE of 3.407.967. Despite its low MAE, AdaBoostRegressor results in higher prediction errors than XGBRegressor or GradientBoostingRegressor. We analyzed their absolute and negative error proportions. In Figure 7, the error proportion for a particular error rate can be observed for each model.

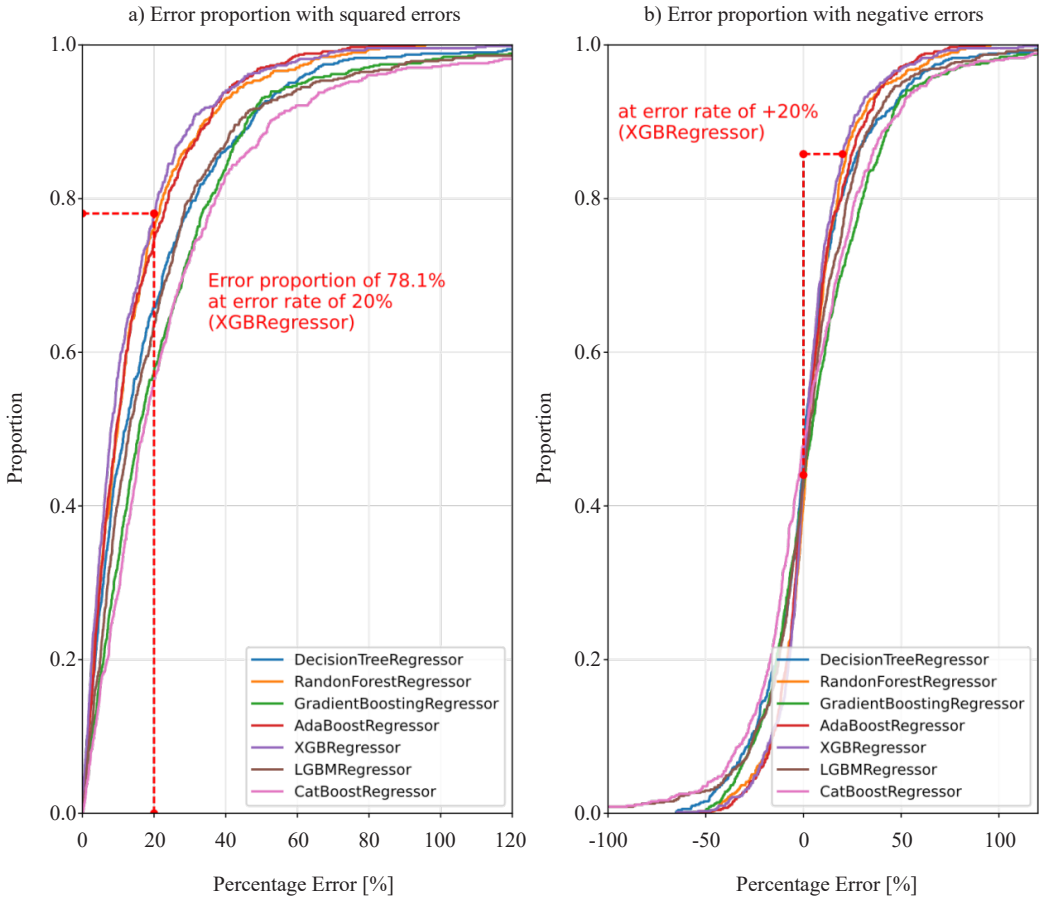

Figure 7. Error proportion of regressions on the test set (Method 1). The ratio was measured at a predefined error rate of ±20%

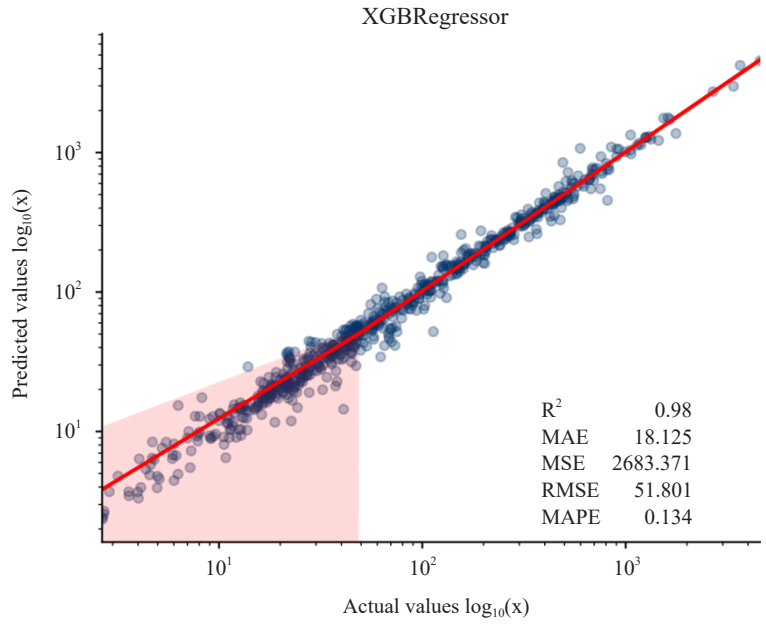

Figure 8. Model fit of XGBRegressor for the test set (Method 1). Actual values (x-axis) vs. predicted values (y-axis) in a logarithmic scale $\log_{10}x$

Figure 7a (squared errors) shows an error proportion of 78.1% for the XGBRegressor for an error rate of +20%. Therefore, 78.1% of test samples have an error of lower or equal to +20% in the case of the absolute error for the XGBRegressor. The AdaBoostRegressor shows a definite error proportion of 75.7% at an error rate of +20% but obtains a higher actual error proportion than the XGBRegressor at error rates over +50%. It can be concluded that the AdaBoostRegressor is slightly more susceptible to errors. The error proportion for non-absolute errors between 0% and +20% is measured at 41.8%, as shown in Figure 7b. It can be said that the error proportion for an error rate between 0% and -20% equals 36.3%. For almost 80% of test samples, the XGBRegressor predicts nearly as many samples to be low as it predicts to be high.

Figure 8 shows the goodness-of-fit and the correlation between predicted and actual values. The red line shows the actual values, whereas the blue dots are the predicted values. The opacity of the predicted values is a measure of their frequency. The predicted values adapt well to the individual features but are slightly noisy under $\log_{10}$ 50. The confidence interval, shown as red colored area, significantly increases.

The XGBRegressor is validated with 14 samples collected from current product development projects, as shown in Figure 9. Only four of the 14 samples (28.6%) are within the defined error rate, and six (42.9%) barely fall below or above. The remaining four samples show a significantly high error rate. Statistically, the XGBRegressor achieved a MAPE of 0.465 and an MAE of 65.098 on the 14 validation samples. Due to the model's appearance as a black box, it is impossible to identify the exact cause of the error since it would be pure speculation. A misinterpretation of the manufacturing costs probably caused the outliers.

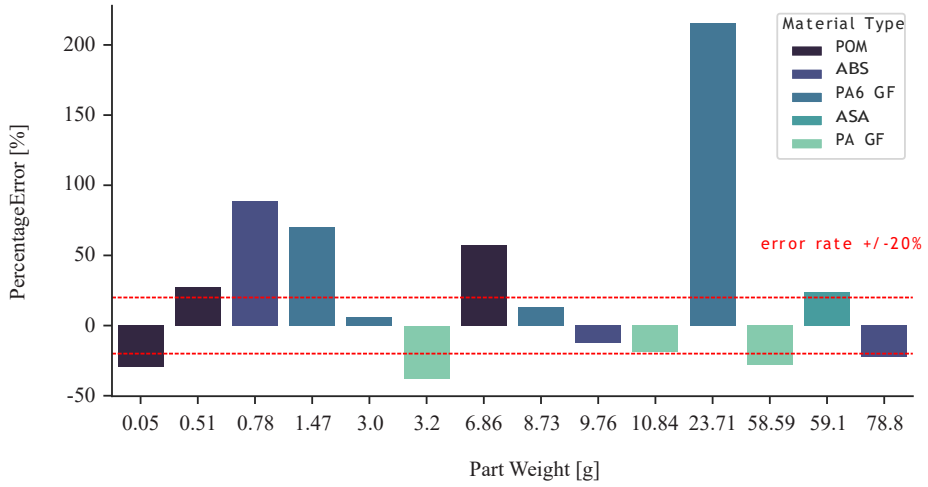

Figure 9. Validation results of XGBRegressor (Method 1)

4.2 *Method 1b: Full cost prediction with neural networks*

Auto-Keras was utilized to explore various neural network structures and determine the optimal architecture. Table 3 presents the evaluated neural networks. Auto-Keras initially configured an architecture with one hidden layer comprising 1,024 neurons, and a ReLU activation function, as well as a single neuron in the output layer, given the regression task at hand. Auto-Keras then inserted additional hidden layers with 32 neurons and ReLU activations iteratively to the architecture. Consequently, the network with ten hidden layers, where the fourth hidden layer had 1,024 neurons, while all other layers had a size of 32, outperformed all other neural networks, achieving an R^2 of 0.511 and a MAPE of 0.708. Adding more hidden layers resulted in overfitting, therefore no deeper networks were considered.

Considering the results of Wang et al. [19], we expected the results of NNs to be significantly better. Nevertheless, they can only be compared to a limited extent due to using a different optimization technique to minimize the loss and a different costing structure.

Table 3. Evaluation metrics of NN architectures. Metrics in €/1,000 pcs

Architecture	Metrics				
	R^2	MAE	MSE	RMSE	MAPE
1 Hidden Layer	0.398	132.864	80,336.913	283.438	1.240
3 Hidden Layers	0.490	94.243	68,037.291	260.840	0.760
6 Hidden Layers	0.501	103.075	66,586.668	258.044	0.771
10 Hidden Layers	0.511	91.633	65,219.788	255.382	0.708

Table 4. Evaluation metrics of regressions on the test set (Method 2). Metrics were measured on a test set with a sample size of 711. MAE and RMSE are measured in €/1,000 pcs

Cost Component	Model	Metrics				
		R^2	MAE	MSE	RMSE	MAPE
Manufacturing Costs	DecisionTreeRegressor	0.950	17.121	3,358.482	57.952	0.215
	RandomForestRegressor	0.956	15.562	2,958.599	54.393	0.185
	GradientBoostingRegressor	0.965	13.985	2,368.050	48.663	0.169
	AdaBoostRegressor	0.952	13.252	3,218.766	56.734	0.158
	XGBRegressor	0.938	14.927	4,157.684	64.480	0.193
	LGBMRegressor	0.949	15.878	3,442.423	58.672	0.275
	CatBoostRegressor	0.941	18.878	3,967.871	62.991	0.388
Tool Costs	DecisionTreeRegressor	0.483	5.491	399.739	19.994	0.516
	RandomForestRegressor	0.622	4.957	292.536	17.104	0.505
	GradientBoostingRegressor	0.669	4.546	256.030	16.001	0.460
	AdaBoostRegressor	0.626	4.212	289.365	17.011	0.417
	XGBRegressor	0.325	7.066	521.933	22.846	0.829
	LGBMRegressor	0.624	5.892	291.096	17.062	0.747
	CatBoostRegressor	0.652	5.136	268.953	16.400	0.613

The comparative analysis of tree-based models and neural networks for predicting the full cost demonstrated that tree-based models outperformed neural networks in terms of our performance metrics. The complexity of the problem and the underlying training dataset rendered neural networks unsuitable for this task. Furthermore, tree-based models are computationally less expensive and have superior explainability compared to neural networks, which is a crucial aspect of the final model as employees must be able to use the system in their daily work. Each node in a decision tree represents a decision rule based on a particular feature, which is used to split the data into smaller subsets. Therefore, decision trees provide a clear and interpretable representation of how the input features influence the output predictions. In contrast, neural networks are more complex and difficult to interpret, as they consist of multiple hidden layers and

a large number of parameters that interact in non-linear ways. Techniques for interpreting neural networks exist, such as gradient-based methods [34] or layer-wise relevance propagation [35], however, they may not provide a complete picture of how the network is making its predictions.

Based on these results, we refrain from evaluating neural networks for subsequent cost prediction methods, and instead focus on tree-based models for the following methods. Overall, these findings highlight the importance of careful evaluation and selection of appropriate machine learning algorithms for specific prediction tasks, taking into account both predictive performance and practical considerations such as computational efficiency and model interpretability.

4.3 *Method 2: Cost component prediction*

This method's basic idea was to eliminate material cost prediction errors to achieve a more accurate model with a lower MAE and MAPE. Therefore, only the cost components, such as manufacturing costs and tool costs, are predicted. Subsequently, the full costs are calculated by adding the material costs using the costing scheme in Figure 1. The performance measures of each model in predicting the cost components on test samples are shown in Table 4. Due to the unbalanced range of values, the results of the two cost components are only comparable to a certain extent. Therefore, MAPE should be used as the leading metric.

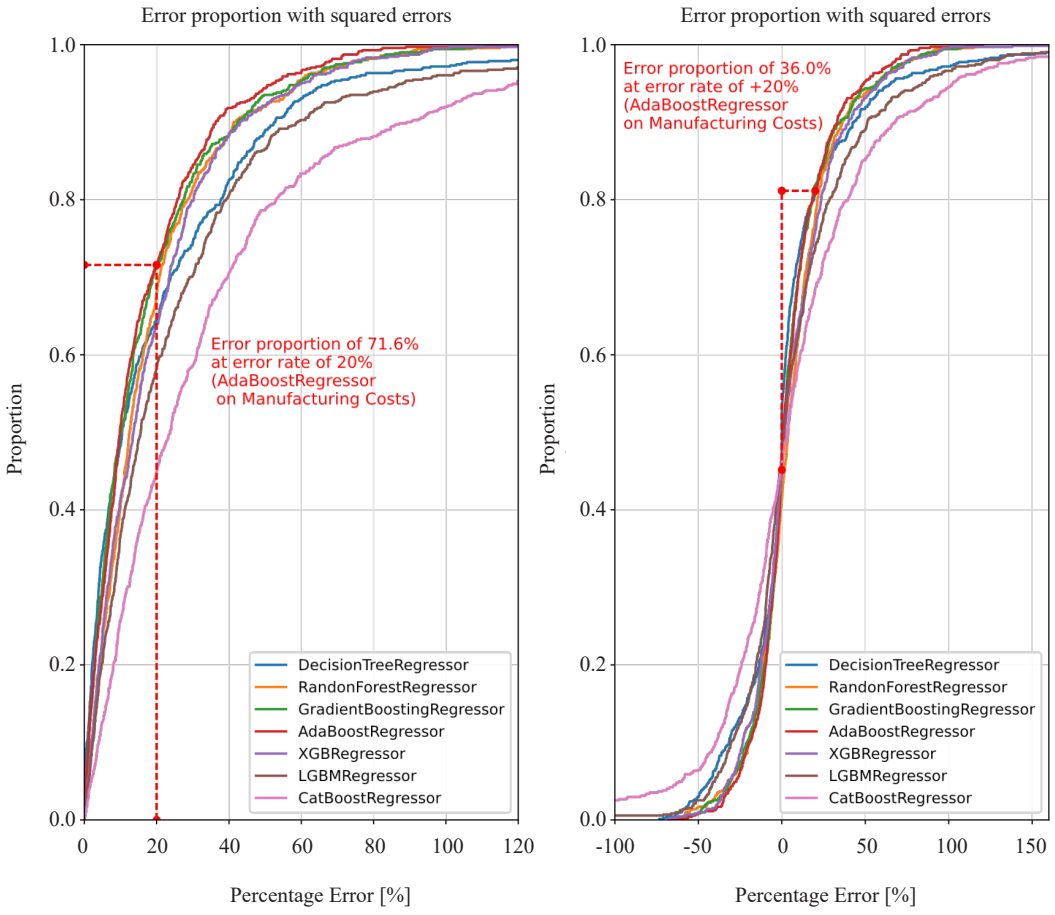

Figure 10. Error proportion of manufacturing cost regressions on the test set (Method 2). The proportion was measured at a predefined error rate of ±20%

Table 4 shows the GradientBoostingRegressor's lowest MSE and RMSE for manufacturing and tool costs. However, based on the MAE and MAPE, the AdaBoostRegressor is the best model for predicting manufacturing and

tool costs because squaring errors overweight outliers. Although the AdaBoostRegressor achieves a slightly better MAE of 13,252 on the test set of manufacturing costs, the MSE of 3,218.766 and the RMSE of 56.734 are significantly higher than the MSE of 2,368.050 and RMSE of 48.663 achieved by the GradientBoostingRegressor. The CatBoostRegressor has the poorest predictive accuracy of manufacturing costs with a MAPE of 0.388.

The AdaBoostRegressor reaches an error proportion of 71.6% at an error rate of +20%, predicting the manufacturing costs shown in Figure 10a. The error proportion for an error rate between 0% and +20% is 35.6% (Figure 10b).

The results in Table 4 show that the tool cost prediction for all metrics has a relatively low goodness of fit. The GradientBoostingRegressor has the highest R^2 score, with 0.669. The model also achieves the most insufficient MSE of 256.030 and RMSE of 16.001. The XGBRegressor seems to be the model with the poorest tool costs prediction scores, with MAE of 7.066, RMSE of 22.846, and MAPE of 0.829. Moreover, the AdaBoostRegressor achieves a comparably low error proportion of 57.0% at an error rate of +20%, shown in Figure 11a. That is, compared to the manufacturing costs error proportion as depicted in Figure 10a, a 14.6% lower error proportion, at an error rate of +20%.

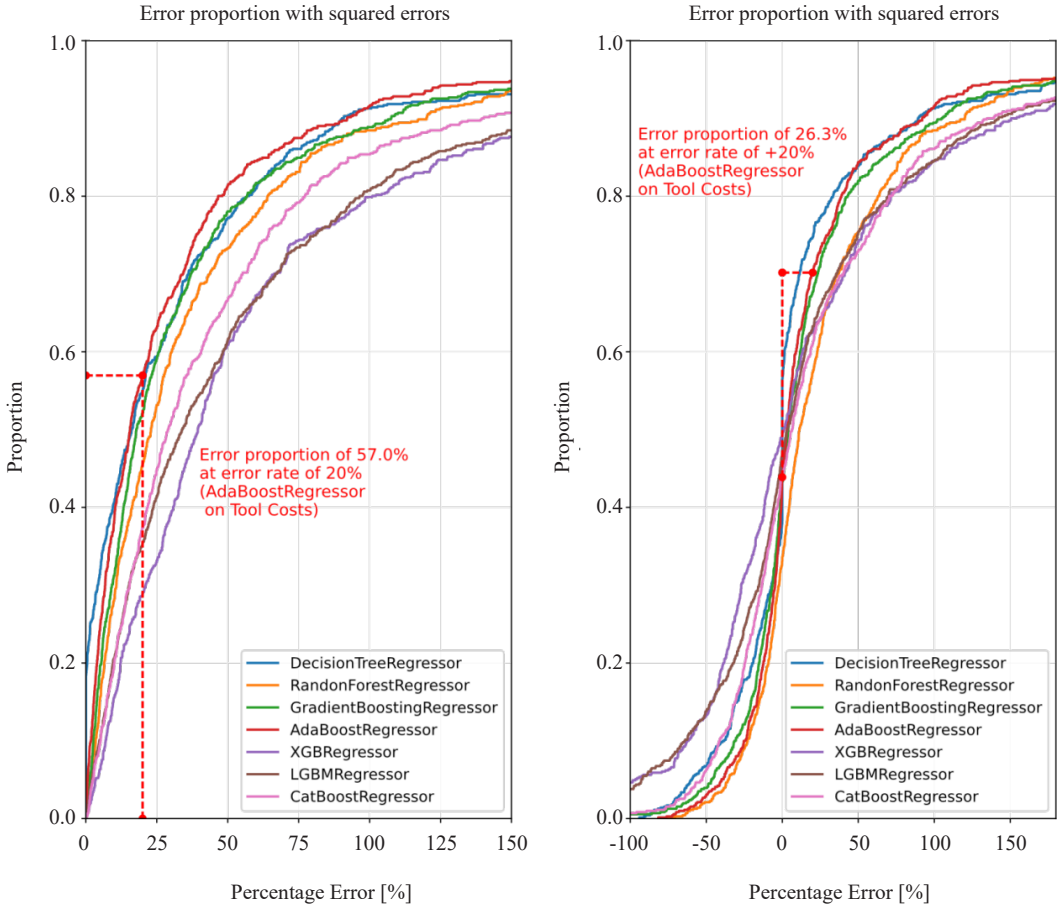

Figure 11. Error proportion of tool cost regressions on the test set (Method 2). Proportion measured at a predefined error rate of ±20%

Method 2 thus reveals the impact of the two cost components on the products' costs. AdaBoostRegressor shows a 9.04 higher MAE for manufacturing costs than an MAE of 4.212 for tool costs. It can be said that even if the tool costs prediction models show a poorer MAPE, the manufacturing cost prediction is mainly responsible for a more significant error proportion of the full cost. Method 2 was stacked and validated with AdaBoostRegressor for manufacturing and tool costs. After manufacturing costs and tool costs were predicted with the stacked model, material costs were

calculated using part weight and raw material prices. Subsequently, the overhead surcharges were calculated and summed up to total full costs, according to Figure 4-6.

Similar to model 1, the goodness-of-fit of the stacked model is plotted in Figure 12. The model achieves an R^2 of 0.969, MAPE of 0.116, and MAE of 17.246. That corresponds to a marginal decrease of the MAE of 0.879 compared to the XGBRegressor of Method 1 shown in Table 2.

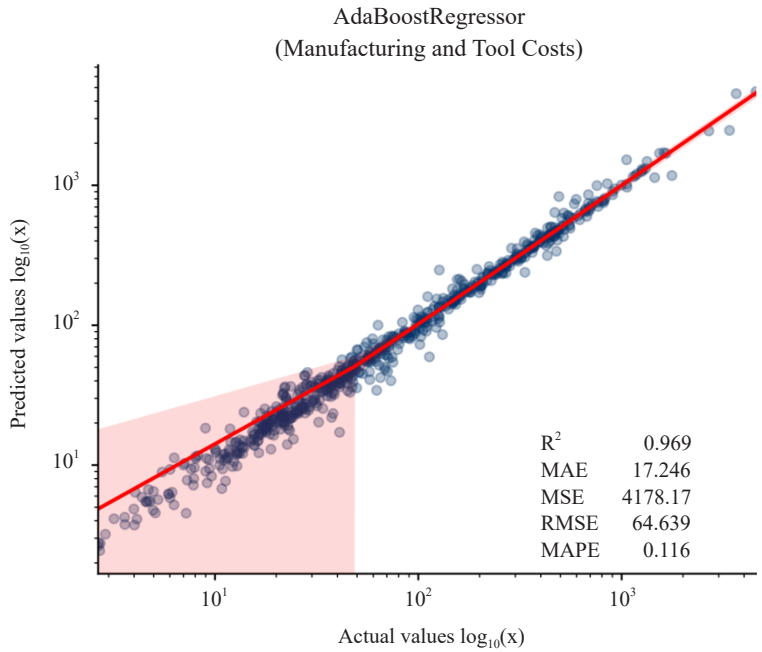

Figure 12. Model fit of stacked model on the test set (Method 2). Actual values (x-axis) vs. predicted values (y-axis) in a logarithmic scale $\log_{10}$x

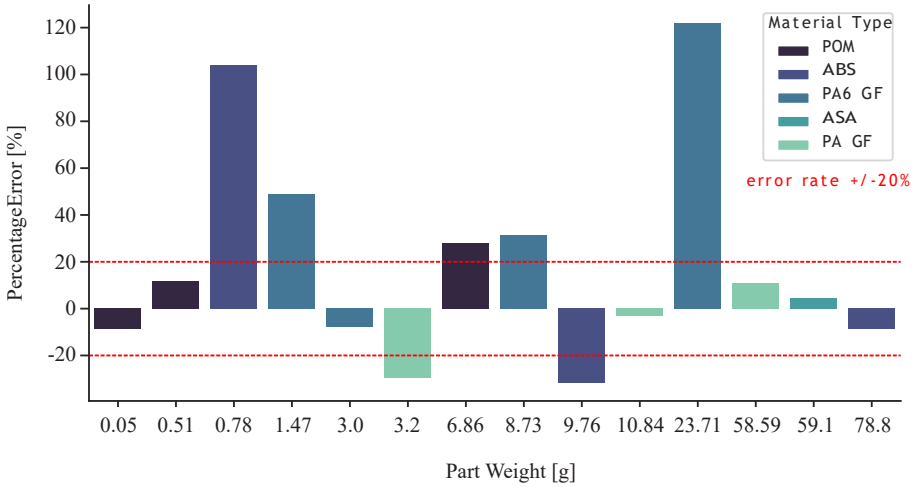

Figure 13. Validation results of stacked model (Method 2). AdaBoostRegressor is used for both manufacturing and tool costs

Figure 13 reveals that seven validation samples (50.0%) fall within the specified error rate. The sample with the highest positive deviation reaches an error rate of 121.89%. In contrast, the sample with the highest negative deviation comes with an error rate of -27.24% - the validation results in a MAPE of 0.323 and an MAE of 36.737. Even though

the error rate is still relatively high, the MAE was decreased by 28.361 compared to the validation results of Method 1, implying that uncertainties of the material cost prediction have been eliminated.

The parts with the most significant errors reach a deviation of over 100%. The component of manufacturing costs primarily causes the high deviations observed in the outliers. Both outliers, which get a more than 100% deviation, were calculated as targets with 16 cavities.

4.4 *Method 3: Cost prediction based on routing structure*

The basis of this approach is to predict only the required parameters needed to calculate the cost, as depicted in Figure 1. Several models to predict the parameters of the routing structure are trained and tested with the same data set used for Methods 1 and 2. The results of the regression models are reviewed and compared in Table 5. They predict continuous parameters such as production times and investment costs. Table 5 shows that the AdaBoostRegressor model is the most accurate for predicting labor time, tool investment, and tool maintenance costs. For cycle time, GradientBoostingRegressor turned out to be a preferable model.

Table 5. Evaluation metrics of regressions on the test set (Method 3). Metrics were measured on the test set with a sample size of 711. MAE and RMSE are measured in €/1,000 pcs

Component	Model	Metrics				
		R^2	MAE	MSE	RMSE	MAPE
Labor Time	DecisionTreeRegressor	0.473	0.021	0.0038	0.0613	0.199
	RandomForestRegressor	0.743	0.018	0.0018	0.0428	0.195
	GradientBoostingRegressor	0.775	0.015	0.0016	0.0400	0.182
	AdaBoostRegressor	0.834	0.014	0.0012	0.0344	0.163
	XGBRegressor	0.743	0.020	0.0018	0.0428	0.233
Cycle Time	DecisionTreeRegressor	0.721	1.958	12.443	3.527	0.106
	RandomForestRegressor	0.841	1.515	7.104	2.665	0.085
	GradientBoostingRegressor	0.851	1.394	6.651	2.579	0.078
	AdaBoostRegressor	0.812	1.550	8.383	2.895	0.084
	XGBRegressor	0.835	1.542	7.363	2.714	0.087
Tool Invest Costs	DecisionTreeRegressor	0.816	22,415.506	-	40,672.167	0.351
	RandomForestRegressor	0.885	18,634.963	-	32,214.278	0.321
	GradientBoostingRegressor	0.903	16,074.153	-	29,542.817	0.283
	AdaBoostRegressor	0.908	15,218.920	-	28,829.453	0.274
	XGBRegressor	0.893	16,148.846	-	31,082.287	0.283
Tool Maintenance Costs	DecisionTreeRegressor	0.466	5.115	349.648	18.699	1.670
	RandomForestRegressor	0.584	4.127	272.736	16.515	1.305
	GradientBoostingRegressor	0.509	3.869	321.778	17.938	1.079
	AdaBoostRegressor	0.643	3.320	233.972	15.296	1.030
	XGBRegressor	0.144	5.982	560.869	23.683	1.642

Labor time is predicted with a MAPE of 0.163 by the AdaBoostRegressor, which is merely 0.07 better than the weakest regression model (XGBRegressor). Because a single worker usually handles several machines simultaneously, the range of values is tiny. GradientBoostingRegressor has nearly the same MAE as the AdaBoostRegressor, but a higher MSE, indicating that the GradientBoostingRegressor causes more outlier errors. Moreover, the AdaBoostRegressor achieves an error proportion of 78.9% with an error rate of +20%, presented in Figure 14a. The test set for cycle time is predicted with MAPE of 0.078 and MAE of 1.394 by the GradientBoostingRegressor, which are good results caused by a positive correlation coefficient of 0.52 relative to part weight.

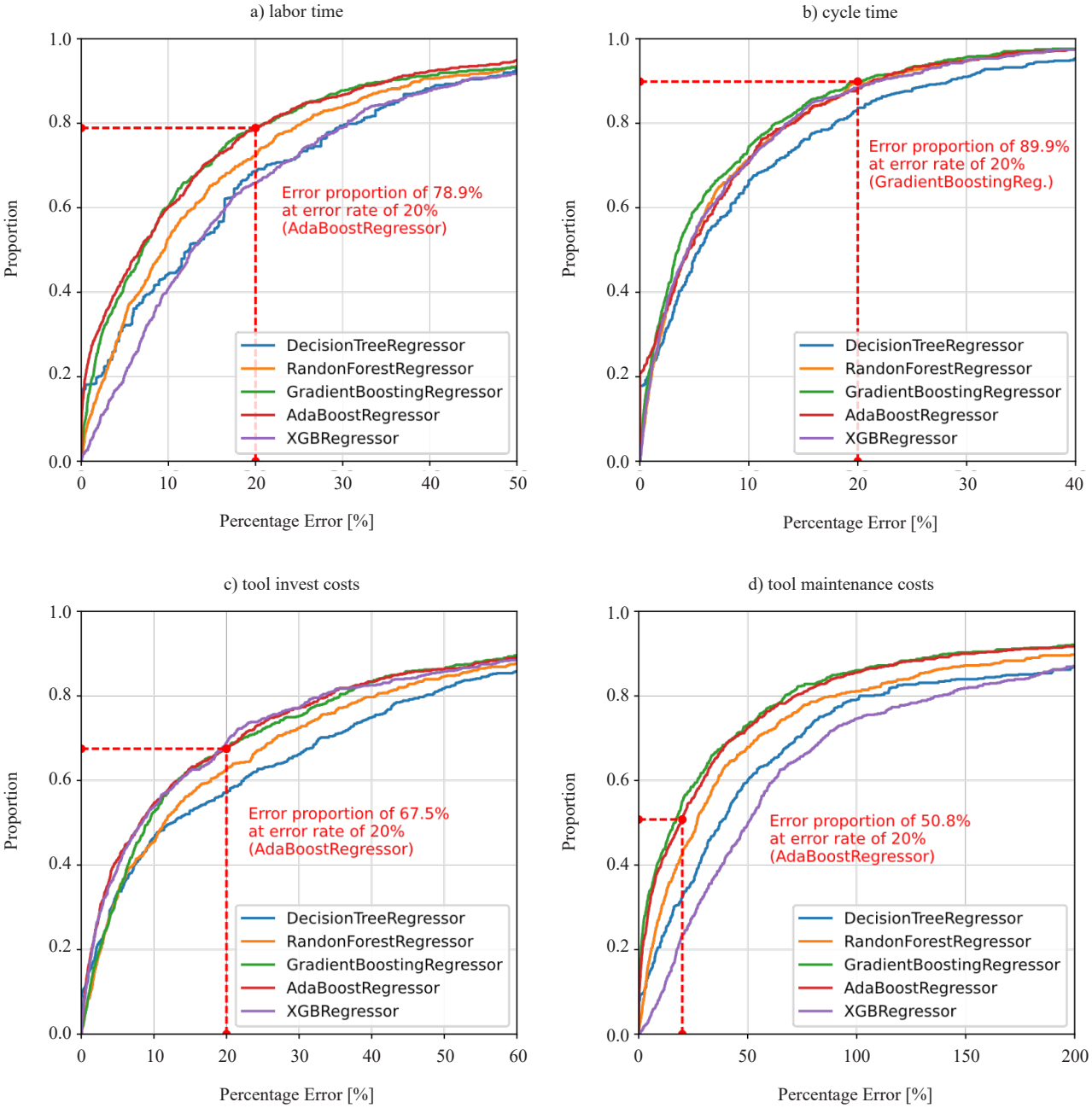

Figure 14. Error proportion of structure regressions on the test set (Method 3). The proportion was measured at a predefined error rate of ±20%

Furthermore, the GradientBoostingRegressor scores a relatively high error proportion of 89.9% at an error rate of +20%, shown in Figure 14b. For tool investment and maintenance costs (Figures 14c and 14d), the number of cavities predicted by the best classifier model in Table 6 is added as a test feature. That should lead to a more accurate prediction since the investment costs depend on the number of cavities. Compared to the previous components, the tool investment costs have a very high range of values. The models achieve a high goodness-of-fit on the test samples. AdaBoostRegressor, in particular, scores an R^2 of 0.908. Predicting tool maintenance costs is one of the most challenging tasks for the models. All models achieve a relatively weak R^2 on the test set. Even the best model, AdaBoostRegressor, merely reaches a MAPE of 1.030, which is 0.679 higher than the poorest MAPE of the other components; DecisionTreeRegressor reaches a MAPE of 0.351 for tool investment costs. This results in a comparatively low error proportion of 50.8% at an error rate of +20%, plotted in Figure 14d.

The individual models of the respective components are compared in Figure 14 to enable comparability. Decision Tree regression is one of the weakest models in all four components. However, this was expected because the Decision Tree represents a simple tree-based model and is highly prone to overfitting its predictions [26]. Both boosting methods, Adaptive Boosting and Gradient Boosting, belong to the most robust models, and they achieve almost equally good results for all four routing structure components. Nevertheless, the drawback of the AdaBoostRegressor is its extensive training delay compared to other models.

Table 6 provides the results of the classification models for the number of cavities and the machinery type. Primarily tree-based models were used for classification since they require very little data preparation and are highly versatile [26]. Additionally, a clustering algorithm was trained and compared (KNeighborsClassifier). Like the regression models, the Adaptive Boost model (AdaBoostClassifier) also seems to be the most promising model for classification.

Table 6. Evaluation metrics of classifiers on the test set (Method 3). Metrics in €/1,000 pcs

Component	Model	Metrics			
		Accuracy	F1 Score	Recall	Precision
Cavity	DecisionTreeClassifier	0.748	0.746	0.748	0.746
	RandomForestClassifier	0.850	0.848	0.850	0.849
	GradientBoostingClassifier	0.857	0.855	0.857	0.858
	AdaBoostClassifier	0.871	0.869	0.871	0.875
	XGBClassifier	0.852	0.851	0.852	0.852
	KNeighborsClassifier	0.734	0.729	0.734	0.732
Machinery Type	DecisionTreeClassifier	0.654	0.652	0.654	0.655
	RandomForestClassifier	0.757	0.749	0.757	0.768
	GradientBoostingClassifier	0.727	0.721	0.727	0.733
	AdaBoostClassifier	0.761	0.751	0.761	0.778
	XGBClassifier	0.723	0.719	0.723	0.727
	KNeighborsClassifier	0.301	0.274	0.301	0.319

AdaBoostClassifier achieves an accuracy of 0.871 for cavity prediction and 0.761 for machinery-type prediction on the test set. As mentioned above, the outputs of the AdaBoostClassifier in terms of cavity classification are used as

features to predict tool investment and maintenance costs. Errors in cavity classification may have led to sequential errors in the regression task. The results reveal that the KNeighborsClassifier is the weakest classifier, with an accuracy of 0.734 for cavity and 0.301 for machinery type.

Like Method 2, the best models of Method 3 were stacked together to calculate and compare the full cost to the test set. Material costs are calculated in the same way as in Method 2. First, a unit time is calculated according to the predicted cavity number and cycle time to estimate the manufacturing costs. This metric is subsequently scaled up using the classified machine's hourly rate. A significant part of the manufacturing costs consists of the machine setup costs. These may be relatively high for small quantities and represent a substantial proportion of the manufacturing costs. During the setup sequence, the machine stands still, and the resulting fixed costs must be distributed over the lot size. These setup costs, the machinery costs during the production, and the labor costs represent the manufacturing costs. To obtain the full cost, they are summed up with the material costs, the tool costs, and the respective overhead surcharges.

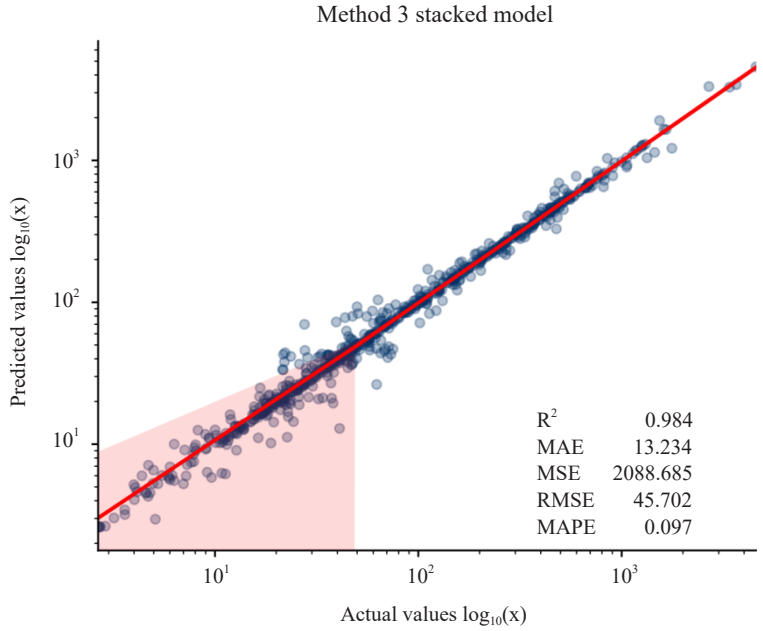

Figure 15. Model fit of stacked model on the test set (Method 3). Actual values (x-axis) vs. predicted values (y-axis) in a logarithmic scale $\log_{10}x$

By stacking the models, the MAE is decreased by 4.012, compared to the MAE of the stacked model of Method 2, as shown in Figure 15. The stacked model achieves a comparably high R^2 of 0.98 and a MAPE value of 0.097. Finally, the stacked model is validated similarly to the previous models. 10 out of 14 samples (71.4%) are within the defined error rate, and the highest error rate reaches 107,25%. The stacked model achieves a MAPE of 0.277 and an MAE of 23.225 on the 14 validation samples. That corresponds to a further decrease of MAE of 13,512 compared to Method 2.

Similar to the previous two methods, the components with weights of 0.78 g and 23.71 g were above the predefined error rate. The cavity classifier model (Table 6, AdaBoostClassifier) predicts a four-cavity tool instead of a 16-cavity tool. The imputed lot sizes probably cause this in the lower region between 100,000 and 40,000. Furthermore, for the PA6 GF part with a weight of 23.71 g, the tool maintenance costs are mispredicted with a deviation of 689%. That is probably due to the giant component dimension (22.6 mm × 86.5 mm × 117 mm), which generally results in higher maintenance costs per piece.

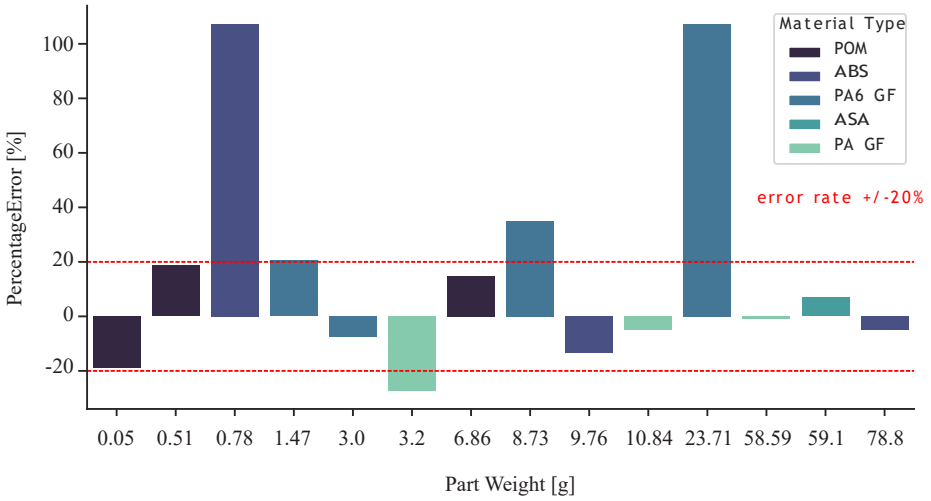

Figure 16. Validation results of stacked model (Method 3)

4.5 *Summary*

Method 1 directly predicts the full cost. As mentioned, we mainly trained and compared tree-based algorithms with different adaptations. In Method 1, the full cost of a product is directly estimated based on available data. The results show that the most accurate model is the XGBRegressor, and it achieves a MAPE of 0.134 and an R^2 of 0.980. Furthermore, the XGBRegressor reaches an error proportion of 78.1% at an error rate of +20%. It was expected that various uncertainties due to overfitting or underfitting could be eliminated by stacking multiple models, thereby achieving a more accurate cost estimation.

In Method 2, several models for predicting the cost components, such as manufacturing and tooling costs, are trained and tested. Subsequently, the predicted costs are summarized with the material costs, and overhead surcharges are added to calculate a product's full cost. Method 2 reveals an RMSE of 64.639, which is 12.838 higher than Method 1. Furthermore, the R^2 of Method 2 is slightly lower than the other methods, as depicted in Table 7. Consequently, the cost components' prediction leads to a minor improvement in the estimation accuracy, according to a MAPE value of 0.116.

In Method 3 (see Figure 16), all routing structure data are predicted. These parameters are used to calculate the full cost based on the costing scheme presented in Figure 1. Method 3 showed that the MAE value of 13.234 could be decreased by 4.891, compared to Method 1. Moreover, the good-of-fitness with an R^2 of 0.984 is slightly better. A significant improvement was also observed in the error proportion, which increased by 8.0 percentage points compared to the error proportion of Method 1.

Table 7 summarizes the results of the three cost calculation methods. It can be concluded that the prediction of individual components leads to a more accurate full-cost forecast of plastic injection molding parts.

Table 7. Comparison of results for hypothesis 2. Error proportion measured at an error rate of +20% for absolute errors. Metrics in €/1,000 pcs

Method	Model	Metrics					
		R^2	MAE	MSE	RMSE	MAPE	ErrorProp.
Method 1 (Direct Full Cost)	XGBRegressor	0.980	18.125	2,683.371	51.801	0.134	78.1
Method 2 (Cost Components)	Stacked M2	0.969	17.246	4,178.170	64.639	0.116	81.4
Method 3 (Routing Structure)	Stacked M3	0.984	13.234	2,088.685	45.702	0.097	86.1

5. Limitations

The research approach is a single case study meaning that all data come from one company. The plastic injection molding parts, the subject of this project, are components of the company's products, not their final ones. An additional limitation can be found in the product development process. The development cycle applied is rather strict and involves several steps with related milestones. After each milestone, a management decision about the further steps is made. The same is true for the applied cost calculation schema. The development process and cost calculation are closely related. It was a management requirement that the ML approaches have to get along with the existing data.

Furthermore, employees from different departments must be able to use the system in their daily routines. A further limitation is the lack of a large sample size for training an accurate NN. It is well known that NNs typically require an extensive data set for precise predictions. All these factors limit the comparability of the results with other companies producing plastic injection molding parts.

6. Conclusion and future work

The primary objective of this study was to determine the most accurate method for estimating the full cost of plastic injection molding parts using machine learning algorithms. To achieve this, various models were trained using three different cost calculation methods, based on a dataset comprising 4,036 samples from 14 products. The most fine-grained cost calculation model, Method 3, which involved predicting the routing structure data, yielded the most accurate cost prediction, with a Mean Absolute Percentage Error (MAPE) of 0.097 and a Mean Absolute Error (MAE) of 13.234. However, this approach required a significant amount of effort due to the need to train, tune, and evaluate multiple models.

To verify whether Neural Networks (NNs) outperformed tree-based ML algorithms in predicting full cost (Method 1), various NN architectures with varying hidden layers were trained and tested. The best architecture, with ten hidden layers and a MAPE of 0.708, led to an underperforming result compared to tree-based approaches. Due to their computational efficiency and superior explainability, subsequent evaluations focused solely on tree-based models. However, increasing the training dataset size could improve NN performance.

The results of this study indicate that the most fine-grained cost calculation model was able to predict the full cost of ten out of 14 products within a ±20% deviation. Two additional products were slightly outside this range, and two products had a 100% deviation. This suggests that supervised ML approaches can support cost estimation for new products in the early stages of the product development cycle. However, such systems are not yet mature enough to be relied upon entirely.

Further research is needed to optimize the cost estimation process in several areas, including expanding the sample size and data basis, identifying reasons for the outliers, exploring other AI technologies and approaches, and incorporating 3D CAD data and numerical data from injection molding simulations in the model training process.

Conflict of interest

The authors Ploder, Bernsteiner, and Nocker certify that they have no affiliations with or involvement in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this manuscript.

Mr. Klocker is an employee at the commissioning company that participated in this research project.

References

[1] Park J-H, Seo K-K, Wallace D, Lee K-I. Approximate product life cycle costing method for the conceptual product design. *CIRP Annals*. 2002; 51: 421-424. Available from: doi: 10.1016/S0007-8506(07)61551-0.

[2] Drury CM. *Management and Cost Accounting*. Springer; 2013.

[3] Mitchell TM. Introduction. In: *Machine Learning*. New York, NY, USA: McGraw-Hill; 1997. p. 1-19.
[4] Murphy K. Introduction. In: *Machine Learning: A Probabilistic Perspective*. Massachusetts, USA: MIT Press; 2012. p. 1-25.
[5] Plinke W, Utzig BP. Kalkulation (Kostenträgerstückrechnung). In: Plinke W, Utzig BP (eds.) *Industrielle Kostenrechnung: Eine Einführung*. Berlin, Heidelberg: Springer; 2020. p. 101-135.
[6] Molcho G, Cristal A, Shpitalni M. Part cost estimation at early design phase. *CIRP Annals*. 2014; 63: 153-156. Available from: doi: 10.1016/j.cirp.2014.03.107.
[7] Ehrlenspiel K, Kiewert A, Lindemann U, Mörtl M. Kostenfrüherkennung bei der Entwicklung - entwicklungsbegleitende Kalkulation. In: Ehrlenspiel K, Kiewert A, Lindemann U, Mörtl M (eds.) *Kostengünstig Entwickeln und Konstruieren: Kostenmanagement bei der integrierten Produktentwicklung*. Berlin, Heidelberg: Springer; 2020. p. 477-523.
[8] Niazi A, Dai J, Balabani S, Seneviratne L. Product cost estimation: Technique classification and methodology review. *Journal of Manufacturing Science and Engineering*. 2006; 128(2); 563-575. Available from: doi: 10.1115/1.2137750.
[9] Ben-Arieh D, Qian L. Activity-based cost management for design and development stage. *International Journal of Production Economics*. 2003; 83: 169-183. Available from: doi: 10.1016/S0925-5273(02)00323-7.
[10] Lange F. Wirtschaftlichkeitsbetrachtung. In: Lange F (ed.) *Prozessgerechte Topologieoptimierung für die Additive Fertigung*. Light Engineering für die Praxis. Berlin, Heidelberg: Springer; 2021. p. 97-104.
[11] Plinke W, Utzig BP. Der Kostenbegriff und seine Unterbegriffe. In: Plinke W, Utzig BP (eds.) *Industrielle Kostenrechnung: Eine Einführung*. Berlin, Heidelberg: Springer; 2020. p. 23-39.
[12] Rosato DV, Rosato MG. *Injection Molding Handbook*. Springer Science & Business Media; 2012.
[13] Fagade AA, Kazmer D. Early cost estimation for injection molded components. *Journal of Injection Molding Technology*. 2000; 4: 97-106.
[14] Plinke W, Utzig BP. Systeme der industriellen Kosten- und Leistungsrechnung (IKR). In: Plinke W, Utzig BP (eds.) *Industrielle Kostenrechnung: Eine Einführung*. Berlin, Heidelberg: Springer; 2020. p. 51-57.
[15] Ning F, Shi Y, Cai M, Xu W, Zhang X. Manufacturing cost estimation based on a deep-learning method. *Journal of Manufacturing Systems.* 2020; 54: 186-195. Available from: doi: 10.1016/j.jmsy.2019.12.005.
[16] Bodendorf F, Franke J. A machine learning approach to estimate product costs in the early product design phase: a use case from the automotive industry. *Procedia CIRP*. 2021; 100: 643-648. Available from: doi: 10.1016/j.procir.2021.05.137.
[17] Matel E, Vahdatikhaki F, Hosseinyalamdary S, Evers T, Voordijk H. An artificial neural network approach for cost estimation of engineering services. *International Journal of Construction Management.* 2019; 22(7): 1-14. Available from: doi: 10.1080/15623599.2019.1692400.
[18] Verlinden B, Duflou JR, Collin P, Cattrysse D. Cost estimation for sheet metal parts using multiple regression and artificial neural networks: A case study. *International Journal of Production Economics.* 2008; 111: 484-492. Available from: doi: 10.1016/j.ijpe.2007.02.004.
[19] Wang HS, Wang YN, Wang YC. Cost estimation of plastic injection molding parts through integration of PSO and BP neural network. *Expert Systems with Applications*. 2013; 40: 418-428. Available from: doi: 10.1016/j.eswa.2012.01.166.
[20] Shi Y, Eberhart R. A modified particle swarm optimizer. In: *1998 IEEE International Conference on Evolutionary Computation Proceedings*. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360); 1998. p. 69-73.
[21] Bhavnani HD. Developing an emotion recognition tool for tweets. *Cloud Computing and Data Science.* 2022; 4(1): 49-59.
[22] Murat Günaydın H, Zeynep Doğan S. A neural network approach for early cost estimation of structural systems of buildings. *International Journal of Project Management.* 2004; 22: 595-602. Available from: doi: 10.1016/j.ijproman.2004.04.002.
[23] Brede S, Küster B, Stonis M, Mücke M, Overmeyer L. Part based mold quotation with methods of machine learning. In: Nyhuis P, Herberger D, Hübner M (eds.) *Proceedings of the Conference on Production Systems and Logistics: CPSL 2020*. 2020. Available from: doi: 10.15488/9664.
[24] Yoo S, Kang N. Explainable artificial intelligence for manufacturing cost estimation and machining feature visualization. *Expert Systems with Applications*. 2021; 183(7): 115430. Available from: doi: 10.1016/j.eswa.2021.115430.
[25] Loyer J-L, Henriques E, Fontul M, Wiseall S. Comparison of machine learning methods applied to the estimation

of manufacturing cost of jet engine components. *International Journal of Production Economics*. 2016; 178: 109-119. Available from: doi: 10.1016/j.ijpe.2016.05.006.
[26] Aurelien G. *Hands-On Machine learning with Scikit-Learn, Keras & Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd ed. O'Reilly Media; 2019.
[27] Razi MA, Athappilly K. A comparative predictive analysis of neural networks (NNs), nonlinear regression and classification and regression tree (CART) models. *Expert Systems with Applications*. 2005; 29: 65-74. Available from: doi: 10.1016/j.eswa.2005.01.006.
[28] Markham IS, Mathieu RG, Wray BA. Kanban setting through artificial intelligence: a comparative study of artificial neural networks and decision trees. *Integrated Manufacturing Systems.* 2000; 11: 239-246.
[29] Loh W-Y. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery.* 2011; 1: 14-23.
[30] Moreira J, Soares C, Jorge A, Sousa J. Ensemble approaches for regression: A survey. *ACM Computing Surveys*. 2012; 45(1): 1-40.
[31] Bergstra J, Yamins D, Cox D. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In: van der Walt S, Millman J, Huff K (eds.) *Proceedings of the 12th Python in Science Conference*. 2013. p. 13-19.
[32] Jin H, Song Q, Hu X. *Auto-Keras: An Efficient Neural Architecture Search System*. 2019.
[33] Myttenaere A de, Golden B, Le Grand B, Rossi F. Mean absolute percentage error for regression models. *Neurocomputing*. 2016; 192: 38-48. Available from: doi: 10.1016/j.neucom.2015.12.114.
[34] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*. 2020; 128: 336-359.
[35] Bach S, Binder A, Montavon G, Klauschen F, Müller K-R, Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one.* 2015; 10: e0130140. Available from: doi: 10.1371/journal.pone.0130140.